MODELING AND ANALYSIS OF INFORMATION SYSTEMS, VOL. 27, NO. 4, 2020
journal homepage: www.mais-journal.ru

THEORY OF COMPUTING

On the Modeling of Sequential Reactive Systems by Means of Real Time Automata

E. M. Vinarskii[1], V. A. Zakharov[1]

DOI: 10.18255/1818-1015-2020-4-396-411

[1]Lomonosov Moscow State University, GSP-1, Leninskie Gory, Moscow, 119991, Russia.

MSC2020: 68Q45
Research article
Full text in Russian

Received November 16, 2020
After revision December 1, 2020
Accepted December 16, 2020

Sequential reactive systems include hardware devices and software programs which operate in continuous interaction with the external environment, from which they receive streams of input signals (data, commands) and in response to them form streams of output signals. Systems of this type include controllers, network switches, program interpreters, system drivers. The behavior of some reactive systems is determined not only by the sequence of values of input signals, but also by the time of their arrival at the inputs of the system and the delays in computing the output signals. These aspects of reactive system computations are taken into account by real-time models of computation which include, in particular, real-time finite state machines (TFSMs). However, in most works where this class of real-time automata is studied a simple variant of TFSM semantics is used: the transduction relation computed by a TFSM is defined so that the elements of an output stream, regardless of their timestamps, follow in the same order as the corresponding elements of the input stream. This straightforward approach makes the model easier to analyze and manipulate, but it misses many important features of real-time computation. In this paper we study a more realistic semantics of TFSMs and show how to represent it by means of Labeled Transition Systems. The use of the new TFSM model also requires new approaches to the solution of verification problems in the framework of this model. For this purpose, we propose an alternative definition of TFSM computations by means of Labeled Transition Systems and show that the two definitions of semantics for the considered class of real-time finite state machines are in good agreement with each other. The use of TFSM semantics based on Labeled Transition Systems opens up the possibility of adapting well known real-time model checking techniques to the verification of sequential reactive systems.

Keywords: reactive system; finite state machine; verification; security property; labeled transition system; simulation

INFORMATION ABOUT THE AUTHORS

Evgeney Maximovich Vinarskii	orcid.org/0000-0002-7328-0942. E-mail: vinevg2015@gmail.com –
Vladimir Anatolyevich Zakharov correspondence author	orcid.org/0000-0002-3794-9565. E-mail: zakh@cs.msu.ru doctor of sciences, professor.

Funding: The reported study was funded by RFBR, project number 18-01-00854.

For citation: E. M. Vinarskii and V. A. Zakharov, “On the Modeling of Sequential Reactive Systems by Means of Real Time Automata”, *Modeling and analysis of information systems*, vol. 27, no. 4, pp. 396-411, 2020.

МОДЕЛИРОВАНИЕ И АНАЛИЗ ИНФОРМАЦИОННЫХ СИСТЕМ, ТОМ 27, № 4, 2020
сайт журнала: www.mais-journal.ru

THEORY OF COMPUTING

О моделировании последовательных реагирующих систем при помощи автоматов, работающих в реальном времени

Е. М. Винарский[1], В. А. Захаров[1]

DOI: 10.18255/1818-1015-2020-4-396-411

[1]Московский государственный университет имени М. В. Ломоносова, Ленинские горы, 1, г. Москва, 119991 Россия.

УДК 519.7
Научная статья
Полный текст на русском языке

Получена 16 ноября 2020 г.
После доработки 1 декабря 2020 г.
Принята к публикации 16 декабря 2020 г.

Последовательные реагирующие системы включают в себя устройства и программы, вычисления которых состоят в непрерывном взаимодействии с внешней средой, от которой они получают потоки входных сигналов (данных, команд) и в ответ на них формируют потоки выходных сигналов. К системам такого вида относятся контроллеры, сетевые коммутаторы, интерпретаторы программ, системные драйверы. Поведение некоторых реагирующих систем определяется не только последовательностью значений входных сигналов, но также временем их поступления на вход системы и задержками вычисления выходных сигналов. Эти особенности вычисления реагирующих систем хорошо учитываются вычислительными моделями реального времени, к числу которых относятся, в частности, конечные автоматы-преобразователи реального времени (Timed Finite State Machines, TFSMs). Однако в большинстве работ, посвященных изучению этой модели вычислений, используется упрощенная семантика TFSMs: элементы выходного потока, независимо от сопутствующих пометок времени, располагаются в том же порядке, в котором следуют соответствующие им элементы входного потока. Такое упрощение семантики делает модель менее адекватной для многих приложений, но зато облегчает решение задач анализа и преобразования таких автоматов. В данной статье мы изучаем модель TFSM с более реалистичной семантикой. Переход к новой модели TFSM требует и новых подходов к решению задачи верификации автоматов в этой модели. Для этой цели мы предлагаем альтернативное определение вычислений TFSM с использованием размеченных систем переходов и показываем, что два определения семантики для рассматриваемого класса автоматов реального времени хорошо взаимосвязаны друг с другом. Использование семантики TFSM, основанной на размеченных системах переходов, открывает возможность применения ранее известных методов верификации систем вычислений реального времени для анализа поведения последовательных реагирующих систем.

Ключевые слова: реагирующая система; конечный автомат; верификация; свойство безопасности; размеченная система переходов; симуляция

ИНФОРМАЦИЯ ОБ АВТОРАХ

Евгений Максимович Винарский | orcid.org/0000-0002-7328-0942. E-mail: vinevg2015@gmail.com
–
Владимир Анатольевич Захаров | orcid.org/0000-0002-3794-9565. E-mail: zakh@cs.msu.ru
автор для корреспонденции | доктор физико-математических наук, профессор.

Финансирование: Работа выполнена при финансовой поддержке гранта РФФИ N 18-01-00854.

Для цитирования: E. M. Vinarskii and V. A. Zakharov, “On the Modeling of Sequential Reactive Systems by Means of Real Time Automata”, *Modeling and analysis of information systems*, vol. 27, no. 4, pp. 396-411, 2020.

Введение

Последовательные реагирующие системы включают в себя устройства и программы, вычисления которых состоят в непрерывном взаимодействии с внешней средой; они получают извне потоки входных данных (запросов, управляющих сигналов, команд и др.) и в ответ формируют потоки выходных данных (откликов). К системам такого вида относятся контроллеры, адаптеры, сетевые коммутаторы, интерпретаторы программ, системные драйверы. Одной из самых простых математических моделей последовательных реагирующих систем являются конечные автоматы-преобразователи (Finite State Machines, FSMs), позволяющие адекватно описывать поведение синхронных систем. Вычисления этих систем разделены на такты, на каждом такте на вход системы поступает запрос и на выходе формируется соответствующий отклик, зависящий от поступившего запроса и текущего состояния системы. Модель FSM была предложена еще в начале развития теории автоматов (см. [1]). Она хорошо изучена, и на основе этой модели разработаны эффективные алгоритмы синтеза и анализа поведения синхронных управляющих систем с памятью (см., например, [2]).

Вместе с тем, во многих приложениях приходится иметь дело с асинхронными реагирующими системами, вычисления которых протекают не столь регулярно, как в синхронных системах. Поведение асинхронных систем определяется не только последовательностью значений входных сигналов, но также временем их поступления на входы системы, задержками вычисления выходных сигналов, продолжительностью пребывания системы в том или ином состоянии управления и др. Чтобы использовать конечные автоматы-преобразователи для описания поведения асинхронных реагирующих систем, необходимо снабдить автоматы дополнительными возможностями для моделирования течения физического времени.

Существуют различные подходы к расширению концепции FSM в зависимости от используемых для этой цели средств. Вероятно, наиболее развитым расширением концепции конечных автоматов, работающих в реальном времени, является модель временны́х автоматов (Timed Automata, TA), впервые предложенная в статье [3]. Каждый TA может иметь несколько специальных вещественных переменных — таймеров. В зависимости от значения таймеров определяются условия пребывания TA в состояниях управления — инварианты состояний TA, — а также условия осуществления переходов из одних состояний управления в другие состояния — предохранители переходов TA. Ход физического времени моделируется увеличением показаний таймеров на одну и ту же величину (вещественное число). Автомату также предоставлена возможность сброса показаний некоторых таймеров при совершении переходов. В рамках модели TA можно описывать многие особенности вычислений, протекающих в реальном времени, и поэтому эта модель нашла широкое применение для решения задач проектирования и верификации вычислительных систем реального времени. В работах [3—5] было показано, что многие алгебраические свойства классических конечных автоматов присущи и временны́м автоматам. Однако было обнаружено также (см. [6, 7]), что большинство задач анализа поведения временны́х автоматов оказываются вычислительно трудными. Это объясняется, в первую очередь, тем, что TA предоставлена большая свобода в обращении с таймерами.

Сложные манипуляции с таймерами могут быть неизбежными, если возникает необходимость синхронизировать несколько событий по ходу вычисления. Однако во многих практически важных случаях поведение систем управления не так сложно, и реакция системы определяется исключительно текущим состоянием управления и параметрами входных данных. Поэтому, чтобы избежать трудностей, возникающих при работе с такой сложной вычислительной моделью, как TA, авторы [8, 9] предложили снабдить FSM всего лишь одним таймером, показания которого сбрасываются при каждом переходе. Расширения модели автоматов-преобразователей такого вида получили название TFSM (Timed Finite State Machine). Были выделены три семейства TFSM в зависимости

от допустимого *modus operandi* с таймерами: (i) TFSM с синхронизированными предохранителями, которые запускают переходы только тогда, когда временна́я метка входа удовлетворяет соответствующему временно́му ограничению [8]; (ii) TFSM с тайм-аутами, которые заставляют машину выполнять переход по истечении предписанного времени ожидания входного сигнала [9]; (iii) TFSM с временны́ми предохранителями и тайм-аутами [10]. В статье [10] было установлено, что эти три семейства TFSM обладают сходными вычислительными возможностями — машины одного семейства способны моделировать вычисления машин другого семейства. Кроме того, в [10] был предложен метод абстракций TFSM — особая модификация метода регионов [3], — при помощи которой оказалось возможным эффективно сводить многие задачи синтеза и анализа TFSM к аналогичным задачам для классических конечных автоматов (см. [10—13]).

Для внешнего наблюдателя поведение реагирующей системы проявляется в преобразовании потока входных данных (запросов) с пометками времени их подачи на вход системы в поток выходных данных (откликов), которые также помечены временем их появления на выходе системы. Естественно ожидать, что элементы выходного потока следуют в порядке возрастания пометок времени. Однако отличительная особенность всех тех классов TFSM, которые были введены и исследованы в статьях [8—12], состоит в том, что в выходном потоке отклики, независимо от сопутствующих пометок времени, располагаются в том же порядке, в котором следуют соответствующие им запросы. Такое упрощение операционной семантики TFSM облегчает решение задач анализа и синтеза, но, вместе с тем, делает указанные модели TFSM непригодными для некоторых приложений.

Обратимся, в частности, к примеру контроллера программно-конфигурируемой сети (ПКС) [14]. Программно-конфигурируемая сеть состоит из множества взаимосвязанных коммутаторов, находящихся под управлением контроллера. Заголовки пакетов, поступающих во входной буфер коммутатора, сопоставляются с таблицей коммутации, чтобы выбрать подходящее правило, которое либо пересылает пакет в какой-либо выходной буфер коммутатора, либо отбрасывает его. Если для поступившего пакета в таблице коммутации нет подходящего правила, то коммутатор обращается к контроллеру. Контроллер ПКС можно рассматривать как последовательную реагирующую систему, которая получает на вход запросы на предоставление новых правил коммутации и формирует в ответ, руководствуясь некоторой политикой маршрутизации пакетов, управляющие команды, модифицирующие таблицы коммутации. Эти команды доставляются коммутаторам по каналам сети с некоторой задержкой, и поэтому порядок выполнения команд, вносящих изменения в таблицы коммутации, может отличаться от того порядка, в котором они были сформированы контроллером ПКС. Возможна такая ситуация, описанная в [15], когда контроллер получает пару запросов R_1 и R_2 на добавление новых правил от коммутаторов сети C_1 и C_2 в моменты времени t_1 и t_2 соответственно, где $t_1 < t_2$. В ответ на каждый из этих запросов R_i контроллер формирует команду A_i добавления в таблицы коммутации новых правил, и эта команда может быть доставлена коммутатору C_i и выполнена им с некоторой задержкой τ_i. Если $t_2 + \tau_2 < t_1 + \tau_1$, то в течение некоторого времени ПКС может пребывать в ошибочной конфигурации, в которой коммутатор C_1 обрабатывает пакеты по старым правилам, а коммутатор C_2 уже применяет к пакетам новые правила. Однако достижимость этой ошибочной конфигурации ПКС не может быть обнаружена при помощи модели TFSM, использующей операционную семантику, которая была введена в работах [8, 9], поскольку эта модель будет показывать, что входная последовательность запросов $(R_1, t_1), (R_2, t_2)$ преобразуется в выходную последовательность $(A_1, t_1 + \tau_1), (A_2, t_2 + \tau_2)$, в которой порядок следования команд отличается от порядка их выполнения.

Чтобы справиться с этим недостатком обычных TFSM и сделать их поведение более “реалистичным”, в статье [16] были внесены некоторые изменения в семантику TFSM. Новая модель TFSM отражает важную особенность вычислений реагирующих систем: порядок следования откликов в выходном потоке определяется временем их генерации, а не порядком следования соответствующих им

запросов во входном потоке: если запрос b поступает на вход машины после сигнала a, то возможно, что ответ на запрос a будет следовать за ответом на запрос b. Усовершенствованная операционная семантика TFSM позволяет явно обнаруживать описанное выше ошибочное поведение контроллера ПКС. Вместе с тем, новая семантика TFSM привносит дополнительные трудности в анализ поведения автоматов. Например, как показано в статье [16], даже такое простое свойство как свойство строгой детерминированности вычислений оказывается нелокальным в новой модели TFSM, и его проверка является вычислительно трудной задачей. Существенные трудности возникают также с проверкой свойств безопасности вычислений, поскольку вычисления TFSM в новой операционной семантике не обладают свойством монотонности: если входная последовательность α' продолжает входную последовательность α, то нельзя гарантировать, что соответствующая ей выходная последовательность β будет являться началом более протяженной последовательности β', которая является откликом автомата на входную последовательность α'. Отсутствие свойства монотонности значительно усложняет задачу верификации TFSM, и, в частности, препятствует применению метода абстракций, который столь успешно использовался в работах [10—13] для верификации, минимизации и тестирования TFSM, работающих в более простой операционной семантике.

Для преодоления этой принципиальной трудности, возникающей при решении задач верификации TFSM с модифицированной операционной семантикой, мы предлагаем в этой статье альтернативный подход к определению семантики TFSM, который, по нашему мнению, лучше подходит для традиционных методов проверки. Для определения вычислений TFSM мы воспользуемся размеченными системами переходов (Labeled Transition Systems, LTS), которые строятся на основе концепции конфигурации TFSM. Новая операционная семантика TFSM позволяет увидеть явную взаимосвязь этой модели вычислений с моделью ТА [3]. Однако пространство состояний в системе переходов $LTS(\mathcal{T})$, соответствующей TFSM $\mathcal{T}$, оказывается несчетным. Наша долгосрочная цель — разработать алгоритм, который для каждой TFSM $\mathcal{T}$ строит конечную $LTS_{fin}(\mathcal{T})$, которая симуляционно эквивалентна системе переходов $LTS(\mathcal{T})$ и, таким образом, может использоваться для проверки свойств вычислений TFSM $\mathcal{T}$. В этой статье мы представляем некоторые предварительные результаты наших исследований.

В разделах 1 и 2 мы вводим понятие TFSM с модифицированной операционной семантикой и показываем на простом примере, как можно использовать новую модель TFSM для адекватного описания поведения контроллеров ПКС. В разделе 3 мы определяем семантику для рассматриваемых TFSM на основе размеченных систем переходов, изучаем ее взаимосвязь с теоретико-автоматной семантикой и представляем некоторые результаты, которые устанавливают взаимосвязь рассматриваемой модели TFSM и модели временны́х автоматов [3]. В заключительном разделе мы обсуждаем некоторые новые задачи, которые возникают на основании полученных результатов.

1. Основные определения и обозначения

В этом разделе приводятся определения основных понятий, связанных с моделью вычислений конечных автоматов преобразователей, работающих в реальном времени (TFSM), и определяется операционная семантика TFSM, основанная на понятии вычисления автомата.

Рассмотрим два непустых конечных алфавита A (*входной алфавит*) и B (*выходной алфавит*). Элементы первого алфавита обозначают запросы, поступающие на вход реагирующей системы, а элементы второго алфавита — это отклики, которые реагирующая система вырабатывает в ответ на запросы. Конечную последовательность символов входного алфавита $\alpha = a_1, a_2, \ldots, a_n \in A^*$ будем называть *входным словом*, а конечную последовательность символов выходного алфавита $\beta = b_1, b_2, \ldots, b_n \in B^*$ – *выходным словом*.

Реакция конечного асинхронного автомата-преобразователя на входной символ a зависит от следующих факторов:

- текущего состояния автомата,

- времени поступления входного символа a,
- времени, которое требуется автомату для обработки этого входного символа (запроса) и формирования выходного символа (отклика).

Для моделирования времени будем использовать вещественную переменную t, принимающую значения в множестве неотрицательных вещественных чисел $\mathbb{R}_0^+$. Значение этой переменной будет обозначать время поступления входных символов и время выдачи выходных символов. Всякую возрастающую последовательность неотрицательных вещественных чисел $\tau = t_1, t_2, \ldots, t_n, \ldots$ будем называть *временно́й последовательностью*.

Допустимым интервалом (задержкой) является всякий интервал вида $g = \langle u, v \rangle$, где $0 < u \le v$, $\langle$ — один из ограничителей (или [, $\rangle$ — один из ограничителей) или]. Допустимые интервалы и задержки являются параметрами переходов TFSM. Допустимый интервал перехода TFSM из состояния s — это промежуток времени, в течение которого этот переход является активными, начиная с того момента, когда автомат оказывается в состоянии s. Задержка — это промежуток времени, в течение которого TFSM формирует отклик на входной запрос, начиная с момента поступления этого запроса. *Левым концом* (*правым концом*) допустимого интервала (задержки) $\langle u, v \rangle$ назовем число u (соответственно, число v). Задержку вида $[d, d]$ с одинаковыми концами будем называть *точечной* и использовать для ее обозначения запись d вместо $[d, d]$).

Пусть $w = x_1, x_2, \ldots x_n$ — входное (выходное) слово и $\tau = t_1, t_2, \ldots, t_n$ — временная последовательность. Тогда всякую пару (x_i, t_i), где $1 \le i \le n$, назовем *временны́м символом*, а последовательность временны́х символов

$$\alpha = (x_1, t_1), (x_2, t_2), \ldots, (x_n, t_n)$$

назовём *входным (выходным) временны́м словом* и обозначим записью (w, τ). Последнее значение t_n в последовательности τ обозначим записью $t(\alpha)$.

Пусть задано некоторое множество допустимых интервалов G и некоторое множество задержек D. Тогда *конечным автоматом-преобразователем, работающим в реальном времени* (сокращенно, TFSM, Timed Finite State Machine) над алфавитами A и B, множеством допустимых интервалов G и множеством задержек D, назовём набор $\mathcal{T} = (S, \rho, s_0)$, в котором:

- S — конечное *множество состояний управления*;
- $\rho \subseteq (S \times A \times G \times B \times D \times S)$ — конечное множество *трансдукций*;
- $s_0 \in S$ — *начальное состояние управления*.

Всякий кортеж $(s, a, g, b, d, s') \in \rho$ называется *трансдукцией* в $\mathcal{T}$. Трансдукция является элементарным действием TFSM $\mathcal{T}$. Это действие выполняется, если спустя некоторое время $t \in g$ с тех пор, как автомат оказался в состоянии управления s, на его вход поступает символ a; тогда автомат немедленно переходит в состояние s' и выдает выходной символ b спустя некоторое время $\delta \in d$ после поступления входного символа a. Таким образом, вычисление TFSM $\mathcal{T}$ состоит в преобразовании входного временно́го слова в выходное временно́е слово. Более формально это преобразование определяется так.

Вычислением TFSM $\mathcal{T} = (S, \rho, s_0)$ на входном временно́м слове

$$\alpha = (a_1, t_1), (a_2, t_2), \ldots, (a_n, t_n)$$

называется такая конечная последовательность

$$r = s_0 \xrightarrow{(a_1,t_1)/(b_1,\tau_1)} s_1 \xrightarrow{(a_2,t_2)/(b_2,\tau_2)} \ldots \xrightarrow{(a_n,t_n)/(b_n,\tau_n)} s_n,$$

что для каждого $i \in \{1, \ldots, n\}$ существует трансдукция $(s_{i-1}, a_i, g_i, b_i, d_i, s_i) \in \rho$, удовлетворяющая следующим двум условиям:

1. $t_i - t_{i-1} \in g_i$ (полагая при этом $t_0 = 0$);

2. $\tau_i = t_i + \delta$, где $\delta \in d_i$.

Будем говорить, что вычисление r TFSM $\mathcal{T}$ *преобразует* входное временно́е слово α в такое выходное временно́е слово $\beta = (b_{j_1}, \tau_{j_1}), (b_{j_2}, \tau_{j_2}), \ldots, (b_{j_n}, \tau_{j_n})$, которое является перестановкой последовательности пар $\gamma = (b_1, \tau_1), (b_2, \tau_2), \ldots, (b_n, \tau_n)$. Данное определение вычисления TFSM на входном временно́м слове отличается от аналогичного определения, используемого в статьях [10—13], тем, что в указанных работах результатом вычисления TFSM считается не временно́е слово β, а последовательность выходных временны́х символов γ, которая не обязана быть упорядоченной по возрастанию пометок времени $\tau_1, \tau_2, \ldots$.

Основной функциональной характеристикой TFSM $\mathcal{T}$ является *отношение временно́й трансдукции* $TR(\mathcal{T})$, которое представляет собой множество всех таких пар (α, β), где:

- α – входное временно́е слово;
- β – выходное временно́е слово, которое получено в результате преобразования входного временно́го слова α некоторым вычислением r TFSM $\mathcal{T}$.

Введенное здесь отношение временно́й трансдукции отличается от обычного для автоматов-преобразователей отношения словарной трансдукции, в котором входное временно́е слово α преобразуется в указанную выше последовательность выходных временны́х символов γ, а не в соответствующее этой последовательности выходное временно́е слово β.

Если все задержки в множестве D являются точечными, то TFSM $\mathcal{T}$ будем называть *TFSM с точечными задержками*. В данной работе мы рассматриваем только TFSM с точечными задержками. На рис. 1 приведен пример TFSM с точечными задержками. Этот автомат имеет вычисление

$$r = s_0 \xrightarrow{(a_1,1.5)/(b_1,4.5)} s_1 \xrightarrow{(a_2,2.7)/(b_2,3.7)} s_2 \xrightarrow{(a_3,5)/(b_3,7)} s_3,$$

которое преобразует входное временно́е слово $\alpha = (a_1, 1.5), (a_2, 2.7), (a_3, 5)$ в выходное временно́е слово $\beta = (b_2, 3.7), (b_1, 4.5), (b_3, 7)$.

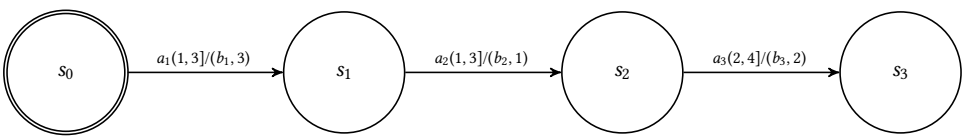

Fig. 1. TFSM with point delays

Рис. 1. TFSM с точечными задержками

2. Свойство безопасности для систем реального времени

Ошибочное поведение информационной системы может проявляться по-разному. В самом простом случае вычисление системы считается ошибочным, если оно достигает некоторого недопустимого состояния; тогда факт ошибки может быть зафиксирован, и никакие последующие действия системы не могут отменить этого факта. Поведение системы, которое не содержит такого рода ошибок, обычно называют безопасным. Для реагирующей системы недопустимым может считаться такое состояние вычисления, при достижении которого обнаруживается, что реакция системы, представленная последовательностью откликов, полученных в ответ на последовательность запросов, поданных на вход системы, не удовлетворяет некоторым заданным ограничениям. Если реагирующая система является синхронной, то эта реакция может быть определена непосредственно по заданной конечной последовательности запросов. Вычисления любой синхронной реагирующей системы устроены так, что ее реакция на последующие запросы не может изменить результаты выполнения тех ее действий, которые были инициированы предшествующими запросами. Благодаря этой особенности, требования безопасности вычислений синхронных реагирующих систем можно определять в терминах свойств отношений трансдукции, вычисляемых этими системами.

Однако поведение асинхронных реагирующих систем, работающих в реальном времени, значительно сложнее. Поступление запроса на вход асинхронной реагирующей системы и появление соответствующего этому запросу отклика на выходе системы может отделять некоторый промежуток времени. В течение этого промежутка времени система может отреагировать на последующие запросы и, тем самым, изменить результат выполнения ранее инициированных ею откликов. Образным примером этого эффекта может служить поведение человека, который, уронив вначале на пол стеклянный шар, успевает затем положить на место падения предмета мягкий ковер, и шар остается целым. В данном случае между принятием решения о выполнении действия по запросу “уронить шар на пол” и завершением осуществления этого действия реагирующая система успевает выполнить действие по следующему запросу “переместить ковер”, и, тем самым, предотвратить нарушение требования безопасности.

В этом разделе мы покажем, какие трудности возникают при проверке свойств безопасности для рассматриваемой модели TFSM, в которой семантика автоматов определяется отношением временно́й трансдукции.

Вычисление реагирующей системы обычно можно охарактеризовать бесконечной последовательностью событий (ω-словом). Тогда согласно определению [17] множество ω-слов P_{safe} называется *свойством безопасности вычислений*, если каждое ω-слово $\alpha \notin P_{safe}$ имеет такой конечный префикс w, что для любого ω-слова α', имеющего тот же префикс w, также выполняется отношение $\alpha' \notin P_{safe}$. Согласно этому определению если вычисление α не является безопасным, то ошибка может быть обнаружена после конечного числа шагов вычисления w, и ее уже нельзя будет устранить впоследствии. Вычисления реагирующих систем, работающих в реальном времени, представляются в виде бесконечных последовательностей событий с возрастающими пометками времени (временны́х ω-слов). Тогда свойство безопасности вычислений для реагирующих систем, работающих в реальном времени, можно определить так. Подмножество временны́х ω-слов $P_{safe} \subseteq (A \times \mathcal{R}^+)^\omega$ является *свойством безопасности вычислений в реальном времени*, если каждое временно́е ω-слово $\alpha \notin P_{safe}$ обладает конечным префиксом w таким, что для любого временно́го ω-слова α', имеющего то же временно́е слово w в качестве префикса, также выполняется отношение $\alpha' \notin P_{safe}$. Это определение можно использовать и в случае TFSM, заменив временны́е ω-слова α парами временны́х ω-слов (α, β), где β — результат преобразования входного временно́го ω-слова α.

В основу алгоритмов верификации свойств безопасности положены следующие соображения. Согласно приведенному выше определению свойства безопасности P_{safe} существует множество конечных слов L_{safe}, которое для любого ω-слова α и конечного слова β удовлетворяет условию: $\alpha \notin P_{safe}$ тогда и только тогда, когда α имеет такой префикс β, что $\beta \in L_{safe}$. Тогда для обнаружения того, что некоторое вычисление реагирующей системы нарушает требование безопасности P_{safe}, необходимо и достаточно убедиться, что какое-либо начало этого вычисления характеризуется некоторым словом $\beta \in L_{safe}$. Язык L_{safe} определяется свойством P_{safe} неоднозначно, для его описания можно использовать логические или алгебраические формулы, различные виды автоматов и это придает большую гибкость и разнообразие алгоритмам верификации свойств безопасности вычислений реагирующих систем. Однако при верификации моделей TFSM, описанных в предыдущем разделе, этот подход сталкивается с принципиальными трудностями.

Рассмотрим модель коммутатора ПКС C, который получает от контроллера ПКС команды на обновление правил в таблицах коммутации пакетов или запросы на предоставление информации об использовании правил коммутации и отправляет в ответ подтверждения о выполнении команд или запрашиваемую информацию. Может случиться так (см. [15]), что контроллер ПКС при получении входного временного слова $(R_1, t_1), (R_2, t_2)$, где $t_1 < t_2$, выдаёт входное временное слово $(A_2, \tau_1), (A_1, \tau_2)$, где $\tau_2 < \tau_1$, т.е. второе правило будет установлено раньше первого. При моделировании вычислений коммутатора ПКС необходимо учитывать, что выполнение команд занимает некоторое время, и

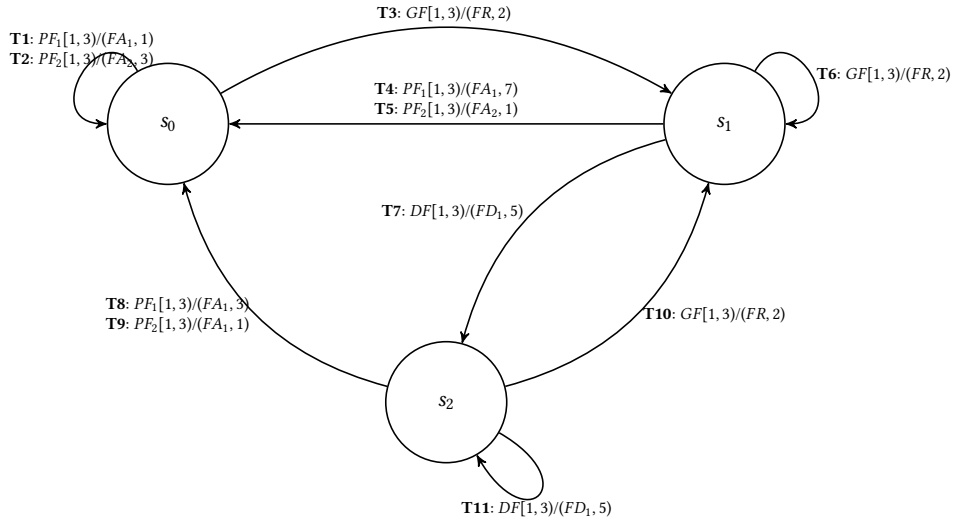

Fig. 2. Model of SDN switch $\mathcal{C}$

Рис. 2. Модель коммутатора ПКС $\mathcal{C}$

поэтому подтверждения о внесенных изменениях в таблицы коммутации доставляются контроллеру с некоторой задержкой. TFSM, описывающий поведение коммутатора $\mathcal{C}$, изображен на рис. 2. Отметим, что $\mathcal{C}$ обрабатывает следующие входные запросы:

- PF_i — добавить правило коммутации пакетов r_i в таблицу коммутатора;
- GF_i — предоставить информацию об использовании правила коммутации r_i;
- DF_i — удалить из таблицы правило коммутации пакетов r_i.

При этом коммутатор $\mathcal{C}$ отправляет следующие данные:

- FA_i — подтверждение о добавлении правила r_i в таблицу коммутации;
- FR_i — информация о частоте применения правила r_i;
- FD_i — подтверждение об удалении правила r_i из таблицы коммутации.

Помимо корректной обработки поступающих команд и запросов коммутатор также должен обеспечить своевременное предоставление информации контроллеру. Это означает, в частности, что подтверждения о выполненной модификации таблиц коммутации должны быть доставлены контроллеру в том же порядке, в каком поступали команды на проведение соответствующих модификаций. Как видно из приведенных выше определений, это требование является свойством безопасности вычислений коммутатора (P_{safe}). Рассмотрим вычисление предложенной модели коммутатора $\mathcal{C}$ на входном временно́м слове

$$\alpha = (GF_1, 1.5), (PF_1, 3.7), (PF_2, 5.5).$$

TFSM $\mathcal{C}$ преобразует входное слово α в выходное временно́е слово

$$\beta = (FR_1, 3.5), (FA_2, 8.5), (FA_1, 10.7).$$

Таким образом, реакция автомата явно показывает, что требование соответствия порядка следования сообщений-подтверждений порядку поступления команд нарушено и, следовательно, это вычисление нарушает требование безопасности. Тем не менее, входное слово α можно продолжить таким образом:

$$\alpha' = (GF_1, 1.5), (PF_1, 3.7), (PF_2, 5.5), (PF_1, 6.7),$$

что TSFM $\mathcal{C}$ преобразует его в выходное временное слово

$$\beta' = (FR_1, 3.5), (FA_1, 7.7), (FA_2, 8.5), (FA_1, 10.7),$$

которое удовлетворяет свойству безопасности P_{safe}. Данный пример показывает, что операционная семантика TFSM, которая определена в терминах преобразования временны́х слов, мало пригодна для эффективного решения на ее основе задачи верификации свойств безопасности вычислений TFSM. Это объясняется, прежде всего, тем, что отношение временно́й трансдукции не обладает свойством монотонности: если вычисление r_2 TFSM $\mathcal{T}$ является продолжением вычисления r_1, и при этом каждое из вычислений $r_i, i = 1, 2$, преобразует входное временно́е слово α_i в выходное временно́е слово и β_i, то слово α_1 является префиксом слова α_2, но слово β_1 не обязано быть префиксом слова β_2. Вычисления TFSM в той операционной семантике, которая использовалась в работах [10—13], обладают свойством монотонности, и поэтому для этих моделей реагирующих систем таких трудностей проверки свойств безопасности вычислений не возникает.

Для преодоления этого затруднения целесообразно определить альтернативную операционную семантику TFSM при помощи размеченных систем переходов (LTS), которая не имела бы отмеченного выше недостатка. Далее необходимо установить взаимосвязь между двумя введенными семантиками TFSM — теоретико-автоматной семантикой и семантикой, основанной на LTS, — и показать корректность альтернативной операционной семантики. После этого можно приступать к разработке алгоритмов верификации свойств безопасности вычислений TFSM, основываясь на размеченных системах переходов, соответствующих этим автоматам-преобразователям.

3. Размеченные системы переходов для TFSM

Операционная семантика временны́х автоматов-преобразователей, определенная в разделе 1, хорошо подходит для моделирования реагирующих систем, поскольку она отражает многие важные эффекты вычислений, выполняемых в реальном времени. Однако эта семантика препятствует применению традиционных методов и инструментов анализа поведения систем реального времени, поскольку в ней отсутствует понятие состояния вычислений как моментального снимка вычислительного процесса. Чтобы преодолеть этот недостаток, мы вводим понятие конфигурации, которая позволяет представлять поведение временного конечного трансдьюсера с помощью размеченной системы переходов на множествах таких конфигурациях.

3.1. Конфигурации размеченной системы переходов

Конфигурация TFSM $\mathcal{T}$ содержит информацию о текущем состоянии управления, показании таймера TFSM, а также о всех тех откликах (символах выходного алфавита), которые уже были сформированы ранее, но не были опубликованы, поскольку срок их задержки еще не истек. На множестве конфигураций TFSM $\mathcal{T}$ определены переходы трех типов: продвижение времени, получение входного символа (запроса) и выдача выходного символа (отклика).

Конфигурация TFSM $\mathcal{T} = (S, \rho, s_0)$ — это всякая тройка вида $q = \langle s, t, TC \rangle$, где

- $s \in S$ — состояние управления $\mathcal{T}$, которое будем обозначать записью $q.s$;
- $t \in \mathcal{R}_0^+$ — *входная метка времени*, которую будем обозначать записью $q.t$;
- $TC = \{(b_1, \tau_1), (b_2, \tau_2), \ldots, (b_n, \tau_n)\}$ — *выходной временно́й контекст* конфигурации q, который представляет собой множество выходных временны́х символов и обозначается записью $q.c$.

Выходная метка времени $q.t$ обозначает время, прошедшее с момента начала вычисления или приема автоматом последнего входного символа, а выходной временно́й контекст $q.c$ — это множество откликов TFSM $\mathcal{T}$, приготовленных для выдачи на выходе автомата, с указанием текущего времени их задержки. Множество всех конфигураций TFSM $\mathcal{T}$ обозначим записью $Q(\mathcal{T})$.

Размером конфигурации q назовем количество $|q.c|$ элементов в выходном временно́м контексте конфигурации q. Обозначим записью $q.T$ значение $min(\tau_1, \ldots, \tau_n)$, если $|q.c| = n > 0$; если $|q.c| = 0$, то будем полагать $q.T = \infty$. Значение $q.T$ — это время, спустя которое TFSM способна выдать очередной выходной символ без взаимодействия с внешней средой. Если $q.c = \{(b_1, \tau_1), \ldots, (b_n, \tau_n)\}$,

$b \in B$, и $\delta \le q.T$, то условимся обозначать записью $q + \delta$ конфигурацию

$$\langle q.s, t + \delta, \{(b_1, \tau_1 - \delta), \dots, (b_n, \tau_n - \delta)\}\rangle,$$

записью $in(q, b, d)$ — конфигурацию $\langle q.s, 0, q.c \cup \{(b, d)\}\rangle$, а записью $out(q, b)$ — конфигурацию $\langle q.s, q.t, q.c \setminus (b, 0)\rangle$.

При сопоставлении введенного здесь понятия конфигурации TFSM и понятия состояния временного автомата (ТА) [3] можно заметить, что параметры $t, \tau_1, \dots, \tau_n$ конфигурации q играют роль показаний таймеров состояния ТА, а совокупность параметров $s, b_1, \dots, b_n$ выполняет функции состояния управления ТА.

3.2. Размеченная система переходов TFSM

Размеченная система переходов $\mathcal{LTS}(\mathcal{T})$ TFSM $\mathcal{T} = (S, \rho, s_0)$ над входным алфавитом A, выходным алфавитом B, множествами допустимых интервалов G и задержек D — это тройка $(Q(\mathcal{T}), q_0, \rho_{\mathcal{T}})$, где

- $q_0 = \langle s_0, 0, \varnothing\rangle$ — *начальная конфигурация*,
- $\rho_{\mathcal{T}} \subseteq (Q \times \mathcal{R}^+ \times Q) \cup (Q \times A \times Q) \cup (Q \times B \times Q)$ — *отношение переходов*, определённое для каждой конфигурации $q = \langle s, t, \{(b_1, \tau_1), (b_2, \tau_2), \dots, (b_m, \tau_m)\}\rangle$ следующим образом:
  1) для любого числа $\delta \in \mathcal{R}^+$ такого, что $\delta < q.T$, в множестве $\rho_{\mathcal{T}}$ существует *переход продвижения времени* $q \xrightarrow{\delta} q + \delta$;
  2) для каждой трансдукции $(s, a, \langle u, v\rangle, c, d, s')$ TFSM $\mathcal{T}$ такой, что $t \in \langle u, v\rangle$, в множестве $\rho_{\mathcal{T}}$ существует *переход приема запроса* $q \xrightarrow{a} in(q, c, d)$;
  3) для каждой пары $(b, 0) \in q.c$ в множестве $\rho_{\mathcal{T}}$ существует *переход выдачи отклика* $q \xrightarrow{b} out(q, b)$.

Произвольную последовательность переходов $\pi = q_1 \xrightarrow{x_1} q_2 \xrightarrow{x_2} \cdots q_k \xrightarrow{x_k} q_{k+1}$, каждый из которых исходит из той же конфигурации, которой достиг предыдущий переход, назовем *путем* в $\mathcal{LTS}(\mathcal{T})$. Если конфигурация q_1, из которой исходит путь π, — это начальная конфигурация $\mathcal{LTS}(\mathcal{T})$, то такой путь назовем *начальным путем*. Особо выделим также пути, которые опустошают выходной временно́й контекст конфигураций без прочтения входных символов. *Завершающим путем* из конфигурации q будем называть путь $\overline{\pi}(q)$ $\mathcal{LTS}(\mathcal{T})$, который удовлетворяет следующему рекурсивному определению:

1) если $q.c = \varnothing$, то $\overline{\pi}(q)$ — пустая последовательность переходов;
2) если $q.c \ne \varnothing$ и $(b, q.T) \in q.c$ — любой из выходных временны́х символов с наименьшей задержкой $q.T$, то $\overline{\pi}(q) = q \xrightarrow{q.T} q + q.T \xrightarrow{b} \overline{\pi}(out(q + q.T, b))$.

Путь π будем называть *приведенным*, если в нем не встречаются несколько идущих подряд переходов продвижения времени. Очевидно, что любые два идущих подряд перехода продвижения времени $q \xrightarrow{\delta_1} q' \xrightarrow{\delta_2} q''$ можно заменить одним переходом $q \xrightarrow{\delta_1+\delta_2} q''$. Проделав это преобразование для всех пар идущих подряд переходов продвижения времени в произвольном пути π, который ведет из конфигурации q_1 в конфигурацию q_k, получим приведенный путь π' из q_1 в q_k, в котором переходы продвижения времени перемежаются переходами приема запросов и выдачи откликов. Этот приведённый путь π' определяется по пути π однозначно; назовем его *редукцией* пути π.

Начальный путь π назовем *полным*, если последняя конфигурация пути q_k удовлетворяет условию $q_k.c = \varnothing$. Каждому пути π в $\mathcal{LTS}(\mathcal{T})$ поставим в соответствие пару, состоящую из входного и выходного временны́х слов $TR(\pi) = (\alpha, \beta)$, по следующим правилам:

1. Если π — это пустой путь, то $TR(\pi) = (\varepsilon, \varepsilon)$.
2. Пусть π' — путь в $\mathcal{LTS}(\mathcal{T})$ из конфигурации q_0 в конфигурацию q', для которого $TR(\pi') = (\alpha', \beta')$, и пусть путь π получен из пути π' добавлением перехода $E = q' \xrightarrow{x} q$. Если переход E — это переход

- продвижения времени, то $TR(\pi) = TR(\pi')$;
- приема запроса, то $TR(\pi) = (\alpha, \beta')$, где $\alpha = \alpha', (x, t(\alpha') + q'.t)$;
- выдачи отклика, то $TR(\pi) = (\alpha', \beta)$, где $\beta = \beta', (x, t(\alpha') + q'.t)$.

Исходя из этого определения, нетрудно заметить, что для любого пути π и его редукции π' верно равенство $TR(\pi) = TR(\pi')$.

Записью $TR(\mathcal{LTS}(\mathcal{T}))$ обозначим множество всех пар $TR(\pi)$, соответствующих полным путям π в $\mathcal{LTS}(\mathcal{T})$. Как видно из предыдущего замечания, при изучении отношения $TR(\mathcal{LTS}(\mathcal{T}))$ можно ограничиться рассмотрением одних лишь приведённых путей.

Покажем, что размеченная система переходов $\mathcal{LTS}(\mathcal{T})$ полностью характеризует поведение TFSM $\mathcal{T}$.

Утверждение 1. *Для любого TFSM $\mathcal{T}$ верно равенство $TR(\mathcal{T}) = TR(\mathcal{LTS}(\mathcal{T}))$.*

Доказательство. Вначале докажем индукцией по n, что для любого вычисления TFSM $\mathcal{T}$, которое преобразует входное временно́е слово $\alpha = (a_1, t_1), \ldots, (a_n, t_n)$ в выходное временно́е слово $\beta = (b_1, \tau_1), \ldots, (b_n, \tau_n)$ и оканчивается в состоянии s_n, существует приведенный полный путь π в $\mathcal{LTS}(\mathcal{T})$, который ведет в конфигурацию $(s_n, \tau_n - t_n, \emptyset)$ и удовлетворяет равенству $TR(\pi) = (\alpha, \beta)$. Отсюда будет следовать включение $TR(\mathcal{T}) \subseteq TR(\mathcal{LTS}(\mathcal{T}))$.

Базис индукции. Пусть TFSM $\mathcal{T}$ имеет вычисление $r = s_0 \xrightarrow{(a_1,t_1)/(b_1,\tau_1)} s_1$. Тогда согласно определению вычисления TFSM, существует трансдукция $(s_0, a_1, g, b_1, d, s_1) \in \rho$, удовлетворяющая условиям $t_1 \in g$ и $\tau_1 = t_1 + d$. Тогда согласно определению отношения переходов в $\mathcal{LTS}(\mathcal{T})$ существует полный путь $\pi = q_0 \xrightarrow{t_1} q_1 \xrightarrow{a_1} q_2 \xrightarrow{d} q_3 \xrightarrow{b_1} q_4$. Нетрудно видеть, что $TR(\pi) = ((a_1, t_1), (b_1, \tau_1))$ и при этом $q_4 = (s_1, d, \emptyset)$.

Индуктивный переход. Пусть TFSM $\mathcal{T}$ имеет вычисление r', которое преобразует входное временно́е слово $\alpha' = (a_1, t_1), \ldots, (a_n, t_n)$ в выходное временное слово $\beta' = (b_1, \tau_1), \ldots, (b_n, \tau_n)$, достигает состояния управления s_n, и может быть продолжено выполнением некоторой трансдукции $s_n \xrightarrow{(a,t)/(b,\tau)} s_{n+1}$. Тогда по индуктивному предположению вычислению r' соответствует некоторый полный редуцированный путь π' в $\mathcal{LTS}(\mathcal{T})$, для которого имеет место равенство $TR(\pi') = (\alpha, \beta)$. Рассмотрим в этом пути последний переход приема запроса $q \xrightarrow{a_n} q'$, и пусть k — это наименьшее число, для которого выполнено неравенство $t_n \le \tau_k$. Тогда путь π' можно разделить на две половины π'_1 и π'_2, первая из которых оканчивается переходом приема запроса $q \xrightarrow{a_n} q'$. Так как это последний переход приема запроса в пути π', вторая половина π'_2 пути π' имеет вид

$$\pi'_2 = q' \xrightarrow{\tau_k - t_n} q'_1 \xrightarrow{b_k} q'_2 \xrightarrow{\tau_{k+1} - \tau_k} q'_3 \xrightarrow{b_{k+1}} \cdots \xrightarrow{\tau_n - \tau_{n-1}} q'_{m-1} \xrightarrow{b_n} q'_m,$$

и состоит только из переходов продвижения времени и переходов выдачи откликов $b_k, \ldots, b_n$. При этом для всех конфигураций q'_i этой половины пути выполняется равенство $q'_i.s = s_n$.

Предположим, что параметры t и τ выполненной трансдукции $s_n \xrightarrow{(a,t)/(b,\tau)} s_{n+1}$ удовлетворяют неравенствам $\tau_i \le t \le \tau_{i+1}$ и $\tau_j \le \tau \le \tau_{j+1}$ для некоторой пары чисел i, j, $k \le i \le j \le n$. Тогда вычисление r TFSM $\mathcal{T}$, которое продолжает вычисление r' указанным выполнением трансдукции преобразует входное временно́е слово $\alpha = (a_1, t_1), \ldots, (a_n, t_n), (a, t)$ в выходное временно́е слово

$$\beta = (b_1, \tau_1), \ldots, (b_j, \tau_j), (b, \tau), (b_{j+1}, \tau_{j+1}), \ldots, (b_n, \tau_n).$$

Но, как можно заметить, для указанных значений параметров t и τ, вставив переход приема запроса a и переход выдачи отклика b в последовательность переходов пути π'_2, можно построить

следующий приведенный путь из конфигурации q' в $\mathcal{LTS}(\mathcal{T})$:

$$\begin{aligned}\pi_2 \quad = \quad & q' \xrightarrow{\tau_k - t_n} q'_1 \xrightarrow{b_k} q'_2 \xrightarrow{\tau_{k+1}-\tau_k} q'_3 \xrightarrow{b_{k+1}} \cdots \xrightarrow{\tau_i - \tau_{i-1}} q'_{2i-1} \xrightarrow{b_i} q'_{2i} \\ & \xrightarrow{t-\tau_i} q_{2i+1} \xrightarrow{a} q_{2i+2} \xrightarrow{\tau_{i+1}-t} q_{2i+3} \xrightarrow{b_{i+1}} \cdots \xrightarrow{b_j} q_{2j} \\ & \xrightarrow{\tau-\tau_j} q_{2j+1} \xrightarrow{b} q_{2j+2} \xrightarrow{\tau_{j+1}-\tau} q_{2j+3} \xrightarrow{b_{j+1}} \cdots \xrightarrow{b_n} q_{m+4}.\end{aligned}$$

Сцеплением путей π'_1 и π_2 получаем последовательность переходов π в $\mathcal{LTS}(\mathcal{T})$, являющуюся полным путем, которому соотвествует пара временны́х слов (α, β).

Аналогичным образом можно построить путь π в $\mathcal{LTS}(\mathcal{T})$, которому соответствует пара временны́х слов (α, β), и в тех случаях, когда параметры t и τ выполненной трансдукции $s_n \xrightarrow{(a,t)/(b,\tau)} s_{n+1}$ удовлетворяют неравенствам $\tau_n \le t$ или $\tau_n \le \tau$.

Тем самым, обоснование индуктивного перехода завершено, и вместе с этим получено обоснование включения $TR(\mathcal{T}) \subseteq TR(\mathcal{LTS}(\mathcal{T}))$.

Для обоснования включения $TR(\mathcal{LTS}(\mathcal{T})) \subseteq TR(\mathcal{T})$ достаточно показать, что каждый полный редуцированный путь π в $\mathcal{LTS}(\mathcal{T})$, для которого $TR(\pi) = (\alpha, \beta)$, соответствует вычислению TFSM $\mathcal{LTS}(\mathcal{T})$, которое преобразует входное временно́е слово α в выходное временное слово β. Но, как видно из определения множества пар $TR(\pi) = (\alpha, \beta)$, вычисление r, соответствующее пути π, однозначно определяется переходами приема запросов вида $q \xrightarrow{a} in(q, c, d)$ в этом пути. При этом временны́е пометки входного слова α, на котором проводится вычисление r, и временны́е пометки соответствующего выходного слова β однозначно определяются переходами продвижения времени в пути π. Таким образом, вычисление r восстанавливается по пути π в однозначно. □

Основное преимущество $\mathcal{LTS}(\mathcal{T})$ заключается в том, что эта система переходов имеет точно такое же устройство, какое имеют трассовые модели, используемые для верификации временны́х автоматов [3]. Тем самым, мы установили взаимосвязь рассматриваемой модели TFSM и модели временны́х автоматов, и это позволяет применять инструментальные средства верификации моделей программ, подобные комплексу верификации Uppaal [18], для анализа поведения TFSM с модифицированной операционной семантикой. Однако такой анализ сопряжен с определенными трудностями: пространство состояний $\mathcal{LTS}(\mathcal{T})$ может быть континуально-бесконечным. Отчасти это связано с тем, что размер конфигураций в $\mathcal{LTS}(\mathcal{T})$ в общем случае может быть неограниченно большим. Тем не менее, при определенных условиях этой неограниченности размера конфигураций можно избежать.

Пусть u, v — пара вещественных чисел, удовлетворяющих неравенству $0 < u \le v$. TFSM $\mathcal{T} = (S, \rho, s_0)$ называется (u, v)-*прогрессивным*, если для каждой трансдукции $(s, a, g, b, d, s') \in \rho$ ее допустимый интервал $g = (u', v')$ удовлетворяет условиям $u \le u'$ и $v' \le v$, т.е. допустимые интервалы всех трансдукций $\mathcal{T}$ вложены в интервал (u, v). Тогда справедливо следующее утверждение.

Утверждение 2. *Если $\mathcal{T}$ является (u, v)-прогрессивной TFSM с точечными задержками, то существует такое целое число ℓ, что $|q.c| < \ell$ выполняется для любой конфигурации q, достижимой в $\mathcal{LTS}(\mathcal{T})$ из начальной конфигурации q_0.*

Доказательство. Пусть в трансдукциях $\mathcal{T}$ минимальная левая граница допустимого интервала равна $u > 0$, а максимальная точечная задержка равна $d_{max} > 0$.

Рассмотрим $\ell = \lceil \frac{d_{max}}{u} \rceil$. Предположим, что из начальной конфигурации q_0 достижима конфигурация q, для которой $|q.c| = \ell + 1$. Это означает, что TFSM $\mathcal{T}$ имеет вычисление

$$\begin{aligned}r \quad = \quad & s_0 \xrightarrow{(a_1,t_1)/(b_1,d_1)} \cdots \xrightarrow{(a_{n-\ell-1},t_{n-\ell-1})/(b_{n-\ell-1},d_{n-\ell-1})} s_{n-\ell-1} \xrightarrow{(a_{n-\ell},t_{N-\ell})/(b_{n-\ell},d_{N-\ell})} \\ & \cdots \xrightarrow{(a_{n-1},t_{N-1})/(b_{N-1},d_{n-1})} s_{n-1} \xrightarrow{(a_n,t_n)/(b_n,d_n)} s_n,\end{aligned}$$

которому в размеченной системе переходов $\mathcal{LTS}(\mathcal{T})$ соответствует путь из начальной конфигурации q_0 в конфигурацию

$$q = \langle s_n, t_n, \{(b_{j_1}, d_{j_1} - \sum_{i=j_1}^{n} t_i), (b_{j_2}, d_{j_2} - \sum_{i=j_2}^{n} t_i), \dots, (b_{j_{\ell+1}}, d_{j_{\ell+1}} - \sum_{i=j_{\ell+1}}^{n} t_i)\}\rangle,$$

где $j_1, \dots, j_{\ell+1}$ такие, что $1 \le j_1 < \dots < j_{\ell+1} \le n$, следовательно, $j_1 < n - \ell - 1$.

Тогда для временно́й метки символа b_{j_1} в этой конфигурации верны неравенства

$$d_{j_1} - \sum_{i=j_1}^{n} t_i \le d_{max} - \sum_{i=j_1}^{n} t_i \le d_{max} - \sum_{i=j_1}^{n} u \le d_{max} - u(\ell + 1).$$

Поскольку $\ell = \lceil \frac{d_{max}}{u} \rceil$, приходим к заключению о том, что $d_{j_1} - \sum_{i=j_1}^{n} t_i < 0$, вопреки тому, что все временные метки в конфигурациях являются неотрицательными. □

Ещё одна трудность при анализе размеченной системы переходов $\mathcal{LTS}(\mathcal{T})$ обусловлена возможностью продолжить любой путь только переходами продвижения времени. Чтобы обнаружить и исключить из дальнейшего рассмотрения такие избыточные пути, мы выделим из всего множества конфигураций так называемые *блокирующие конфигурации* [17]. Для каждого состояния s TFSM $\mathcal{T}$ обозначим записью $up(s)$ максимальную верхнюю границу v допустимых интервалов $g = (u, v)$ всех трансдукций (s, a, g, b, d, s'), исходящих из состояния управления s. Конфигурация q называется *блокирующей конфигурацией*, если $q.c = \emptyset$ и $q.t > up(q.s)$. Все остальные конфигурации в $Q(\mathcal{T})$ называются *прогрессивными*. Из определения следует, что только прогрессивные конфигурации имеют значение для анализа преобразования входных временны́х слов в выходные. Если путь в $\mathcal{LTS}(\mathcal{T})$ достигает блокирующей конфигурации, то не существует такого продолжения пути в котором присутствуют переходы приема запросов или выдачи откликов. Поэтому анализировать пути, исходящие из таких конфигураций, нет необходимости. Следующее утверждение предлагает алгоритм для обнаружения *блокирующих конфигураций* в (u, v)-прогрессивных TFSM.

Утверждение 3. *Для каждого состояния s в (u, v)-прогрессивной TFSM $\mathcal{T}$ существует такая величина $c_s \ge 0$, что произвольная конфигурация $q = (s, TC)$ является блокирующей тогда и только тогда, когда $q.t > c_s$.*

Доказательство. Пусть s — состояние управления TFSM $\mathcal{T}$, и пусть $v = up(s)$. Рассмотрим множество конфигураций, $Q_s = \{q \in Q \mid (q.s = s) \wedge (q.t \ge v) \wedge (q.c = \emptyset)\}$ и соответствующее множество временных меток $TS_s = \{t \mid (q \in Q_s) \wedge (q.t = t)\}$. Легко видеть, что TS_s имеет точную нижнюю грань c_s, которая удовлетворяет условиям утверждения. □

Однако количество прогрессивных конфигураций может быть континуально-бесконечно. Поэтому следующим шагом в наших дальнейших исследованиях будет отыскание такого отношения эквивалентности ~ на множестве прогрессивных конфигураций $\mathcal{LTS}(\mathcal{T})$, чтобы фактор-модель $\mathcal{LTS}_{fin}(\mathcal{T}) = \mathcal{LTS}(\mathcal{T})/\sim$ была конечной и при этом симуляционно эквивалентной системе переходов $LTS(\mathcal{T})$. Построенная таким образом модель $\mathcal{LTS}_{fin}(\mathcal{T})$ открывает возможности применения традиционных методов для верификации реагирующих систем реального времени, представленных временны́ми конечными автоматами-преобразователями.

4. Заключение

Основной результат статьи — это описание и исследование новой операционной семантики для модели TFSM, предложенной ранее в работе [16]. Благодаря этому удалось показать, что вычисления TFSM можно моделировать конечными временны́ми автоматами. В связи с этим возникает ряд вопросов, касающихся сложности такого моделирования. Например, интересно выяснить, какого количества таймеров и состояний достаточно иметь конечному временно́му автомату для моделирования заданной TFSM. В том случае, если будет обнаружено, что сложность временно́го автомата, моделирующего вычисления TFSM, существенно превосходит размер этой TFSM, сведение задачи верификации TFSM к аналогичной задаче для временны́х автоматов может оказаться нецелесообразным, и для этой цели будет выгоднее разрабатывать независимые методы анализа поведения автоматов-преобразователей.

В связи с этим можно заметить, что использование методов верификации, опирающихся на размеченные системы переходов осложняется тем, что количество конфигураций в $LTS(\mathcal{T})$, которая представляет всевозможные вычисления TFSM $\mathcal{T}$, может быть несчётно. Поэтому необходимо создать такие методы преобразования бесконечной системы переходов $LTS(\mathcal{T})$ в конечную систему переходов $\mathcal{LTS}_{fin}(\mathcal{T})$, связанную с $LTS(\mathcal{T})$ отношениями. В наших дальнейших исследованиях мы собираемся найти подходящее отношение симуляции ~, которое (i) сохраняет преобразование входных временных слов в выходные временные слова, осуществляемое TFSM $\mathcal{T}$, и (ii) предложить эффективный алгоритм построения $\mathcal{LTS}_{fin}(\mathcal{T})$ такой, что $LTS(\mathcal{T})$ ~ $\mathcal{LTS}_{fin}(\mathcal{T})$. Мы предполагаем, что такое отношение может быть найдено при помощи определения понятия шаблона конфигурации, которое будет играть ту же роль для анализа TFSM, что для анализа свойств временных автоматов играет система регионов (см. [3]).

Авторы выражают признательность рецензенту за ценные замечания, позволившие улучшить облик этой статьи.

References

[1] A. Gill *et al.*, «Introduction to the Theory of Finite-state Machines», 1962.

[2] A. Y. Savelev, «Prikladnaya teoriya cifrovyh avtomatov», 1987, In Russian.

[3] R. Alur and D. Dill, «A theory of timed automata», *Theoretical computer science*, vol. 126, no. 2, pp. 183–235, 1994.

[4] E. Asarin, P. Caspi, and O. Maler, «Timed regular expressions», *Journal of the ACM*, vol. 49, no. 2, pp. 1–35, 2001.

[5] E. Asarin, P. Caspi, and O. Maler, «A Kleene theorem for timed automata», in *Proceedings of 12-th Annual IEEE Symposium on Logic in Computer Science (LICS'97)*, IEEE, 1997, pp. 160–171.

[6] R. Alur and P. Madhusudan, «Decision Problems for Timed Automata: A Survey, Formal Methods for the Design of Real-Time Systems», in *Proceedings of the 4-th International School on Formal Methods for the Design of Computer, Communication, and Software Systems (SFM'04)*, Springer, 2004, pp. 1–24.

[7] S. Lasota and I. Walukiewicz, «Alternating timed automata», *ACM Transactions on Computational Logic (TOCL)*, vol. 9, no. 2, pp. 1–26, 2008.

[8] M. Gromov, K. El-Fakih, N. Shabaldina, and N. Yevtushenko, «Distinguing Non-deterministic Timed Finite State Machines», in *Formal Techniques for Distributed Systems, Lecture Notes in Computer Science*, vol. 5522, Springer, 2009, pp. 137–151.

[9] M. G. Merayo, M. Nunez, and I. Rodriguez, «Formal testing from timed finite state machines», *Computer networks*, vol. 52, no. 2, pp. 432–460, 2008.

[10] D. Bresolin, K. El-Fakih, T. Villa, and N. Yevtushenko, «Deterministic Timed Finite State Machines: Equivalence Checking and Expressive Power», *Proceedings of the 5-th International Symposium on Games, Automata, Logics and Formal Verification*, pp. 203–216, 2014.

[11] A. Tvardovskii and N. Yevtushenko, «Minimizing timed Finite State Machines», *Tomsk State University Journal of Control and Computer Science*, vol. 29, no. 4, pp. 77–83, 2014.

[12] A. S. Tvardovskii, N. V. Yevtushenko, and M. L. Gromov, «Minimizing finite state machines with time guards and timeouts», *Proceedings of the Institute for System Programming of the RAS*, vol. 29, no. 4, pp. 139–154, 2017.

[13] A. S. Tvardovskii and N. V. Yevtushenko, «Deriving homing sequences for Finite State Machines with timed guards», *Sistemnaya informatika*, vol. 17, pp. 1–10, 2020.

[14] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, «OpenFlow: enabling innovation in campus networks», *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 69–74, 2008.

[15] E. Vinarskii, J. Lopez, N. Kushik, N. Yevtushenko, and D. Zeghlache, «A Model Checking Based Approach for Detecting SDN Races», in *Proceedings of the 31-st IFIP WG 6.1 International Conference on Testing Software and Systems (ICTSS)*, Springer, 2019, pp. 194–211.

[16] E. M. Vinarskii and V. A. Zakharov, «On the verification of strictly deterministic behavior of Timed Finite State Machines», *Proceedings of ISP RAS*, vol. 30, no. 3, pp. 325–340, 2018.

[17] C. Baier and J.-P. Katoen, *Principles of model checking*. Cambridge: MIT Press Cambridge, 2008.

[18] G. Behrmann, A. David, and K. G. Larsen, «A tutorial on Uppaal», in *Proceedings of the International School on Formal Methods for the Design of Computer, Communication, and Software Systems*, Springer, 2004, pp. 200–236.