УДК 681.32 DOI: 10.31114/2078-7707-2022-4-71-78

Тестирование систем с параллелизмом поведения на основе сокращенного графа достижимых состояний

Л.Д. Черемисинова

Объединений институт проблем информатики НАН Беларуси, г. Минск,
cld@newman.bas-net.by

***Аннотация* — Рассматривается задача верификации реактивных систем управления с параллелизмом поведения. Полагается, что описание функционирования устройства, заданного на языке параллельных автоматов, корректно, проверке подлежит схемная (или программная) реализация. Генерация тестовой последовательности основана на обходе графа достижимых состояний параллельного автомата. Предлагается метод сокращения графа достижимости, который основан на предположении независимости параллельно происходящих переходов автомата. Усечение графа достижимости достигается за счет доопределения частичного порядка на множестве переходов, позволяющего существенно уменьшить число рассматриваемых состояний системы управления и, соответственно, число вершин графа. Получаемое сокращение графа достижимых состояний позволяет решать задачу построения тестов для более сложных систем управления.**

***Ключевые слова* — параллельный автомат, верификация, граф достижимых состояний, спецификация на проектирование.**

I. Введение

С увеличением сложности и ростом требований к качеству проектируемых управляющих систем резко возрастает трудоемкость их тестирования, которая растет быстрее, чем сложность тестируемых систем. Тестирование становится одним из наиболее важных и широко используемых методов проверки схемных реализаций или программного обеспечения. По разным оценкам, в настоящее время на этап тестирования и отладки приходится до 60 – 80 % общих затрат на разработку управляющих систем.

Одной из новых технологий для решения этой проблемы является тестирование схемной реализации системы управления на соответствие модели ее требуемого поведения, задаваемого спецификацией на проектирование [1, 2, 3]. Под тестированием далее понимается проверка схемной реализации на вход-выходное соответствие модели (input-output conformance), которая состоит в 1) генерации проверяющей последовательности на основе заданной спецификации; 2) моделировании схемной реализации на полученной тестовой последовательности; 3) наблюдении ее реакций и 4) определении, соответствуют ли вход-выходные реакции схемной реализации системы управления спецификации на ее проектирование.

Можно выделить следующие два основных способа генерации входных воздействий (для проверяющей последовательности): генерация псевдослучайных наборов воздействий и генерация наборов воздействий и проверяющей последовательности в целом, исходя из описания модели. Минусы первого подхода заключаются в том, что число возможных входных воздействий в каждом состоянии схемы (и модели) экспоненциально зависит от числа переменных, и, главное, не все такие наборы попадают в область определения модели (а значит, и реализации). Второй подход состоит в генерации тестовой последовательности на основе модели, описывающей желаемое поведение системы, которое задается спецификацией на проектирование устройства на некотором языке. Предполагается, что описание спецификации является правильным и корректным. Тесты строятся на основе спецификации алгоритмическим способом. Такой подход обеспечивает не только наиболее полное тестирование системы на области ее функционирования, но и позволяет сократить длину проверяющей последовательности

В работе задача тестирования на основе моделей рассматривается для случая реактивных систем управления [4]. Их особенность (в отличие от систем трансформационного типа) заключается в непрерывном обмене сигналами с внешней средой. Наиболее известным способом моделирования реактивных систем является конечный автомат. Задача верификации на основе моделей, представленных конечными автоматами, достаточно хорошо изучена [5, 6, 7].

Существует ряд систем, в которых выразительных средств аппарата конечных автоматов оказывается недостаточно. К ним относятся системы, обеспечивающие управление взаимодействующими компонентами, которые работают параллельно и асинхронно. Параллелизм, присутствующий в объектах управления, отражается в функциональной модели цифровых систем, управляющих данными объектами. Параллелизм в спецификации возникает по разным причинам. Например, это может быть многоблочная система, в которой некоторые последовательности операций выполняются

параллельно, но в разных компонентах. И, наконец, системы с «истинным» параллелизмом», когда некоторые процессы в системе происходят параллельно и независимо даже и в одном и том же компоненте, и нет необходимости контролировать порядок их выполнения.

Для цифровых систем рассматриваемого в настоящей работе класса устройств характерно также и то, что управляющие воздействия и сигналы о состоянии объектов управления описываются булевыми переменными, лишь небольшой процент всей информации является числовым. Для задания спецификации таких систем целесообразно использовать языки описания систем с параллелизмом поведения. Наиболее популярными формализмами при тестировании на основе моделей являются сети взаимодействующих конечных автоматов, сети Петри [8, 9, 10] и системы помеченных переходов (labelled transition systems – LTS) [11]. Вопросы тестирования систем параллелизмом поведения в настоящее время еще недостаточно хорошо изучены. В настоящей работе в качестве языка задания спецификации используется язык ПРАЛУ представления параллельных алгоритмов логического управления, а точнее задание описаний в стандартном виде моделью, названной параллельным автоматом [4]. Алгоритмы в таком виде представляют собой подкласс цветных сетей Петри – расширенные сети свободного выбора.

Основным инструментом, лежащим в основе методов анализа поведенческих свойств моделей, является граф достижимых состояний устройства, в процессе обхода которого генерируется тестовая последовательность. Верификация на основе графа достижимости является одним из наиболее изученных и естественных подходов к верификации систем управления. Построение графа достижимости (в явном или неявном виде) всегда возможно для моделей с конечным пространством состояний. Однако проблема заключается в экспоненциальном росте размера пространства возможных состояний системы. Как результат, граф достижимости сталкивается с проблемой взрыва числа состояний, что негативно влияет на размер практически решаемых задач и производительность тестирования сложных систем.

Анализируя граф достижимости, можно исследовать корректность параллельного алгоритма управления, но это исследование будет неполным. Для полноты анализа необходимо учесть также логические характеристики параллельного автомата: достижимые состояния на множестве его переменных, которые существенно влияют на порядок выполнения переходов и их срабатывание.

В настоящей работе предлагаются пути сокращения графа достижимых состояний и метод построения сокращенного графа достижимости для автоматов с параллелизмом поведения. В основе метода лежит введенное отношение частичного порядка на множестве параллельно срабатывающих переходов автомата и условия частичного упорядочения пар переходов, порождаемые учетом их информационного взаимодействия. Кроме того, граф достижимости параллельного алгоритма содержит достижимые полные состояния не только на множестве его меток, но и на множестве его логических переменных.

II. Постановка задачи тестирования систем с параллелизмом поведения

При тестировании программных и схемных реализаций систем управления на основе моделей тестовая последовательность генерируется на основе спецификации на проектирование. Предполагается, что описание спецификации является правильным и корректным. Внутренняя структура тестируемой реализации может рассматриваться как черный ящик. Оценка полноты тестирования определяется степенью покрытия сценариев работы устройства, определяемых, исходя из спецификации.

Основным средством тестирования схемной реализации на соответствие спецификации является моделирование, для проведения которого предварительно строится тестовая последовательность, (или просто тест). Тестирование осуществляется на уровне вход-выходных последовательностей путем выполнения экспериментов над реальным устройством или его моделью на некотором языке проектирования. При моделировании тестовые наборы подаются на входные полюсы моделируемой системы, определяются изменения значений сигналов, происходящие на ее выходах, которые сравниваются с эталонными значениями. Если модель корректна и тестируемое устройство реализует заданное спецификацией поведение (и только его), то успешное прохождение тестов, сгенерированных надлежащим образом на основе данной модели, может служить достаточной гарантией правильности реализации системы.

Задача синтеза проверяющего теста заключается в построении конечного множества воздействий на систему, по реакциям на которые можно определить правильность ее функционирования. В связи с практической значимостью и теоретическим интересом наиболее изученной областью верификации на основе моделей является тестирование конечных автоматов. Тестовая последовательность формируется путем объединения нескольких подпоследовательностей, которые, как правило, имеют перекрытия. В литературе встречается достаточное число работ, в которых предлагаются эвристики для сокращения числа перекрытий с целью уменьшения общей длины тестовой последовательности. Тестовая последовательность строится следующим образом:

– заранее, до начала процесса тестирования, в виде единого маршрута по графу достижимых состояний, который проходит через все дуги графа;

– заранее в виде множества маршрутов, которые начинаются в начальной вершине (состоянии) графа и покрывают в совокупности все дуги графа;

– в процессе моделирования системы, в этом случае тестовые наборы генерируются динамически при обходе графа достижимых состояний в зависимости от откликов схемы на поданные наборы.

Очевидно, что построение тестовой последовательности в виде единого маршрута возможно, если граф достижимых состояний является сильно связным, т. е. если из каждой его вершины достижима любая другая вершина. Если граф достижимости не является сильно связным, то возможно построение тестовой последовательности в виде множества маршрутов из начальной вершины графа. В этом случае при моделировании управляющей системы требуется ее рестарт (сброс триггеров блока памяти) при переходе от одного теста к другому. После рестарта производится выбор следующей тестовой последовательности.

Задача построения тестовой последовательности на основе графа достижимых состояний заключается в построении такого кратчайшего ориентированного маршрута (чередующейся последовательности вершин и дуг) на орграфе $G = (V, E)$, который проходит через каждую дугу графа, по крайней мере, один раз (в общем случае не один раз). В этой постановке задача построения кратчайшего ориентированного маршрута аналогична задаче китайского почтальона [12], в которой ищется кратчайший путь, проходящий через все дуги заданного орграфа. В силу трудоемкости решения такой задачи нахождение точного решения (с минимумом числа возможных повторных прохождений дуг графа достижимости) в общем случае не представляется возможным.

Для решения задачи обхода графа достижимых состояний можно использовать один из известных методов обхода дуг ориентированного графа, разработанных для случая детерминированных автоматных моделей [6, 7, 13, 14, 15]. По последовательности переходов алгоритма управления, соответствующих меткам найденной последовательности дуг ориентированного графа, можно определить входные стимулы тестовой последовательности для подачи на входы тестируемой реализации системы управления. Следует заметить, что таким образом для каждого достижимого состояния системы определяются только те стимулы, которые заданы в реализуемой спецификации, т. е. тестирование проводится только на заданной спецификацией области определения. Если в реализации допустимы какие-то другие стимулы, то это никак не влияет на процесс тестирования. Фактически это означает, что для заданной реализации исходного модельного алгоритма управления тестируется некоторая часть ее функциональности, заданная переходами по входным стимулам и состояниям модельного алгоритма, достижимым из начального состояния.

Далее задача верификации рассматривается для случая реактивных систем управления [4] с параллелизмом поведения, которое задано на языке параллельных автоматов. Тестовая последовательность строится на основе предварительно построенного графа достижимых состояний параллельного автомата и представляется последовательностью четверок (S_i^b, X_i, S_i^e,Y_i), где S_i^b и S_i^e – начальное и конечное множества частичных состояний, X_i – входной стимул, Y_i – изменение значений переменных при подаче на вход схемы входного набора X_i.

III. Параллельный автомат для задания поведения систем с параллелизмом поведения

Для задания поведения систем управления далее используется модель параллельного автомата на языке ПРАЛУ [4], которая представляет подкласс раскрашенных сетей Петри – расширенные сети свободного выбора. В отличие от классического конечного автомата в параллельном автомате рассматриваются так называемые частичные состояния. В том смысле, что в любой момент времени параллельный автомат может одновременно находиться в нескольких таких состояниях.

Параллельный автомат можно рассматривать как динамическую дискретную систему, которая допускает параллельно происходящие процессы, которым соответствуют параллельные частичные состояния. Множество таких состояний в каждый момент времени определяет полное (глобальное) состояние S_t, которое можно интерпретировать как маркировку сети Петри. Следует отметить, что количество полных состояний растет экспоненциально с ростом числа частичных состояний и переходов соответствующего параллельного автомата [4]. Таким образом, преобразование параллельного автомата даже с несколькими десятками частичных состояний и переходов в эквивалентную последовательную форму может оказаться практически невозможным.

Алгоритм поведения систем управления на языке параллельных автоматов задается тройкой:

– множеством S частичных состояний;

– множествами X и Y входных и выходных переменных;

– множеством T переходов между подмножествами частичных состояний S_{i1}, $S_{i2} \subseteq S$.

Предполагается, что на изменение значений входных переменных из множества X не накладывается никаких ограничений (они изменяются во внешней среде системы, описываемой автоматом), а значения выходных переменных из множества Y изменяются только в результате срабатывания некоторых переходов параллельного автомата.

Переход параллельного автомата задается в форме $\tau_i = (\gamma_i^1, k_i^1) \rightarrow (\gamma_i^2, k_i^2)$, где метки γ_i^1 и γ_i^2 трактуются как подмножества частичных состояний из S, в которых автомат находится перед и после срабатывания i-го перехода; k_i^1 и k_i^2 – элементарные конъюнкции входных и выходных булевых переменных,

трактуемых как условие перехода и выходные сигналы, сопровождающие переход.

Переход срабатывает, когда текущее полное состояние S_t автомата включает все частичные состояния из γ_i^1, а переменные автомата принимают значения, обращающие элементарную конъюнкцию k_i^1 в единицу. После срабатывания перехода переменным из k_i^2 присваиваются значения выходных переменных, обращающие ее в единицу (состояния других переменных не изменяются), а полное состояние автомата S_t заменяется на $S_{t+1} = (S \setminus \gamma_i^1) \cup \gamma_i^2$. Переходы параллельного автомата могут происходить последовательно или параллельно, Очевидно, что порядок срабатывания переходов зависит не только от состояния автомата S_t, но и от условий, определенных на множестве достигнутых к этому моменту значений переменных (аналогично раскрашенным сетям Петри): не только от $\gamma_i^1 \subseteq S_t$, но и $k_i^1 \wedge W_t = 1$, где W_t – состояние автомата на множестве логических переменных системы управления, которое было достигнуто к этому моменту времени. Переход разрешен, когда выполняются оба эти условия.

В случае корректного алгоритма (что предполагается при тестировании его реализации) для любой пары переходов τ_i и τ_j $(i \neq j)$ выполняются следующие условия: если $\gamma_i^1 \cap \gamma_j^1 \neq \varnothing$, то $\gamma_i^1 = \gamma_j^1$, и, если переходы τ_i и τ_j параллельны, то $\gamma_i^2 \neq \gamma_j^2$.

Таким образом, динамика сети описывается в пространстве достигаемых состояний Si на множестве частичных состояний и состояний на множестве логических сигналов. Для вычисления следующего состояния St+1 необходимо определить множество T переходов τi = (γi1, ki1) → (γi2, ki2), которые срабатывают при текущем состоянии St и множестве Wt значений переменных. Следующие состояния пересчитываются следующим образом:

– St+1 = (St \ γ1) ∪ γ2, где γ1 = ∪ γi1/τi ∈ T и γ2 = ∪ γi2/τi ∈ T – объединения множеств частичных состояний из γi1 и γi2 для всех переходов τi ∈ T;

– состояние W_{t+1} на множестве переменных получается из W_t таким изменением значений тех выходных переменных, которые входят в конъюнкции k_i^2 всех переходов $\tau_i \in T$, чтобы все $k_i^2 = 1$.

В качестве примера приведем описание простого параллельного автомата из работы [16]. Автомат определен на множестве $S = \{1, 2, \ldots, 11\}$ частичных состояний, множествах $X = \{x_1, x_2\}$ и $Y = \{y_1, \bar{y}_2\}$ входных и выходных (управляющих) переменных и девяти переходах:

$\tau_1 = (1, x_1 x_2) \rightarrow (10, y_1\ \bar{y}_2)$;

$\tau_2 = (10, \bar{x}_2) \rightarrow (2.3.4, –)$;

$\tau_3 = (2, –) \rightarrow (5.6, y_1)$;

$\tau_4 = (3.5, x_2) \rightarrow (8, –)$;

$\tau_5 = (4, \bar{x}_1) \rightarrow (7, \bar{y}_1)$;

$\tau_6 = (4, x_1) \rightarrow (9, y_2)$;

$\tau_7 = (7, \bar{x}_2) \rightarrow (9, –)$;

$\tau_8 = (6.8.9, –) \rightarrow (11, \bar{y}_2)$;

$\tau_9 = (11, x_1) \rightarrow (1, –)$.

IV. Генерация графа достижимых состояний и его сокращение

Основной подход к тестированию систем управления состоит в исследовании пространства всех возможных состояний его модели и всех переходов, которые система может совершать между этими состояниями. Перечисление пространства состояний отображается в графе достижимых состояний, обход которого лежит в основе анализа поведенческих свойств системы управления с параллелизмом поведения. Граф достижимых состояний, задавая динамику поведения системы управления при всевозможных изменениях сигналов на его входных полюсах, может служить также и спецификацией для реализации системы.

Граф достижимости представляет собой ориентированный граф, вершинам которого соответствуют все возможные полные состояния S_t автомата, а дугам –переходы между этими состояниями. Дуга графа помечается символом перехода τ_i и связывает вершины S_t и S_p графа, если результат срабатывания перехода τ_i меняет состояние S_t автомата на S_p. Граф достижимости легко получается путем вычисления всех полных состояний S_t (начиная с начальной), которые ставятся в соответствие вершинам графа, и дуг, связывающих вершины, которые помечаются теми переходами автомата, которые вызывают изменение соответствующих полных состояний [16].

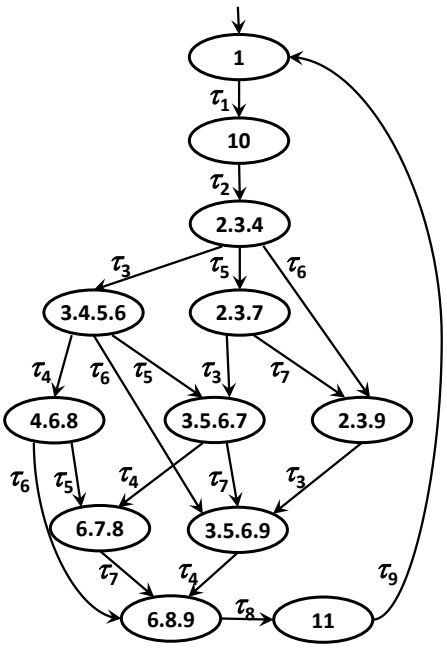

Рис. 1. Граф достижимых состояний

Граф достижимости приведенного выше параллельного алгоритма имеет 12 вершин (рис. 1), помеченных полными состояниями S_i. Из вершины P_g графа исходит дуга, заходящая в вершину P_h, если в исходном задании имеется переход $\tau_i = (\gamma_i^1, k_i^1) \rightarrow (\gamma_i^2, k_i^2)$, такой, что $\gamma_i^1 \subseteq S_g$ и $S_h = (S_g \setminus \gamma_i^1) \cup \gamma_i^2$). В общем случае граф достижимости является ориентированным мультиграфом, так как он может содержать кратные дуги, различающиеся метками переходов.

V. Сокращение графа достижимых состояний

Построение графа достижимости (в явном или неявном виде) всегда возможно для моделей с конечным пространством состояний, однако число разметок, составляющих граф достижимости, растет экспоненциально с ростом числа состояний. Основной причиной роста числа разметок является наличие параллельно выполняемых переходов автомата. В зависимости от причин возникновения параллельности в спецификации на проектирование системы проверка отношения реализации допускает или запрещает упорядочение параллельных состояний (и разметок с их участием).

В настоящее время известны подходы к сокращению размерности графа достижимости (числа вершин и дуг) без потери важной информации для решаемой задачи (обзор таких методов можно найти в работе [17]). Основным недостатком предложенных правил редукции графа является то, что их применимость ограничена относительно специфическими структурами. Наиболее подходящий подход (для рассматриваемой в настоящей работе задачи) к редукции графа достижимости основан на частичном упорядочении параллельных переходов (partial order reduction [18]). В основе методов сокращения графа лежит предположение о коммутативности асинхронных одновременно происходящих процессов. Такие методы применяется в методах формальной верификации сетей Петри, однако эти методы не учитывают информационное взаимодействие переходов, что неприемлемо при решении задачи генерации тестов на основе графа достижимых состояний.

Усечение графа достижимости параллельного автомата путем частичного упорядочения параллельных переходов основано на том, что порядок, в котором выполняются некоторые параллельные переходы, в общем случае не важен. Можно считать их выполняемыми одновременно. Это позволяет сократить число возможных полных состояний S_t автомата (и вершин графа).

Переход $\tau_i = (\gamma_i^1, k_i^1) \rightarrow (\gamma_i^2, k_i^2)$ срабатывает в состоянии $S_t(W_t)$ автомата, если $\gamma_i^1, \subseteq S_t$ и при текущих значениях входных сигналов конъюнкция $k_i^1 = 1$ на W_t. Пусть в состоянии $S_t(W_t)$ возможно срабатывание двух (и более) переходов $\tau_i = (\gamma_i^1, k_i^1) \rightarrow (\gamma_i^2, k_i^2)$ и $\tau_j = (\gamma_j^1, k_j^1) \rightarrow (\gamma_j^2, k_j^2)$, т. е. $\gamma_i^1, \gamma_j^1 \subseteq S_t$, $k_i^1 \wedge W_t \neq 0$, $k_j^2 \wedge W_t \neq 0$. Выполнение перехода τ_j не препятствует срабатыванию перехода τ_i, если значения переменных, упомянутых в конъюнкции k_i^1, совместимы с их значениями в конъюнкции k_j^2, т. е. установка значений переменных, обращающих конъюнкцию k_j^2 в 1, не приводит к тому, что получается $k_i^1 \neq 1$ (или $k_j^2 \wedge k_i^1 \neq 0$).

Пара параллельных переходов τi и τj являются совместимыми (коммутативными) в состоянии St (Wt), если они могут выполняться в любом порядке, приводя в итоге к одному и тому же полному состоянию St+1 автомата. Совместимость имеет место, если переходы τi и τj:

1) имеют совместимые входные условия: $ki1 \wedge kj1 \neq 0$;

2) не препятствуют срабатыванию друг друга: $kj2 \wedge ki1 \neq 0$ и $ki2 \wedge kj1 \neq 0$;

3) определяют совместимые изменения выходных сигналов: $k_i^2 \wedge k_j^2 \neq 0$.

Для сокращения числа достижимых полных состояний (и числа вершин графа достижимости) предлагается запускать совместимые переходы одновременно. Если в полном состоянии $S_t(W_t)$ автомата возможно срабатывание совместимых параллельных переходов $\tau_i = (\gamma_i^1, k_i^1) \rightarrow (\gamma_i^2, k_i^2)$ и $\tau_j = (\gamma_j^1, k_j^1) \rightarrow (\gamma_j^2, k_j^2)$, то они выполняются одновременно, переводя автомат в состояние $S_{t+1} = (S_t \setminus (\gamma_i^1 \cup \gamma_j^1)) \cup (\gamma_i^2 \cup \gamma_j^2)$ с установкой значений выходных переменных, обращающих $k_j^2 \vee k_i^2 = 1$. Это определение можно расширить на случай более двух переходов: набор переходов, разрешенных в некотором полном состоянии $S_t(W_t)$, может запускаться одновременно, если они взаимно совместимы.

VI. Построение сокращенного графа достижимых состояний

Если в процессе построения графа достижимости для очередной достигнутой маркировки возможно срабатывание пары (или более) параллельных переходов, не находящихся в отношении следования, то следующее полное состояние строится как результат одновременного срабатывания тех из этих переходов, которые являются совместимыми.

Предложенный в работе [16] алгоритм генерации графа достижимых состояний для параллельного автомата состоит в выполнении двух этапов: нахождения множества всех полных состояний автомата, задающих вершины графа, и построения отношения следования на их множестве, задающих дуги графа и их пометку переходами автомата. В настоящей работе предлагается сразу строить сокращенный граф достижимых состояний.

Для построения сокращенного графа достижимости состояний задается начальное состояние $S1 \subseteq S$ и множество T переходов $\tau i = (\gamma i1, ki1) \rightarrow (\gamma i2, ki2)$

автомата. Для каждого полученного полного состояния Sk:

– происходит просмотр всех τi переходов и формирование множества Tk тех из них, которые достижимы из Sk: τi ∈ Tk, если γi1 ⊆ Sk;

– из переходов множества Tk формируются максимальные подмножества Tkl совместимых переходов, совокупность которых представляет собой покрытие множества Tk;

– по каждому подмножеству Tkl совместимых переходов формируется полное состояние Skj: Skj = (Sk \ (∪ γi1/τi ∈ Tkl)) ∪ (∪ γi2/τi ∈ Tkl), которое объявляется новым достижимым полным состоянием и порождает вершину графа достижимости (если оно отлично от уже полученных);

– в граф достижимости вводится дуга из *k*-й в *j*–ю вершины, помеченная множеством переходов из T_k^l.

Этот процесс повторяется для каждого из вновь полученных полных состояний. Процесс заканчивается построением графа достижимых состояний, когда не получается новых полных состояний, отличных от уже полученных.

Усеченный граф достижимости для приведенного выше параллельного автомата приведен на рис. 2. Из описания автомата (и рис. 2), видно, что в состоянии S_3 = {2, 3, 4} возможно срабатывание трех переходов $\tau_3 = (2, -) \rightarrow (5.6, y_1)$; $\tau_5 = (4, \bar{x}_1) \rightarrow (7, \bar{y}_1)$ и $\tau_6 = (4, x_1) \rightarrow (9, y_2)$: T_3 = {τ_3, τ_5, τ_6}. Максимальными совместимыми подмножествами переходов являются T_3^1 = {τ_3, τ_5} и T_3^2 = {τ_3, τ_6}, они дают минимальное покрытие множества T_3. Срабатывание переходов из T_3^1 = {τ_3, τ_5} порождает новое полное состояние S_4 = {3, 5, 6, 7}, которое достижимо при срабатывании переходов τ_3 и τ_5 в любом порядке (в зависимости от сигналов, поступающих на входы автомата). Срабатывание переходов из T_3^2 = {τ_3, τ_6} порождает новое полное состояние S_5 = {3, 5, 6, 9}. В граф достижимости вводятся вершины P_4 и P_5 и две дуги: (P_3, P_4) и (P_3, P_5).

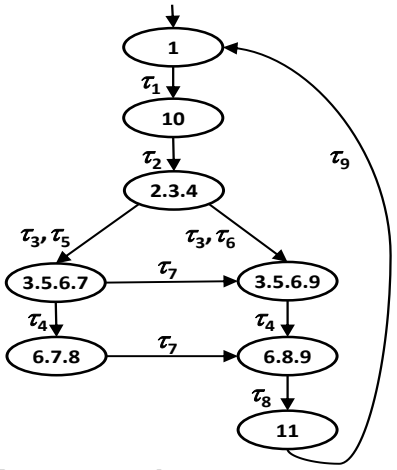

Рис. 2. Усеченный граф достижимых состояний

В состоянии S4 = {3, 5, 6, 7} возможно срабатывание двух переходов: τ4 = (3.5, x2) → (8, –) и τ7 = (7, $\bar{x}$2) → (9, –). Эти переходы не совместимы, соответственно максимальными совместимыми подмножествами переходов являются T41 = {τ4} и T42 = {τ7}, первое из них порождает новую вершину графа P6 = {6, 7, 8} и дугу (4, 6), второе – дугу (4, 5).

VII. Построение тестовых последовательностей по графу достижимости

Задача синтеза проверяющего теста заключается в построении конечного множества воздействий на систему, по реакциям на которые можно определить правильность ее функционирования. Искомые тесты представляются парами векторов: вектор тестового воздействия, компонентам которого соответствуют значения входных переменных из *X*, и вектор реакций анализируемой системы, компонентам которого соответствуют значения выходных переменных из *Y*.

Для того чтобы построить проверяющую последовательность, будем помечать дуги графа достижимости не только переходами, но и парой: значения входных переменных, вызывающих их срабатывание, и значения изменяемых ими переменных (рис. 3). В таком расширенном графе достижимости могут появляться кратные дуги, различающиеся присвоенными им метками.

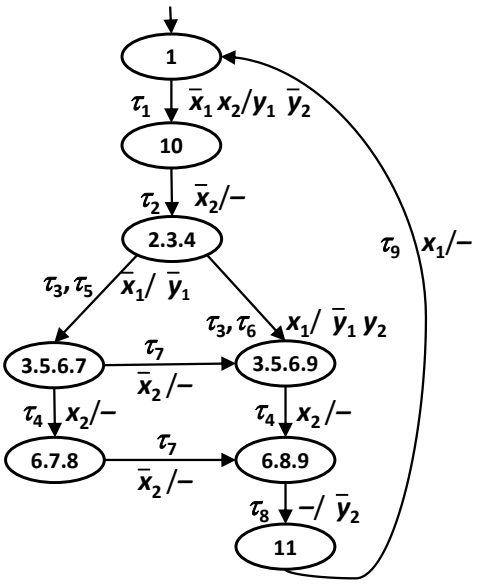

Рис. 3. Граф достижимости с метками условий переходов

Для решения задачи обхода графа достижимых состояний можно использовать один из известных методов обхода дуг ориентированного графа. По последовательности переходов, соответствующих меткам найденной последовательности дуг, можно определить входные стимулы тестовой последовательности для подачи на входы тестируемой реализации системы управления. Таким образом для каждого достижимого состояния системы определяются только те стимулы, которые заданы в реализуемой спецификации, т. е. тестирование

проводится только на заданной спецификацией области определения. Если в реализации допустимы какие-то другие стимулы, то это никак не влияет на процесс тестирования. Фактически это означает, что для заданной реализации исходного модельного алгоритма управления тестируется некоторая часть ее функциональности, заданная переходами по входным стимулам и состояниям модельного алгоритма, достижимым из начального состояния.

Обход дуг графа достижимости (рис. 3) задается следующим циклическим маршрутом, начинающимся и заканчивающимся в начальном состоянии 1:

1, τ_1, 10; τ_2, 2.3.4; τ_3, τ_5, 3.5.6.7; τ_4, 6.7.8; τ_7, 6.8.9; τ_8, 11; τ_9, 1; τ_1, 10; τ_2, 2.3.4; τ_3, τ_5, 3.5.6.7; τ_7, 3.5.6.9; τ_4, 6.8.9; τ_8, 11; τ_9, 1; τ_1, 10; τ_2, 2.3.4; τ_3, τ_6, 3.5.6.9; τ_4, 6.8.9; τ_8, 11; τ_9, 1.

При проверке соответствия между состояниями и выходными откликами схемной реализации и ее моделью, задаваемой параллельным автоматом, тестовая последовательность строится в виде четверок (S_i^b, X_i, S_i^e, Y_i), где S_i^b и S_i^e – начальное и конечное множества состояний, X_i – входной стимул, Y_i – изменение значений переменных при подаче на вход схемы входного набора X_i. При отображении тестовой последовательности начальное состояние S_i^b (кроме S_1^b) будем опускать, считая его равным S^e_{i-1}:

(1, $\bar{x}_1 x_2$, 10, $y_1 \bar{y}_2$), ($\bar{x}_2$, 2.3.4, –), ($\bar{x}_1$, 3.5.6.7, $\bar{y}_1$), (x_2, 6.7.8, –), ($\bar{x}_2$, 6.8.9, –), (–,11, $\bar{y}_2$), (x_1, 1, –), ($\bar{x}_1 x_2$, 10, $y_1 \bar{y}_2$), ($\bar{x}_2$, 2.3.4, –), ($\bar{x}_1$, 3.5.6.7, y_1), ($\bar{x}_2$, 3.5.6.9, –), (x_2, 6.8.9, –), (–,11, $\bar{y}_2$), (x_1, 1, –), ($\bar{x}_1 x_2$, 10, $y_1 \bar{y}_2$), ($\bar{x}_2$, 2.3.4, –), (x_1, 3.5.6.9, $y_1 y_2$), (x_2, 6.8.9, –), (–,11, y_2), (x_1, 1, –).

Следует заметить, что обход усеченного графа достижимых состояний позволяет получить только одну из последовательностей состояний, которая может быть реализована при функционировании анализируемой системы управления. Это происходит в силу агрегации входных воздействий на систему управления при ее тестировании, вызванной группированием параллельных переходов при доопределении частичного порядка на их множестве. Однако для корректных описаний параллельных автоматов этого достаточно, так как такой автомат при любом порядке срабатывания переходов перейдет в итоге к одному и тому же состоянию.

VIII. Заключение

Рассмотрена задача построения тестовой последовательности для системы с параллелизмом поведения, функционирование которой задано на языке параллельных автоматов. Тестовая последовательность находится в процессе обхода графа достижимых состояний автомата. Размер графа существенно зависит от степени параллельности переходов автомата. Предлагаемый метод построения сокращенного графа достижимых состояний параллельного автомата основан на предположении независимости срабатывания параллельно происходящих переходов автомата. Сокращение графа достигается доопределением частичного порядка на множестве переходов. Получаемое сокращение графа достижимости позволяет решать задачу построения тестов для сложных систем управления с параллелизмом поведения.

Литература

[1] Hoffman L. Talking Model-Checking Technology // Communications of the ACM. 2008. V. 51. № 07/08. P. 110–112.

[2] Tretmans J. Model based testing with labelled transition systems. Formal Methods and Testing: Lecture Notes in Computer Science (Springer). 2008. 4949. P. 1–38.

[3] Карпов Ю. Г. Model Checking: верификация параллельных и распределенных программных систем. СПб: БХВ-Петербург, 2010. 560 с.

[4] Закревский А. Д. Параллельные алгоритмы логического управления. Минск: Ин-т техн. кибернетики НАН Беларуси. 1999. 202 с.

[5] Lee D., Yannakakis M. Principles and methods of testing finite state machine – a survey // Proceedings of the IEEE. 1996. V. 84. № 8. P. 1090–1123.

[6] Бурдонов И. Б., Косачев А.С., Кулямин В.В. Неизбыточные алгоритмы обхода ориентированных графов. Детерминированный случай // Программирование. 2003. № 5. С. 11–30.

[7] Kanso B., Chebaro O. Compositional testing for FSM-based models // International Journal of Software Engineering & Applications (IJSEA). 2014. V. 5. № 3. P. 9–23

[8] Watanabe H., Kudoh T. Test Suite Generation Methods for Concurrent Systems based on Coloured Petri Nets // Proceedings of the 2nd Asia-Pacific Software Engineering Conference. 1995. P. 242–251.

[9] Zhu H., He X.D. A Methodology of Testing High-level Petri Nets // Information and Software Technology. 2002. V. 44. P. 473–489.

[10] Liu J., Ye1 X., Zhou J., Song X. I/O Conformance Test Generation with Colored Petri Nets // Applied Mathematics and Information Sciences. 2014. V. 8. № 6. P. 2695–2704.

[11] Ponce de Leon H., Delphine Longuet S.H. Model-based Testing for Concurrent Systems with Labeled Event Structures // Software Testing, Verification & Reliability. 2014. V. 24. № 7. P. 558–590.

12] Thimbleby H. The directed Chinese Postman Problem. Software Practice and Experience. 2003. Vol. 33. № 11. P. 1081–1096.

[13] Lee D., Yannakakis M. Principles and methods of testing finite state machine – a survey // Proceedings of the IEEE. 1996. Vol. 84. № 8. P. 1090–1123.

[14] Вельдер С. Э., Лукин М. А., Шалыто А. А., Яминов Б. Р. Верификация автоматных программ. СПб.: Наука. 2011. 244 с.

[15] Черемисинова Л. Д. Построение тестов полного перебора для оценки энергопотребления последовательностных схем // Информатика. 2017. № 4. С. 104–110.

[16] Поттосин Ю.В., Романов В.И., Черемисинова Л.Д. Верификация систем с параллелизмом поведения на основе графа достижимых состояний // Информатика. 2019. № 2 (16). С. 62–72.

[17] Karatkevich, A. Dynamic Analysis of Petri Net-based Discrete Systems. Springer-Verlag. 2007. Vol. 358. 166 p.

[18] *Lluch-Lafuente A., Edelkamp S., Leue S.* Partial Order Reduction in Directed Model Checking. Proceedings of the 9th International SPIN Workshop on Model Checking of Software. Springer-Verlag Berlin, Heidelberg.. 2002 April, 11–13. P. 112-127.

Testing Systems with Behavior Parallelism Based on a Reduced Reachability Graph

L.D. Cheremisinova

The United Institute of Informatics Problems of NAS of Belarus, Minsk,

cld@newman.bas-net.by

***Abstract* — The paper deals with the problem of verification of the functional correctness of reactive control systems with respect to their design specification. When solving the problem of implementing control devices, one has to deal with the parallelism present in control objects, which is displayed in the specification for control system designing. It is assumed that the description of the device operation is given in the language of parallel automata [4] and it is correct. The subject is to analyze the implementation of the control system for input-output conformance, which is performed by simulation of its behavior on the area of its desired operation.**

The test sequence generation is based on traversing the graph of reachable states of the parallel automaton. But the size of the reachability graph grows exponentially with the number of states and transitions. In addition, for systems with behavioral parallelism the degree of parallelism of the automaton has a very negative impact on the graph size and performance of testing because of necessity to enumerate all possible orders of transitions firing.

A method of reduction of the reachability graph for a parallel automaton is proposed, which is based on the assumption that the parallel automaton transitions are independent and may be executed in different order. The reduction of the reachability graph is achieved by extending the partial order on the set of transitions, which makes it possible to significantly reduce the number of considered states of the control system and, accordingly, the number of graph vertices and arcs.

The obtained reduction of reachability graphs of parallel automata allows us to solve problem of constructing tests for more complex control systems with parallelism of behavior.

***Keywords* — parallel automaton, verification, reachability graph, design specification.**

REFERENCES

[1] Hoffman L. Talking Model-Checking Technology // Communications of the ACM. 2008. V. 51. № 07/08. P. 110–112.

[2] Tretmans J. Model based testing with labelled transition systems. Formal Methods and Testing: Lecture Notes in Computer Science (Springer). 2008. 4949. P. 1–38.

[3] Karpov YU. G. Model Checking: verifikatsiya parallel'-nykh i raspredelennykh programmnykh sistem (Model Checking: Verification of Parallel and Distributed Software Systems). SPb.: BKHV-Peterburg, 2010. 560 p. (in Russian).

[4] Zakrevskij A. D. Parallel'nye algoritmy logicheskogo upravlenija (Parallel Logic Control Algorithms). Minsk: Institut tehnicheskoj kibernetiki Nacional'noj akademii nauk Belarusi, 1999. 202 p. (in Russian).

[5] Lee D., Yannakakis M. Principles and methods of testing finite state machine – a survey // Proceedings of the IEEE. 1996. V. 84. № 8. P. 1090–1123.

[6] Burdonov I. B, Kosachev A. S., Kuljamin V. V. Neizbytochnye algoritmy obhoda orientirovannyh grafov. Determinirovannyj sluchaj (Irredandant algorithms for traversal of directed graphs. The determinate case) // Programmirovanie. 2003. № 5. P. 11–30 (in Russian).

[7] Kanso B., Chebaro O. Compositional testing for FSM-based models // International Journal of Software Engineering & Applications (IJSEA). 2014. V. 5. № 3. P. 9–23

[8] Watanabe H., Kudoh T. Test Suite Generation Methods for Concurrent Systems based on Coloured Petri Nets // Proceedings of the 2nd Asia-Pacific Software Engineering Conference. 1995. P. 242–251.

[9] Zhu H., He X.D. A Methodology of Testing High-level Petri Nets // Information and Software Technology. 2002. V. 44. P. 473–489.

[10] Liu J., Ye1 X., Zhou J., Song X. I/O Conformance Test Generation with Colored Petri Nets // Applied Mathematics and Information Sciences. 2014. V. 8. № 6. P. 2695–2704.

[11] Ponce de Leon H., Delphine Longuet S.H. Model-based Testing for Concurrent Systems with Labeled Event Structures // Software Testing, Verification & Reliability. 2014. V. 24. № 7. P. 558–590.

12] Thimbleby H. The directed Chinese Postman Problem. Software Practice and Experience. 2003. Vol. 33. № 11. P. 1081–1096.

[13] Lee D., Yannakakis M. Principles and methods of testing finite state machine – a survey // Proceedings of the IEEE. 1996. Vol. 84. № 8. P. 1090–1123.

[14] Vel'der S. E., Lukin M. A., Shalyto A. A., Yaminov B. R. Verifikatsiya avtomatnykh programm (Verification of Automaton Programs). SPb.: Nauka. 2011. 244 p.

[15] Cheremisinova L. D. Postroyeniye testov polnogo perebora dlya otsenki energopotrebleniya posledovatel'nostnykh skhem (Test Generation for Power Consumption of Sequential Circuits) // Informatics. 2017. № 4. P. 104–110.

[16] Pottosin Yu. V., Romanov V. I., Cheremisinova L. D. Verifikatsiya sistem s parallelizmom povedeniya na osnove grafa dostizhimykh sostoyaniy (Verification of Systems with Behavior Parallelism on the Base of the Graph of Reachable States). Informatics. 2019. 2019. Vol. 16. № 2. P. 62–72 (in Russian).

[17] Karatkevich, A. Dynamic Analysis of Petri Net-based Discrete Systems. Springer-Verlag. 2007. Vol. 358. 166 p.

[18] *Lluch-Lafuente A., Edelkamp S., Leue S.* Partial Order Reduction in Directed Model Checking. Proceedings of the 9th International SPIN Workshop on Model Checking of Software. Springer-Verlag Berlin, Heidelberg.. 2002 April, 11–13. P. 112-127.