ИЗВЕСТИЯ ВЫСШИХ УЧЕБНЫХ ЗАВЕДЕНИЙ
Т. 56, № 9/2 ФИЗИКА 2013

УДК 519.713

С.А. ПРОКОПЕНКО, М.В. ЖИГУЛИН, А.В. КОЛОМЕЕЦ, Н.В. ЕВТУШЕНКО

СИНТЕЗ ДИАГНОСТИЧЕСКИХ ТЕСТОВ ДЛЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ПО СРЕЗАМ РАСШИРЕННОГО АВТОМАТА[1]

Предлагается метод построения диагностических тестов по модели расширенного автомата, которые могут быть использованы на этапе отладки программного обеспечения.

Ключевые слова: *расширенный автомат, диагностический тест.*

Синтез тестов для программного обеспечения является актуальной технической задачей. Вместе с тем, чтобы иметь возможность строить качественные тесты, необходима адекватная формальная модель. Одной из таких моделей является расширенный автомат [1].

Под *расширенным автоматом* понимается пятерка $M = (S, X, Y, T, V)$, где S – непустое конечное множество состояний автомата, X – непустое конечное множество входных символов (входной алфавит), Y – непустое конечное множество выходных символов (выходной алфавит), V – конечное, возможно пустое, множество контекстных переменных, T – множество переходов между состояниями из S.

Каждый переход t из T является семеркой (s, x, P, op, y, up, s'), где s и s' принадлежат множеству состояний S и являются начальным и конечным состояниями перехода; $x \in X$ есть входной символ, и $D_{\text{inp}}x$ обозначает множество входных векторов, компонентами которых являются значения параметров, соответствующих входному символу x (входные параметры); $y \in Y$ – выходной символ, и $D_{\text{out}}y$ обозначает множество выходных векторов, компонентами которых являются значения параметров, соответствующих выходному символу y (выходные параметры); P, op и up – функции, определенные над входными параметрами и контекстными переменными из V:

$P : D_{\text{inp}}x \times D_v \to$ {истина, ложь} – предикат, где D_v – множество контекстных векторов;

$op : D_{\text{inp}}x \times D_v \to D_{\text{out}}y$ – функция вычисления значения выходного параметра;

$up : D_{\text{inp}}x \times D_v \to D_v$ – функция обновления значения контекстной переменной.

В [2] предлагается подход к диагностике ошибок в расширенных автоматах на основе построения эквивалентного конечного автомата [3] и проверяющего теста, построенного классическим методом Василевского [4].

Однако если области определения входных параметров и контекстных переменных расширенного автомата не являются конечными, то эквивалентный конечный автомат построить нельзя. Более того, даже если области определения входных параметров и контекстных переменных расширенного автомата являются конечными, то эквивалентный конечный автомат может иметь тысячи состояний и миллионы переходов. С такой сложностью не справляются даже современные автоматизированные системы синтеза тестов. В этом случае предложено строить тесты на основе так называемых срезов расширенных автоматов [5, 6].

Под *срезом расширенного автомата* понимается автомат, из которого могут быть удалены переходы, предикаты, функции обновления значений контекстных переменных, вычисления значений выходных параметров и, кроме того, сами контекстные переменные или выходные параметры. Существуют четыре типа срезов расширенного автомата [6], которые имеют конечно автоматное поведение и сохраняют свойства достижимости и различимости состояний, требуемые для построения тестов. В данной работе рассматриваются только два таких среза.

Срез расширенного автомата с сохранением свойств достижимости (R-срез) используется для упрощения решения задачи достижимости в расширенных автоматах. Для построения данного среза из расширенного автомата требуется удалить все контекстные переменные, от которых не зависят предикаты переходов; сами переходы, функции обновления значений контекстных переменных которых зависят от указанных контекстных переменных; функции вычисления значений выходного параметра остальных переходов, зависящие от данных контекстных переменных, при этом выходному параметру присваивается любое допустимое значение. Построенный R-срез сохраняет свойства достижимости состояний исходного расширенного автомата и обладает важным свойством: если параметризованная входная последовательность переводит R-срез из состояния s

[1] Работа выполнена при частичной поддержке гранта Министерства образования и науки РФ ТЕМПЛАН № 8.4055.2011.

в состояние s', то данная последовательность переводит расширенный автомат из состояния s в состояние s'.

Для решения задачи различимости состояний расширенного автомата строится контекстно-свободный срез (*FSM*-срез), который не содержит контекстных переменных и выходных параметров. Для построения *FSM*-среза из расширенного автомата удаляются переходы, предикаты которых зависят только от контекстных переменных; предикаты переходов, зависящие от входных параметров и контекстных переменных, заменяются предикатами, зависящими от входных параметров; кроме того, удаляются все контекстные переменные и выходные параметры, а также соответствующие функции обновления значений контекстных переменных и вычисления значений выходных параметров. Если параметризованная входная последовательность переводит *FSM*-срез из состояния s в состояние s', то данная последовательность переводит расширенный автомат из состояния s в состояние s'. Если параметризованная входная последовательность различает состояния s и s' в *FSM*-срезе, то она также различает любые конфигурации с состояниями s и s' в расширенном автомате.

Таким образом, можно построить тест для расширенного автомата, не моделируя его поведения. Алгоритм построения проверяющего теста для полностью определенного и детерминированного расширенного автомата [1] на основе *R*- и *FSM*-срезов и множества проверяемых переходов представлен в [7]. Для каждого проверяемого перехода $t = (s, x, P, op, y, up, s')$ по *R*- или *FSM*-срезу строится параметризованная входная последовательность α, переводящая расширенный автомат из начального состояния в состояние s. С использованием *FSM*-среза строится множество параметризованных входных последовательностей $W_{s'}$, различающих состояние s' с остальными состояниями *FSM*-среза. В проверяющий тест заносятся последовательности $\alpha x W_{s'}$. Если состояние s не достижимо в *R*-срезе, то выбирается следующий проверяемый переход. Из проверяющего теста удаляются последовательности, являющиеся начальными отрезками других последовательностей. Результаты компьютерных экспериментов показали, что построенный вышеописанным образом проверяющий тест обнаруживает не только однократные и двукратные ошибки переходов, выходов и функций *op*, но и не менее 80 процентов однократных и двукратных предикатных ошибок и ошибок функции *up*.

Говорят, что на переходе $t = (s, x, P, op, y, up, s')$ при реализации произошла *выходная ошибка*, если параметризованный выходной символ отличается от параметризованного выходного символа перехода t. На переходе t произошла *ошибка перехода*, если финальное состояние перехода отличается от финального состояния s'перехода t. На переходе t произошла *предикатная ошибка*, если предикат перехода отличается от предиката P перехода t. На переходе t произошла *ошибка функции op* или *up*, если данные функции отличаются от соответствующих функций перехода t.

Однако проверяющий тест позволяет лишь обнаруживать ошибки в программных реализациях, но не позволяет указать, где именно произошла ошибка, т.е. локализовать ошибку. Поэтому для установления перехода, в котором имеет место ошибка, необходимо использовать диагностический тест.

Рассмотрим параметризованную последовательность $\alpha = a_1 a_2 \ldots a_n$ проверяющего теста и соответствующую ей параметризованную выходную последовательность $\beta = b_1 b_2 \ldots b_n$ расширенного автомата спецификации. Пусть под действием входной последовательности α расширенный автомат проходит цепочку состояний $s_0 \to s_1 \to \ldots s_{n-1} \to s_n$. Если на переходе из состояния s_i в состояние s_{i+1} под действием параметризованного символа a_{i+1} произошла выходная ошибка или ошибка функции *op*, то выходная реакция автомата в состоянии s_i на входной символ a_{i+1} будет отличаться от эталонной выходной реакции b_{i+1}. Поэтому для установления начального состояния перехода автомата, в котором произошла выходная ошибка или ошибка функции *op*, необходимо промоделировать расширенный автомат на параметризованной входной последовательности $a_1 a_2 \ldots a_i$.

Мутации переходов расширенного автомата, в которых происходят ошибки переходов, предикатов (при условии, что данные ошибки не влияют на свойство детерминированности автомата, т.е. простые ошибки) и функций *up* (если контекстная переменная не является выходным параметром), определяются сложнее, поскольку выходная реакция на данном переходе совпадает с эталонной реакцией, а сам факт ошибки проявляется при подаче оставшейся части тестовой последовательности. Поэтому предлагается каждый префикс $Pref_i(\alpha)$ тестовой последовательности $\alpha = a_1 a_2 \ldots a_n$, $i = 1, \ldots, n$, продлять множеством различающих последовательностей W_{s_i} того состояния s_i, в которое попадает расширенный автомат-спецификация (или соответствующий срез

этого автомата) из начального состояния под действием последовательности $Pref_i(\alpha)$. Если реакция реализации отличается от реакции спецификации на последовательностях из множества $Pref_i(\alpha)W_{s_i}$, то начальное состояние перехода, в котором произошла ошибка, получается путем моделирования расширенного автомата на последовательности $Pref_{i-1}(\alpha)$. Такой тест также позволяет обнаруживать переходы с выходными ошибками или ошибками функции *op*.

Компьютерные эксперименты по локализации ошибок в реализациях некоторых телекоммуникационных протоколов и реальных технических систем [7], поведение которых описывается расширенным автоматом, на основе построенного в [7] проверяющего теста, достроенного различающими множествами состояний до диагностического теста, дают следующие результаты. Проверяющий тест позволяет обнаружить все однократные/двукратные ошибки переходов, выходов, функций *op*, в то время как диагностический тест позволяет локализовать все однократные ошибки данного типа. Проверяющий тест также обнаруживает не менее 80 % однократных/двукратных простых предикатных ошибок и ошибок функции *up*, однако диагностический тест позволяет локализовать лишь однократные ошибки указанных типов. Для локализации ошибок большей кратности требуются дополнительные исследования.

СПИСОК ЛИТЕРАТУРЫ

1. Petrenko A., Boroday S., and Groz R. // IEEE TSE. – 2004. – V. 30. – No. 1. – P. 29–42.
2. El-Fakih H., Prokopenko S., Yevtushenko N., and Boshmann G. // Lecture Notes in Computer Science. – 2003. – V. 2644. – P. 197–210.
3. Гилл А. Введение в теорию конечных автоматов. – М.: Наука, 1966. – 272 с.
4. Василевский М.П. // Кибернетика. – 1973. – Т. 9. – Вып. 4. – С. 93–108.
5. Коломеец А.В., Михайлов Ю.В. // Вестник Томского государственного университета. – 2008. – Т. 3. – № 4. – С. 110–118.
6. El-Fakih H., Kolomeets A., Prokopenko S., and Yevtushenko N. // Lecture Notes in Computer Science. – 2008. – P. 308–317.
7. Коломеец А.В. Алгоритмы синтеза проверяющих тестов для управляющих систем на основе расширенных автоматов: дис. … канд. техн. наук. – Томск: Том. гос. ун-т, 2010. – 129 с.

Национальный исследовательский Томский государственный университет,
г. Томск, Россия
E-mail: ninayevtushenko@yahoo.com, maxzh81@gmail.com

Поступила в редакцию 15.07.13.

Прокопенко Светлана Анатольевна, к.т.н., доцент.
Жигулин Максим Владимирович, к.т.н., ведущ. инженер;
Коломеец Антон Владимирович, к.т.н., инженер;
Евтушенко Нина Владимировна, д.т.н., зав. кафедрой.

S.A. PROKOPENKO, M.V. ZHIGULIN, A.V. KOLOMEETS, N.V. YEVTUSHENKO

EFSM BASED METHOD FOR DERIVING TESTS FOR SOFTWARE DEBUGGING

Based on the model of an Extended Finite State Machine a method is proposed how to derive distinguishing tests which can be used for software debugging.

Keywords: *Extended Finite State Machine, distinguishing test.*