На правах рукописи

Коломеец Антон Владимирович

АЛГОРИТМЫ СИНТЕЗА ПРОВЕРЯЮЩИХ ТЕСТОВ ДЛЯ УПРАВЛЯЮЩИХ СИСТЕМ НА ОСНОВЕ РАСШИРЕННЫХ АВТОМАТОВ

05.13.01 – Системный анализ, управление и обработка информации (в отраслях информатики, вычислительной техники и автоматизации)

Автореферат
диссертации на соискание ученой степени кандидата технических наук

Томск – 2010

Работа выполнена на кафедре информационных технологий в исследовании дискретных структур ГОУ ВПО «Томский государственный университет»

Научный руководитель: доктор технических наук, профессор
Евтушенко Нина Владимировна

Официальные оппоненты: доктор технических наук, профессор
Матросова Анжела Юрьевна

кандидат технических наук, доцент
Громаков Евгений Иванович

Ведущая организация: Институт вычислительного моделирования СО РАН,
г. Красноярск

Защита состоится 14 октября 2010 года в 10.30 на заседании диссертационного совета Д 212.267.12 при ГОУ ВПО «Томский государственный университет» по адресу: 634050, г. Томск, пр. Ленина, 36, II уч. корпус, ауд. 212 б.

С диссертацией можно ознакомиться в Научной библиотеке ГОУ ВПО «Томский государственный университет» по адресу: 634050, г. Томск, пр. Ленина, 34а.

Автореферат разослан 10 сентября 2010 г.

Ученый секретарь
диссертационного совета
к. ф.-м. н., доцент

П.Ф. Тарасенко

ОБЩАЯ ХАРАКТЕРИСТИКА РАБОТЫ

Актуальность проблемы. Построение качественных проверяющих тестов для дискретных управляющих систем, в частности, для текоммуникационных протоколов, является актуальной технической задачей. Для построения проверяющего теста с гарантированной полнотой необходимо иметь, в первую очередь, адекватную математическую модель поведения системы и ошибки (неисправности). Классические хорошо изученные модели, такие как конечный автомат, полуавтомат, входо-выходной полуавтомат имеют слишком большое число состояний (и переходов), что затрудняет их использование для решения практических задач. Кроме того, эти модели описывают не все аспекты поведения управляющей системы, например, не описывают ситуации, когда область определения некоторого параметра (например, временной переменной) бесконечна. Поэтому в настоящее время активно исследуется вопрос о построении тестов с гарантированной полнотой на основе более компактных моделей, одной из которых является расширенный автомат. Модель расширенного автомата достаточно часто используется при описании телекоммуникационных протоколов, в частности, расширенный автомат достаточно просто построить по описанию поведения в языке SDL или If или UML. Однако практически все методы синтеза тестов для расширенных автоматов доставляют проверяющие тесты, полнота которых остается неизвестной, и соответственно разработка методов синтеза проверяющих тестов с гарантированной полнотой для расширенных автоматов является актуальной задачей.

Цель работы. Целью данной работы является исследование моделей ошибок в расширенных автоматах, анализ возможности применения для этой модели известных конечно автоматных методов, и разработка на их основе алгоритмов синтеза проверяющих тестов с гарантированной полнотой для расширенных автоматов.

Методы исследования. Для реализации поставленной цели в работе используются средства и методы дискретной математики, в частности, методы теории автоматов. Оценка качества тестов, построенных предложенными алгоритмами, производится с помощью компьютерных экспериментов.

Научная новизна работы состоит в следующем.

1. Предложено понятие мутационного расширенного автомата, посредством которого можно более компактно описать множество ошибок заданного класса в расширенном автомате. Показано, каким образом можно использовать конечно автоматные методы при построении проверяющих тестов по мутационному расширенному автомату.

2. Предложены три конечно автоматных среза расширенного автомата и алгоритмы построения проверяющего теста с гарантированной полнотой на их основе.

3. Исследованы ошибки в программных реализациях и установлено соответствие между программными ошибками некоторых классов и ошибками в соответствующем расширенном автомате. Проведены компьютерные эксперименты, показывающие, что проверяющие тесты, построенные по расширенному автомату, можно использовать для обнаружения определенных ошибок в программных реализациях, в частности, в программных реализациях телекоммуникационных протоколов и технических систем.

Достоверность полученных результатов. Положения и выводы, формулируемые в диссертации, доказываются с применением аппарата дискретной математики. Эффективность предложенных методов подтверждается посредством компьютерных экспериментов.

Практическая ценность. Результаты работы могут быть использованы при построении проверяющих тестов для дискретных управляющих систем, поведение которых описано расширенными автоматами, в частности, для телекоммуникационных протоколов, поведение которых описано в языках SDL и If, а также при синтезе тестов для программных реализаций, для которых можно построить соответствующий расширенный автомат. Разработанный программный комплекс для синтеза проверяющих тестов на основе расширенных автоматов может быть также использован для проведения различных компьютерных экспериментов с расширенными автоматами.

Реализация полученных результатов. Исследования, результаты которых изложены в диссертации, проводились в рамках следующих проектов:

1. Проект TAROT в рамках 6^{й} рамочной программы ЕС «Мобильность молодых ученых», 2003 - 2007 гг.

2. НИР «Разработка математических и программных средств обеспечения надежного и безопасного доступа к электронным ресурсам коллективного пользования» (в рамках инновационного проекта ТГУ), 2006 - 2007 гг.
3. НИР «Проведение прикладных и проблемно-ориентированных поисковых исследований в области информационно-телекоммуникационных систем с участием научных организаций Франции (шифр заявки «2009-04-1.4-00-02-003»)», госконтракт №02.514.12.4002 от 09.06.2009 - 2010 гг.

Основные положения, выносимые на защиту.

1. Модель мутационного расширенного автомата, позволяющая компактно описать все ошибочные реализации заданного класса, и алгоритм построения проверяющих тестов для такой модели неисправности. Модель расширенного мутационного автомата позволяет строить эквивалентный мутационный конечный автомат единым образом для всех классов ошибок.
2. Метод построения конечно автоматного среза расширенного автомата, который имеет не больше состояний, чем исходный расширенный автомат, и алгоритм построения по такому срезу проверяющего теста с гарантированной полнотой.
3. Установленное соответствие между программными ошибками некоторых классов и ошибками в расширенном автомате. Проведенные компьютерные эксперименты показали эффективность тестирования таких программных ошибок с помощью тестов, построенных на основе соответствующего расширенного автомата.

Апробация работы. Все теоретические и практические результаты, составившие основу диссертационной работы, по мере их получения обсуждались на семинаре кафедры информационных технологий в исследовании дискретных структур радиофизического факультета ТГУ. Кроме того, результаты работы докладывались на следующих научных конференциях: Российской конференции с международным участием «Новые информационные технологии в исследовании дискретных структур» (Томск, 2003 и 2008, Иркутск, 2004, Шушенское, 2006), международной научной студенческой конференции «Студент и научно – технический прогресс» (Новосибирск, 2004 и 2006), на международной конференции по

тестированию программного обеспечения, ICST'2008, (Лиллехаммер, Норвегия, 2008)

Структура и объем работы. Диссертация состоит из введения, 4 глав, заключения и списка используемой литературы. Диссертация содержит 11 рисунков и 3 таблицы. Объем диссертации составляет 108 страницы, в том числе: титульный лист - одна страница, оглавление – две страницы, основной текст – 96 страницы, библиография из 88 наименований – 11 страницы, приложение – 21 страниц.

Публикации. По результатам проведенных исследований опубликовано 12 статей в научных журналах, докладах и тезисах докладов на конференциях различного уровня. Работы [1] и [2] опубликованы в изданиях, входящих в список ВАК.

СОДЕРЖАНИЕ РАБОТЫ

Во **введении** дается общая характеристика работы, обосновывается актуальность исследований, определяется тематика и формулируется цель работы, кратко излагаются основные задачи и результаты, выносимые на защиту.

Первая глава посвящена основным понятиям и обозначениям, относящимся к конечным и расширенным автоматам, и, кроме того, содержит обзор известных методов по синтезу проверяющих тестов для расширенных автоматов. Под *конечным автоматом* (или просто) автоматом понимается пятёрка $\mathcal{S} = (S, I, O, T_S, s_0)$, где S - непустое конечное множество состояний с выделенным начальным состоянием s_0, I - непустое конечное множество входных символов, называемое входным алфавитом, O - непустое конечное множество выходных символов, называемое выходным алфавитом, $T_S \subseteq I \times S \times S \times O$ - отношение поведения. В данной работе мы рассматриваем инициальные автоматы, то есть автоматы с фиксированным начальным состоянием; такие автоматы описывают поведение систем, обладающих сигналом сброса. Инициальному автомату можно сопоставить специальную словарную функцию, которая описывает отображение входных слов (слов во входном алфавите) в выходные слова. Если каждому входному слову соответствует единственное выходное слово, то автомат называется *детерминированным*: в противном случае автомат называется *недетерминированным*.

Формально под *расширенным автоматом* M понимается пятерка (S, X, Y, T, V), где S - непустое конечное множество состояний

автомата, X - непустое множество входных символов, называемое входным алфавитом, Y - непустое множество выходных символов, называемое выходным алфавитом, V - конечное, возможно, пустое множество контекстных переменных, T - множество переходов между состояниями из S.

Каждый переход t из множества T есть семёрка (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 $D_{inp}x$ обозначает множество входных векторов, компонентами которых являются значения параметров, соответствующих входному символу x (далее *входные параметры*); $y \in Y$ - выходной символ, и $D_{out}y$ обозначает множество выходных векторов, компонентами которых являются значения параметров, соответствующих выходному символу y (далее *выходные параметры*); P, op и up - функции, определенные над входными параметрами и контекстными переменными из V:

- P: $D_{inp}x \times D_V \rightarrow$ {Истина, Ложь} - предикат, где D_V - множество контекстных векторов, то есть векторов, компонентами которых являются значения контекстных переменных;

- op : $D_{inp}x \times D_V \rightarrow D_{out}y$ - функция вычисления выходного параметра;

- up : $D_{inp}x \times D_V \rightarrow D_V$ - функция вычисления значения контекстной переменной.

Известно, что если область определения контекстных переменных и входных параметров конечна, то по расширенному автомату можно построить эквивалентный конечный автомат, который реализует ту же словарную функцию. В общем случае эквивалентный конечный автомат строится путем моделирования поведения расширенного автомата на параметризированных входных последовательностях. Состояниями конечного автомата являются пары «состояние, вектор значений контекстных переменных», которые называются конфигурациями, а входными символами – пары «входной символ, вектор значений входных параметров».

Раздел 1.5 первой главы содержит краткий обзор известных результатов по синтезу тестов для расширенных автоматов. В приведенном обзоре отмечается, что в основном при синтезе тестов по расширенному автомату используются те же критерии, что и при тестировании программного обеспечения. Как показывают

проведенные компьютерные эксперименты, такие тесты не являются достаточно качественными при проверке управляющей части расширенного автомата. Методы синтеза тестов, основанные на различимости конфигураций, доставляют проверяющие тесты, полнота которых, вообще говоря, не известна, так же как и методы, основанные на частичном моделировании расширенного автомата. Причиной, в частности, является тот факт, что не известны необходимые и достаточные условия эквивалентности (или различимости) двух расширенных автоматов. Методы построения проверяющих тестов по соответствующему мутационному конечному автомату, имеют два недостатка. Во-первых, для каждого класса ошибок определяется специальный набор правил, по которым строится мутационный конечный автомат. Во-вторых, как показывают компьютерные эксперименты, даже для одиночных ошибок переходов в расширенном автомате сложность тестов, построенных по мутационному конечному автомату, практически не отличается от сложности тестов, построенных по модели «черного ящика», которые являются очень длинными. Абстракции и срезы для расширенных автоматов, активно используемые при верификации и тестировании программ, практически не используются при синтезе проверяющих тестов для расширенных автоматов. Не известна взаимосвязь между реальными ошибками (например, в программных реализациях) и ошибками в расширенном автомате. Перечисленные выше задачи частично решаются в настоящей диссертационной работе.

При построении тестов с гарантированной полнотой необходимо ввести модель неисправности. Во **второй главе** рассматриваются модели ошибок для расширенного автомата. Модель неисправности $<\mathcal{M}, \cong, Sub(\mathcal{MM})>$, как обычно, содержит три компоненты, а именно, эталонное поведение, которое представляется детерминированным полностью определенным расширенным автоматом (спецификацией $\mathcal{M}$), отношение конформности или соответствия, и область неисправности, то есть множество проверяемых систем, поведение каждой из которых также описано расширенным автоматом. В данной работе отношением конформности является отношение эквивалентности $\cong$, и область неисправности $Sub(\mathcal{MM})$ совпадает с множеством всех полностью определенных подавтоматов специального мутационного автомата $\mathcal{MM}$. Посредством моделирования по мутационному расширенному автомату строится

конечный мутационный автомат. Если такой конечный автомат существует, то задача синтеза проверяющих и диагностических тестов сводится к задаче синтеза таких тестов по мутационному конечному автомату, то есть по модели неисправности $<A_M, \cong, Sub(A_{MM})>$.

В диссертации рассматриваются неисправности переходов, выходов и функций, вычисляющих значения контекстных переменных и выходных параметров в расширенном автомате. Под ошибкой выхода понимается изменение в результате ошибки выходного символа перехода. Ошибкой перехода называется такая ошибка, в результате, которой изменяется конечное состояние перехода. В результате ошибки предиката изменяется предикатное выражение. Например, в некотором предикате знак равенства может замениться на знак неравенства. Ошибка присвоения изменяет функции вычисления значений выходных параметров и/или функции вычисления новых значений контекстных переменных на некотором переходе. В данной работе мы ограничиваемся простыми ошибками присвоения, т.е. ошибками, когда знак алгебраической операции изменяется на знак другой операции, например, вместо знака сложения ошибочно использован знак вычитания. Для каждого класса ошибок строится модель неисправности без перехода к конечным автоматам.

После моделирования эталонного и мутационного автоматов, получается модель неисправности для конечных автоматов, для которой известны методы синтеза проверяющих тестов с гарантированной полнотой. Однако, как известно, длина полного теста относительно такой модели неисправности, существенно зависит от степени недетерминизма мутационного автомата; поэтому мы предлагаем разбивать множество переходов автомата-спецификации на классы и строить мутационный автомат для каждого класса выделенных переходов отдельно, тем самым, позволяя строить достаточно короткие тесты для каждого мутационного автомата. Можно предложить следующий алгоритм разбиения множества переходов конечного автомата на подмножества при построении проверяющих тестов по мутационному автомату.

Алгоритм 1 построения проверяющего теста на основе мутационного автомата.

Вход: Полностью определенный детерминированный связный конечный автомат $M = (M, I, O, T', s_0)$, который определяет возможные

переходы в мутационном автомате; приведенная форма $\mathcal{S}$ автомата (M, I, O, T', s_0).

Выход: Полный проверяющий тест относительно модели неисправности $(\mathcal{S}, \cong, \cup Sub(\mathcal{MM}))$

Полагаем $J := 1$, $B := M \times I$.

Шаг 1. Если множество B не является пустым, то в множество B_J заносим максимальное число пар $(m, i) \in B$ таких, что частичный подавтомат автомата $\mathcal{M}$ после удаления всех переходов, соответствующих этим парам, остается связным.

J++

Если множество $B = \varnothing$, то Шаг 2.

Шаг 2. Пусть $\pi = \{B_1, \ldots, B_K\}$ есть разбиение множества переходов расширенного автомата на классы.

$J := 1$

Пока $J \le K$

Строим мутационный автомат $\mathcal{MM}$: для каждой пары $(m, i) \in B_J$ добавляем все переходы $(m, i, o, m'), o \in O, m' \in m$,

Строим полный проверяющий тест TS_J относительно модели $<\mathcal{S}, \cong, Sub(\mathcal{MM})>$;

J++;

Шаг 3. $TS = \cup TS_J$.

Из множества TS удаляются последовательности, которые являются собственными начальными отрезками других тестовых последовательностей.

Конец.

Теорема 1. Проверяющий тест TS, доставляемый алгоритмом 2.2, является полным относительно модели неисправности $(\mathcal{S}, \cong, \cup Sub(\mathcal{MM}))$.

По аналогии с конечным автоматом можно построить систему подмножеств переходов при построении проверяющих тестов по мутационному расширенному автомату. Однако, в отличие от конечного автомата, в расширенном автомате проверка на связность осуществляется следующим образом:

1. Строится мутационный расширенный автомат, в котором на выделенные переходы вносятся все возможные ошибки заданного класса.

2. По мутационному расширенному автомату строится эквивалентный конечный автомат.

3. Полученный эквивалентный конечный автомат проверяется на связность.

Проведенные компьютерные эксперименты по построению проверяющих тестов с гарантированной полнотой на основе мутационного расширенного автомата показывают, что в большинстве случаев мутационные конечные для реальных телекоммуникационных протоколов содержат достаточно большое число состояний, входных и выходных символов. Поэтому тесты для таких расширенных автоматов в ряде случаев не удается построить описанным выше методом.

В **третьей главе** мы предлагаем для построения проверяющих тестов использовать срезы (упрощенные копии) расширенного автомата с конечно автоматным поведением, число состояний которых сравнимо с числом состояний исходного расширенного автомата. Предложены алгоритмы построения *R*- и FSM-срезов путем удаления контекстных переменных, переходов и выходных параметров сохраняя при этом как можно больше возможностей для достижения и различения состояний в срезе. Число состояний в таких срезах не превышает число состояний исходного расширенного автомата, причем FSM-срез имеет конечно автоматное поведение, то есть для такого среза можно построить проверяющий тест с гарантированной полнотой хорошо известными автоматными методами.

Для упрощения решения задачи достижимости в расширенном автомате, мы предлагаем построить срез $Slice_R(M)$, который сохраняет свойства достижимости исходного расширенного автомата M. Мы называем такой срез *R-срезом* и для его построения предлагаем следующий алгоритм.

Алгоритм 2 построения *R*-среза расширенного автомата

Вход: Расширенный автомат M

Выход: Срез $Slice_R(M)$ автомата M

Шаг 1: Переменная v_i удаляется из множества контекстных переменных V, если ни одна из свободных переменных любого предиката расширенного автомата не зависит от переменной v_i. Другими словами, удаляются все контекстные переменные, не влияющие на выполнимость предикатов. Пусть $V' \subseteq V$ есть сокращенное множество контекстных переменных.

Шаг 2: Если на некотором переходе функция определения значения контекстной переменной зависит от некоторой переменной

из множества $V \backslash V'$, то данное соотношение удаляется из расширенного автомата.

Шаг 3: Если существует переход в расширенном автомате $\mathcal{M}$, на котором функция определения выходного параметра зависит от контекстной переменной из множества $V \backslash V'$, то данное соотношение удаляется из расширенного автомата. Соответствующему выходному параметру присваивается любое допустимое значение. **Конец.**

Удаление контекстных переменных, от которых не зависит ни один из предикатов расширенного автомата, не влияет на достижимость состояний. Поэтому справедливо следующее утверждение. Пусть $\mathcal{M}$ - расширенный автомат и его срез $Slice_R(\mathcal{M})$ получен по алгоритму 2.

Теорема 2. Если параметризованная входная последовательность α переводит расширенный автомат $Slice_R(\mathcal{M})$ из начального состояния s в состояние s', то последовательность α также переводит автомат M из начального состояния s в состояние s'. Более того, если под действием параметризированного входного символа $(x,\boldsymbol{\rho})$ в автомате $Slice_R(\mathcal{M})$ выполним переход $t = (s,x,P,op',y',up',s')$, то под действием параметризированного входного символа $(x,\boldsymbol{\rho})$ выполним соответствующий переход в автомате M.

Для того чтобы упростить проблему и достижимости, и различимости состояний в расширенном автомате, мы предлагаем построить специальный срез расширенного автомата с конечно автоматным поведением, то есть срез $Slice_{FSM}(\mathcal{M})$, который не содержит контекстных переменных и выходных параметров.

Идея построения среза состоит в удалении переходов, на которых предикат зависит от контекстных переменных, т.е. переходов, выполнение которых зависит от значений контекстных переменных. Тем не менее, некоторые переходы, имеющие предикаты, зависящие от контекстных переменных можно сохранить, воспользовавшись следующим свойством. Пусть, например, P есть дизъюнкция предикатов P_1 и P_2, и предикат P_1 не зависит от контекстных переменных. Тогда переход с предикатом P будет возбужденным, если для заданных значений входных параметров предикат P_1 принимает значение «ИСТИНА». В общем случае такая замена предиката возможна, если предикат P можно представить в виде суперпозиции предикатов P_1 и P_2, $P = f(P_1, P_2)$, из которых предикат P_1 зависит только от входных параметров, и $f(1, 0) = f(1, 1) = 1$. В этом случае

предикат P можно заменить предикатом P_1. При такой замене переход, возбужденный при наличии предиката P_1, останется возбужденным и для предиката P.

Алгоритм 3 построения FSM-среза расширенного автомата

Вход: Расширенный автомат $\mathcal{M}$

Выход: Конечно автоматный срез $Slice_{FSM}(\mathcal{M})$ автомата $\mathcal{M}$

Шаг 1: Из расширенного автомата M удаляются все переходы, на которых предикат зависит только от контекстных переменных. Если предикат P можно представить в виде композиции предикатов P_1 и P_2, $P = f(P_1, P_2)$, из которых предикат P_1 зависит только от входных параметров и $f(1, 0) = f(1, 1) = 1$, то предикат P заменяется предикатом P_1.

Шаг 2: Из полученного расширенного автомата удаляются все контекстные переменные и функции вычисления контекстных переменных (up).

Шаг 3: Из полученного расширенного автомата удаляются все выходные параметры и соответствующие функции вычисления выходных параметров (op). Полученный автомат обозначается $Slice_{FSM}(\mathcal{M})$. **Конец**.

Как обычно, для различения двух *контекстно свободных* расширенных автоматов, т.е. автоматов без контекстных переменных, строится различающий автомат. Такой различающий автомат практически совпадает с пересечением автоматов за исключением того, что содержит специальное состояние и специальный выходной символ, отмечающие ситуацию, когда сравниваемые автоматы имеют различное поведение.

Различающим автоматом $(\mathcal{M}_1/s \otimes \mathcal{M}_2/q)/(s,q)$ контекстно-свободных расширенных автоматов $\mathcal{M}_1/s$ и $\mathcal{M}_2/q$ называется наибольший связный подавтомат расширенного автомата $\langle (S\times Q)\cup fail, X, Y\cup fail, T\rangle$, где S – множество состояний автомата $\mathcal{M}_1$, Q – множество состояний автомата $\mathcal{M}_2$, $fail \notin S \times Q$ - специальное состояние и специальный выходной символ, X и Y – входной и выходной алфавиты, на которых определены автоматы $\mathcal{M}_1$ и $\mathcal{M}_2$. Множество переходов различающего автомата определяется следующим образом. Для каждой пары состояний $(s, q) \in S \times Q$ и для каждой пары переходов из состояний s и q по входному символу x, который определен в каждом из состояний s и q, с предикатами P и R, проверяем, является предикат $(P \ \& \ R)$ выполнимым, то есть проверяем, существует ли набор значений

входных параметров, на котором предикат (P & R) принимает значение «Истина». Если такой набор ρ существует и выходной символ b на этих переходах один и тот же, то в автомате $(M_1/s \otimes M_2/q)/(s,q)$ есть переход $(s, q) \rightarrow (x, \rho)/ b \rightarrow (s', q')$. Если выходные символы на переходах из состояний s и q по входному символу x, с предикатами P и R, не совпадают, то в различающем автомате существует переход $(s, q) \rightarrow (x, \rho)/fail \rightarrow fail$.

На основе R- и FSM-срезов можно предложить следующий алгоритм построения проверяющего теста для расширенного автомата.

Алгоритм 4 построения проверяющего теста для проверки ошибок переходов/выходов на выделенном подмножестве переходов расширенного автомата

Вход: Расширенный автомат M и подмножество E переходов этого автомата, подлежащих проверке, число k, определяющее длину параметризированной входной последовательности, которая будет строиться по R-срезу $Slice_R(M)$

Выход: Тест, проверяющий наличие одиночных ошибок переходов/выходов на подмножестве E переходов (если такой тест можно построить на основе R-среза и среза $Slice_{FSM}(M)$)

Шаг 1: Строятся FSM-срез $Slice_{FSM}(M)$ и R-срез $Slice_R(M)$, $TS = \varnothing$.

Шаг 2: Рассматривается каждый переход $t = (s, x, P, op, y, up, s')$ из множества E. Если в FSM-срезе $Slice_{FSM}(M)$ состояние s достижимо из начального состояния по параметризированной входной последовательности α и существует переход из состояния s под действием входного символа x, то αx заносится в тест TS, и Шаг 3.

Если состояние s недостижимо в FSM-срезе, то посредством моделирования R-среза проверяется, можно ли построить последовательность α длины k, переводящую R-срез из начального состояния в состояние s, после которой выполним переход t. Последовательность αx заносится в тест TS, и Шаг 3. Если состояние s недостижимо в R-срезе по последовательности длины k, то переходим к проверке следующего перехода.

Шаг 3: На основе алгоритма 3 проверяется, с какими состояниями различимо состояние s' в FSM-срезе и строится множество W_s различающих последовательностей. В тест TS заносятся последовательности $\alpha x W_s$.

Шаг 4: Если рассмотрены все переходы из множества E, то из множества TS удаляются последовательности, которые являются

собственными начальными отрезками других тестовых последовательностей. **Конец.**

Предложенный в предыдущем разделе метод гарантированно обнаруживает в расширенном автомате только одиночные ошибки переходов/выходов. Поскольку известно, что тесты, построенные для обнаружения одиночных неисправностей, обычно обнаруживают и много других неисправностей, то мы провели компьютерные эксперименты, чтобы оценить полноту построенных тестов относительно других классов ошибок. Компьютерные эксперименты показали, что для двойных ошибок переходов/выходов тест также практически всегда оказывается полным; соответственно, достаточно реально предположение, что тест будет полным и для большего числа ошибок переходов-выходов. Хотя в работе не гарантируется полнота построенного теста для классов предикатных ошибок и ошибок присвоения, однако, согласно проведенным компьютерным экспериментам, построенный тест оказался достаточно хорошим для обнаружения, как простых предикатных ошибок, так и для обнаружения простых ошибок присвоения.

Другой возможностью построения проверяющих тестов с гарантированной полнотой для расширенного автомата является его частичное моделирование. В данной работе при моделировании расширенного автомата мы ограничиваем число состояний соответствующего конечного автомата; все остальные переходы в конечном автомате полагаются неопределенными. В этом случае полный проверяющий тест строится для частичного, возможно неприведенного конечного автомата. Были проведены компьютерные эксперименты, чтобы посмотреть зависимость полноты теста от числа промоделированных состояний. Эксперименты показали, что при ограничении числа состояний конечного автомата в два раза полнота теста составляет примерно 80%.

Модель расширенного автомата по семантике очень близка к программной реализации. В **четвертой главе** устанавливается соответствие между некоторыми классами ошибок в программных реализациях и ошибками в расширенных автоматах. Поскольку мы не смогли найти программный продукт, позволяющий по программной реализации строить расширенный автомат, по которому далее строится тест, то для проверки полноты тестов, построенных алгоритмами, предложенными в диссертации, был разработан

генератор программных реализаций по текстовому описанию расширенного автомата. Генератор позволяет автоматически вносить ошибки заданного класса в генерируемую программную реализацию. Компьютерные эксперименты показали, что для ряда ошибок в программных реализациях тесты, построенные на основе расширенного автомата, обнаруживают до 80% внесенных ошибок.

В **заключении** формулируются основные результаты работы.

1. Предложена модель мутационного расширенного автомата, которая позволяет единым образом описывать наиболее традиционные ошибки, рассматриваемые в расширенных автоматах. В качестве предикатных ошибок и ошибок присвоения рассматриваются достаточно простые ошибки, которые, как показывает проведенный анализ, достаточно полно описывают ошибки, в частности, в программных реализациях телекоммуникационных протоколов.

2. Предложен алгоритм построения проверяющего теста на основе мутационного расширенного автомата путем разбиения множества переходов таким образом, чтобы соответствующий мутационный автомат имел как можно больше детерминированных переходов, и соответственно позволял строить достаточно качественные проверяющие тесты полиномиальной длины (в зависимости от размеров автомата).

3. Предложены алгоритмы для построения срезов (упрощенных версий) расширенного автомата, два из которых имеют не больше состояний, чем исходный расширенный автомат, и сохраняют свойства достижимости и различимости состояний. На основе срезов расширенного автомата предложен алгоритм построения проверяющего теста, обнаруживающего одиночные ошибки переходов/выходов на заданном множестве переходов расширенного автомата. Как показывают проведенные компьютерные эксперименты, тесты, построенные этим методом, обнаруживают и другие, в том числе, кратные предикатные ошибки и ошибки присвоения на переходах расширенного автомата.

4. Установлено соответствие между программными ошибками некоторых классов и ошибками в расширенном автомате; проведенные компьютерные эксперименты показали, что такие программные ошибки могут быть обнаружены тестами, построенными на основе соответствующего расширенного автомата.

Разработан программный комплекс для синтеза проверяющих тестов на основе расширенных автоматов; разработанный комплекс, в частности, может быть использован для проведения различных компьютерных экспериментов с расширенными автоматами.

Список публикаций по теме диссертации

1. Громов М.Л., Евтушенко Н.В., *Коломеец А.В.* К синтезу условных тестов для недетерминированных автоматов // Программирование. – 2008. – № 6. – С. 1–11.

2. Жигулин М.В., *Коломеец А.В.* Оценка полноты проверки при пассивном тестировании на основе автоматной модели // Известия ТПУ. – 2009. – Т. 314, № 5. – С. 225–228.

3. *Коломеец А.В.*, Прокопенко С.А. Метод синтеза диагностических тестов для расширенных конечных автоматов // Вестник ТГУ. Приложение. – 2003. – № 6. – С. 174–177.

4. *Коломеец А.В.* Метод синтеза проверяющих тестов для расширенных конечных автоматов // Студент и научно-технический прогресс. Информационные технологии : материалы XLII международной научной студенческой конференции. – Новосибирск, 2004. – С. 174–176.

5. Громов М.Л., *Коломеец А.В.*, Евтушенко Н.В. Синтез диагностических тестов для автоматных сетей // Вестник ТГУ. Приложение. – 2004. – № 9 (I). – С. 204–209.

6. *Коломеец А.В.* Экспериментальное исследование диагностических тестов для локализации одиночных выходных ошибок и ошибок переходов в расширенных автоматах // Наука. Технологии. Инновации : материалы всероссийской научной конференции молодых ученых : в 7 ч. – Новосибирск : Изд-во НГТУ, 2006. – Ч. 1. – С. 238–240.

7. *Коломеец А.В.*, Шабалдин А.В., Спицина Н.В. Автоматизация тестирования реализаций протоколов прикладного уровня в лабораторном практикуме // Современные средства и системы автоматизации : материалы IV научно-практической конференции. – Томск : Изд-во ТУСУР, 2003. – С. 222–225.

8. *Коломеец А.В.*, Прокопенко С.А. Соответствие между ошибками в программных реализациях протоколов и расширенных автоматах // Вестник ТГУ. Приложение. – 2005. – № 14. – С. 154–157.

9. *Коломеец А.В.*, Прокопенко С.А. Метод синтеза проверяющих тестов для расширенных автоматов без построения эквивалентного конечного автомата // Вестник ТГУ. Приложение. – 2006. – № 18. – С. 62–66.

10. Михайлов Ю.В., *Коломеец А.В.* Автоматизация внесения ошибок в программные реализации протоколов на основе модели расширенного автомата // Наука. Технологии. Инновация : материалы всероссийской научной конференции молодых ученых : в 7 ч. – Новосибирск : Изд-во НГТУ, 2006. – Ч. 1. – С. 49–51.

11. El-Fakih K., *Kolomeez A.*, Prokopenko S., Yevtushenko N. Extended Finite State Machine Based Test Derivation Driving By User Defined Faults // International Conference ICST'2008 / IEEE. – 2008. – P. 308-317.

12. Михайлов Ю.В., *Коломеец А.В.* Проверка переходов в расширенном автомате на основе срезов // Вестник ТГУ. Управление, вычислительная техника и информатика. – 2008. – № 3 (4). – С. 110–118.

Тираж 120 экз.
Отпечатано в ООО «Позитив-НБ»
634050 г. Томск, пр. Ленина 34а