На правах рукописи

Царев Федор Николаевич

Методы построения конечных автоматов на основе эволюционных алгоритмов

Специальность 05.13.11 – Математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей

АВТОРЕФЕРАТ

диссертации на соискание ученой степени

кандидата технических наук

Санкт-Петербург – 2012 г.

Работа выполнена на кафедре «Компьютерные технологии» Санкт-Петербургского национального исследовательского университета информационных технологий, механики и оптики (НИУ ИТМО).

Научный руководитель	доктор технических наук, профессор Шалыто Анатолий Абрамович
Официальные оппоненты	Тулупьев Александр Львович, доктор физико-математических наук, доцент, заведующий лабораторией теоретических и междисциплинарных проблем информатики Санкт-Петербургского института информатики и автоматизации Российской академии наук
	Тропченко Александр Ювенальевич, доктор технических наук, профессор, профессор кафедры «Вычислительная техника» НИУ ИТМО
Ведущая организация	Институт проблем управления им. В. А. Трапезникова Российской академии наук

Защита диссертации состоится 29 ноября 2012 года в 17 часов 00 минут на заседании диссертационного совета Д212.227.06 при НИУ ИТМО по адресу: 197101, Санкт-Петербург, Кронверкский пр., д. 49, Центр Интернет-образования.

С диссертацией можно ознакомиться в библиотеке НИУ ИТМО.

Автореферат разослан «26» октября 2012 года.

Ученый секретарь диссертационного совета,
доктор технических наук, профессор — Л. С. Лисицына

Общая характеристика работы

Актуальность проблемы. В последнее время при разработке программного обеспечения (ПО) для управляющих систем все шире применяется автоматное программирование – парадигма программирования, при использовании которой программу предлагается строить в виде совокупности автоматизированных объектов управления, каждый из которых содержит систему управления (один или несколько взаимодействующих конечных автоматов, в дальнейшем – автоматов) и объект управления. При этом программы, построенные на основе указанной парадигмы, называются автоматными. Важным достоинством автоматного программирования является то, что при его использовании существенно упрощается верификация программ на основе метода *Model Checking*, так как построение модели Крипке по автоматной программе и обратный переход могут быть автоматизированы.

При использовании инструментальных средств для поддержки автоматного программирования таких, как, например, *UniMod*, около 70% исходного кода автоматной программы может быть сгенерировано автоматически. Уровень автоматизации программирования этого класса систем станет значительно выше, если удастся автоматизировать процесс построения автоматов, что и является предметом исследования в настоящей работе.

В последнее десятилетие активно развивается область исследований, называемая поисковой инженерией ПО (Search-Based Software Engineering), в которой для решения задач программной инженерии предлагается применять алгоритмы поисковой оптимизации, в частности, эволюционные алгоритмы (генетические алгоритмы, эволюционные стратегии и метод спуска на основе случайных мутаций).

Как указано выше, и при использовании автоматного программирования возникает задача, для решения которой можно, а иногда и необходимо, применять методы поисковой инженерии ПО. Это определяется тем, что построение автоматов вручную может представлять существенную сложность, а в ряде случаев построить автоматы вручную не удается.

Разработка методов решения указанной задачи является одним из шагов к автоматическому программированию и позволит резко повысить уровень автоматизации построения автоматных программ и снизить влияние человеческого фактора на их качество.

Поэтому исследования, направленные на разработку методов построения автоматов на основе эволюционных алгоритмов, являются актуальными.

Цель диссертационной работы – разработка методов построения автоматов на основе эволюционных алгоритмов.

Основные задачи диссертационной работы состоят в следующем:

1. Разработать метод построения автоматов по обучающим примерам на основе эволюционных алгоритмов.
2. Разработать метод выполнения операции скрещивания для генетических алгоритмов, учитывающий поведение автомата на обучающих примерах.
3. Разработать метод построения автоматов по обучающим примерам и темпоральным свойствам на основе эволюционных алгоритмов и верификации.

4. Разработать технологию построения автоматов по обучающим примерам и темпоральным свойствам.
5. Разработать инструментальное средство для автоматизации построения автоматов.
6. Внедрить разработанные методы при построении автомата управления моделью беспилотного самолета и в учебный процесс.

Научная новизна. В работе получены следующие новые научные результаты, которые выносятся на защиту:

1. Метод построения автоматов по обучающим примерам на основе эволюционных алгоритмов. Его основное отличие от известных состоит в том, что в предлагаемые алгоритмы добавлен новый шаг «Расстановка выходных воздействий», который выполняется перед вычислением функции приспособленности.
2. Метод выполнения операции скрещивания для генетических алгоритмов, учитывающий поведение автомата на обучающих примерах. Показано, что генетический алгоритм, использующий разработанный метод выполнения операции скрещивания, осуществляет построение автоматов по обучающим примерам быстрее, чем генетический алгоритм, использующий традиционный метод выполнения операции скрещивания, эволюционная стратегия и метод спуска на основе случайных мутаций.
3. Метод построения автоматов по обучающим примерам и темпоральным формулам на основе эволюционных алгоритмов и верификации. Его основное отличие от известных состоит в том, что для вычисления функции приспособленности совместно применяются обучающие примеры и метод *Model Checking*.

Методы исследования. В работе используются методы теории автоматов, дискретной математики и эволюционных алгоритмов.

Достоверность научных положений, выводов и практических рекомендаций, полученных в диссертации, подтверждается корректным обоснованием постановок задач, точной формулировкой критериев, а также результатами экспериментов по использованию предложенных в диссертации методов.

Практическое значение работы состоит в том, что на основе предложенных методов разработана технология автоматизированного построения автоматов на основе эволюционных алгоритмов. На ее основе создано инструментальное средство для автоматизации построения автоматов, по которым, как отмечено выше, автоматически может быть сгенерирован программный код. Предложенные в работе эволюционные алгоритмы позволяют решить задачи построения автоматов, которые не удается решить вручную, а для других автоматов – существенно сократить затраты времени на их построение по сравнению с известными методами, что подтверждается результатами экспериментальных исследований, приведенными в работе.

Внедрение результатов работы. Результаты диссертации использовались при выполнении научно-исследовательских работ по следующим государственным контрактам: «Технология генетического программирования для генерации автоматов управления системами со сложным поведением» (государственный контракт

№ 02.514.11.4044 от 18.05.2007 г. по Федеральной целевой программе «Исследования и разработки по приоритетным направлениям развития научно-технологического комплекса России на 2007–2013 годы»), «Разработка методов совместного применения генетического и автоматного программирования для построения систем управления беспилотными летательными аппаратами» (государственный контракт № П1188 от 27.08.2009 г. по Федеральной целевой программе «Научные и научно-педагогические кадры инновационной России» на 2009–2013 годы), «Разработка методов машинного обучения на основе генетических алгоритмов для построения управляющих конечных автоматов» (государственный контракт № П2174 от 09.11.2009 г. по Федеральной целевой программе «Научные и научно-педагогические кадры инновационной России» на 2009–2013 годы), «Применение методов искусственного интеллекта в разработке управляющих программных систем» (государственный контракт № П2236 от 11.11.2009 г. по Федеральной целевой программе «Научные и научно-педагогические кадры инновационной России» на 2009–2013 годы), в рамках проекта «Подготовка и переподготовка профильных специалистов на базе центров образования и разработок в сфере информационных технологий в Северо-Западном федеральном округе» (Государственный контракт № 07.P20.11.0028 от 07.09.2011 г.), а также в учебном процессе на кафедре «Компьютерные технологии» в рамках курса «Теория автоматов и программирование».

Апробация результатов работы. Основные положения диссертационной работы докладывались на следующих научных и научно-практических конференциях: IV Международная научно-практическая конференция «Интегрированные модели и мягкие вычисления в искусственном интеллекте» (Коломна, 2007), научно-техническая конференция «Научное программное обеспечение в образовании и научных исследованиях» (СПбГПУ, 2008), XII Всероссийская конференция по проблемам науки и высшей школы «Фундаментальные исследования и инновации в технических университетах» (СПбГПУ, 2008), Second Spring Young Researchers' Colloquium on Software Engineering (SYRCoSE'2008, SPbSU, 2008), III и IV Международная научно-практическая конференция «Современные информационные технологии и ИТ-образование» (ВМК МГУ, 2008, 2009), научно-практическая конференция студентов, аспирантов, молодых ученых и специалистов «Интегрированные модели, мягкие вычисления, вероятностные системы и комплексы программ в искусственном интеллекте (ИММВИИ-2009)» (Коломна, 2009), VI и VII межвузовская конференция молодых ученых (СПбГУ ИТМО, 2009, 2010), X, XI и XII Международная конференции по мягким вычислениям и измерениям (СПбГЭТУ (ЛЭТИ), 2009, 2010, 2011), Международная научная конференция «Компьютерные науки и информационные технологии» памяти А. М. Богомолова (Саратовский государственный университет имени Н. Г. Чернышевского, 2009), 32-я конференция молодых ученых и специалистов Института проблем передачи информации им. А. А. Харкевича РАН «Информационные технологии и системы» (2009), XL научная и учебно-методическая конференция профессорско-преподавательского и научного состава СПбГУ ИТМО (2011), 14-th Annual Graduate Students Workshop (part of the «Genetic and Evolutionary Computation Conference» GECCO – 2011, Dublin, organized by ACM SIGEVO, 2011), вторая межвузовская

научная конференция по проблемам информатики (СПИСОК, СПбГУ, 2011), XLI научная и учебно-методическая конференция НИУ ИТМО (2012), 15-th Annual Graduate Students Workshop (part of the «Genetic and Evolutionary Computation Conference» GECCO – 2012, Philadelphia, organized by ACM SIGEVO, 2012), третья российская конференция с международным участием «Технические и программные средства систем управления, контроля и измерения» (Институт проблем управления имени В. А. Трапезникова РАН, 2012) – пленарный доклад.

Публикации. На тему диссертации опубликованы 23 печатные работы, в том числе шесть статей, из которых пять в журналах из перечня ВАК.

Свидетельства о регистрации программ для ЭВМ. В рамках диссертационной работы получены три свидетельства о регистрации программ для ЭВМ.

Структура диссертации. Диссертация изложена на 196 страницах и состоит из введения, четырех глав, заключения и двух приложений. Список источников содержит 170 наименований. Работа проиллюстрирована 41 рисунком и 14 таблицами.

Содержание работы

В **первой главе** приводится обзор работ, посвященных автоматному программированию, поисковой инженерии ПО и применению эволюционных алгоритмов для построения автоматов.

При этом отметим, что поисковая инженерия ПО является новой областью исследований, и считается, что на апрель 2012 г. по этой тематике было опубликовано всего 1020 работ, охватывающие различные подходы (например, такие, как метод роя частиц или меметические алгоритмы). Однако применительно к построению автоматов в литературе используются лишь эволюционные алгоритмы, которые и будут совершенствоваться в настоящей работе. Существует несколько типов эволюционных алгоритмов: генетические алгоритмы, эволюционные стратегии, метод спуска на основе случайных мутаций, которые и рассматриваются ниже.

Генетические алгоритмы обеспечивают поиск оптимальных решений одновременно в нескольких точках. Вначале случайным образом генерируется некоторое число решений (особей), образующих начальное *поколение*. Далее особи *скрещиваются* и *мутируют,* формируя новое поколение. Скрещивание – операция, которая по двум особям генерирует одну или несколько особей, сочетающих в себе свойства «родителей». Мутация – операция небольшого изменения особи.

Эволюционные стратегии имеют два отличия от генетических алгоритмов: в них не используется операция скрещивания и, как правило, каждое поколение состоит только из одной особи. Опишем (1+1)-эволюционную стратегию. Процесс поиска начинается со случайно выбранного допустимого решения. На каждой итерации к текущей особи применяется операция мутации. Она устроена так, что в ее результате может получиться (пусть и с малой вероятностью) любое допустимое решение. Если результат этой операции – решение с бо́льшим значением функции приспособленности, то оно становится текущим. Работа алгоритма считается оконченной, когда будет достигнуто необходимое значение функции приспособленности.

Метод спуска на основе случайных мутаций состоит в следующем. Процесс поиска начинается со случайно выбранной точки в пространстве решений задачи. На

каждой итерации к текущей особи применяется операция мутации. Она устроена так, что в результате ее выполнения могут получиться только особи, незначительно отличающиеся от текущей. Как и в эволюционной стратегии, если результат этой операции – решение с бо́льшим значением функции приспособленности, то оно становится текущим. Работа алгоритма считается оконченной, когда будет достигнуто необходимое значение функции приспособленности.

На основании проведенного обзора работ по применению эволюционных алгоритмов для построения автоматов можно сделать следующие выводы:

- генетические алгоритмы, использующие высокоуровневое представление автоматов, более эффективны, чем генетические алгоритмы, использующие представление конечных автоматов в виде битовых строк;
- для построения *управляющих* автоматов (с выходными воздействиями и с пометками дуг булевыми формулами и событиями) ранее применялись только методы, использующие *вычисление функции приспособленности на основе моделирования*;
- в работах, использующих эволюционную стратегию и метод спуска на основе случайных мутаций, выбор этих методов обоснован тем, что для автоматов трудно предложить осмысленный способ выполнения операции скрещивания;
- известна только одна работа (C. Johnson), посвященная построению автоматов с вычислением функции приспособленности на основе верификации, однако предлагаемый в ней алгоритм, в котором используются только темпоральные формулы, является весьма медленным.

На основе результатов обзора формулируются задачи, решаемые в настоящей диссертации.

Вторая глава посвящена методам построения автоматов *по обучающим примерам* на основе эволюционных алгоритмов, выполнения операции скрещивания с учетом поведения автомата на обучающих примерах, а также построения управляющих автоматов на основе эволюционных алгоритмов по обучающим примерам и темпоральным свойствам. При этом отметим, что методы построения *управляющих* автоматов на основе обучающих примеров из литературы не известны.

Исходными данными для построения автомата в настоящей работе являются:

- множество событий $E = \{e_1, e_2, \ldots, e_n\}$;
- множество входных переменных $X = \{x_1, x_2, \ldots, x_m\}$;
- множество выходных воздействий $Z = \{z_1, z_2, \ldots, z_k\}$;
- множество обучающих примеров (тестов) Tests;
- максимальное число состояний в искомом автомате K.

Опишем структуру тестов: i-й из них состоит из двух последовательностей – входной Input[i] и выходной Answer[i]. *Элементами входной последовательности* являются пары (e, f), где e – некоторое событие из множества E, а f – булева формула от входных переменных, задающая условие перехода. Выходными данными в задаче построения управляющего автомата по набору тестов является автомат, содержащий не более K состояний, удовлетворяющий каждому тесту из множества Tests и

обладающий свойством непротиворечивости, либо сообщение о том, что такого автомата не существует. Автомат *A удовлетворяет тесту* Test = (Input, Answer), если результат обработки входной последовательности Input из начального состояния автомата совпадает с последовательностью Answer.

Результат обработки входной последовательности Input из состояния *s* определяется рекурсивно:

- если последовательность Input пуста, то результат ее обработки также является пустой последовательностью;
- если последовательность Input не пуста, а ее первый элемент имеет вид (*e*, *f*), то результат обработки последовательности Input есть конкатенация двух последовательностей: результата обработки элемента входной последовательности (*e*, *f*) в состоянии *s* и результата обработки оставшейся части последовательности Input из состояния, в которое ведет переход из *s* по событию *e* и условию, эквивалентному *f*. Если результат обработки оставшейся части последовательности Input не определен, то и результат обработки последовательности также не определен.

Результатом обработки элемента входной последовательности (*e*, *f*) в состоянии *s* является последовательность выходных воздействий, соответствующая переходу, который помечен событием *e* и условием, эквивалентным *f*. Если же такого перехода из состояния *s* нет или таких переходов несколько, то результат обработки указанного элемента не определен.

Автомат в эволюционном алгоритме представляется в виде объекта, который содержит описания переходов для каждого состояния. Каждый переход описывается событием, при поступлении которого этот переход выполняется, условием и *числом выходных воздействий*, которые должны быть сгенерированы при выборе этого перехода. Таким образом, в особи кодируется только «каркас» автомата.

Для построения автоматов в диссертации существующие эволюционные алгоритмы предлагается модифицировать, добавив перед вычислением функции приспособленности в каждый из них дополнительный шаг – расстановку выходных воздействий. Он необходим, так как особь в эволюционном алгоритме, как указано выше, представляет собой «каркас» автомата. При этом расстановка выходных воздействий должна выполняться так, чтобы значение функции приспособленности получившегося в результате автомата было максимальным или близким к нему.

Опишем алгоритм расстановки выходных воздействий для дискретных воздействий. Для каждого перехода *T* и каждой последовательности выходных воздействий zs вычисляется величина *C*[*T*][zs] – число раз, когда при обработке входной последовательности, соответствующей одному из тестов, на переходе *T* должны быть выработаны выходные воздействия, образующие последовательность zs. Далее, каждый переход помечается той последовательностью zs_0, для которой величина *C*[*T*][zs_0] максимальна. На рисунке 1 слева изображена особь эволюционного алгоритма до применения алгоритма расстановки выходных воздействий, а справа – после его применения.

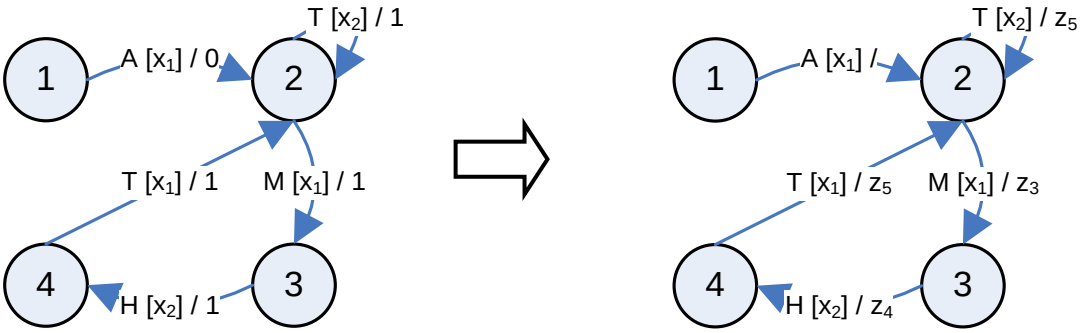


Рисунок 1 – Пример работы алгоритма расстановки выходных воздействий

Предлагается функцию приспособленности вычислять на основе редакционного расстояния. При этом выполняются следующие действия: на вход автомату подается каждая из последовательностей Input[i]. Обозначим результат обработки входной последовательности Input[i], как Output[i]. Если этот результат не определен, то будем считать, что Output[i] равен пустой последовательности. После этого вычисляется значение вспомогательной переменной $FF_1 = \frac{\sum_{i=1}^{n}(1 - \frac{ED(Output[i], Answer[i])}{\max(|Output[i]|, |Answer[i]|)})}{n}$, где как ED($A$, B) обозначено редакционное расстояние между последовательностями A и B. Отметим, что значение FF_1 лежит в пределах от 0 до 1. При этом чем «лучше» автомат соответствует тестам, тем больше ее значение.

Функция приспособленности зависит не только от того, насколько «хорошо» автомат работает на тестах, но и от числа переходов, которые он содержит. Она вычисляется по формуле: $FF_2 = \begin{cases} 10 \cdot FF_1 + \frac{1}{M} \cdot (M - cnt), FF_1 < 1 \\ 20 + \frac{1}{M} \cdot (M - cnt), FF_1 = 1 \end{cases}$, где M – число, большее числа переходов в автомате, а cnt – число переходов в автомате, кодируемом рассматриваемой особью.

Учет числа переходов в функции приспособленности необходим по двум причинам. Во-первых, минимизация числа переходов приводит к тому, что в результирующем автомате отсутствуют не используемые в тестах переходы. Во-вторых, чем меньше в автомате переходов, тем более «общее» поведение он задает. Таким образом, решается проблема переобучения (overfitting), заключающаяся в том, что автомат демонстрирует правильное поведение только на входных последовательностях, соответствующих набору обучающих примеров.

При выполнении операции мутации с заданной вероятностью (по умолчанию, она равна 0.05) выполняется одно из действий: изменение описания каждого из переходов и удаление или добавление перехода для каждого состояния.

При изменении описания перехода с равной вероятностью выполняется одно из следующих действий: 1. Изменение состояния, в которое ведет переход (изменяется на случайно выбранное). 2. Изменение события, по которому выполняется переход (изменяется на случайно выбранное). 3. Изменение числа выходных воздействий, вырабатываемых на этом переходе (с равной вероятностью это число либо

уменьшается на единицу, либо увеличивается на единицу, но при этом не может стать отрицательным или превзойти некоторое заданное ограничение). 4. Изменение условия (изменяется на случайно выбранное из числа тех, которые встречаются в тестах вместе с событием, которым помечен переход).

После выполнения операции мутации к каждому состоянию автомата применяется алгоритм удаления дублированных и неоднозначных переходов. Обозначим список переходов, поступающих на вход этого алгоритма как L_{in}, а список переходов, получающийся в результате его работы, как L_{out}. Для его формирования выполняются следующие операции: по очереди рассматриваются переходы, входящие в список L_{in}; очередной переход t добавляется в L_{out}, если в нем еще нет перехода, помеченного тем же событием, что и t, и условием, имеющим общую выполняющую подстановку с условием перехода t.

Скрещивание описаний автоматов производится следующим образом. Обозначим как P1 и P2 «родительские» особи, а как S1 и S2 – особи-«потомки». Начальные состояния S1.is и S2.is имеют номер 0, так как у всех особей нулевое состояние является начальным. Скрещивание описаний автоматов производится отдельно для каждого состояния. Обозначим список переходов из состояния номер i автомата P1 как P1.T[i], а список переходов из состояния номер i автомата P2 как P2.T[i].

При использовании традиционного метода скрещивания списки переходов S1.T[i] и S2.T[i] формируются следующим образом:

1. Строится общий список переходов, в который помещаются переходы, входящие как в P1.T[i], так и в P2.T[i].
2. К полученному списку применяется случайная перестановка. Далее возможны два равновероятных варианта: либо в S1.T[i] помещаются первые |P1.T[i]| переходов из полученного списка, а в S2.T[i] – оставшиеся переходы; либо в S1.T[i] помещаются первые |P2.T[i]| переходов из полученного списка, а в S2.T[i] – оставшиеся переходы.

К получившимся в результате скрещивания автоматам S1 и S2 применяется алгоритм удаления дублированных и противоречивых переходов.

Перейдем к изложению метода скрещивания с учетом поведения автомата на обучающих примерах. Списки переходов S1.T[i] и S2.T[i] строятся следующим образом:

1. В автоматах P1 и P2 помечаются те переходы, которые выполняются при обработке 10% тестов, для которых значение $\frac{\text{ED}(\text{Output}[i], \text{Answer}[i])}{\max(|\text{Output}[i]|, |\text{Answer}[i]|)}$ минимально.
2. Список переходов S1.T[i] формируется следующим образом: сначала в него копируются помеченные переходы из P1.T[i], затем помеченные переходы из P2.T[i], а затем непомеченные переходы из P1.T[i].
3. Список переходов S2.T[i] формируется следующим образом: сначала в него копируются помеченные переходы из P2.T[i], затем помеченные переходы из P1.T[i], а затем непомеченные переходы из P2.T[i].

Как и в известном методе скрещивания, к получившимся в результате скрещивания автоматам S1 и S2 применяется алгоритм удаления дублированных и противоречивых переходов.

В экспериментах, выполненных в диссертации, рассмотрены шесть алгоритмов: генетический алгоритм, в котором используется только традиционная операция скрещивания (обозначается ГА-1); метод спуска на основе случайных мутаций (МС); (1+1)-эволюционная стратегия (ЭС); генетический алгоритм, в котором используется только операция скрещивания с учетом поведения автомата на обучающих примерах (ГА-2); гибридный алгоритм, состоящий из метода ГА-2 и метода спуска на основе случайных мутаций (ГА-2+МС); гибридный алгоритм, состоящий из метода ГА-2 и (1+1)-эволюционной стратегии (ГА-2+ЭС). Эти алгоритмы применялись для построения автомата управления часами с будильником (известен из литературы) и для построения автоматов по обучающим примерам, сгенерированным случайным образом.

В экспериментах по построению автомата управления часами с будильником входные данные состояли из 38 тестов (суммарная длина входных последовательностей – 242, выходных – 195). Было проведено по 1000 запусков каждого алгоритма, для каждого из которых записывалось время работы. В эволюционных алгоритмах время работы измеряется числом запусков функции приспособленности, так как, как правило, ее вычисление является более трудоемким, чем выполнение операций мутации и скрещивания.

В таблице 1 приведены минимальные, максимальные, средние и медианные значения числа вычислений функции приспособленности для каждого из алгоритмов.

Таблица 1 – Результаты вычислительных экспериментов по построению автомата управления часами с будильником

Алгоритм	**Минимум**	**Максимум**	**Среднее**	**Медиана**
ГА-1	1093938	41794531	6783215	5014202
МС	1387	9710090	1275439	792481
ЭС	1325	9915947	1317674	901615
ГА-2	51977	1196233	205451	142013
ГА-2+МС	46311	780469	127712	103904
ГА-2+ЭС	38458	2714324	129330	84778

Для каждой пары алгоритмов был проведен статистический тест *ANOVA*. Его результаты приведены в таблице 2. В ней для каждой пары алгоритмов указано *p*-значение. Если это значение меньше 0.05, то можно сделать вывод о том, что время работы соответствующих алгоритмов на рассматриваемой задаче существенно различается.

Таблица 2 – Результаты статистического теста

	ГА-1	МС	ЭС	ГА-2	ГА-2+МС	ГА-2+ЭС
ГА-1	-	$<3\cdot10^{-16}$	$<3\cdot10^{-16}$	$<3\cdot10^{-16}$	$<3\cdot10^{-16}$	$<3\cdot10^{-16}$
МС	-	-	0.5178	$<3\cdot10^{-16}$	$<3\cdot10^{-16}$	$<3\cdot10^{-16}$
ЭС	-	-	-	$<3\cdot10^{-16}$	$<3\cdot10^{-16}$	$<3\cdot10^{-16}$
ГА-2	-	-	-	-	$<3\cdot10^{-16}$	$<3\cdot10^{-16}$
ГА-2+МС	-	-	-	-	-	0.7446
ГА-2+ЭС	-	-	-	-	-	-

Для проведения экспериментов по построению управляющих автоматов по обучающим примерам, сгенерированным случайным образом, были сгенерированы автоматы, содержащие 4, 5, …, 10 состояний. Далее по каждому из автоматов был сгенерирован случайным образом набор тестов (при этом суммарная длина входных последовательностей в этих тестах составляла 150*k* для автомата из *k* состояний). Для каждого алгоритма и каждого набора тестов было проведено 100 запусков. Для каждого запуска записывалось время работы. В таблице 3 приведено среднее время работы алгоритмов на случайно сгенерированных тестах.

Таблица 3 – Среднее время работы алгоритмов на тестах, сгенерированных случайным образом

Число состояний	ГА-1	МС	ЭС	ГА-2	ГА-2+МС	ГА-2+ЭС
4	25514587	5432560	6017983	875514	629232	460564
5	62124022	10711175	13519507	2032536	1545070	1241844
6	161261630	29291065	36216142	4537179	3721738	2981650
7	376970651	80844608	66850953	10928249	8799660	6815845
8	881025771	144343577	179974240	27500471	20036385	13877897
9	2149301686	350141493	409724265	56380237	41708779	32884793
10	4496075865	830139535	924940477	131551924	108635507	85621418

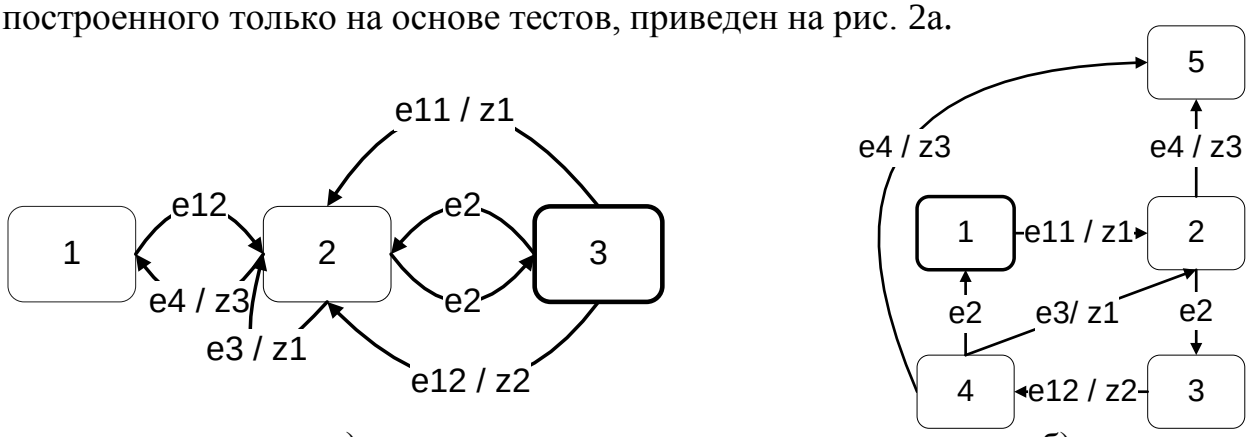
По результатам выполненных экспериментов алгоритмы можно разбить на четыре группы: ГА-1; МС и ЭС; ГА-2; ГА-2+ЭС и ГА-2+МС. Указанное разбиение на группы обладает тем свойством, что алгоритмы внутри группы имеют статистически неразличимое время работы, а для алгоритмов из разных групп оно существенно различается.

Перейдем к описанию метода построения автоматов по обучающим примерам и темпоральным формулам на основе эволюционных алгоритмов и верификации. Его основная особенность состоит в том, что исходными данными, в дополнение к тем, которые использовались в методе, описанном выше, являются *LTL*-формулы, описывающие требования к управляющему автомату. При вычислении функции приспособленности учитывается как поведение автомата на обучающих примерах, так и число выполняющихся для автомата *LTL*-формул, составляющих спецификацию.

Для вычисления функции приспособленности автомат, задаваемый рассматриваемой особью, запускается на всех тестах и проверяется на соответствие всем темпоральным формулам. Функция приспособленности вычисляется по формуле $\mathrm{FF} = \mathrm{FF}_2 \cdot (1 + \frac{n_1}{n_2})$. Здесь n_2 – общее число темпоральных формул в спецификации, а n_1 – число формул, которые выполняются для рассматриваемого автомата. Операции скрещивания и мутации выполняются так же, как и в методе, описанном выше.

Экспериментальное исследование предложенного метода проводилось на задаче построения автомата управления дверьми лифта. Эта система содержит пять входных событий и три выходных воздействия. Входные данные состояли из девяти тестов и 11 темпоральных свойств. Построение осуществлялось с помощью генетического алгоритма, использующего скрещивание с учетом поведения автомата на обучающих примерах. При этом сравнивались между собой два варианта – не использующий верификацию и использующий ее. Было проведено по 1000 запусков каждого из

алгоритмов, и для каждого из запусков записывалось время работы. Пример автомата, построенного только на основе тестов, приведен на рис. 2а.



а) б)

Рисунок 2 – Построенные автоматы управления дверьми лифта

Этот автомат обладает тем недостатком, что может отдать команду на закрытие дверей после того, как они сломаются, или начать открывать (закрывать) двери, когда они уже открыты (закрыты). При использовании верификации моделей построенный автомат (рис. 2б) всегда удовлетворял всем требованиям спецификации.

При построении автомата управления дверьми лифта только на основе тестов среднее значение вычислений функции приспособленности оказалось равным $7.479 \cdot 10^4$ (минимальное число вычислений – $2.184 \cdot 10^4$, максимальное – $2.999 \cdot 10^5$). При использовании верификации совместно с тестами среднее значение числа вычислений функции приспособленности оказалось равным $7.246 \cdot 10^5$ (минимальное число вычислений – $7.054 \cdot 10^4$, максимальное – $5.492 \cdot 10^6$). Эксперименты показали, что при построении автоматов только на основе тестов в девяти случаях из 1000 результатом являлся автомат, который полностью удовлетворяет спецификации.

Третья глава посвящена технологии построения управляющих автоматов на основе эволюционных алгоритмов и инструментальному средству для автоматизированного построения автоматов.

Исходными данными для построения управляющих автоматов на основе эволюционных алгоритмов являются описания поставщиков событий и объектов управления (в общем случае их может быть несколько), а также спецификация программы.

На основе описаний формируются списки входных переменных, выходных воздействий и событий. Далее по спецификации программы строится набор тестов (обучающих примеров) и, при необходимости, темпоральных свойств.

К набору тестов предъявляются следующие требования. Во-первых, если система со сложным поведением, для управления которой строится автомат, имеет несколько режимов работы, то система тестов должна содержать тесты для каждого из этих режимов и для переходов между ними. Во-вторых, тесты не должны противоречить друг другу – если входная последовательность в одном из тестов является префиксом входной последовательности в другом тесте, то и выходная последовательность из первого теста должна быть префиксом выходной последовательности из второго теста. В-третьих, набор тестов должен содержать все существующие в системе входные события, входные переменные и выходные воздействия.

Тесты совместно с темпоральными формулами подаются на вход эволюционному алгоритму. Отметим, что эволюционный алгоритм одинаков для всех задач – для новой задачи необходим только новый набор тестов, входных и выходных воздействий и, возможно, параметров алгоритма (рис. 3).

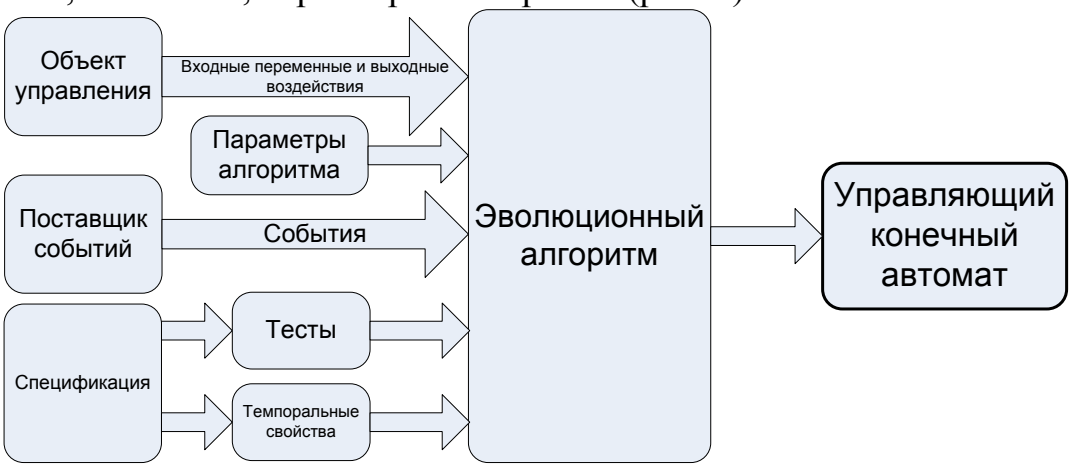


Рисунок 3 – Технология построения автоматов на основе эволюционных алгоритмов

Эта технология положена в основу разработанного инструментального средства, исходный код которого размещен в открытом доступе в сети Интернет по адресу: http://code.google.com/p/gabp/. Входные данные для этого инструментального средства описываются в формате *XML*. Результатом работы инструментального средства является *XML*-описание графа переходов автомата в формате инструментального средства *UniMod*. Построенный автомат соответствует обучающим примерам и удовлетворяет темпоральным свойствам, заданным во входном файле.

В **четвертой главе** приводятся результаты внедрения разработанных методов на примере построения автомата управления моделью беспилотного самолета и в учебный процесс.

Для построения системы управления моделью беспилотного самолета при исполнении «мертвой петли» метод построения автоматов по обучающим примерам был применен для случая не только дискретных (как в прототипе), но и непрерывных выходных воздействий. В этом случае при предложенном выборе функции приспособленности расстановка непрерывных выходных воздействий осуществляется на основе решения набора систем линейных уравнений. Моделирование беспилотного самолета проводилось с использованием симулятора *FlightGear* (http://www.flightgear.org).

Для построения автомата были записаны несколько обучающих примеров, каждый из которых соответствовал выполнению «мертвой петли» под управлением пилота-человека. Каждый пример состоял из нескольких тысяч наборов входных и выходных воздействий. Время работы алгоритма составляло около 10 часов для одного набора тестов. Было проведено 50 запусков генетического алгоритма, в каждом из которых выбирался автомат с наибольшим значением функции приспособленности. Полеты моделей самолетов, управляемых выбранными автоматами, были просмотрены экспертом. В результате был выбран один автомат, который наиболее точно осуществлял управление моделью самолета. Результаты этой части работы составили предмет статьи, принятой в журнал «Известия РАН. Теория и системы управления».

Для внедрения в учебный процесс были разработаны две виртуальные лаборатории (для языков программировании *Java* и *C#*) для обучения построению конечных автоматов на основе эволюционных алгоритмов. Обе виртуальные лаборатории имеют модульную архитектуру – содержат ядро, предоставляющее базовую функциональность, которая расширяется с помощью подключаемых модулей (плагинов). Ядро лаборатории позволяет просматривать и сохранять в виде файлов графики зависимости значений некоторых функций (например, максимального, минимального и среднего значения функции приспособленности особей поколения) от номера поколения. Кроме этого поддерживается возможность визуализации особей и их сохранения в текстовом формате. Также ядро программы поддерживает подключение плагинов «на лету» (во время работы программы). С использованием этих виртуальных лабораторий с 2008 года проводятся лабораторные работы по курсу «Теория автоматов и программирование», которые были выполнены более чем 150 студентами третьего курса кафедры «Компьютерные технологии» НИУ ИТМО. Часть отчетов по лабораторным работам опубликована на сайте http://is.ifmo.ru/labs/.

Заключение

В результате диссертационного исследования получены следующие результаты:

1. Разработан метод построения автоматов по обучающим примерам на основе эволюционных алгоритмов. Его основное отличие от известных состоит в том, что в эволюционные алгоритмы добавлен новый шаг «Расстановка выходных воздействий», который выполняется перед вычислением функции приспособленности.
2. Разработан метод выполнения операции скрещивания для генетических алгоритмов, учитывающий поведение автомата на обучающих примерах. Показано, что генетический алгоритм, использующий разработанный метод выполнения операции скрещивания, осуществляет построение автоматов по обучающим примерам быстрее, чем генетический алгоритм, использующий традиционный метод выполнения операции скрещивания, эволюционная стратегия и метод спуска на основе случайных мутаций.
3. Разработан метод построения автоматов по обучающим примерам и темпоральным свойствам на основе эволюционных алгоритмов и верификации. Его основное отличие от известных состоит в том, что для вычисления функции приспособленности совместно применяются обучающие примеры и верификация.
4. Разработана технология построения автоматов по обучающим примерам и темпоральным свойствам.
5. Разработано инструментальное средство для автоматизации построения автоматов.
6. Разработанные методы использованы при построении автомата управления моделью беспилотного самолета и внедрены в учебный процесс.

Статьи в журналах из перечня ВАК

1. *Царев Ф. Н.* Совместное применение генетического программирования, конечных автоматов и искусственных нейронных сетей для построения системы управления беспилотным летательным аппаратом //Научно-технический вестник СПбГУ ИТМО. 2008. Выпуск 53. Автоматное программирование, с. 42–60.
2. *Царев Ф. Н., Давыдов А. А., Соколов Д. О.* Применение генетического программирования и методов сокращенных таблиц переходов и деревьев решений для построения автоматов управления моделью беспилотного летательного аппарата //Научно-технический вестник СПбГУ ИТМО. 2008. Выпуск 53. Автоматное программирование, с. 60–79.
3. *Царев Ф. Н.* Метод построения управляющих конечных автоматов на основе тестовых примеров с помощью генетического программирования // Информационно-управляющие системы. 2010. № 5, с. 31–36.
4. *Царев Ф. Н., Егоров К. В., Шалыто А. А.* Применение генетического программирования для построения автоматов управления системами со сложным поведением на основе обучающих примеров и спецификации // Научно-технический вестник Санкт-Петербургского государственного университета информационных технологий, механики и оптики. 2010. № 5 (69), с. 81–86.
5. *Царев Ф. Н., Александров А. В., Казаков С. В. Сергушичев А. А., Шалыто А. А.* Генерация конечных автоматов для управления моделью беспилотного самолета // Научно-технический вестник СПбГУ ИТМО. 2011. № 2, с. 3–11.

Другие публикации

6. *Царев Ф. Н., Егоров К. В., Шалыто А. А.* Совместное применение генетического программирования и верификации для построения автоматов управления системами со сложным поведением // Труды СПИИРАН. 2010. Вып. 15, с. 123–135.
7. *Царев Ф. Н., Шалыто А. А.* Применение генетического программирования для построения мультиагентной системы одного класса /Международная научно-техническая мультиконференция «Проблемы информационно-компьютерных технологий и мехатроники». Материалы международной научно-технической конференции «Многопроцессорные вычислительные и управляющие системы» (МВУС`2007). Таганрог: НИИМВС. Т.2, с. 46–51.
8. *Царев Ф. Н., Шалыто А. А.* Применение генетического программирования для генерации автоматов в задаче об «умном муравье» /Сборник научных трудов. IV-я Международная научно-практическая конференция «Интегрированные модели и мягкие вычисления в искусственном интеллекте». М.: Физматлит. 2007, с. 590–597.
9. *Царев Ф. Н., Шалыто А. А.* О построении автоматов с минимальным числом состояний для задачи об «умном муравье» //Сборник докладов X международной конференции по мягким вычислениям и измерениям. СПбГЭТУ «ЛЭТИ». Т.2. 2007, с. 88–91.
10. *Царев Ф. Н., Шалыто А. А.* Применение генетических алгоритмов для построения автоматов с минимальным числом состояний для задачи об «Умном муравье» / Тезисы научно-технической конференции «Научно-программное обеспечение в образовании и научных исследованиях». СПбГПУ. 2008, с. 209–215.

11. *Tsarev F., Davydov A., Sokolov D.,* Application of Genetic Algorithms for Construction of Moore Automaton and Systems of Interacting Mealy Automata in "Artificical Ant" Problem /Proceeding of the Second Spring Young Researchers' Colloquium on Software Engineering (SYRCoSE'2008). SPbSU. 2008. Vol.1, pp.51–54.
12. *Tsarev F., Davydov A., Sokolov D., Shalyto A.* Application of Genetic Programming for Generation of Controllers represented by Automata / Preprints of the 13th IFAC Symposium on Information Control Problems in Manufacturing. Moscow. 2009, pp. 684–689.
13. *Царев Ф. Н.* Построение автоматов управления системами со сложным поведением на основе тестов с помощью генетического программирования /Сборник докладов международной конференции по мягким вычислениям и измерениям (SCM`2009). СПбГЭТУ «ЛЭТИ». Т. 1, с. 231–234.
14. *Царев Ф. Н.* Метод построения автоматов управления системами со сложным поведением на основе тестов с помощью генетического программирования / Материалы Международной научной конференции «Компьютерные науки и информационные технологии» памяти А. М. Богомолова. Саратов: СГУ. 2009, с. 216–219.
15. *Tsarev F., Alexandrov A., Sergushichev A., Kazakov S.* Genetic Algorithm for Induction of Finite Automaton with Continuous and Discrete Output Actions / Proceedings of the 2011 GECCO Conference Companion on Genetic and Evolutionary Computation. NY.: ACM. 2011, pp. 775 – 778.
16. *Царев Ф. Н., Егоров К. В.* Совместное применение генетического программирования и верификации моделей для построения автоматов управления системами со сложным поведением /Сборник трудов конференции «Информационные технологии и системы» (ИТИС`09). М.: Институт проблем передачи информации им. А. А. Харкевича РАН. 2009, с. 77–82.
17. *Царев Ф. Н., Александров А. В., Казаков С. В., Сергушичев А. А.* Генетическое программирование на основе обучающих примеров для построения конечных автоматов управления моделью беспилотного самолета /Сборник докладов международной конференции по мягким вычислениям и измерениям (SCM`2010). СПбГЭТУ «ЛЭТИ». Т. 1, с. 263–267.
18. *Царев Ф. Н., Егоров К. В., Парфенов В. Г.* Совместное применение генетического программирования и верификации моделей для построения автоматов управления системами со сложным поведением /Труды XVII Всероссийской научно-методической конференции «Телематика`2010». Т. 2. СПбГУ ИТМО. 2010, с. 344, 345.
19. *Царев Ф. Н., Егоров К. В.* Применение генетического программирования для построения автоматов управления системами со сложным поведением на основе верификации моделей и обучающих примеров // Сборник «Список-2011». Материалы второй межвузовской научной конференции по проблемам информатики». МатМех СПбГУ. 2011, с. 343–350.
20. *Tsarev F., Egorov K.* Finite State Machine Induction using Genetic Programming Based on Testing and Model Checking / Proceedings of the 2011 GECCO Conference

Companion on Genetic and Evolutionary Computation. NY.: ACM. 2011, pp. 759–762.

21. *Царев Ф. Н., Давыдов А. А., Соколов Д. О., Шалыто А. А.* Виртуальная лаборатория обучения генетическому программированию для генерации управляющих конечных автоматов / Сборник докладов III Международной научно-практической конференции «Современные информационные технологии и ИТ-образование». ВМК МГУ. М.: МАКС Пресс, 2008, с. 179–183.
22. *Царев Ф. Н., Тяхти А. С., Чебатуркин А. А., Шалыто А. А.* Виртуальная лаборатория для обучения методам искусственного интеллекта для генерации управляющих конечных автоматов /Сборник докладов IV Международной научно-практической конференции «Современные информационные технологии и ИТ-образование». М.: ИНТУИТ.РУ, МГУ. 2009, с. 222–227.
23. *Tsarev F., Chivilikhin D., Ulyantsev V.* Test-Based Extended Finite-State Machines Induction with Evolutionary Algorithms and Ant Colony Optimization / Proceedings of the 2012 GECCO Conference Companion on Genetic and Evolutionary Computation. NY. : ACM. 2012, pp. 603–606.