ПОСТРОЕНИЕ АВТОМАТОВ УПРАВЛЕНИЯ СИСТЕМАМИ СО СЛОЖНЫМ ПОВЕДЕНИЕМ НА ОСНОВЕ ВЕРИФИКАЦИИ И СЦЕНАРИЕВ РАБОТЫ

К. В. Егоров

аспирант кафедры компьютерных технологий;
kegorof@gmail.com

Ф. Н. Царев

аспирант кафедры компьютерных технологий;
fedor.tsarev@gmail.com

А. А. Шалыто

д.т.н., профессор, заведующий кафедрой технологии программирования;
shalyto@mail.ifmo.ru

Санкт-Петербургский государственный университет информационных технологий, механики и оптики

Аннотация: Настоящая работа описывает способ построения автоматов управления на основе верификации, позитивных и негативных сценариев работы. Предлагается применять генетический алгоритм, где особь в каждом поколении — это конечный автомат, а временные утверждения и сценарии работы учитываются при вычислении функции приспособленности, мутации и скрещивании.

Введение

Автоматное программирование — это парадигма программирования, в рамках которой программы предлагается проектировать в виде совокупности взаимодействующих автоматизированных объектов управления [1]. В автоматных программах выделяют три типа объектов: поставщики событий, система управления и объекты управления. Система управления представляет собой конечный автомат или систему взаимодействующих конечных автоматов. Поставщики событий генерируют события, а система управления по каждому событию может совершать переход, считывая значения входных переменных у объектов управления для проверки условия перехода.

Существуют различные способы построения автоматов управления со сложным поведением. Чаще всего такие системы строятся эвристически, но они могут содержать ошибки и требуют дополнительных проверок. В работе [2] был предложен способ построения автоматов с помощью генетического программирования на основе тестовых примеров. Однако такой вариант построения конечных автоматов требовал дополнительной валидации и верификации, а в случае обнаружения ошибки, необходимо было изменять тестовые примеры и заново строить систему. В работах [3–5] было предло-

жено строить систему на основе обучающих примеров совместно с темпоральными формулами. Формулы записываются на языке логики линейного времени (*Linear Temporal Logic, LTL*) и позволяют утверждать, что построенная система соответствует заявленной спецификации. Результат верификации учитывается при мутации, скрещивании и при вычислении функции приспособленности, причем вклад каждой темпоральной формулы — значение на отрезке [0, 1]. При этом: 0 — формула нарушается сразу же в стартовом состоянии, 1 — формула выполняется.

Построение автоматов управления на основе сценариев работы

Позитивный сценарий работы представляет собой последовательность пар: входное воздействие и соответствующий ему список выходных воздействий. Такой сценарий должен выполняться в требуемом конечном автомате. В тестовых примерах [2] не было однозначного соответствия между входными воздействиями и выходными, каждый тест записывался как входная последовательность и ожидаемая выходная последовательность воздействий. Теперь же предлагается перейти к интуитивно понятному представлению теста, когда мы заранее знаем список действий на каждое из событий. Это в некоторой степени сужение задачи, которое позволяет ускорить построение конечного автомата управления.

Негативный сценарий представляет собой последовательность входных воздействий, которая, в противоположность позитивной последовательности, не должна выполняться в автомате управления. Причем все префиксы негативного сценария выполняются, только последний переход не должен совершаться. Как неоднократно замечалось в предыдущих работах [4, 5], нельзя утверждать о правильности построенной системы только на основе тестов. Это же утверждение переносится и на сценарии работы (позитивные и негативные), так как они не имеют такой же выразительности как временные утверждения.

Особь в каждом поколении представляет собой конечный автомат с событиями на переходах и с переменным числом выходных воздействий, подобно методу на основе тестовых примеров из работы [2]. Выходные воздействия для каждого из переходов определяются алгоритмом расстановки пометок. Его основная идея состоит в том, что среди всех полностью или частично проходящих позитивных сценариев работы выбирается та последовательность выходных воздействий для выбранного перехода, которая чаще всего встречается.

Темпоральные формулы и сценарии работы используются при вычислении функции приспособленности, при скрещивании и при мутации. Вклад позитивных сценариев вычисляется при помощи редакционного расстояния [6]. Для его вычисления выполняются следующие действия: на вход авто-

мату подается каждая из последовательностей *Input[i]*. Обозначим последовательность выходных воздействий, которую сгенерировал автомат на входе *Input[i]*, как *Output[i]*. После этого вычисляется величина FF_{ptest}:

$$FF_{ptest} = \frac{\sum_{i=1}^{n} \left(1 - \frac{ED(Output[i], Answer[i])}{\max(|Output[i]|, |Answer[i]|)}\right)}{n}.$$

Здесь как *ED(A, B)* обозначено редакционное расстояние между строками *A* и *B*, как *Answer[i]* обозначена эталонная выходная последовательность, которую должен генерировать автомат на входе *Input[i]*. Отметим, что значения этой функции лежат в пределах от 0 до 1, при этом, чем «лучше» автомат соответствует позитивному сценарию, тем больше значение функции приспособленности.

Вклад в функцию приспособленности каждого негативного сценария дискретен. Если существует последовательность событий в конечном автомате, на которой сценарий выполняется, значит он проходит, и его вклад — –1. Если последовательности не существует, значит вклад сценария равен 0. Вклад всех негативных сценариев считается как отношение суммы вкладов каждого из сценария к общему числу позитивных сценариев. Конечная формула функции приспособленности принимает вид:

$$FF = FF_{ptest} + FF_{ptest} \times FF_{LTL} - FF_{ptest} \times FF_{ntest}.$$

Здесь FF_{ptest} — вклад позитивных сценариев работы, FF_{LTL} — вклад LTL-формул, FF_{ntest} — вклад негативных сценариев. Каждый из вкладов находится в интервале [0, 1].

Скрещивание двух автоматов управления описано в работах [2, 4]. Данный процесс может учитывать как LTL-формулы, так и сценарии работы (позитивные и негативные). Скрещивание по тестам [2] переносится на позитивные сценарии без изменений, его основная идея заключается в том, что выбирается несколько лучше всего проходящих сценариев, и переходы, задействованные в выбранных сценариях, переходят в новую особь из двух родителей без изменений, а оставшиеся распределяются между потомками случайным образом.

Мутация может случайным образом изменить входное воздействие перехода, его конечное состояние или удалить переход конечного автомата. Переход, нарушающий LTL-формулу или позволяющий пройти негативному сценарию, с большей вероятностью подвергается мутации, чем остальные.

Экспериментальные исследования

Были проведены экспериментальные исследования на примере автомата управления дверьми лифта из работы [4]. Было проведено 1000 построений

для каждого из перечисленных выше методов. При использовании только тестовых примеров построенный автомат менее чем в 1% случаев соответствовал спецификации, при использовании совместно верификации и тестовых примеров среднее число вычислений функции приспособленности оказалось равным — 827 857. При совместном использовании тестов и контрактов [5] — 710 882. При использовании верификации совместно с позитивными и негативными сценариями работы — 161 592.

Заключение

В работе исследована возможность применения верификации совместно с позитивными и негативными сценариями работы для автоматического построения автоматов управления систем со сложным поведением, предложен метод мутации и скрещивания на основе негативных сценариев работы, проведено экспериментальное исследование метода на примере построения автомата управления дверьми лифта.

В результате проведенных экспериментов было показано, что применение сценариев работы приводит к ускорению построения автомата управления. Однако заметим, что как тестовыми примерами, так и сценариями работы не всегда можно полностью описать поведение системы, так что целесообразно использовать их совместно с LTL-формулами.

Литература

1. *Поликарпова Н. И., Шалыто А. А.* Автоматное программирование. СПб.: Питер, 2010.
2. *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». Саратов: СГУ, 2009. С. 216–219.
3. *Johnson C.* Genetic Programming with Fitness based on Model Checking. Lecture Notes in Computer Science. Springer Berlin/Heidelberg, 2007. V. 4445. Pp. 114–124.
4. *Егоров К. В., Царев Ф. Н., Шалыто А. А.* Применение генетического программирования для построения автоматов управления системами со сложным поведением на основе обучающих примеров и спецификации // Научно-технический вестник СПбГУ ИТМО. 2010. № 69. С. 81–85.
5. *Егоров К. В., Шалыто А. А.* Применение генетического программирования для построения автоматов управления системами со сложным поведением на основе контрактов и тестовых примеров // Сборник научных трудов VI Международной научно-практической конференции «Интегрированные модели и мягкие вычисления в искусственном интеллекте. М.: Физмалит, 2011. С. 610–615.
6. *Левенштейн В. И.* Двоичные коды с исправлением выпадений, вставок и замещений символов // Доклады Академии Наук СССР. 163.4. С. 845–848.