ПОСТРОЕНИЕ УПРАВЛЯЮЩИХ КОНЕЧНЫХ АВТОМАТОВ ПО СЦЕНАРИЯМ РАБОТЫ НА ОСНОВЕ РЕШЕНИЯ ЗАДАЧИ УДОВЛЕТВОРЕНИЯ ОГРАНИЧЕНИЙ[1]

Ульянцев В.И.

магистрант кафедры «Компьютерные технологии» НИУ ИТМО,
ulyantsev@rain.ifmo.ru

Научный руководитель —

А.А. Шалыто

д. т. н, проф., зав. каф. «Технологии программирования» НИУ ИТМО,
shalyto@mail.ifmo.ru

Аннотация: Настоящая работа является продолжением исследований в области поисковой программной инженерии. Предлагается алгоритм построения управляющих конечных автоматов по сценариям работы, основанный на решении задачи удовлетворения ограничений. Данный алгоритм предоставляет возможность построения автоматов, удовлетворяющих не только требованию непротиворечивости, но и требованию полноты.

Введение

Парадигма автоматного программирования [1] для реализации сущности со сложным поведением подразумевает выделение системы управления и объекта управления. На начальном этапе проектирования программы выделяются события, входные переменные и выходные воздействия. После этого проектирование программы может идти разными путями. Один из них состоит в составлении сценариев работы программы, по которым далее эвристически или автоматизировано строится управляющий автомат.

К управляющим автоматам зачастую предъявляются два требования:

- обязательное требование непротиворечивости — не должно быть двух переходов, исходящих из одного состояния управляющего автомата и одновременно выполнимых при некоторой комбинации события и входных переменных;
- необязательное требование полноты — любой комбинации события и входных переменных должен соответствовать переход в каждом состоянии.

Ранее авторами был предложен метод [2] построения автоматных программ, удовлетворяющих требованию непротиворечивости, но не удовлет-

[1] Исследование поддержано федеральной целевой программой «Научные и научно-педагогические кадры инновационной России» на 2009–2013 годы, соглашение 14.B37.21.0397.

воряющих требованию полноты. В настоящей работе предлагается метод, не обладающий данным недостатком, основанный на решении задачи удовлетворения ограничений (constraint satisfaction problem — CSP).

Постановка задачи

Управляющим конечным автоматом называется детерминированный конечный автомат, каждый переход которого помечен *событием*, последовательностью *выходных воздействий* и *охранным условием*, представляющим собой логическую формулу от *входных переменных*.

Автомат получает события от так называемых *поставщиков событий* (в их роли могут выступать внешняя среда, интерфейс пользователя и т. д.) и генерирует выходные воздействия для *объекта управления*. При поступлении события автомат выполняет тот соответствующий ему переход, для которого охранное условие оказывается истинным. При выполнении перехода генерируются выходные воздействия, которыми он помечен, и автомат переходит в соответствующее состояние. Отметим, что состояния такого автомата не делятся на допускающие и недопускающие. Пример управляющего автомата приведен на рис. 1.

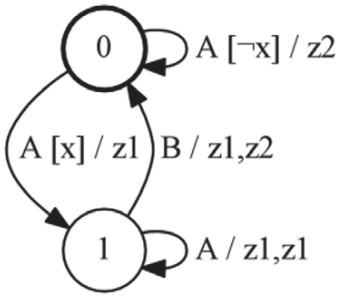

Рис. 1. Пример управляющего автомата

Для данного автомата множество входных событий равно $\{A, B\}$, охранные условия зависят от единственной логической входной переменной x, множество выходных воздействий равно $\{z1, z2\}$. Далее состояние автомата с номером 0 будем считать начальным.

В качестве исходных данных для построения управляющего конечного автомата используется множество *сценариев работы*. Сценарием работы будем называть последовательность $T_1 \ldots T_n$ троек $T_i = \langle e_i, f_i, A_i \rangle$, где e_i — входное событие, f_i — булева формула от входных переменных, задающая охранное условие, A_i — последовательность выходных воздействий. В дальнейшем тройки T_i будем называть *элементами сценария*.

Будем говорить, что автомат, находясь в состоянии *state*, *удовлетворяет элементу сценария* T_i, если из *state* исходит переход, помеченный событием e_i, последовательностью выходных воздействий A_i и охранным условием,

тождественно равным f_i как булева формула. Автомат удовлетворяет сценарию работы $T_1 \ldots T_n$, если он удовлетворяет каждому элементу данного сценария, находясь при этом в состояниях пути, образованного соответствующими переходами.

Решается задача построения управляющего конечного автомата, удовлетворяющего требованию полноты, с заданным числом состояний C по заданному множеству сценариев работы S_c, которым автомат должен удовлетворять.

Алгоритм построения управляющих автоматов

Предлагаемый алгоритм включает в себя пять основных этапов.

1. Построение дерева сценариев (рис. 2).
2. Построение графа совместимости вершин дерева сценариев.
3. Построение набора ограничений на целочисленные переменные, задающего требования к «раскраске» построенного графа и выражающей непротиворечивость и полноту системы переходов искомого автомата.
4. Использование стороннего пакета *Choco* [3], решающего задачу удовлетворения построенный ограничениям.
5. Построение автомата по найденной выполняющей подстановке.

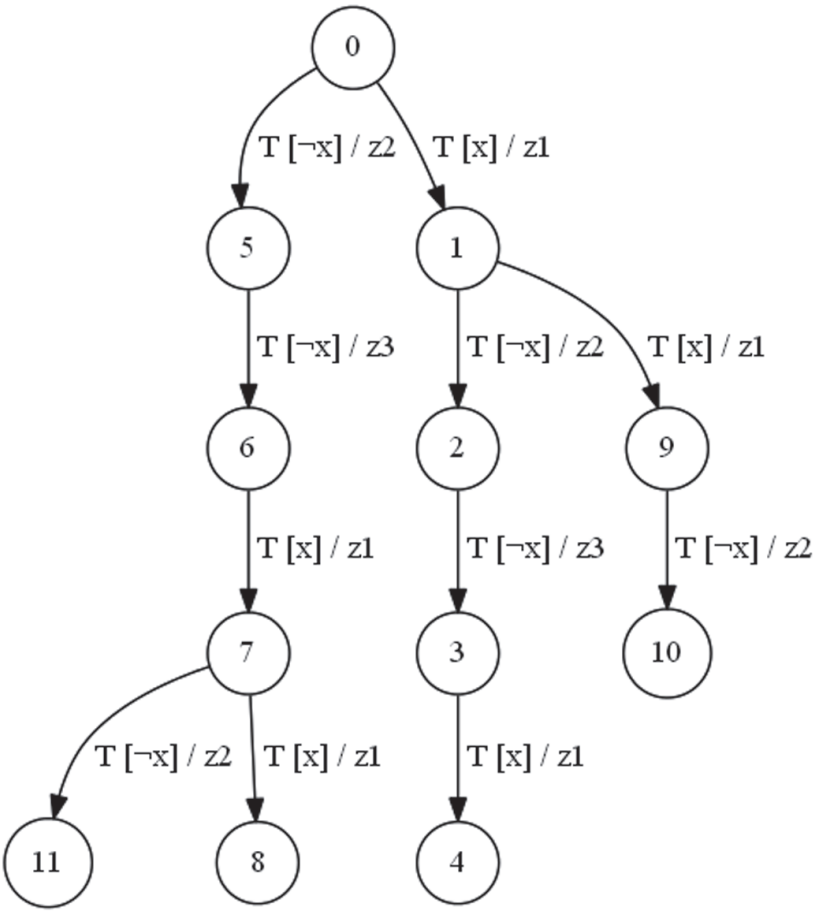

Рис. 2. Пример дерева сценариев

Опишем подробно этап построения набора ограничений, другие этапы описаны в [2]. Будем в дальнейшем множество всех условий переходов, встречающихся в сценариях работы S_c обозначать символом F. Множество входных событий e будем обозначать как E. Под F_e будем подразумевать множество условий переходов, помеченных входных событием e.

Для построения набора ограничений создадим, и будем использовать следующие целочисленные переменные.

1. Переменные x_v соответствуют цвету каждой вершины дерева сценариев v и принимают значения от 0 до $C-1$. Напомним, что вершины одного цвета будут объединены в одно состояние результирующего автомата.
2. Переменные $y_{i,e,f}$ являются вспомогательными для построения ограничений, задающих непротиворечивость (детерминированность) искомого управляющего автомата, и хранят в себе информацию о его переходах. Данные переменные являются вспомогательными, так как наличие в результирующем автомате переходов не определяется исходя из их значений. Используются переменные для каждого состояния i результирующего автомата (значение от 0 до $C-1$), каждого события e, каждого условия перехода f из множества F_e, встречающегося в сценариях. Каждая переменная принимает значения от 0 до $C-1$ и соответствует номеру состояния, в которое ведет переход искомого автомата из состояния i по событию e и условию перехода f.
3. Переменные $z_{i,e,f}$ используются для задания требования полноты искомого автомата и принимают значения 0 или 1, то есть по своей сути являются логическими. Данные переменные задаются для каждого состояния i результирующего автомата (значение от 0 до $C-1$), каждого события e, каждого условия перехода f из F_e, встречающегося в сценариях. Переменная $z_{i,e,f}$ равна 1, если существует вершина v в дереве сценариев, цвет которой равен i ($x_v=i$), и из нее ведет ребро, помеченное событием e и условием перехода f. В противном случае значение переменной равно 0. Таким образом, данные переменные хранят информацию о структуре переходов результирующего автомата, получающегося в результате объединения вершин дерева сценариев.

Составим набор ограничений на указанные переменные, задающий требования полноты и непротиворечивости искомого автомата.

1. $x_0=0$ — ограничение, задающее соответствие корня дерева сценариев начальному состоянию искомого автомата. В настоящем методе начальным состоянием автомата считается состояние с номером 0.
2. $x_v \neq x_u$ (для каждой несовместимой пары вершин дерева сценариев u и v, то есть соединенных ребром в графе совместимости) — ограничения, задающие непротиворечивость искомого автомата. Они гарантируют отсутствие различающих последовательностей, ведущих из одного состояния автомата. Число ограничений данного вида равно

числу ребер графа совместимости, то есть в худшем случае таких ограничений может быть $O(n^2)$, где n — число вершин дерева сценариев.

3. $(x_v = i) \Rightarrow (x_u = y_{i,e,f})$ (для каждого цвета i и каждого ребра дерева сценариев v_u, помеченного событием e и условием перехода f) — ограничения, задающие детерминированность искомого автомата. А именно, если вершине v присвоен цвет i, то цвет вершины u совпадает со значением переменной $y_{i,e,f}$, хранящей номер состояния автомата, в которое ведет переход из состояния i, помеченный событием e и условием перехода f. Число данных ограничений равно $C \cdot (n-1)$.
4. $z_{i,e,f} = 1 \Leftrightarrow (x_{v_1} = i \vee \ldots \vee x_{v_n} = i)$ (для каждого цвета i, входного события e, условия перехода f из F_e, встречающегося в заданных сценариях, и вершин $v_1 \ldots v_n$ дерева сценариев, из которых ведет ребро, помеченное событием e и условием перехода f) — ограничения, необходимые для правильного задания значений переменных $z_{i,e,f}$. Количество данных ограничений оценивается как $O(C \cdot |E| \cdot |F|)$, где как $|E|$ обозначено число событий, а как $|F|$ — число различных условий перехода.
5. $\left(\sum_{f \in F_e} (z_{i,e,f} \cdot c(f)) = 0 \right) \vee \left(\sum_{f \in F_e} (z_{i,e,f} \cdot c(f)) = 2^m \right)$ (для каждого цвета i и каждого события e) — ограничения, задающие требование полноты искомого автомата. Здесь как $c(f)$ обозначена функция, которая возвращает число выполняющих подстановок для булевой формулы f. При подсчете $c(f)$ считается, что булева формула зависит от всех m переменных, содержащихся в сценариях (например, $c(\text{true}) = 2^m$, а $c(x_1 \vee \neg x_2) = 2^{m-2}$. Сумма $\sum_{f \in F_e} (z_{i,e,f} \cdot c(f))$ равна числу комбинаций значений входных переменных values, для которых существует переход из состояния i, помеченный событием e и условием перехода f таким, что выполняется $f(values)$. Условие полноты искомого автомата выражается тем, что или для любого значения входных переменных найдется переход, или ни для одного из значений переменных перехода не существует. Заметим, что условие того, что для любого значения входных переменных найдется переход, можно выразить как $\sum_{f \in F_e} (z_{i,e,f} \cdot c(f)) = 2^m$, так как считается, что требование непротиворечивости уже выполнено, то есть все формулы, для которых выполняется $z_{i,e,f} = 1$, попарно не имеют общих выполняющих подстановок.

Приведенные ограничения пяти типов составляют набор, задающий требования непротиворечивости и полноты искомого автомата, удовлетворяющего сценариям S_c и содержащего C управляющих состояний.

Экспериментальное исследование

Было проведено экспериментальное исследование, сравнивающее ранее разработанный и предложенный методы. Рассматривались задачи построения автоматов с 4, 6, 8 и 10 состояниями по сценариям работы различной суммарной длины. Процент автоматов, удовлетворяющих требованию полноты, построенных при помощи ранее разработанного метода, уменьшался с ростом числа состояний — для 10 состояний такие автоматы находились менее, чем в половине случаем. Предлагаемый же алгоритм гарантированно находил управляющие автоматы, удовлетворяющие требованию полноты.

Заключение

Разработан метод автоматизированного построения управляющих конечных автоматов по сценариям работы. Этот метод основан на сведении указанной задачи к задаче выполнимости ограничений. Результаты экспериментов показали наличие большого числа входных данных, для которых ранее предложенный метод находил управляющий автомат, не удовлетворяющий требованию полноты, в то время как разработанный метод справлялся с задачей, пусть и с меньшей производительностью.

Литература

1. *Поликарпова Н.И., Шалыто А.А.* Автоматное программирование. СПб: Питер, 2009.
2. *Ulyantsev V., Tsarev F.* Extended Finite-State Machine Induction using SAT-Solver / Proceedings of the Tenth International Conference on Machine Learning and Applications, ICMLA 2011, Honolulu, HI, USA, 18–21 December 2011. IEEE Computer Society, 2011. Vol. 2. P. 346–349
3. Choco, java library for constraint satisfaction problems (CSP) and constraint programming (CP). http://www.emn.fr/z-info/choco-solver/