УДК 004.4'242

МЕТОД ПОСТРОЕНИЯ КОНЕЧНЫХ АВТОМАТОВ ВЕРХНЕГО УРОВНЯ ДЛЯ УПРАВЛЕНИЯ МОДЕЛЬЮ БЕСПИЛОТНОГО САМОЛЕТА НА ОСНОВЕ ОБУЧАЮЩИХ ПРИМЕРОВ

С.В. Казаков, Ф.Н. Царев, А.А. Шалыто

Для управления объектом со сложным поведением предлагается строить систему конечных автоматов, состоящую из автомата верхнего уровня, который необходим для переключения между режимами управления, и автоматов нижнего уровня, каждый из которых обеспечивает управление объектом в одном режиме. Данная работа продолжает тему построения автоматов нижнего уровня с помощью алгоритма генетического программирования на основе обучающих примеров: рассматривается построение автомата верхнего уровня с использованием обучающих примеров и уже построенных автоматов нижнего уровня, при этом генетическое программирование не используется. Приводятся результаты экспериментального исследования предложенного метода.

Ключевые слова: конечные автоматы, беспилотный самолет.

Введение

В настоящее время одной из актуальных задач является задача разработки программного обеспечения для управления объектами со сложным поведением. При этом для некоторых объектов управления такая задача является чрезвычайно сложной и трудно решаемой. Таким объектом может быть, например, транспортное средство (автомобили, самолеты, вертолеты, космические корабли) или другая сложная техника (роботы, техника на производстве).

Рассмотрим подробнее задачу управления беспилотным самолетом. Существует несколько подходов к ее решению. Один из них состоит в выделении «идеальной» траектории из нескольких полетов, выполненных человеком, и последующее следование ей. Такой подход описан в работе [1].

Другой подход – использование конечных автоматов для управления. В современной литературе чаше всего рассматриваются два подхода к управлению с использованием автоматов. В одном из них применяется один автомат, который осуществляет весь процесс управления, во втором – система автоматов, состоящая из головного автомата (автомата верхнего уровня), который необходим для переключения между режимами управления, и автоматов нижнего уровня, каждый из них обеспечивает управление объектом в одном режиме. Головной автомат взаимодействует с автоматами нижнего уровня за счет вложенности.

Построение автоматов нижнего уровня можно производить либо вручную (эвристически) [2], либо, например, с помощью алгоритмов генетического программирования. Некоторые из алгоритмов генетического программирования описаны в работах [3–6].

Автомат верхнего уровня можно построить аналогично. Один из алгоритмов генетического программирования для построения автоматов верхнего уровня описан в работе [7]. В этой работе вычисление функции приспособленности базировалось на моделировании поведения самолета во внешней среде, что для одной особи занимало около пяти минут на двух двухъядерных компьютерах, а для всего процесса – около двух недель, что является существенным недостатком этого подхода. С целью устранения указанного недостатка в работе [8] было предложено использовать обучающие примеры [9] как замену моделированию. Такой подход также основан на генетическом программировании, однако, в отличие от работы [7], он состоит в построении автоматов нижнего уровня, каждый из которых управлял объектом в одном режиме. При этом построение одного автомата занимало от 5 до 20 часов.

Эффективность использования обучающих примеров для построения автоматов нижнего уровня позволила авторам сделать предположение, что построение головного автомата на основе обучающих примеров также будет весьма эффективно. При этом авторы использовали третий подход к построению головного автомата, состоящий в разработке алгоритма его построения на основе обучающих примеров и уже построенных автоматов нижнего уровня. Вместе с автоматами нижнего уровня также должны быть заданы последовательности значений входных параметров для каждого автомата. Считается, что последовательность параметров для автомата нижнего уровня задает поведение беспилотного самолета под управлением данного автомата.

Подходы с использованием обучающих примеров позволяют при большом их числе избавиться от неточностей, допускаемых человеком при их записи.

Отметим, что используемый в данной работе объект со сложным поведением (беспилотный самолет) в дальнейшем может быть заменен на любой другой объект, который может управляться в нескольких режимах работы.

Взаимодействие беспилотного самолета с иерархической системой автоматов

Схема взаимодействия беспилотного самолета с иерархической системой автоматов приведена на рис. 1.

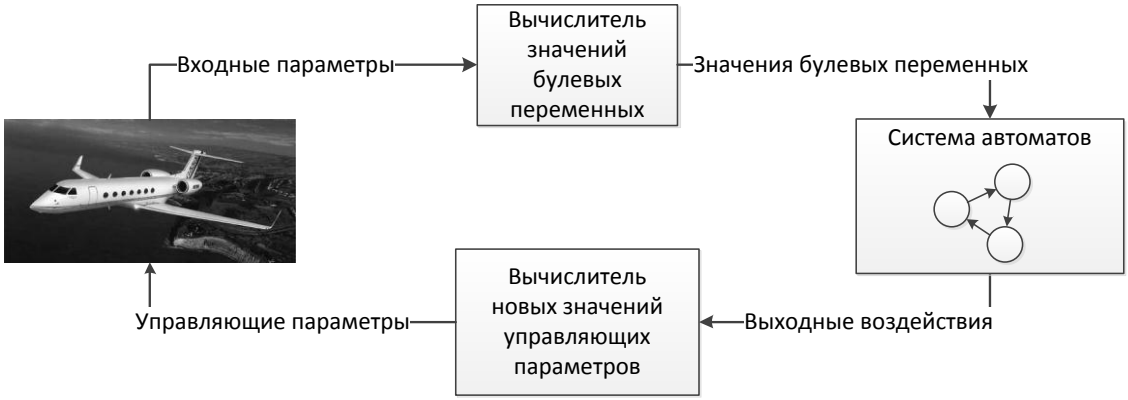

Рис. 1. Схема взаимодействия беспилотного самолета с иерархической системой автоматов

Эта схема, кроме самолета и иерархической системы автоматов, содержит также два вычислительных блока, первый из которых по значениям входных параметров формирует значения всех булевых переменных, используемых в автоматах, а второй по приращениям, формируемым автоматами нижнего уровня, вычисляет значения управляющих параметров.

Структура обучающего примера

Исходными данными для построения головного автомата являются набор обучающих примеров и построенные автоматы нижнего уровня, каждый из которых соответствует одному режиму управления самолетом. Обучающие примеры, задающие эталонное поведение, создаются человеком.

Обучающий пример состоит из двух последовательностей – входных и управляющих параметров. Эти последовательности разделены на фрагменты, соответствующие разным режимам полета самолета. При этом неизвестно, какой из режимов соответствует какому фрагменту последовательности.

Структура автомата верхнего уровня

Будем предполагать, что для каждого режима существует одно состояние автомата. Это состояние реализуется автоматом нижнего уровня. Таким образом, головной автомат содержит столько состояний, сколько используется режимов (сколько имеется автоматов нижнего уровня).

Так как автомат верхнего уровня непосредственно не управляет самолетом, а только обеспечивает переключение между режимами, то на его графе переходов дуги содержат только условия перехода и не содержат воздействий. Для обеспечения переходов будем использовать условия следующего вида: «[!]*x*1 & [!]*x*2 & … & [!]*x*k». Здесь *x*i – булевы переменные. Более сложные условия не использовались, так как для рассматриваемой задачи они не требовались, а также потому, что их трудно построить автоматически.

Алгоритм построения автомата верхнего уровня

Предлагаемый алгоритм состоит из двух этапов:

- идентификация режимов в каждом обучающем примере;
- определение дуг между состояниями автомата и условий на них.

Для идентификации режимов каждому фрагменту последовательности обучающего примера необходимо сопоставить автомат нижнего уровня. Это сопоставление осуществляется за счет вычисления редакционного расстояния между двумя последовательностями входных параметров – той последовательности параметров, которая записана в обучающем примере, и последовательности, которая задается вместе с каждым автоматом нижнего уровня. В результате идентификации для каждого фрагмента обучающего примера становится известен режим, соответствующий этому фрагменту.

В ходе выполнения второго этапа по результатам идентификации определяются дуги между состояниями автомата и условия на них. Для этого необходимо для каждых двух соседних фрагментов обучающего примера добавить дугу между состояниями, которые соответствуют этим фрагментам. Для каждой дуги автомата из обучающего примера можно выделить обучающий поднабор, который состоит из нескольких последовательностей входных параметров. Каждая такая последовательность разделена на две части, которые соответствуют выполнению и невыполнению условия перехода. Определение условия перехода выполняется с помощью перебора всех возможных условий указанного вида с проверкой его работоспособности на обучающем поднаборе.

Экспериментальная проверка

Для проверки эффективности и работоспособности предложенного метода была выбрана задача построения системы автоматов, управляющей беспилотным самолетом, который должен взлететь,

набрать высоту, далее выполнять команды, которые приходят с Земли, и в конечном итоге приземлиться. При этом возможен приход трех команд:

- выполнить мертвую петлю;
- начать снижаться для того, чтобы приземлиться;
- начать набирать высоту до необходимой для выполнения мертвой петли.

При этом последние две команды могут отменять друг друга, но команду выполнения мертвой петли отменить нельзя, пока она не будет выполнена.

Для записи обучающих примеров и проверки построенной системы автоматов был выбран свободно распространяемый кроссплатформенный симулятор FlightGear (http://www.flightgear.org), который применительно к настоящей работе позволяет осуществлять как ручное, так и программное управление моделью самолета.

Построение автоматов нижнего уровня

Управление беспилотным самолетом в рассматриваемой задаче должно осуществляться в следующих режимах: запуск двигателя; разгон; набор высоты (взлет); сбалансированный полет; мертвая петля; снижение; приземление; торможение.

При этом автоматы нижнего уровня были построены следующим образом:

- автомат для выполнения мертвой петли был «выращен» с помощью алгоритма генетического программирования, описанного в работе [8];
- автоматы для всех остальных режимов были построены вручную.

Построение автомата верхнего уровня

Для построения этого автомата было использовано 13 обучающих примеров. Число этих примеров было выбрано эвристически, однако экспериментальная проверка показала, что автомат верхнего уровня, построенный с их использованием, работает корректно. После этого вручную был сформирован набор булевых переменных, которые будут использоваться в автомате верхнего уровня на переходах.

На рис. 2 приведен автомат, построенный с помощью изложенного выше алгоритма. В этом автомате используются следующие булевы переменные:

- x0 – двигатель работает длительное время (более 0,5 с);
- x1 – высота меньше нулевого порога;
- x2 – мертвая петля была выполнена;
- x3 – пришла команда на выполнение мертвой петли;
- x4/x5 – пришла команда начать набирать высоту/снижаться;
- x6 – скорость относительно земли больше 59 км/ч;
- x7 – высота больше 1381 футов ≈ 400 метров;
- x8 – прошло 50 с после переключения в данный режим;
- x9 – прошло 28 с после переключения в данный режим;
- x10 – высота больше 196 футов ≈ 60 метров.

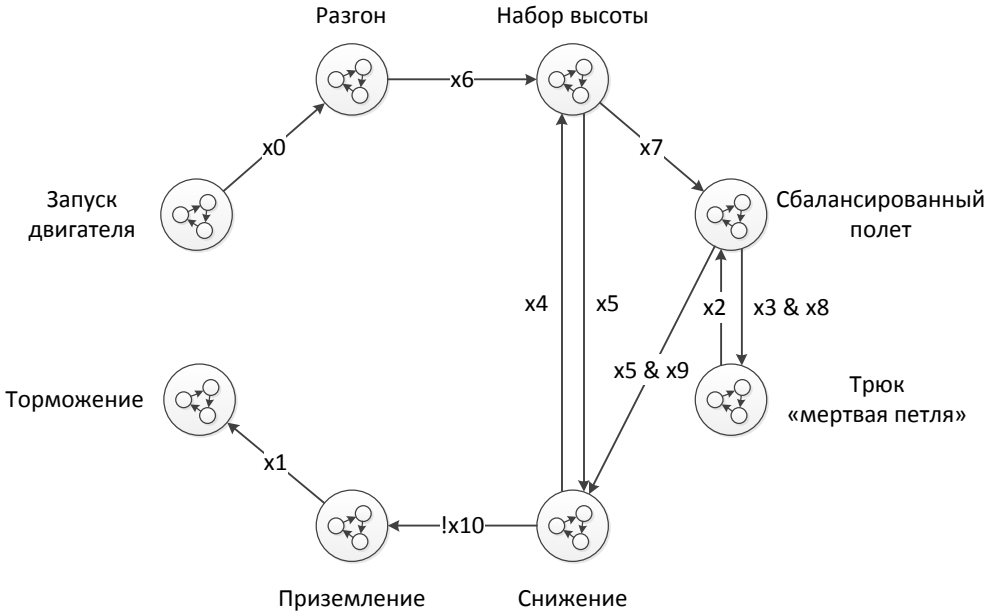

Рис. 2. Построенный автомат верхнего уровня

Оценка эффективности процесса построения автомата верхнего уровня

На основе изложенного выше алгоритма построения автомата верхнего уровня была написана программа на языке Java, которая также обеспечивает взаимодействие с симулятором в соответствии со схемой на рис. 1. Эта программа сначала строит автомат верхнего уровня, а затем обеспечивает реализацию схемы взаимодействия.

Вычисления производились на одном ядре компьютера с процессором Intel Core 2 Duo T7250 с тактовой частотой 2 ГГц под управлением операционной системы Microsoft Windows 7. При этом время построения автомата верхнего уровня составило менее двух минут.

Анализ построенного автомата

Видеозапись одного из полетов самолета под управлением построенной системы автоматов доступна по адресу http://www.youtube.com/watch?v=dq5AVzqXug0. Анализ этой видеозаписи показывает, что самолет ведет себя корректно во всех режимах.

В результате многократного наблюдения за полетом самолета в симуляторе не было обнаружено странностей в его поведении. В большинстве случаев полет проходил гладко, без каких-либо больших отклонений от предполагаемой траектории на всех режимах полета. На основании этого был сделан вывод о том, что вся система автоматов была построена корректно.

Заключение

В работе предложен метод построения автомата верхнего уровня на основе обучающих примеров и построенных автоматов нижнего уровня для каждого используемого режима. Благодаря использованию обучающих примеров появилась возможность отказаться от использования моделирования для оценки построенного автомата, что позволило уменьшить время его построения на несколько порядков.

Предложенный метод был апробирован на задаче построения системы автоматов для управления беспилотным самолетом во время выполнения всего процесса полета.

Статья подготовлена в рамках работ 1.6 и 2.10 «Разработка и апробация учебно-методического обеспечения для реализации проектного подхода к обучению студентов путем проведения инициативных научных исследований» проекта «Подготовка и переподготовка профильных специалистов на базе центров образования и разработок в сфере информационных технологий. Лот 1. «Подготовка и переподготовка профильных специалистов на базе центров образования и разработок в сфере информационных технологий в Северо-Западном Федеральном округе».

Литература

1. Coates A., Abbeel P., Ng A. Y. Learning for Control from Multiple Demonstrations // Proceedings of the 25th International Conference on Machine Learning. – Helsinki, 2008. – P. 144 – 151.

2. Paraschenko D., Shalyto A., Tsarev F. Modeling Technology for One Class of Multi-Agent Systems with Automata Based Programming // IEEE International Conference on Computational Intelligence for Measurement Systems and Applications (CIMSA 2006). – Spain, 2006. – P. 15 – 20 [Электронный ресурс]. – Режим доступа: http://is.ifmo.ru/science/CIMSA2006-1.pdf , своб.
3. Гладков Л.А., Курейчик В.В., Курейчик В.М. Генетические алгоритмы. – М.: Физматлит, 2006. – 366 с.
4. Рассел С., Норвиг П. Искусственный интеллект: современный подход. – М.: Вильямс, 2006. – 1408 с.
5. Koza J. R. Genetic programming: on the programming of computers by means of natural selection. – MIT Press, 1992. – 819 p.
6. Курейчик В.М. Генетические алгоритмы. Состояние. Проблемы. Перспективы // Известия РАН. Теория и системы управления. – 1999. – № 1. – С. 144–160.
7. Поликарпова Н.И., Точилин В.Н., Шалыто А.А. Метод сокращенных таблиц для генерации автоматов с большим числом входных переменных на основе генетического программирования // Известия РАН. Теория и системы управления. – 2010. –№ 2. – С. 100–117.
8. Александров А.В., Казаков С.В., Сергушичев А.А., Царев Ф.Н., Шалыто А.А. Применение генетического программирования на основе обучающих примеров для генерации конечных автоматов, управляющих объектами со сложным поведением // Научно-технический вестник СПбГУ ИТМО. – 2011. – № 2. – С. 3–11.
9. Царев Ф.Н. Метод построения управляющих конечных автоматов на основе тестовых примеров с помощью генетического программирования // Информационно-управляющие системы. – 2010. – № 5. – С. 31–36.

Казаков Сергей Владимирович – Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики, магистрант, svkazakov@rain.ifmo.ru

Царев Федор Николаевич – Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики, аспирант, fedor.tsarev@gmail.com

Шалыто Анатолий Абрамович – Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики, доктор технических наук, профессор, зав. кафедрой, shalyto@mail.ifmo.ru