1 ТЕХНОЛОГИИ АВТОМАТНОГО ПРОГРАММИРОВАНИЯ И ИСКУССТВЕННОГО ИНТЕЛЛЕКТА

УДК 004.4'242

ГЕНЕРАЦИЯ КОНЕЧНЫХ АВТОМАТОВ ДЛЯ УПРАВЛЕНИЯ МОДЕЛЬЮ БЕСПИЛОТНОГО САМОЛЕТА

А.В. Александров, С.В. Казаков, А.А. Сергушичев, Ф.Н. Царев, А.А. Шалыто

Для генерации автоматов управления объектами со сложным поведением предлагается применять генетическое программирование. При этом вместо известного подхода, в котором для оценки качества управляющего автомата используется моделирование, занимающее обычно большое время, применяется подход, в котором используется сравнение поведения автоматов с поведением, обеспечиваемым за счет управления человеком. Особенность рассматриваемого подхода состоит в том, что он позволяет использовать объекты управления не только с дискретными, но и с непрерывными параметрами. Применение подхода иллюстрируется на примере создания автомата, управляющего моделью самолета в режиме «мертвая петля».

Ключевые слова: конечные автоматы, генетическое программирование, беспилотный самолет.

Введение

В последнее время для программирования систем со сложным поведением все шире применяется автоматное программирование, в рамках которого поведение программ описывается с помощью конечных детерминированных автоматов (в дальнейшем – автоматов) [1].

В автоматном программировании программы предлагается строить в виде набора автоматизированных объектов управления. Каждый такой объект состоит из объекта управления и системы управления (системы управляющих автоматов). Система автоматов получает на вход события и переменные из внешней среды и от объекта управления. На основании этих данных система управления вырабатывает выходные воздействия для объекта управления.

Для многих задач управляющие автоматы удается строить эвристически, но существуют задачи, для которых такое построение невозможно или затруднительно. К этому классу относятся, например, задача «Умный муравей» [2–4], задача «Умный муравей-3» [5] и задача об управлении моделью беспилотного летательного аппарата [6].

Существует несколько подходов к решению последней задачи. Один из них состоит в выделении «идеальной» траектории из нескольких полетов, выполненных человеком, и последующем следовании ей. Такой подход описан в работе [7].

Другой подход – использование автоматов для управления беспилотным летательным аппаратом и построение таких автоматов с помощью генетических алгоритмов, описанных в работах [8–12].

В работе [13] для генерации конечного автомата верхнего уровня, управляющего моделью беспилотного самолета, применяется алгоритм генетического программирования, основанный на использовании метода сокращенных таблиц для представления конечных автоматов. При этом вычисление функции приспособленности базируется на моделировании поведения самолета во внешней среде, которое занимает достаточно большое время. Этот алгоритм, как и описываемый в данной работе, относится к генетическому программированию, так как особь обладает сложной структурой [10].

Цель настоящей работы – разработка лишенного указанного недостатка метода, основанного на генетическом программировании, для построения автоматов, управляющих объектами со сложным поведением. Для этого предлагается строить автоматы управления таким объектами отдельно для каждого из их режимов работы с помощью генетического программирования, используя обучающие примеры, создаваемые для каждого режима. Задавая большое число обучающих примеров, можно, как и в работе [7], избавиться от неточностей, допускаемых человеком при управлении. Такой подход является развитием идей, предложенных в работе [14], в которой рассматривались только объекты, управляемые дискретными выходными воздействиями.

В настоящей работе указанный подход обобщается на объекты, которые управляются с помощью не только дискретных, но и непрерывных воздействий. При этом в качестве примера объекта со сложным поведением рассмотрена модель самолета в режиме «мертвая петля».

Для объединения автоматов, управляющих режимами, с помощью метода сокращенных таблиц [13] или эвристического метода строится головной автомат, каждое из состояний которого соответствует одному из режимов. В результате формируется иерархическая система взаимодействующих автоматов. Вопрос о построении головного автомата в данной работе не рассматривается.

Кроме более высокого быстродействия, метод обладает еще одним преимуществом по сравнению с вычислением функции приспособленности с помощью моделирования: в предложенном методе эту функцию не требуется изменять как при переходе от одного режима к другому, так и при переходе от одного объекта управления к другому.

Постановка задачи

Исходными данными для построения управляющего конечного автомата является набор обучающих примеров (тестов), структура которых подробно описана ниже. Тесты, задающие эталонное поведение, создаются человеком.

Задача алгоритма генетического программирования состоит в построении конечного автомата, который задает поведение объекта управления, наиболее близкое к эталонному.

Органы управления

Объект управления характеризуется набором органов управления, посредством воздействия на которые объектом можно управлять. Параметры, соответствующие органам управления, будем называть управляющими. Параметры некоторых органов управления могут принимать лишь конечное множество значений – такие органы называются дискретными. Параметры других органов управления характеризуются вещественными значениями – такие органы называются непрерывными. Будем также называть управляющие воздействия на непрерывные органы управления непрерывными, а управляющие воздействия на дискретные органы – дискретными. В данной работе в каждый момент времени значения управляющих параметров – это накопленные за предыдущие моменты времени значения управляющих воздействий.

Подход, предлагаемый в настоящей работе, ориентирован на объекты управления как с дискретными, так и с непрерывными органами управления. Если все органы дискретны, то следует использовать методику, изложенную в работе [14].

Непрерывное воздействие изменяет параметр органа управления на некоторую вещественную величину, а дискретное – устанавливает соответствующий орган управления в конкретное значение. Заметим, что последовательное выполнение действий с одним органом управления эквивалентно сумме воздействий на него в случае непрерывного воздействия и последнему воздействию – в случае дискретного.

Например, одним из непрерывных управляющих параметров является угол поворота руля самолета. Непрерывным воздействием на этот орган управления является изменение угла его поворота на некоторое значение. Тогда последовательность поворотов руля на x и y градусов эквивалентна повороту руля на $x + y$ градусов.

В свою очередь, примером дискретного исполнительного органа является стартер. Дискретное воздействие на него – включение или выключение. Тогда последовательность включений и выключений стартера эквивалентна последнему, совершенному над ним действию.

Структура теста

Тест T состоит из двух частей: $T.in$ и $T.ans$. Каждая из них является последовательностью длины $T.len$ – первая из них состоит из значений входных параметров, а вторая – из соответствующих эталонных значений управляющих параметров, которые записываются в ходе экспериментов, проводимых человеком.

Каждый элемент $T.in[t]$ последовательности входных параметров состоит из P чисел – значений этих параметров в момент времени t.

Элемент $T.ans[t]$ включает в себя два набора: $T.ans[t].d$ и $T.ans[t].c$. Последовательность $T.ans[t].d$ состоит из D дискретных, а $T.ans[t].c$ – из C непрерывных параметров. Таким образом, $T.ans[t]$ состоит из $D + C$ чисел.

Обозначим через $T.out$ набор управляющих параметров, выданных автоматом на тесте T. Структура $T.out$ совпадает со структурой $T.ans$.

Предлагаемый алгоритм генетического программирования

Существуют различные модели алгоритмов генетического программирования, например, классический и островной [15].

Классический алгоритм генетического программирования состоит из следующих шагов: 1. Создание начальной популяции. 2. Вычисление значений функции приспособленности. 3. Отбор особей для скрещивания. 4. Скрещивание. 5. Мутация.

Поколение, полученное после шага 5, становится текущим, и шаги 2–5 повторяются, пока не будет достигнуто условие останова.

Предлагаемый алгоритм отличается от классического тем, что перед шагом 2 для максимизации значения функции приспособленности введен дополнительный шаг – расстановка значений выходных

воздействий на переходах «скелета» автомата. Под «скелетом» понимается автомат, значения выходных воздействий которого будут определены в дальнейшем. «Скелет» задается в виде полной таблицы переходов – для каждого состояния и каждого условия истинности или ложности входной переменной задается переход. В скелете возможны два типа переходов. Для переходов первого типа символы выходных воздействий не указываются (предыдущие значения управляющих параметров не изменяются). Для переходов второго типа соответствующие символы указываются.

Ниже приведено описание этого алгоритма в порядке, удобном для понимания.

Особь в предлагаемом алгоритме генетического программирования

Конечный автомат в предлагаемом алгоритме представляется в виде объекта, который содержит описания переходов для каждого из состояний и номер начального состояния автомата, причем число состояний автомата фиксировано и является параметром алгоритма. Для каждого перехода задается условие, при котором он выполняется. Это условие имеет вид «x_i» или «$\neg x_i$», где x_i – i-ое утверждение (предикат) о состоянии объекта управления (например, «двигатель включен» для объекта управления «самолет»). Эти условия составляются вручную до запуска генетического алгоритма и не изменяются в течение его работы. При этом на переходах в особи не задаются значения выходных воздействий z_j, где z_j – j-ый кортеж изменений управляющих параметров. Таким образом, в особи кодируется только «скелет» автомата, а конкретные выходные воздействия, вырабатываемые на переходах, определяются с помощью алгоритма расстановки действий.

Работа автомата

Будем считать, что генерируемый автомат является синхронным – все такты его работы одинаковы, а их величина определяется инерционностью объекта управления. При этом под тактом его работы будем понимать запуск автомата на одном наборе входных параметров.

На каждом такте система управления получает значения всех входных параметров. После этого вычисляются логические значения всех условий объекта управления в порядке увеличения их номеров. После вычисления соответствующего значения условия оно подается на вход автомата. Автомат в течение одного такта может совершить несколько переходов, и результирующее воздействие автомата в каждый момент времени t может быть составлено из нескольких последовательно выполненных воздействий на отдельных переходах. Напомним, что каждый параметр результирующего воздействия – сумма воздействий в случае непрерывного параметра и последнее воздействие – в случае дискретного.

На рис.1 изображен возможный «скелет» автомата.

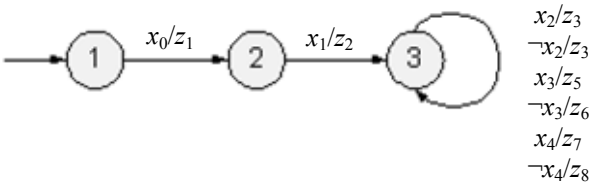

Рис. 1. «Скелет» автомата

Для данного скелета x_0–x_4 – заданные предикаты, а значения выходных воздействий (кортежей) $z_1, \ldots, z_8$ определяются в дальнейшем с помощью алгоритма их расстановки.

Функция приспособленности

Выбранная функция приспособленности отражает близость поведения автомата к эталонному и имеет вид

$$Fitness = 1 - \sqrt{\frac{1}{N}\sum_{i=1}^{N}\frac{Dist(T[i].out, T[i].ans)^2}{Dist(T[i].ans, 0)^2}}\,. \qquad (1)$$

Для вычисления ее значения на вход автомата подается каждая из N последовательностей $T[i].in$ – входной набор i-го теста и определяется последовательность значений управляющих параметров $T[i].out$, которую генерирует автомат.

Здесь под $Dist(out, ans)$ понимается «расстояние» между выходной и эталонной последовательностями управляющих параметров, которое вычисляется по формуле

$$Dist(out, ans) = \sqrt{\sum_{t=1}^{T.len}\left(\sum_{k=1}^{D}\left[out[t].d[k] \neq ans[t].d[k]\right] + \sum_{k=1}^{C}\left(out[t].c[k] - ans[t].c[k]\right)^2\right)}.$$

Как предлагалось выше, перед вычислением функции приспособленности к «скелету» автомата применяется алгоритм расстановки воздействий. Этот алгоритм расставляет значения воздействий на переходах так, чтобы максимизировать значение функции приспособленности при заданном «скелете» автомата.

Алгоритм расстановки воздействий – дополнительный шаг в алгоритме генетического программирования

При использовании этого алгоритма на переходах «скелета» автомата на его вход подается набор параметров из каждого теста, который был ранее назван входным. При этом запоминаются переходы, совершаемые автоматом. После этого определяются значения выходных воздействий, при которых функция приспособленности достигает максимума – это происходит, когда сумма

$$\sum_{i=1}^{N}\frac{Dist(T[i].out, T[i].ans)^2}{Dist(T[i].ans, 0)^2}$$

минимальна.

Суммы по каждому параметру можно минимизировать независимо друг от друга. Значит, необходимо минимизировать сумму

$$\sum_{i=1}^{N}\frac{\sum_{t=1}^{T[i].len}\left[T[i].out[t].d[k] \neq T[i].ans[t].d[k]\right]}{Dist(T[i].ans, 0)^2} \qquad (2)$$

для каждого k от 1 до D и сумму

$$\sum_{i=1}^{N}\frac{\sum_{t=1}^{T[i].len}\left(T[i].out[t].c[m] - T[i].ans[t].c[m]\right)^2}{Dist(T[i].ans, 0)^2} \qquad (3)$$

для m от 1 до C. Далее считаем индексы k и m зафиксированными.

Расстановка дискретных воздействий. Выделим из суммы (2) слагаемые, соответствующие каждому переходу:

$$\sum_{i=1}^{N}\sum_{j=1}^{n}\sum_{t\in Time_{i,j}}\frac{\left[T[i].out[t].d[k] \neq T[i].ans[t].d[k]\right]}{Dist(T[i].ans, 0)^2} = \sum_{j=1}^{n}\sum_{i=1}^{N}\sum_{t\in Time_{i,j}}\frac{\left[T[i].out[t].d[k] \neq T[i].ans[t].d[k]\right]}{Dist(T[i].ans, 0)^2}.$$

Здесь $Time_{i,j}$ – множество таких моментов времени t, при которых номер последнего выполненного перехода автомата, запущенного на тесте i, равен j, а n – число переходов в автомате.

Таким образом, для каждого j от 1 до n необходимо минимизировать выражение

$$\sum_{i=1}^{N}\sum_{t\in Time_{i,j}}\frac{\left[T[i].out[t].d[k] \neq T[i].ans[t].d[k]\right]}{Dist(T[i].ans, 0)^2}.$$

Зафиксируем j. Как и в случае, когда обучающий пример состоит из одного теста, $T[i].out[t].d[k]$ при $t \in Time_{i,j}$ равно значению k-го дискретного параметра на переходе j. Обозначим его через u.

Сгруппируем слагаемые с одинаковым значением $T[i].ans[t].d[k]$. При этом получим:

$$\sum_{h=1}^{G}\sum_{i=1}^{N}\sum_{t\in TimeV_{i,j,h}}\frac{\left[u \neq T[i].ans[t].d[k]\right]}{Dist(T[i].ans, 0)^2} = \sum_{h=1}^{G}\sum_{i=1}^{N}\frac{\left[u \neq v_h\right]}{Dist(T[i].ans, 0)^2}\sum_{t\in TimeV_{i,j,1}}1 =$$

$$= \sum_{h=1}^{G}\sum_{i=1}^{N}\frac{\left[u \neq v_h\right]}{Dist(T[i].ans, 0)^2}\left|TimeV_{i,j,h}\right| = \sum_{i=1}^{N}\frac{\sum_{h=1}^{G}\left[u \neq v_h\right]\cdot\left|TimeV_{i,j,h}\right|}{Dist(T[i].ans, 0)^2}.$$

Здесь v_h – h-ое значение k-го дискретного параметра, G – число возможных значений этого параметра, $TimeV_{i,j,h}$ – множество тех моментов времени t из $Time_{i,j}$, при которых $T[i].ans[t].d[k]$ равно v_h.

Выбрав v_h в качестве значения дискретного параметра на i-ом переходе, получим следующее значение суммы:

$$\sum_{i=1}^{N}\frac{1}{Dist(T[i].ans,0)^2}\left(\left|Time_{i,j}\right|-\left|TimeV_{i,j,h}\right|\right)=$$

$$=\sum_{i=1}^{N}\frac{1}{Dist(T[i].ans,0)^2}\left|Time_{i,j}\right|-\sum_{i=1}^{N}\frac{1}{Dist(T[i].ans,0)^2}\left|TimeV_{i,j,h}\right|. \qquad (4)$$

Для минимизации суммы (4) необходимо выбрать то значение v_h, при котором $\sum_{i=1}^{N}\frac{1}{Dist(T[i].ans,0)^2}\left|TimeV_{i,j,h}\right|$ максимально. Таким образом, на j-ом переходе в качестве значения k-го дискретного параметра следует выбрать v_h, где $h=\arg\max_h\sum_{i=1}^{N}\frac{1}{Dist(T[i].ans,0)^2}\left|TimeV_{i,j,h}\right|$.

Расстановка непрерывных воздействий. Ввиду того, что величина изменения m-го непрерывного параметра в момент времени t ($T[i].out[t].c[m]$) равна сумме изменений этого параметра на всех выполненных переходах к этому моменту, из формулы (3) получим:

$$S=\sum_{i=1}^{N}\frac{1}{Dist(T[i].ans,0)^2}\sum_{t=1}^{T[i].len}\left(\sum_{j=1}^{n}\alpha_{i,j}[t]u_j-T[i].ans[t].c[m]\right)^2.$$

Здесь, как и раньше, u_j – неизвестная величина изменения m-го непрерывного параметра на j-ом переходе, а $\alpha_{i,j}[t]$ – число выполнений j-го перехода к моменту времени t автомата, запущенного на тесте i.

Производная S по u_h имеет вид

$$S'_{u_h}=\sum_{i=1}^{N}\frac{1}{Dist(T[i].ans,0)^2}\sum_{t=1}^{T[i].len}2\alpha_{i,h}[t]\left(\sum_{j=1}^{n}\alpha_{i,j}[t]u_j-T[i].ans[t].c[m]\right).$$

Приравняв все S'_{u_h} к нулю, для каждого непрерывного параметра m получим систему из n уравнений:

$$\begin{cases}\sum_{j=1}^{n}\left(\sum_{i=1}^{N}\frac{1}{Dist(T[i].ans,0)^2}\sum_{t=1}^{T[i].len}\alpha_{i,1}[t]\alpha_{i,j}[t]\right)u_j=\sum_{i=1}^{N}\frac{1}{Dist(T[i].ans,0)^2}\sum_{t=1}^{T[i].len}T[i].ans[t].c[m]\alpha_{i,1}[t];\\ \sum_{j=1}^{n}\left(\sum_{i=1}^{N}\frac{1}{Dist(T[i].ans,0)^2}\sum_{t=1}^{T[i].len}\alpha_{i,2}[t]\alpha_{i,j}[t]\right)u_j=\sum_{i=1}^{N}\frac{1}{Dist(T[i].ans,0)^2}\sum_{t=1}^{T[i].len}T[i].ans[t].c[m]\alpha_{i,2}[t];\\ \dots\\ \sum_{j=1}^{n}\left(\sum_{i=1}^{N}\frac{1}{Dist(T[i].ans,0)^2}\sum_{t=1}^{T[i].len}\alpha_{i,n}[t]\alpha_{i,j}[t]\right)u_j=\sum_{i=1}^{N}\frac{1}{Dist(T[i].ans,0)^2}\sum_{t=1}^{T[i].len}T[i].ans[t].c[m]\alpha_{i,n}[t].\end{cases}$$

Каждая из этих систем решается так же, как и в случае одного теста. Полученные u_j – искомые величины изменений рассматриваемого непрерывного m-го параметра на переходе j.

Отметим, что линейность полученной системы уравнений определяется видом выбранной функции приспособленности. Первоначально авторами в качестве этой функции было выбрано выражение

$$Fitness=\frac{1}{N}\sum_{i=1}^{N}\sqrt{1-\frac{Dist(T[i].out,T[i].ans)^2}{Dist(T[i].ans,0)^2}},$$

которое приводило к системе нелинейных уравнений, что резко усложняло решение задачи.

Операторы алгоритма генетического программирования

Неотъемлемой частью любого алгоритма генетического программирования являются операторы отбора, мутации и скрещивания.

Оператор отбора. Этот оператор необходим для выделения из текущего поколения наиболее подходящих для решения задачи особей и добавления их в промежуточное поколение. Для сравнения особей применяется функция приспособленности, сопоставляющая каждой особи число, характеризующее, на-

сколько хорошо автомат, соответствующий особи, подходит для решения задачи. При этом отметим, что чем больше это число, тем лучшей считается особь.

В данной работе используется *турнирный метод* отбора [15]. В этом методе случайным образом выбирается k особей из текущего поколения. Среди них определяется лучшая особь, которая и добавляется в промежуточное поколение. Турнир проводится столько раз, сколько особей в поколении.

Оператор мутации. При применении этого оператора выполняется с заданной вероятностью каждое из действий:

- изменение начального состояния;
- добавление, удаление или изменение случайного перехода в «скелете» автомата.

Под изменением перехода понимается изменение состояния, в которое выполняется переход по условию, на случайно выбранное.

Оператор скрещивания. В скрещивании принимают участие две особи. Результатом применения оператора также являются две особи. Обозначим «родительские» особи как $P1$ и $P2$, а «дочерние» – $C1$ и $C2$. Тогда для стартовых состояний $C1.is$ и $C2.is$ справедливо одно из следующих утверждений:

- $C1.is = P1.is$ и $C2.is = P2.is$;
- $C1.is = P2.is$ и $C2.is = P1.is$.

Далее для каждой ячейки таблицы переходов $t[i][j]$ с равной вероятностью выбирается один из следующих вариантов:

- $C1.t[i][j] = P1.t[i][j]$ и $C2.t[i][j] = P2.t[i][j]$;
- $C1.t[i][j] = P2.t[i][j]$ и $C2.t[i][j] = P1.t[i][j]$.

Эффективность метода

Для проверки эффективности предложенного метода была поставлена задача генерации автомата, обеспечивающего управление самолетом в режиме «мертвая петля».

Взаимодействие сгенерированного автомата с моделью беспилотного самолета. Существуют симуляторы, моделирующие полет самолета. Авторами был выбран свободный кроссплатформенный симулятор *FlightGear* (http://www.flightgear.org), который применительно к настоящей работе позволяет осуществлять как ручное, так и автоматное управление моделью самолета. На рис. 2 представлен снимок экрана симулятора *FlightGear* в момент начала разгона.

Рис. 2. Снимок экрана симулятора *FlightGear* в момент начала разгона

Симулятор позволяет осуществлять сохранение параметров полета (скорость, направление полета и т.д.) и параметров самолета (положение руля, элеронов, состояние стартера и т.п.). Параметры полета являются входными параметрами для системы управления, а параметры самолета – управляющими, так как за счет их изменений выполняется управление самолетом. Значения управляющих параметров формируются автоматом (рис. 3).

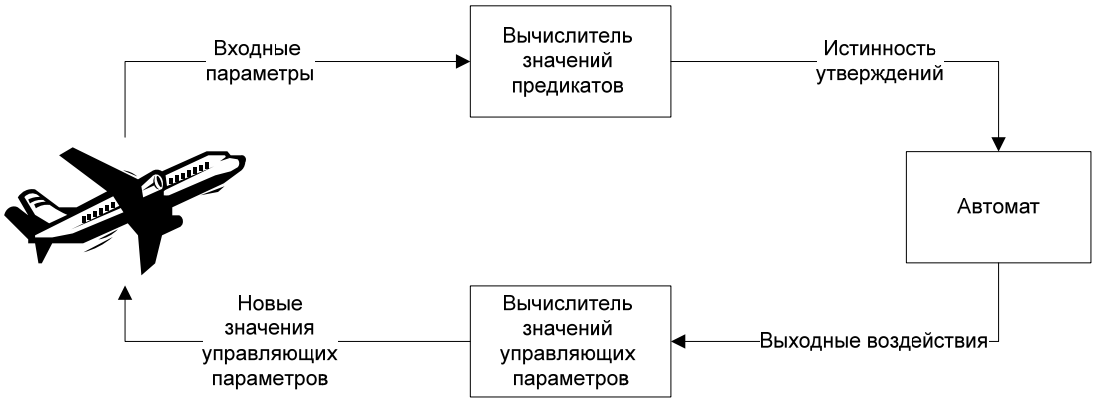

Рис. 3. Управление самолетом с помощью автомата

Задача генерации автомата, выполняющего «мертвую петлю». Эта задача может быть сформулирована следующим образом: требуется построить автомат, под управлением которого модель самолета делает мертвую петлю, а затем ровно пролетает несколько секунд.

Использование предлагаемого метода. Кратко рассмотрим основные шаги применения предлагаемого метода на рассматриваемом примере:

- сформирован необходимый и достаточный набор утверждений о состоянии самолета;
- записаны три набора тестов, которые рассматривались независимо друг от друга:
  - каждый набор состоял из 10 тестов;
  - каждый тест состоял из нескольких тысяч наборов входных и управляющих параметров;
  - опрос параметров производился 10 раз в секунду;
- алгоритм генетического программирования запускался для каждого из трех наборов тестов и каждого набора параметров алгоритма.

Перечислим параметры алгоритма и их значения:

- размер поколения – 100 особей;
- число состояний автомата – от двух до пяти;
- вероятность мутации – 0,5;
- метод отбора – турнирный;
- размер элиты – две особи;
- величина такта – 0,1 с.

Вычисления производились на одном ядре компьютера с процессором Intel Core 2 Duo T7250 с тактовой частотой 2 ГГц под управлением операционной системы Microsoft Windows XP. Язык программирования – Java.

В среднем время работы алгоритма составляло около 10 часов для одного набора параметров и одного набора тестов. При этом было вычислено около двух тысяч поколений. Таким образом, создание и обработка одного поколения в среднем занимала около 20 с или 0,2 с на один автомат, что значительно меньше, чем в работе [13], где это занимало около 5 минут.

Выбранные утверждения:

- x_0 – двигатель включен;
- x_1 – ускорение изменения направления движения самолета больше нуля;
- x_2 – скорость изменения направления движения самолета больше нуля;
- x_3 – величина отклонения от начального направления меньше одного градуса;
- x_4 – величина отклонения от начального направления больше нуля (отклонение влево);
- x_5 – ускорение изменения крена (угла наклона) больше нуля;
- x_6 – скорость изменения крена больше нуля;
- x_7 – крен самолета маленький (меньше одного градуса);
- x_8 – крен больше нуля (самолет завалился на правый бок);
- x_9 – ускорение изменения вертикальной скорости самолета больше нуля;
- x_{10} – скорость изменения вертикальной скорости больше нуля;
- x_{11} – вертикальная скорость маленькая (меньше 0,1 метра в секунду);
- x_{12} – вертикальная скорость больше нуля (самолет поднимается).

Управляющими органами являются магнето, стартер, дроссель, элероны, руль высоты и руль направления. Первые два органа являются дискретными, а остальные – непрерывными. При этом, напри-

мер, воздействие «включить стартер» является дискретным, а «повернуть руль на 0,5 градуса вправо» – непрерывным.

В результате запусков алгоритма генетического программирования было установлено:

- автоматы с тремя-четырьмя состояниями «ведут себя» достаточно хорошо;
- с увеличением числа состояний автомата его структура становится все менее логичной, а поведение ухудшается.

Записанные тесты. Приведем ссылки на некоторые видеозаписи полетов под управлением человека:

- успешное выполнение «мертвой петли» (этот полет вошел в обучающий набор) – http://www.youtube.com/watch?v=G5Kcx0ohpNo;
- неудачное выполнение – http://www.youtube.com/watch?v=OGVTch-a97A.

Полученные результаты. Было проведено около 50 запусков генетического алгоритма, в каждом из которых выбирался автомат с наибольшим значением функции приспособленности. Эти запуски отличались друг от друга используемыми параметрами и наборами тестов.

Полеты моделей самолетов, управляемых выбранными автоматами, были просмотрены авторами. После этого автомат, используемый в полете, больше других похожем на «идеальную» «мертвую петлю», был назван лучшим. Этот автомат имеет четыре состояния и 68 переходов. При этом отметим, что «идеальная» «мертвая петля» может отличаться от эталонной, которая выполняется вручную.

В процессе наблюдения за ходом выполнения «мертвой петли» под управлением лучшего автомата было установлено, что в зависимости от параметров среды и самолета при запуске автомата возможны три варианта выполнения «петли»:

1. В большинстве случаев модель самолета, как и при управлении вручную, выполняет одну «мертвую петлю» и летит дальше;
2. Иногда модель самолета может выполнить несколько «мертвых петель» с некоторым интервалом. Это происходит в случае, когда значения параметров модели самолета в конце выполнения «мертвой петли» схожи со значениями параметров в начале ее выполнения. Это приводит к тому, что если автомат после осуществления «петли» находится в том же состоянии, в котором был в начале, то поведение модели зацикливается. В ходе экспериментов авторы неоднократно наблюдали выполнение двух «мертвых петель» подряд. Выполнение большего числа «петель» не наблюдалось;
3. Модель может вообще не выполнить «мертвую петлю», так как автомат не справляется с управлением, что, правда, бывает крайне редко.

Определение условий, при которых выполняется каждый из этих вариантов полета, требует дальнейших исследований.

Пример полета самолета под управлением лучшего из полученных автоматов

Ниже приведены ссылки на видеозаписи трех вариантов реализации «мертвой петли» под управлением лучшего автомата:

1. Выполнение «мертвой петли», близкое к «идеальному» (http://www.youtube.com/watch?v=TzrLoJjjVTA);
2. Выполнение «мертвой петли» с заваливанием на левый борт с последующим выравниванием (http://www.youtube.com/watch?v=C6WV7x2bqE8);
3. Последовательное выполнение двух «мертвых петель» (http://www.youtube.com/watch?v=yFiG4yz67Ks).

Заключение

Разработан метод генетического программирования на основе обучающих примеров для построения конечных автоматов, управляющих объектом со сложным поведением. Выполнено экспериментальное исследование предложенного метода, и показано, что автоматически сгенерированный автомат может обеспечивать лучшее управление, чем ручное. В большинстве случаев автоматически сгенерированный автомат обеспечивает корректное выполнение задания. Показано, что предложенный метод позволяет значительно быстрее оценивать генерируемые автоматы по сравнению с прототипом [13]. Предложенный метод был применен к задаче управления моделью беспилотного самолета в режиме «мертвая петля».

Кроме более высокого быстродействия метода, еще одно его преимущество по сравнению с вычислением функции приспособленности с помощью моделирования состоит в том, что в предложенном методе эту функцию не требуется изменять как при переходе от одного режима к другому, так и при переходе от одного объекта к другому.

Исследование выполнено по Федеральной целевой программе «Научные и научно-педагогические кадры инновационной России на 2009–2013 годы» в рамках государственного контракта П1188 от 27 августа 2009 года.

Литература

1. Поликарпова Н.И., Шалыто А.А. Автоматное программирование. – СПб: Питер, 2010. – 176 с.
2. Angeline P., Pollack J. Evolutionary Module Acquisition // Proceedings of the Second Annual Conference on Evolutionary Programming. Cambridge: MIT Press, 1993. – P. 154–163.
3. Jefferson D., Collins R., Cooper C., Dyer M., Flowers M., Korf R., Taylor C., Wang A. The Genesys System: Evolution as a Theme in Artificial Life // Proceedings of Second Conference on Artificial Life. – MA: Addison-Wesley, 1992. – P. 549–578.
4. Царев Ф.Н., Шалыто А.А. Применение генетического программирования для генерации автомата в задаче об «Умном муравье» / Сборник трудов IV-ой Международной научно-практической конференции. – М.: Физматлит, 2007. – Т. 2. – С. 590–597.
5. Бедный Ю.Д., Шалыто А.А. Применение генетических алгоритмов для построения автоматов в задаче «Умный муравей». – СПбГУ ИТМО, 2007 [Электронный ресурс]. – Режим доступа: http://is.ifmo.ru/works/ant, свободный. Яз. рус. (дата обращения 09.02.2011).
6. Паращенко Д.А., Царев Ф.Н., Шалыто А.А. Технология моделирования одного класса мультиагентных систем на основе автоматного программирования на примере игры «Соревнование летающих тарелок». Проектная документация. – СПбГУ ИТМО, 2006. [Электронный ресурс]. – Режим доступа: http://is.ifmo.ru/unimod-projects/plates/, свободный. Яз. рус. (дата обращения 09.02.2011).
7. 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.
8. Гладков Л.А., Курейчик В.В., Курейчик В.М. Генетические алгоритмы. – М.: Физматлит, 2006.
9. Рассел С., Норвиг П. Искусственный интеллект: современный подход. – М.: Вильямс, 2006.
10. Koza J.R. Genetic programming: On the Programming of Computers by Means of Natural Selection. – Cambridge: MIT Press, 1992.
11. Курейчик В.М. Генетические алгоритмы. Состояние. Проблемы. Перспективы // Известия РАН. Теория и системы управления. – 1999. – № 1. – С. 144–160.
12. Курейчик В.М., Родзин С.И. Эволюционные алгоритмы: генетическое программирование // Известия РАН. Теория и системы управления. – 2002. – № 1. – С. 127–137.
13. 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.
14. Царев Ф.Н. Метод построения управляющих конечных автоматов на основе тестовых примеров с помощью генетического программирования // Информационно-управляющие системы. – 2010. – № 5. – С. 31–36.
15. Яминов Б. Генетические алгоритмы. – СПбГУ ИТМО, 2005. [Электронный ресурс]. – Режим доступа: http://rain.ifmo.ru/cat/view.php/theory/unsorted/genetic-2005, свободный. Яз. рус. (дата обращения 09.02.2011).

Александров Антон Вячеславович – Санкт-Петербургский государственный университет информационных технологий, механики и оптики, студент, alantbox@gmail.com

Казаков Сергей Владимирович – Санкт-Петербургский государственный университет информационных технологий, механики и оптики, студент, kazakov_sergey_v@mail.ru

Сергушичев Алексей Александрович – Санкт-Петербургский государственный университет информационных технологий, механики и оптики, студент, alsergbox@gmail.com

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

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