Материал опубликован в тезисах научно-технической конференции «Научно-программное обеспечение в образовании и научных исследованиях». СПбГУ ПУ. 2008, с. 209–215.

Ф. Н. Царев, А. А. Шалыто

ПРИМЕНЕНИЕ ГЕНЕТИЧЕСКИХ АЛГОРИТМОВ ДЛЯ ПОСТРОЕНИЯ АВТОМАТОВ С МИНИМАЛЬНЫМ ЧИСЛОМ СОСТОЯНИЙ ДЛЯ ЗАДАЧИ ОБ «УМНОМ МУРАВЬЕ»

ВВЕДЕНИЕ

В последнее время все шире применяется автоматное программирование, в рамках которого поведение программ описывается с помощью конечных детерминированных автоматов [1]. Для многих задач автоматы удается строить эвристически, однако существуют задачи, для которых такое построение затруднительно. К задачам этого класса относится, в частности, и задача об «Умном муравье» [2 – 4].

В указанных работах генерация автоматов выполнялась с помощью генетических алгоритмов [5, 6] на однопроцессорной ЭВМ. Однако построенные в этих публикациях автоматы содержат большое число состояний.

Цель настоящей работы – построение (также с помощью однопроцессорной ЭВМ) автомата с минимальным числом состояний для решения задачи об «Умном муравье».

ПОСТАНОВКА ЗАДАЧИ

Игра [2] происходит на поверхности тора размером 32 на 32 клетки (рис. 1). В некоторых из них (на рисунке обозначены черным цветом) находятся яблоки. Всего на поле 89 яблок. В клетке, помеченной меткой «Start», находится муравей, который смотрит на восток. В любой момент времени он занимает одну клетку и может смотреть в одном из четырех направлений (север, юг, запад, восток).

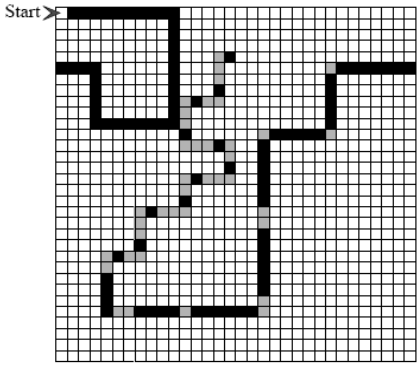


Рис. 1. Игровое поле

Муравей умеет определять, находится ли непосредственно перед ним яблоко. За один ход муравей может совершить одно из четырех действий: сделать шаг вперед, съедая яблоко, если оно там находится; повернуть налево; повернуть направо; ничего не делать.

Съеденные муравьем яблоки не восстанавливаются, а муравей жив на протяжении всей игры. Расположение яблок фиксировано. Игра длится 200 ходов, по истечении которых подсчитывается число яблок, съеденных муравьем. Это значение и есть результат игры. Цель игры – создать муравья, который за указанное число ходов съест как можно больше яблок.

Один из способов описания поведения муравья – конечный автомат с действиями на переходах (автомат Мили), в котором одна входная переменная булева типа (находится ли яблоко перед муравьем), а множество выходных воздействий указано выше.

Эвристически построенный автомат из работы [3], граф переходов которого изображен на рис. 2, описывает поведение муравья, который съедает 81 яблоко за 200 ходов, а все яблоки – за 314 ходов.

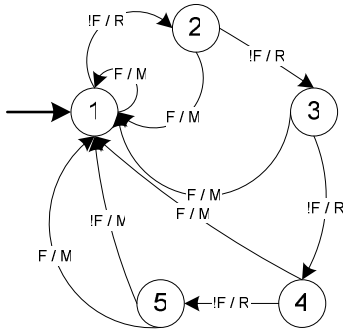


Рис. 2. Простой автомат, описывающий поведение муравья

Поясним используемые на рис. 2 обозначения. Пометки на переходах имеют формат условие / действие. Условия обозначаются следующим образом: F – перед муравьем есть еда; !F – перед муравьем нет еды.

Для действий приняты следующие обозначения: M – «Сделать шаг вперед»; L – «Повернуть налево»; R – «Повернуть направо»; N – «Ничего не делать». Последнее действие в дальнейшем не используется.

В работах [2–4] для построения автоматов, решающих описанную задачу, применяются генетические алгоритмы. С их помощью были построены автоматы, позволяющие муравью съесть все яблоки. Эти автоматы содержат тринадцать [2], одиннадцать [3] и восемь [4] состояний.

В настоящей работе для построения автоматов сравниваются два подхода: генетическое программирование [7] и перебор. Приведено описание алгоритма генетического программирования, с помощью которого приблизительно за **шесть часов** построен автомат с **семью** состояниями, позволяющий муравью съесть все яблоки. Описан также алгоритм перебора, с помощью которого были перебраны **все** автоматы из шести и менее состояний, и для каждого числа состояний установлено максимальное число яблок, которое может съесть муравей.

ОПИСАНИЕ АЛГОРИТМА ГЕНЕТИЧЕСКОГО ПРОГРАММИРОВАНИЯ

Каждая особь представляет собой некоторый автомат, описывающий поведение муравья. Хромосома особи состоит из номера начального состояния и описаний всех состояний автомата. Описание одного состояния содержит описания двух переходов, соответствующих тому, что перед муравьем либо есть яблоко, либо нет. Описание перехода состоит из номера состояния, в которое автомат переходит, и действия, выполняемого на этом переходе.

Генерация начального поколения. Начальное поколение состоит из фиксированного числа случайно сгенерированных автоматов. Все автоматы в поколении имеют одинаковое наперед заданное число состояний.

Мутация. При мутации случайно выбирается один из четырех равновероятных вариантов изменения: начального состояния; действия на переходе; состояния, в которое ведет переход; условия на переходе.

Скрещивание. Оператор скрещивания подробно описан в работе [8].

Формирование следующего поколения. В качестве основной стратегии формирования следующего поколения используется элитизм [9]. При обработке текущего поколения отбрасываются все особи, кроме нескольких наиболее приспособленных. Доля выживающих особей постоянна для каждого поколения и является одним из параметров алгоритма.

Эти особи переходят в следующее поколение. После этого оно дополняется до требуемого размера следующим образом: пока оно не заполнено, выбираются две особи из текущего поколения, и они с некоторой вероятностью скрещиваются или мутируют. Обе особи, полученные в результате мутации или скрещивания, добавляются в новое поколение.

Кроме этого, если на протяжении достаточно большого числа поколений не происходит увеличения приспособленности, то применяются «малая» и «большая» мутации поколения. При «малой» мутации поколения ко всем особям, кроме 10% лучших, применяется оператор мутации. При «большой» мутации каждая особь либо мутирует, либо заменяется на случайно сгенерированную.

Число поколений до «малой» и «большой» мутации постоянно во время работы алгоритма.

Вычисление функции приспособленности. Функция приспособленности особи (автомата) равна $F + \frac{200 - T}{200}$, где F – число яблок, съедаемое за 200 ходов муравьем, поведение которого задается этим автоматом, а T – номер хода, на котором муравей съедает последнее яблоко.

ОПИСАНИЕ АЛГОРИТМА ПЕРЕБОРА АВТОМАТОВ

В работе [8] рассмотрен полный перебор автоматов, и показано, что даже при пяти состояниях их перебор на однопроцессорной машине практически невозможен. Далее описан более эффективный алгоритм полного перебора, с помощью которого были найдены оптимальные автоматы с пятью и шестью состояниями.

Этот алгоритм имеет рекурсивную структуру. В каждый момент времени перебираемый автомат уже частично построен – некоторые его переходы определены. За счет этого можно говорить, что обработка нескольких автоматов идет параллельно – один частично построенный автомат соответствует нескольким полностью построенными. Кроме этого, управляющий муравьем автомат находится в некотором состоянии, а муравей – в некоторой клетке поля, смотрит в некоторую сторону, и известно состояние поля – в каких его клетках находятся яблоки.

На основании этих данных можно определить, из какого состояния (текущее состояние автомата) и по какому значению входной переменной необходимо сделать переход (есть ли яблоко перед муравьем). Если соответствующий переход определен, то он выполняется.

В противном случае выполняется перебор всех возможных вариантов определения этого перехода автомата и происходит рекурсивный вызов описанной процедуры. Для перебора вариантов перехода выполняется перебор состояний и перебор выходных воздействий с целью определения состояния, в которое ведет переход, и выходного воздействия, помечающего его. Этим состоянием может быть либо уже существующее состояние, либо новое. Если добавляется новое состояние, то оно получает наименьший неиспользуемый номер.

Кроме этого, перебор ускоряется с помощью следующего отсечения – если муравей, управляемый рассматриваемый автоматом, уже точно не сможет съесть еды больше, чем наилучший из рассмотренных (например, если осталось слишком мало ходов), то рассмотрение этого автомата прекращается.

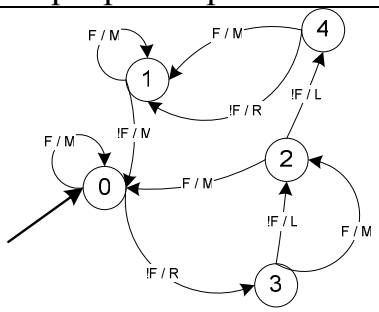
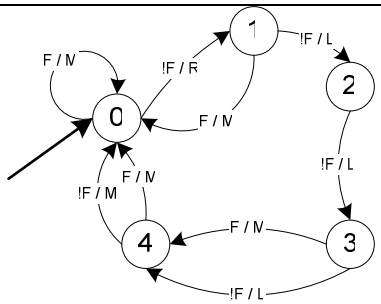
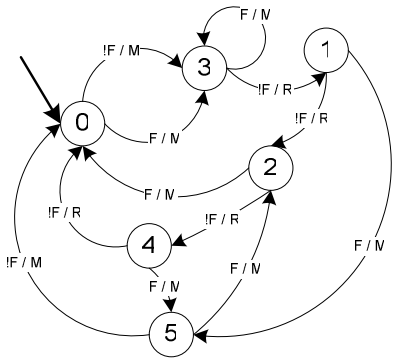
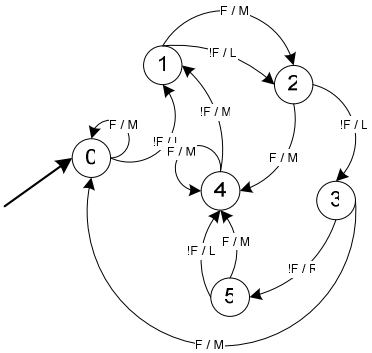
С помощью описанного алгоритма удалось найти оптимальные автоматы из пяти и шести состояний. Время вычислений на однопроцессорной ЭВМ составило около **полутора часов для пяти состояний** и около **шести суток для шести состояний**. Ускорение в несколько сотен раз по сравнению с алгоритмом перебора, рассмотренным в работе [8], объясняется тем, что предлагаемый алгоритм осуществляет перебор автоматов одновременно с моделированием их работы.

В заключение раздела отметим, что рассматриваемая задача хорошо иллюстрирует понятие комбинаторного взрыва в объеме вычислений при увеличении числа состояний даже для достаточно простых автоматов.

РЕЗУЛЬТАТЫ ПОСТРОЕНИЯ АВТОМАТОВ

В табл. 1 приведены автоматы, построенные с помощью генетического программирования и с помощью перебора. На рисунках в таблице используются такие же обозначения, как и на рис. 2.

Таблица 1. Автоматы из пяти и шести состояний, построенные генетическим программированием и перебором

Число состояний	Генетическое программирование	Перебор
5	 Муравей съедает 83 яблока. Время построения – менее десяти секунд (от 40000 до 100000 вычислений функции приспособленности).	 Муравей съедает 83 яблока. Время построения – полтора часа. Переход из состояния 2 по значению входной переменной «Перед муравьем есть яблоко» не имеет значения.
6	 Муравей съедает 85 яблок. Время построения – порядка одной минуты (от 100000 до 2000000 вычислений функции приспособленности).	 Муравей съедает 85 яблок. Время построения – шесть суток.

На рис. 3 изображен граф переходов автомата с **семью** состояниями, построенного разработанным алгоритмом генетического программирования, который позволяет муравью съесть все яблоки за 190 ходов.

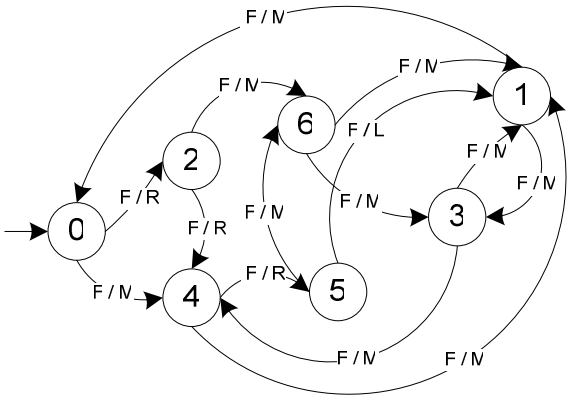


Рис. 3. Автомат, позволяющий муравью съесть все яблоки

В табл. 2 указано максимальное число яблок, которое может съесть муравей, управляемый автоматом, в зависимости от числа его состояний.

Таблица 2. Зависимость максимального числа яблок от числа состояний автомата

Число состояний	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Число яблок	42	42	58	77	83	85	89

Значение для семи состояний получено с помощью генетического программирования, для шести и меньше – с помощью перебора. Отметим, что и для числа состояний, меньшего семи, те же результаты были получены с помощью генетического программирования.

ЗАКЛЮЧЕНИЕ

В работе решен вопрос о минимальном числе состояний автомата, необходимом для решения задачи об «Умном муравье». Предлагаемый алгоритм генетического программирования позволяет строить оптимальные (в смысле числа яблок, съедаемых за 200 ходов) автоматы для заданного числа состояний. При этом построение осуществляется с помощью генетического программирования примерно в тысячу раз (для пяти состояний) и примерно в десять тысяч раз (для шести состояний) быстрее, чем при использовании предложенного алгоритма перебора. Ускорение по сравнению с алгоритмом полного перебора [8] составляет сотни тысяч раз.

ЛИТЕРАТУРА

1. *Шалыто А. А.* Технология автоматного программирования / Труды первой Всероссийской научной конференции "Методы и средства обработки информации" М.: МГУ. 2003. http://is.ifmo.ru/works/tech_aut_prog/

2. *Jefferson D., Collins R., Cooper C., Dyer M., Flowers M., Korf R., Taylor C., Wang A.* The Genesys System. 1992. www.cs.ucla.edu/~dyer/Papers/AlifeTracker/Alife91Jefferson.html

3. *Angeline P. J., Pollack J.* Evolutionary Module Acquisition // Proceedings of the Second Annual Conference on Evolutionary Programming. 1993. http://www.demo.cs.brandeis.edu/papers/ep93.pdf

4. *Chambers L.* Practical Handbook of Genetic Algorithms. Complex Coding Systems. Volume III. CRC Press, 1999.

5. *Гладков Л. А., Курейчик В. В., Курейчик В. М.* Генетические алгоритмы. М.: Физматлит, 2006.

6. *Рассел С., Норвиг П.* Искусственный интеллект: современный подход. М.: Вильямс, 2006.

7. *Koza J. R.* Genetic programming: on the programming of computers by means of natural selection. MIT Press, 1992.

8. *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. http://is.ifmo.ru/download/ant_ga_min_number_of_state.pdf

9. *De Jong K.* An analysis of the behavior of a class of genetic adaptive systems. PhD thesis. Univ. Michigan. Ann Arbor, 1975.