Статья опубликована в журнале «Информационные технологии». 2008. №1, с. 42-45, 89.

ПРИМЕНЕНИЕ ГЕНЕТИЧЕСКИХ АЛГОРИТМОВ ДЛЯ СОЗДАНИЯ УПРАВЛЯЮЩИХ АВТОМАТОВ В ЗАДАЧЕ О «ФЛИБАХ»

Е.А. Мандриков, В.А. Кулев, А.А. Шалыто, д-р техн. наук, проф.

Санкт-Петербургский государственный
университет информационных технологий механики и оптики

В работах [1, 2] рассмотрена задача о “Флибах”, суть которой состоит в том, чтобы построить флиб – конечный детерминированный автомат (в дальнейшем автомат), который будет предсказывать битовое значение некоторого параметра окружающей среды на основе ранее полученных значений.

Эта задача может быть достаточно просто решена эвристически, если число состояний автомата не ограничивать. В противном же случае, её эвристически не решить.

Поэтому в указанных работах рассматривалась автоматическая генерация автоматов с помощью генетических алгоритмов.

В работе [2] предложен генетический алгоритм, который сравнительно редко строит автоматы, позволяющие предсказывать значение указанного параметра со 100% точностью. При этом вопрос о числе состояний автоматов специально не рассматривался, но их число значительно превышало необходимое для получения 100% точности.

Цель настоящей работы – найти компромисс между числом состояний автомата и точностью предсказаний.

1. Постановка задачи

Сформулируем задачу о “Флибах” более подробно. Эту задачу можно рассматривать как моделирование простейшего живого существа, которое способно предсказывать изменения параметра среды, обладающего **периодичностью**. При этом живое существо (флиб) удобно моделировать конечным автоматом с действиями на переходах – автоматом Мили. На вход флиба подается битовое значение параметра окружающей среды. Флиб формирует значение выходной переменной – возможное значение рассматриваемого параметра в следующий момент времени.

Задача состоит в повышении точности (по сравнению с работами [1,2]) предсказания значения, которое будет иметь параметр среды в следующий момент времени.

2. Описание окружающей среды

В качестве входных воздействий для флиба (как и в работах [1,2]) используется периодическая последовательность значений битов, названная битовой маской, которая задает среду. Для обеспечения периодичности маска в программе зациклена.

3. О возможных решениях

Сформулируем условия, при которых задача может быть решена эвристически. Таким решением будет автомат, число состояний в котором равно длине битовой маски. Продемонстрируем это на примере. Пусть задана битовая маска “1111010010”, длина которой равна 10. На рис.1 приведен граф переходов с линейной структурой для автомата, который решает поставленную задачу со 100% точностью предсказаний при числе состояний 10.

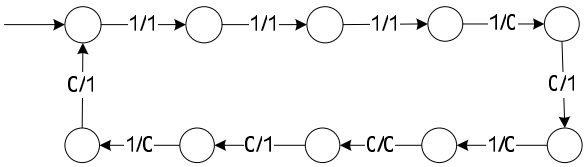

Рис.1. Граф переходов эвристически построенного автомата

В этом автомате дуги помечены символами битовой маски, а выходные воздействия – левый циклический сдвиг этой маски. Петли в каждом из состояний умалчиваются.

Этот автомат обладает 100% точностью предсказаний в том смысле, что если на него подать битовую маску несколько раз, то он её столько же раз и воспроизведет.

Этот автомат имеет сравнительно большое число состояний, так как строится с помощью универсального алгоритма, который не учитывает свойств заданной битовой маски. Число состояний может быть сокращено, возможно, за счет потери точности предсказания, если эту специфику учитывать. Как будет показано в разд. 5, предложенный в следующем разделе генетический алгоритм позволит находить компромисс между точностью предсказаний и числом состояний автомата.

4. Описание генетического алгоритма

Представление автоматов. В данной работе, так же как и в работе [1], автоматы представляются в виде битовых строк фиксированной длины. При этом каждому автомату соответствует строка, состоящая из блоков. Каждому блоку соответствует состояние автомата и переходы из этого состояния. Каждый переход характеризуется парой чисел – номером состояния, в которое автомат переходит, и значением выходного воздействия.

Общая схема. Генетический алгоритм в общем случае состоит из пяти этапов (формирование начальной популяции, тестирование популяции (приспособленность особей), скрещивание (кроссовер), мутация, формирование следующего поколения) и двух операторов (скрещивания двух особей и мутации одной).

Однако для каждой задачи с целью учета ее специфики генетический алгоритм должен быть адаптирован. Также для каждой задачи необходимо задать оценочную функцию, которая называется *fitness-функцией*. Она предназначена для оценки приспособленности особи.

Опишем, как реализуются этапы, операторы и строится *fitness-функция* в данном случае.

Формирование начальной популяции. Начальная популяция заполняется особями – автоматами, число которых равно размеру популяции.

Тестирование популяции. После формирования поколения необходимо оценить приспособленность каждого автомата. Для этого каждый из построенных автоматов устанавливается в начальное состояние, а затем на его вход многократно подаются значения параметра среды, заданные битовой маской. Состояния, в которые при этом переходит автомат, будем называть использованными (существуют также состояния, которые при этих входных воздействиях не могут быть достигнуты).

Лучшим предсказателем является тот автомат, который наиболее точно предсказывает входную последовательность. Если в указанных выше работах в качестве оценочной функции предлагалось использовать количество правильно угаданных значений, то в настоящей работе в качестве такой функции предлагается использовать соотношение:

$$\mathtt{Fitness} = \begin{cases} \text{если } \mathtt{P} = \mathtt{N}, \text{ то } \mathtt{fitness} = \mathtt{P} * \mathtt{N} + \mathtt{L}; \\ \text{если } \mathtt{P} <> \mathtt{N}, \text{ то } \mathtt{fitness} = \mathtt{P} * \mathtt{N} + \mathtt{N} - \mathtt{L}. \end{cases}$$

Здесь `N` – длина битовой маски, `P` – количество верно предсказанных значений, `L` – количество использованных состояний автомата.

Выбор данного соотношения основан на том факте, что автомат с числом использованных состояний, равным длине битовой маски, имеет простую структуру, описанную в разд. 3. Построив такой автомат, можно пытаться минимизировать количество использованных состояний, не ухудшая при этом количество верно предсказанных значений.

Скрещивание. Для каждой операции скрещивания выбирается пара особей из предыдущей популяции. Особи отбираются при помощи “рулетки” [3] с вероятностью, пропорциональной значению функции приспособленности. Эта операция повторяется число раз, которое пропорционально размеру предыдущей популяции. Коэффициент, определяющий это число, будет являться одним из параметров генетического алгоритма на рассматриваемом этапе.

Мутация. Для каждой операции мутации выбирается случайная особь из предыдущей популяции. Как и при скрещивании, особь выбирается при помощи “рулетки”, а количество мутаций зависит от размера предыдущей популяции. Соотношение количества мутаций к размеру популяции также будет являться параметром генетического алгоритма на рассматриваемом этапе.

Формирование следующего поколения. В результате скрещиваний и мутаций строится расширенная промежуточная популяция, состоящая из предыдущей популяции и результатов скрещиваний и мутаций. Из нее выбираются особи для формирования новой популяции такого же размера, как и предыдущая популяция. Выбор осуществляется следующим образом:

- в новую популяцию переносятся лучшие особи из промежуточной популяции. Их количество является параметром алгоритма на рассматриваемом этапе;
- из оставшихся особей промежуточной популяции при помощи “рулетки” выбираются особи для переноса в новую популяцию. Перенос продолжается до тех пор, пока размер новой популяции не достигнет размера предыдущей.

В итоге формируется новая популяция, которая готова к тестированию и следующей итерации генетического алгоритма.

Оператор скрещивания. Для скрещивания двух особей используется одноточечный оператор скрещивания двух битовых строк.

Оператор мутации. Оператор мутации для каждого состояния автомата с вероятностью, являющейся параметром алгоритма, случайным образом изменяет действия на переходах и номера состояний, в которые автомат переходит.

Заметим, что после выполнения оператора скрещивания или мутации, может получиться автомат с недостижимыми состояниями, но, в отличие от работы [2], вводить операцию восстановления связей между состояниями не потребовалось.

5. Эксперименты

Ниже приводятся результаты экспериментов, основанных на применении трех алгоритмов, изложенных в работах [1, 2], и предлагаемого алгоритма.

Все эксперименты проводились при 100 особях в поколении. Число воздействий среды на флиб – 100. Число поколений – 400. В таблицах с результатами экспериментов (табл. 1, 2) приводятся минимальная, средняя и максимальная точности предсказаний построенных автоматов за 30 запусков. При этом для всех трех ранее предложенных алгоритмов авторами настоящей работы были повторены оба эксперимента по программе, опубликованной в работе [2].

Эксперимент 1. Битовая маска из 19 символов: 1111010010111101001. Результаты эксперимента приведены в табл. 1.

Таблица 1. Результаты первого эксперимента

	Минимальный результат	Усредненный результат	Максимальный результат
Алгоритм №1	71	80,87	89
Алгоритм №2	84	92,10	100
Алгоритм №3	89	95,07	100
Предложенный алгоритм	95	99,47	100

Результаты запусков алгоритмов представлены также графически (рис.2).

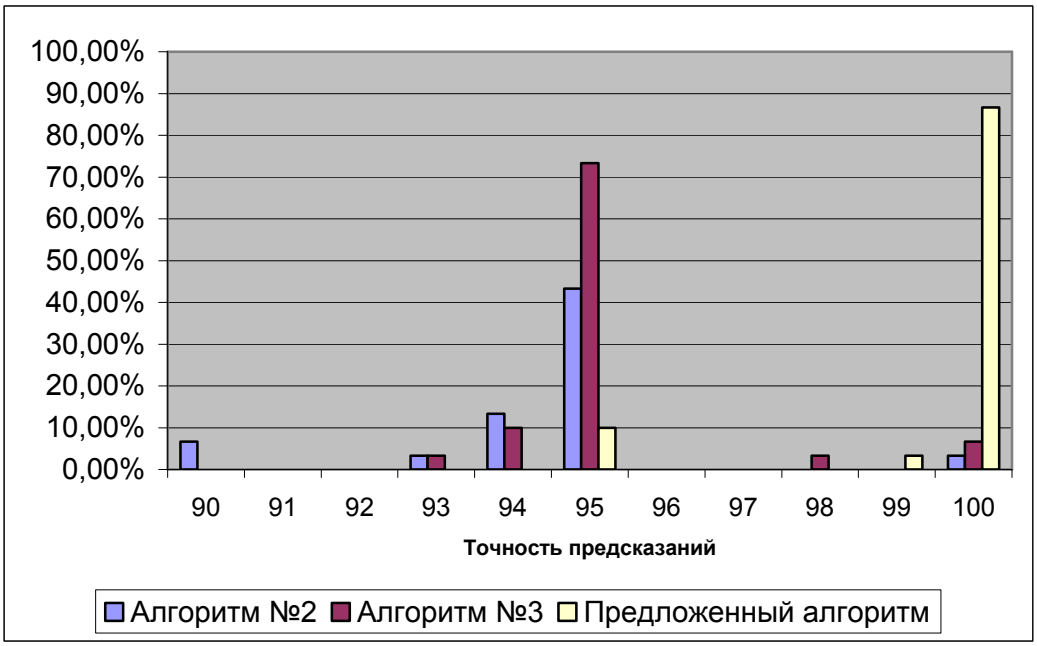

Рис.2. Частота получения предсказателей

Из рассмотрения рис. 2 следует, что предлагаемый алгоритм значительно чаще строит предсказатели, обладающие 100% точностью – в 86% случаев, против 7% у прототипа (наиболее близкого аналога).

Выше вопрос о числе состояний предсказателей не рассматривался. Переходя к этому вопросу применительно к предлагаемому алгоритму, приведем сначала графики значений предлагаемой *fitness-функции*, так как она зависит от числа используемых состояний.

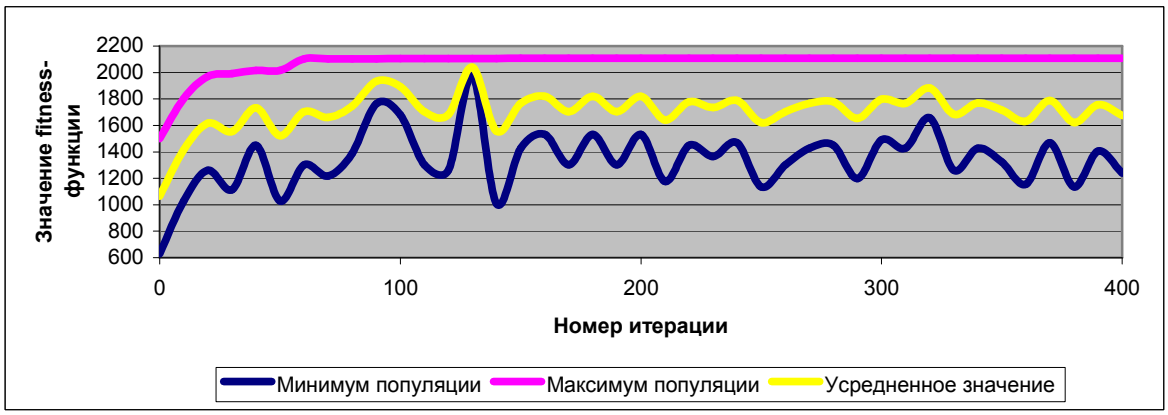

Рис. 3. Изменения *fitness-функции* в ходе первого эксперимента

Из рассмотрения графиков следует, что уже после 60 итераций алгоритм обеспечивает приближение *fitness-функции* к насыщению, что означает получение 100%-предсказателя и минимизацию числа состояний по сравнению с прототипом. Проиллюстрируем графически высказанные утверждения.

На рис. 4 показано, что при 60 итерациях был получен автомат со 100% точностью предсказаний.

Рис. 4. Зависимость точности предсказаний от номера итерации

На рис. 5 показано, что для получения 100%-предсказателя количество используемых состояний росло, а после получения такого предсказателя выбранная *fitness-функция* обеспечила уменьшение числа используемых состояний. При этом отметим что, несмотря на то, что битовая маска содержит 19 символов, в ходе эксперимента строятся 100%-предсказатели с числом состояний, превышающим эту величину (21 состояние), а в конце эксперимента – с 12 состояниями.

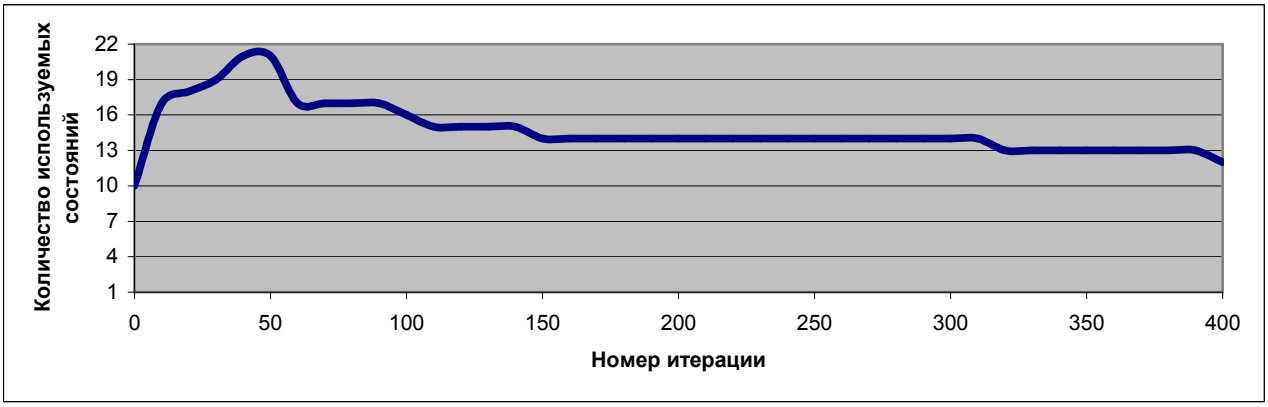

Рис. 5. Зависимость количества используемых состояний от номера итерации

На рис. 6 приведен граф переходов 100%-предсказателя с 12 состояниями, построенный в первом эксперименте. В этом графе неиспользуемые состояния и переходы в них, естественно, не отображены. При этом отметим, что в приведенном графе некоторые из переходов возможно и не используются.

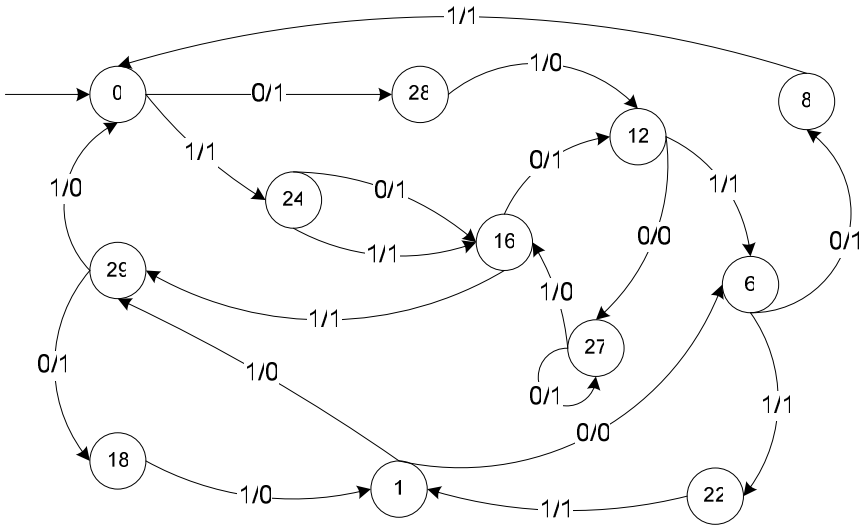

Рис. 6. Наилучший построенный автомат для первого эксперимента

Эксперимент 2. Битовая маска из 31 символа: 10101111011000111101111100110 01. Результаты эксперимента приведены в табл. 2.

Таблица 2. Результаты второго эксперимента

	Минимальный результат	Усредненный результат	Максимальный результат
Алгоритм №1	66	72,43	80
Алгоритм №2	82	89,63	96
Алгоритм №3	85	90,90	97
Предложенный алгоритм	96	97,53	100

Результаты запусков алгоритмов представлены также графически (рис.7).

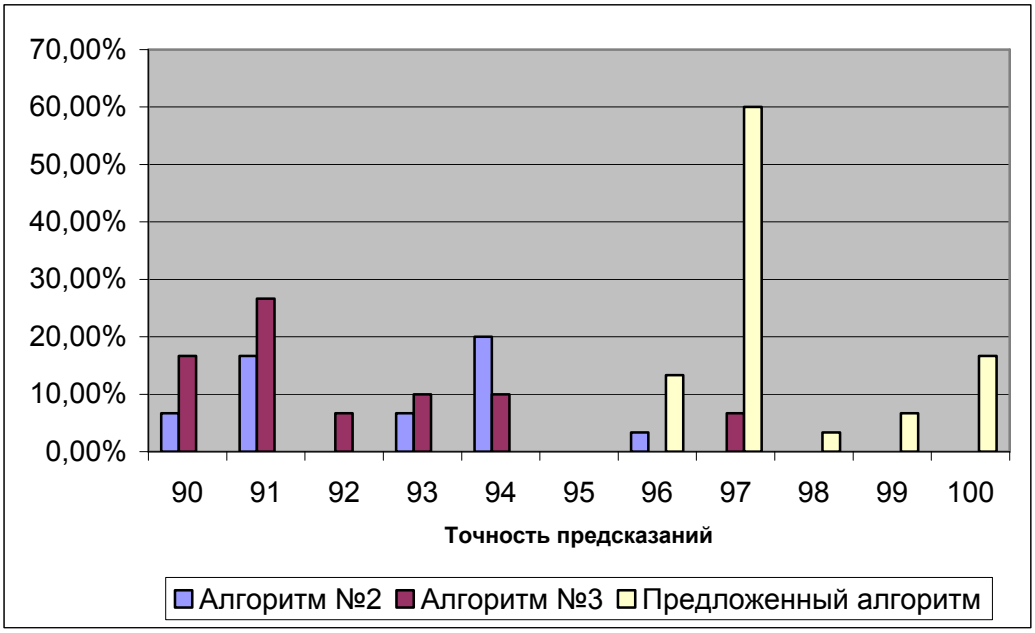

Рис. 7. Частота получения предсказателей

Из рассмотрения рис. 7 следует, что в прототипе для рассматриваемой битовой строки вообще не удавалось построить 100%-предсказатель, в то время как предлагаемый алгоритм это позволяет сделать в 17% случаев.

Переходя к вопросу о числе состояний автомата в этом эксперименте, вновь рассмотрим графики значений *fitness-функции*, представленные на рис. 8.

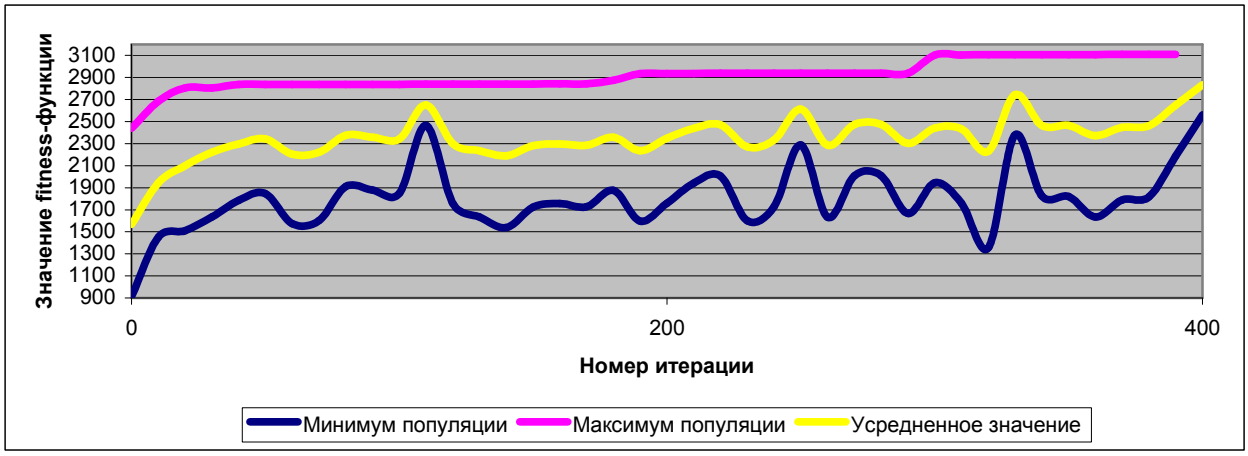

Рис. 8. Изменения *fitness-функции* в ходе второго эксперимента

На рис. 9 показано, что в районе 310 итерации был получен автомат со 100% точностью предсказаний.

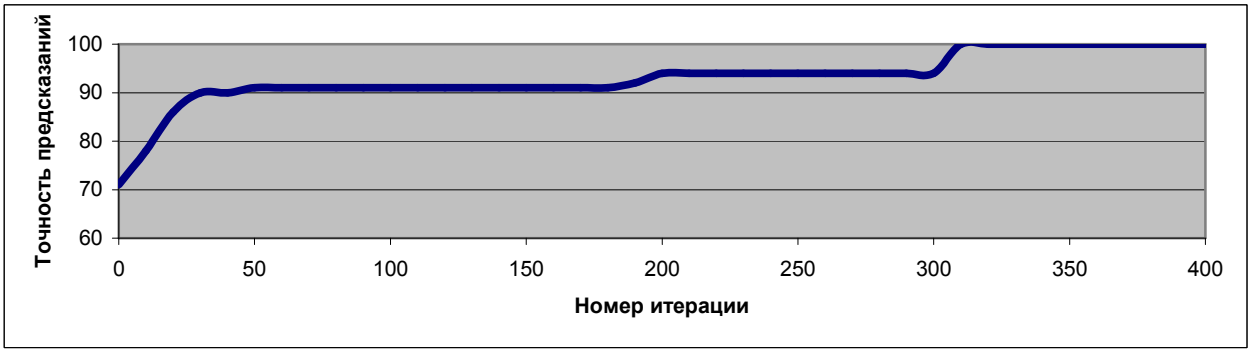

Рис. 9. Зависимость точности предсказаний от номера итерации

Рис. 10 иллюстрирует тот факт, что для получения 100%-предсказателя количество используемых состояний росло, а после получения такого предсказателя выбранная *fitness-функция* обеспечила уменьшение числа используемых состояний. При этом отметим, что в ходе данного эксперимента количество состояний автомата не превышало длины битовой маски (31), а в конце эксперимента был построен 100%-предсказатель с 20 состояниями.

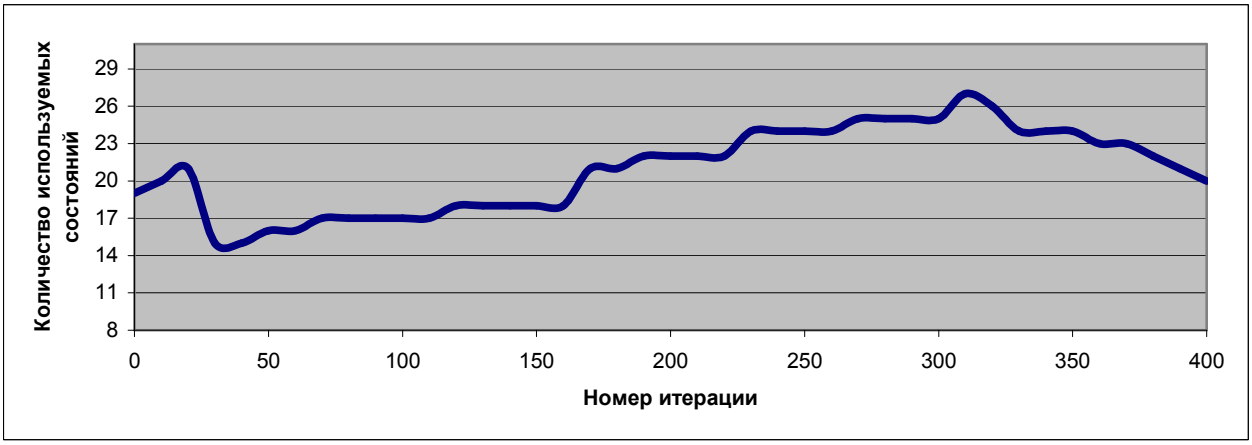

Рис. 10. Зависимость количества используемых состояний от номера итерации

На рис. 11 приведен граф переходов 100%-предсказателя с 20 состояниями, построенный во втором эксперименте. В этом графе имеет место та же ситуация для неиспользуемых состояний и переходов, что и в графе на рис. 6.

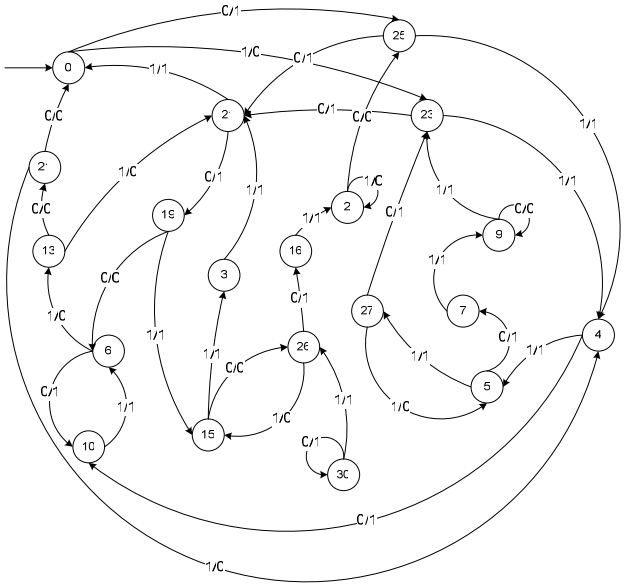

Рис. 11. Наилучший построенный автомат для второго эксперимента

Из рассмотренных примеров следует, что предложенный генетический алгоритм значительно эффективнее прототипа как по частоте построения 100%-предсказателей, так и по числу их состояний.

Таким образом, показано, что если автомат имеет число состояний меньшее, чем длина битовой маски, то предложенный генетический алгоритм позволяет учесть ее специфику.

Заключение

Вычислительные эксперименты проводились на компьютере с процессором *Core 2 Duo* с тактовой частотой 1830 МГц. Время вычислений при этом в среднем составляло 40 с.

Отметим еще раз, что эффективность предлагаемого подхода базируется на новизне *fitness-функции*, которая в отличие от прототипа учитывает не только количество верно предсказанных значений, но и число состояний автомата и длину битовой маски.

Список литературы

1. **Воронин О., Дьюдни А.** Дарвинизм в программировании // Мой компьютер. 2004. № 35. http://www.mycomp.kiev.ua/text/7458
2. **Лобанов П. Г., Шалыто А. А.** Использование генетических алгоритмов для автоматического построения конечных автоматов в задаче о “Флибах” / Материалы 1-й Российской мультиконференции по проблемам управления. Сборник докладов 4-й Всероссийской научной конференции "Управление и информационные технологии" (УИТ-2006). СПбГЭТУ "ЛЭТИ". 2006, с.144–149. http://is.ifmo.ru/works/flib/
3. **Николенко С. И.** Лекция по генетическим алгоритмам. http://logic.pdmi.ras.ru/~sergey/teaching/ml/notes-04-genetic.pdf