УДК 519.7

ГЕНЕТИЧЕСКИЕ АЛГОРИТМЫ В МАТЕМАТИЧЕСКОМ МОДЕЛИРОВАНИИ

Литвиненко В.И., к.т.н., доцент, Фефелов А.А., аспирант, Ткачук А.А. аспирант

Херсонский государственный технический университет

Генетические алгоритмы являются эффективным инструментом для решения многих сложных задач. Математический аппарат таких алгоритмов предельно прост и базируется на принципах естественной эволюции живых организмов в природе. В статье приводится подробное описание структуры генетического алгоритма и работы отдельных ее элементов на примере классической схемы. Также рассмотрен пример использования генетических алгоритмов при исследовании математических моделей, представленных в виде систем алгебраических уравнений.

Ключевые слова: генетический алгоритм, индивидуум, целевая функция, система алгебраических уравнений

Литвиненко В.І., Фефелов А.О., Ткачук А.О. ГЕНЕТИЧНІ АЛГОРИТМИ В МАТЕМАТИЧНОМУ МОДЕЛЮВАННІ / Херсонський державний технічний університет, Україна

Генетичні алгоритми є ефективним інструментом для рішення багатьох складних задач. Математичний апарат таких алгоритмів гранично простий і базується на принципах природної еволюції живих організмів у природі. У статті приводиться докладний опис структури генетичного алгоритму і роботи окремих його елементів на прикладі класичної схеми. Також розглянутий приклад використання генетичних алгоритмів при дослідженні математичних моделей, представлених у виді систем алгебраїчних рівнянь.

Ключові слова: генетичний алгоритм, індивідуум, цільова функція, система алгебраїчних рівнянь

Litvinenko V.I., Fefelov A.A., Tkachuk A.A. THE GENETIC ALGORITHMS IN MATHEMATICAL MODELING / Kherson state technical university, Ukraine

The genetic algorithms are the effective tools to find the solution of many difficult tasks. The mathematical engine of such algorithms is extremely simple and based on principles of natural evolution of individuals. The paper deals with the detailed description of genetic algorithm structure and activities of its separate elements on an example of the classical scheme. An example of using the genetic algorithms in research of mathematical models shown as systems of algebraic equations, is also considered.

Key words: genetic algorithm, individual, objective function, system of algebraic equations.

1. ВВЕДЕНИЕ

Прикладное эволюционное моделирование является одним из направлений эволюционного моделирования [1,2] пытающееся использовать принципы, на которых построена согласованная и эффективная работа биологических организмов на основе эволюции и естественного отбора. Согласно учению Дарвина, жизнь – это борьба, в которой выживают лишь сильнейшие особи и производят потомство. Естественный отбор является важнейшим фактором эволюции живых организмов в природе. Благодаря ему только сильнейшие организмы выживают и производят потомство. Если организм не выдерживает того или иного испытания на жизнеспособность, например не распознает опасности в хищнике или не находит своей пищи, то он погибает. Согласованность и эффективность работы элементов биологических организмов наводит на мысль – можно ли использовать принципы биологической эволюции использовать для решения практических задач? В настоящее время генетические алгоритмы, как один из наиболее распространенных видов эволюционных алгоритмов, уже продемонстрировали способность находить новые решения при конструировании многих сложных систем [3,4,5]. Благодаря механизму скрещивания, генетические алгоритмы позволяют исследовать значительно более широкое многообразие возможных решений задачи по сравнению с обычными компьютерными программами. Механизм естественного отбора выбраковывает худшие решения и поощряет лучшие, давая переходить из поколения в поколение и постоянно совершенствоваться, скрещиваясь, друг с другом. Благодаря этим свойствам генетические алгоритмы хорошо подходят для оптимизации многомерных, плохо определенных функций. В настоящее время генетические алгоритмы уже продемонстрировали высокую эффективность при решении разного рода задач и прежде чем перейти к рассмотрению конкретного примера, мы ответим на вопрос – как работает генетический алгоритм?

2. МЕХАНИЗМ РАБОТЫ КЛАССИЧЕСКОГО ГЕНЕТИЧЕСКОГО АЛГОРИТМА

Воспользуемся простейшим примером. Рассмотрим генетический алгоритм, работающий по схеме, изображенной на рисунке 1. Допустим, что нам необходимо найти максимум некоторой функции $y = f(x)$ (рис. 2). Для начала эволюционного процесса, прежде всего, необходимо наличие совокупности индивидуумов или популяции. Блок 2 на рисунке 1 символизирует процесс создания такой совокупности.

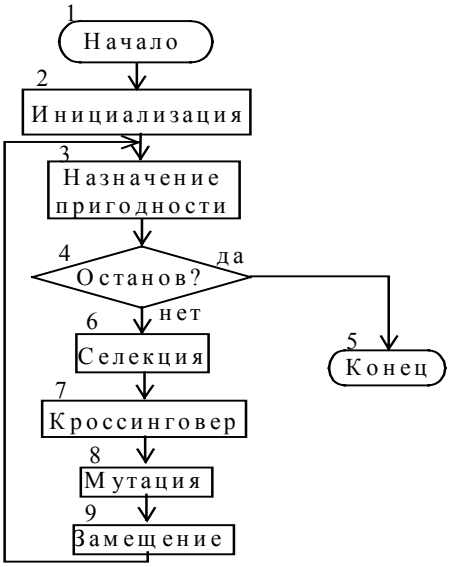

Рис. 1. Блок-схема простого генетического алгоритма

Обычно популяция инициализируется случайным образом, и для нашего примера будет состоять из ряда значений аргумента функции – x. То есть каждый индивидуум будет представлять собой одно из решений поставленной задачи. В таблице 1 показан фрагмент полученной популяции. В реальных условиях размер популяции выбирают в пределах от 50 до 500 индивидуумов в зависимости от сложности задачи. В нашем примере остановимся на популяции, состоящей из 6 индивидуумов. Числовое значение x является генетическим кодом индивидуума, который должен храниться в памяти ЭВМ в виде строки фиксированной длины. Как же в таком случае преобразовать эти числа в строки? Для этого необходимо задаться интервалом поиска и точностью искомого решения. Будем искать максимум функции y на интервале от 0 до 3.2 с точностью 0.1.

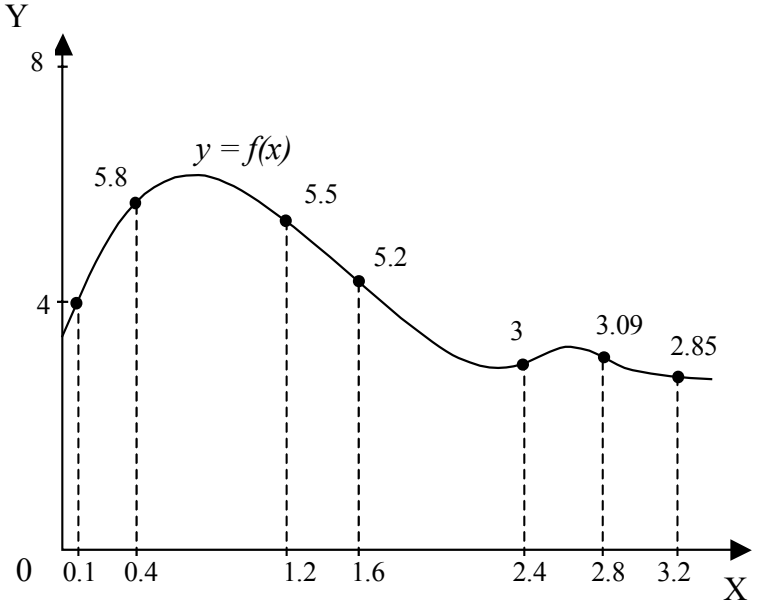

Рис. 2. Задача оптимизации. Найти максимум функции на заданном интервале.

Тогда можно разбить данный интервал на некоторое количество значений x, равное $N_x = 3.2 / 0.1 = 32$. В результате мы имеем следующее: на интервале от 0 до 3.2 существует 32 значения аргумента x и какое-то из них соответствует максимуму функции $y = f(x)$ с удовлетворяющей нас точностью $\varepsilon = 0.1$.

Теперь осталось выбрать способ представления числа и задать его в виде строки. Существует ряд способов представления чисел в генетических алгоритмах: в десятичной системе, в двоичной системе, с плавающей точкой, с использованием виртуального алфавита символов и т.д. Самым распространенным является бинарное представление, т.е. в двоичной системе. Длина строки выбирается исходя из вышеприведенных расчетов. В нашем примере для представления 32-х значений аргумента x достаточно иметь строку длины $l = log_2 32 = 5$ символов. После проведения всех необходимых преобразований наша популяция будет выглядеть, так как показано в таблице 2. В этом случае индивидуум с кодом 00000 будет соответствовать левой границе интервала ($x = 0$), а индивидуум с кодом 11111 – правой ($x = 3.1$). Любая другая комбинация нулей и единиц будет соответствовать значению аргумента внутри заданного интервала и никогда не нарушит его границы.

Таблица 1. - Фрагмент популяции

X	Значение
X_1	0.1
X_2	1.2
X_3	2.4
X_4	1.6
X_5	0.4
X_6	2.8
…	…

Итак, с кодированием мы разобрались, следуем далее по схеме (рис. 1). В третьем блоке производится оценка индивидуумов популяции. Действительно, для того, чтобы включить в работу механизм естественного отбора необходимо, прежде всего, выяснить какие особи в популяции являются сильными, а какие – слабыми. В генетических алгоритмах оценка производится на основании значений целевой функции (в нашем примере это функция $y = f(x)$).

Таблица 2. Кодирование индивидуумов

X	Значение	Индивидуум
X_1	0.1	00001
X_2	1.2	01100
X_3	2.4	11000
X_4	1.6	10000
X_5	0.4	00100
X_6	2.8	11100

После оценки, всем индивидуумам присваиваются так называемые значения пригодности (таблица 3). Так как мы решаем задачу максимизации, то «сильнее» будет тот индивидуум, у которого значение пригодности выше. Основное требование к пригодности состоит в том, что она должна выражаться скалярным числом. Если же целевая функция не в состоянии удовлетворить это требование, то на ее основе вводят функцию пригодности, которая осуществляет все необходимые преобразования.

Таблица 3. Назначение пригодности

№	Индивидуум	Пригодность
1	00001	4
2	01100	5.5
3	11000	3
4	10000	5.2
5	00100	5.8
6	11100	3.09

Следующим этапом идет проверка условия останова генетического алгоритма (блок 4, рис. 1). Никакого особого интереса этот блок не представляет, поэтому подробно останавливаться на нем мы не будем, заметим лишь, что условий останова бывает несколько (например, достигнуто нужное качество решения, пройдено определенное количество итераций и т.д.) и какое из них необходимо использовать в той или иной ситуации зависит от типа решаемой задачи и приоритетов пользователя.

Следующие три блока (6, 7 и 8) – это *генетические операторы*, непосредственно отвечающие за ход эволюционного процесса. Рассмотрим их по отдельности.

Блок 6 – оператор отбора (селекция). На основании предварительно вычисленных значений пригодности выбирает из популяции наиболее «сильные» индивидуумы, т.е. представляет собой идеализированный вариант естественного отбора в природе. Существует несколько вариантов реализации механизма отбора. Мы рассмотрим самый простой из них – турнирный отбор. Из популяции случайно выбираются несколько индивидуумов, и среди них устраивается турнир (сравниваются значения пригодности). Победитель турнира, т.е. индивидуум с наибольшим значением пригодности переходит к следующему этапу. Необходимо отметить, и это видно из сказанного выше, что алгоритм работы оператора селекции построен таким образом, чтобы дать шанс даже самым «слабым» индивидуумам быть выбранными для дальнейшего использования.

Блок 7 – «кроссинговер» - играет роль основного поискового средства генетического алгоритма. Механизм его работы прост и основан на механизме природного скрещивания. Пара индивидуумов выстраивается друг перед другом; внутри каждого из них выбирается точка пересечения, и они обмениваются частями генетического кода, расположенными до этой точки или после нее. Существуют, также, модификации кроссинговера, использующие более одной точки пересечения. Предельным случаем является равномерный кроссинговер, когда соответствующие гены могут поменяться местами с некоторой вероятностью, обычно равной 0.5. В любом случае смысл кроссинговера состоит в нахождении новых комбинаций генетического кода, обладающих более высокой пригодностью, чем родители. Допустим из нашей популяции (таблица 3) для скрещивания были выбраны индивидуумы с номерами 1 и 2. Проиллюстрируем этот процесс (рис. 3).

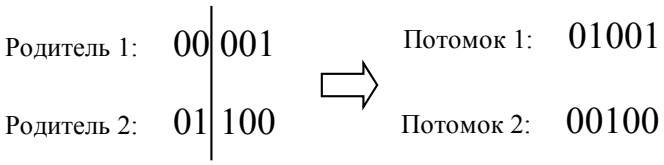

Рис. 3. Одноточечный кроссинговер

При расшифровке генетического кода потомков мы получим значения аргумента функции соответственно: 0.9 и 0.7. Интересный факт: наша популяция до этого не содержала индивидуумов с такими значениями; кроме того, если посмотреть на график (рис. 2), то можно заметить, что и первый и второй потомки оказались «сильнее» чем их родители, ведь они расположены ближе к искомой цели. Таким образом, проводя операцию скрещивания для всех индивидуумов, прошедших отбор мы получаем новое поколение, которое будет «сильнее» предыдущего. Возникает вопрос: так как процесс отбора и кроссинговера носит случайный характер, то почему же тогда популяция из поколения в поколение становится «сильнее»? Как из случайности образуется закономерность? Каким образом возникает движущая сила эволюционного процесса? Ответы на эти вопросы дает теория «строительного блока», предложенная в свое время Джоном Холландом – одним из основоположников рассматриваемого здесь направления в науке. Согласно этой теории лучшее (другими словами самое «сильное») решение строится из небольших участков генетического кода (генов или их совокупностей), обладающих повышенной пригодностью. Такие участки называются *строительными блоками*. Индивидуумы – носители строительных блоков “сильнее” других в популяции и образуют подмножества, для которых можно определить общий *шаблон*. Шаблон – это строка той же длины, что и индивидуум, но использующая расширенный алфавит символов. Он содержит дополнительный символ «*», называемый символом безразличия. Например, шаблон вида 1*0* представляет подмножество индивидуумов (1100, 1101, 1000, 1001). Для нашей популяции (таблица 3) средняя пригодность индивидуумов с номерами 3, 4 и 6 оказывается ниже, чем индивидуумов с номерами 1, 2, 5. Отсюда вывод – в данной популяции шаблон вида 0**** «сильнее» чем 1****. Значит с уверенностью можно сказать, что ген со значением 0, стоящий в первой позиции индивидуума является строительным блоком решения данной задачи. На графике (рис. 2) область, определяемая шаблоном 0****, то есть множество значений аргумента *x* от 0 до 1.6 действительно содержит искомое решение. При более детальном изучении шаблонов Холланд обратил внимание на некоторые закономерности, выраженные им в *теореме о шаблоне*. “Селективная” часть этой теоремы говорит нам о том, что в популяции из поколения в поколение будет наблюдаться экспоненциальный рост количества индивидуумов, содержащих шаблон H в том случае, если средняя пригодность этих индивидуумов выше средней пригодности популяции, и экспоненциальное уменьшение количества индивидуумов представителей H в случае если их средняя пригодность ниже средней по популяции. Формально данное утверждение можно выразить так:

$$m(H,t) = \frac{m(H,t-1)\cdot \bar{f}(H,t-1)}{\bar{f}(t-1)}, \qquad (1)$$

где $m(H,t)$ - количество индивидуумов, содержащих шаблон H в поколении t; $m(H,t-1)$ - количество индивидуумов, содержащих шаблон H в поколении t-1; $\bar{f}(H,t-1)$ - средняя пригодность индивидуумов, содержащих H в поколении t–1; $\bar{f}(t-1)$ - средняя пригодность популяции в поколении t-1.

В результате популяция будет постоянно смещаться «дрейфовать» в сторону лучшего решения, существующего на данный момент, и если при помощи кроссинговера будет обнаружено еще лучшее решение, то это приведет к изменению направления «дрейфа» и процесс должен повторяться до тех пор, пока не будет найдено оптимальное решение.
Оказывается не так уж «слеп» механизм эволюции, несмотря на случайный характер его составляющих.

Теперь вернемся к нашей схеме (рис. 1). У нас остались не рассмотренными еще два блока: восьмой и девятый.

Блок 8 – оператор мутации. Он работает следующим образом: внутри индивидуума выбирается позиция, в которой происходит изменение значения гена на любое другое, выбираемое случайно из алфавита. На рисунке 4 показано графическое представление процесса мутации.

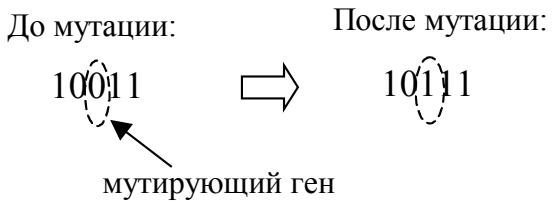

Рис. 4. Схема мутации

И опять возникает вполне резонный вопрос: если ранее рассмотренные операторы селекции и кроссинговера приводят популяцию к оптимальному решению, то чем объясняется необходимость включения оператора мутации? Ответ дает все та же теорема о шаблоне. Дело в том, что распространение одних шаблонов по популяции неизбежно приводит к вытеснению других. Следовательно, у индивидуумов появляется все больше сходных участков. Повышенная сложность решаемой задачи приводит к тому, что через несколько поколений все индивидуумы популяции станут тождественны друг другу и при этом не будет найдено лучшее решение. Очевидно, что эволюция в конвергированной популяции невозможна, так как кроссинговер становится просто бессмысленным. Мутация некоторым образом зашумляет данный процесс, как бы перебрасывая популяцию в новые области пространства решений и не давая генетическому алгоритму «застревать» в одной точке.

И, наконец, девятый блок – «замещение» - самый неинтересный блок рассматриваемой схемы. Он символизирует процесс замены старой популяции на новую, созданную операторами селекции, кроссинговера и мутации, завершая, тем самым, формирование нового поколения.

Небольшой вывод. В данном разделе мы рассмотрели схему простого генетического алгоритма, которая может использоваться для решения большого числа различных задач. Важным моментом остается адаптация структуры алгоритма под конкретную задачу. Несмотря на то, что большая часть схемы при этом не изменяется, существуют, по крайней мере, две особенности, которые необходимо учитывать каждый раз при решении новой задачи.

Первая особенность заключается в выборе способа преобразования параметров задачи в строки генетического кода (кодирование индивидуумов). Как говорилось выше, для кодирования можно использовать различные системы счисления с различными размерами алфавита символов, кроме того, многомерные задачи требуют внесения в строку информации обо всех параметрах, что в свою очередь усложняет процесс кодирования/декодирования и т.д.

Вторая особенность состоит в том, что каждая задача требует разработки своей собственной целевой функции и функции пригодности. От качества выбранных функций во многом зависит эффективность работы генетического алгоритма.

Вот, собственно все, что касается базовых принципов эволюционного подхода при решении задач. А теперь, перейдем к рассмотрению конкретного примера – примера решения систем уравнений.

3. ПРИМЕР РЕШЕНИЯ СИСТЕМ АЛГЕБРАИЧЕСКИХ УРАВНЕНИЙ

Рассмотрим систему алгебраических уравнений, содержащую *n* уравнений с *n* неизвестными. В общем виде ее можно записать так:

$$\begin{cases} f_1(x_1, x_2, ..., x_n) = 0; \\ f_2(x_1, x_2, ..., x_n) = 0; \\ .... \\ f_n(x_1, x_2, ..., x_n) = 0. \end{cases} \quad (2)$$

Генетический алгоритм – это метод, который предназначен для решения оптимизационных задач. Поэтому, прежде всего, нам необходимо преобразовать систему (2) так, чтобы она выглядела как задача оптимизации. При условии наличия ограничений на переменные $x_1, x_2, ..., x_n$ (эти ограничения, к сожалению, необходимы потому, что алгоритм работает со строками фиксированной длины; в предыдущем разделе мы более подробно останавливались на соответствующих преобразованиях) нам остается лишь позаботиться об организации целевой функции задачи. Для единственного уравнения $f(\bar{x}) = 0$ эта функция может быть записана в виде:

$$\Phi(\bar{x}) = |f(\bar{x})| \to min, \quad (3)$$

где $\bar{x} = (x_1, x_2, ..., x_n)$ - картеж переменных-аргументов функции *f*.

Для системы уравнений справедлива будет такая запись:

$$\Phi(\bar{x}) = \left(|f_1(\bar{x})|, |f_2(\bar{x})|, ..., |f_n(\bar{x})|\right), \quad (4)$$

где $|f_i(\bar{x})| \to min$, $i = \overline{1,n}$.

Значением целевой функции будет вектор $\bar{g}(g_1, g_2, ..., g_n)$, у которого $g_i = |f_i(\bar{x})|; i = \overline{1,n}$.

Итак, наша целевая функция – это вектор. Но ведь пригодность должна выражаться скалярным числом. Получается, что здесь мы имеем дело с тем случаем, когда необходимо введение функции пригодности φ. Эта функция объединяет все компоненты вектора $\bar{g}$ и возвращает скаляр, который и будет выражать пригодность индивидуумов. Для задания функции φ воспользуемся нехитрым приемом – посчитаем длину вектора $\bar{g}$:

$$\varphi = |\bar{g}| = \sqrt{\sum_{i=1}^{n} g_i^2} \to min, \quad (5)$$

В результате мы получили многомерную, многоцелевую и в общем случае нелинейную задачу оптимизации с одинаковыми приоритетами целей, которые нам удалось объединить при помощи функции пригодности, которую, в свою очередь, необходимо минимизировать.

Функция пригодности образует в пространстве параметров целевую поверхность с буграми и впадинами, подобно той, которая показана на рисунке 5.

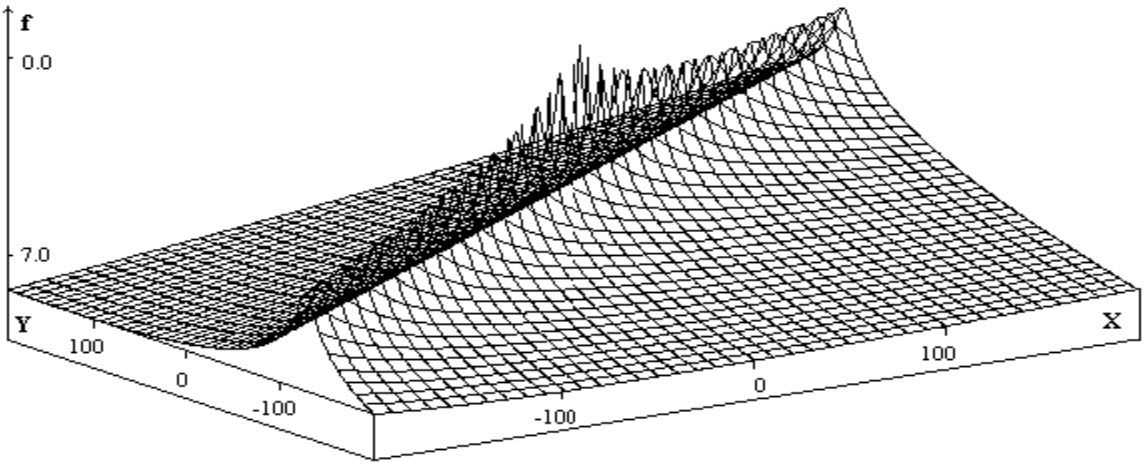

Рис. 5. Целевая поверхность

На рисунке 5 показана целевая поверхность для системы уравнений $\begin{cases} 2x+3y=7 \\ 3x+5y=12 \end{cases}$. Функция пригодности изменена на $f_r(\bar{g}) = ln(1+\varphi)$ для придания графику выразительности. На графике виден корень системы – самая высокая точка поверхности.

Оптимумы такой поверхности могут являться корнями системы алгебраических уравнений. Хотя генетический алгоритм, рассмотренный здесь в качестве примера, способен отыскивать только глобальный оптимум в заданной области пространства поиска, небольшая модификация позволит вести поиск одновременно нескольких альтернативных или локальных оптимумов. Интересно, что несовместность системы или отсутствие корней в заданной области нисколько не запутают процесс поиска, а лишь приведут к тому, что генетический алгоритм укажет на самый высокий холм или самую глубокую впадину на заданном участке целевой поверхности и это будет означать, что он нашел область наибольшего сближения графиков отдельных элементов системы.

То, что мы сейчас рассмотрели – это разработка целевой функции и функции пригодности генетического алгоритма для решения конкретной задачи. Для полной адаптации алгоритма нам, кроме этого, необходимо выбрать способ кодирования индивидуумов. Для представления переменных будем использовать бинарное кодирование, а картеж будем преобразовывать в строку путем конкатенации частей генетического кода, соответствующих отдельной переменной. Например, если воспользоваться кодированием, рассмотренным в предыдущем разделе, то картеж переменных $\bar{x}(0.1;1.2;2.4)$ преобразуется в строку вида: 000010110011000. Причем генетические операторы «воспринимают» такие индивидуумы как одно целое, не различая границ между отдельными переменными. Эти границы будут учитываться уже потом – при декодировании индивидуумов.

Отдельного разговора заслуживает выбор точности решения. Как было показано в предыдущем разделе, пространство поиска генетического алгоритма дискретно. Оно разбито на множество областей (гиперкубов) и алгоритм сходится не к точному оптимальному решению задачи, а к оптимальному гиперкубу внутри которого, находится это решение. Для того, чтобы увеличить точность решения необходимо уменьшить размер гиперкуба, т.е. уменьшить размер шага дискретизации для всех переменных. Это можно сделать двумя способами.

Первый способ. Оставив неизменными первоначальные интервалы поиска для всех переменных, увеличить длину индивидуума. Так как система уравнений содержит несколько переменных, то даже небольшое увеличение размера индивидуума приведет к значительному росту количества ячеек (гиперкубов) в пространстве. Генетический алгоритм – это метод перебора вариантов. Рост количества возможных вариантов ухудшает эффективность его работы. Поэтому разработан второй способ, предполагающий динамическое сжатие интервала поиска при сохранении длины индивидуума. Этот способ более эффективен и поэтому реализован в данной работе.

Для практической апробации описанной методики была выбрана система нелинейных алгебраических уравнений:

$$\begin{cases} x_1 - x_2^2 + 3^{x_3+x_4} - 5x_2x_5 - 2x_3 - 15 = 0; \\ 2(x_1 - x_2)^2 + 2x_2^3 + (x_3 - x_5)^3 + 15 = 0; \\ 3(x_2 - x_3)^2 - (x_2x_3 + 1)^2 + 3x_5^2 - 2 = 0; \\ 8x_1 - 7x_2^2 + (3x_2 - x_3x_4)^2 = 0; \\ x_1 \cdot 2^{x_2} + 5^{x_3} - 3x_4 + 3x_5^2 - 3 = 0. \end{cases} \quad (6)$$

Параметры генетического алгоритма приведены ниже (текст программы в приложении)
— начальный диапазон значений для всех переменных: [-50, 50];
— количество бит представления для всех переменных: $c = 5$;
— функция сжатия интервала для всех переменных представлена формулой:

$$V_i = 1 - \frac{2^{s_i}}{2^{c_i}}, \quad (7)$$

где s_i - степень сжатия интервала ($0 < s_i < c_i$), $i = \overline{1,n}$;
— степень сжатия интервала для всех переменных: $s = 3$;
— размер популяции: 50;
— тип селекции: турнир;
— размер турнира: 4;
— тип кроссинговера: однородный (uniform);
— вероятность кроссинговера: 0,9;
— вероятность мутации: 0,07.
В результате были получены следующие корни системы:
$x_1 = 0{,}706366$;
$x_2 = -1{,}956912$;
$x_3 = -0{,}925596$;
$x_4 = 1{,}373399$;
$x_5 = 1{,}495860$.

Как упоминалось ранее этот набор корней не единственный на выбранном интервале. Для того чтобы генетический алгоритм был способен к поиску одновременно нескольких наборов корней, необходима его модификация, предполагающая использование «ниш» в развитии популяции [6].

4. ЗАКЛЮЧЕНИЕ

В данной статье описан разработанный генетический алгоритм для решения систем нелинейных алгебраических уравнений. Проведен численный эксперимент для исследования эффективности генетического алгоритма. При адаптации классической схемы под конкретную задачу показаны:
- способ кодирования индивидуумов популяции;
- способ задания целевой функции.

К сожалению надо констатировать тот факт, что наш алгоритм разрабатывался по классической схеме и способен искать только глобальный оптимум на заданном интервале поиска. Это означает, что мы можем получить только одно решение системы уравнений. Однако системы нелинейных уравнений могут иметь несколько корней на выбранном интервале. В этом случае одно из направлений модификации генетического алгоритма заключается в разделении популяции на отдельные участки (ниши), в каждой из которых осуществляется поиск альтернативных вариантов решений (возможных корней системы).

ЛИТЕРАТУРА

1. Букатова И.Л. Эволюционное моделирование и его приложения. М.: Наука, 1979. 231
2. Фогель Л, Оуенс А., Уолш М. Искусственный интеллект и эволюционное моделирование. М.: Мир,1969. 230 с.
3. Holland, J.H. “Adaptation in Natural and Artificial Systems”, Ann Arbor, MI: The University of Michigan Press. 2nd edn. (1992) Boston,MA: MIT Press.
4. Холланд Х. Д. Генетические алгоритмы // В мире науки. — 1992. № 9. — С. 32—40.
5. Риоло Р. Л. Естественный отбор в мире битов // В мире науки. — 1992. № 9. — С. 160—163.
6. Miller B. L., Shaw M. J. Genetic algorithms with dynamic niche sharing for multimodal function optimization.