

Е.П.ИВАНЧЕНКО, студентка, *elena_iva@list.ru*
Я.Н.ВЫХОДЦЕВ, студент, *999fff@gmail.com*
Санкт-Петербургский государственный горный университет

E.P.IVANCHENKO, student, *elena_iva@list.ru*
I.N.VYKHODTCEV, student, *999fff@gmail.com*
Saint Petersburg State Mining University

ТЕСТИРОВАНИЕ ПРОСТОГО ГЕНЕТИЧЕСКОГО АЛГОРИТМА

Проведено тестирование классического генетического алгоритма на основе целевых функций с заданным условием поиска. Написана программа, реализующая данный алгоритм, с большим набором параметров.

Ключевые слова: генетический алгоритм, оптимизация, особь, кроссинговер, мутация, отбор, мультимодальность, многомерность.

TESTING OF SIMPLE GENETIC ALGORITHM

Authors have carried out a number of tests of classic genetic algorithm on some target functions with preassigned search terms. For this reason the program realizing this algorithm was developed. It has a wide diversity of parameters.

Key words: genetic algorithm, optimization, individual, crossing, mutation, selection, multimodal, multidimensional.

Генетические алгоритмы относятся к глобальным эмпирическим методам оптимизации. Они обязаны своим появлением Чарльзу Дарвину и его эволюционной теории, опубликованной в 1859 г. в работе «Происхождение видов», в которой были провозглашены основные принципы эволюционной теории:

- 1) наследственность;
- 2) изменчивость;
- 3) естественный отбор.

Основателем современной теории генетических алгоритмов считается Джон Холланд (John Holland), сумевший применить теорию Дарвина к искусственным системам. Его работа «Адаптация в естественных и искусственных системах» («Adaptation in Natural and Artificial Systems»), опубликованная в 1975 г., стала классикой в этой области. В ней Холланд впервые ввел термин генетический алгоритм. Сейчас описанный там алгоритм называют классическим или

каноническим. Само же понятие генетический алгоритм стало очень широким и зачастую к нему относят алгоритмы, сильно отличающиеся от классического [1].

Исследование генетического алгоритма базируется на программной реализации базовых принципов алгоритма. Специально для этого авторами была написана программа на языке программирования C# в среде Microsoft Visual Studio 2010. Ядром этой программы является класс, который содержит методы, выполняющие базовые операции: кодирование случайных особей, кроссинговер, мутация, отбор. Особи представлены в двоичном коде, при этом каждая особь содержится в 64-битном контейнере.

Интерфейс программы для многомерной системы представлен на рис.1, для одномерной – на рис.2. Программа обладает следующими возможностями: имитировать эволюцию популяции пошагово с помощью кнопок создания, кроссинговера, мутации,

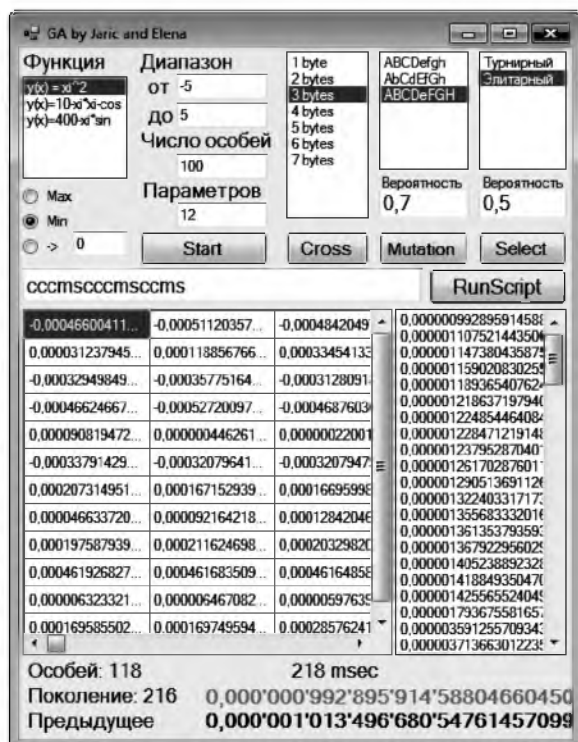


Рис.1. Главное окно «Genetic Algorithm by Jaric and Elena. Multidimensional»

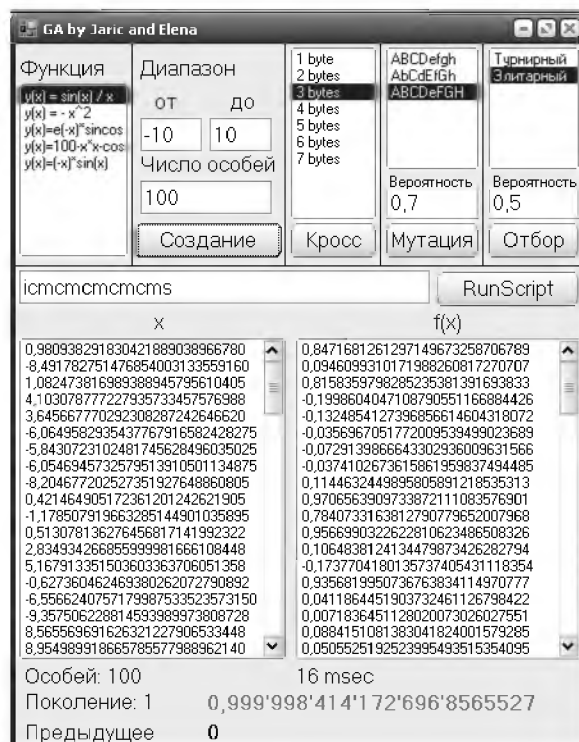


Рис.2. Главное окно «Genetic Algorithm by Jaric and Elena. One-dimensional»

отбора; также задавать параметры этих операций, кроме того, можно следить за размером существующей популяции, временем, затраченным на последнюю операцию, и лучшим найденным значением. Все перечисленное применяется для нескольких тестовых функций:

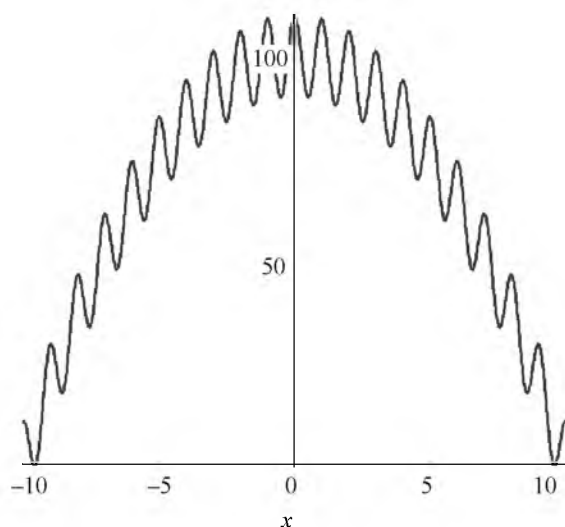


Рис.3. Функция Растригина

- функции 1-й переменной:

$$-x^2;$$

$$\frac{\sin(x)}{x};$$

$100 - x^2 + 10\cos(2\pi x)$ (функция Растригина) (рис.3);

$$-x \sin(\sqrt{|x|});$$

- функции нескольких переменных:

$$\sum_{i=1}^n x_i^2 \text{ (см. рис.1);}$$

$\sum_{i=1}^n (10 + x^2 - 10\cos(2\pi x))$ (функция Растригина);

$$1 + \sum_{i=1}^n \frac{x^2}{4000} - \prod_{i=1}^n \cos\left(\frac{x}{\sqrt{i}}\right) \text{ (функция Гриванка).}$$

Чтобы смоделировать эволюционный процесс, описанный Дарвином, сгенерируем программным способом начальную популяцию – несколько особей со случайным набором хромосом. Особью в нашем случае является число, представленное в двоичной

форме. Каждый бит является геном особи (рис.4).

Генетический алгоритм имитирует эволюцию этой популяции как циклический процесс скрещивания индивидумов и смены поколений (рис.5) [3].

Сначала происходит отбор лучших особей для скрещивания. При этом для каждой особи высчитывается функция приспособленности. Чем больше (или меньше) ее значение, тем больше у особи шансов выжить и дать потомство. Самые слабые особи погибают, сильные – выживают.

Передача генетической информации происходит при кроссинговере. Кроссинговер – обмен гомологичными участками хромосом. Он бывает одноточечный и многоточечный. Это означает, что на обеих хромосомах в одних и тех же местах появляются точки разрыва и некоторые участки хромосом, лежащие между одними и теми же точками, меняются местами (рис.6).

Таким образом, дочерняя особь получает гены обоих родителей.

В нашей программе кроссинговер реализован методом рулетки. При этом вероятность выбора особи p_i , зависит от ее функции приспособленности f_i и вычисляется по формуле

$$p_i = \frac{f_i}{\sum_k f_k}.$$

Рулеточный круг делится на сектора, причем площадь i -го сектора пропорциональна значению p_i . После этого n раз «вращается» рулетка, где n – размер популяции, и по сектору, на котором останавливается рулетка, определяется особь, выбранная для скрещивания [2]:

$$p_1 > p_2 > p_3 > \dots > p_n;$$

$$p_1 + p_2 + p_3 + \dots + p_n = 1.$$

В природе иногда происходят мутации. Они возникают вследствие случайного изменения одной или нескольких позиций в хромосоме. Если мутация улучшила приспособленность организма, она передается потомкам. Иначе особь умирает, не пройдя

естественный отбор. Таким образом, мутации разбавляют генофонд популяции «свежими» генами и особями, что помогает ей выжить. Если же все особи в популяции становятся слишком похожими – она вымирает. В оптимизации это означает, что решение сошлось к одному значению.

В нашей программе используется мутация трех видов (рис.7).

Первый вид мутации делит особь пополам, затем меняет половинки местами. Второй вид – разбивает особь на пары значений, после чего они последовательно меняются местами. Третий вид – наиболее используемый – «природная мутация», заключается в произвольном выборе гена и изменении его на случайное значение. Происходит непрерывная передача генов от родителей к потомкам, распространение лучших генов и вытеснение худших. Наследственность является одним из важнейших элементов генетических алгоритмов.

Такой процесс эволюции может продолжаться до бесконечности. Критерием останова может служить заданное время работы, количество поколений или сходжение популяции. В такой ситуации кроссинговер практически никак не изменяет популяции. Вышедшие из этой области за счет мутации особи склонны вымирать, так как чаще имеют меньшую приспособленность, особенно если данный экстремум является глобальным максимумом. Таким образом, сходжение популяции обычно означает, что найдено лучшее или близкое к нему решение.

Проведено изучение эффективности генетических алгоритмов, примененных для поиска экстремумов тестовых функций, приведенных выше.

Авторы исследовали, как зависит эффективность и время работы алгоритма от начального количества особей. Для получения более точных результатов было проведено несколько серий опытов. На рис.8 изображено распределение полученных решений (целое значение функции равно единице) и график зависимости времени работы программы от количества особей. Данные значения аппроксимируются линейно: с ростом количества особей эффективность и время

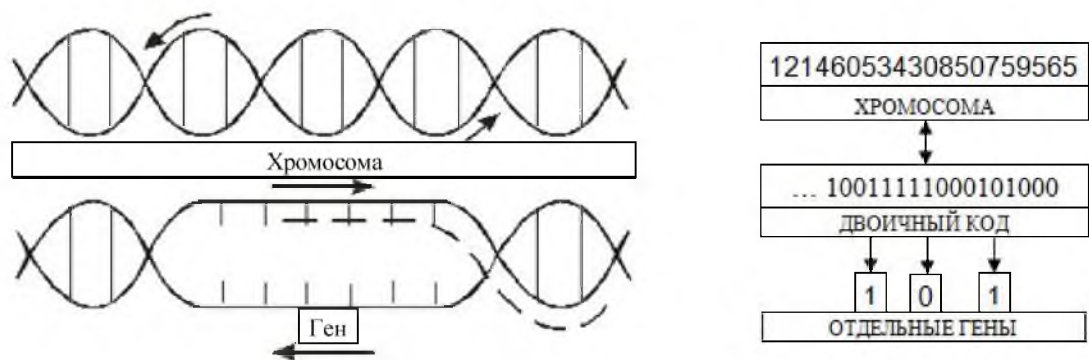


Рис.4. Двоичное представление хромосом

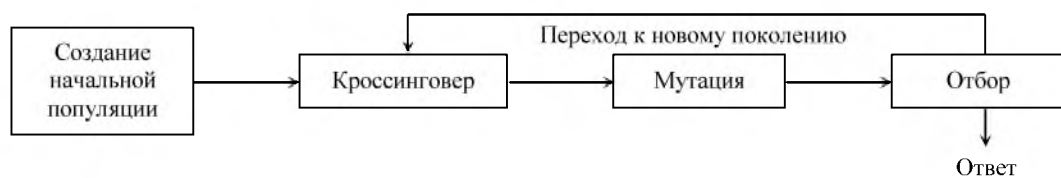


Рис.5. Эволюция популяции

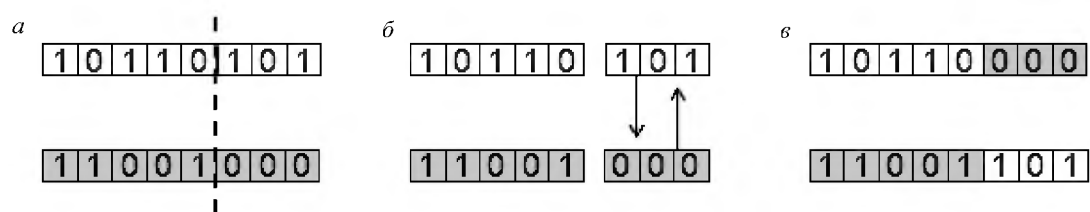


Рис.6. Кроссингвер: а – 1-я стадия; б – 2-я стадия; в – 3-я стадия

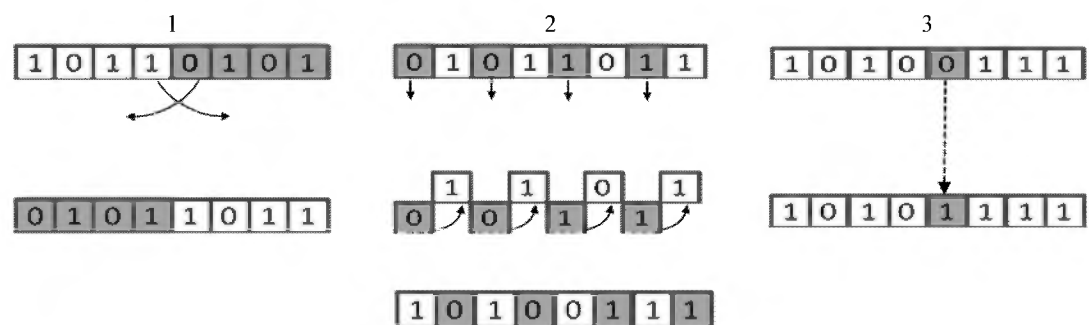


Рис.7. Виды мутации

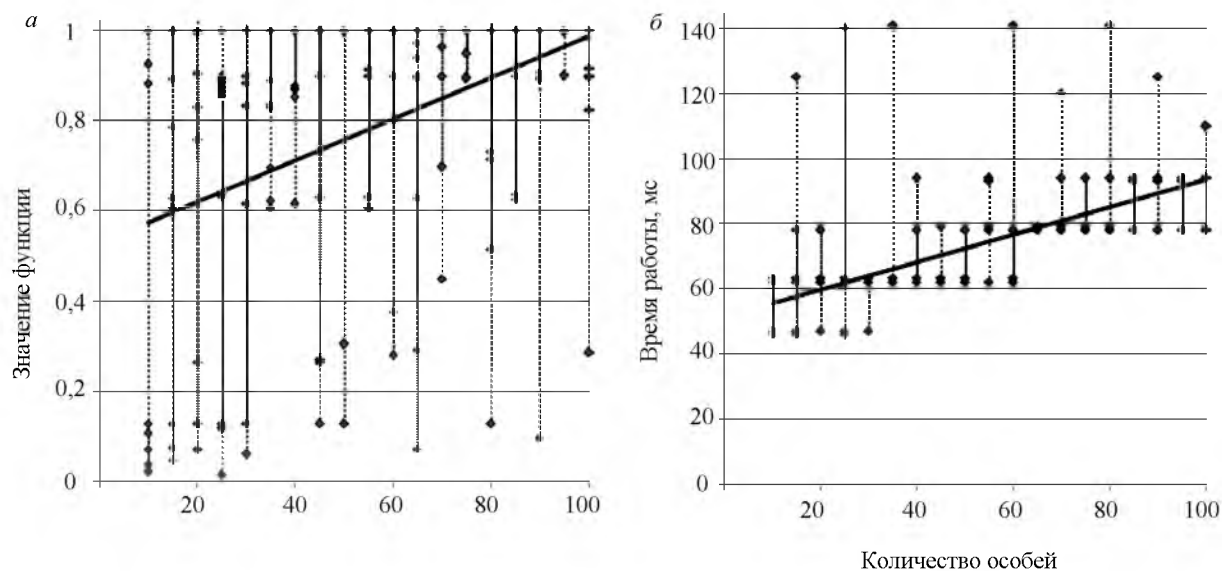


Рис. 8. Зависимость значения функции (а) и времени работы программы (б) от начального количества особей

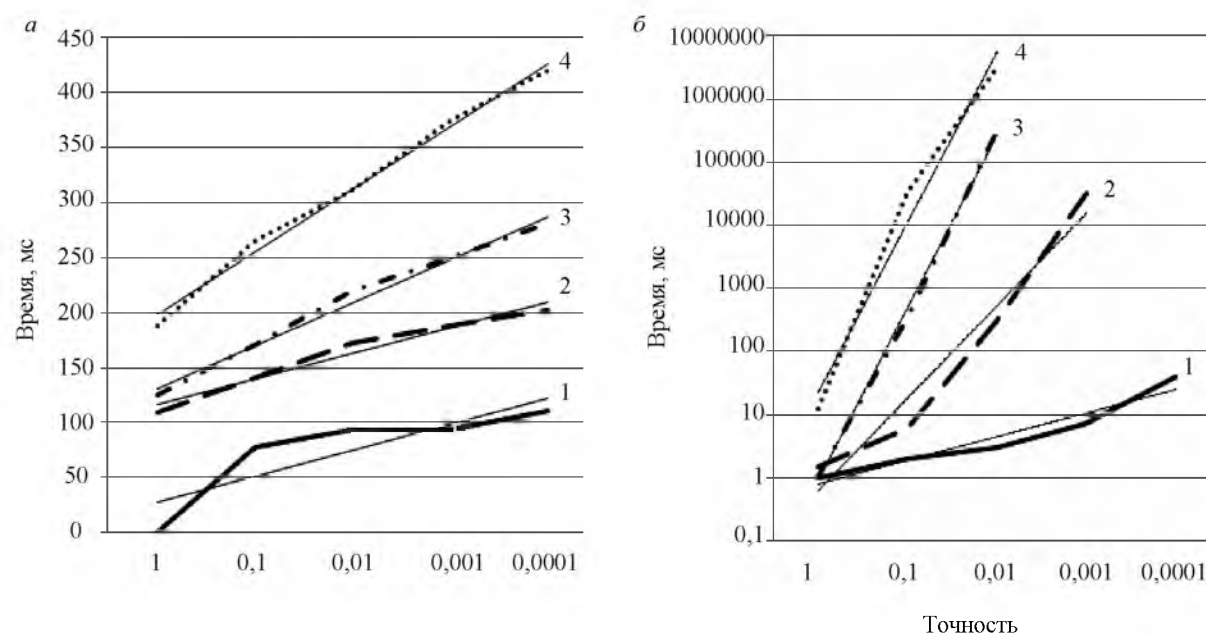


Рис. 9. Зависимость времени работы генетического алгоритма (а) и переборного метода (б) от заданной точности поиска
1-4 – количество параметров

работы программы растут прямо пропорционально.

Помимо этого, в ходе экспериментов было выявлено, что в популяциях с малым размером гены распространяются слишком быстро. Все особи становятся похожими (популяция сошлась) еще до того, как най-

дено решение задачи. Также, начиная с некоторого момента, увеличение размера популяции не влечет за собой существенного увеличения точности. Это значит, что начальное количество особей необходимо подбирать исходя из каждой конкретной задачи и требований к результату.

Кроме того, авторы сравнили быстродействие генетического алгоритма и переборного метода. Для этих целей была написана программа, реализующая метод полного перебора с заданной точностью в данном диапазоне. Время работы генетического алгоритма (рис.9, *а*) прямо пропорционально требуемой точности. Время работы переборного метода при этом растет экспоненциально (рис.9, *б*). Очевидно, что со сложными задачами генетические алгоритмы справляются во много раз быстрее без потери точности. Зная зависимость времени работы от точности на небольшом интервале, можно определить примерное время работы алгоритма для достижения необходимой точности.

Преимущества генетических алгоритмов также рассматриваются в [3]. Выводы этой работы в целом совпадают с результатами проведенных исследований.

Следует заметить, что каждая полученная особь является одним из решений поставленной задачи. Более приспособленные особи – это более подходящие ответы. Этим генетические алгоритмы отличаются от большинства других алгоритмов оптимизации, которые оперируют лишь с одним решением, улучшая его.

Исследовав влияние эмпирических параметров на эффективность генетического алгоритма, можно сказать, что генетический алгоритм представляет собой универсальную методику оптимизации, которая за разумное время находит очень близкое к лучшему решение, однако требует тщательной настройки и тестирования.

Приложение рассмотренной методики в специальной области имеет два основных аспекта: с одной стороны, задача оптимизации возникает в ходе идентификации объекта, в ходе подбора оптимальных параметров

модели. В качестве функции приспособленности выступает одна или несколько характеристик модели. Необходимо найти такие значения этих характеристик, при которых значение этой функции наиболее близко к глобальному экстремуму. В этой связи генетический алгоритм можно рассматривать и как способ построения оптимальной зависимости между входными и выходными сигналами и способ определения параметров уже построенной зависимости. Его эффективность по сравнению с методом наименьших квадратов требует исследования, но ясно, что она сильно зависит от параметров генетического алгоритма.

С другой стороны, часто ставятся оптимизационные задачи на графах, например, проектирование технологических линий, так как обычно технологическая схема представляется в виде ориентированного графа. На сегодняшний день такие задачи, как поиск кратчайшего пути в графе или задачи о размещении, чаще всего решаются методом перебора или методом ветвей и границ. Сравнение эффективности генетического алгоритма с методом перебора уже проводилось выше и показало значительное преимущество первого в решении сложных задач.

ЛИТЕРАТУРА

1. Гладков Л.А. Генетические алгоритмы / Л.А.Гладков, В.В.Курейчик, В.М.Курейчик. М., 2010.
2. QAI Main Page: Генетические алгоритмы и не только. Qai, 2003-2010. <http://qai.narod.ru/>
3. NeuroProject: AI & data analysis. «НейроПроект», 1998-2011. <http://neuroproject.ru/>

REFERENCES

1. Gladkov L.A., Kurejchik V.V., Kurejchik V.M. Genetic algorithms. Moscow, 2010.
2. QAI Main Page. Qai, 2003-2010. <http://qai.narod.ru/>
3. NeuroProject: AI & data analysis. «Neuroproekt», 1998-2011. <http://neuroproject.ru/>

Научный руководитель канд. техн. наук, доцент *А.Г.Певнева*