Интернет-журнал «Науковедение» ISSN 2223-5167 http://naukovedenie.ru/

Том 8, №6 (2016) http://naukovedenie.ru/vol8-6.php

URL статьи: http://naukovedenie.ru/PDF/32TVN616.pdf

Статья опубликована 20.12.2016

Ссылка для цитирования этой статьи:
Кошев А.Н., Салмин В.В., Генералова А.А., Бычков Д.С. Разработка генетического алгоритма с адаптивными мутациями для определения глобального экстремума функции n-переменных // Интернет-журнал «НАУКОВЕДЕНИЕ» Том 8, №6 (2016) http://naukovedenie.ru/PDF/32TVN616.pdf (доступ свободный). Загл. с экрана. Яз. рус., англ.

УДК 681.3

Кошев Александр Николаевич
ФГБОУ ВО «Пензенский государственный университет архитектуры и строительства», Россия, Пенза
Профессор кафедры «Информационно-вычислительные системы»
Доктор химических наук
E-mail: koshev@pguas.ru

Салмин Владимир Васильевич
ФГБОУ ВО «Пензенский государственный университет», Россия, Пенза
Заведующий кафедрой «Транспортные машины»
Доктор технических наук, профессор
E-mail: salmin-penza@yandex.ru

Генералова Александра Александровна[1]
ФГБОУ ВО «Пензенский государственный университет архитектуры и строительства», Россия, Пенза
Магистрант кафедры «Информационно-вычислительные системы»
ФГБОУ ВО «Пензенский государственный университет», Россия, Пенза
Доцент кафедры «Транспортные машины»
Кандидат технических наук
E-mail: generalova_aa@mail.ru

Бычков Дмитрий Сергеевич
ФГБОУ ВО «Пензенский государственный университет», Россия, Пенза
Студент 5 курса факультета «Машиностроения и транспорта»
E-mail: deciptikon@mail.ru

Разработка генетического алгоритма с адаптивными мутациями для определения глобального экстремума функции n-переменных

Аннотация. Генетические алгоритмы - это достаточно новый и довольно мощный механизм поиска экстремума многоэкстремальных функций. Много практических задач сводятся к нахождению экстремума, следовательно, данные задачи можно свести и задачам для решения средствами генетических алгоритмов. Генетические алгоритмы оперируют такими объектами, как “ген”, “хромосома”, “популяция”. Принцип работы основан на идеях эволюции живой природы.

[1] 440031, г. Пенза, улица Кижеватова, дом 30, квартира 198

Статья посвящена работе генетического алгоритма. Рассматривается вопрос генерации начальной популяции. Выделяются существующие методы формирования начальной популяции и предлагается новый метод - инициализация популяции на основе заданного закона распределения.

Для проведения эксперимента была написана программа, реализующая генетический алгоритм. В качестве тестовых функций взяты одно экстремальная и многоэкстремальная функции. Результаты показывают, генерация начальной популяции на основе закона распределения увеличивают точность решения, уменьшает время работы алгоритма.

В предлагаемом алгоритме предложено условие досрочного прерывания ГА, которое производится по условию высокой плотности найденных точек в окрестностях друг друга.

Ключевые слова: экстремум; генетический алгоритм; адаптивная мутация; сходимость функции; фитнесс-функция

Проблемы принятия оптимальных решений возникают в различных областях науки и техники, оказывая решающее влияние на развитие систем автоматизации проектирования, управления и научных исследований [1, 2]. Общим для этих проблем является то, что их математическая модель может быть сформулирована как задача оптимизации, связанная с поиском таких значений управляемых переменных, которые обеспечивают экстремальное значение (максимум или минимум) одной из наиболее важных технико-экономических характеристик объекта (процесса, системы, устройства и т.д.) при условии, что другие характеристики удовлетворяют заданной совокупности технических требований. При этом основные трудности численного решения сформулированной экстремальной задачи связаны с ее размерностью и видом оптимизируемой функции, которая в общем случае может быть нелинейной, разрывной, недифференцируемой и многоэкстремальной.

ГА - это методы построения алгоритмов поиска глобального экстремума функции, принцип которых основан на теории эволюции Ч. Дарвина [3, 4]. Хромосома в терминологии ГА - это некоторое потенциальное решение задачи. Ген - это элемент хромосомы, представляющий собой некоторую переменную исследуемой функции. Локус - местоположение гена в хромосоме. Популяция - это конечное множество хромосом. Понятия целевая функция, функция приспособленности, фитнесс-функция являются равнозначными.

Первый ГА был описан Д. Гольдбергом на основе работ Д. Холланда. Предварительно простой ГА случайно генерирует популяцию последовательностей - хромосом (альтернативных упорядоченных и неупорядоченных решений) [5]. Затем производится копирование последовательностей хромосом и перестановка их частей. Далее простой ГА реализует множество простых операций над начальной популяцией и генерирует новые решения. Простой ГА состоит из трёх операторов: селекция, кроссинговер, мутация [4, 5].

Общую последовательность вычислений можно характеризовать следующим образом [6].

1. Формируется начальная популяция - исходное множество решений.
2. Определяется приспособленность особей популяции к условиям существования. В технических приложениях - это величина критерия качества для конкретного решения.
3. Осуществляется выбор родительских особей (все особи-варианты представляются хромосомой - цепочной записью значений свойств, определяющих описание технического решения).

4. Осуществляется скрещивание (кроссинговер) родительских особей - обмен частями описаний между вариантами технических решений.
5. В варианты-потомки вносятся деструктивные изменения-мутации как следствие воздействия окружения. На практике данный прием обеспечивает появление большего разнообразия в комбинациях свойств вариантов решения.
6. Оценивается приспособленность полученных таким образом вариантов-потомков. Лучшие варианты заменяют собой менее приспособленные - с «худшим» критерия качества.
7. Фиксируется факт окончания «прожитой» эпохи. Осуществляется переход в новую эпоху - к началу новой итерации в расчетах вариантов технических решений (переход к п.2 алгоритма).

В предложенном алгоритме применяются следующие переменные:

v - количество итераций;

N - размер контрольной популяции (а также начальной), количество особей после применения фитнес-функции;

b - шаг отбора популяции фитнес-функцией, *b*=1 - отбор на каждом шаге;

H - количество особей, с которыми скрещивается *i*-тая особь, также количество потомков от одной особи;

Xi - количество переменных, с которыми работает ГА (количество хромосом) (при *Xi*=2 задача сводится к нахождению оптимума на поверхности);

OPT - указание на поиск максимума (при *OPT*=«Max») или минимума (при *OPT*=«Min»);

Ximin - массив минимальных значений для каждого искомого параметра (переменной);

Ximax - массив максимальных значений для каждого искомого параметра (переменной);

Rv - параметр агрессивности приближения, чем он больше, тем выше шанс выпадения неверных результатов, чем он меньше, тем дольше сходится алгоритм;

Dad - параметр указывающий на включение родителей в отобранную популяцию (Dad=«Life», Dad=«Live») или исключение их из отобранной популяции (Dad=«Dead», ...);

$R_otb_{i,x}$ - величина x-вого гена i-той особи в массиве всех геномов отобранной популяции R_otb;

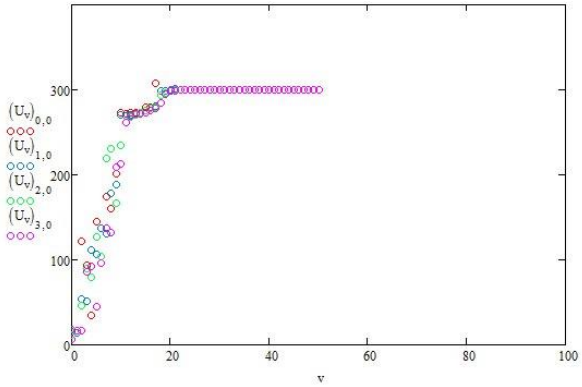
$R_otb_{j,x}$ - величина x-вого гена j-той особи в массиве всех геномов отобранной популяции R_otb.

Сформированная начальная популяция (в данном алгоритме по закону нормального распределения, но возможно применение и иных законов распределения) эволюционирует от поколения к поколению с применением оператора мутаций [7], показанном на рисунке 1.

$$mut \leftarrow norm\left[1, 0, \left|\frac{R_otb_{i,x} - R_otb_{j,x} + 1}{10^{\frac{-2i+(N-1-H)}{N-1-H}}}\right|\right]$$

Рисунок 1. *Модель адаптивных мутаций (разработано авторами)*

Мутация особей напрямую зависит от положения (приспособленности) особи. Чем хуже приспособлена особь, тем она дальше от оптимума, следовательно, ей сообщаются большие мутации, чтобы она могла удалиться от текущего неоптимального положения [7, 8]. *Dad* входит в цикл отбора и влияет на размер интервала его перебора (при Dad=«Live», перебор соседей начинается с i и заканчивается $i+H$, при Dad=«Dead», перебор соседей начинается с $i+1$ и заканчивается $i+H$). При Dad=«Live» в популяции появляются особи, не скрещивающиеся с другими (так как особь скрещивается с собой, набор генов в ней не меняется, следовательно, она остается неизменной, но, благодаря адаптивным мутациям, она мутирует тем больше, чем ниже она в списке), но также подверженные мутациям.



Gen(100,10,1,3,0,0) =

	0	1
0	300.02	299.984
1	299.819	300.042
2	299.833	300.089
3	299.996	300.196
4	300.195	300.154
5	299.885	300.423
6	300.249	299.623
7	300.052	299.522
8	300.821	300.2
9	299.505	300.745

Рисунок 2. *Показатели наиболее приспособленных особей на каждой итерации работы ГА, где U_v - множество генов всех особей отобранной популяции на v-той итерации (разработано авторами)*

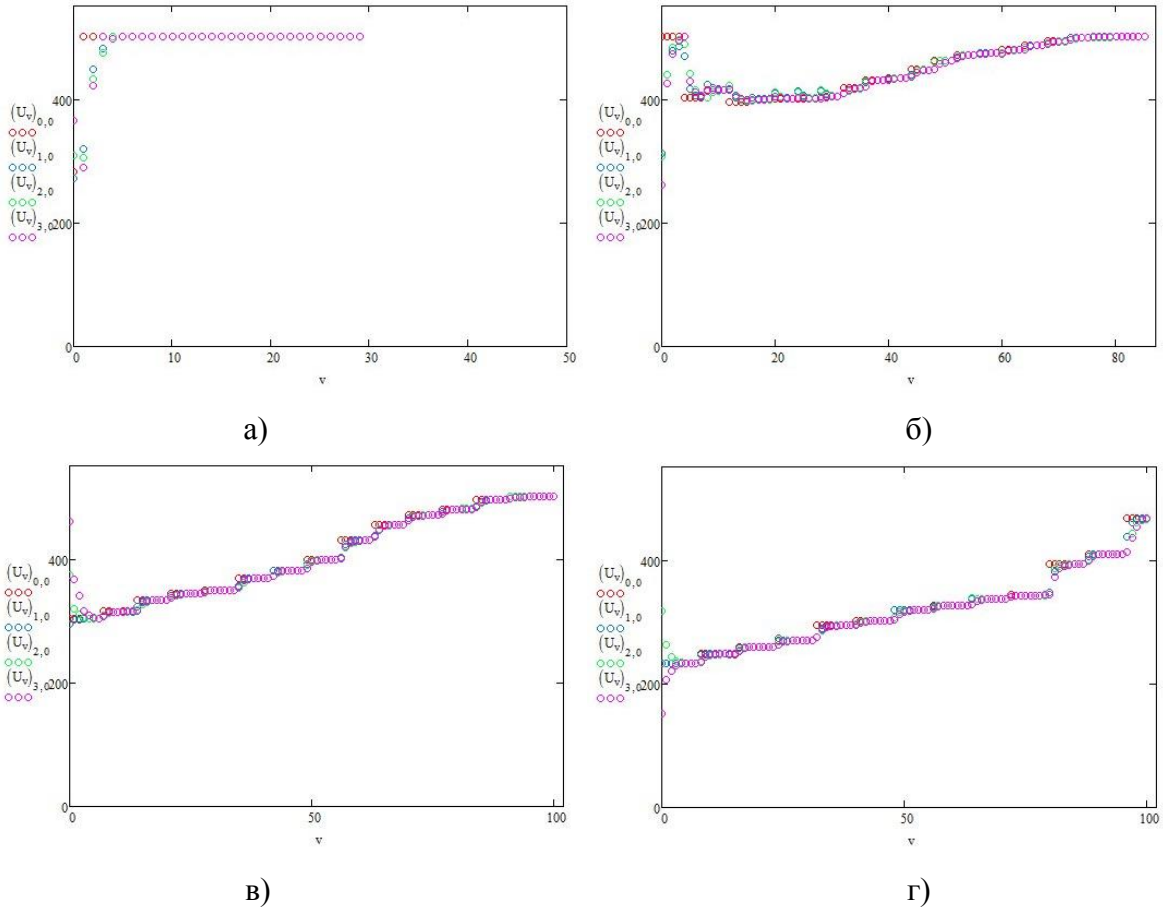


Рисунок 3. *Изменение количества итераций и сходимости поиска, где U_v - множество генов всех особей отобранной популяции на v-той итерации (разработано авторами)*

На рисунке 3 показано изменение количества итераций и сходимость поиска в зависимости от изменения шага отбора популяции b. На рисунке (3.а) оптимум был определён за 5 итераций, при шаге *b*=1; на рисунке (3.б) за 75 итераций при шаге *b*=4; на рисунке (3.в) за 93 итерации при шаге *b*=7; на рисунке (3.г) за 100 итераций при шаге *b*=8. На основании этих графиков можно сделать вывод: большой шаг отбора дольше сходится к оптимуму, но имеет большой разброс, что уменьшает вероятность попадания в локальный минимум, тогда как малый шаг быстро сходится к оптимальной точке, но имеет возможность попасть в локальный минимум.

Исследование эффективности работы ГА проводилось на тестовой функции (1):

$$\mathrm{Funkcia}(X) := \frac{40}{\frac{(X_0-3)^2}{16}+\frac{(X_1-2)^2}{16}+1} \tag{1}$$

$$\mathrm{Fitness}\ (i, R_mut) := \left| \begin{array}{l} X \leftarrow \left(R_mut^T\right)^{\langle i \rangle} \\ FUN \leftarrow 10^{10} \ \ \text{on error } FUN \leftarrow \mathrm{Funkcia}(X) \\ Z \leftarrow \frac{1}{FUN} \\ Z \end{array} \right.$$

Рисунок 4. *Вычисление фитнесс-функции (разработано авторами)*

На рисунке 4 показан модуль вычисления фитнесс-функции для *i*-той особи, обладающей набором генов подвергнутых мутациям (в случае ошибки выдает очень малое значение фитнесс-функции, порядка 10^{-10}).

В общем случае сходимость алгоритма означает способность алгоритма приводить к результату за конечное число шагов [9, 10].

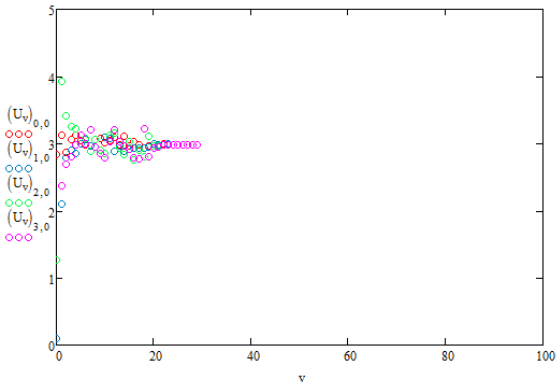


Рисунок 5. *График сходимости функции (разработано авторами)*

На рисунке 5 показан график сходимости *U* первых 4-х особей по итерациям (алгоритм завершился досрочно вследствие близости значений особей, по критерию остановки *D*).

Получаемая промежуточная популяция является исходной для дальнейшего выполнения операторов кроссинговера и мутации. Так как мутации применяются для всех параметров на основе нормального распределения, то становятся очевидны отклонения результата ГА от оптимума., популяция с большой вероятностью попадёт в оптимум, но также может оказаться и рядом с ним. На рисунке 6 показаны отклонения результата работы

ГА от среднего значения при варьировании шага отбора *b* с 1 до 4 и параметра агрессивности приближения *Rv* равного 1, 10, 100.

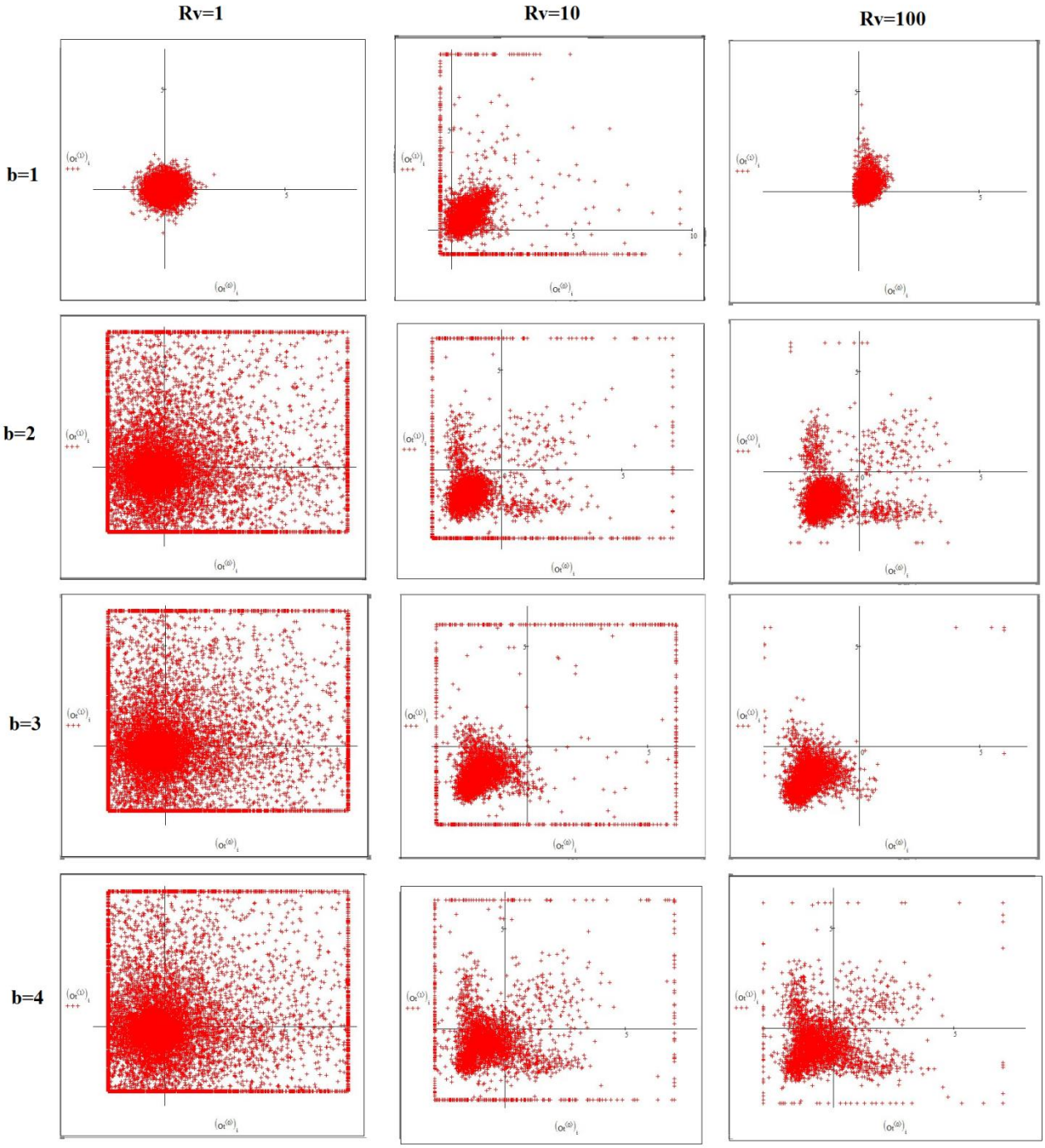


Рисунок 6. *Отклонения результата ГА от среднего (разработано авторами)*

На рисунке 6 показано отклонения результата ГА от среднего значения. Скопление точек в виде прямоугольника обусловлено введением границ, при пересечении которых особь принудительно помещается на соответствующую границу (эта процедура введена с целью ограничения разброса ГА и сосредоточения особей в интересуемой области поиска). Эволюцию искусственной популяции - поиска множества решений некоторой проблемы формально можно описать алгоритмом, который представлен на рисунке 7.

```
Gen(Condition, Ximin, Ximax) := | C ← Condition
                                | (v N b H Xi OPT Rv) ← (C0 C1 C2 C3 C4 C5 C6)
                                | R ← rnorm(N, (Ximax0 + Ximin0)/2, (Ximax0 − Ximin0)/4)
                                | for j ∈ 1..Xi − 1
                                |   | R ← augment(R, rnorm(N, (Ximaxj + Ximinj)/2, (Ximaxj − Ximinj)/4))
                                |   | R
                                | R_otb ← R
                                | L ← 0
                                | D ← 5
                                | while vv ≤ v ∧ L < 10
                                |   | for x ∈ 0..Xi − 1
                                |   |   | MIN ← Ximinx
                                |   |   | MAX ← Ximaxx
                                |   |   | k ← 0
                                |   |   | for i ∈ 0..N − 1 − H
                                |   |   |   for j ∈ i + 1..i + H
                                |   |   |     | mut ← rnorm[1, 0, |(R_otbi,x − R_otbj,x + 1) / 10^((−2i+(N−1−H))/(N−1−H))|]
                                |   |   |     | RV ← if(i = 0, 1, Rv^√L)
                                |   |   |     | Q ← (2·R_otbi,x + R_otbj,x)/3 + mut0,0·RV
                                |   |   |     | R_mutk,x ← if(Q ≥ MIN, if(Q ≤ MAX, Q, MAX), MIN)
                                |   |   |     | k ← k + 1
                                |   | for i ∈ 0..k − 1
                                |   |   Fi ← Fitness(i, R_mut)
                                |   | if vv/b = ceil(vv/b)
                                |   |   | Sort ← sort(F)
                                |   |   | Opt ← if(OPT = "Min", reverse(Sort), Sort)
                                |   |   | for i ∈ 0..N − 1
                                |   |   |   for j ∈ 0..k − 1
                                |   |   |     if Opti = Fj
                                |   |   |       | for x ∈ 0..Xi − 1
                                |   |   |       |   R_otbi,x ← R_mutj,x
                                |   |   |       | D ← (Σ_{i=0}^{N−2} √(Σ_{j=0}^{Xi−1} (R_otbi+1,j − R_otbi,j)^2)) / (N − 2)  if vv ≥ 20
                                |   |   |       | D ← 5  otherwise
                                |   |   |       | R_otb
                                |   |   | R_otb
                                |   | L ← L + 1  if D ≤ 1
                                |   | R_otb ← R_mut  otherwise
                                |   | Historyvv ← R_otb
                                |   | vv ← vv + 1
                                | History
```

Рисунок 7. *Математическая модель генетического алгоритма (разработано авторами)*

Генетический алгоритм (рисунок 7) рассчитан на оптимизацию по *Xi* параметрам (переменным, хромосомам) в течении *v* итераций, включает в себя следующие шаги:

1) формирование начальной популяции (*N* особей с количеством хромосом равным *Xi*) по закону нормального распределения от точки (*Ximin*+*Ximax*)/2 с матожиданием (*Ximax*-*Ximin*)/4;

2) скрещивание *i*-той особи с *H* соседями спереди, начиная с *i*+1;

3) адаптивные мутации, зависящие от разницы значений скрещиваемых хромосом (*R_otbi,x*-*R_otbj,x*) и положения особи в ранее отсортированной популяции [(-*2i*+*N-1-H*)/(*N-1-H*), при *i*=0 это выражение равно 1, при *i*=*N*-1-*H* это выражение равно -1], худшие особи имеют в 100 раз большие мутации нежели лучшие, это позволяет сократить время поиска и увеличить сходимость результата;

4) сортировка популяции производится по величине фитнесс-функции так, что особи с высоким значением фитнесс-функции помещаются вначале списка, а особи с низким значением фитнесс-функции, соответственно, в конце списка, отбирается всего *N* особей. Сортировка может производиться не на каждом шаге, а на каждом *b*-том шаге во время итераций между сортировкой и отбором (если таковые имеются, *b*>1). Величина популяции растет по закону (*N*-*H*)·*H* на каждой итерации, далее для скрещивания отбираются всё те же *N*-*H* особей, и снова каждая особь скрещивается с *H* соседями. Общее выражение для величины популяции *K,* зависимой от шага сортировки, отбора *b,* величины отбираемой популяции *N* и количество соседей для скрещевания *H,* можно представить как:

$$K(b,N,H) := N\cdot H^{b} - \sum_{j=2}^{b+1} H^{j} \qquad (2)$$

5) вместе с сортировкой, на каждом *b*-том шаге производится вычисление среднего расстояния *D* между особью *i* и особью *i*+1, производится измерение длин *N*-1 отрезков в *Xi*-мерном Евклидовом пространстве;

6) если не отработали *v* итераций, и не сработало досрочное условие прерывания работы алгоритма, то на шаг 2). Досрочное прерывание работы ГА и вывод полученной информации производится с целью уменьшения времени ожидания финала оптимизации, и производится по условию высокой плотности найденных точек в окрестностях друг друга, т.е. *D*<=1, при этом условие начинает выполнятся не сразу после начала работы ГА, а по происшествии 20 итераций (*vv*>=20), в общем случае *vv* растет от *vv*=0 до *vv*=*v*, без учета досрочного прерывания работы ГА;

7) Выход.

Визуализация движения популяции к оптимуму осуществляется при помощи функции *Soc* (рисунок 8).

```
Soc(xmin, Nx, xmax, ymin, Ny, ymax, Mut) := | StepX ← (xmax − xmin)/Nx
                                            | StepY ← (ymax − ymin)/Ny
                                            | for i ∈ 0..Ny − 1
                                            |   for j ∈ 0..Nx − 1
                                            |     | X ← xmin + j·StepX
                                            |     | Y ← ymin + i·StepY
                                            |     | Z_{i,j} ← Funkcia((X; Y))
                                            | for i ∈ 0..rows(Mut) − 1
                                            |   | JJ ← trunc((Mut_{i,0} − xmin)/StepX)
                                            |   | II ← trunc((Mut_{i,1} − ymin)/StepY)
                                            |   | I ← if(II ≥ 0, if(II ≤ Ny − 1, II, Ny), 0)
                                            |   | J ← if(JJ ≥ 0, if(JJ ≤ Nx − 1, JJ, Nx), 0)
                                            |   | X ← J·StepX + xmin
                                            |   | Y ← I·StepY + ymin
                                            |   | M_{I,J} ← Funkcia((X; Y))
                                            | (Z  M)^T
```

Рисунок 8. *Вычисление функции Soc (разработано авторами)*

Функции *Soc* зависит от минимальных и максимальных ограничений (рисунок 9), а также от количества точек и результата деятельности генетического алгоритма, а также отвечает за правильное формирование массивов поверхности функции (*Z*) и массива с точками (шариками) (*M*). Эта функция позволяет следить за выполнением ГА, а также формировать анимации деятельности алгоритма.

ГА работает до тех пор, пока не будет выполнено заданное количество поколений (итераций) процесса эволюции, или на некоторой генерации будет получено заданное качество, или вследствие преждевременной сходимости при попадании в некоторый локальный оптимум (рисунок 10).

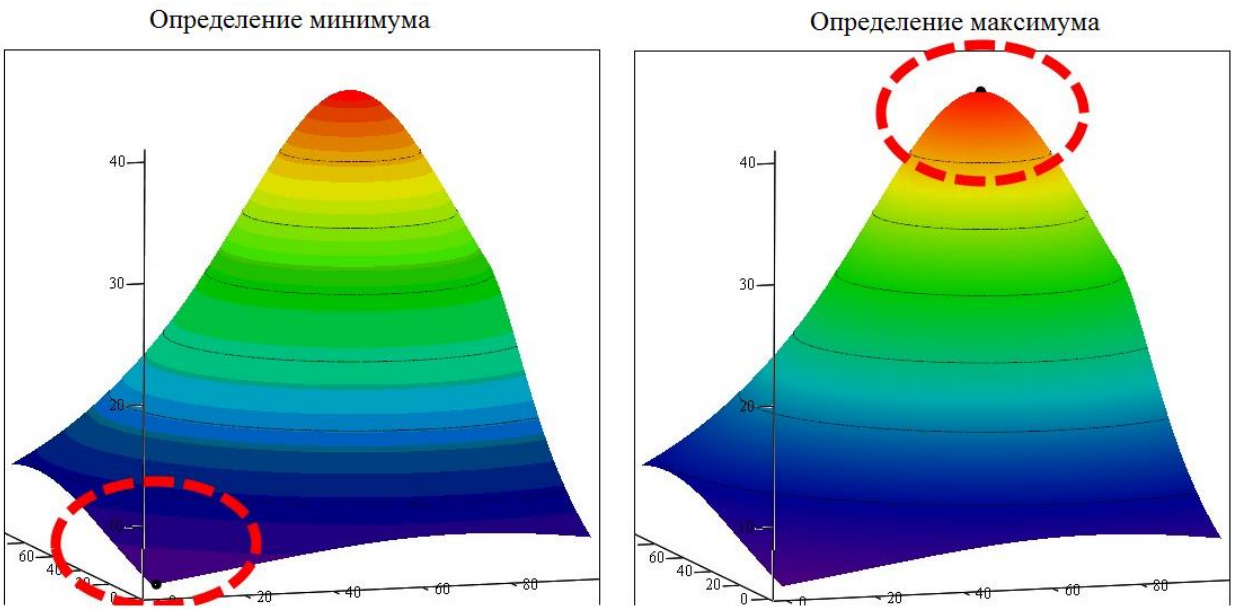
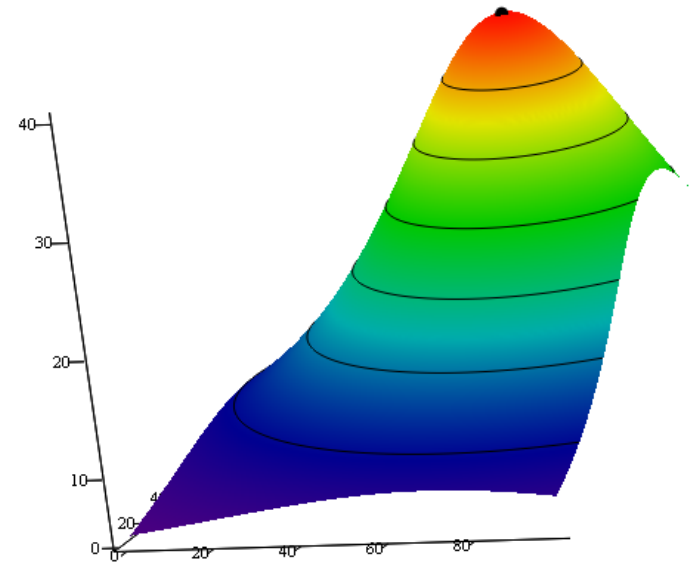


Рисунок 9. *Определение экстремумов функции (разработано авторами)*



U_{Rows-1} =

	0	1
0	2.986	2.005
1	2.986	2.005
2	2.986	2.005
3	2.986	2.005
4	2.986	2.005
5	2.986	2.005
6	2.986	2.005
7	2.986	2.005
8	2.986	2.005
9	2.986	2.005

DD_0, DD_1

Рисунок 10. *Положение особей популяции в конце эволюции (разработано авторами)*

Исследование эффективности работы ГА проводилось на тестовой многоэкстремальной функции (3):

$$f(x,y) = (\cos(2x^2) - 1.1)^2 + (\sin(0.5x) - 1.2)^2 - (\cos(2y^2) - 1.1)^2 + (\sin(0.5y) - 1.2)^2 \qquad (3)$$

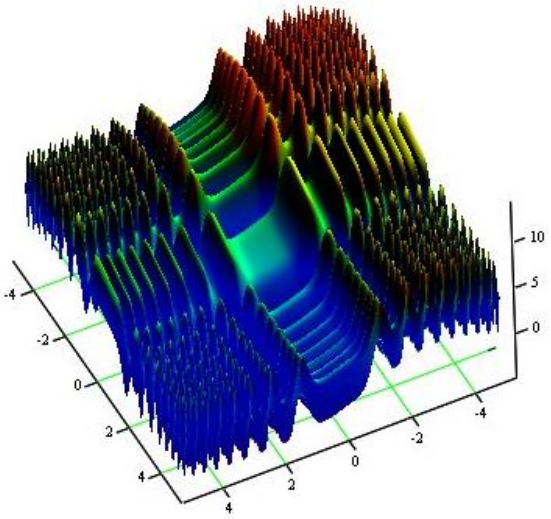


Рисунок 11. *Тестируемая многоэкстремальная функция (разработано авторами)*

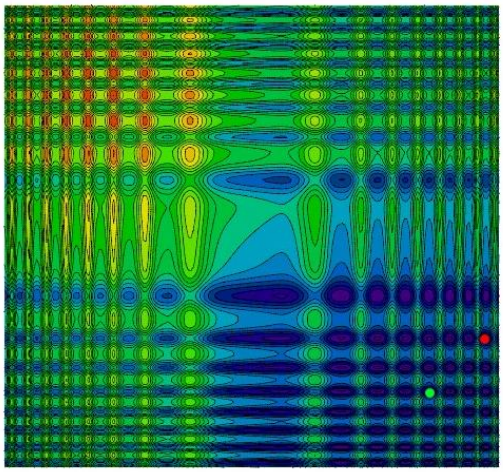


Рисунок 12. *Результат работы предложенного ГА (разработано авторами)*

На рисунке 12 показан результат определения глобального экстремума функции (3). Красной точкой обозначен найденный глобальный экстремум при помощи разработанного ГА, зеленой точной обозначен действительный глобальный минимум. Стоит отметить, что они почти идентичны по глубине.

Перед использованием ГА, в процессе определения экстремумов, целесообразно проводить анализ возможного вида зависимости, а также гладкости функции, чтобы использовать подходящие настройки и алгоритмы формирования и отбора популяции в ГА. Так же стоит иметь в виду некую вероятностную структуру ГА, так как случайные мутации могут приводить популяцию к противоречивым результатам в ряде запусков ГА.

Таким образом, в реализуемом алгоритме предложено условие досрочного прерывания ГА, которое производится по условию высокой плотности найденных точек в окрестностях друг друга.

Предложенный алгоритм позволяет определять точки экстремумов недифференцируемых и многоэкстремальных функций, а также определять максимальный или минимальный элемент матриц большого размера, элементы которых не являются случайными числами, а представляют собой данные с некоторыми зависимостями.

Продемонстрировано решение задачи оптимизации в системе автоматизированного проектирования Mathcad для функций одним и несколькими экстремумами.

ЛИТЕРАТУРА

1. Батищев, Д.И. Генетические алгоритмы решения экстремальных задач: Учеб. пособие / Д.И. Батищев; Воронеж. гос. техн. ун-т, Нижегор. гос. ун-т. - Воронеж: ВГТУ, 1995. - 69 с.
2. Гладков, Л.А. Генетические алгоритмы / Л.А. Гладков, В.В Курейчик, В.М. Курейчик - 2-е изд., испр и доп. - М.: ФИЗМАТЛИТ, 2006. - 320 с.
3. Уральский Н.Б., Сизов В.А., Капустин Н.К. Применение модифицированного генетического алгоритма для распараллеливания задачи умножения матриц большой размерности в гетерогенных системах обработки данных // Интернет-журнал «НАУКОВЕДЕНИЕ» Том 8, №2 (2016) http://naukovedenie.ru/PDF/62TVN216.pdf (доступ свободный). Загл. с экрана. Яз. рус., англ. DOI: 10.15862/62TVN216.
4. Курейчик, В.В. Эволюционные методы принятия решений с синергетическими и гомеостатическими принципами управления / В.В. Курейчик // Перспективные информационные технологии и интеллектуальные системы. - 2002. - №5. - С. 6-10.
5. Генералов, К.А. Математическое обеспечение и программные средства реализации генетических алгоритмов на основе теории нумерации. [Текст]: дис. … канд. техн. наук: 05.13.17, 05.13.11: защищена 15.12.09: утв. 12.03.10 / Генералов Константин Александрович. - Пенза, 2009. - 178 с.
6. Дьячков Ю.А., Семёнов А.А., Генералова, А.А. Прикладная оптимизация в проектировании колесных машин. Учеб. пособие. - М.: Мир науки, 2016. - 210 с.
7. Генералов, К.А. Создание начальной популяции генетических алгоритмов на основе плотности распределения / К.А. Генералов // Вычислительные системы и технологии обработки информации: межвуз. сб. науч. тр. - Вып. 6. - Пенза: Информационно-издательский центр ПензГУ, 2006. - С. 5-11.
8. Лебедев, Б.К. Формирование гомологических структур хромосом при кодировании списков / Б.К. Лебедев // Перспективные информационные технологии и интеллектуальные системы. - 2003. - №11. - С. 24-32.
9. Генералов, К.А. Анализ эффективности использования генетических алгоритмов в задачах при использовании различных языков программирования / К.А. Генералов // Вопросы современной науки и практики. Университет им. В.И. Вернандского. Том 2. Серия Технические науки. - 2008. - №12 (2) - С. 148-155.
10. Генералов, К.А. Специализированный язык программирования как наиболее эффективное средство использования генетических алгоритмов / К.А. Генералов // Известия высших учебных заведений. Поволжский регион. Технические науки. - 2008. - №3 - С. 25-32.

Koshev Alexander Nikolaevich
Penza state university of architecture and construction, Russia, Penza
E-mail: koshev@pguas.ru

Salmin Vladimir Vasilevich
Penza state university, Russia, Penza
E-mail: salmin-penza@yandex.ru

Generalova Aleksandra Aleksandrovna
Penza state university, Russia, Penza
Penza state university of architecture and construction, Russia, Penza
E-mail: generalova_aa@mail.ru

Bichkov Dmitriy Sergeevich
Penza state university, Russia, Penza
E-mail: deciptikon@mail.ru

The development of genetic algorithm with adaptive mutations to determine the global extremum function of n-variables

Abstract. Most of the problems of science and technology relates to a broad class of search for optimal solutions of problems, i.e. to optimization problems. Among them, there are problems in which the use of traditional methods of searching for solutions ineffective or practically impossible (for example, tasks, search for solutions which is associated with a bust). To find the best solution, as a rule, carried out aiming, casual and combinatorial search. In this connection, a large number of developed techniques making these tasks.

One of the most effective mechanisms for solving optimization problems is the mechanism of genetic algorithms. Genetic algorithms operate on objects such as "gene", "chromosome", "population". The principle of operation is based on the ideas of evolution of living nature.

The article is devoted to the genetic algorithm. The question of the generation of the initial population. Available to existing methods of forming the initial population and a new method - initialization of the population on the basis of a given distribution law.

For the experiment a program was written that implements the genetic algorithm. As test functions taken one extreme and multiextremal function. The results show that the generation of the initial population on the basis of the law of distribution increases the precision and reduces working time of the algorithm.

In the proposed algorithm the proposed condition for termination of genetic algorithm, which is produced by the high density of points found in the vicinity of each other.

Keywords: extremum; genetic algorithm; adaptive mutation; the convergence of functions; fitness function