Санкт-Петербургский государственный университет

Факультет прикладной математики – процессов управления

**Лабораторная работа №3**

**Отчет**

**по дисциплине «Алгоритмы и структуры данных»**

«Исследование генетического алгоритма. Изучение различных кодировок генотипа»

Вариант 7

Автор работы: Дацык Р.В.

Группа: 22.Б15-пу

Преподаватель: Дик А.Г.

Санкт-Петербург, 2023

**Оглавление**

[**1.** **Цель работы** 3](#_Toc151824592)

[**2.** **Задачи** 3](#_Toc151824593)

[**3.** **Введение** 3](#_Toc151824594)

[**4.** **Теоретическая часть** 3](#_Toc151824595)

[**5.** **Алгоритм метода** 4](#_Toc151824596)

[**6.** **Описание программы** 6](#_Toc151824597)

[**7.** **Рекомендации пользователю** 7](#_Toc151824598)

[**8.** **Рекомендации программисту** 8](#_Toc151824599)

[**9.** **Контрольный пример** 8](#_Toc151824600)

[**10.** **Вывод** 12](#_Toc151824601)

[**11.** **Список использованной литературы** 13](#_Toc151824602)

# **Цель работы**

Исследование двух основных способов кодирования генотипа хромосом в генетическом алгоритме и проверка их эффективности.

# **Задачи**

* Изучение особенностей кодирования генетических алгоритмов.
* Написание программы поиска минимума функции с использованием генетического алгоритма.
* Тестирование программы на выбранной тестовой функции.
* Анализ результатов работы алгоритма и предложение улучшений.

# **Введение**

Генетические алгоритмы являются эффективными методами оптимизации, вдохновленными принципами естественного отбора. Они применяются в различных областях, включая поиск оптимальных параметров в задачах оптимизации.

# **Теоретическая часть**

### **4.1. Особенности генетических алгоритмов**

Генетические алгоритмы являются методом оптимизации, использующим принципы естественного отбора и генетики. Основные компоненты алгоритма включают в себя популяцию, функцию приспособленности, операторы селекции, кроссинговера и мутации.

### **4.2. Кодирование генотипа хромосом**

Кодирование генотипа определяет, как представить индивида (хромосому) в алгоритме. В данной работе рассматриваются два типа кодирования: вещественное и логарифмическое. Вещественное кодирование используется для представления реальных чисел, в то время как логарифмическое позволяет работать с широким диапазоном значений.

# **Алгоритм метода**

1. **Инициализация популяции:** Создается начальная популяция индивидов с учетом выбранной кодировки генотипа.
2. **Оценка приспособленности:** Каждый индивид оценивается на основе функции приспособленности, которая определяется целевой функцией.
3. **Селекция:** Выбираются родители для создания нового поколения. В данной реализации используется турнирный отбор.
4. **Кроссинговер:** Происходит скрещивание выбранных родителей для создания потомства.
5. **Мутация:** Происходит случайное изменение генов потомства с учетом заданной вероятности мутации.
6. **Замена поколения:** Новое поколение заменяет старое.
7. **Повторение:** Шаги 2-6 повторяются заданное количество раз (число поколений).

На рисунке 5.1 представлена блок-схема алгоритма.

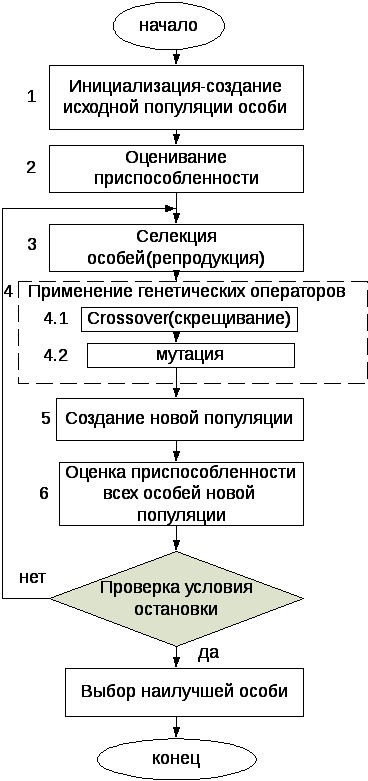


Рисунок 5*.*1 Блок-схема логики генетического алгоритма.

# **Описание программы**

Алгоритм реализован на языке python 3.10 с использованием следующих пакетов: numpy и tkinter.

В коде представлен набор функций, используемых в генетическом алгоритме для оптимизации функции. Давайте рассмотрим каждую из них:

Таблица 6.1. Описание функций

|  |  |  |  |
| --- | --- | --- | --- |
| Имя функции | Аргумент | Возвращаемое значение | Описание функции |
| **clear\_entry(entry, value)** | entry - объект Entry из библиотеки Tkinter, value - значение, которое нужно установить в поле ввода. | None | Функция очищает содержимое виджета Entry и устанавливает в него новое значение. |
| **update\_table(data)** | data - данные, которые необходимо отобразить в Treeview. | None | Функция обновляет содержимое Treeview виджета, удаляя предыдущие строки и добавляя новые данные. |
| **run\_genetic\_algorithm(encoding\_type)** | encoding\_type - тип кодирования, строковое значение ('logarithmic' или 'real'). | None | Функция запускает генетический алгоритм с заданными параметрами и обновляет интерфейс с результатами. |
| **enforce\_bounds(individual)** | individual - массив, представляющий генотип особи. | Массив, представляющий генотип особи, где значения находятся в пределах установленных ограничений. | Функция приводит значения генов в пределах установленных ограничений (нижней и верхней границ) |
| **crossover(parent1, parent2)** | parent1 и parent2 - массивы, представляющие генотипы родителей. | Кортеж из двух массивов, представляющих генотипы потомков после кроссовера. | Функция выполняет одноточечный кроссовер между родителями и возвращает двух потомков |
| **mutate(individual)** | individual - массив, представляющий генотип особи. | Массив, представляющий мутировавший генотип особи. | Функция вносит мутации в генотип особи с использованием маски вероятности мутации. |
| **select\_parents(population, fitness\_values)** | population - массив особей, fitness\_values - значения приспособленности особей. | Одномерный массив, представляющий генотип выбранного родителя. | Функция использует турнирный отбор для выбора родителя из популяции. |
| **evaluate\_fitness(x, y)** | x и y - значения генов особи | Значение целевой функции для заданных генов. | Функция вычисляет значение целевой функции для особи с заданными значениями генов. |
| **initialize\_population()** | None | Массив, представляющий начальную популяцию особей. | Функция инициализирует начальную популяцию особей с учетом выбранного типа кодирования. |

# 

# **Рекомендации пользователю**

Для успешного запуска программы необходимо устройство с операционной системой Linux, macOS или Windows, а также среда разработки, поддерживающей запуск python 3.10

Для запуска рассчета функции требуется указать вероятность мутации, количество хромосом, минимальное значение гена, максимальное значение гена, количество поколений, а затем нажать кнопку рассчитать в вещественном или логарифмическом кодировании. Как только программа завершит свое выполнение, она выведет таблицу с результатами разных поколений, а также в какой точке достигается лучшее решение.

# **Рекомендации программисту**

Для запуска программы необходима 64-битная операционная система Windows, Linux или macOS. Для работы с кодом необходима среда разработки, совместимая с python 3.10 и библиотеки numpy и tkinter.

Исходный код программы и необходимые текстовые файлы доступны по ссылке:

<https://github.com/CapTopGrade/Algorithms/tree/main/4%20lab>

# **Контрольный пример**

Произведено тестирование программы на тестовой функции . Результаты включают в себя число итераций и количество вычислений целевой функции.

Все иллюстрации представлены ниже в контрольном примере.

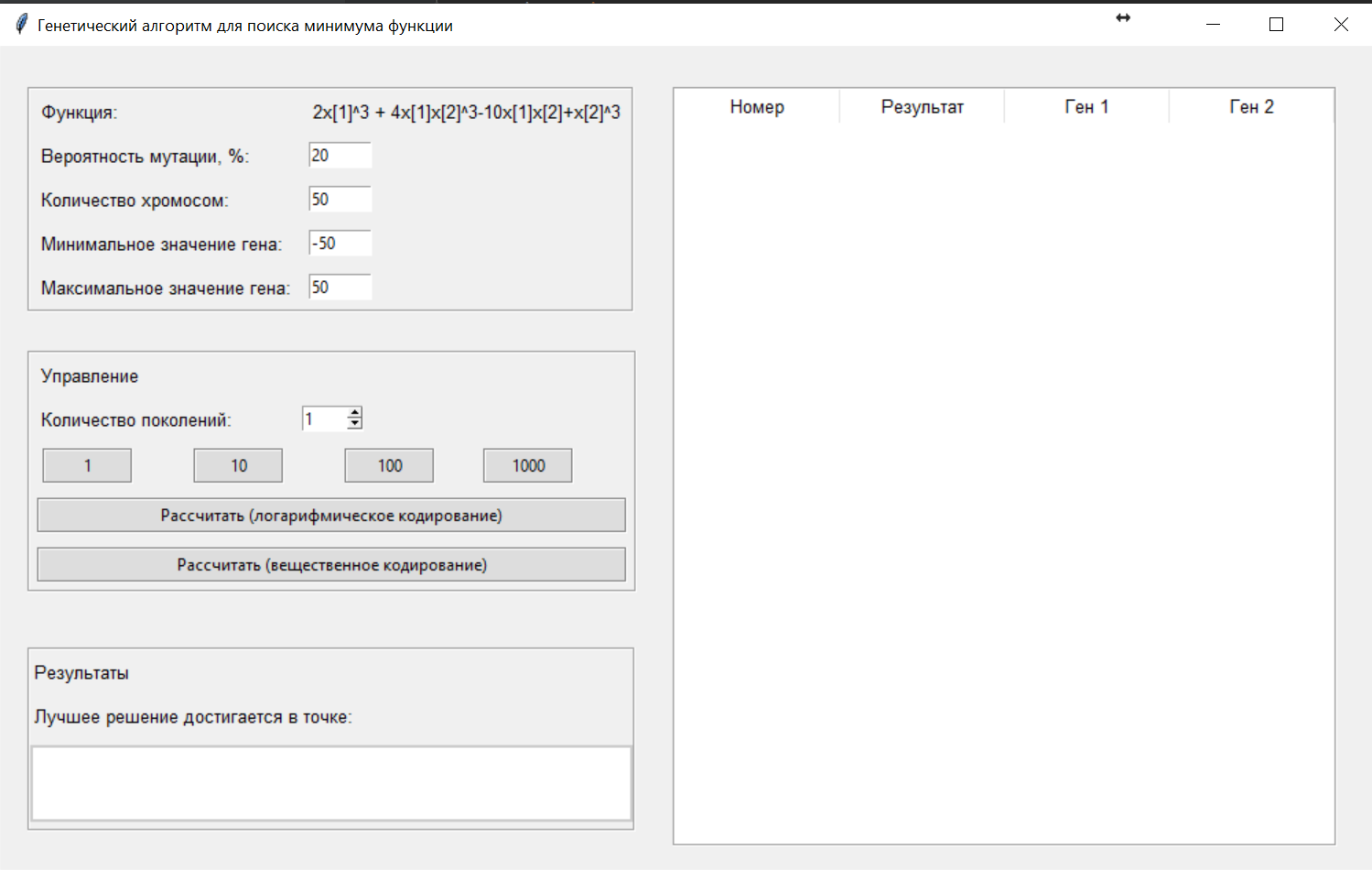


Рис. 9.1 Интерфейс программы

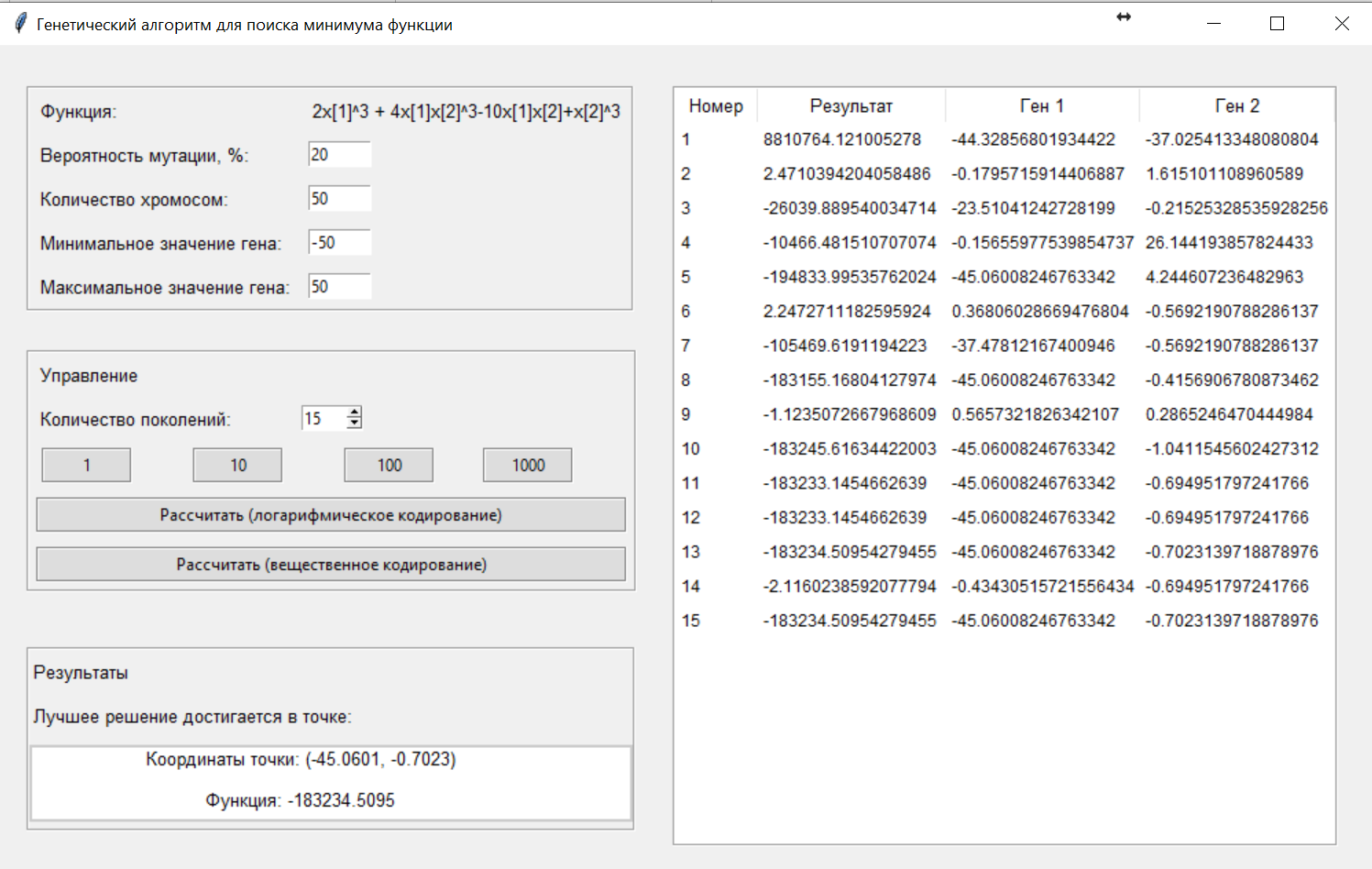


Рис. 9.2 Пример вывода программы при вещественном кодировании

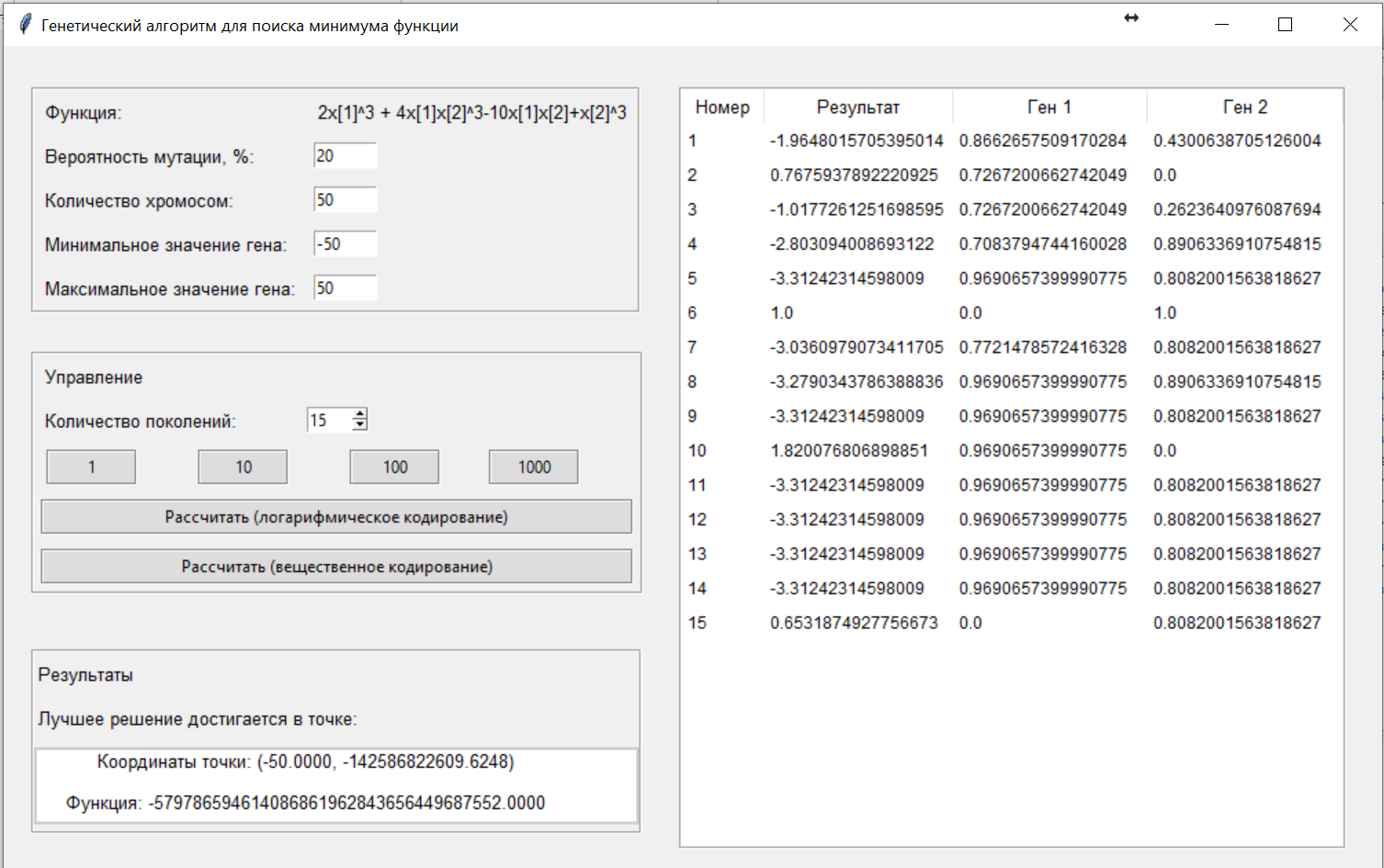


Рис. 9.3 Пример вывода программы при логарифмическом кодировании

Программа успешно находит минимум функции, представленной тестовой функцией, с учетом выбранных параметров. Представлены результаты для каждого поколения, включая номер поколения, лучший результат, а также значения генов, обеспечивающих минимум функции.

# **Вывод**

В результате проведенной работы были исследованы два способа кодирования генотипа хромосом в генетическом алгоритме: вещественное и логарифмическое. Программа успешно находит минимум функции, а результаты тестирования представлены в отчете. Рекомендации пользователям и программистам помогут улучшить использование и эффективность программы.

# **Список использованной литературы**

[1] Описание работы генетического алгоритма: <https://loginom.ru/blog/ga-math>

[2] Документация библиотеки Tkinter: <https://docs.python.org/3/library/tkinter.html>

[3] Основные понятия генетических алгоритмов: <https://studylib.ru/doc/2222704/tema-3.-lekciya-11.-osnovnye-ponyatiya-geneticheskih-algoritmov>