**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по учебной практике**

Тема: Генетические алгоритмы в задаче поиска минимального остовного дерева

|  |  |  |
| --- | --- | --- |
| Студент гр. 3388 |  | Трунов Б.Г. |
| Преподаватель |  | Жангиров Т.Р. |

Санкт-Петербург

2025

**АННОТАЦИЯ**

Цель практики: - освоение генетических алгоритмов на примере решения задачи поиска минимального остовного дерева. В ходе практики изучены: функция приспособленности, понятие хромосомы, гена, отбора, скрещивания и мутации. Для данной задачи также был разработан графический интерфейс с помощью библиотеки python3 Flet.

**1. Выбор и использование Flet для GUI**

Для создания графического интерфейса приложения был выбран фреймворк **Flet**.  
**Flet** — это современная библиотека для Python, позволяющая быстро создавать кроссплатформенные GUI-приложения с использованием декларативного подхода, схожего с Flutter.

Причины выбора Flet:

* **Простота и скорость разработки**: Flet позволяет создавать сложные интерфейсы с минимальным количеством кода.
* **Кроссплатформенность**: Приложение работает на Windows, Linux, macOS и в браузере без изменений кода.
* **Современный внешний вид**: Flet предоставляет готовые компоненты с современным дизайном.
* **Гибкость**: Легко интегрируется с другими Python-библиотеками (например, для визуализации графов используется matplotlib и networkx).
* **Поддержка реактивности**: Изменения состояния автоматически отражаются в интерфейсе.

**2. Реализованный функционал GUI**

Структура приложения

* **Модульная архитектура**: Код разделен на логические модули (страницы, компоненты, конфиг, роутинг).
* **Навигация**: Используется система маршрутов ([routes.py](vscode-file://vscode-app/c:/Users/User/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html)) для перехода между страницами:
  + Главная страница алгоритма
  + Страница конфигурации параметров
  + Страница 404 (ошибка)
* Основные компоненты интерфейса
  + **Боковое меню** ([menu\_component.py](vscode-file://vscode-app/c:/Users/User/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html)): Быстрый переход между основными разделами.
  + **Главная страница** ([home.py](vscode-file://vscode-app/c:/Users/User/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html)):
    - Отображение текущего графа.
    - Панель управления запуском алгоритма (кнопки запуска, перехода по шагам, область для отладочных сообщений).
* **Страница конфигурации** ([config\_page.py](vscode-file://vscode-app/c:/Users/User/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html)):
  + Настройка параметров генетического алгоритма (вероятности, размеры популяции, количество поколений, тип отбора).
  + Генерация случайного графа, загрузка/сохранение графа в файл.
  + Визуализация графа.
* **Визуализация графа** ([graph\_component.py](vscode-file://vscode-app/c:/Users/User/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html), [graphs.py](vscode-file://vscode-app/c:/Users/User/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html)):
  + Используется matplotlib и networkx для отрисовки графа, изображение встраивается в интерфейс через base64.
* **Кастомные компоненты**:
  + - Кнопки ([button.py](vscode-file://vscode-app/c:/Users/User/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html))
    - Поля ввода ([text\_input.py](vscode-file://vscode-app/c:/Users/User/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html))
    - Слой страницы ([page\_layer.py](vscode-file://vscode-app/c:/Users/User/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html))

Пример реализованного интерфейса:

Современный внешний вид, адаптивная верстка (используются ResponsiveRow, Container).

Все параметры и действия пользователя мгновенно отражаются в интерфейсе.

**3. Текущее состояние программы**

**Готова архитектура приложения**: реализованы все основные страницы, компоненты, навигация.

**Реализована работа с графом**: генерация, визуализация, загрузка и сохранение.

**Реализована настройка параметров генетического алгоритма**: все параметры можно менять через GUI.

**Подготовлена панель управления запуском алгоритма**: кнопки запуска, перехода по шагам, область для вывода сообщений.

**4. Планы по доработке**

**Добавление самой реализации генетического алгоритма**:

Реализация логики работы алгоритма поиска минимального остовного дерева с использованием выбранных параметров.

Интеграция алгоритма с интерфейсом: запуск, пошаговое выполнение, отображение промежуточных и финальных результатов.

Вывод отладочных сообщений и визуализация процесса работы алгоритма.

**5. Вывод**

Использование Flet позволило быстро создать современный, удобный и расширяемый интерфейс для работы с графами и настройки параметров генетического алгоритма.  
На данный момент реализована вся необходимая инфраструктура для дальнейшей интеграции и визуализации работы самого алгоритма.  
Дальнейшая работа будет направлена на реализацию и интеграцию генетического алгоритма в существующий GUI.

**Демонстрация GUI на момент 30.07.2025**

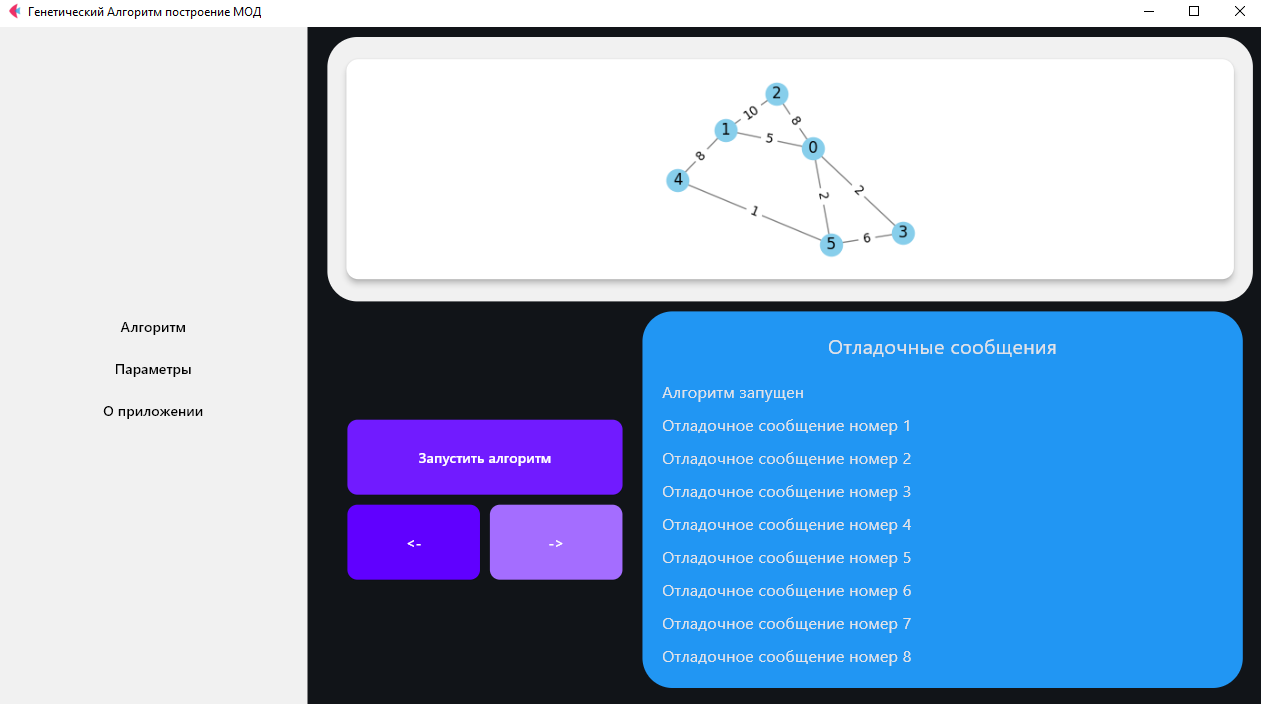


Рис.1 Основная страница работы алгоритма

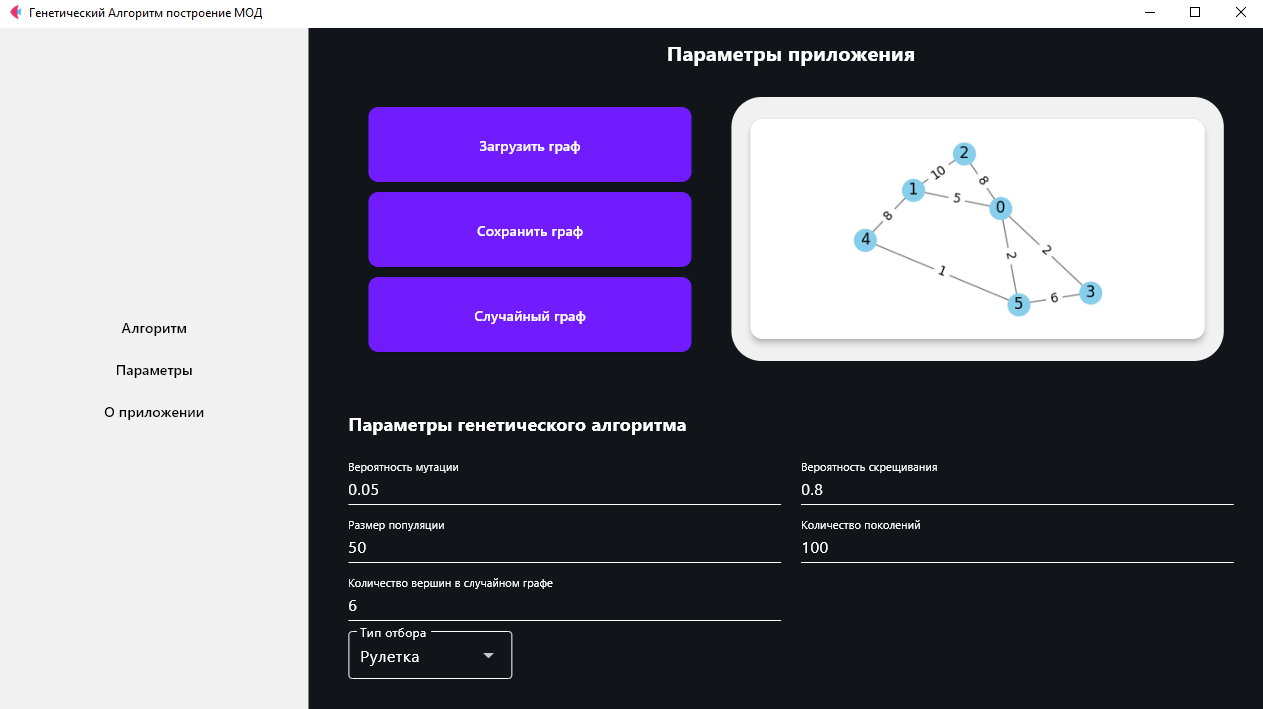


Рис.2 Страница с параметрами генетического алгоритма

Далее был разработан класс *GeneticAlgorithmMST* со следующими возможностями:

* Генерация случайных графов
* Инициализация популяции случайными остовными деревьями
* Реализация операторов генетического алгоритма:
* Селекция (турнирная и рулетка)
* Скрещивание (одноточечное)
* Мутация (инверсия битов)
* Проверка корректности остовного дерева
* Вычисление приспособленности особи

Интеграция алгоритма в GUI

Алгоритм был интегрирован в графический интерфейс с возможностями:

* Запуск алгоритма с настраиваемыми параметрами
* Пошаговое выполнение алгоритма
* Визуализация процесса эволюции
* Отображение лучшего найденного решения
* История изменения приспособленности

Доработка интерфейса и исправление ошибок

На данном этапе были выполнены следующие улучшения:

Исправление ошибки отображения графа на странице конфигурации. Была обнаружена и исправлена ошибка, при которой после загрузки или генерации графа на странице конфигурации граф не отображался сразу, а только после перезахода на роут. Проблема заключалась в том, что компонент графа не обновлялся после изменения данных. Решение включало:

* Принудительное обновление состояния страницы
* Очистка кэша изображений графа
* Пересоздание представления страницы

Улучшения интерфейса:

* Добавлена возможность загрузки и сохранения графов в формате JSON
* Реализована генерация случайных графов с настраиваемым количеством вершин
* Добавлены элементы управления для пошагового выполнения алгоритма
* Улучшена визуализация процесса эволюции с помощью слайдера
* Добавлено отображение статистики работы алгоритма
* 6. Финальная доработка и тестирование
* На завершающем этапе были выполнены:
* Тестирование всех функций приложения
* Оптимизация производительности
* Документирование кода
* Создание констант для параметров по умолчанию

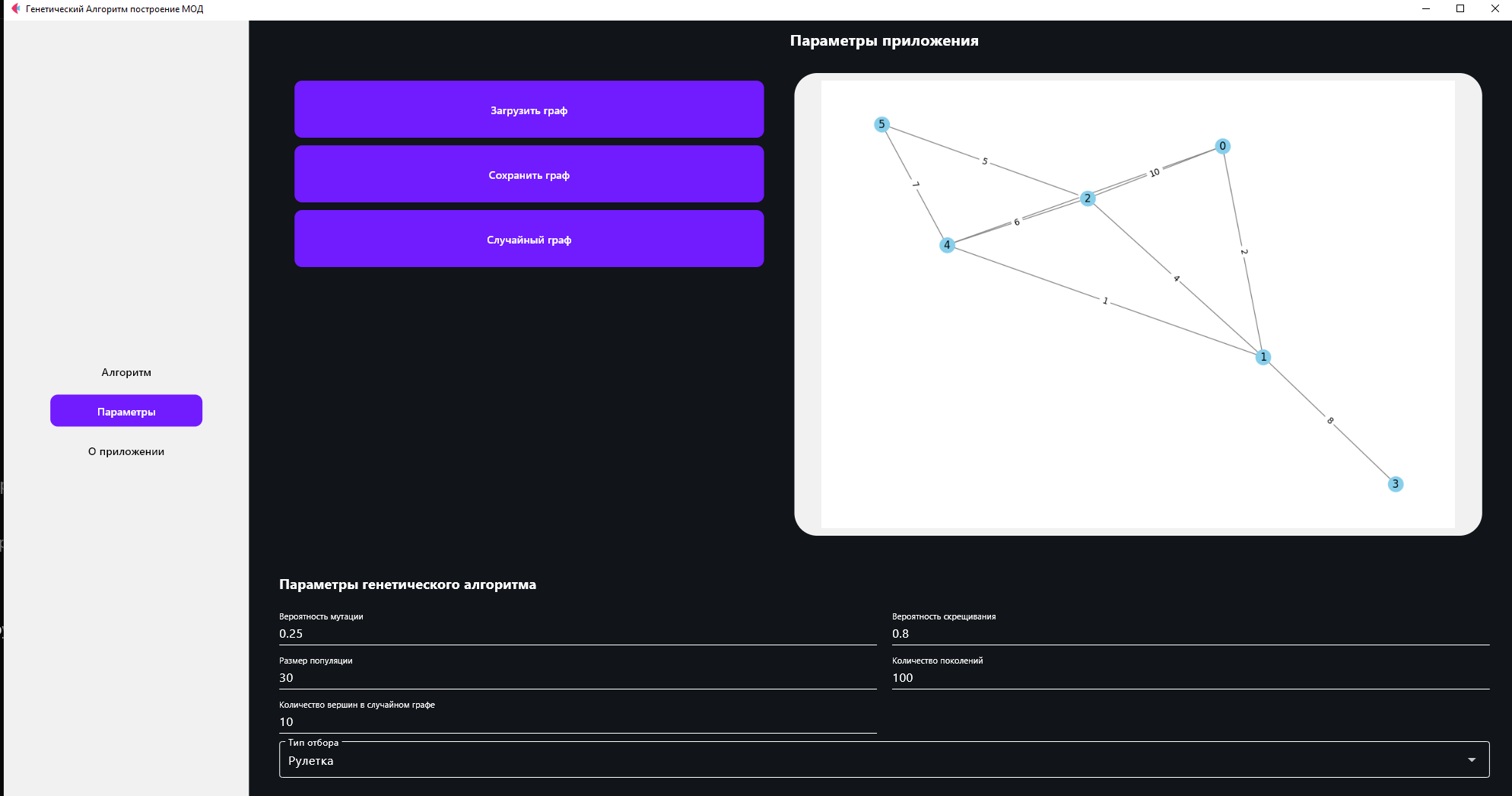


Рис.3 Итоговая версия GUI, страница “Параметры”.

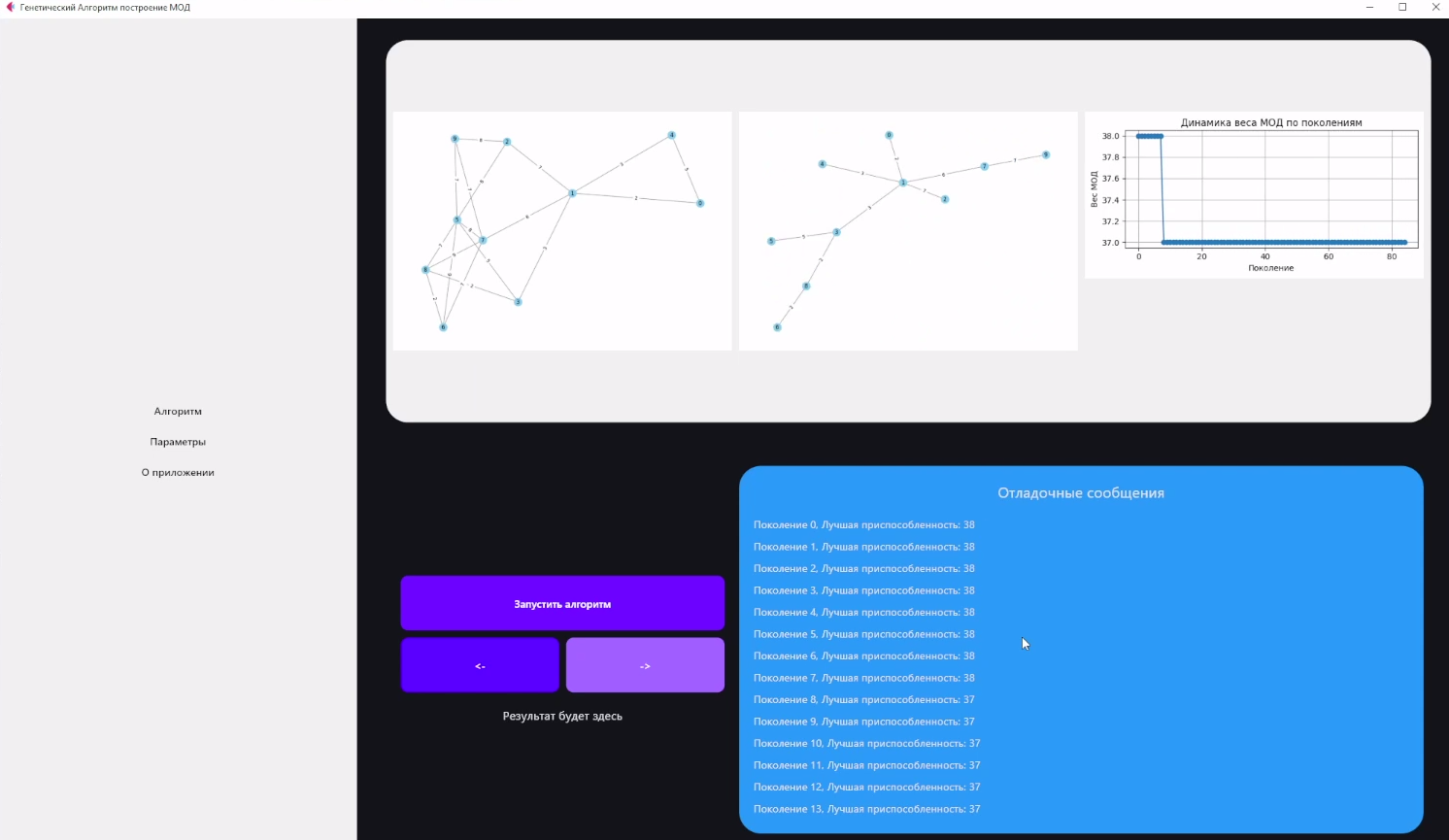


Рис.4 Итоговая версия GUI, главная страница.

**Выводы**

В ходе выполнения работы был реализован программный комплекс для поиска минимального остовного дерева (МОД) с помощью генетического алгоритма и современного графического интерфейса на Python (Flet). В процессе разработки были достигнуты следующие результаты:

Разработана архитектура приложения

Проект построен по модульному принципу, что обеспечивает удобство поддержки и расширения функционала. Были реализованы отдельные модули для логики алгоритма, визуализации, управления параметрами и взаимодействия с пользователем.

Реализован генетический алгоритм для задачи МОД

Алгоритм поддерживает настройку основных параметров (размер популяции, вероятность мутации и скрещивания, количество поколений, тип селекции). Особое внимание уделено корректной работе операторов мутации и кроссовера, чтобы в популяции всегда присутствовали только валидные остовные деревья.

Создан удобный и наглядный интерфейс

Пользователь может легко настраивать параметры, загружать и сохранять графы, запускать алгоритм и наблюдать за процессом эволюции в реальном времени. Визуализация графа, остовного дерева и динамики веса МОД по поколениям делает работу с программой интуитивно понятной.

Проведена отладка и устранение ошибок

В процессе тестирования были выявлены и исправлены критические ошибки, связанные с застоем популяции и отсутствием эволюции. Внесены улучшения в селекцию, мутацию и кроссовер, что позволило добиться реального поиска оптимального решения.

Обеспечена стабильная работа алгоритма

После внесённых исправлений алгоритм корректно ищет минимальное остовное дерево, а веса МОД действительно изменяются по поколениям, что подтверждает эффективность реализованного подхода.

В целом, поставленные задачи были успешно решены: создано современное, гибкое и наглядное приложение для решения задачи поиска МОД с помощью генетического алгоритма.