

**МИНОБРНАУКИ РОССИИ**  
**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**  
**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**  
**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**  
**Кафедра МО ЭВМ**

**ОТЧЕТ**  
**по летней практике**  
**Тема: Генетические алгоритмы**

Студенты гр. 0382

Азаров М.С.  
Шангичев В.А.  
Санников В.А.  
Диденко Д.В.

Преподаватель

Жангиров. Т.Р.

Санкт-Петербург

2022

### **Цель работы.**

Ознакомится с понятием *Генетического алгоритма (ГА)*. Решить поставленную задачу с помощью приобретенных знаний в области методов ГА. Разработать полноценное приложение решающую данную задачу , с удобным пользовательским интерфейсом и визуализацией работы алгоритма.

### **Задание.**

Разработать и реализовать программу, решающую одну из оптимизационных задач (файл “Варианты”) с использованием генетических алгоритмов (ГА), а также визуализирующая работу алгоритма. В качестве языков программирования можно выбрать: C++, C#, Python, Java, Kotlin. Все параметры ГА необходимо определить самостоятельно исходя из выбранного варианта. Разрешается брать один и тот же вариант разным бригадам, но при условии, что будут использоваться разные языки программирования.

### **Вариант: 11**

*Минимальное остовное дерево* (minimal spanning tree — MST) в связанном взвешенном неориентированном графе — это остовное дерево этого графа, имеющее минимальный возможный вес, где под весом дерева понимается сумма весов входящих в него рёбер.

Задача заключается в нахождении минимального остовного дерева для вводимого графа.

#### Входные данные:

- Список вершин
- Список ребер

**Язык:** Python

## РАЗДЕЛ 1. Формирование прототипа GUI и выбор метода решения задачи

### 1. Предварительный набросок интерфейса программы:

- Главное меню:

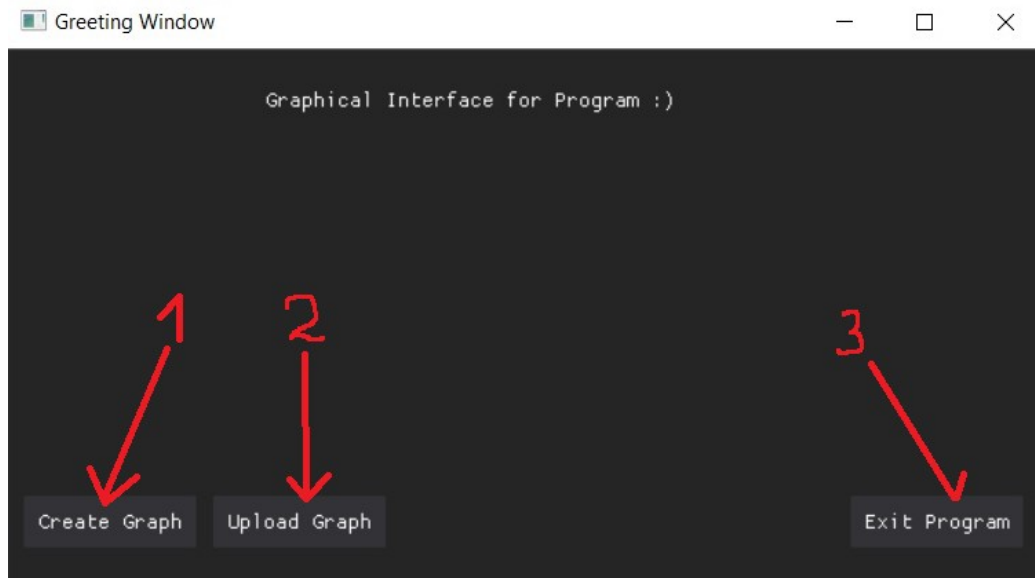


Рисунок 1: Предварительный набросок окна "Главное меню"

1 — Кнопка создания графа.

2 — Кнопка загрузки графа из файла.

3 — Кнопка выхода из программы.

- Основное окно программы.

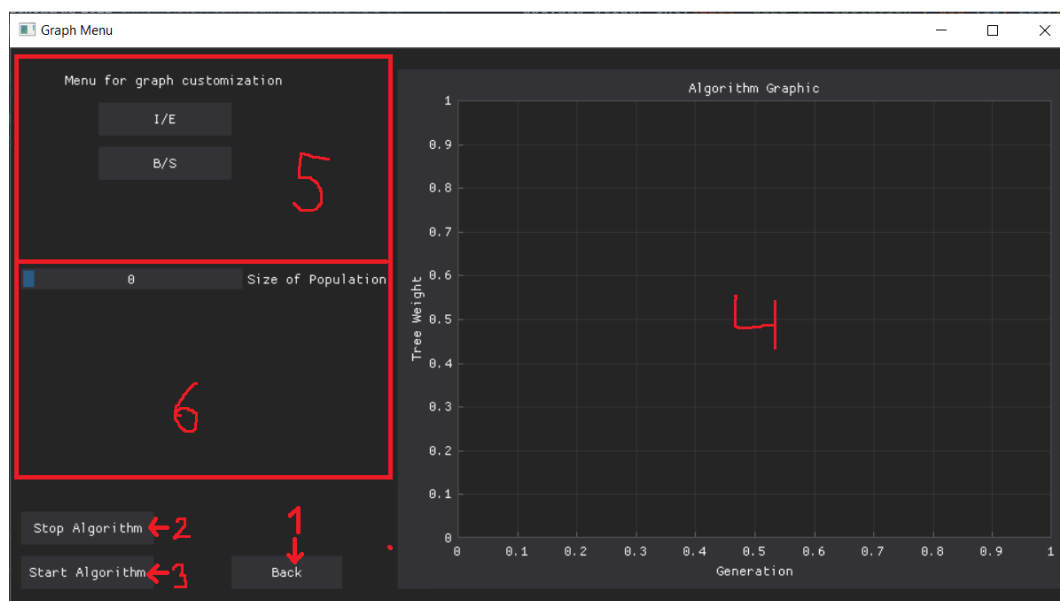


Рисунок 2: Предварительный набросок окна "Основное окно программы"

- 1 — Вернутся в главное меню.
- 2 — Досрочное завершение алгоритма.
- 3 — Кнопка отвечающая за запуск алгоритма.
- 4 — Область визуализации работы ГА.
- 5 — Область управлением визуализации алгоритма.
- 6 — Область настроек параметров алгоритма ГА.

## 2. Описание сценария взаимодействия пользователь с программой.

### **«Главное меню»**

При запуске приложения открывается первое окно «Главное меню». В нем пользователь может выбрать следующие действия:

- «создать граф» - при выборе этого действия будет открываться дополнительное окно для ввода графа . После успешного ввода пользователю откроется «Основное окно программы».
- «загрузить граф» - открываться проводник с возможностью выбора нужного файла. (Формат файла еще не уточнен и требует дальнейшего рассмотрения ) После выбора файла пользователю откроется «Основное окно программы».
- «выйти из программы» - завершение программы.

### **«Основное окно программы»**

После того как введен или загружен нужный граф пользователю открывается «Основное окно программы».

В этом окне будут присутствовать область настроек параметров алгоритма (см. Рисунок 2 , пункт 6), такие как размер популяции, вероятность кроссинговера и т. д. Эти параметры пользователю нужно будет указать (либо оставить значения по умолчанию) перед запуском алгоритма.

После указания параметров алгоритма пользователю нужно будет нажать кнопку «Запуска алгоритма» (см. Рисунок 2 , пункт 3), после которой запустится демонстрация работы алгоритма, а область настройки параметров алгоритма перестанет быть доступной для изменений.

Демонстрация алгоритма будет показываться справа в области 4 на Рисунке 2. Каждая особь будет отображаться точкой на координатной плоскости, где по оси Y будет шкала весов каждой особи (вес особи — вес графа соответствующей этой особи), а по оси X поколения . Чтобы увидеть граф соответствующей особи, нужно будет нажать на нужную особь и ее граф откроется в доп. окне.

Это очень удобное представление работы алгоритма , так как чтобы увидеть лучшую особь поколения нужно посмотреть на самую нижнюю точку в одном поколении. Родители, потомки (особи получившиеся в результате кроссинговера) и мутанты будут отображаться разными точками (будет легенда по которой можно понять кто есть кто).

Предполагается что визуализация алгоритма будет иметь два режима работы «Пошаговый» и «Автоматический». При пошаговом режиме каждый этап работы алгоритма и его последующая визуализация будет осуществляться по нажатию определенной клавиши. При автоматическом режиме пользователю не нужно будет нажимать клавиши , алгоритм будет переходить на следующий этап по истечению небольшого промежутка времени.

Управление режимами визуализации будет располагаться в области 5 Рисунок 2. Также в этой области будет находится кнопка «Перехода к концу алгоритма».

После схождения алгоритма откроется окно с окончательным ответом (минимальным остовным деревом). После нажатия кнопки «Ок» пользователь вернется на окно «Основное окно программы» , в котором уже

будут доступны для изменения параметры алгоритма . Таким образом пользователь сможет изменить параметры алгоритма для текущего загруженного графа или вернуться в «Главное меню» для указания нового графа.

Также во время выполнения алгоритма можно будет нажать кнопку «Досрочного завершения» , в случае если алгоритм не сходится. В таком случае алгоритм прекратит работу , параметры алгоритма будут доступны для изменения и пользователь сможет указать новые параметры алгоритма, либо загрузить другой граф.

### 3. Определение и обоснование параметров, модификации ГА для решения выбранной задачи

Для решения поставленной задачи будут проводиться эксперименты с различными вариантами кроссовера, отбора и мутации. Генетический код каждой особи представлен двоичным числом, количество разрядов в котором равно количеству ребер в графе. В качестве базового решения было решено использовать однородный кроссовер с отбором родителей по принципу рулетки с последующей двоичной мутацией с вероятностью мутации особи около 5%, где каждый ген мутирует с вероятностью 10%, и элитарным отбором с долей элитных особей 10%. Численность популяции планируется содержать от 100 особей, причем будут уместны эксперименты с нефиксированным количеством особей. Такой выбор параметров обоснован необходимостью поддержания широкого генетического разнообразия в генерациях, т. к. в произвольном графе (в худшем случае в полном) большинство наборов ребер не будут являться остовными деревьями, и в случае скудного генетического разнообразия в некоторых случаях дерево получить будет невозможно.

Пробная версия фитнес-функции будет выдавать  $\varepsilon$  (очень малое

положительное число), если генетический код особи не представляет дерево и  $1 / \langle \text{вес остоного дерева} \rangle$  в противном случае. Данная функция будет по мере необходимости дополняться. Например, интересна и достойна рассмотрения идея регуляции количества ребер. В таком случае необходимо будет определить, какое количество ребер у особей ведет к наиболее частому появлению генетических кодов, представляющих остовные деревья, и обеспечить сходимость к данному количеству ребер с помощью фитнес функции.