ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

**(МОСКОВСКИЙ ПОЛИТЕХ)**

Кафедра «Инфокогнитивные технологии»

Практические и лабораторные занятия по дисциплине

«Проектирование интеллектуальных систем»

Лабораторная работа № 1

**«Решение оптимизационных задач с помощью генетических** **алгоритмов»**

Группа 224-322

Студент Леонов Владислав Денисович

Преподаватель Кружалов Алексей Сергеевич

**Москва 2023**

**Цель**

Ознакомиться с подходом и приобрести практический навык решения оптимизационных задач с помощью генетических алгоритмов (ГА).

**Краткое описание**

Разработка программы, которая осуществляет поиск кратчайшего пути для информационного пакета (сообщения) в компьютерной сети с помощью генетических алгоритмов. Задача коммивояжёра - одна из самых известных задач комбинаторной оптимизации, заключающаяся в поиске самого выгодного маршрута, проходящего через указанные города хотя бы по одному разу с последующим возвратом в исходный город. В условиях задачи указываются критерий выгодности маршрута (кратчайший, самый дешёвый, совокупный критерий и тому подобное) и соответствующие матрицы расстояний, стоимости и тому подобного.

**Требования к функциональности компьютерной программы**

Возможность задания топологии сети с указанием ее размерности и пропускной способностью каналов.

• Настройки работы генетического алгоритма: размер популяции, количество поколений, варианты кроссовера, вероятность мутации и др.

• Указание исходных данных (компьютер-отправитель и компьютер-получатель) и автоматическое заполнение исходных данные топологии сети.

• Два режима работы:

– пошаговый - на экране должны отображаться все представители (хромосомы) одного поколения до и после применения каждого оператора (скрещивания, селекции, редукции и мутации).

– циклический - на экране должны отражаться только агрегированные данные по каждому поколению и итоговый набор хромосом.

• На одной из экранных форм должны быть указаны ФИО и e-mail автора приложения, ссылка на учебный курс и год разработки. Эти данные должны быть продублированы в тексте программы (каждого программного модуля).

• Дополнительно необходимо реализовать возможность динамического изменения исходных данных (матрицы связанности графа) во время пошагового режима работы алгоритма.

**Содержание отчета**

• Название и цель работы.

• Задание, краткое описание предметной области и выбранной задачи.

• Блок-схема с пояснениями реализованного ГА и каждого оператора в отдельности.

• Протоколы проведенных экспериментов (5+), представленные в графиков (допускаются скриншоты в случае программной реализации функциональности).

• Выводы и рекомендации по использованию разработанных ГА.

**Выполнение работы**

1. Настройки работы генетического алгоритма: размер популяции, количество поколений, варианты кроссовера, вероятность мутации и др. Выполнение данного пункта показано на рисунке 1.

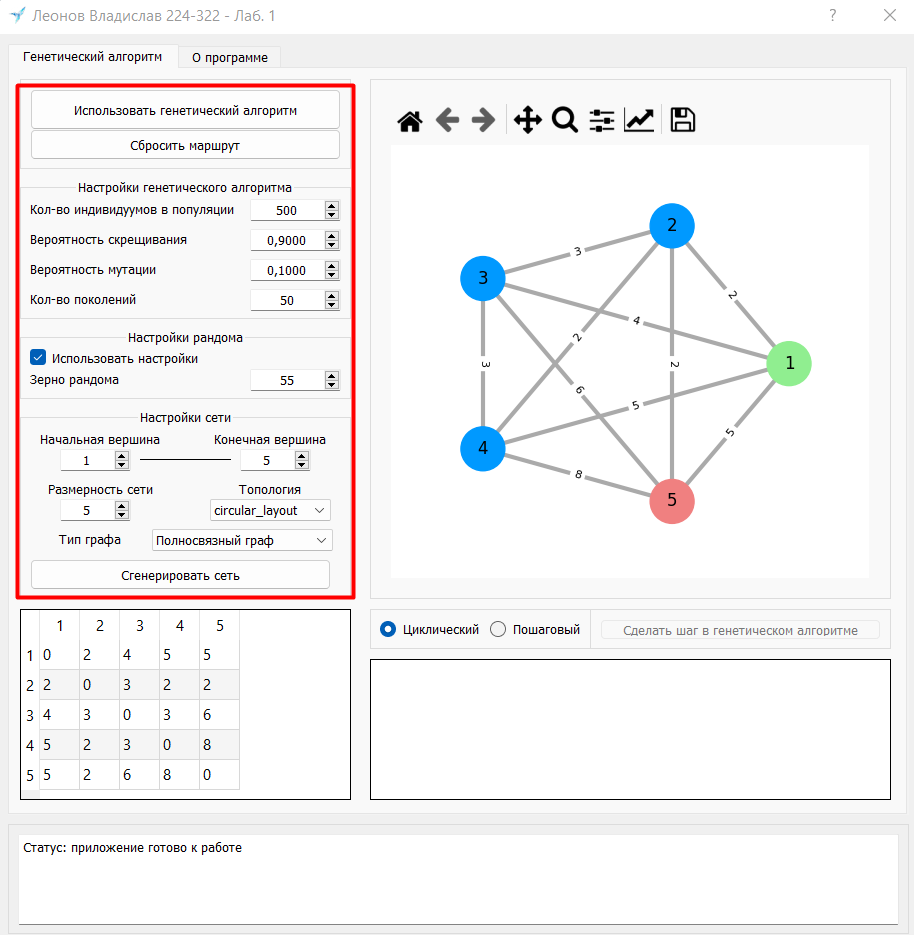


Рисунок 1 – Панель с настройками работы программы

1. Указание исходных данных (компьютер-отправитель и компьютер-получатель) и автоматическое заполнение исходных данные топологии сети. Выполнение данного пункта показано на рисунке 2.

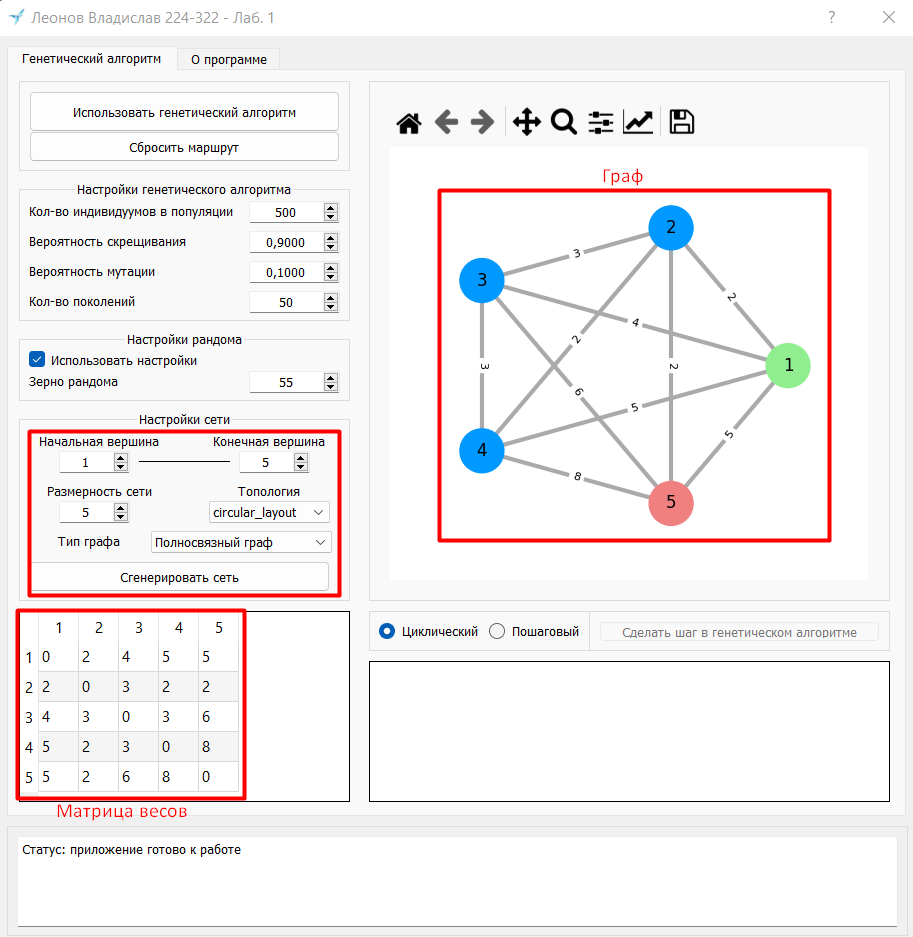


Рисунок 2 – Исходные данные топологии сети

1. Два режима работы:
2. пошаговый - на экране должны отображаться все представители (хромосомы) одного поколения до и после применения каждого оператора (скрещивания, селекции, редукции и мутации).
3. циклический - на экране должны отражаться только агрегированные данные по каждому поколению и итоговый набор хромосом.

Для пошагово режима необходимо включить режим «Пошаговый», после включения которого будет доступна кнопка «Сделать шаг в генетическом алгоритме». Один шаг - это полный цикл работы генетического алгоритма, в таблицу под графом будет выведен текущий результат шага. Активация первого шага происходит по нажатию на кнопку «Использовать генетический алгоритм». Выполнение пошагового режима показано на рисунке 3.

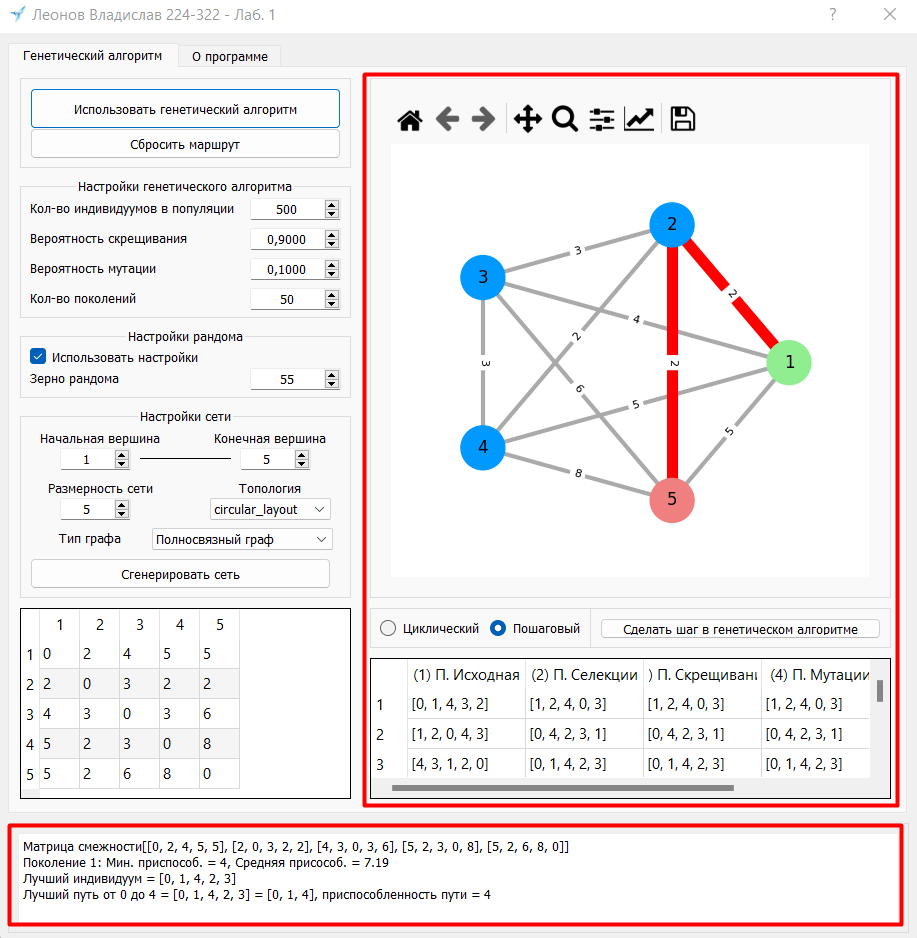


Рисунок 3 – Выполнение пошагового режима

Циклический режим проходится по всем поколениям генетического алгоритма и в конце выводит таблицу итоговой популяции, рисует маршрут на графе, в текстовое поле в самом низу приложения выводит всю информацию об поколениях, а также выводится график зависимости минимальной и средней приспособленности от поколений. Выполнение пошагового режима показано на рисунке 4.

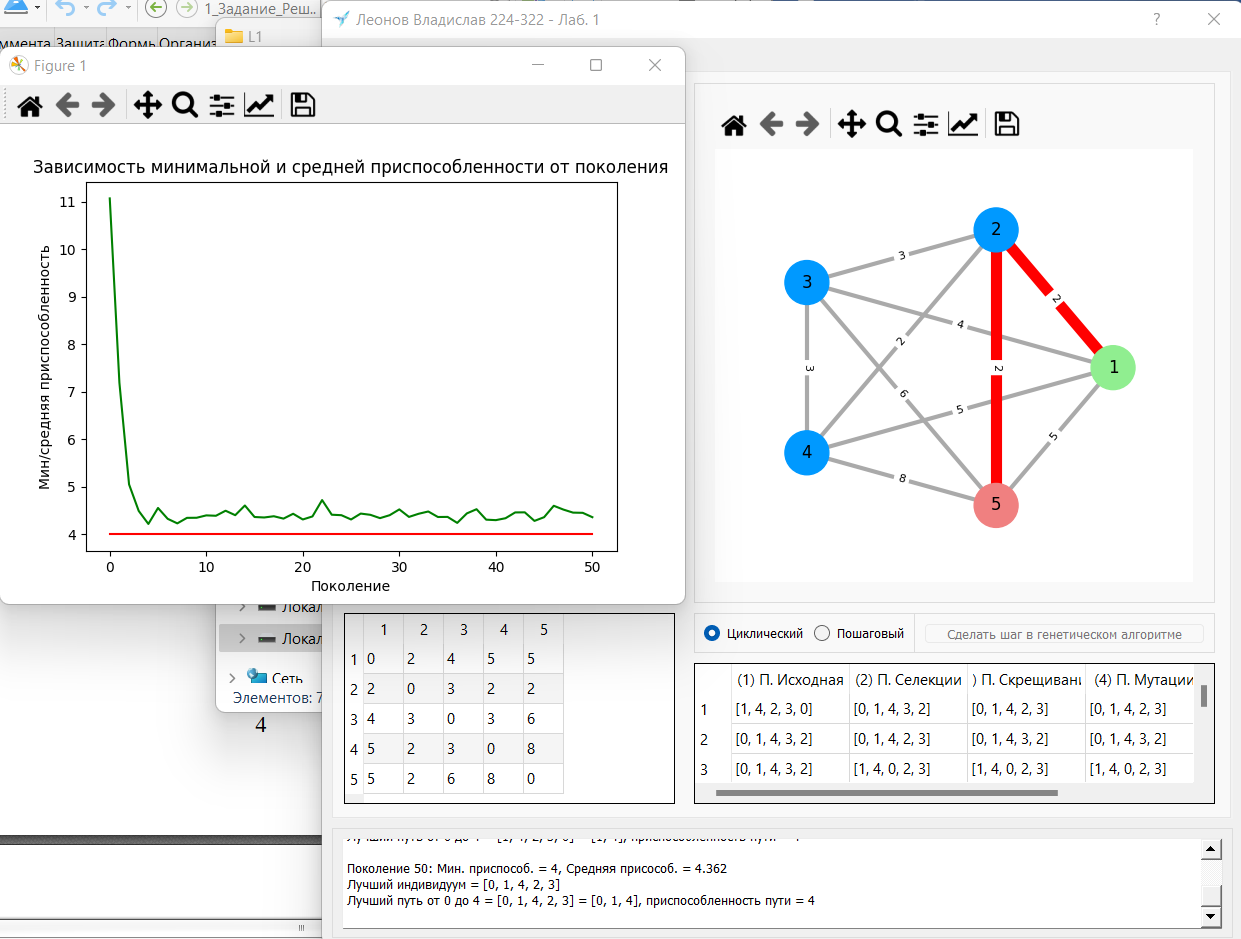


Рисунок 4 – Выполнение циклического режима

1. На одной из экранных форм должны быть указаны ФИО и e-mail автора приложения, ссылка на учебный курс и год разработки. Эти данные должны быть продублированы в тексте программы (каждого программного модуля). Выполнение данного пункта показано на рисунке 5.

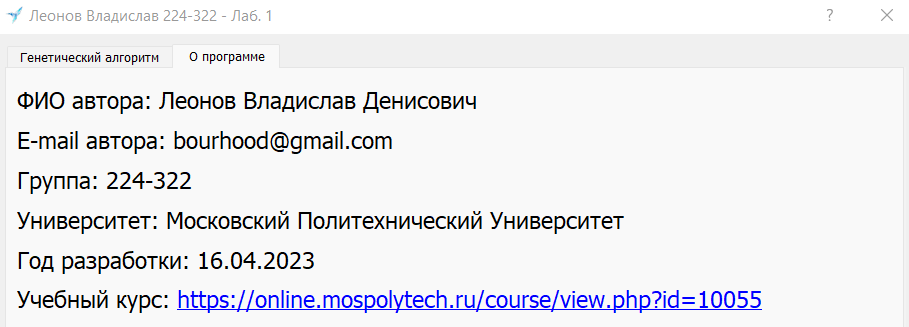


Рисунок 5 – Информация о приложении

1. Дополнительно необходимо реализовать возможность динамического изменения исходных данных (матрицы связанности графа) во время пошагового режима работы алгоритма. Выполнение данного пункта показано на рисунке 6.

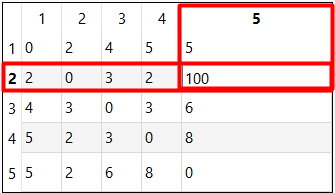
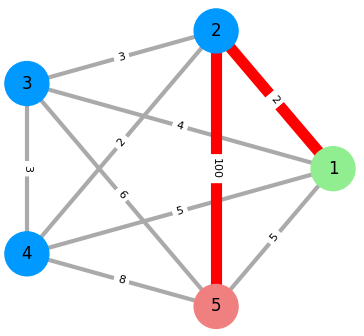
 

Рисунок 6 – Изменение матрицы связанности

1. Блок-схема с пояснениями реализованного ГА и каждого оператора в отдельности.

Блок схема генетического алгоритма представлена на рисунке 7.

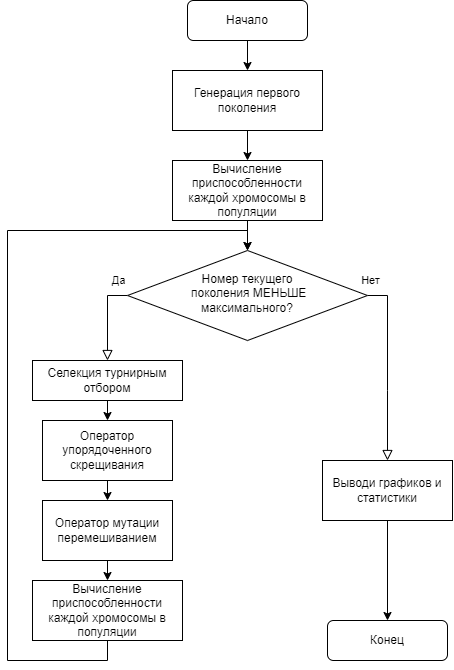


Рисунок 7 – Блок схема генетического алгоритма

Блок схема операции скрещивания представлена на рисунке 8.

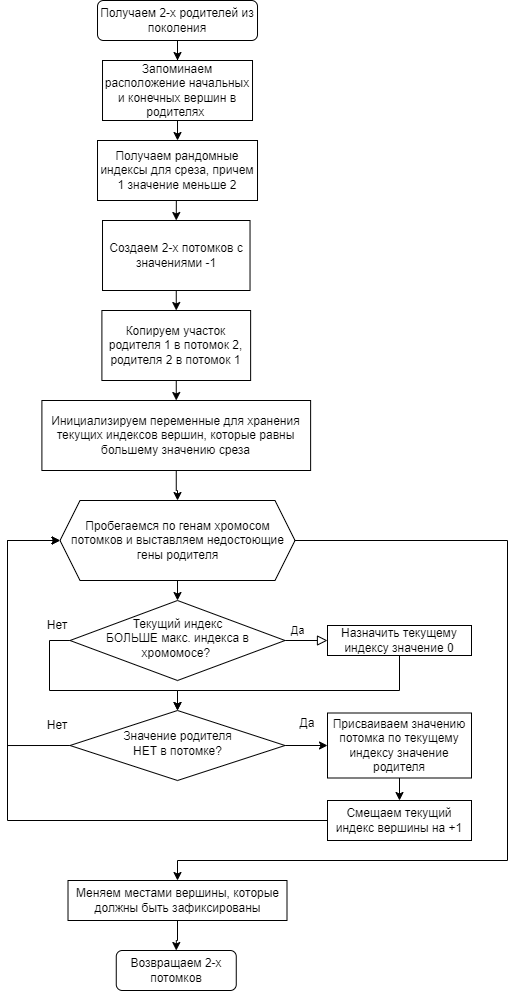


Рисунок 8 – Блок схема генетического алгоритма

Блок схема операции мутации представлена на рисунке 9.

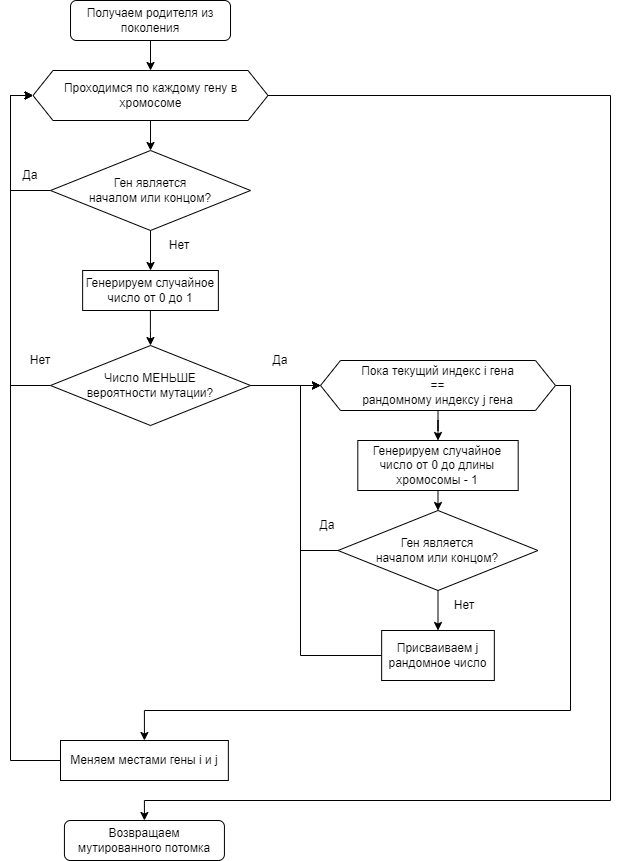


Рисунок 9 – Блок схема генетического алгоритма

1. Протоколы проведенных экспериментов (5+), представленные в графиков (допускаются скриншоты в случае программной реализации функциональности).
2. Эксперимент 1 **(Правильно)**: количество вершин – 5; начальная вершина – 1; конечная вершина – 5; количество индивидуумов в популяции – 500; вероятность скрещивания – 0,9; вероятность мутации – 0,1; количество поколений – 50; тип графа – полносвязный граф. Результат эксперимента 1 приведен на рисунке 10.

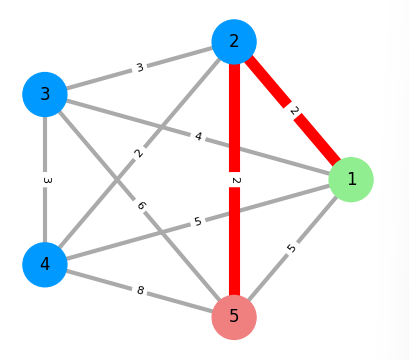


Рисунок 10 – Результат эксперимента 1

1. Эксперимент 2 **(Правильно)**: количество вершин – 7; начальная вершина – 1; конечная вершина – 4; количество индивидуумов в популяции – 500; вероятность скрещивания – 0,9; вероятность мутации – 0,1; количество поколений – 50; тип графа – граничный граф. Результат эксперимента 2 приведен на рисунке 11.

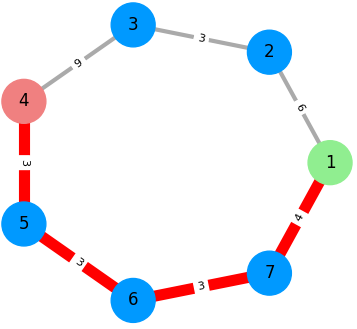


Рисунок 11 – Результат эксперимента 2

1. Эксперимент 3 **(Неправильно)**: количество вершин – 10; начальная вершина – 1; конечная вершина – 2; количество индивидуумов в популяции – 50; вероятность скрещивания – 0,9; вероятность мутации – 0,1; количество поколений – 50; тип графа – рандомный граф. Результат эксперимента 3 приведен на рисунке 12.

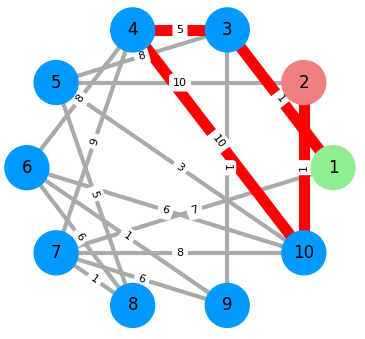


Рисунок 12 – Результат эксперимента 3

1. Эксперимент 4 **(Правильно)**: количество вершин – 10; начальная вершина – 1; конечная вершина – 2; количество индивидуумов в популяции – 100; вероятность скрещивания – 0,9; вероятность мутации – 0,1; количество поколений – 200; тип графа – рандомный граф. Результат эксперимента 4 приведен на рисунке 13.

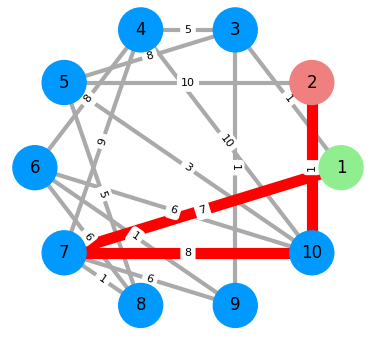


Рисунок 13 – Результат эксперимента 4

1. Эксперимент 5 **(Правильно)**: количество вершин – 8; начальная вершина – 4; конечная вершина – 1; количество индивидуумов в популяции – 100; вероятность скрещивания – 0,9; вероятность мутации – 0,1; количество поколений – 50; тип графа – рандомный граф. Результат эксперимента 5 приведен на рисунке 14.

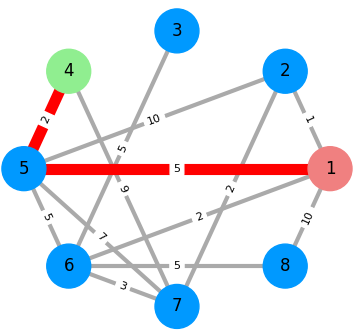


Рисунок 14 – Результат эксперимента 5

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № | Кол-во вершин | Начальная вершина | Конечная вершина | Кол-во особей | Вероятность скрещивания | Вероятность мутации | Поколений |
| 1 | 5 | 1 | 5 | 500 | 0,9 | 0,1 | 50 |
| 2 | 7 | 1 | 4 | 500 | 0,9 | 0,1 | 50 |
| 3 | 10 | 1 | 2 | 50 | 0,9 | 0,1 | 50 |
| 4 | 10 | 1 | 2 | 100 | 0,9 | 0,1 | 200 |
| 5 | 8 | 4 | 1 | 100 | 0,9 | 0,1 | 50 |

**Вывод:**

Генетический алгоритм зависит от исходных данных и настроенных параметров, кол-во индивидуумов влияет на ширину поиска возможных решений, поэтому чем данный параметр больше, тем больше вероятность того, что генетический алгоритм сможет найти верный путь. Вероятность мутации и скрещивания помогают алгоритму находить новые возможные решения, которые отличаются от решений в исходной популяции.