Отчет по Лабораторной работе №4.

Автор: Дулаев Дмитрий

1. Результаты решения тестовых экземпляров задачи коммивояжера

Проблема	Размер	Параметры popsize и gens	Длина маршрута	Количество итераций до сходимости	Оптимальный маршрут
XQF131	131	10, 10000	785.572	9568	564
XQL662	662	10, 10000	15973.472	9955	2513
PKA379	379	10, 10000	5209.109	9921	1332

2. Описание организации операторов инициализации, кроссовера и мутации. Описание введенных параметров.

Ссылка на код:
https://github.com/Timoniche/GeneticAlgorithms/tree/main/HW4/lab3/src/main/java/lab3

A. Инициализация

Просто рандомная перестановка городов

B. Кроссовер

Реализован алгоритм упорядоченного кроссовера (orderCrossover) из описания лабораторной:

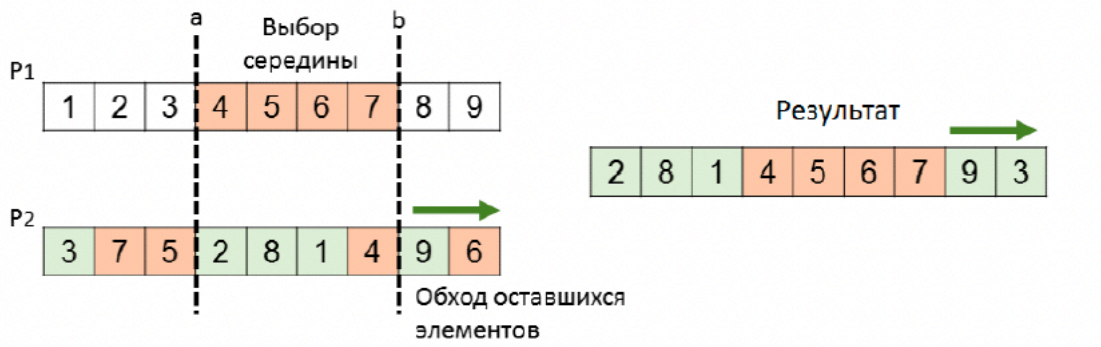

Рисунок 3.5 – Пример реализации упорядоченного кроссовера

C. Мутация

Реализованы SWAP, INVERSION, SCRAMBLE мутации, описанные в самой лабораторной.

Вероятность выбора алгоритма высчитывалась через коэффициент decay (аналогично с прошлой лабораторной работой):

```
double decay = 1 - (generationNumber * 1.0) / GENERATIONS;
if (random.nextDouble() <= 1 - decay * decay) {
    ...INVERSION (exploration)
} else {
    ...SCRAMBLE/SWAP (exploitation)
}
```

Также был введен SCRAMBLE_THRESHOLD = 0.3 для выбора между SCRAMBLE/SWAP exploitation-алгоритмами

D. Selection

RouletteWheelSelection был заменен на TournamentSelection с вероятностью 0.98

3. **Вопросы:**
    1. **Можно ли определить, что полученное решение является глобальным оптимумом?**

Можно брут-форс алгоритмом, что бывает трудно вычислимо. Можно брать какие-то совсем грубые нижние оценки (например, сумму минимальных расстояний между каждой парой городов), чтобы проверить решение-сертификат на адекватность

    2. **Можно ли допускать невалидные решения (с повторением городов). Если да, то как обрабатывать такие решения и как это повлияет на производительность алгоритма?**

Наверное, можно придумать какой-то жадный алгоритм, который повторял бы города. Тогда бы пришлось тратить дополнительное время, чтобы игнорировать/корректировать такие решения, производительность бы упала

    3. **Как изменится задача, если убрать условие необходимости возврата в исходную точку маршрута?**

Интуитивно, кажется, что это усложнит задачу оптимизации: можно начинать и заканчивать в любом узле, что увеличивает пространство поиска. Возможно есть какие-то эвристики для улучшения мутации/кроссовера , в то же время я думаю, что подходы, использованные в этой лабораторной, справятся с этой задачей. Понятно, что как минимум нужно будет изменить фитнесс-функцию (не считать в векторе решения последнее ребро).