Университет ИТМО

Институт прикладных компьютерных наук
Глубокое обучение и генеративный искусственный интеллект

ОТЧЕТ ПО 4-Й ЛАБОРАТОРНОЙ РАБОТЕ
курса
«Эволюционные вычисления»

ГЕНЕТИЧЕСКИЙ АЛГОРИТМ ДЛЯ РЕШЕНИЯ ЗАДАЧИ КОММИВОЯЖЁРА

Студент:
Группа № M4130 *Батурина Ксения Александровна*

Санкт-Петербург 2024

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Код доступен в репозитории на GitHub: https://github.com/xeniabaturina/ITMO_EVOL/tree/main/lab4.

Цель работы:

Получить навыки разработки эволюционных алгоритмов для решения комбинаторных задач на примере задачи коммивояжера.

Задачи работы:

1. Скачать проект.
2. Реализовать алгоритм.
3. Провести эксперименты.
4. Ответить на вопросы:
   - Можно ли определить, что полученное решение является локальным оптимумом?
   - Можно ли допускать невалидные решения (с повторением городов)? Если да, то как обрабатывать такие решения и как это повлияет на производительность алгоритма?
   - Как изменится задача, если убрать условие необходимости возврата в исходную точку маршрута?

1 TRAVELLING SALESMAN PROBLEM

1.1 Описание проекта и тестирования

В рамках данной лабораторной были реализованы классы:

- `TspCrossover`: Кроссовер происходит между двумя родителями (p1 и p2) в случайных точках и создает два потомка, объединяя части родительских маршрутов между точками кроссовера. Затем дети дополняются городами из родительских маршрутов, которые не вошли в выбранный сегмент.
- `TspFactory`: Эта функция генерирует случайные маршруты для начальной популяции. Каждый маршрут представляет собой перестановку номеров городов от 1 до numberOfCities. После генерации маршрут случайным образом перемешивается.
- `TspMutation`: Реализует оператор мутации для задачи TSP. Она случайным образом выбирает сегмент маршрута в каждом решении популяции и изменяет его позицию в маршруте.

Кроме того, реализована функция `getFitness` в классе `TspFitnessFunction`.

1.2 Результаты расчётов

Для тестирования были выбраны задачи:

- $XQF131$
- $XQG237$
- $PMA343$

Ниже приведены график сходимости алгоритма для одного из запусков задачи и таблица с результатами:

Рисунок 1 — Сходимость алгоритма

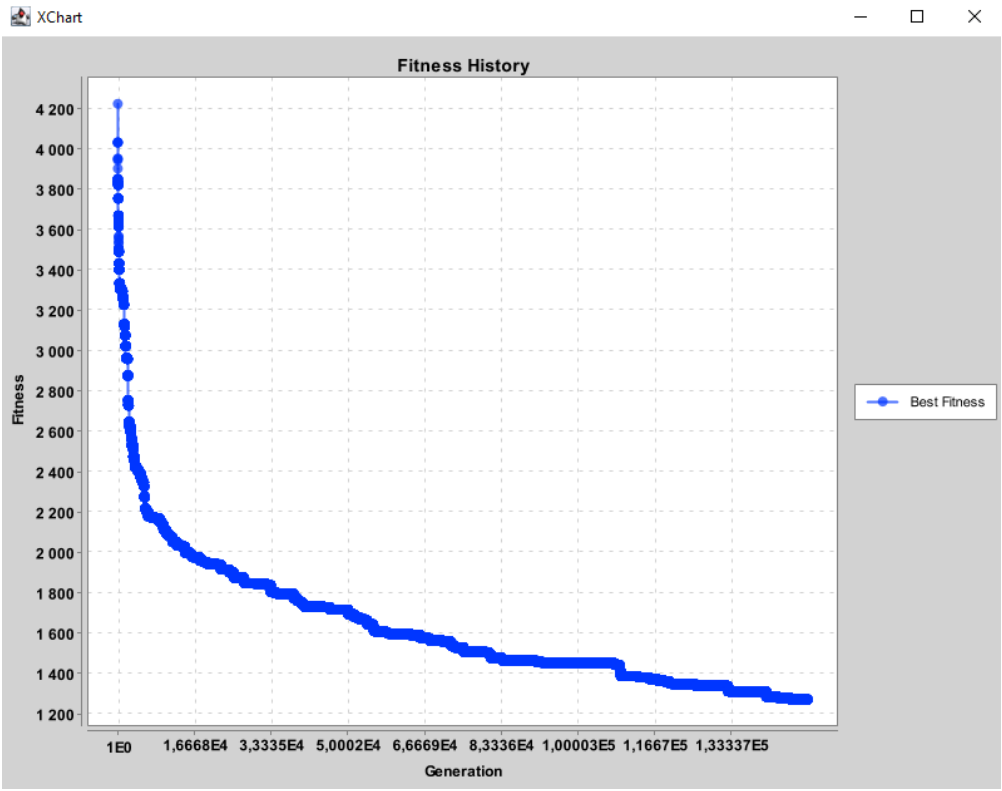

Таблица 1 — Результаты решения тестовых экземпляров задачи коммивояжера.

Проблема	Размер	Popsize и gens	Длина маршр.	Кол-во ит.	Оптим. маршр.
$XQF131$	131	100; 150000	4240	142940,60	1377,86
$XQG237$	237	100; 150000	11400	147147	4296,78
$PMA343$	343	100; 150000	33666,67	117728,4	4926,56

2 ВОПРОСЫ

2.1 Можно ли определить, что полученное решение является локальным оптимумом?

Локальный оптимум — это решение, которое является наилучшим в некоторой окрестности пространства поиска, но не обязательно является глобальным оптимумом. Алгоритм может сходиться к локальному оптимуму, но не обязательно. Для определения локального оптимума можно, например, проверять сходимость алгоритма к одному и тому же решению на нескольких итерациях или анализировать изменение целевой функции на каждой итерации.

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

Можно допускать невалидные решения, если того требует задача, или необходимость допускать такие решения связана с самим алгоритмом. Чтобы обрабатывать их, можно включить проверку на дубликаты городов в пути перед оценкой функции приспособленности. Кроме того, если маршрут содержит повторяющиеся города, его можно отклонить или изменить. Обработка невалидных решений может занимать дополнительное время.

2.3 Как изменится задача, если убрать условие необходимости возврата в исходную точку маршрута?

Если убрать условие возврата в исходную точку маршрута, задача станет открытой (Open TSP). В этом случае коммивояжер может завершить путь в любом городе, не обязательно возвращаясь в начальный город. Производительность может измениться, так как отпадает необходимость возврата в исходную точку, что может упростить поиск оптимального решения и потенциальнго может привести к другому оптимальному маршруту, чем в традиционной задаче TSP.