**Министерство науки и высшего образования Российской Федерации**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«Национальный исследовательский университет ИТМО»**

**(Университет ИТМО)**

**Факультет Институт прикладных компьютерных наук**

**Образовательная программа 01.04.02**

Отчет

по лабораторной работе №1

по дисциплине **«Генетические алгоритмы»**

Вариант 21 – Алгоритм топологической сортировки Кана

Выполнил: **студент группы M4130 Шакиров К. И.**

Проверил: **Муратов С. Ю.**

Санкт-Петербург

2024

# «Сложность алгоритмов и их оптимизация»

## Цель работы.

Получить навыки вычисления сложности алгоритмов и их оптимизации

различными методами.

## Задачи, выполняемые при выполнении работы.

### Реализовать на любом ЯП алгоритм, согласно варианту задания.

### Вычислить сложность алгоритма, привести расчёты, результаты нагрузочных тестов с замером затраченного времени и ресурсов.

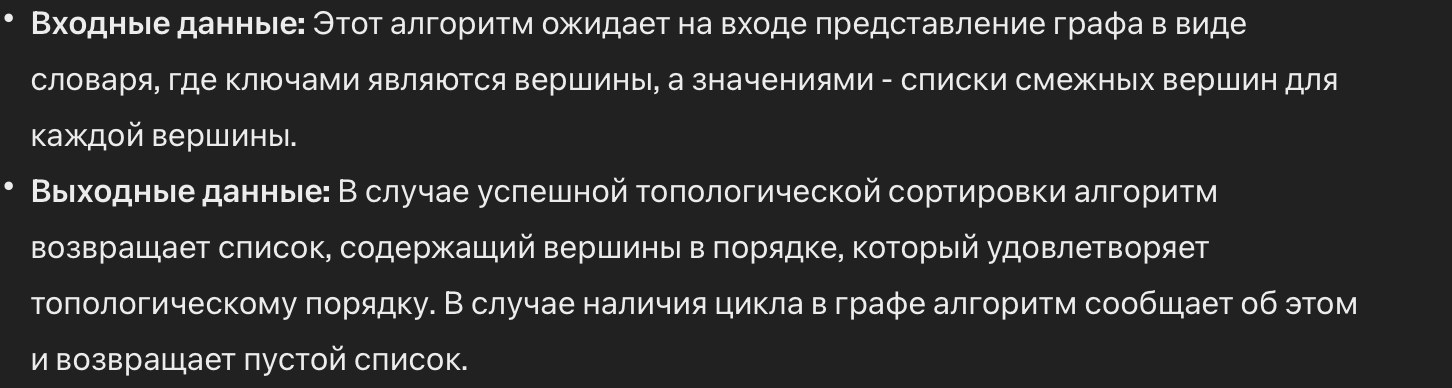
### Выполнить оптимизацию как алгоритмическую если возможно, с выносом инварианта, например, так и программными методами выбранного ЯП.

### Вычислить сложность оптимизированного алгоритма, привести расчёты, результаты нагрузочных тестов с замером затраченного времени и ресурсов.

### Описать различие величин сложности, результатов, привести обоснование.

## Ход работы.

### Был определен формат входных данных, тест производительности (рисунки 1-2).



Данные

Изображение выглядит как текст, Шрифт, снимок экрана

Автоматически созданное описание

Тесты производительности алгоритма

### Была реализована базовая версия алгоритма и проведено тестирование и описана сложность (рисунки 3-5).

Изображение выглядит как текст, снимок экрана, Шрифт

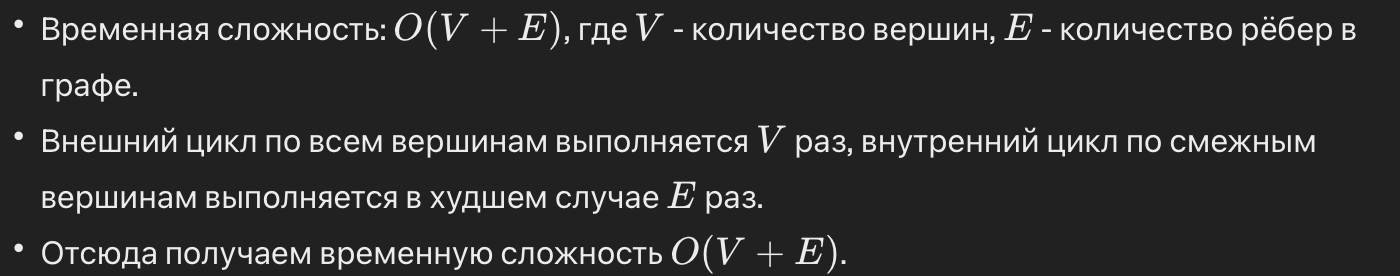
Автоматически созданное описание

Реализация базовой версии алгоритма Кана

Изображение выглядит как текст, снимок экрана, Шрифт, чек

Автоматически созданное описание

Результат тестирования базовой версии



Описание сложности базовой версии

### Была реализована оптимизированная версия алгоритма с использованием с использованием кучи, произведено тестирование и описана сложность (рисунки 6-8).

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Реализация с использованием defaultdict и deque

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Результат тестирования улучшенной версии

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Описание сложности улучшенной версии

### Для наглядности был построен график зависимости времени работы от количества вершин в графе (рисунок 9).

Изображение выглядит как линия, текст, График, Шрифт

Автоматически созданное описание

График зависимости времени выполнения от количества ребер в графе

## Различие величин сложности

### Базовый алгоритм:

### Этот алгоритм использует массив для подсчёта степеней захода для каждой вершины и обычный список для хранения очереди вершин с нулевой степенью захода.

### Улучшенный алгоритм:

### Этот алгоритм также основан на алгоритме Кана, но использует более эффективные структуры данных, такие как `defaultdict` и `deque`, чтобы улучшить производительность и эффективность использования памяти.

## Выводы

Итак, базовый алгоритм и улучшенный алгоритм оба выполняют топологическую сортировку графа методом Кана. Оба алгоритма имеют одинаковую асимптотическую сложность O(V + E), где V - количество вершин, а E - количество рёбер в графе. Однако, улучшенный алгоритм использует более эффективные структуры данных, такие как `defaultdict` и `deque`, что может улучшить производительность за счёт уменьшения констант в алгоритме. В обоих случаях выходные данные представляют собой список вершин в порядке топологической сортировки или пустой список в случае наличия цикла.

Ссылка на репозиторий: https://github.com/PapaJool/ea-lab1