МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОГО ГОСУДАРСТВЕННОГО АВТОНОМНОГО ОБРАЗОВАТЕЛЬНОГО УЧРЕЖДЕНИЯ ВЫСШЕГО ОБРАЗОВАНИЯ

«Национальный исследовательский технологический университет «МИСИС»

Институт Компьютерных Наук

Отчет

Реализация алгоритма Дейкстры при помощи черпаков. Операции с черпаками.

По курсу: Комбинаторика и теория графов

Ссылка на репозиторий:

https://github.com/ov3rvoid/combinatorics.git

Журавлёв Сергей Романович

Группа БИВТ-23-6

Содержание

---

1. Формальная постановка задачи

Задача:
Для заданного ориентированного графа с неотрицательными весами рёбер требуется найти минимальные расстояния от стартовой вершины s до всех других вершин графа.

Входные данные:

- Граф $G = (V, E)$, где:
  - V — множество вершин;
  - E — множество рёбер с весами $w(u, v) \geq 0$ для каждого ребра $(u, v) \in E$
- Стартовая вершина $s \in V$

Выходные данные:
Минимальные расстояния $d(v)$ от стартовой вершины s до всех других вершин $v \in V$

---

2. Теоретическое описание алгоритма и его характеристики

Описание алгоритма Дейкстры с использованием очереди с приоритетом (черпака):
Очередь с приоритетом используется для поддержания списка вершин, упорядоченных по минимальному расстоянию от начальной вершины. Алгоритм повторяет процесс, пока не будут найдены кратчайшие расстояния до всех вершин графа.

Шаги алгоритма:

1. **Инициализация:**
   - Все расстояния $d(v)$ инициализируются как бесконечность, за исключением стартовой вершины s, для которой $d(s) = 0$.
   - Используется очередь с приоритетом (черпак) для извлечения вершины с минимальным расстоянием.
2. **Обработка:**
   - Извлекается вершина с минимальным расстоянием.
   - Для каждой соседней вершины обновляется минимальное расстояние, и она добавляется в очередь с приоритетом.
3. **Завершение:**

- Алгоритм завершает работу, когда все вершины были обработаны.

Характеристики:

- **Временная сложность:** O((|V|+|E|)·log|V|), где:
  - |V| — количество вершин;
  - |E| — количество рёбер.
- **Пространственная сложность:** O(|V|+|E|).

---

3. Сравнительный анализ с аналогичными алгоритмами

Критерий	Алгоритм Дейкстры с очередью с приоритетом (черпак)	Алгоритм Дейкстры с приоритетной очередью
Структура данных	Массив черпаков	Приоритетная очередь (heapq)
Временная сложность	O((|V|+|E|)·log|V|)	V
Пространственная сложность	O(|V|+|E|).	V
Применимость	Для графов с малыми весами рёбер	Универсальная
Скорость на практике	Высокая для графов с ограниченными весами рёбер	Средняя

Вывод:
Метод с использованием очереди с приоритетом (черпак) эффективнее на графах с ограниченными весами рёбер, когда максимальный вес рёбер W значительно меньше количества вершин |V|.

---

4. Перечень инструментов, используемых для реализации

Для реализации алгоритма использовались следующие инструменты:

- **Язык программирования:** Python 3.
- **Редактор кода:** Visual Studio Code.
- **Библиотека:** heapq для реализации очереди с приоритетом.
- **Тестирование:** Pytest для автоматизированного тестирования.

---

5. Описание реализации и процесса тестирования

Реализация алгоритма:
Код реализован в файле dijkstra.py. Основные компоненты:

1. **Класс Graph:**
   - Представляет граф с методами для добавления рёбер и выполнения алгоритма Дейкстры.

2. **Метод dijkstra:**
    - Выполняет алгоритм Дейкстры с использованием очереди с приоритетом (черпака) для вычисления кратчайших расстояний.
3. **Тесты:**
    - В тестах проверяются стандартные случаи (графы с несколькими вершинами и рёбрами), крайние случаи (графы с одной вершиной, без рёбер и пустые графы), а также обработка исключений (например, при добавлении неверных рёбер или выходе за пределы вершин).

Пример входных данных:

6 7
0 1 10
0 2 5
1 2 2
1 3 1
2 3 9
3 4 4
4 5 3

Пример вывода:

Минимальные расстояния: [0, 7, 5, 8, 12, 15]

Процесс тестирования:
Тестирование проводилось с использованием библиотеки pytest. Были проверены следующие сценарии:

1. **Пустой граф:**
    - Ожидаемый результат: d = [0,∞,∞,…]
2. **Граф с одним ребром:**
    - Проверка корректности для минимального графа.
3. **Сложный граф:**
    - Проверка корректности на графах с несколькими рёбрами и путями.
4. **Граф с недопустимыми вершинами:**
    - Проверка обработки ошибок при добавлении рёбер с недопустимыми вершинами.

---

6. Преимущества и ограничения реализации

Преимущества:

1. Эффективность на графах с ограниченными весами рёбер.
2. Простота реализации с использованием стандартной библиотеки Python.

Ограничения:

1. Неэффективность на графах с большими весами рёбер (в таких случаях лучше использовать другие алгоритмы).

2. Требования к памяти, особенно при больших графах.

---

7. Заключение

Реализация алгоритма Дейкстры с использованием очереди с приоритетом (черпак) показала хорошую эффективность для графов с ограниченными весами рёбер. Алгоритм правильно вычисляет кратчайшие расстояния и обрабатывает исключения при неверных данных. Тестирование показало, что реализация работает корректно в разных сценариях. Этот алгоритм особенно полезен для графов с ограниченным диапазоном весов рёбер.

Основные выводы:

1. Алгоритм эффективно работает для графов с ограниченным диапазоном весов рёбер.
2. Для универсальных графов с большими весами предпочтительнее использовать другие структуры данных, например, приоритетную очередь.