Московский авиационный институт
(национальный исследовательский университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

Лабораторная работа №9 по курсу «Дискретный анализ»

Студент: Т. Д. Голубев
Преподаватель: А. А. Кухтичев
Группа: М8О-306Б-22
Дата:
Оценка:
Подпись:

Москва, 2024

Лабораторная работа №9

Задача: Задан взвешенный ориентированный граф, состоящий из n вершин и m ребер. Вершины пронумерованы целыми числами от 1 до n. Необходимо найти величину максимального потока в графе при помощи алгоритма Форда-Фалкерсона. Для достижения приемлемой производительности в алгоритме рекомендуется использовать поиск в ширину, а не в глубину. Истоком является вершина с номером 1, стоком – вершина с номером n. Вес ребра равен его пропускной способности. Граф не содержит петель и кратных ребер.

1 Описание

Требуется реализовать алгоритм Форда-Фалкерсона. Точнее алгоритм Эдмондса-Карпа. Это модификация алгоритма Форда-Фалкерсона.

В [1] приведён алгоритм Форда-Фалкерсона: «Поиск максимального потока методом Форда-Фалкерсона проводится последовательно. Вначале поток нулевой (и величина его равна нулю). На каждом шаге мы увеличиваем значение потока. Для этого мы находим "дополняющий путь по которому можно пропустить ещё немного вещества, и используем его для увеличения потока. Этот шаг повторяется, пока есть дополняющие пути.».

Алгоритм Эдмондса-Карпа заключается в том, чтобы искать кратчайший «дополняющий путь» с помощью обхода в ширину (BFS).

Сложность такого алгоритма – $O(V \cdot E^2)$, где V – количество вершин в графе, E – количество рёбер.

2 Исходный код

main.cpp	
std::vector<TVertex> FindPath(const TGraph& graph, TVertex from, TVertex to)	Поиск пути с помощью BFS.

```
1 using TVertex = uint16_t;
2 using TWeight = uint64_t;
3 using TGraph = std::vector<std::unordered_map<TVertex, TWeight>>;
```

3 Консоль

```
cat-mood@nuclear-box:~/programming/mai-da-labs/lab09/build$ ./lab09_exe
10 20
1 2 15
1 3 10
2 4 10
2 5 10
3 6 20
4 7 10
4 8 10
5 8 15
6 8 25
6 9 10
7 10 5
8 10 20
8 9 15
9 10 10
3 7 5
4 6 5
5 7 10
1 6 20
2 8 15
3 9 5
35
```

4 Тест производительности

Тест производительности представляет из себя следующее: Решение с использованием BFS сравнивается с решением, использующим DFS. В первом тесте 10 вершин и 20 рёбер. Во втором – 100 вершин и 1000 рёбер.

```
cat-mood@nuclear-box:~/programming/mai-da-labs/lab09$ ./a.out <./build/test.txt
BFS: 69 ms
DFS: 97 ms

cat-mood@nuclear-box:~/programming/mai-da-labs/lab09$ ./a.out <./build/large_test.txt
BFS: 7683 ms
DFS: 16423 ms
```

Как видно, решение с использованием BFS работает быстрее.

5 Выводы

Выполнив девятую лабораторную работу по курсу «Дискретный анализ», я реализовал алгоритм Форда-Фалкерсона. В ходе работы не сталкивался с проблемами. Решил задачу сходу, руководствуюсь Корменом и лекциями Никиты Константиновича.

Список литературы

[1] Томас Х. Кормен, Чарльз И. Лейзерсон, Рональд Л. Ривест, Клиффорд Штайн Алгоритмы: построение и анализ. - 3-е изд. - М.: ООО "И. Д. Вильямс 2013. - 1328 с.