

Міністерство освіти і науки України
Львівський національний університет імені Івана Франка
Факультет прикладної математики та інформатики

Кафедра дискретного аналізу та інтелектуальних систем

Лабораторна робота №10
ГРАФИ, АЛГОРИТМ ДЕЙКСТРИ
з курсу “Алгоритми та структури даних”

Виконав:
студент групи ПМІ-13
Лук'янчук Денис
Євгенійович

Львів – 2024

Алгоритм Дейкстри — алгоритм на графах, відкритий Дейкстрою. Знаходить найкоротший шлях від однієї вершини графу до всіх інших вершин. Класичний алгоритм Дейкстри працює тільки для графів без ребер від'ємної довжини.

Алгоритм виконання:

1. Створюється клас `MyGraph` з приватними полями `matrix` (матриця суміжності графу) та `num` (кількість вершин у графі).
2. У конструкторі `MyGraph` виділяється пам'ять для матриці розміром `num` на `num` та заповнюється нулями.
3. Метод `add` додає ребро між вершинами `i` та `j` з вагою `k` у матрицю суміжності.
4. Метод `algorithm` виконує BFS для знаходження найкоротших відстаней від вершини 1 до всіх інших вершин.
5. У методі `algorithm` створюються та ініціалізуються масиви `vis` (відвідані вершини) та `dis` (відстані до вершин).
6. Початкова вершина позначається як відвідана, її відстань встановлюється як 0 та додається до черги.
7. Запускається основний цикл BFS, доки черга не стане порожньою.
8. В кожній ітерації циклу вибирається вершина з початка черги (`vetrex`).
9. Перевіряється кожен сусід цієї вершини. Якщо сусід не відвіданий і існує ребро між поточною вершиною та сусідом, та якщо додавання ваги ребра до відстані від поточної вершини до сусіда зменшує відстань до сусіда, тоді відстань до сусіда оновлюється, і сусід додається до черги.
10. Після закінчення BFS виводяться знайдені найкоротші відстані від вершини 1 до всіх інших вершин.
11. У `main` функції створюється об'єкт класу `MyGraph` з 7 вершинами та додаються ребра між вершинами з вагами.
12. Викликається метод `print`, який виводить матрицю суміжності графу.
13. Викликається метод `algorithm`, який знаходить найкоротші відстані.

Приклад

Дано: кількість вершин = 7; вага ребр = {5, 8, 4, 10, 15, 13, 9, 2}

```
Graph
1 : 0 5 8 0 0 4
2 : 5 0 10 15 0 0
3 : 8 10 0 13 0 9
4 : 0 15 13 0 2 0
5 : 0 0 0 2 0 0
6 : 4 0 9 0 0 0
```

```
Shortest distances
Vertex 1: 0
Vertex 2: 5
Vertex 3: 8
Vertex 4: 20
Vertex 5: 22
Vertex 6: 4
```

Приклад Unit-тесту(без помилок)

```
[=====] Running 4 tests from 1 test suite.
[-----] Global test environment set-up.
[-----] 4 tests from HashTableTest
[ RUN     ] HashTableTest.InsertTest
[ OK      ] HashTableTest.InsertTest (0 ms)
[ RUN     ] HashTableTest.SearchTest
[ OK      ] HashTableTest.SearchTest (0 ms)
[ RUN     ] HashTableTest.RemoveTest
[ OK      ] HashTableTest.RemoveTest (0 ms)
[ RUN     ] HashTableTest.ContainsTest
[ OK      ] HashTableTest.ContainsTest (0 ms)
[-----] 4 tests from HashTableTest (2 ms total)

[-----] Global test environment tear-down
[=====] 4 tests from 1 test suite ran. (4 ms total)
[ PASSED ] 4 tests.
```

Приклад Unit-тесту(з помилкою)

```
Which is: 3
4
[ FAILED ] DijkstraTest.ShortestPaths (1 ms)
[ RUN      ] DijkstraTest.UnreachableVertex
C:\Users\Евгений\Documents\Test_10\Test_10\Test_10.cpp(67): error: Expected equality of
distance[2]
Which is: 3
numeric_limits<int>::max()
Which is: 2147483647

[ FAILED ] DijkstraTest.UnreachableVertex (0 ms)
[ RUN      ] DijkstraTest.NegativeWeights
C:\Users\Евгений\Documents\Test_10\Test_10\Test_10.cpp(83): error: Expected equality of
distance[2]
Which is: -1
1

[ FAILED ] DijkstraTest.NegativeWeights (1 ms)
[-----] 3 tests from DijkstraTest (3 ms total)

[-----] Global test environment tear-down
[=====] 3 tests from 1 test suite ran. (5 ms total)
[ PASSED ] 0 tests.
[ FAILED ] 3 tests, listed below:
[ FAILED ] DijkstraTest.ShortestPaths
[ FAILED ] DijkstraTest.UnreachableVertex
[ FAILED ] DijkstraTest.NegativeWeights

3 FAILED TESTS
```

Висновок: Код реалізує алгоритм пошуку найкоротших відстаней у зваженому напрямленому графі за допомогою BFS. Він обчислює відстані від заданої вершини до всіх інших вершин та виводить результати. Граф заданий у вигляді матриці суміжності, де ребра між вершинами мають вагу. Використання черги допомагає ефективно обробляти вершини графа.