МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

КРЕМЕНЧУЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ   
ІМЕНІ МИХАЙЛА ОСТРОГРАДСЬКОГО

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ЕЛЕКТРИЧНОЇ ІНЖЕНЕРІЇ   
ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА АВТОМАТИЗАЦІЇ ТА ІНФОРМАЦІЙНИХ СИСТЕМ

НАВЧАЛЬНА ДИСЦИПЛІНА  
 «Алгоритми та методи обчислень»

ЗВІТ

З ПРАКТИЧНИХ РОБІТ

Виконав:

студент групи КН-23-1

Ярковий Т.С.

Кременчук 2023

**Практична робота № 6**

**Тема. Графи. Найкоротші шляхи**

**Мета:** набути практичних навичок розв’язання задач пошуку найкоротших шляхів у графі та оцінювання їх асимптотичної складності.

**Завдання**

1. Дано мапу міст України з такими відстанями між деякими містами: map\_ukraine =

{

'Київ': {'Харків': 472, 'Одеса': 478, 'Львів': 542},

'Харків': {'Київ': 472, 'Одеса': 789, 'Львів': 1056},

'Одеса': {'Київ': 478, 'Харків': 789, 'Львів': 789},

'Львів': {'Київ': 542, 'Харків': 1056, 'Одеса': 789}

}

Знайти найкоротший маршрут для подорожі між містами Харків та Одеса за допомогою алгоритму Дейкстри.

Масив найкоротших шляхів з попередньою вершиною:

('Харків', 0, 'Харків'), ('Київ', ∞, 0), ('Львів', ∞, 0), ('Одеса', ∞, 0).

Обираємо вершину з найменшою відстанню і обраховуємо відстані до сусідніх вузлів: ('Харків', 0, 'Харків'),

Порівнюємо довжину шляху обраної вершини + вага відповідного ребра з вагою шляху вершини порівняння.

Київ: min(0+472, ∞)= 472.

Львів: min(0+1056, ∞)= 1056.

Одеса: min(0+789, ∞)= 789.

Розглядаємо інші, ще не розглянуті вершини:

('Київ', 472, 'Харків'), ('Львів', 1056, 'Харків'), ('Одеса', 789, 'Харків')

Аналогічно обираємо вершину з найменшою відстанню і обраховуємо відстані до сусідніх вузлів: ('Київ', 472, 'Харків'),

Львів: min(472+542, 1056)= 1014.

Одеса: min(472+478, 789)= 789.

Розглядаємо інші, ще не розглянуті вершини:

('Львів', 1056, 'Київ'), ('Одеса', 789, 'Харків')

Далі маємо обрати Одесу, але це точка прибуття і вона має розглядатися останньою, тому розглядаємо ('Львів', 1056, 'Київ'):

Одеса: min(1056+789, 789)= 789.

Усі вершини розглянуті, отримано масив найкоротших шляхів:

('Харків', 0, 'Харків'), ('Київ', 472, 'Харків'), ('Львів', 1056, 'Київ'), ('Одеса', 789, 'Харків').

Відповідь: найкоротший шлях маршруту для подорожі це Харків-Одеса, шлях має відстань 789.

**Контрольні питання**

1. Що таке граф і які головні складові його структури?

Граф – це математична структура, що складається з множини вершин та множини ребер. Основані складові це 𝑉 – множина вершин графа, 𝐸 – множина ребер, що їх з’єднують. Кожне ребро являє собою кортеж (𝑣, 𝑤), де 𝑤, 𝑣 ∈ 𝑉. Сюди можна додавати третій компонент, що відображає вагу ребра.

1. Які алгоритми використовуються для пошуку найкоротших шляхів у графах?

Алгоритм Дейкстри, алгоритм Белмена-Форда, алгоритм Форйла-Форшала.

1. Як працює алгоритм Дейкстри і які його особливості?

Принцип роботи:

1. Ініціалізуємо масив Найкоротший[ ] так, що Найкоротший[s] = 0, а Найкоротший[v] = нескінченність для всіх інших вершин v.

2. Ініціалізуємо чергу з пріоритетами Q, додаючи всі вершини графа разом з відстанню до них у Найкоротший[].

3. Доки черга Q не порожня: Вилучаємо вершину u з найменшою відстанню з Q. Для кожного сусіднього вузла v з вершини u:

– Викликаємо процедуру Релаксація(u, v), яка:

– Якщо Найкоротший[u] + вага(u, v) < Найкоротший[v]:

– Зменшуємо ключ Q для вершини v на нову відстань (Найкоротший[u] + вага(u, v)).

4. По завершенні алгоритму масив Найкоротший[] містить найкоротші відстані від вершини 𝑠 до всіх інших вершин графа.

Особливості: граф не повинен містити циклів з від’ємною вагою, ваги ребер можуть набувати довільних значень (включаючи від’ємні), має кращу асимптотичну складність у найкращому випадку, особливо коли граф є розрідженим.

1. Що таке алгоритм Белмена–Форда і коли його варто застосовувати?

Алгоритм Белмена–Форда – це також алгоритм пошука мінімального шляху між вершинами на графі. Алгоритм Белмана–Форда є універсальним і може працювати з графами з від’ємними вагами ребер і знаходити від’ємні цикли, але він має гіршу асимптотичну складність порівняно з алгоритмом Дейкстри у найкращому випадку.

1. Як працює алгоритм Флойда–Форшала і які його переваги та недоліки?

Як працює:

•На першому кроці ініціалізуйте матрицю розв'язків так

само, як і матрицю вхідного графа.

•Потім оновити матрицю розв'язків, розглядаючи всі

вершини як проміжні.

•Ідея полягає в тому, щоб вибрати всі вершини по черзі і

оновити всі найкоротші шляхи, які включають вибрану

вершину як проміжну вершину в найкоротшому шляху.

•Коли ми вибираємо вершину з номером k як проміжну, ми

вже розглянули вершини {0, 1, 2, .. k-1} як проміжні

вершини.

•Для кожної пари (i, j) вихідної та кінцевої вершин можливі

два випадки.

• k не є проміжною вершиною на найкоротшому

шляху від i до j. Ми залишаємо значення dist[i][j ]

без змін.

• k є проміжною вершиною на найкоротшому шляху

від i до j. Оновлюємо значення dist[i][j ] як dist[i][ k]

+ dist[k][j ], якщо dist[i][j] > dist[i][k] + dist[k][j].

Переваги: визначає найкоротші шляхи між усіма парами вершин у графі, ефективно працює для графів будь-якої ваги, включаючи графи з від'ємними вагами на ребрах.

Недоліки: має кубічну асимптотичну складність, цей алгоритм ефективний лише для невеликих графів, оскільки в інших випадках його складність може бути неприйнятною.