Київський національний університет імені Тараса Шевченка

Факультет комп`ютерних наук та кібернетики

Алгоритми та складність

Завдання №9

“Алгоритм Джонсона для розріджених графів”

Варіант 15

Виконав студент 2-го курсу

Групи К-29

Пащенко Дмитро Вікторович

2020

**Завдання**:

Реалізувати алгоритм Джонсона для розріджених графів, застосувавши його до моделі, описаної у Вашому варіанті. Алгоритм Джонсона включає алгоритми Беллмана-Форда та Дейкстри. Останній, в свою чергу, реалізувати з використанням піраміди Фібоначчі.

**Варіант 15:**

Предметна область: Залізниця

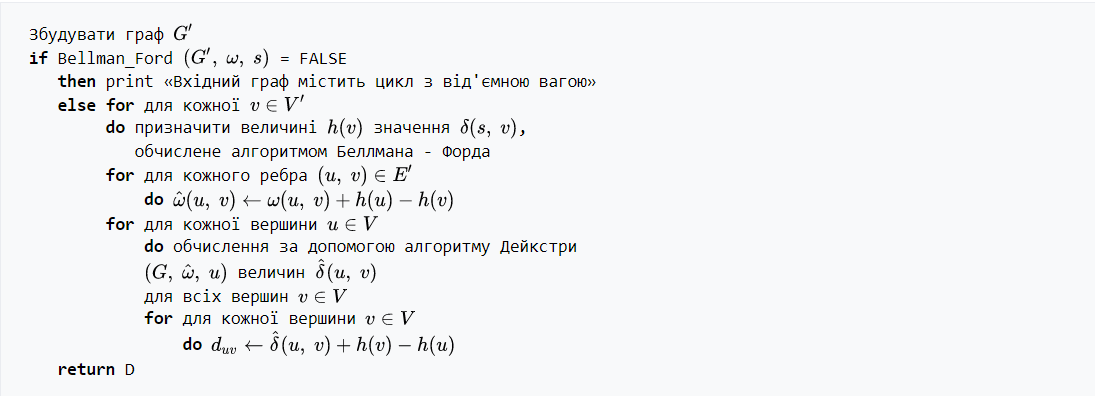
Об'єкти: Дороги, Станції

Примітка: Є безліч залізничних доріг. У відомстві кожної дороги знаходиться безліч станцій.

**Теорія:**

* **Постановка задачі пошуку найкоротшого шляху.** Дано орієнтований граф з ваговою функцією . Вагою шляху назвемо суму ваг ребер, що входять в цей шлях: . Вхідними даними для алгоритму є граф , вагова функція та стартова вершина . Потрібно знайти найкоротші шляхи від вершини до всіх вершин графа.
* **Алгоритм Беллмана-Форда** — алгоритм пошуку найкоротшого шляху в зваженому графі. Знаходить найкоротші шляхи від однієї вершини графа до всіх інших. На відміну від алгоритму Дейкстри, алгоритм Беллмана—Форда допускає ребра з негативною вагою.
* **Алгоритм Дейкстри** знаходить найкоротший шлях від однієї вершини графа до всіх інших вершин. Класичний алгоритм Дейкстри працює тільки для графів без циклів від'ємної довжини.
* **Алгоритм Джонсона** дозволяє знайти найкоротші шляхи між усіма парами вершин зваженого орієнтованого графа. Цей алгоритм працює, якщо у графі містяться ребра з додатною чи від'ємною вагою, але відсутні цикли з від'ємною вагою.

**Алгоритм**

В алгоритмі Джонсона використовують алгоритм Беллмана-Форда та алгоритм Дейкстри, втілені у вигляді підпрограм. Ребра зберігають у вигляді переліків суміжних вершин. Алгоритм повертає звичайну матрицю розміром , де , або видає повідомлення про те, що вхідний граф містить цикл із від'ємною вагою.

**Складність алгоритму**

Якщо в алгоритмі Дейкстри неспадну чергу з пріоритетами втілено у вигляді піраміди Фібоначчі, то тривалість роботи алгоритму Джонсона дорівнює .

**Мова програмування:** С++.

**Інтерфейс користувача**

Введення даних відбувається через консоль.

Вхідні дані: команди

* 0 – завершити програму
* 1 – додати вершину (тут: станцію, за назвою)
* 2 – додати ребро (тут: довжина залізної дороги)
* 3 – вивести граф (у вигляді списків суміжності)
* 4 – вивести матриці оптимальних ваг та шляхів (алгоритм Джонсона)

**Модулі програми:**

* **void Graph<T>::initSignleSource(size\_t main\_vertex)**

Скидає всіх батьків та ваги вершин, обнуляє вагу початкової вершини.

**Складність:** .

* **bool Graph<T>::relax(size\_t from, size\_t to, double weight)**

Ослаблює вершину. Повертає істинне значення, якщо ослаблення відбулося.

**Складність:** .

* **void Graph<T>::addVertex(T const &data)**

Додає вершину.

**Складність:** .

* **void Graph<T>::addEdge(size\_t from, size\_t to, double weight)**

Додає ребро.

**Складність:** .

* **bool Graph<T>::bellmanFordAlgorithm(size\_t init\_vertex)**

Реалізує алгоритм Беллмана-Форда для вказаної вершини.

**Складність:** .

* **void Graph<T>::dijkstraAlgorithm(size\_t init\_vertex)**

Реалізує алгоритм Дейкстри для вказаної вершини (на основі піраміди Фібоначчі).

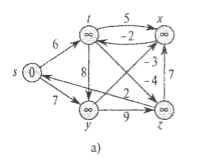
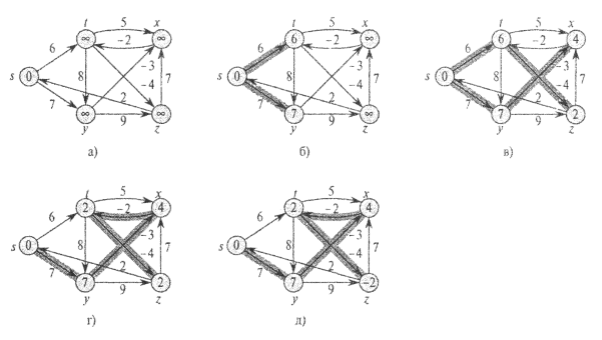
**Складність:** .

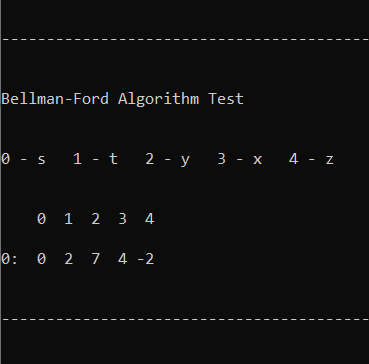
* **std::pair<vector<vector<double>>,vector<vector<size\_t>>> Graph<T>::johnsonAlgorithm()**

Реалізує алгоритм Джонсона. Виводить матрицю мінімальних ваг вершин та матрицю оптимальних шляхів.

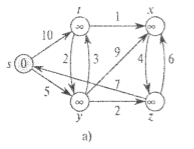
**Складність:** .

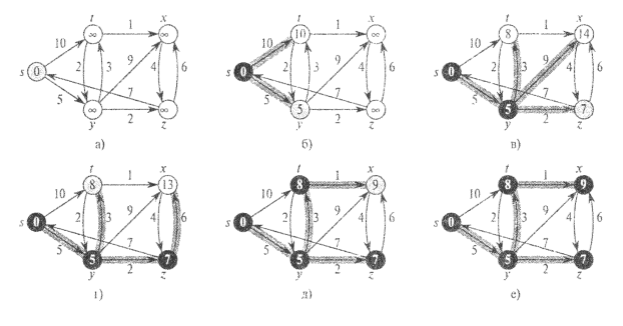
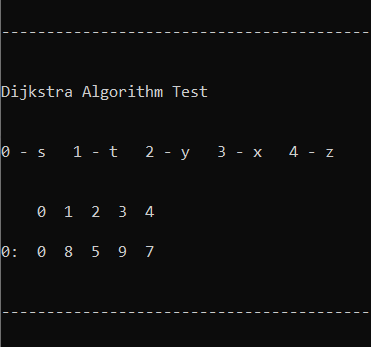
**Тестові приклади**

1. Спочатку протестував алгоритм Беллмана-Форда. Побудував граф на рисунку знизу. Вершинами є станції, а ваги ребер – відстані між станціями. Шукав мінімальну відстань від станції «s» до інших станцій.
2. Запустив алгоритм Беллмана-Форда. Його кроки зображені на рисунку знизу (послідовне ослаблення всіх ребер графу (V-1)-разів, спочатку ослаблюються ребра, інцидентні вершині з меншим номером). Програма вивела вагу мінімального шляху до кожної вершини (перераховані в рядку зверху) від вершини «s».



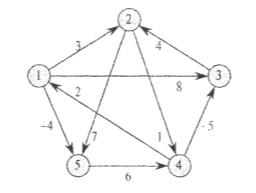
Переконався, що практичні результати співпали з теоретичними.

1. Далі перевірив алгоритм Дейкстри. Він використовує піраміду Фібоначчі, у правильності реалізації якої переконалися в лабораторній роботі №8. Побудував граф на рисунку знизу. Вершини залишилися ті самі, змінилися лише ребра та їх ваги.
2. Запустив алгоритм Дейкстри. Його кроки зображені на рисунку знизу (послідовно вилучаємо з піраміди вершини з найменшою вагою та ослаблюємо всі інцидентні їй ребра, допоки піраміда не стане порожньою). Вивід програми не змінюється.

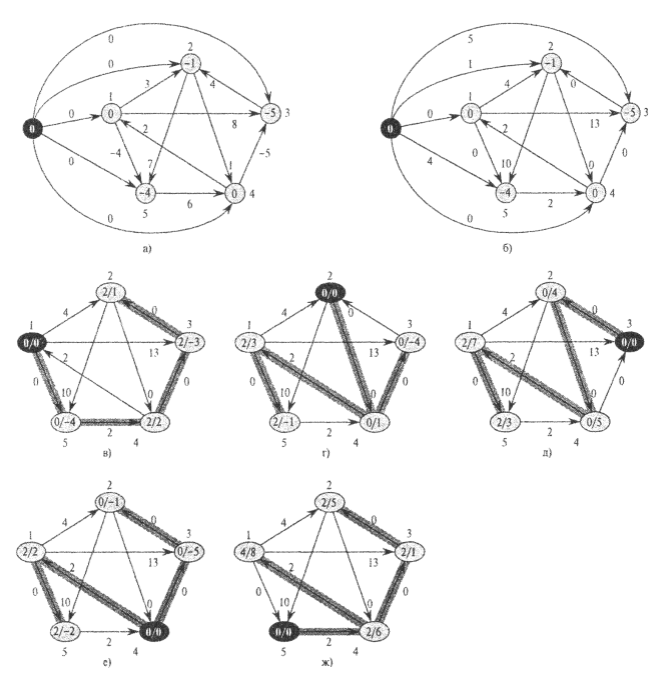


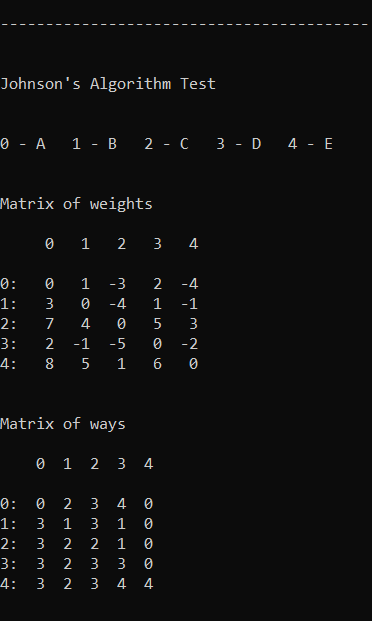
Переконався, що практичні результати співпали з теоретичними.

1. Нарешті перевірив алгоритм Джонсона. Побудував граф на рисунку знизу. Задля зручності імена станцій вказуватися не будуть. Шукатимемо найкоротші шляхи між усіма вершинами.



1. Запустив алгоритм Джонсона. Його кроки зображені на рисунку знизу (спочатку створюється розширений граф з новою вершиною, що сполучена з усіма іншими вершинами, причому ребра мають нульову вагу; потім з нової вершини запускається алгоритм Беллмана-Форда, який, окрім пошуку шляхів, перевіряє наявність циклів з від’ємною вагою, - таких циклів не виявлено; далі значення ваг ребер запам’ятовуються та замінюються на інші – тепер від’ємних ваг немає; для кожної вершини запускається алгоритм Дейкстри, після нього значення мінімальних ваг вершин перераховуються та заносяться у відповідну матрицю, також заповнюється рядок матриці шляхів). Алгоритм повертає матриці ваг та шляхів.





Переконався, що результати співпали (вершини запрограмованого графа нумеруються з нуля). На рисунку варто звернути увагу саме на праві значення в кожній вершині - вони остаточні, зліва ж проміжні. Користуватися таблицею шляхів просто – обирається рядок, що відповідає стартовій вершині, та стовпчик, що відповідає кінцевій вершині. На їх перетині вказаний стовпчик (вершина), куди варто перейти. Такі переходи здійснюються, доки стовпчик не почне переходити сам у себе (це станеться, коли кінцева вершина досягнута). У таблиці ваг обирається рядок, що відповідає початковій вершині, та стовпчик, що відповідає кінцевій вершині. На їх перетині знаходиться шукана вага мінімального шляху.

**Висновки**

Успішно реалізував алгоритм Джонсона. Для алгоритму Дейкстри використав піраміду Фібоначчі – як результат, алгоритм пришвидшився. Алгоритм Джонсона чудово показав себе як алгоритм пошуку мінімальних шляхів у розрідженому графі.

**Література**

* Кормен, Лейзерсон, Рівест, Штайн. Алгоритми: побудова і аналіз, 2-е видання. – 2005
* <https://uk.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC_%D0%91%D0%B5%D0%BB%D0%BB%D0%BC%D0%B0%D0%BD%D0%B0%E2%80%94%D0%A4%D0%BE%D1%80%D0%B4%D0%B0>
* <https://uk.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC_%D0%94%D0%B5%D0%B9%D0%BA%D1%81%D1%82%D1%80%D0%B8>
* <https://uk.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC_%D0%94%D0%B6%D0%BE%D0%BD%D1%81%D0%BE%D0%BD%D0%B0>