2. *Коцовський В. М. конспект лекцій «Дискретна математика та теорія алгоритмів. Частина 1» - Ужгород , 2016.*
3. *Давидов, В.Г. Програмування та основи алгоритмізації [Текст]/В.Г. Давидов - М.: Вища школа, 2003*
4. *Бондарчук Ю. В., Олійник Б. В. посібник «Основи дискретної математики» (для студентів-інформатиків) – Київ, 2007*
5. *Алгоритм рекурсивний: опис, аналіз, особливості та приклади URL: https://presa.com.ua/navchannia/algoritm-rekursivnij-opis-analiz-osoblivosti-ta-prikladi*

УДК 519.1

Зимич А. П., студент 1 курсу
спеціальності 122 «Комп'ютерні науки»
Ніколюк П. К., д-р фіз.-мат. наук,
Професор кафедри комп'ютерних наук

АЛГОРИТМ ФОРДА-ФАЛКЕРСОНА. ЗНАХОДЖЕННЯ МАКСИМАЛЬНИХ ПОТОКІВ В ГРАФАХ

Донецький національний університет імені Василя Стуса, м. Вінниця

Поняття потоку та розрізу

Коли ми говоримо про потоки та розрізи у графах, ми зазвичай маємо на увазі орієнтовані графи, де кожне ребро має напрямок.

Потік в орієнтованому графі – це функція, яка надає кожному ребру графа невід'ємне число, яке ми називатимемо "потоком". При цьому необхідно задовольняти наступні дві умови:

- Пропускна здатність: потік через кожне ребро не може перевищувати його пропускну здатність.
- Закон збереження потоку: кількість потоку, що втікає у вершину, має дорівнювати кількості потоку, що випливає з цієї ж вершини, за винятком джерела та стоку.

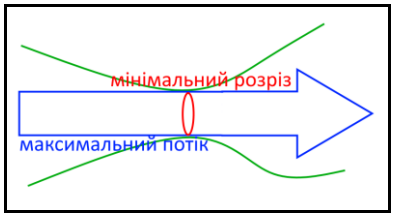
Джерелом ми називаємо вершину графа, з якої починається потік. Стоком ми називаємо вершину, в яку потік повинен потрапляти.

Максимальний потік у графі – це максимальна кількість одиниць потоку, яка може пройти від витоку до стоку. Іншими словами, це максимальна кількість товарів, інформації чи будь-яких інших одиниць, які можуть бути перевезені за графом від витоку до стоку.

Мінімальний розріз у графі – це мінімальна кількість ребер, які необхідно видалити з графа, щоб розділити його на дві частини, що не перетинаються, що містять виток і стік відповідно. Іншими словами, розріз - це безліч ребер, які, якщо їх видалити з графа, призведуть до того, що джерело і стік будуть у різних компонентах зв'язності.

Мінімальний розріз завжди дорівнює максимальному потоку у графі (малюнок 1), це випливає з теореми Форда-Фалкерсона. Отже, знаходження

максимального потоку та мінімального розрізу зводиться до однієї і тієї ж задачі, і багато алгоритмів використовують цей факт для знаходження обох значень одночасно.



Малюнок 5. Мінімальний розріз (червоний) та максимальний потік (синій)

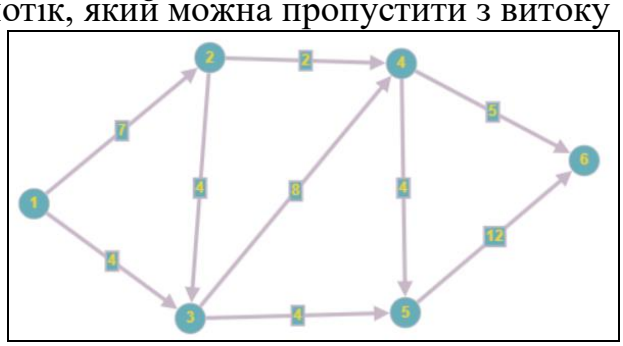
Алгоритм Форда-Фалкерсона

Алгоритм Форда-Фалкерсона є популярним алгоритмом для вирішення проблеми максимального потоку в потоковій мережі. Проблема максимального потоку передбачає визначення максимального обсягу потоку, який можна надіслати від вершини-джерела (виток) до вершини-приймача (стік) в спрямованому зваженому графі з урахуванням обмежень пропускної здатності на ребрах.

Алгоритм працює ітеративно, знаходячи шлях доповнення, який є шляхом від витоку до стоку на залишковому графі, тобто графу, отриманому шляхом віднімання поточного потоку від пропускної здатності кожного ребра. Тоді алгоритм збільшує потік уздовж цього шляху на максимально можливу величину, яка є мінімальною пропускною здатністю ребер уздовж шляху[1].

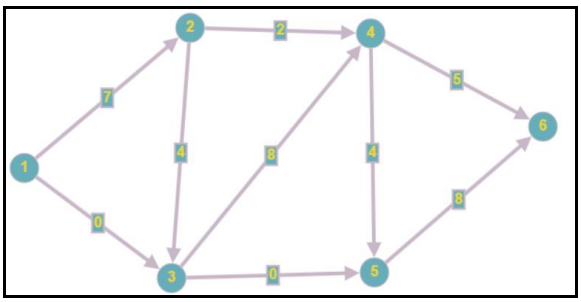
Принцип дії

Принцип дії алгоритму на прикладі графа G (малюнок 2). Є орієнтований граф, у якому вага ребра позначає пропускну здатність між вершинами. Потрібно знайти максимальний потік, який можна пропустити з витоку 1 в стік 6.



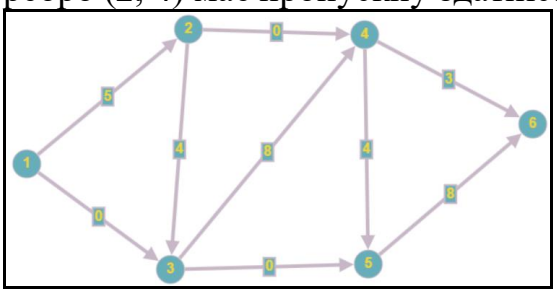
Малюнок 6. Граф G

Допустимо, спочатку ми пройдемо шлях 1 -> 3 -> 5 -> 6. Скільки потоку можна провести цим шляхом? 4 одиниці потоку, оскільки ребро (1, 2) пропустить лише 4. Подивимося на наш граф після першої ітерації алгоритму (малюнок 3)



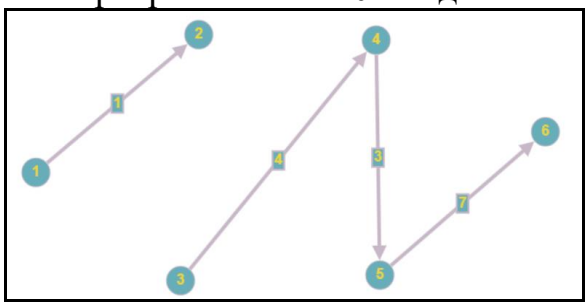
Малюнок 7. Граф після 1-ої ітерації

Від ваги ребер ми забрали ту кількість потоку, яку ми вже пропустили через них. Ребра з вагою 0 ми можемо ігнорувати, оскільки тепер через них вже не можна переправляти потік. Тепер нам потрібно знайти наступний шлях до стоку, допустимо це буде 1 -> 2 -> 4 -> 6. Цей шлях пропускає максимум 2 одиниці потоку, оскільки ребро (2, 4) має пропускну здатність 2 одиниці.



Малюнок 8. Граф після 2-ої ітерації

Пройдемо шлях 1 -> 2 -> 3 -> 4 -> 6, пропускна спроможність шляху – 3. Також після цього відразу пройдемо шлях 1 -> 2 -> 3 -> 4 -> 5 -> 6, його пропускна здатність – 1. Тепер видалимо ребра вага яких 0 і подивимося на граф.



Малюнок 9. Граф після 3-ої та 4-ої ітерації

Можемо помітити, що тепер ми не можемо пройти з витоку до стоку, тому можемо вважати роботу алгоритму завершеною. Складемо потоки, які ми пройшли і отримаємо максимальний потік для даного графа – 10[2].

Висновок

Алгоритм Форда-Фалкерсона є популярним методом для вирішення задачі максимального потоку в потоковій мережі. Він дозволяє визначити максимальний обсяг потоку, який можна відправити від витоку до стоку у спрямованому зваженому графі з урахуванням обмежень пропускної спроможності на ребрах.

Цей метод зручний для застосування на практиці при вирішенні завдань, пов'язаних з оптимізацією транспортних потоків, розподілом ресурсів та інших

задач, пов'язаних із потоками. Розуміння алгоритму Форда-Фалкерсона також необхідне для подальшого вивчення складніших алгоритмів знаходження максимального потоку у графі.

Список літературних джерел

1. *Ford-Fulkerson Algorithm for Maximum Flow Problem. URL: https://www.geeksforgeeks.org/ford-fulkerson-algorithm-for-maximum-flow-problem/*
2. *Алгоритм Форда-Фалкерсона URL: https://web.archive.org/web/20230508223942/https://habr.com/ru/articles/566248/*
3. *зручний візуалізатор графів з великим набором алгоритмів. URL: https://github.com/UnickSoft/graphonline*

УДК 004.41

Ілик В.В., студент 1 курсу спеціальності 122 «Комп'ютерні науки»
Горяшин А.С., асистент кафедри інформаційних технологій

РОБОТА З ДАТАМИ ЗАСОБАМИ ПРОГРАМУВАННЯ МОВОЮ PYTHON

Донецький національний університет імені Василя Стуса

Робота з датами є важливим елементом в програмуванні, оскільки дата і час використовуються в багатьох програмних проектах, наприклад, в системах управління проектами, облікових системах та соціальних мережах. Мова Python має вбудовані функції для маніпулювання датами, які уможливлюють роботу в цій області використання простим та змістовним.

Питання роботи з датами є дуже актуальним в програмуванні, особливо при розробці веб-програм, мобільних додатків, ігор та програм, пов'язаних з аналітикою даних. Python є однією з найпопулярніших мов у світі, і, як наслідок, запит на програмістів Python, які можуть ефективно працювати з датами, також зростає [1].

Востаннє у 2019 році було проведено дослідження з використанням мови програмування Python для роботи з датами (Garcia et al., 2019). Дослідження розглянуло декілька методів роботи з датами, а також провело порівняння ефективності різних методів та їхню точність.

Метою даної роботи є вивчення основ методів маніпулювання датами в мові програмування Python та їх застосування для вирішення практичних завдань.

1. Описати базові функції Python для роботи з датами

2. Дослідити бібліотеки Python для роботи з датами та зіставити їх з базовими функціями