К.В. Квітка, аспір.
А.В. Панішев, д.т.н., проф.
Житомирський державний технологічний університет

АЛГОРИТМИ ПОБУДОВИ НАЙКОРОТШИХ ШЛЯХІВ У ГРАФІ

В теорії графів задача про найкоротший шлях полягає в знаходженні шляху між двома вершинами (або вузлами) такого графу, щоб сума ваг ребер, з яких він складається, була мінімальна. Прикладом може бути знаходження найкоротшого шляху між двома пунктами на дорожній мапі (в цьому випадку, вершинами є пункти, а ребрами – частини дороги із вагами, рівними часу, необхідному для подолання цієї частини).

Така задача часто згадується як задача про найкоротший шлях між парою вершин, але існують і інші варіанти про найкоротший шлях, що відрізняються декількома параметрами:

– задача про найкоротші шляхи з одним входом (необхідно знайти найкоротші шляхи між вхідною вершиною та всіма іншими вершинами);

– задача про найкоротші шляхи з одним виходом (необхідно знайти найкоротші шляхи з усіх вершин графа до однієї вихідної вершини);

– задача про найкоротші шляхи для всіх пар (необхідно знайти найкоротші шляхи між кожною парою вершин у графі).

Ці варіанти задач мають значно дієвіші алгоритми ніж спрощений підхід із запуском алгоритму пошуку найкоротшого шляху між усіма вершинами.

Для будь-яких двох вершин можуть існувати декілька шляхів які їх з'єднують. Шлях, що з'єднує вершини і має мінімальну можливу довжину, називається найкоротшим шляхом.

Математично задача про найкоротший шлях записується так:

Нехай $G = (V, E)$ – орієнтований граф (рис. 1), кожне ребро якого позначено невід'ємним числом (вага ребра). Позначимо деяку вершину 1 й назвемо її витоком. Знайти найкоротші шляхи від витоку 1 до всіх інших вершин графа G.

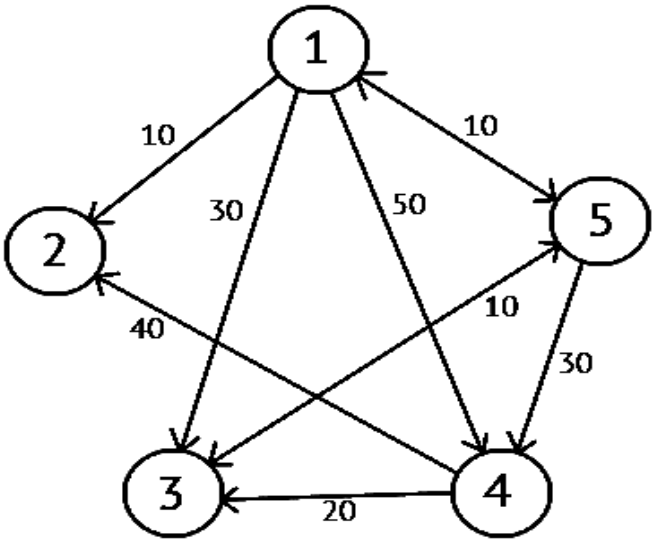

Рис. 1. Орієнтований граф

Одним із найбільш ефективних алгоритмів для пошуку найкоротшого шляху із вихідної вершини у всі інші вершини зваженого графа є алгоритм, запропонований у 1959 р. Дейкстрой. Алгоритм працює тільки для графів без ребер від'ємної ваги.

Ідея алгоритму полягає в наступному. Кожній вершині присвоюємо мітку – мінімальну відому відстань від цієї вершини до сусідніх. Алгоритм працює покроково – на кожному кроці він «відвідує» одну вершину і намагається зменшувати мітки. Робота алгоритму завершується, коли всі вершини «відвідані».

Даний алгоритм широко застосовується в програмуванні і технологіях, наприклад, його використовують протоколи маршрутизації OSPF і IS-IS.

Алгоритм Дейкстри може бути узагальнений і для випадку, коли деякі із дуг мають від'ємне значення. Ідея такого узагальнення належить Форду. Алгоритм полягає в наступному. Спочатку величині потоку присвоюється значення 0. Потім величина потоку ітеративно збільшується за допомогою пошуку збільшуючого шляху (шлях від джерела до стоку, вздовж якого можна послати більший потік). Процес повторюється, поки можна знайти збільшуючий шлях.

Ще одним алгоритмом для пошуку найкоротшого шляху у графі є алгоритм Беллмана–Форда. Даний алгоритм знаходить найкоротші шляхи від однієї вершини графа до всіх інших. На відміну від алгоритму Дейкстри, алгоритм Беллмана–Форда допускає ребра з негативною вагою.

Сам алгоритм Форда–Беллмана являє собою кілька фаз. На кожній фазі проглядаються всі ребра графа, і алгоритм намагається справити релаксацію (relax, ослаблення) уздовж кожного ребра (a, b) ваги c. Релаксація вздовж ребра – це спроба покращити значення $d[b]$ значенням $d[a] + c$. Фактично це означає, що ми намагаємося покращити значення для вершини b, користуючись ребром (a, b) і поточним значенням для вершини a. Стверджується, що достатньо $n - 1$ фази алгоритму, щоб коректно порахувати довжини всіх найкоротших шляхів у графі (цикли негативного ваги відсутні).

Задача про знаходження найкоротшого шляху між кожною парою вершин може бути розв'язана за допомогою багаторазового повторення алгоритму Дейкстри. Але існують алгоритми більш ефективні ніж процедура багаторазового повторення алгоритму Дейкстри – це алгоритми Флойда і Данцига. В обох алгоритмах для довжини дуг допускаються від'ємні значення, але не допускаються наявність контурів від'ємної довжини.

В алгоритмі Флойда як вихідні дані є матриця D^0. Спочатку із цієї матриці вираховується матриця D^1. Потім за матрицею D^1 вираховується матриця D^2 тощо. Процес повторюється доти, доки за матрицею D^{N-t} не буде вирахувана матриця D^N.

Алгоритм Данцига є близьким до алгоритму Флойда, але відрізняється від попереднього лише іншим порядком виконання одних і тих самих операцій.

Для розв'язання задачі про найкоротший шлях для кожної пари вершин зваженого орграфа $G = (V, E)$ використовують алгоритм Флойда–Воршелла. Час його роботи оцінюється величиною, пропорційною n^3. Перевага алгоритму Флойда–Воршелла перед алгоритмом Дейкстри полягає в тому, що він або вирішує задачу на графі з від'ємними вагами дуг або знаходить у ньому контури негативного ваги.

Задача про найкоротші шляхи це задача про пошук шляху, що має мінімальну довжину між парою вершин. Існує багато варіантів задачі про найкоротший шлях, що відрізняються декількома параметрами. Для кожного із варіантів існують свої алгоритми, які є більш ефективні в кожному конкретному випадку.

Найбільш ефективним алгоритмом для пошуку найкоротшого шляху із вихідної вершини у всі інші вершини зваженого графа є алгоритм Дейкстри. Коли деякі із дуг мають від'ємне значення застосовується алгоритм Форда.

Для знаходження найкоротших шляхів від однієї вершини графа до всіх інших використовуємо алгоритм Беллмана–Форда.

Якщо ж необхідно знайти найкоротший шлях між кожною парою вершин, то доцільно застосувати алгоритми Флойда і Данцига.