УДК 004.942
DOI https://doi.org/10.32782/IT/2023-2-12

Володимир БАБИЧ
старший викладач кафедри комп'ютерних наук, Львівський торговельно-економічний університет, вул Туган-Барановського, 10, м. Львів, Україна, 79005, babych_v@lute.lviv.ua
***ORCID:** 0000-0003-1996-9332*

Анатолій КОСТЕНКО
кандидат фізико-математичних наук, професор кафедри комп'ютерних наук, Львівський торговельно-економічний університет, вул Туган-Барановського, 10, м. Львів, Україна, 79005, avak54@lute.lviv.ua
***ORCID:** 0000-0002-2162-7852*

Василь ПЛЕША
старший викладач кафедри комп'ютерних наук, Львівський торговельно-економічний університет, вул Туган-Барановського, 10, м. Львів, Україна, 79005, plesha_v@i.ua
***ORCID:** 0000-0001-5321-9602*

Михайло ПЛЕША
кандидат фізико-математичних наук, доцент кафедри комп'ютерних наук, Львівський торговельно-економічний університет, вул Туган-Барановського, 10, м. Львів, Україна, 79005, milan@lute.lviv.ua
***ORCID:** 0009-0002-6496-1102*

Леся ХМІЛЯРЧУК
старший викладач кафедри комп'ютерних наук, Львівський торговельно-економічний університет, вул Туган-Барановського, 10, м. Львів, Україна, 79005, _lkh@ukr.net
***ORCID:** 0000-0002-1753-6472*

Бібліографічний опис статті: Бабич, В., Костенко, А., Плеша, В., Плеша М., Хмілярчук, Л. (2023). Задача пошуку найкоротшого шляху: порівняльний аналіз основних алгоритмів. *Information Technology: Computer Science, Software Engineering and Cyber Security,* 2, 99–106, doi: https://doi.org/10.32782/IT/2023-2-12

ЗАДАЧА ПОШУКУ НАЙКОРОТШОГО ШЛЯХУ: ПОРІВНЯЛЬНИЙ АНАЛІЗ ОСНОВНИХ АЛГОРИТМІВ

За допомогою графів будуються математичні моделі зв'язків між певними елементами. Наприклад, у вигляді графа можуть бути зображені транспортні, інформаційні, комп'ютерні та інші мережі, карти автомобільних, залізничних, повітряних шляхів, лабіринти і т.п.

Питання про найкоротший шлях і досі є однією з найактуальніших тем у галузі досліджень. Знаходження найкоротших шляхів у графі використовується у різних сферах діяльності, наприклад, для знаходження оптимального маршруту між двома об'єктами на карті місцевості, у логістиці вантажних перевезень, у системах комутації інформаційних пакетів в мережі Internet тощо.

У цій статті представлено основні принципи трьох алгоритмів пошуку найкоротшого шляху та проведено їхнє порівняння шляхом аналізу часової та просторової складності. Крім того, узагальнено область застосування різних алгоритмів. Алгоритм Дейкстри – це класичний алгоритм отримання найкоротшого шляху від конкретної вершини до будь-якої ншої. Його широко використовують в дорожних мережах. Цей алгоритм можна використовувати лише тоді, коли у графі не існує жодного ребра з від'ємною вагою. Алгоритм Беллмана-Форда можна використовувати на графах з від'ємною вагою ребер, якщо граф не містить негативного циклу, доступного з вихідної вершини. Результат роботи цього алгоритму можна використати для визначення існування циклу від'ємної ваги у графі. Алгоритм Флойда-Уоршелла – це алгоритм динамічного програмування, який може вирішити проблему найкоротшого шляху між будь-якими двома вершинами. Метод використовується на зважених графах, у яких можуть бути як додатні, так і від'ємні ваги ребер, проте у ньому не має бути від'ємних циклів. Таким чином, цей метод загальніший у порівнянні з алгоритмом Дейкстри. Однак у практичному застосуванні ці три алгоритми безпосередньо не застосовуються, а проводиться їхня модифікація та оптимізація для підвищення ефективності.

***Ключові слова:** задача пошуку найкоротшого шляху, алгоритм Дейкстри, алгоритм Беллмана-Форда, алгоритм Флойда-Уоршелла, граф.*

Volodymyr BABYCH
Senior Lecturer at the Department of Computer Sciences, Lviv University of Trade and Economics, Tugan-Baranovskoho str., 10, Lviv, Ukraine, 79005, babych_v@lute.lviv.ua
***ORCID:** 0000-0003-1996-9332*

Anatoly KOSTENKO
Candidate of Physical and Mathematical Sciences, Professor at the Department of Computer Sciences, Lviv University of Trade and Economics, Tugan-Baranovskoho str., 10, Lviv, Ukraine, 79005, avak54@lute.lviv.ua
***ORCID:** 0000-0002-2162-7852*

Vasyl PLESHA
Senior Lecturer at the Department of Computer Sciences, Lviv University of Trade and Economics, Tugan-Baranovskoho str., 10, Lviv, Ukraine, 79005, plesha_v@i.ua
***ORCID:** 0000-0001-5321-9602*

Mykhailo PLESHA
Candidate of Physical and Mathematical Sciences, Associate Professor at the Department of Computer Sciences, Lviv University of Trade and Economics, Tugan-Baranovskoho str., 10, Lviv, Ukraine, 79005, milan@lute.lviv.ua
***ORCID:** 0009-0002-6496-1102*

Lesya KHMILYARCHUK
Senior Lecturer at the Department of Computer Sciences, Lviv University of Trade and Economics, Tugan-Baranovskoho str., 10, Lviv, Ukraine, 79005, _lkh@ukr.net
***ORCID:** 0000-0002-1753-6472*

To cite this article: Babych, V., Kostenko, A., Plesha, V., Plesha, M., Khmilyarchuk, L. (2023). Zadacha poshuku naikorotshoho shliakhu: porivnialnyi analiz osnovnykh alhorytmiv [The problem of finding the shortest path: a comparative analysis of the main algorithms]. *Information Technology: Computer Science, Software Engineering and Cyber Security,* 2, 99–106, doi: https://doi.org/10.32782/IT/2023-2-12

THE PROBLEM OF FINDING THE SHORTEST PATH: A COMPARATIVE ANALYSIS OF THE MAIN ALGORITHMS

Mathematical models of connections between certain elements are built with the help of graphs. For example, transport, information, computer and other networks, maps of road, railway, air routes, labyrinths, etc. can be depicted in the form of a graph.

The question of the shortest path is still one of the most relevant topics in the field of research. Finding the shortest paths in a graph is used in various fields of activity, for example, to find the optimal route between two objects on a terrain map, in freight logistics, in information packet switching systems on the Internet, etc.

This article presents the basic principles of three algorithms for finding the shortest path and compares them by analyzing time and space complexity. In addition, the scope of application of various algorithms is summarized. Dijkstra's algorithm is a classic algorithm for obtaining the shortest path from a particular vertex to any other. It is widely used in road networks. This algorithm can be used only when the graph does not have any edge with negative weight. The Bellman-Ford algorithm can be used on graphs with negative edge weights, as long as the graph does not contain a negative cycle accessible from the source vertex. The result of this algorithm can be used to determine the existence of a negative weight cycle in a graph. The Floyd-Warshall algorithm is a dynamic programming algorithm that can solve the shortest path problem between any two vertices. The method is used on weighted graphs, which can have both positive and negative edge weights, but must not contain negative cycles. Thus, this method is more general in comparison with Dijkstra's algorithm. However, in practical application, these three algorithms are not directly used, but their modification and optimization are carried out to improve efficiency.

Key words: *problem of finding the shortest path, Dijkstra's algorithm, Bellman-Ford algorithm, Floyd-Warshell algorithm, graph.*

Вступ. Проблема знаходження найкоротшого шляху є однією з класичних проблем теорії графів. Її метою є пошук найкоротшого шляху між двома вузлами графа. Класичним алгоритмом пошуку найкоротшого шляху є знаходження шляху між двома вузлами таким

чином, щоб сума ваг складових його ребер була мінімальною. Задача про знаходження найкоротшого шляху є однією із найбільш фундаментальних і типових задач теорії графів. Інакше вона ще називається задачею про мінімальний шлях. Практичне значення задачі визначається тим, що вона є елементом складніших задач та моделлю для розробки ефективних методів вирішення задач теорії графів.

Значимість розв'язання даної задачі визначається її різними практичними застосуваннями. Наприклад, в GPS-навігаторах здійснюється пошук найкоротшого шляху між двома перехрестями. Вершинами є перехрестя, а дороги – ребрами, які лежать між ними. Якщо сума довжин доріг між перехрестями мінімальна, тоді знайдений шлях найкоротший.

Відзначимо, що на практиці за допомогою цієї задачі можна розв'язувати не лише задачі обчислення довжини шляху, але й фінансові, матеріальні, паливно-енергетичні задачі. Так, наприклад, у економічних за змістом задачах довжиною ребра можуть бути такі величини, як час, вартість, обсяг витрачених ресурсів. При проходженні шляху ці характеристики додаються, відповідно, як і самі довжини при розрахунку загальної довжини. Це дає змогу мінімізувати вищевказані показники.

Теоретично цілком допустиме дослідження графів із ребрами, у яких значення ваг є від'ємними. Однак задачі такого типу, як правило, не мають простого економічного змісту, і складно вирішуються. Задача про найкоротший шлях має рішення для будь-якого зв'язного графа. При цьому рішенням задачі вважається найкоротший шлях і його довжина. Зрозуміло, що довжина найкоротшого шляху визначається однозначно, в той час як найкоротших шляхів може бути декілька. Рішення задачі про найкоротший шлях може бути проведено двома способами: на основі алгоритму повного перебору усіх можливих шляхів або методом динамічного програмування на графі.

Під час вирішення задачі про найкоротший шлях на основі алгоритму повного перебору шляхів будуються всі можливі шляхи між початковою і кінцевою точками, обчислюються довжини усіх отриманих шляхів і вибирається мінімальна з них. Вибраний шлях (один або кілька) і буде розв'язком задачі. При цьому в алгоритм перебору шляхів можна вносити певні поправки, які дозволяють ігнорувати деякі явно невигідні варіанти. Наприклад, не потрібно продовжувати обчислення довжини шляху, якщо його довжина вже перевищує мінімальну довжину знайдених раніше шляхів. Зрозуміло, що основним недоліком такого алгоритму є його низька швидкодія та велика трудомісткість для складних графів.

Основою методу динамічного програмування є розбиття складної задачі на простіші підзадачі. Якщо задачу можна вирішити оптимально, розбиваючи її на підзадачі, а потім рекурсивно шукати оптимальні рішення підзадач, то вона має оптимальну структуру.

Аналіз досліджень і публікацій. У 1959 р. Дейкстра запропонував алгоритм пошуку у графі, який можна використати для вирішення задачі знаходження найкоротшого шляху між двома вершинами для будь-якого графа з додатними вагами (Dijkstra, 1959). Цей алгоритм пізніше був модифікований Лі в 2006 році і застосований до системи наведення автомобіля. Він має дві модифікації, а саме: найкоротший шлях та найшвидший алгоритм (Chen et al., 2009). Алгоритм визначення найкоротшого шляху обчислює найкоротший маршрут між кожною парою вершин графа, в той час як алгоритм найшвидшого шляху визначає шлях з мінімальним часом його проходження. Майбутній час проходження шляху можна передбачити на основі моделей прогнозування, використовуючи дані про час проходження маршруту (Chen; Makki; Pissinou, 2009).

Meghanathan (2012) розглянув алгоритм Дейкстри та алгоритм Беллмана-Форда для пошуку найкоротшого шляху в графі. Він дійшов висновку, що часова складність алгоритму Дейкстри дорівнює O(|E|·log|V|), в той час як складність алгоритму Беллмана-Форда – O(|V|·|E|) (Meghanathan, 2012) (тут і далі |V| та |E| – позначення, відповідно, кількості вершин та кількості ребер графа (V,E)).

Lili Cao та співавт. (2005) виявили, що пошук найкоротшого шляху є важливим для численних програм на основі графів, особливо розроблених для соціальних мереж. Прикладом може бути платформа LinkedIn, де користувачі виконують запити, щоб знайти найкоротший шлях до «соціальних посилань», яка пов'язує їх із певним користувачем. Цей тип запитів є складним завданням для графів середнього розміру, і стає обчислювально нездійсненним для графів, які моделюють сучасні соціальні мережі, більшість з яких містять мільйони вузлів і мільярди ребер. Автори пропонують програму Atlas – новий підхід до масштабованих наближених найкоротших шляхів між вузлами графа, використовуючи колекцію охоплюючих дерев. Охоплюючі дерева легко генерувати, вони компактні щодо графів, і їх можна розподілити між комп'ютерами для паралелізації

запитів. У роботі показано ефективність такого підходу на прикладі шести великих соціальних графів від Facebook, Orkut та Renren, найбільший з яких включає 43 мільйони вузлів та 1 мільярд ребер. Описано методи поступового оновлення, оскільки соціальні графи з часом змінюються. Було зафіксовано динаміку графів, використовуючи 35 щоденних знімків мережі Facebook, і показано, що Atlas може з часом амортизувати вартість оновлення дерев. Нарешті, Atlas застосовано до кількох графічних додатків і показано, що результати є наближені до ідеальних (Cao L., Zhao X., Zheng H., Zhao B., 2011).

Wadhwa (2000) заявив, що дослідники вирішили проблему проєктування мережі (проблема кабелю та траншеї), яка передбачає компроміс між витратами на використання та капітальними витратами на будівництво мережі. Більша мережа (дерево найкоротшого шляху) може коштувати дорожче, але може зменшити витрати на використання, включаючи привабливіші шляхи відправлення-призначення. І навпаки, менша мережа (мінімальне охоплююче дерево) може збільшити витрати на використання. Надано евристику, яка дає оптимальні або майже оптимальні рішення (Wadhwa, 2000).

Pallottino і Scutella (1997) описали алгоритм найкоротшого шляху в транспортних моделях: класичні та інноваційні аспекти. Вони розглянули алгоритми найкоротшого шляху при перевезеннях у двох аспектах. Перший – це класичні алгоритми, які є найбільш цікавими у транспортних задачах або щодо теоретичних міркувань, або щодо їхньої ефективності, а також з огляду на їхнє практичне використання в транспортних моделях. Другий аспект вказує на динамічні проблеми щодо найкоротшого шляху, які часто виникають у транспортній галузі. Автори проаналізували основні особливості проблем, які існують у відповідних умовах щодо часу транспортування та функцій витрат, загальної «хронологічної» алгоритмічної парадигми, яка називається Chrono-SPT (Pallottino; Scutellà, 1998).

Ahmat (2005) докладно вивчав комунікаційні мережі. У дослідженні описані основні поняття теорії графів та їхній зв'язок із комунікаційними мережами, представлено деякі проблеми оптимізації, пов'язані з протоколами маршрутизації та моніторингом мережі, і показано, що багато проблем оптимізації є NP-повними або NP-складними. Також описано деякі загальні інструменти, які використовуються для створення топологій мережі на основі теорії графів. Хіао вирішив проблему найшвидших онлайн-відповідей на запити, використовуючи насичену симетрію в графах. Алгоритм Дейкстри є найвідомішим та широко використовуваним для вирішення проблеми знаходження найкоротшого шляху, оскільки він швидкий і використовує структуру даних «купа» для пріоритетних черг запитів найкоротшого шляху, необхідних у багатьох додатках (Ahmat, 2009).

Steinhardt (2006) дійшов до висновку, що алгоритм Дейкстри спеціалізується на пошуку найкоротших шляхів між вершинами графа. Andrew V. Goldberg (2008) вивчав алгоритми найкоротшого шляху «вершина–вершина» (P2P). Останнім часом набули розвитку алгоритми пошуку найкоротшого шляху від точки до точки з попередньою обробкою. На практиці алгоритми виявилися ефективними на дорожніх мережах та деяких інших видах графів. Є кілька питань, особливо теоретичних, які залишаються відкритими. Отже, необхідне теоретичне обґрунтування цих алгоритмів. Обчислювальні охоплення – це ще один набір відкритих питань. Можна змінити стандартний алгоритм знаходження найкоротшого шляху для всіх пар, щоб досягти результату за однаковий час, тобто $O^*(n^2)$ для розріджених графів. Оскільки результат для задачі про всі пари дорівнює n^2, можливості для удосконалення обмежені.

Shirinivas та співавт. (2010) представили важливість теоретичних ідей графів у різних областях комп'ютерних додатків, таких як алгоритм найкоротшого шляху в мережі, пошук мінімального охоплюючого дерева, пошук планарності графа, алгоритми пошуку матриць суміжності, алгоритми пошуку зв'язності, алгоритми знаходження циклу в графі, алгоритми пошуку елемента в структурі даних (DFS, BFS) (Shirinivas; Vetrivel; Elango, 2010).

Sommer (2010) досліджував обробку запитів із найкоротшим шляхом у мережах як теоретично, так і практично. Експериментальне дослідження проводилося з використанням дорожньо-транспортної мережі. Дослідження показало простий і загальний метод для ефективної підтримки найкоротших запитів у неорієнтованих графах з дуже низькими накладними витратами на попередню обробку та конкурентним часом запитів за рахунок точності. Ефективність цього методу, який є розумною альтернативою існуючим точним методам, спеціально розробленим для транспортних мереж, доведена для різних типів графів (Sommer, 2010).

Abbasi та ін. (2011) розглядали динамічну проблему знаходження найкоротшого шляху, яку застосовують у динамічних потоках мінімальних витрат для задачі перетворення.

Дослідження показало, що ця проблема еквівалентна класичній задачі пошуку найкоротшого шляху у так званій розширеній у часі мережі (Abbasi; Ebrahimnejad, 2011).

Hung (2003) аналізує обернену проблему знаходження найкоротшої довжини шляху (ISPL) у задачах вдосконалення транспортної мережі та цінового потоку (Hung Cheng-Huang, 2003).

Li та співавт. (2008) запропонували ефективний алгоритм під назвою Li-Qi (LQ) для пошуку простого шляху з найменшою сумарною вагою від конкретної початкової або вихідної вершини до кожної іншої вершини графа (Li T.; Qi L.; Ruan, 2008).

Виклад основного матеріалу. До найефективніших алгоритмів знаходження найкоротшого шляху на графах належать алгоритм Дейкстри (використовується для знаходження оптимального маршруту між двома вершинами), алгоритм Флойда-Уоршелла (для знаходження найкоротшого шляху між парами вершин) і алгоритм Беллмана-Форда (для знаходження оптимального маршруту між усіма парами вершин). Зазначені алгоритми легко виконуються при малій кількості вершин у графі. При збільшенні їхньої кількості задача пошуку найкоротшого шляху ускладнюється, і тому без застосування сучасної комп'ютерної техніки практично не обійтися.

Алгоритм Дейкстри. Алгоритм голландського вченого Едсгера Дейкстри знаходить усі найкоротші шляхи з однієї наперед заданої вершини графа до всіх інших (Dijkstra's algorithm). За його допомогою, при наявності всієї необхідної інформації, можна, наприклад, дізнатися, яку послідовність доріг краще використовувати, щоб дістатися з одного міста до кожного з інших, або в які країни вигідніше експортувати нафту тощо.

Недолік цього методу – неможливість обробки графів, у яких є ребра з від'ємними вагами. Наприклад, якщо для деякої системи передбачено збиткові маршрути, то для роботи з нею алгоритм Дейкстри не підходить.

Щоб цей алгоритм реалізувати, потрібно два масиви: логічний visited – для зберігання інформації про відвідані вершини, і чисельний distance, в який будуть заноситися знайдені довжини найкоротших шляхів. Отже, є граф G = (V, E). Кожна з вершин, які входять у множину V, спочатку відзначена як невідвідана, тобто елементам масиву visited присвоєно значення false.

Оскільки найвигідніші шляхи тільки належить знайти, на початку значенням кожного елемента вектора distance буде таке число, яке завідома є більшим від будь-якого потенційного шляху (зазвичай це є нескінченність, але в програмі використовують, наприклад, максимальне значення конкретного типу даних). Як вихідний пункт вибирається вершина s, для неї записується нульовий шлях: distance[s] = 0, оскільки немає ребра від s до s (метод не передбачає петель).

Далі знаходяться всі сусідні вершини (до яких існує ребро від s) (наприклад, t і u) та досліджуються по черзі, а саме обчислюється вартість маршруту від s до кожної з них:

distance[t]=distance[s] + вага інцидентного до s і t ребра;

distance[u]=distance[s] + вага інцидентного до s і u ребра.

Цілком ймовірно, що до тієї чи іншої вершини від s існує кілька шляхів, тому вартість шляху до такої вершини в масиві distance доведеться переглядати, тоді найбільше (неоптимальне) значення ігнорується, а найменше ставитися у відповідність до вершини.

Після обробки суміжних із s вершин вона позначається як відвідана: visited [s] = true, і активною стає та вершина, шлях до якої від вершини s є мінімальним. Припустимо, шлях від s до u коротший, ніж від s до t, отже, вершина u стає активною. Далі таким самим чином досліджуються її сусіди, за винятком вершини s. Наступним кроком u позначається як пройдена: visited[u] = true, активною стає вершина t, і вся процедура повторюється для неї. Алгоритм Дейкстри триває доти, доки всі доступні від s вершини не будуть досліджені.

Алгоритм Флойда–Уоршелла. Цей метод названо на честь двох американських дослідників Роберта Флойда і Стівена Уоршелла, які одночасно відкрили його у 1962 році.

Алгоритм Флойда–Уоршелла – динамічний алгоритм обчислення значень найкоротших шляхів для кожної з вершин графа. Метод працює на зважених графах, з додатними та від'ємними вагами ребер, але без від'ємних циклів. Таким чином, цей метод загальніший у порівнянні з алгоритмом Дейкстри, оскільки останній не працює з від'ємною вагою ребер, до того ж класична його реалізація має на меті визначення оптимальних відстаней від однієї вершини до всіх інших.

Для реалізації алгоритму Флойда–Уоршелла сформуємо матрицю суміжності D[][] графа G = (V, E), в якому кожна вершина понумерована від 1 до |V|. Ця матриця має розмір |V|·|V|, і кожному її елементові D[i][j] присвоєно вагу ребра від вершини і до вершини j. Під час вико-

нання алгоритму ця матриця буде перезаписуватися: в кожну з її комірок буде заноситися значення, яке визначає оптимальну довжину шляху від вершини i до вершини j (відмова від виділення спеціального масиву для цього заощадить пам'ять і час).

Тепер розглянемо вміст матриці найкоротших шляхів. Оскільки кожен її елемент D[i][j] повинен вміщувати найменший із наявних маршрутів, то відразу можна сказати, що для одиничної вершини він дорівнює нулю, навіть якщо вона має петлю (від'ємні цикли не розглядаються), отже, всім елементам головної діагоналі (D[i][i]) потрібно надати нульове значення.

А щоб нульові недіагональні елементи (матриця суміжності могла мати нульові значення там, де немає безпосереднього ребра між вершинами i та j) змінили по можливості своє значення, позначимо їх рівними нескінченності (в програмі це може бути, наприклад, максимально можлива довжина шляху в графі, або просто – велике число).

Основна частина алгоритму складається із трьох циклів:

```
for k від 1 до |V| do
    for i від 1 до |V| do
      for j від 1 до |V| do
      if D[i][k]+D[k][j] < D[i][j] then D[i][j] ← D[i][k]+D[k][j]
```

Найкоротший шлях від вершини i до вершини j може проходити як через них самих, так і через множину інших вершин k ∈ (1, …, |V|). Оптимальним від i до j буде шлях, який або проходить через k, або не проходить. Якщо шлях проходить через вершину k, потрібно замінити значення найкоротшого шляху D[i][j] сумою D[i][k] + D[k][j]. Іншими словами, як результат виконання алгоритму матриця суміжності буде замінена матрицею найкоротших шляхів.

Алгоритм Флойда–Уоршелла ефективний при обчисленні усіх найкоротших шляхів у щільних графах, коли є багато пар ребер між парами вершин. Його складність O(|V|3). У випадку розріджених графів ефективнішим є алгоритм Дейкстри.

Алгоритм Беллмана–Форда. Історія алгоритму пов'язана з трьома незалежними математиками Лестером Фордом, Річардом Беллманом та Едвардом Муром, іноді його називають алгоритмом Беллмана–Форда–Мура. Цей метод використовується в деяких протоколах дистанційно-векторної маршрутизації, наприклад у RIP (Routing Information Protocol).

Алгоритм обчислює найкоротші шляхи від однієї вершини до всіх інших у зваженому графі (як і алгоритм Дейкстри). Він може працювати на графах, які мають ребра з від'ємними вагами. Проте застосувати його можна не до всіх таких графів, оскільки кожен черговий прохід шляхом, який складається з ребер з від'ємною сумою ваг, лише покращує результат. Нескінченна кількість покращень робить неможливим визначення одного конерктного оптимального значення. Тому алгоритм Беллмана–Форда не можна застосувати до графів з від'ємними циклами, проте він дозволяє визначити такі графи.

Щоб знайти всі найкоротші шляхи від вершини s до всіх інших за допомогою алгоритму Беллмана–Форда, потрібно скористатися методом динамічного програмування: розбити задачу на підзадачі та знайти їхній розв'язок. Розв'язком кожної такої підзадачі є визачення найкоротшого шляху від однієї окремо взятої вершини до якоїсь іншої.

Для зберігання результатів роботи алгоритму створимо одновимірний масив d[]. У i-му елементі цього масиву буде зберігатися значення найкоротшого шляху від вершини s до вершини i (якщо таке існує). Спочатку присвоїмо елементам масиву d[] значення, які дорівнюють умовній нескінченності (наприклад, число, яке більше за суму всіх ваг), а елементові d[s] присвоїмо значення нуль, оскільки відомо, що шлях від вершини s до неї самої рівний 0. Необхідно припустити, що інші вершини з вершини s недоступні. Під час виконання алгоритму для деяких з них це припущення виявиться хибним і будуть обчислені оптимальні значення довжини шляху до цих вершин з вершини s.

Отже, задано граф G=(V, E), n=|V|, a m=|E|. Позначимо суміжні вершини цього графа символами v і u, а вагу ребра (v, u) символом w. Іншими словами, вага ребра з вершини v до вершини u будет дорівнювати w.

Основна частина алгоритму Белмана–Форда:

```
for i від 1 до n-1 do
    for j от 1 до m do
    if d[v] + w(v, u) < d[u] then d[u] ← d[v] + w(v, u)
```

На кожному n-му кроці здійснюється спроба покращити значення елементів масиву d[]: якщо сума ваги ребра w(v, u) і ваги, яка зберігається в d[v], менша ваги d[u], то вона присвоюється останньому.

Складність алгоритму при представленні графа списком ребер становить O(|V|·|E|). Якщо ж граф задати матрицею суміжності, алгоритм буде виконуватися за час O(|E|3).

Висновки. У підсумку, проаналізувавши основні алгоритми знаходження найкоротших шляхів у графі, зазначимо. Якщо необхідно знайти відстань від однієї вершини до іншої або до всіх вершин графа і ваги всіх ребер графа

є додатними або дорівнюють нулю, то найефективнішим є алгоритм Дейкстри із часом роботи O(|*E*|·lg|V|). Якщо ж ваги ребер можуть бути від'ємними, то необхідно застосовувати алгоритм Беллмана–Форда, час роботи якого O(|V|·|E|). Якщо необхідно знайти відстані між усіма парами вершин графа, граф є розрідженим і всі ребра мають невід'ємні ваги, то можна виконати |V| разів алгоритм Дейкстри. Якщо необхідно знайти відстані між усіма парами вершин, ваги ребер можуть бути від'ємними і граф не є розрідженим (|*E*| прямує до $|V|^2$), то доречно використовувати алгоритм Флойда–Уоршелла. Жоден з наведених алгоритмів не може бути застосований для графів, які містять негативні цикли. Проте алгоритм Беллмана-Форда, як і алгоритм Флойда–Уоршелла, допоможуть такі цикли виявити.

ЛІТЕРАТУРА:

1. Dijkstra's algorithm. Wikipedia: The Free Encyclopedia. URL: https://en.wikipedia.org/wiki/Dijkstra%27s_algorithm (application date: 20.06.2023)

2. Abbasi S., Ebrahimnejad S. Finding the Shortest Path in Dynamic Network using Labeling Algorithm. *International Journal of Business and Social Science*. 2011. Vol. 2. № 20. P. 239–243.

3. Ahmat K. A. (n.d.). Graph Theory and Optimization Problems for Very Large Networks. City University of New York. New York, Information Technology. 2009. 5 p.

4. Chen K., Makki K., Pissinou N. A real-time wireless route guidance system for urban traffic management and its performance evaluation. 2009 IEEE 70th Vehicular Technology Conference Fall. 2009. 5 p.

5. Dijkstra E.W. A note on two problems in connexion with graphs. *Numerische Mathematik.* 1959. Vol. 1. P. 269–271.

6. Hung Cheng-Huang. Inverse Shortest Path Length Problem. School of Industrial and systems engineering Georgia Institute of Technology. 2003. 24 p.

7. Li T., Qi L., Ruan D. An efficient Algorithm for the single source Shortest Path Problem in Graph Theory. International Conference on Intelligent System and Knowledge Engineering. 2008. P. 152–157.

8. Cao L., Zhao X., Zheng H., Zhao B. ZhaoApproximating Shortest Path in Social Graph. UC Santa Barbara: Computer Science Department. 2011. 20 p.

9. Meghanathan D. N. (n.d.). Review of Graph Theory. MS: Department of Computer Science, Jackson State University. 2012.

10. Pallottino S., Scutellà M.G. Shortest Path Algorithms in Transportation models: classical and innovative aspects. *Equilibrium and Advanced Transportation Modelling* / Patrice Marcotte, Sang Nguyen. Springer New York, NY, 1998. P.245–281.

11. Shirinivas S.G., Vetrivel S., Elango N.M. Application of Graph theory in Computer Science an overview. International Journal of Engineering Science and Technology. 2010. Vol. 2 (9). P. 4610–4621.

12. Sommer C. Approximate Shortest Path and Distance Queries in Networks. *Graduate School of Information Science and Technology*: Department of Computer Science, 2010. Tokyo, 2010.

13. Wadhwa Sh. Analysis of a network design problem. Lehigh University. 2000.

REFERENCES:

1. Dijkstra's algorithm. Wikipedia: The Free Encyclopedia. URL: https://en.wikipedia.org/wiki/Dijkstra%27s_algorithm (application date: 20.06.2023)

2. Abbasi, S., Ebrahimnejad, S. (2011). Finding the Shortest Path in Dynamic Network using Labeling Algorithm. *International Journal of Business and Social Science, 1, 20*, 239–243.

3. Ahmat, K. A. (n.d.). (2009). Graph Theory and Optimization Problems for Very Large Networks. *City University of New York. New York, Information Technology*.

4. Chen, K., Makki, K., Pissinou, N. (2009). A real-time wireless route guidance system for urban traffic management and its performance evaluation. *2009 IEEE 70th Vehicular Technology Conference Fall*.

5. Dijkstra, E.W. (1989). A note on two problems in connexion with graphs. *Numerische Mathematik, 1*, 269–271.

6. Hung, Cheng-Huang. (2003). Inverse Shortest Path Length Problem. *School of Industrial and systems engineering Georgia Institute of Technology*.

7. Li, T., Qi, L., Ruan, D. (2008). An efficient Algorithm for the single source Shortest Path Problem in Graph Theory. *International Conference on Intelligent System and Knowledge Engineering,* 152–157.

8. Cao, L., Zhao, X., Zheng, H., Zhao, B. (2011). ZhaoApproximating Shortest Path in Social Graph. *UC Santa Barbara: Computer Science Department*.

9. Meghanathan, D. N. (n.d.). (2012). Review of Graph Theory. *MS: Department of Computer Science, Jackson State University*.

10. Pallottino, S., Scutellà, M.G. (1998). Shortest Path Algorithms in Transportation models: classical and innovative aspects. *Equilibrium and Advanced Transportation Modelling (pp. 245–281).* NY: Springer New York.

11. Shirinivas, S.G., Vetrivel, S., Elango, N.M. (2010). Application of Graph theory in Computer Science an overview. *International Journal of Engineering Science and Technology, 2(9)*, 4610–4621.

12. Sommer, C. (2010). Approximate Shortest Path and Distance Queries in Networks. *Graduate School of Information Science and Technology*. Tokyo, 2010.

13. Wadhwa, Sh. (2000). Analysis of a network design problem. Lehigh University.