ФАКУЛЬТЕТ КОМП'ЮТЕРНИХ НАУК ТА КІБЕРНЕТИКИ
КИЇВСЬКОГО НАЦІОНАЛЬНОГО УНІВЕРСИТЕТУ
ІМЕНІ ТАРАСА ШЕВЧЕНКА

Андрій Заворотинський
Роман Якимів
Тетяна Шакотько

Дослідження операцій

Транспортна задача та потоки на мережах

Навчальний посібник

для здобувачів вищої освіти першого (бакалаврського) рівня
ОПП «Інформатика» спеціальності F3 «Комп'ютерні науки»

Київ – 2026

УДК 519.8(075.8)

Рецензенти:

доктор фізико-математичних наук, професор Ігор Самойленко;
кандидат фізико-математичних наук, доцент Тарас Панченко, завідувач кафедри теорії та технології програмування факультету комп'ютерних наук та кібернетики.

Рекомендовано до друку вченою радою факультету комп'ютерних наук та кібернетики Київського національного університету імені Тараса Шевченка
(протокол № 13 від 21 квітня 2026 р.)

Ухвалено науково-методичною комісією факультету комп'ютерних наук та кібернетики Київського національного університету імені Тараса Шевченка
(протокол № 10 від 20 квітня 2026 р.)

кандидат фізико-математичних наук, доцент кафедри дослідження операцій Андрій Володимирович Заворотинський;

кандидат фізико-математичних наук, доцент, доцент кафедри дослідження операцій Роман Ярославович Якимів;

заступник декана факультету комп'ютерних наук та кібернетики, асистент кафедри дослідження операцій Тетяна Іванівна Шакотько.

Дослідження операцій. Транспортна задача та потоки на мережах : навчальний посібник / А. В. Заворотинський, Р. Я. Якимів, Т. І. Шакотько. — Київ, 2026. — 83 с.

Навчальний посібник присвячено транспортній задачі та потокам на мережах. У ньому викладено матеріал відповідного розділу обов'язкової дисципліни «Дослідження операцій» для здобувачів вищої освіти першого (бакалаврського) рівня, які навчаються за освітньо-професійною програмою «Інформатика» спеціальності F3 «Комп'ютерні науки».

Видання призначене для студентів факультету комп'ютерних наук та кібернетики Київського національного університету імені Тараса Шевченка, а також для всіх, хто цікавиться методами дослідження операцій та їх застосуванням у сучасних інформаційних технологіях.

Зміст

Передмова

Навчальний посібник *«Дослідження операцій. Транспортна задача та потоки на мережах»* підготовлено для здобувачів першого (бакалаврського) рівня вищої освіти спеціальності F3 «Комп'ютерні науки» (освітня програма «Інформатика») факультету комп'ютерних наук та кібернетики Київського національного університету імені Тараса Шевченка. Видання призначено для методичної підтримки практичних занять, організації самостійної роботи студентів і систематизації основних понять, моделей, методів та алгоритмів розв'язування задач з відповідного розділу курсу «Дослідження операцій».

Посібник створено на основі багаторічного досвіду викладання дисципліни на кафедрі дослідження операцій. Під час його підготовки враховано специфіку математичної підготовки студентів IT-напрямів, потребу в поєднанні строгого теоретичного викладу з алгоритмічною чіткістю, а також необхідність показати прикладну цінність оптимізаційних моделей для сучасної інформатики. Добір матеріалу орієнтовано на формування вмінь будувати математичні моделі, аналізувати структуру задачі, вибирати адекватний метод розв'язування та інтерпретувати отриманий результат.

У центрі уваги видання перебувають транспортна задача та задачі про потоки на мережах як класичні розділи дослідження операцій. Ці теми природно поєднують математичне програмування, елементи теорії графів, алгоритмічні процедури та прикладні постановки, пов'язані з логістикою, маршрутизацією, розподілом ресурсів, передаванням даних і аналізом пропускної здатності мереж. У посібнику розглянуто постановку транспортної задачі, умову балансу, способи побудови початкового опорного плану, метод потенціалів, а також задачі про найкоротший шлях і максимальний потік на мережі.

Викладачами кафедри дослідження операцій у попередні роки було підготовлено низку навчальних і методичних видань, присвячених методам оптимізації, транспортній задачі, потокам на мережах і практичним аспектам лінійного програмування. Ці праці становлять важливу методичну основу викладання відповідного розділу курсу. Водночас розвиток освітніх програм, зміна акцентів у підготовці студентів, поява нових прикладних контекстів і потреба в більш цілісному, сучасному та узгодженому поданні матеріалу зумовлюють необхідність їх оновлення.

Структуру посібника підпорядковано завданням практичного навчання. Кожний розділ побудовано за єдиною схемою і містить такі підрозділи: *Короткі теоретичні відомості*, *Алгоритми*, *Типові задачі*, *Задачі для самостійного розв'язування*, *Прикладні задачі*, *Питання для самоконтролю*. Така організація матеріалу дає змогу послідовно перейти від основних означень і теоретичних фактів до опанування алгоритмів, розв'язування типових вправ, самостійної роботи та змістової інтерпретації моделей у прикладних ситуаціях.

Для полегшення сприйняття та кращої навігації в тексті використано структуровані оточення для означень, теорем, алгоритмів, прикладів, зауважень і задач. Таке оформлення дає змогу чітко відокремити різні типи навчального матеріалу, підкреслити логіку викладу та зробити роботу з посібником зручнішою як під час аудиторних занять, так і в процесі самостійного опрацювання.

Під час підготовки рукопису інструменти штучного інтелекту використовувалися виключно як допоміжний засіб для мовного редагування, уточнення стилістики та структурування окремих фрагментів тексту. Усі змістові рішення, математичні формулювання, добір задач і перевірку коректності матеріалу виконано авторами, які несуть повну відповідальність за якість видання.

Є підстави сподіватися, що посібник буде корисним студентам під час аудиторної роботи, самостійного опрацювання матеріалу та підготовки до контрольних заходів, а викладачам слугуватиме зручною методичною основою для проведення практичних занять із цієї тематики.

Вступ

Дослідження операцій посідає важливе місце в фундаментальній математичній підготовці здобувачів спеціальності F3 «Комп'ютерні науки». Ця дисципліна формує системний підхід до побудови математичних моделей, аналізу обмежень, вибору критеріїв оптимальності та пошуку ефективних рішень у задачах розподілу ресурсів, планування, керування та організації складних систем. Для майбутніх фахівців з комп'ютерних наук цей розділ математики є не лише теоретичним фундаментом, а й дієвим інструментом формалізації та розв'язування прикладних задач, пов'язаних з інформаційними технологіями, логістичними процесами, телекомунікаційними системами та обчислювальними мережами.

Серед численних моделей і методів дослідження операцій особливе місце належить транспортній задачі та задачам про потоки на мережах. Саме ці розділи наочно демонструють, як математична постановка практичної проблеми приводить до побудови чіткої оптимізаційної моделі, а далі — до розроблення й застосування ефективного алгоритму. Транспортна задача дає змогу дослідити принципи збалансування попиту й запасів, побудови допустимого плану перевезень і послідовного поліпшення цього плану до оптимального. Задачі про потоки на мережах, своєю чергою, відкривають можливість вивчати маршрути, пропускні спроможності, структуру мережі та закономірності передавання ресурсів або інформації в системах різної природи.

Історично методи оптимізації, що становлять зміст цього розділу курсу, виникли як відповідь на реальні економічні, транспортні та інженерні виклики. Транспортна задача, перші постановки якої пов'язують з працями Ф. Гічкока та Л. Канторовича у 30–40-х роках XX століття, стала одним із перших переконливих прикладів застосування математичного програмування до масових процесів розподілу та перевезення. Подальший розвиток теорії графів і алгоритмів на мережах, зокрема праці Л. Форда, Д. Фалкерсона, Е. Дейкстри та інших дослідників, істотно розширив межі застосування оптимізаційних підходів і зробив можливим систематичне розв'язування задач маршрутизації, аналізу мережевих структур і керування потоками. У сучасних умовах ці ідеї набули особливої ваги у зв'язку з розвитком глобальних телекомунікаційних мереж, комп'ютерних систем і цифрової інфраструктури.

Сьогодні транспортні та мережеві моделі є невід'ємною складовою сучасних IT-застосувань. Транспортні задачі природно виникають під час балансування навантаження між серверами та хмарними сховищами, у задачах оптимального розподілу завдань в обчислювальних кластерах, плануванні використання ресурсів і координації потоків даних між вузлами системи. Задачі про потоки на мережах, зокрема пошук найкоротшого шляху та знаходження максимального потоку, становлять теоретичну основу побудови маршрутів у комп'ютерних мережах, аналізу пропускної здатності каналів передавання даних, оцінювання надійності мережевої інфраструктури та проєктування ефективних інформаційних систем. Завдяки цьому вивчення транспортної задачі й потоків на мережах виходить далеко за межі класичних прикладів з логістики і стає безпосередньо пов'язаним із завданнями сучасної інформатики.

Особливу методичну цінність цих тем становить їх наочність і водночас змістова глибина. Транспортна задача та задачі про потоки на мережах дають змогу працювати з моделями, які легко інтерпретуються в термінах постачання, маршрутів, вузлів, дуг, запасів, потреб і пропускних спроможностей, але при цьому вимагають чіткого математичного мислення, уважного врахування обмежень і послідовного застосування алгоритмів. Саме тому вони є зручним навчальним матеріалом для формування в студентів уміння переходити від реальної ситуації до математичної моделі, а від моделі — до обґрунтованого оптимального розв'язку.

Важливою особливістю опанування цих тем є тісне поєднання теоретичної строгості

з алгоритмічною реалізацією. Означення транспортної моделі, опорного плану, потенціалів, мережі, маршруту, потоку, розрізу, пропускної спроможності та збільшуючого шляху мають бути засвоєні не лише на рівні понять і формулювань, а й доведені до рівня послідовних алгоритмічних дій. Саме тому ключову увагу в посібнику приділено алгоритмам побудови початкового опорного плану, застосуванню методу потенціалів, пошуку найкоротшого шляху та знаходженню максимального потоку на мережі. Для фахівця з комп'ютерних наук вміння перейти від абстрактної математичної моделі до чіткої процедури розв'язування є однією з найважливіших професійних компетентностей.

Суттєвою рисою цього матеріалу є також те, що він поєднує кілька ліній математичної підготовки. З одного боку, транспортна задача природно пов'язана з лінійним програмуванням і системою його основних понять; з іншого боку, задачі про потоки на мережах спираються на графові моделі, властивості шляхів, розрізів і зв'язності. Така міжтемна єдність робить відповідний розділ курсу особливо цінним у дидактичному плані: студент бачить, як різні математичні ідеї працюють узгоджено, взаємно доповнюють одна одну і приводять до практично значущих результатів. У цьому сенсі транспортна задача та потоки на мережах слугують одним із найзручніших прикладів інтеграції математичного програмування, дискретної математики й алгоритмічного мислення.

Матеріал цього посібника присвячено саме тим питанням, які утворюють ядро відповідного розділу курсу «Дослідження операцій». Його побудовано з орієнтацією на послідовне засвоєння основних понять, методів і алгоритмів, а також на формування практичних навичок розв'язування задач. Теоретичні положення супроводжуються алгоритмічними схемами, типовими прикладами та задачами різного рівня, що дає змогу поєднати розуміння математичної суті методу з умінням застосовувати його в конкретній ситуації. Такий підхід є особливо важливим для студентів комп'ютерних наук, для яких математична модель має бути не лише правильно побудованою, а й придатною до подальшої алгоритмізації та програмної реалізації.

Отже, вивчення транспортної задачі та потоків на мережах слугує природним містком між класичною математикою та сучасними комп'ютерними науками. Засвоєння цього матеріалу створює необхідну базу для подальшого вивчення складніших оптимізаційних методів, теорії алгоритмів, мережевих моделей і засобів їх практичного застосування в реальних IT-проєктах. Саме тому цей розділ курсу «Дослідження операцій» має не лише самостійну навчальну цінність, а й важливе значення для загальної професійної підготовки майбутнього фахівця з комп'ютерних наук.

Тема 1. Транспортна задача: модель, початковий план і метод потенціалів

Короткі теоретичні відомості

Транспортна задача є однією з класичних задач лінійного програмування і водночас важливою складовою курсу «Дослідження операцій», оскільки поєднує елементи математичного програмування, мережевого моделювання та алгоритмічної оптимізації. Вона належить до спеціального класу оптимізаційних моделей, для яких структура обмежень настільки виразна, що дає змогу застосовувати не загальні універсальні методи, а спеціалізовані алгоритми. Одним із таких алгоритмів є метод потенціалів, який спирається на табличне подання транспортної моделі, використовує її балансну природу та допускає наочну графову інтерпретацію. Саме тому вивчення транспортної задачі має подвійне значення: з одного боку, воно розкриває важливий спеціальний клас оптимізаційних моделей, а з іншого — демонструє, як структура задачі впливає на вибір алгоритму та на форму його практичної реалізації.

На відміну від загального симплекс-методу, що застосовується до задач із довільною матрицею обмежень, транспортна задача має спеціальну систему обмежень типу «запаси–потреби». У цій моделі кожний рядок транспортної таблиці відповідає пункту відправлення із заданим запасом продукції, а кожний стовпчик — пункту призначення з фіксованою потребою. Така організація даних дає змогу подати задачу в компактній та наочній табличній формі, яка істотно полегшує аналіз допустимих розв'язків. Завдяки цьому початкові опорні плани можна будувати за простими конструктивними правилами, не звертаючись щоразу до загальної симплексної таблиці. Подальше покращення розв'язку здійснюється через локальні перерозподіли перевезень по циклах транспортної таблиці. Саме ця риса робить транспортну задачу зручною як для теоретичного вивчення, так і для алгоритмічної реалізації.

Метод потенціалів спирається на ідеї двоїстості, які в транспортній задачі набувають наочної та обчислювально зручної форми. Для кожного опорного плану обчислюються потенціали рядків і стовпчиків, що задають внутрішні оцінки тарифів перевезення і фактично дають змогу перевірити, чи можна поліпшити поточний план. У межах цього підходу оптимальність перевіряється не безпосереднім порівнянням усіх можливих планів, а аналізом небазисних клітин транспортної таблиці. Якщо для жодної з таких клітин неможливо виконати перерозподіл, що зменшив би сумарні витрати, то поточний опорний план є оптимальним. Якщо ж знаходиться клітина, введення якої в базис дає змогу поліпшити план, то разом із базисними клітинами вона породжує єдиний цикл, за яким виконується перерахунок перевезень. У результаті зберігається допустимість плану, тобто виконання всіх балансних умов, і водночас зменшується значення цільової функції. Такий механізм покрокового покращення робить метод потенціалів не лише математично обґрунтованим, а й дуже прозорим з погляду обчислювальної логіки.

Для майбутнього фахівця з комп'ютерних наук транспортна задача має виразне прикладне значення. Передусім вона є природною моделлю для широкого кола задач логістики, розподілу ресурсів, планування постачання, координації потоків даних і балансування навантаження між різними вузлами системи. Крім того, мережева інтерпретація транспортної задачі безпосередньо пов'язує її з моделями мінімальної вартості потоку, маршрутизацією та плануванням пропускних здатностей. Отже, вивчення транспортної задачі створює природний перехід до ширшого кола мережевих моделей і алгоритмів.

Не менш важливо й те, що метод потенціалів наочно розкриває фундаментальну ідею алгоритмічної оптимізації — послідовне покращення розв'язку. Процес розв'язування починається з побудови початкового опорного плану: він ще не є оптимальним, але вже

задовольняє всі обмеження задачі. Далі за допомогою потенціалів цей план крок за кроком наближається до оптимального. Кожний такий крок має чітку логіку та спирається на зрозумілі локальні зміни в таблиці. Для майбутніх фахівців з комп'ютерних наук такий підхід є надзвичайно корисним: він демонструє, як математично складна проблема зводиться до прозорого алгоритму. Опанування транспортної задачі та методу потенціалів формує міцне підґрунтя для розуміння складніших мережевих моделей і розроблення ефективних методів оптимізації в сучасних інформаційних системах.

IT-коментар: основні застосування транспортної задачі

Транспортна задача має багато прикладних інтерпретацій у комп'ютерних науках та інформаційних технологіях. Найтиповіші з них такі.

1. **Балансування навантаження в розподілених обчисленнях.** У системах з багатьма серверами, обчислювальними вузлами або віртуальними машинами потрібно розподілити набір задач між доступними ресурсами так, щоб мінімізувати сумарні витрати часу, енергії або використання мережі. У такій постановці джерела відповідають обчислювальним вузлам із доступною потужністю, а пункти призначення — завданням або групам запитів із певними потребами в ресурсах.

2. **Розміщення даних у дата-центрах і хмарних сховищах.** Під час реплікації, резервування або міграції даних необхідно визначити, як найкраще розподілити обсяги інформації між серверами чи сховищами з урахуванням їх місткості та вартості передавання. Транспортна модель дає змогу формалізувати таку задачу і знайти план розміщення, що мінімізує сумарні витрати на передавання й зберігання.

3. **Маршрутизація й передавання потоків даних у мережах.** У мережевих системах часто потрібно розподілити потоки даних між каналами зв'язку або мережевими вузлами з урахуванням їх пропускної здатності, вартості передавання чи затримок. У спрощених постановках транспортна задача може використовуватися як модель початкового розподілу таких потоків між доступними маршрутами або точками обробки.

4. **Планування обслуговування запитів у сервісних системах.** У веб-сервісах, хмарних платформах і центрах обробки звернень транспортна задача виникає під час розподілу запитів користувачів між серверами, групами обробки або центрами підтримки. Вага перевезення в такій моделі може відображати час відповіді, вартість обслуговування, завантаженість системи або інші критерії якості сервісу.

5. **Оптимізація взаємодії між компонентами складених IT-систем.** У розподілених застосунках, мікросервісній архітектурі та системах обміну даними між модулями часто потрібно визначити ефективну схему передавання інформації між джерелами й споживачами. Транспортна задача дає змогу описати таку систему в термінах запасів, потреб і вартостей взаємодії та знайти оптимальний план обміну, який мінімізує затримки, навантаження на канали або операційні витрати.

Постановка транспортної задачі та позначення

У пунктах відправлення A_i $(i = 1, \dots, m)$ зосереджено a_i одиниць однорідного продукту, а у пунктах призначення B_j $(j = 1, \dots, n)$ задано потреби b_j одиниць. Вартість перевезення одиниці продукту з A_i у B_j дорівнює c_{ij}. Нехай x_{ij} — кількість продукту, що перевозиться з A_i у B_j. За припущенням збалансованості $\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j$ потрібно скласти план перевезень, який мінімізує сумарні витрати.

Математична модель має вигляд:

$$\begin{aligned} & L(X) = \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} \;\to\; \min, \\ & \sum_{j=1}^{n} x_{ij} = a_i, \qquad i = 1, \dots, m, \\ & \sum_{i=1}^{m} x_{ij} = b_j, \qquad j = 1, \dots, n, \\ & x_{ij} \geq 0, \qquad i = 1, \dots, m,\ j = 1, \dots, n, \\ & \sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j. \end{aligned} \tag{1}$$

Остання рівність визначає *збалансовану транспортну задачу.*

У стандартній векторно-матричній формі задачі лінійного програмування план перевезень зручно подавати як

$$x = \operatorname{vec}(X) = (x_{11}, \dots, x_{1n}, x_{21}, \dots, x_{mn})^{\mathsf{T}} \in \mathbb{R}^{mn}, \qquad c = \operatorname{vec}(C) = (c_{11}, \dots, c_{mn})^{\mathsf{T}} \in \mathbb{R}^{mn}.$$

Вектор запасів і потреб:

$$P_0 = (a_1, \dots, a_m, b_1, \dots, b_n)^{\mathsf{T}} \in \mathbb{R}^{m+n}.$$

Для кожної клітини (i, j) введемо вектор комунікації:

$$P_{ij} = (0, \dots, 0, \underset{i}{1}, 0, \dots, 0,\ 0, \dots, 0, \underset{m+j}{1}, 0, \dots, 0)^{\mathsf{T}} \in \mathbb{R}^{m+n}.$$

Тоді система балансу записується як

$$\sum_{i=1}^{m}\sum_{j=1}^{n} x_{ij}P_{ij} = P_0.$$

Основні означення та факти

Означення 1.1: Маршрут і цикл (ланцюг і цикл клітин)

Послідовність комунікацій (еквівалентно: клітин таблиці)

$$\overline{A_{i_1}B_{j_1}},\ \overline{A_{i_2}B_{j_1}},\ \overline{A_{i_2}B_{j_2}},\ \dots,\ \overline{A_{i_s}B_{j_{s-1}}},\ \overline{A_{i_s}B_{j_s}},$$

у якій індекси $i_1, \dots, i_s$ попарно різні і $j_1, \dots, j_s$ попарно різні, називається *маршрутом* (ланцюгом), що сполучає A_{i_1} та B_{j_s}. Маршрут, до якого додано комунікацію $\overline{A_{i_1}B_{j_s}}$, називається *циклом.*

Зауваження 1.1: Властивості транспортної задачі

1. Збалансована транспортна задача є допустимою та має оптимальний розв'язок.
2. Ранг матриці обмежень дорівнює $m+n-1$; тому ДБР (допустимий базисний розв'язок) містить не більше, ніж $m+n-1$ ненульових перевезень.
3. Якщо всі a_i та b_j — цілі, то існує оптимальний план із цілими x_{ij} (цілочисельність забезпечується тотальною унімодулярністю матриці обмежень).

Теорема 1.1: Критерій базисності: відсутність циклу

План X є ДБР транспортної задачі тоді і тільки тоді, коли з його базисних (основних) клітин неможливо утворити цикл.

Теорема 1.2: Критерій оптимальності (метод потенціалів)

Нехай $X=(x_{ij})$ — ДБР. План X є оптимальним тоді і тільки тоді, коли існують потенціали u_i $(i=1,\dots,m)$ та v_j $(j=1,\dots,n)$ такі, що

$v_j - u_i = c_{ij}$ для базисних клітин (i,j), $\qquad v_j - u_i \le c_{ij}$ для небазисних клітин (i,j).

Незбалансовані транспортні задачі та зведення до збалансованих

Транспортна задача називається *незбалансованою* (відкритою), якщо

$$\sum_{i=1}^{m} a_i \neq \sum_{j=1}^{n} b_j.$$

У цьому випадку задача у стандартній формі з рівностями $\sum_j x_{ij} = a_i$, $\sum_i x_{ij} = b_j$ не має допустимого плану, однак її легко звести до збалансованої (закритої) задачі, доповнивши модель фіктивним пунктом.

Випадок 1: надлишок запасу.
Якщо $\sum_{i=1}^{m} a_i > \sum_{j=1}^{n} b_j$, вводиться *фіктивний пункт призначення* B_{n+1} з потребою

$$b_{n+1} = \sum_{i=1}^{m} a_i - \sum_{j=1}^{n} b_j.$$

Тарифи $c_{i,n+1}$ задаються за змістом моделі:

- $c_{i,n+1}=0$ — якщо «недовикористання запасу» не штрафується (резерв/склад/простій без вартості);
- $c_{i,n+1}=p_i$ — якщо задається *штраф* за невикористаний ресурс у пункті A_i.

Отримуємо збалансовану задачу розмірності $m \times (n+1)$.

Випадок 2: дефіцит запасу.
Якщо $\sum_{i=1}^{m} a_i < \sum_{j=1}^{n} b_j$, вводиться *фіктивний пункт відправлення* A_{m+1} із запасом

$$a_{m+1} = \sum_{j=1}^{n} b_j - \sum_{i=1}^{m} a_i.$$

Тарифи $c_{m+1,j}$ також задаються за змістом:

- $c_{m+1,j} = 0$ — якщо «недопоставка» допускається без штрафу (наприклад, перенесення/відкладення попиту поза моделлю);
- $c_{m+1,j} = q_j$ — якщо задається *штраф* за невиконаний попит у пункті B_j (SLA/penalty).

Отримуємо збалансовану задачу розмірності $(m+1) \times n$.

Алгоритми

Методи побудови початкового допустимого базисного розв'язку

Метод північно-західного кута (МПЗК)

1. Розглянути північно-західну невикреслену клітину (i, j) поточної транспортної таблиці.
2. Покласти
$$x_{ij} := \min\{a_i^{(\cdot)}, b_j^{(\cdot)}\},$$
після чого оновити поточні залишки запасів і потреб.
3. Якщо запас $a_i^{(\cdot)}$ вичерпано (дорівнює нулю), а потреба $b_j^{(\cdot)} > 0$, викреслити відповідний рядок. Якщо ж потребу $b_j^{(\cdot)}$ задоволено (дорівнює нулю), а запас $a_i^{(\cdot)} > 0$, викреслити відповідний стовпчик.
4. Якщо одночасно запас $a_i^{(\cdot)}$ вичерпано, а потребу $b_j^{(\cdot)}$ задоволено (обидва дорівнюють нулю), викреслити лише відповідний рядок або стовпчик. Для невикресленого напрямку залишити нульовий залишок. На наступному кроці це приведе до появи необхідного базисного нуля, зберігаючи загальну кількість базисних клітин рівною $m + n - 1$ без утворення циклів.
5. Повторювати кроки 1–4, доки не буде побудовано повний допустимий базисний розв'язок.

Метод мінімального елемента (ММЕ)

1. На кожному кроці серед невикреслених клітин вибрати клітину (i, j) з найменшим тарифом c_{ij}.
2. Покласти
$$x_{ij} := \min\{a_i^{(\cdot)}, b_j^{(\cdot)}\},$$
після чого оновити поточні залишки запасів і потреб.
3. Якщо запас $a_i^{(\cdot)}$ вичерпано (дорівнює нулю), а потреба $b_j^{(\cdot)} > 0$, викреслити відповідний рядок. Якщо ж потребу $b_j^{(\cdot)}$ задоволено (дорівнює нулю), а запас $a_i^{(\cdot)} > 0$, викреслити відповідний стовпчик.
4. У випадку виродженості, коли одночасно запас і потреба стають нульовими, викреслити лише один із них, залишаючи для іншого нульовий залишок на наступний крок. Це забезпечує появу базисного нуля і збереження $m + n - 1$ базисних клітин без потреби окремо перевіряти утворення циклу.

5. Повторювати кроки 1–4, доки не буде побудовано повний допустимий базисний розв'язок.

Метод викреслювання для виявлення циклу

1. Нехай S — множина виділених клітин транспортної таблиці.
2. Викреслити всі рядки, що містять не більш як одну клітину з множини S.
3. Викреслити всі стовпчики, що містять не більш як одну клітину з множини S.
4. Повторювати кроки 2–3 доти, доки можна виконувати нові викреслювання.
5. Якщо врешті-решт викреслено всі виділені клітини, то цикл відсутній.
6. Якщо після завершення процедури залишилися невикреслені клітини, то серед них існує цикл.

Алгоритм методу потенціалів

1. Побудувати початковий допустимий базисний розв'язок X одним із методів побудови початкового плану.
2. Обчислити потенціали u_i $(i = 1, \dots, m)$ і v_j $(j = 1, \dots, n)$, розв'язавши на базисних клітинах систему
$$v_j - u_i = c_{ij},$$
поклавши один із потенціалів рівним нулю.
3. Для всіх небазисних клітин обчислити оцінки
$$\Delta_{ij} = c_{ij} - v_j + u_i.$$
4. Якщо для всіх небазисних клітин виконується нерівність
$$\Delta_{ij} \geq 0,$$
то поточний план є оптимальним.
5. Якщо існує клітина з від'ємною оцінкою, вибрати небазисну клітину (k, ℓ) з найменшою оцінкою $\Delta_{k\ell} < 0$ і приєднати її до базису.
6. На множині базисних клітин разом із клітиною (k, ℓ) побудувати цикл C (зокрема, за допомогою методу викреслювання).
7. Розставити на клітинах циклу почергово знаки $+$ та $-$, починаючи з клітини (k, ℓ), якій надається знак $+$.
8. Покласти
$$\theta = \min\{x_{ij} \colon (i, j) \in C^-\}.$$
Збільшити перевезення в клітинах зі знаком $+$ на θ, а в клітинах зі знаком $-$ зменшити на θ.

9. Клітина, у якій досягається мінімум θ, виходить з базису; у разі кратного мінімуму одну з таких клітин вилучають за узгодженим правилом вибору.

10. Перейти до кроку 2.

Правило вибору в разі неоднозначності

Якщо найменша від'ємна оцінка досягається в кількох небазисних клітинах, доцільно обирати клітину з найменшими індексами (i, j). Якщо мінімум θ досягається в кількох клітинах циклу зі знаком «мінус», застосовують аналогічне узгоджене правило вибору.

Типові задачі

Задача 1.1: Логістика відбудови: розподіл будівельних матеріалів

Для забезпечення проєктів відбудови необхідно налагодити постачання будівельних матеріалів (модульних конструкцій та цементу) з трьох великих логістичних хабів (A_1 — Київський, A_2 — Львівський, A_3 — Дніпровський) до чотирьох регіонів, що найбільше потребують відновлення (B_1 — Харківська обл., B_2 — Чернігівська обл., B_3 — Миколаївська обл., B_4 — Запорізька обл.).

Відомо, що наявні запаси матеріалів на складах хабів (у сотнях тонн) становлять 120, 150 та 130 відповідно. Зі свого боку, затверджені потреби регіональних майданчиків оцінюються у 110, 90, 100 та 100 сотень тонн.

Матриця витрат $C = (c_{ij})$ відображає вартість перевезення одиниці вантажу (у тис. грн) з урахуванням складності маршрутів, вартості пального та стану дорожньої інфраструктури:

$$C = \begin{pmatrix} 4 & 2 & 5 & 6 \\ 9 & 7 & 8 & 10 \\ 3 & 6 & 4 & 2 \end{pmatrix}.$$

Потрібно скласти такий план перевезень, який повністю задовольнить потреби всіх регіонів, вивезе всі наявні запаси з хабів та забезпечить мінімальні сумарні витрати на транспортну логістику.[a]

[a] У розв'язанні виконується формалізація цієї постановки як збалансованої транспортної задачі: вводяться змінні x_{ij}, перевіряється умова балансу, дані подаються у транспортній таблиці та записується відповідна математична модель.

Розв'язання.

Крок 1. Введення змінних. Нехай x_{ij} — обсяг вантажу (у сотнях тонн), який перевозиться з логістичного хабу A_i до регіону відбудови B_j, де $i = 1, 2, 3$, $j = 1, 2, 3, 4$. За фізичним змістом задачі обсяги перевезень не можуть бути від'ємними:

$$x_{ij} \geq 0, \qquad i = 1, 2, 3, \quad j = 1, 2, 3, 4.$$

Крок 2. Перевірка умови балансу. Тут запаси постачальників $a_1 = 120$, $a_2 = 150$,

$a_3 = 130$, а потреби споживачів $b_1 = 110$, $b_2 = 90$, $b_3 = 100$, $b_4 = 100$. Маємо:

$$\sum_{i=1}^{3} a_i = 120 + 150 + 130 = 400, \qquad \sum_{j=1}^{4} b_j = 110 + 90 + 100 + 100 = 400.$$

Оскільки загальний обсяг запасів дорівнює загальному обсягу потреб ($\sum_i a_i = \sum_j b_j$), задача є збалансованою (закритою).

Крок 3. Транспортна таблиця (дані задачі). Вихідні дані задачі подано у стандартній транспортній таблиці (табл. 1.1): у кожній клітинці (i, j) вказано тариф перевезення c_{ij} (у верхній частині) та невідому змінну x_{ij} (у нижній частині), а в останньому стовпчику та останньому рядку наведено відповідно запаси a_i та потреби b_j.

Табл. 1.1: Транспортна таблиця логістики матеріалів

$A_i \setminus B_j$	B_1	B_2	B_3	B_4	a_i
A_1	4 x_{11}	2 x_{12}	5 x_{13}	6 x_{14}	120
A_2	9 x_{21}	7 x_{22}	8 x_{23}	10 x_{24}	150
A_3	3 x_{31}	6 x_{32}	4 x_{33}	2 x_{34}	130
b_j	110	90	100	100	$\sum_j b_j = \sum_i a_i = 400$

Крок 4. Побудова математичної моделі (ЗЛП). Потрібно мінімізувати сумарні транспортні витрати:

$$L(X) = \sum_{i=1}^{3} \sum_{j=1}^{4} c_{ij} x_{ij} \to \min,$$

тобто цільова функція має вигляд:

$$\begin{aligned} L(X) = {} & 4x_{11} + 2x_{12} + 5x_{13} + 6x_{14} \\ & + 9x_{21} + 7x_{22} + 8x_{23} + 10x_{24} \\ & + 3x_{31} + 6x_{32} + 4x_{33} + 2x_{34} \ \to \ \min. \end{aligned}$$

Система обмежень щодо вивезення всіх матеріалів зі складів (запаси):

$$\begin{cases} x_{11} + x_{12} + x_{13} + x_{14} = 120, \\ x_{21} + x_{22} + x_{23} + x_{24} = 150, \\ x_{31} + x_{32} + x_{33} + x_{34} = 130. \end{cases}$$

Система обмежень щодо повного задоволення потреб регіонів відбудови (попит):

$$\begin{cases} x_{11} + x_{21} + x_{31} = 110, \\ x_{12} + x_{22} + x_{32} = 90, \\ x_{13} + x_{23} + x_{33} = 100, \\ x_{14} + x_{24} + x_{34} = 100. \end{cases}$$

Умови невід'ємності змінних:

$$x_{ij} \geq 0, \qquad i = 1,2,3, \quad j = 1,2,3,4.$$

Задача 1.2: Початковий опорний план МПЗК

Для транспортної задачі 1.1 (табл. 1.1) побудувати початковий опорний план методом північно-західного кута.

Розв'язання (МПЗК).

Використовується правило вибору північно-західної невикресленої клітини та стандартне заповнення $x_{ij} = \min\{a_i^{(\cdot)}, b_j^{(\cdot)}\}$.

Крок 1. Клітина $(1,1)$:

$$x_{11} = \min(120, 110) = 110, \qquad a_1 \to 10, \qquad b_1 \to 0 \text{ (викреслюється стовпчик } B_1).$$

Крок 2. Клітина $(1,2)$:

$$x_{12} = \min(10, 90) = 10, \qquad a_1 \to 0 \text{ (викреслюється рядок } A_1), \qquad b_2 \to 80.$$

Крок 3. Клітина $(2,2)$:

$$x_{22} = \min(150, 80) = 80, \qquad a_2 \to 70, \qquad b_2 \to 0 \text{ (викреслюється стовпчик } B_2).$$

Крок 4. Клітина $(2,3)$:

$$x_{23} = \min(70, 100) = 70, \qquad a_2 \to 0 \text{ (викреслюється рядок } A_2), \qquad b_3 \to 30.$$

Крок 5. Клітина $(3,3)$:

$$x_{33} = \min(130, 30) = 30, \qquad a_3 \to 100, \qquad b_3 \to 0 \text{ (викреслюється стовпчик } B_3).$$

Крок 6. Клітина $(3,4)$:

$$x_{34} = \min(100, 100) = 100, \qquad a_3 \to 0, \qquad b_4 \to 0.$$

(Викреслюється останній рядок і стовпчик). У даному випадку виродженості на проміжних етапах не виникло, тому базисні нулі не додаються.

Отримано початковий опорний план з базисними клітинами

$$(1,1),\ (1,2),\ (2,2),\ (2,3),\ (3,3),\ (3,4),$$

їх рівно $m + n - 1 = 3 + 4 - 1 = 6$.

План перевезень (матриця X).

$$X = \begin{pmatrix} 110 & 10 & 0 & 0 \\ 0 & 80 & 70 & 0 \\ 0 & 0 & 30 & 100 \end{pmatrix}.$$

Транспортну таблицю початкового плану (МПЗК) подано в таблиці 1.2.

Табл. 1.2: Початковий опорний план МПЗК для прикладу 1.1

$A_i \setminus B_j$	B_1	B_2	B_3	B_4	a_i
A_1	4 **110**	2 **10**	5	6	120
A_2	9	7 **80**	8 **70**	10	150
A_3	3	6	4 **30**	2 **100**	130
b_j	110	90	100	100	400

Вартість початкового плану.

$$L(X) = 4\cdot 110 + 2\cdot 10 + 7\cdot 80 + 8\cdot 70 + 4\cdot 30 + 2\cdot 100 = 440 + 20 + 560 + 560 + 120 + 200 = 1900.$$

Задача 1.3: Початковий опорний план ММЕ

Для транспортної задачі з прикладу 1.1 (табл. 1.1) побудувати початковий опорний план методом мінімального елемента.

Розв'язання.

На кожному кроці серед *невикреслених* клітин обирається клітина з мінімальним тарифом c_{ij} (за нічиєї використовується тай-брейк: спочатку максимізується обсяг перевезення $\min\{a_i^{(\cdot)}, b_j^{(\cdot)}\}$, далі — обираються найменші індекси). Після вибору клітини застосовується стандартне правило заповнення $x_{ij} = \min\{a_i^{(\cdot)}, b_j^{(\cdot)}\}$.

Крок 1. Мінімальний тариф 2 знаходиться у клітинах $(1,2)$ та $(3,4)$. За тай-брейком перевіряємо можливі обсяги: для $(1,2)$ маємо $\min\{120, 90\} = 90$, а для $(3,4)$ маємо $\min\{130, 100\} = 100$. Обираємо клітину $(3,4)$, оскільки вона дозволяє перевезти більше вантажу.

$$x_{34} = \min(130, 100) = 100, \qquad a_3 \to 30, \qquad b_4 \to 0 \text{ (викреслюється стовпчик } B_4).$$

Крок 2. Серед невикреслених клітин мінімальний тариф 2 залишився у клітині $(1,2)$.

$$x_{12} = \min(120, 90) = 90, \qquad a_1 \to 30, \qquad b_2 \to 0 \text{ (викреслюється стовпчик } B_2).$$

Крок 3. Серед залишених клітин у стовпчиках B_1 та B_3 мінімальним є тариф 3 у клітині $(3,1)$.

$$x_{31} = \min(30, 110) = 30, \qquad a_3 \to 0 \text{ (викреслюється рядок } A_3), \qquad b_1 \to 80.$$

Крок 4. Залишилися невикресленими лише рядки A_1 та A_2. Мінімальний тариф серед доступних клітин — 4 у клітині $(1,1)$.

$$x_{11} = \min(30, 80) = 30, \qquad a_1 \to 0 \text{ (викреслюється рядок } A_1), \qquad b_1 \to 50.$$

Крок 5. Залишився лише рядок A_2 (залишок 150) і два неповністю задоволені стовпчики: B_1 (залишок 50) та B_3 (потреба 100). Мінімальний тариф серед них — 8 у клітині $(2,3)$.

$$x_{23} = \min(150, 100) = 100, \qquad a_2 \to 50, \qquad b_3 \to 0 \text{ (викреслюється стовпчик } B_3).$$

Крок 6. Залишилася остання клітина $(2,1)$ із тарифом 9:

$$x_{21} = \min(50, 50) = 50, \qquad a_2 \to 0, \qquad b_1 \to 0.$$

Отримано допустимий план перевезень. Кількість заповнених базисних клітин дорівнює $6 = m + n - 1$ $(3 + 4 - 1 = 6)$, отже план є невиродженим опорним.

План перевезень (матриця X).

$$X = \begin{pmatrix} 30 & 90 & 0 & 0 \\ 50 & 0 & 100 & 0 \\ 30 & 0 & 0 & 100 \end{pmatrix}.$$

Транспортну таблицю початкового плану (ММЕ) подано в таблиці 1.3.

Табл. 1.3: Початковий опорний план ММЕ для прикладу 1.1

$A_i \setminus B_j$	B_1	B_2	B_3	B_4	a_i
A_1	4 **30**	2 **90**	5	6	120
A_2	9 **50**	7	8 **100**	10	150
A_3	3 **30**	6	4	2 **100**	130
b_j	110	90	100	100	400

Вартість початкового плану.

$$L(X) = 4{\cdot}30 + 2{\cdot}90 + 9{\cdot}50 + 8{\cdot}100 + 3{\cdot}30 + 2{\cdot}100 = 120 + 180 + 450 + 800 + 90 + 200 = 1840.$$

Як бачимо, застосування методу мінімального елемента дозволило знайти стартовий план, вартість якого (1840) є меншою за вартість плану, побудованого методом північно-західного кута (1900).

Задача 1.4: Перевірка плану на оптимальність та альтернативний план

Для опорного плану, знайденого методом мінімального елемента у прикладі 1.3, перевірити умову оптимальності за допомогою методу потенціалів та знайти альтернативний оптимальний план.

Розв'язання.

Згідно з методом потенціалів, кожному пункту відправлення A_i та пункту призначення B_j ставиться у відповідність потенціал (u_i та v_j відповідно). Для того щоб опорний план був оптимальним, необхідно і достатньо, щоб для всіх *базисних* (заповнених) клітин ($x_{ij} > 0$) виконувалася умова:

$$v_j - u_i = c_{ij}, \tag{2}$$

а для всіх *небазисних* (порожніх) клітин ($x_{ij} = 0$) виконувалася умова:

$$v_j - u_i \le c_{ij}. \tag{3}$$

Крок 1. Обчислення потенціалів. Базисними клітинами нашого плану є

$$(1,1), (1,2), (2,1), (2,3), (3,1), (3,4).$$

Запишемо для них систему рівнянь згідно з (2):

$$\begin{cases} v_1 - u_1 = c_{11} = 4, \\ v_2 - u_1 = c_{12} = 2, \\ v_1 - u_2 = c_{21} = 9, \\ v_3 - u_2 = c_{23} = 8, \\ v_1 - u_3 = c_{31} = 3, \\ v_4 - u_3 = c_{34} = 2. \end{cases}$$

Оскільки система має $m + n = 7$ невідомих і $m + n - 1 = 6$ рівнянь, один із потенціалів можна вибрати довільно. Нехай $u_1 = 0$. Тоді послідовно знаходимо решту потенціалів:

- З першого рівняння: $v_1 - 0 = 4 \implies v_1 = 4$.
- З другого рівняння: $v_2 - 0 = 2 \implies v_2 = 2$.
- Знаючи v_1, з третього рівняння: $4 - u_2 = 9 \implies u_2 = -5$.
- Знаючи u_2, з четвертого рівняння: $v_3 - (-5) = 8 \implies v_3 + 5 = 8 \implies v_3 = 3$.
- Знаючи v_1, з п'ятого рівняння: $4 - u_3 = 3 \implies u_3 = 1$.
- Знаючи u_3, з шостого рівняння: $v_4 - 1 = 2 \implies v_4 = 3$.

Отже, маємо вектори потенціалів:

$$u = (0, -5, 1), \qquad v = (4, 2, 3, 3).$$

Крок 2. Перевірка критерію оптимальності. Для кожної порожньої (небазисної) клітини перевіримо виконання умови (3), тобто $v_j - u_i \le c_{ij}$:

- Для клітини $(1,3)$: $v_3 - u_1 = 3 - 0 = 3 \le 5$ (умова виконується);
- Для клітини $(1,4)$: $v_4 - u_1 = 3 - 0 = 3 \le 6$ (умова виконується);
- Для клітини $(2,2)$: $v_2 - u_2 = 2 - (-5) = 7 \le 7$ (умова виконується **як рівність**);

- Для клітини $(2,4)$: $v_4 - u_2 = 3 - (-5) = 8 \leq 10$ (умова виконується);
- Для клітини $(3,2)$: $v_2 - u_3 = 2 - 1 = 1 \leq 6$ (умова виконується);
- Для клітини $(3,3)$: $v_3 - u_3 = 3 - 1 = 2 \leq 4$ (умова виконується).

Оскільки для всіх небазисних клітин умова $v_j - u_i \leq c_{ij}$ виконується, то поточний план перевезень є **оптимальним** із мінімальними витратами $L_{min} = 1840$ тис. грн. Проте виконання умови для клітини $(2,2)$ як строгої рівності вказує на наявність *альтернативного оптимального плану.*

Крок 3. Побудова альтернативного оптимального плану. Для отримання іншого розподілу вантажу без зміни сумарної вартості побудуємо цикл перерахунку для клітини $(2,2)$. Цикл складатиметься з порожньої клітини $(2,2)$ та базисних клітин:

$$(2,2)^+ \to (1,2)^- \to (1,1)^+ \to (2,1)^- \to (2,2)^+$$

Знайдемо величину зсуву вантажу по циклу θ, яка дорівнює мінімальному значенню серед клітин зі знаком мінус:

$$\theta = \min\{x_{12}, x_{21}\} = \min\{90, 50\} = 50.$$

Перерахуємо обсяги перевезень у вершинах циклу:

- $x'_{22} = x_{22} + \theta = 0 + 50 = 50$ (клітина стає базисною);
- $x'_{12} = x_{12} - \theta = 90 - 50 = 40$;
- $x'_{11} = x_{11} + \theta = 30 + 50 = 80$;
- $x'_{21} = x_{21} - \theta = 50 - 50 = 0$ (клітина стає порожньою).

Інші базисні клітини $(2,3), (3,1), (3,4)$ залишаються без змін.

Альтернативний план перевезень (матриця X'):

$$X' = \begin{pmatrix} 80 & 40 & 0 & 0 \\ 0 & 50 & 100 & 0 \\ 30 & 0 & 0 & 100 \end{pmatrix}.$$

Переконаємося, що вартість логістики для альтернативного плану не змінилася:

$$\begin{aligned} L(X') &= 4 \cdot 80 + 2 \cdot 40 + 7 \cdot 50 + 8 \cdot 100 + 3 \cdot 30 + 2 \cdot 100 \\ &= 320 + 80 + 350 + 800 + 90 + 200 = 1840. \end{aligned}$$

Обидва плани забезпечують мінімум витрат, що дає керівництву гнучкість у прийнятті логістичних рішень (наприклад, можливість відмовитись від маршруту $A_2 \to B_1$ на користь $A_2 \to B_2$).

Задача 1.5: Маршрутизація IT-трафіку

Хмарний провайдер має три дата-центри (A_1, A_2, A_3), виділена пропускна здатність яких становить 100, 80 та 60 Тбіт/с відповідно. Трафік необхідно маршрутизувати до трьох клієнтських регіонів (B_1, B_2, B_3), чия потреба у трафіку становить 90, 110

та 70 Тбіт/с.

Матриця C відображає затримку передачі даних (у мілісекундах) між вузлами:

$$C = \begin{pmatrix} 6 & 3 & 5 \\ 4 & 8 & 7 \\ 5 & 2 & 9 \end{pmatrix}.$$

Потрібно збалансувати задачу, знайти початковий план маршрутизації методом мінімального елемента та перевірити його на оптимальність методом потенціалів.

Розв'язання.

Крок 1. Перевірка балансу та зведення до закритої ТЗ. Знайдемо сумарну пропускну здатність дата-центрів та сумарний попит регіонів:

$$\sum_{i=1}^{3} a_i = 100 + 80 + 60 = 240 \text{ Тбіт/с}, \qquad \sum_{j=1}^{3} b_j = 90 + 110 + 70 = 270 \text{ Тбіт/с}.$$

Оскільки $\sum a_i < \sum b_j$ $(240 < 270)$, задача є **відкритою**. Мережа не може повністю задовольнити попит.

Для зведення задачі до закритої вводимо *фіктивний дата-центр* A_4 із запасом $a_4 = 270 - 240 = 30$ Тбіт/с. Тарифи на перевезення від фіктивного постачальника дорівнюють нулю ($c_{4j} = 0$), оскільки фізично цей трафік не передається (це обсяг "відмовленого"або відкладеного обслуговування).

Крок 2. Побудова початкового плану ММЕ. Складаємо нову матрицю розміром 4×3. На кожному кроці обираємо невикреслену клітину з мінімальним тарифом c_{ij} і заповнюємо її $x_{ij} = \min\{a_i, b_j\}$.

- Найменший тариф у таблиці — 0 у рядку A_4. Обираємо клітину $(4,1)$ (за правилом найменшого індексу): $x_{41} = \min(30, 90) = 30$. Рядок A_4 викреслюється, залишок попиту B_1 стає 60.
- Серед невикреслених мінімальний тариф 2 у $(3,2)$: $x_{32} = \min(60, 110) = 60$. Рядок A_3 викреслюється, залишок $B_2 \to 50$.
- Наступний мінімум 3 у $(1,2)$: $x_{12} = \min(100, 50) = 50$. Стовпчик B_2 викреслюється, залишок $A_1 \to 50$.
- Наступний мінімум 4 у $(2,1)$: $x_{21} = \min(80, 60) = 60$. Стовпчик B_1 викреслюється, залишок $A_2 \to 20$.
- Залишився стовпчик B_3. Для $(1,3)$ маємо $c_{13} = 5$: $x_{13} = \min(50, 70) = 50$. A_1 викреслюється, залишок $B_3 \to 20$.
- Остання клітина $(2,3)$ заповнюється залишками: $x_{23} = \min(20, 20) = 20$.

Кількість базисних клітин: $4 + 3 - 1 = 6$. План невироджений.

Транспортну таблицю початкового плану (ММЕ) подано в таблиці 1.4.

Табл. 1.4: Початковий опорний план ММЕ для задачі маршрутизації

$A_i \setminus B_j$	B_1	B_2	B_3	a_i
A_1	6	3 **50**	5 **50**	100
A_2	4 **60**	8	7 **20**	80
A_3	5	2 **60**	9	60
A_4 (фікт.)	0 **30**	0	0	30
b_j	90	110	70	270

Загальна затримка мережі (вартість):

$$L(X) = 50 \cdot 3 + 50 \cdot 5 + 60 \cdot 4 + 20 \cdot 7 + 60 \cdot 2 + 30 \cdot 0 = 900.$$

Крок 3. Перевірка плану на оптимальність. Згідно з методом потенціалів, для всіх базисних клітин ($x_{ij} > 0$) має виконуватися $v_j - u_i = c_{ij}$, а для небазисних ($x_{ij} = 0$) умова $v_j - u_i \le c_{ij}$.

Складемо систему для базисних клітин $(1,2), (1,3), (2,1), (2,3), (3,2), (4,1)$:

$$\begin{cases} v_2 - u_1 = 3 \\ v_3 - u_1 = 5 \\ v_1 - u_2 = 4 \\ v_3 - u_2 = 7 \\ v_2 - u_3 = 2 \\ v_1 - u_4 = 0 \end{cases}$$

Нехай $u_1 = 0$. Тоді: $v_2 = 3$, $v_3 = 5$. З рівняння $v_3 - u_2 = 7 \implies 5 - u_2 = 7 \implies u_2 = -2$. Знаючи u_2, знаходимо v_1: $v_1 - (-2) = 4 \implies v_1 = 2$. Знаючи v_2, знаходимо u_3: $3 - u_3 = 2 \implies u_3 = 1$. Знаючи v_1, знаходимо u_4: $2 - u_4 = 0 \implies u_4 = 2$. Вектори потенціалів, обчислені для транспортної таблиці в табл. 1.5, мають вигляд: $u = (0, -2, 1, 2)$, $v = (2, 3, 5)$.

Табл. 1.5: Перевірка опорного плану на оптимальність (значення потенціалів)

$A_i \setminus B_j$	B_1	B_2	B_3	a_i	u_i
A_1	6	3 **50**	5 **50**	100	0
A_2	4 **60**	8	7 **20**	80	-2
A_3	5	2 **60**	9	60	1
A_4 (фікт.)	0 **30**	0	0	30	2
b_j	90	110	70	270	
v_j	2	3	5		

Крок 3. Перевірка плану на оптимальність. Для перевірки критерію оптимальності обчислимо *симплекс-різниці* для всіх порожніх (небазисних) клітин за формулою:

$$\Delta_{ij} = c_{ij} - v_j + u_i.$$

Для того щоб план був оптимальним, необхідно, щоб для всіх порожніх клітин виконувалася умова $\Delta_{ij} \geq 0$. Обчислюємо:

- $\Delta_{11} = c_{11} - v_1 + u_1 = 6 - 2 + 0 = 4 \geq 0$ (умова виконується);
- $\Delta_{22} = c_{22} - v_2 + u_2 = 8 - 3 + (-2) = 3 \geq 0$ (умова виконується);
- $\Delta_{31} = c_{31} - v_1 + u_3 = 5 - 2 + 1 = 4 \geq 0$ (умова виконується);
- $\Delta_{33} = c_{33} - v_3 + u_3 = 9 - 5 + 1 = 5 \geq 0$ (умова виконується);
- $\Delta_{42} = c_{42} - v_2 + u_4 = 0 - 3 + 2 = -1 < 0$ (**порушення!**);
- $\Delta_{43} = c_{43} - v_3 + u_4 = 0 - 5 + 2 = -3 < 0$ (**порушення!**).

Оскільки існують клітини з від'ємними симплекс-різницями, **початковий план не є оптимальним**. Найбільше порушення (найменша від'ємна оцінка) спостерігається у клітині $(4, 3)$, де $\Delta_{43} = -3$. Саме цю клітину ми оберемо для введення в базис на наступному кроці (для побудови коригувального циклу).

Табл. 1.6: Перевірка плану на оптимальність: симплекс-різниці Δ_{ij}

$A_i \setminus B_j$	B_1	B_2	B_3	a_i	u_i
A_1	6 (4)	3 **50**	5 **50**	100	0
A_2	4 **60**	8 (3)	7 **20**	80	-2
A_3	5 (4)	2 **60**	9 (5)	60	1
A_4 (фікт.)	0 **30**	0 **(-1)**	0 **(-3)**	30	2
b_j	90	110	70	270	
v_j	2	3	5		

Примітка: У порожніх клітинах у круглих дужках вказані симплекс-різниці Δ_{ij}. Від'ємні значення, що вказують на можливість покращення плану, виділено жирним шрифтом. Максимальне порушення умови оптимальності (найменше значення) знаходиться у клітині $(4,3)$, де $\Delta_{43} = -3$ (див. таблицю 1.6).

Крок 4. Покращення плану (коригувальний цикл). Побудуємо цикл для перспективної клітини $(4,3)$, використовуючи наявні базисні клітини:

$$(4,3)^+ \to (2,3)^- \to (2,1)^+ \to (4,1)^- \to (4,3)^+$$

Величина зсуву трафіку θ:

$$\theta = \min\{x_{23}, x_{41}\} = \min\{20, 30\} = 20.$$

Перераховуємо значення у вершинах циклу:

- $x'_{43} = 0 + 20 = 20$ (стає базисною);
- $x'_{23} = 20 - 20 = 0$ (стає порожньою);
- $x'_{21} = 60 + 20 = 80$;
- $x'_{41} = 30 - 20 = 10$.

Оптимальний план перевезень (X_{opt}):

$$X_{opt} = \begin{pmatrix} 0 & 50 & 50 \\ 80 & 0 & 0 \\ 0 & 60 & 0 \\ 10 & 0 & 20 \end{pmatrix}.$$

Загальна мінімальна затримка (вартість) нової маршрутизації:

$$L(X_{opt}) = 50(3) + 50(5) + 80(4) + 60(2) + 10(0) + 20(0) = 840.$$

Відповідь: Ми зменшили загальну «вартість» затримок з 900 до 840, оптимізувавши розподіл дефіциту. Фіктивні перевезення $x_{41} = 10$ та $x_{43} = 20$ означають, що регіон B_1 недоотримає 10 Тбіт/с, а регіон B_3 — 20 Тбіт/с від своїх запитів.

Задачі для самостійного розв'язування

Задача 1.1: Транспортна задача: МПЗК, ММЕ та метод потенціалів

Кавова компанія має три ростерії (пункти обсмажування кави) A_1, A_2, A_3, з яких свіжообсмажене зерно постачається до чотирьох мереж кав'ярень B_1, B_2, B_3, B_4.

Для кожного варіанта задано:

- вектор запасів зерна на ростеріях $a = (a_1, a_2, a_3)$ у сотнях кілограмів;
- вектор потреб мереж кав'ярень $b = (b_1, b_2, b_3, b_4)$ у сотнях кілограмів;
- матрицю C, елементи якої c_{ij} позначають вартість перевезення одиниці вантажу (100 кг) від i-ї ростерії до j-ї мережі кав'ярень у грошових одиницях.

Потрібно:

1. Побудувати математичну модель транспортної задачі.
2. Знайти початковий опорний план методом північно-західного кута (МПЗК).
3. Знайти початковий опорний план методом мінімального елемента (ММЕ).
4. Взявши за основу кращий із двох знайдених початкових планів, розв'язати задачу методом потенціалів (знайти оптимальний план перевезень та мінімальну вартість логістики).

Варіанти:

1.1 Вектор запасів $a = (100, 150, 50)$, вектор потреб $b = (70, 80, 60, 90)$. Матриця вартостей:

$$C = \begin{pmatrix} 5 & 3 & 6 & 2 \\ 4 & 7 & 3 & 5 \\ 8 & 2 & 4 & 6 \end{pmatrix}.$$

1.2 Вектор запасів $a = (120, 80, 100)$, вектор потреб $b = (50, 90, 70, 90)$. Матриця вартостей:

$$C = \begin{pmatrix} 4 & 6 & 3 & 8 \\ 2 & 5 & 7 & 4 \\ 6 & 4 & 2 & 5 \end{pmatrix}.$$

1.3 Вектор запасів $a = (90, 110, 150)$, вектор потреб $b = (100, 60, 120, 70)$. Матриця вартостей:

$$C = \begin{pmatrix} 7 & 3 & 5 & 6 \\ 4 & 2 & 8 & 3 \\ 5 & 6 & 4 & 7 \end{pmatrix}.$$

1.4 Вектор запасів $a = (200, 100, 100)$, вектор потреб $b = (80, 120, 90, 110)$. Матриця вартостей:

$$C = \begin{pmatrix} 3 & 8 & 4 & 5 \\ 6 & 2 & 7 & 4 \\ 4 & 5 & 3 & 8 \end{pmatrix}.$$

1.5 Вектор запасів $a = (130, 170, 100)$, вектор потреб $b = (110, 90, 100, 100)$. Матриця вартостей:

$$C = \begin{pmatrix} 8 & 4 & 6 & 3 \\ 5 & 7 & 2 & 4 \\ 3 & 5 & 8 & 6 \end{pmatrix}.$$

1.6 Вектор запасів $a = (150, 150, 200)$, вектор потреб $b = (100, 140, 160, 100)$. Матриця вартостей:

$$C = \begin{pmatrix} 2 & 5 & 4 & 7 \\ 8 & 3 & 6 & 5 \\ 4 & 7 & 2 & 3 \end{pmatrix}.$$

1.7 Вектор запасів $a = (80, 120, 100)$, вектор потреб $b = (60, 70, 80, 90)$. Матриця вартостей:

$$C = \begin{pmatrix} 6 & 2 & 5 & 4 \\ 3 & 8 & 4 & 7 \\ 5 & 4 & 7 & 2 \end{pmatrix}.$$

1.8 Вектор запасів $a = (140, 160, 100)$, вектор потреб $b = (90, 110, 120, 80)$. Матриця вартостей:

$$C = \begin{pmatrix} 5 & 7 & 3 & 6 \\ 4 & 2 & 8 & 5 \\ 6 & 4 & 5 & 3 \end{pmatrix}.$$

1.9 Вектор запасів $a = (180, 120, 150)$, вектор потреб $b = (130, 100, 140, 80)$. Матриця вартостей:

$$C = \begin{pmatrix} 4 & 8 & 5 & 2 \\ 7 & 3 & 6 & 4 \\ 2 & 6 & 4 & 8 \end{pmatrix}.$$

1.10 Вектор запасів $a = (110, 190, 100)$, вектор потреб $b = (80, 120, 100, 100)$. Матриця вартостей:

$$C = \begin{pmatrix} 3 & 5 & 8 & 4 \\ 6 & 4 & 2 & 7 \\ 5 & 7 & 4 & 3 \end{pmatrix}.$$

1.11 Вектор запасів $a = (210, 140, 150)$, вектор потреб $b = (160, 100, 120, 120)$. Матриця вартостей:

$$C = \begin{pmatrix} 7 & 4 & 6 & 5 \\ 2 & 8 & 3 & 4 \\ 5 & 3 & 7 & 2 \end{pmatrix}.$$

1.12 Вектор запасів $a = (160, 140, 200)$, вектор потреб $b = (130, 170, 90, 110)$. Матриця вартостей:

$$C = \begin{pmatrix} 4 & 2 & 7 & 6 \\ 5 & 8 & 4 & 3 \\ 6 & 5 & 3 & 8 \end{pmatrix}.$$

1.13 Вектор запасів $a = (100, 200, 150)$, вектор потреб $b = (140, 110, 100, 100)$. Матриця вартостей:

$$C = \begin{pmatrix} 8 & 5 & 3 & 4 \\ 4 & 6 & 7 & 2 \\ 3 & 4 & 5 & 8 \end{pmatrix}.$$

1.14 Вектор запасів $a = (170, 130, 200)$, вектор потреб $b = (150, 150, 100, 100)$. Матриця вартостей:

$$C = \begin{pmatrix} 5 & 4 & 8 & 3 \\ 7 & 2 & 4 & 6 \\ 3 & 6 & 5 & 4 \end{pmatrix}.$$

1.15 Вектор запасів $a = (190, 210, 100)$, вектор потреб $b = (120, 180, 90, 110)$. Матриця вартостей:

$$C = \begin{pmatrix} 6 & 3 & 5 & 8 \\ 4 & 7 & 2 & 5 \\ 8 & 4 & 6 & 3 \end{pmatrix}.$$

Задача 1.2: Відкрита транспортна задача: зведення та метод потенціалів

Мережа крафтових сироварень має три виробничі локації A_1, A_2, A_3, які постачають елітні сорти сиру до чотирьох мереж винних бутиків B_1, B_2, B_3, B_4.

Для кожного варіанта задано:

- вектор пропозиції сиру на сироварнях $a = (a_1, a_2, a_3)$ у десятках кілограмів;
- вектор попиту винних бутиків $b = (b_1, b_2, b_3, b_4)$ у десятках кілограмів;
- матрицю C, елементи якої c_{ij} позначають вартість транспортування одиниці продукції від i-ї сироварні до j-го бутика.

Потрібно:

1. Перевірити задачу на збалансованість (відкрита чи закрита модель).
2. Звести задачу до закритого типу, ввівши фіктивного постачальника або фіктивного споживача (з нульовими тарифами на перевезення).
3. Знайти початковий опорний план.
4. Перевірити знайдений план на оптимальність та за потреби знайти оптимальний план розвезення сиру методом потенціалів.

Варіанти:

2.1 Вектор запасів $a = (120, 100, 80)$, вектор попиту $b = (70, 90, 80, 100)$. Матриця вартостей:

$$C = \begin{pmatrix} 4 & 7 & 2 & 5 \\ 6 & 3 & 8 & 4 \\ 5 & 9 & 3 & 6 \end{pmatrix}.$$

2.2 Вектор запасів $a = (150, 110, 90)$, вектор попиту $b = (60, 80, 70, 90)$. Матриця вартостей:

$$C = \begin{pmatrix} 8 & 3 & 5 & 2 \\ 4 & 6 & 7 & 9 \\ 3 & 5 & 2 & 8 \end{pmatrix}.$$

2.3 Вектор запасів $a = (80, 130, 140)$, вектор попиту $b = (100, 110, 120, 60)$. Матриця вартостей:

$$C = \begin{pmatrix} 5 & 2 & 8 & 4 \\ 7 & 6 & 3 & 9 \\ 4 & 5 & 7 & 2 \end{pmatrix}.$$

2.4 Вектор запасів $a = (200, 150, 100)$, вектор попиту $b = (90, 100, 110, 100)$. Матриця вартостей:

$$C = \begin{pmatrix} 6 & 4 & 3 & 7 \\ 2 & 8 & 5 & 4 \\ 9 & 3 & 6 & 5 \end{pmatrix}.$$

2.5 Вектор запасів $a = (110, 140, 90)$, вектор попиту $b = (100, 120, 80, 90)$. Матриця вартостей:

$$C = \begin{pmatrix} 3 & 8 & 6 & 2 \\ 5 & 4 & 7 & 3 \\ 8 & 2 & 5 & 6 \end{pmatrix}.$$

2.6 Вектор запасів $a = (160, 120, 140)$, вектор попиту $b = (80, 90, 100, 110)$. Матриця вартостей:

$$C = \begin{pmatrix} 7 & 5 & 2 & 8 \\ 3 & 9 & 6 & 4 \\ 5 & 2 & 8 & 3 \end{pmatrix}.$$

2.7 Вектор запасів $a = (90, 100, 120)$, вектор попиту $b = (110, 80, 90, 70)$. Матриця вартостей:

$$C = \begin{pmatrix} 4 & 6 & 8 & 3 \\ 7 & 2 & 5 & 6 \\ 3 & 8 & 4 & 5 \end{pmatrix}.$$

2.8 Вектор запасів $a = (130, 180, 110)$, вектор попиту $b = (100, 90, 80, 100)$. Матриця вартостей:

$$C = \begin{pmatrix} 9 & 3 & 6 & 5 \\ 4 & 8 & 2 & 7 \\ 5 & 4 & 7 & 3 \end{pmatrix}.$$

2.9 Вектор запасів $a = (100, 150, 80)$, вектор попиту $b = (90, 110, 120, 50)$. Матриця вартостей:

$$C = \begin{pmatrix} 2 & 7 & 5 & 8 \\ 6 & 3 & 9 & 4 \\ 8 & 5 & 2 & 6 \end{pmatrix}.$$

2.10 Вектор запасів $a = (170, 130, 150)$, вектор попиту $b = (110, 100, 120, 80)$. Матриця вартостей:

$$C = \begin{pmatrix} 5 & 8 & 4 & 3 \\ 3 & 6 & 7 & 2 \\ 9 & 4 & 5 & 8 \end{pmatrix}.$$

2.11 Вектор запасів $a = (80, 110, 90)$, вектор попиту $b = (70, 100, 90, 60)$. Матриця вартостей:

$$C = \begin{pmatrix} 6 & 2 & 8 & 5 \\ 4 & 7 & 3 & 9 \\ 5 & 4 & 6 & 2 \end{pmatrix}.$$

2.12 Вектор запасів $a = (140, 160, 120)$, вектор попиту $b = (80, 110, 100, 90)$. Матриця вартостей:

$$C = \begin{pmatrix} 3 & 9 & 5 & 4 \\ 8 & 2 & 6 & 7 \\ 4 & 5 & 8 & 3 \end{pmatrix}.$$

2.13 Вектор запасів $a = (90, 120, 100)$, вектор попиту $b = (100, 80, 90, 80)$. Матриця вартостей:

$$C = \begin{pmatrix} 7 & 4 & 3 & 8 \\ 2 & 6 & 9 & 5 \\ 5 & 8 & 4 & 6 \end{pmatrix}.$$

2.14 Вектор запасів $a = (180, 140, 160)$, вектор попиту $b = (120, 110, 100, 90)$. Матриця вартостей:

$$C = \begin{pmatrix} 4 & 5 & 8 & 2 \\ 9 & 3 & 6 & 7 \\ 3 & 8 & 5 & 4 \end{pmatrix}.$$

2.15 Вектор запасів $a = (100, 130, 110)$, вектор попиту $b = (90, 120, 100, 80)$. Матриця вартостей:

$$C = \begin{pmatrix} 8 & 2 & 7 & 5 \\ 4 & 9 & 3 & 6 \\ 6 & 5 & 8 & 2 \end{pmatrix}.$$

Прикладні задачі

Задача (IT) 1.1: CDN: розподіл трафіку між PoP-вузлами

IT-коментар

У глобальних CDN[a]-мережах транспортна задача дає змогу моделювати розподіл трафіку між вузлами присутності та регіональними зонами попиту так, щоб мінімізувати сумарні витрати на транзит і затримки обслуговування. Така модель дає змогу оцінити, як раціонально розподілити навантаження між PoP-вузлами в умовах нерівномірного регіонального попиту. Отриманий оптимальний план показує, які вузли мережі доцільно використовувати як основні точки обслуговування для різних регіонів.

[a] CDN (*Content Delivery Network*) — мережа доставлення вмісту, тобто розподілена система серверів для швидкого надання цифрового контенту користувачам.

Глобальна CDN-мережа має $m = 4$ вузли присутності PoP[a] A_i з доступними добовими пропускними здатностями a_i (ТБ/добу) та $n = 5$ регіональних зон попиту B_j з потребами b_j. Вартість c_{ij} інтерпретується як середня «ціна» обслуговування 1 ТБ трафіку з PoP A_i у зоні B_j (комбінація витрат на транзит і штрафу за затримку). Дано вектори запасів, потреб та матрицю вартостей C:

$$a = (500, 700, 400, 600), \quad b = (400, 500, 600, 450, 350)$$

$$C = \begin{pmatrix} 12 & 25 & 10 & 18 & 30 \\ 20 & 14 & 22 & 15 & 25 \\ 15 & 30 & 12 & 20 & 18 \\ 28 & 16 & 25 & 14 & 22 \end{pmatrix}$$

1. Перевірте умову балансу.
2. Зведіть задачу до збалансованої, доповнивши її фіктивним PoP з нульовими тарифами $c_{5j} = 0$.
3. Побудуйте початковий опорний план методом мінімального елемента (ММЕ).
4. Знайдіть оптимальний план методом потенціалів.
5. Обчисліть мінімальні сумарні витрати та інтерпретуйте результат з погляду ефективного розподілу CDN-трафіку між вузлами мережі.

[a] PoP (*Point of Presence*) — вузол присутності мережевої інфраструктури в певному регіоні, через який надають мережеві сервіси користувачам.

Задача (IT) 1.2: Хмарні ресурси: розміщення задач у дата-центрах

IT-коментар

У хмарних інфраструктурах транспортна задача використовується для моделювання розміщення задач між дата-центрами з урахуванням обмежень на ресурси, вартості запуску та вимог SLA[a]. Вона дає змогу перейти від інтуїтивного розподілу навантаження до формально оптимального плану, який мінімізує операційні витрати. Аналіз такого плану також допомагає виявити, які центри обробки даних є перевантаженими, а які мають резерв потужності.

[a] SLA (*Service Level Agreement*) — угода про рівень сервісу, яка визначає гарантовані параметри якості обслуговування.

Є 4 дата-центри з доступними обчислювальними ресурсами a_i (тис. ядер) та 5 пулів задач із запитами b_j. Тариф c_{ij} — середня вартість запуску одиниці задачі B_j у дата-центрі A_i. Надзвичайно високі значення у матриці C (наприклад, 99) означають порушення SLA, несумісність архітектури або географічні обмеження.

$$a = (120, 180, 100, 200), \quad b = (100, 150, 150, 80, 120)$$

$$C = \begin{pmatrix} 5 & 8 & 15 & 6 & 10 \\ 12 & 4 & 9 & 99 & 8 \\ 10 & 15 & 6 & 8 & 12 \\ 99 & 10 & 12 & 5 & 7 \end{pmatrix}$$

1. Запишіть модель як збалансовану транспортну задачу.
2. Поясніть, чому клітини з $c_{ij} = 99$ найімовірніше не увійдуть до оптимального плану.
3. Побудуйте початковий опорний план за допомогою методу північно-західного кута (МПЗК).

4. Знайдіть оптимальний план методом потенціалів.

5. Обчисліть мінімальні сумарні витрати та інтерпретуйте отриманий план як схему розміщення задач у хмарній інфраструктурі.

Задача (IT) 1.3: MLOps: розподіл навчальних запусків на GPU-кластери

IT-коментар

У MLOps[а] транспортна задача дає змогу розподіляти навчальні експерименти між GPU[б]-кластерами так, щоб мінімізувати сумарну вартість обчислень і накладні витрати на передавання даних. Це особливо важливо для великих ML-проєктів, де різниця в тарифах і доступності обчислювальних ресурсів істотно впливає на загальний бюджет. Оптимальний план дає змогу побачити, які кластери слід використовувати як основні, а які — лише як допоміжні.

[а] MLOps (*Machine Learning Operations*) — сукупність практик організації, автоматизації та супроводу життєвого циклу моделей машинного навчання.

[б] GPU (*Graphics Processing Unit*) — графічний процесор, який широко використовують для паралельних обчислень, зокрема під час навчання моделей машинного навчання.

Є 4 GPU-кластери із доступними лімітами a_i (у десятках GPU-годин) та 5 ML-експериментів із потребами b_j. Вартість c_{ij} — ціна використання ресурсу плюс накладні витрати на міграцію датасетів.

$$a = (30, 40, 20, 50), \quad b = (25, 35, 30, 20, 30)$$

$$C = \begin{pmatrix} 10 & 15 & 12 & 18 & 20 \\ 14 & 9 & 16 & 11 & 13 \\ 18 & 17 & 8 & 14 & 16 \\ 11 & 13 & 15 & 9 & 10 \end{pmatrix}$$

1. Сформулюйте задачу як збалансовану транспортну задачу мінімізації вартості.

2. Побудуйте початковий опорний план.

3. Знайдіть оптимальний план методом потенціалів.

4. Обчисліть мінімальні сумарні витрати.

5. Інтерпретуйте потенціали як «внутрішні ціни» GPU-ресурсів для різних кластерів і поясніть, які кластери є найефективнішими для розміщення експериментів.

Задача (IT) 1.4: Служба підтримки: призначення тікетів на команди

IT-коментар

У сервісних IT-системах транспортна задача дає змогу моделювати розподіл тікетів між командами підтримки так, щоб мінімізувати середній час реакції, ризик ескалації та нерівномірність навантаження. Така модель допомагає переходити від ручного розподілу звернень до обґрунтованого планування, заснованого на кількісних оцінках. У результаті можна не лише зменшити час обробки інцидентів, а й збалансувати навантаження між командами.

Є 4 команди підтримки (A_i) із добовою місткістю a_i (кількість тікетів) та 5 категорій інцидентів (B_j) з обсягами b_j. Вартість c_{ij} — середній час або ризик ескалації при обробці тікета B_j командою A_i.

$$a = (40, 60, 50, 70), \quad b = (50, 40, 60, 30, 20)$$

$$C = \begin{pmatrix} 2 & 5 & 4 & 7 & 3 \\ 6 & 1 & 5 & 4 & 8 \\ 3 & 6 & 2 & 5 & 4 \\ 5 & 4 & 6 & 2 & 5 \end{pmatrix}$$

1. Перевірте умову балансу.
2. Зведіть задачу до збалансованої, увівши фіктивну категорію інцидентів, і дайте їй практичну інтерпретацію як резерву часу або простою команд.
3. Побудуйте початковий опорний план.
4. Знайдіть оптимальний план методом потенціалів.
5. Поясніть практичний зміст базисного нуля, якщо під час розв'язування виникає виродженість.
6. Інтерпретуйте оптимальний план у термінах організації роботи команд підтримки.

Задача (IT) 1.5: Безпека: розподіл часу сканування вразливостей

IT-коментар

У задачах кібербезпеки транспортна задача дає змогу моделювати розподіл часу сканування між агентами безпеки та групами серверів так, щоб мінімізувати навантаження на мережу та ефективно використовувати ресурси команди безпеки. Таке планування допомагає раціонально розподілити обмежений час фахівців між пріоритетними об'єктами перевірки. Оптимальний план також дає змогу зрозуміти, які групи серверів доцільно закріпити за конкретними агентами для зменшення мережевих витрат.

Є 4 агенти безпеки з доступним часом a_i (год/тиждень) та 5 груп серверів, які

потребують b_j годин сканування. Вартість c_{ij} відображає навантаження на мережу.

$$a = (80, 100, 60, 120), \quad b = (70, 90, 80, 50, 70)$$

$$C = \begin{pmatrix} 3 & 9 & 4 & 8 & 5 \\ 7 & 2 & 6 & 5 & 9 \\ 4 & 8 & 3 & 7 & 6 \\ 6 & 5 & 8 & 2 & 4 \end{pmatrix}$$

Для швидкого старту DevOps[a]-інженер запропонував такий жадібний розподіл годин (матриця X): $x_{11} = 70$, $x_{12} = 10$, $x_{22} = 80$, $x_{23} = 20$, $x_{33} = 60$, $\mathbf{x_{34} = 0}$ (базисний нуль), $x_{44} = 50$, $x_{45} = 70$. Усі інші x_{ij} — порожні небазисні клітини.

1. Переконайтеся, що запропонований план X є допустимим опорним планом, перевіривши суми по рядках і стовпчиках та кількість базисних клітин $m + n - 1$.
2. Виконайте перевірку оптимальності цього плану методом потенціалів.
3. Якщо план не є оптимальним, послідовно виконайте коригування до отримання оптимального плану.
4. Обчисліть мінімальні сумарні витрати.
5. Поясніть, як отриманий оптимальний план можна використати для ефективного планування сканування вразливостей.

[a] DevOps (*Development and Operations*) — підхід до узгодження процесів розроблення, супроводу та експлуатації програмних систем.

Питання для самоконтролю

1. Що називається *збалансованою* (закритою) транспортною задачею? Яка умова балансу і що вона гарантує?

2. Які змінні, цільова функція та система обмежень у стандартній математичній моделі транспортної задачі?

3. Що таке *транспортна таблиця*? Які дані в ній розміщуються і як інтерпретуються клітини (i, j)?

4. Що називається *допустимим планом перевезень*? Які умови повинні виконуватися для рядків і стовпчиків таблиці?

5. Що таке *опорний план* (ДБР) транспортної задачі? Скільки базисних клітин має бути в опорному плані і чому?

6. У чому полягає ідея *ациклічності* базису? Як графова інтерпретація пов'язує базисні клітини з циклами?

7. Опишіть кроки методу *північно-західного кута*. У чому його перевага та ключовий недолік?

8. Опишіть кроки методу *мінімального елемента*. Чому він часто дає "кращий" стартовий план, ніж МПЗК?

9. Що таке *виродженість* у транспортній задачі? Коли виникає потреба у *базисному нулі* і навіщо він потрібен?

10. Що таке *потенціали* рядків і стовпчиків? Як їх знаходять за відомим опорним планом?

11. Що таке *оцінки небазисних клітин* (редуковані витрати) у методі потенціалів? Як вони інтерпретуються?

12. Який *критерій оптимальності* в методі потенціалів? Що означає наявність від'ємної оцінки?

13. Як будується *коригувальний цикл* після вибору клітини для входу в базис? Як розставляються знаки «+» і «−» на циклі?

14. Як визначається величина перерахунку θ і яка клітина виходить з базису? Чому після перерахунку план лишається допустимим?

15. Для чого будують *двоїсту задачу* до транспортної? Як пов'язані потенціали з двоїстими змінними та умовами оптимальності?

Тема 2. Потоки на мережах: задача про найкоротший шлях

Короткі теоретичні відомості

Задача про найкоротший шлях у мережах є однією з класичних задач на графах і водночас важливою складовою курсу «Дослідження операцій», оскільки поєднує елементи теорії графів, мережевого моделювання та алгоритмічної оптимізації. На відміну від загальних методів математичного програмування, що працюють з усією системою обмежень, мережева структура задачі дає змогу застосовувати спеціалізований алгоритм, який спирається на графове подання, локальні обчислення на дугах і покрокове розширення множини позначених вершин. Саме тому вивчення цієї задачі має не лише теоретичне, а й виразне прикладне значення.

У багатьох прикладних постановках виникає потреба визначити маршрут між двома вузлами мережі, для якого сумарні витрати є мінімальними. Такі задачі природно виникають у транспортних системах, логістичних моделях, геоінформаційних сервісах, телекомунікаційних мережах, системах навігації та різноманітних інформаційних системах, де потрібно знайти найвигідніший шлях передавання ресурсу, сигналу або даних.

Формально такі задачі описуються за допомогою зважених графів. Кожній дузі графа ставиться у відповідність числова характеристика — вага, яка може означати відстань, час, вартість перевезення, затримку передавання даних або іншу міру витрат. Тоді пошук маршруту з мінімальною сумарною вагою приводить до задачі про найкоротший шлях. Залежно від змісту моделі вага дуги може мати різну прикладну інтерпретацію, але математична сутність задачі залишається однаковою: необхідно знайти шлях, для якого сума ваг усіх дуг є найменшою.

Одним із алгоритмів розв'язування цієї задачі є метод Мінті, який дає змогу знаходити найкоротші шляхи від фіксованої вершини до всіх інших вершин мережі та будувати дерево найкоротших шляхів. Такий підхід є особливо зручним у тих випадках, коли потрібно не лише знайти один оптимальний маршрут, а й описати всю структуру найкоротших шляхів від заданого джерела до інших вершин графа.

Метод Мінті ґрунтується на ідеї поступового нарощування дерева найкоротших шляхів. На кожному кроці серед усіх дуг, що виходять із уже позначених вершин до ще не позначених, вибирається дуга, для якої довжина шляху від кореня зростає найменше. Після цього відповідна вершина одержує позначку, що інтерпретується як довжина найкоротшого знайденого шляху, а вибрана дуга включається до дерева. У результаті формується дерево найкоротших шляхів від заданої початкової вершини до всіх досяжних вершин мережі. Коректність цієї побудови спирається на теорему про оптимальність позначок: кожна позначена вершина отримує значення, що збігається з довжиною найкоротшого шляху від кореня до цієї вершини.

Для майбутнього фахівця з комп'ютерних наук задача про найкоротший шлях має важливе прикладне значення. Передусім вона є базовою моделлю для задач маршрутизації, пошуку шляхів у графах залежностей, навігації в геоінформаційних системах, планування переміщень роботизованих систем і вибору оптимальних маршрутів у транспортних сервісах. Крім того, метод Мінті наочно демонструє характерну для алгоритмів на графах ідею покрокового розширення часткового оптимального розв'язку, що концептуально перегукується з класичними жадібними алгоритмами, зокрема з алгоритмом Дейкстри. Нарешті, ця задача безпосередньо пов'язує графові моделі з оптимізаційними постановками дослідження операцій, оскільки може бути інтерпретована як окремий випадок задачі про оптимальний потік у мережі. Це створює основу для подальшого вивчення задач максимального потоку, потоку мінімальної вартості, мережевого планування та алгоритмічної

оптимізації в задачах ІТ.

ІТ-коментар: основні застосування задачі про найкоротший шлях

Задача про найкоротший шлях має багато прикладних інтерпретацій у комп'ютерних науках та інформаційних технологіях. Найтиповіші з них такі.

1. **Маршрутизація в комп'ютерних мережах.** У протоколах маршрутизації, зокрема OSPF та IS-IS, мережу подають у вигляді графа, де вершини відповідають маршрутизаторам, а дуги — каналам зв'язку. Вага дуги може відображати затримку, вартість використання каналу, адміністративну метрику або іншу характеристику якості з'єднання. Алгоритми пошуку найкоротших шляхів використовують для побудови таблиць маршрутизації, за якими визначають оптимальні маршрути передавання пакетів.

2. **Навігація в цифрових картах і геоінформаційних сервісах.** У системах навігації дорожню мережу також подають у вигляді графа: перехрестя є вершинами, а ділянки доріг — дугами. Вага дуги може означати довжину ділянки, час проїзду, очікувану затримку через трафік або вартість поїздки. У цьому випадку задача про найкоротший шлях лежить в основі побудови маршрутів у навігаційних застосунках і сервісах доставки.

3. **Планування маршрутів у робототехніці та автономних системах.** Під час руху мобільного робота, дрона або автономного транспортного засобу простір можливих переміщень дискретизують і подають у вигляді графа станів або графа конфігурацій. Алгоритм найкоротшого шляху дає змогу знайти маршрут від початкового стану до цільового стану з мінімальними витратами часу, енергії або ризику. Це є базовим елементом систем автономної навігації.

4. **Аналіз графів залежностей у програмних системах.** У програмній інженерії та аналізі програм графи використовують для опису залежностей між модулями, викликами функцій, потоками керування або потоками даних. Пошук найкоротших шляхів може застосовуватися для визначення мінімальних ланцюгів залежностей, аналізу поширення змін, оцінювання впливу компонентів один на одного та побудови оптимальних сценаріїв проходження через систему.

5. **Оптимізація взаємодії в розподілених і хмарних системах.** У розподілених обчисленнях і дата-центрах вузли системи, канали передавання та сервіси можна подати у вигляді мережі. Задача про найкоротший шлях тут виникає під час вибору найефективнішого маршруту передавання даних, організації обміну між сервісами, балансування навантаження та мінімізації затримок у розподілених застосунках. У такій постановці вага дуги може означати час відповіді, мережеву затримку, вартість передавання або рівень завантаження каналу.

Постановка задачі про найкоротший шлях на мережі

Нехай задано орієнтований граф

$$G = (I, U),$$

де I — множина вершин, U — множина дуг. Кожній дузі $(i, j) \in U$ поставлено у відповідність число c_{ij}, яке називають довжиною або вагою дуги.

Означення 2.1: Зважений граф

Зваженим орієнтованим графом називають граф $G = (I, U)$, у якому кожній дузі $(i, j) \in U$ поставлено у відповідність число c_{ij}, що називається довжиною або вагою дуги.

Означення 2.2: Задача про найкоротший шлях

Задача про найкоротший шлях між вершинами i_1 та i_s полягає у знаходженні такого шляху між ними, для якого сума довжин усіх дуг є мінімальною.

Основні означення та факти

Означення 2.3: Шлях

Шляхом у графі називають послідовність дуг

$$l_{i_1 i_s} = \big((i_1, i_2), (i_2, i_3), \ldots, (i_{s-1}, i_s)\big).$$

Вершину i_1 називають початком шляху, а вершину i_s — його кінцем.

Означення 2.4: Довжина шляху

Довжиною шляху

$$l_{i_1 i_s} = \big((i_1, i_2), (i_2, i_3), \ldots, (i_{s-1}, i_s)\big)$$

називають суму довжин усіх дуг, що входять до цього шляху:

$$c(l_{i_1 i_s}) = c_{i_1 i_2} + c_{i_2 i_3} + \cdots + c_{i_{s-1} i_s}.$$

Означення 2.5: Дерево найкоротших шляхів

Деревом найкоротших шляхів називають підграф мережі з фіксованим коренем, у якому для кожної досяжної вершини міститься один із найкоротших шляхів від кореня до цієї вершини.

Теорема 2.1: Властивість найкоротших шляхів

Якщо шлях до вершини j є найкоротшим, то будь-який його початковий фрагмент також є найкоротшим шляхом до відповідної проміжної вершини.

Зауваження 2.1: Зв'язок із задачею про оптимальний потік

Задачу про найкоротший шлях можна розглядати як окремий випадок задачі про оптимальний потік на мережі. Для цього вершину i_1 вважають джерелом одиничної інтенсивності, вершину i_s — стоком одиничної інтенсивності, а всі інші вершини — нейтральними. Якщо пропускні спроможності дуг не обмежують рух потоку, то мінімізація сумарної вартості перевезення одиниці потоку від i_1 до i_s приводить саме до задачі про найкоротший шлях.

Метод Мінті

Метод Мінті[1] застосовується для знаходження найкоротших шляхів від однієї вершини графа до всіх інших вершин. Алгоритм працює за умови, що довжини дуг є невід'ємними.

Ідея методу полягає у поступовому розширенні множини вершин, для яких уже відомі довжини найкоротших шляхів від початкової вершини.

Означення 2.6: Позначена вершина

Якщо для вершини i визначено число h_i, яке дорівнює довжині найкоротшого знайденого шляху від початкової вершини до вершини i, то вершину називають позначеною.

На кожному кроці алгоритму розглядаються всі дуги, що ведуть із позначених вершин до ще не позначених. Серед них вибирається дуга, для якої значення

$$h_i + c_{ij}$$

є мінімальним. Після цього вершина j отримує позначку

$$h_j = h_i + c_{ij}.$$

Ідея методу Мінті

Алгоритм послідовно розширює множину позначених вершин. На кожному кроці вибирається дуга, що забезпечує найменше можливе збільшення довжини шляху від початкової вершини. У результаті формується дерево найкоротших шляхів.

Теорема 2.2: Про оптимальність позначок методу Мінті

Нехай вершина i_s була позначена методом Мінті. Тоді:

1. позначка h_{i_s} дорівнює довжині найкоротшого шляху з вершини i_1 до вершини i_s;
2. шлях $l^*_{i_1 i_s}$, який відновлюється за вибраними попередниками, є найкоротшим шляхом із i_1 до i_s;
3. виконується рівність
$$c\left(l^*_{i_1 i_s}\right) = h_{i_s}.$$

Алгоритми

Нехай i_1 — початкова вершина мережі.

[1] Назва методу пов'язана з короткими публікаціями Джорджа Дж. Мінті 1957 і 1958 років, присвяченими задачі найкоротшого маршруту (*A Comment on the Shortest-Route Problem*, *A Variant on the Shortest-Route Problem*). У сучасній міжнародній літературі споріднений алгоритм найчастіше пов'язують з ім'ям Едсгера В. Дейкстри після його статті 1959 року *A Note on Two Problems in Connexion with Graphs*. У вітчизняній та пострадянській навчальній традиції закріпилася назва «метод Мінті», імовірно, через вплив ранніх праць з дослідження операцій та подальше усталення цього терміна в навчальних курсах і посібниках.

Алгоритм методу Мінті

1. Початкову вершину i_1 позначити числом

$$h_{i_1} = 0.$$

Усі інші вершини вважати непозначеними.

2. Серед усіх дуг (i, j), що виходять із уже позначених вершин i до ще не позначених вершин j, обчислити величини

$$h_i + c_{ij}.$$

3. Вибрати дугу (i, j), для якої значення $h_i + c_{ij}$ є найменшим.

4. Вершині j поставити у відповідність позначку

$$h_j = h_i + c_{ij},$$

і зафіксувати вершину i як попередника вершини j.

5. Додати вершину j до множини позначених вершин.

6. Повторювати кроки 2–5 доти, доки існують дуги, що ведуть із позначених вершин до непозначених.

7. Якщо деякі вершини мережі залишилися непозначеними, то вони є недосяжними з початкової вершини i_1.

Правило вибору в разі неоднозначності

Якщо найменше значення $h_i + c_{ij}$ досягається для кількох дуг, доцільно вибирати дугу з найменшими індексами (i, j). Це забезпечує однозначність побудови дерева найкоротших шляхів.

Зауваження 2.2: Практичний результат

Після завершення алгоритму позначка кожної позначеної вершини дорівнює довжині найкоротшого шляху від початкової вершини i_1. За зафіксованими попередниками можна відновити відповідні найкоротші маршрути.

Типові задачі

Задача 2.1: знаходження найкоротших шляхів методом Мінті

Для заданої орієнтованої мережі знайти методом Мінті найкоротші шляхи від вершини 1 до всіх вершин мережі та побудувати дерево найкоротших шляхів.

Мережа задається матрицею

$$L = \begin{pmatrix} 1 & 1 & 2 & 3 & 3 & 4 & 5 \\ 2 & 3 & 4 & 4 & 5 & 6 & 6 \\ 3 & 1 & 2 & 5 & 2 & 3 & 4 \end{pmatrix}.$$

Кожний стовпчик матриці задає одну дугу мережі: у першому рядку записано початок дуги, у другому — її кінець, а в третьому — довжину.

Потрібно:

1. зобразити задану орієнтовану мережу графічно;
2. послідовно виконати кроки методу Мінті;
3. для кожного кроку вказати множини $P(s)$ та $I(s)$;
4. знайти позначки всіх досяжних вершин;
5. побудувати дерево найкоротших шляхів з коренем у вершині 1;
6. відновити найкоротший шлях від вершини 1 до вершини 6.

Розв'язання.

Спочатку відтворимо мережу за матрицею

$$L = \begin{pmatrix} 1 & 1 & 2 & 3 & 3 & 4 & 5 \\ 2 & 3 & 4 & 4 & 5 & 6 & 6 \\ 3 & 1 & 2 & 5 & 2 & 3 & 4 \end{pmatrix}.$$

Отже, мережа містить дуги

$$(1,2) = 3,\ (1,3) = 1,\ (2,4) = 2,\ (3,4) = 5,\ (3,5) = 2,\ (4,6) = 3,\ (5,6) = 4.$$

Початковий граф, побудований за цими даними, подано на рис. 2.1.

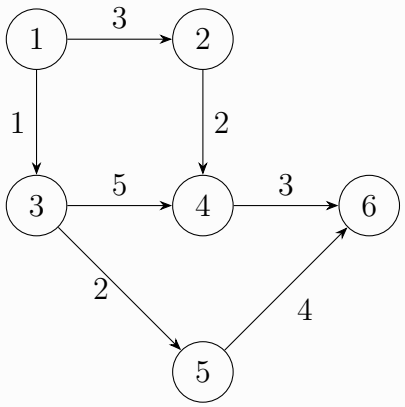

Рис. 2.1: Початковий граф задачі

Крок 0. Позначаємо початкову вершину:

$$h_1 = 0, \qquad P(0) = I(0) = \{1\}.$$

Крок 1. Розглядаємо дуги, що виходять із вершини 1:

$$(1,2), \qquad (1,3).$$

Обчислюємо:

$$h_1 + c_{12} = 0 + 3 = 3, \qquad h_1 + c_{13} = 0 + 1 = 1.$$

Отже,

$$\theta_1 = \min(3, 1) = 1.$$

Мінімум досягається на дузі $(1, 3)$, тому

$$P(1) = \{3\}, \qquad h_3 = 1,$$
$$I(1) = I(0) \cup P(1) = \{1, 3\}.$$

Крок 2. Тепер розглядаємо дуги, що виходять із множини $I(1) = \{1, 3\}$ до непозначених вершин:

$$(1, 2), \qquad (3, 4), \qquad (3, 5).$$

Обчислюємо:

$$h_1 + c_{12} = 0 + 3 = 3, \quad h_3 + c_{34} = 1 + 5 = 6, \quad h_3 + c_{35} = 1 + 2 = 3.$$

Тому

$$\theta_2 = \min(3, 6, 3) = 3.$$

Мінімум досягається на двох дугах: $(1, 2)$ і $(3, 5)$. Вони заходять у різні вершини, тому

$$P(2) = \{2, 5\}, \qquad h_2 = 3, \qquad h_5 = 3,$$
$$I(2) = I(1) \cup P(2) = \{1, 2, 3, 5\}.$$

Крок 3. Розглядаємо дуги, що виходять із множини $I(2) = \{1, 2, 3, 5\}$ до непозначених вершин:

$$(2, 4), \qquad (3, 4), \qquad (5, 6).$$

Обчислюємо:

$$h_2 + c_{24} = 3 + 2 = 5, \quad h_3 + c_{34} = 1 + 5 = 6, \quad h_5 + c_{56} = 3 + 4 = 7.$$

Отже,

$$\theta_3 = \min(5, 6, 7) = 5.$$

Мінімум досягається на дузі $(2, 4)$, тому

$$P(3) = \{4\}, \qquad h_4 = 5,$$
$$I(3) = I(2) \cup P(3) = \{1, 2, 3, 4, 5\}.$$

Крок 4. Розглядаємо дуги, що виходять із множини $I(3) = \{1, 2, 3, 4, 5\}$ до непозначених вершин:

$$(4, 6), \qquad (5, 6).$$

Обчислюємо:

$$h_4 + c_{46} = 5 + 3 = 8, \quad h_5 + c_{56} = 3 + 4 = 7.$$

Тому

$$\theta_4 = \min(8, 7) = 7.$$

Мінімум досягається на дузі $(5, 6)$, отже

$$P(4) = \{6\}, \qquad h_6 = 7,$$
$$I(4) = I(3) \cup P(4) = \{1, 2, 3, 4, 5, 6\}.$$

Крок 5. Усі вершини мережі вже позначені, тому алгоритм завершується. Підсумок кроків методу Мінті наведено в табл. 2.1.

Табл. 2.1: Підсумок кроків методу Мінті

Крок s	θ_s	$P(s)$	$I(s)$
0	–	$\{1\}$	$\{1\}$
1	1	$\{3\}$	$\{1,3\}$
2	3	$\{2,5\}$	$\{1,2,3,5\}$
3	5	$\{4\}$	$\{1,2,3,4,5\}$
4	7	$\{6\}$	$\{1,2,3,4,5,6\}$

Отже, позначки вершин мають вигляд

$$h_1 = 0, \qquad h_2 = 3, \qquad h_3 = 1, \qquad h_4 = 5, \qquad h_5 = 3, \qquad h_6 = 7.$$

Дерево найкоротших шляхів з коренем у вершині 1 утворюють дуги

$$T = \{(1,3),\ (1,2),\ (3,5),\ (2,4),\ (5,6)\}.$$

Кінцевий граф із позначками вершин і виділеним деревом найкоротших шляхів подано на рис. 2.2.

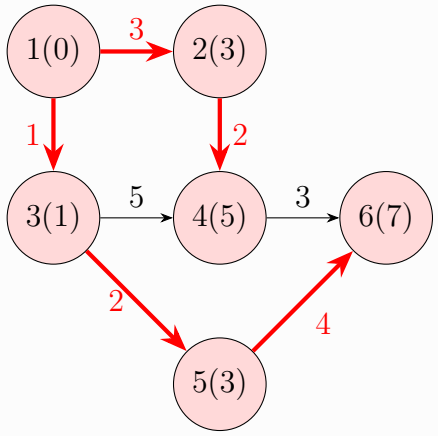

Рис. 2.2: Кінцевий граф із позначками та виділеним деревом найкоротших шляхів

Відновимо найкоротший шлях від вершини 1 до вершини 6. Як видно з рис. 2.2, у вершину 6 заходить дуга $(5,6)$, у вершину 5 — дуга $(3,5)$, а у вершину 3 — дуга $(1,3)$. Отже,

$$l^*_{1,6} = \{(1,3),(3,5),(5,6)\}.$$

Його довжина дорівнює

$$c(l^*_{1,6}) = 1 + 2 + 4 = 7.$$

Це збігається з позначкою вершини 6:

$$c(l^*_{1,6}) = h_6 = 7.$$

Задача 2.2: Маршрутизація в мережі дата-центру

Мережу дата-центру подано орієнтованим графом, у якому вершини відповідають серверам і комутаторам, а дуги — каналам передавання даних. Довжина дуги дорівнює затримці передавання пакета даних у мілісекундах.

Мережа містить 9 вершин. Нехай вершина 1 відповідає центральному серверу керування. Потрібно визначити методом Мінті найкоротші маршрути передавання даних від вершини 1 до інших вузлів мережі.

Наявні дуги мережі задаються матрицею

$$L = \begin{pmatrix} 1 & 1 & 2 & 3 & 2 & 3 & 4 & 5 & 6 & 2 & 7 & 9 & 9 \\ 2 & 3 & 4 & 4 & 5 & 5 & 6 & 6 & 7 & 7 & 8 & 5 & 8 \\ 2 & 2 & 3 & 3 & 4 & 4 & 2 & 2 & 3 & 8 & 2 & 1 & 1 \end{pmatrix}.$$

Кожний стовпчик матриці задає одну дугу мережі: у першому рядку записано початок дуги, у другому — її кінець, а в третьому — довжину.

Потрібно:

1. зобразити мережу дата-центру графічно у вигляді зваженого орієнтованого графа;
2. знайти методом Мінті позначки всіх вершин, досяжних із вершини 1;
3. побудувати дерево найкоротших шляхів з коренем у вершині 1;
4. визначити, чи існують вершини мережі, які не можна досягти з вершини 1, і пояснити, як це виявляється під час роботи алгоритму;
5. знайти найкоротший шлях від вершини 1 до вершини 6;
6. з'ясувати, чи існують альтернативні найкоротші шляхи від вершини 1 до вершини 6, і якщо так, то виписати їх;
7. знайти найкоротший шлях від вершини 1 до вершини 7 та порівняти маршрут, що проходить через вершину 6, з маршрутом, який містить дугу $(2, 7)$;
8. інтерпретувати отримані результати в термінах маршрутизації пакетів у мережі дата-центру.

Розв'язання.

1. Побудова графа мережі.

За матрицею

$$L = \begin{pmatrix} 1 & 1 & 2 & 3 & 2 & 3 & 4 & 5 & 6 & 2 & 7 & 9 & 9 \\ 2 & 3 & 4 & 4 & 5 & 5 & 6 & 6 & 7 & 7 & 8 & 5 & 8 \\ 2 & 2 & 3 & 3 & 4 & 4 & 2 & 2 & 3 & 8 & 2 & 1 & 1 \end{pmatrix}$$

одержуємо дуги мережі:

$$(1,2) = 2, \quad (1,3) = 2, \quad (2,4) = 3, \quad (3,4) = 3, \quad (2,5) = 4, \quad (3,5) = 4,$$

$$(4,6) = 2, \quad (5,6) = 2, \quad (6,7) = 3, \quad (2,7) = 8, \quad (7,8) = 2, \quad (9,5) = 1, \quad (9,8) = 1.$$

Отже, вершина 9 має лише вихідні дуги $(9, 5)$ і $(9, 8)$, але не має жодної вхідної дуги. Початковий граф мережі дата-центру подано на рис. 2.3.

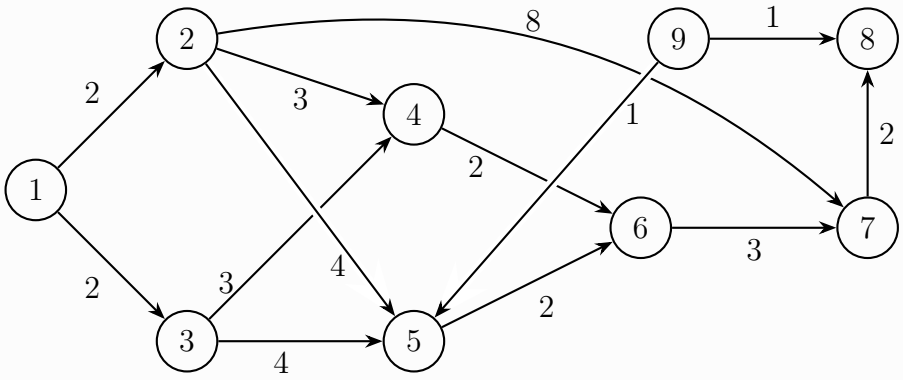

Рис. 2.3: Початковий граф мережі дата-центру

2. Знаходження позначок усіх досяжних вершин методом Мінті.

Крок 0. Позначаємо початкову вершину:

$$h_1 = 0, \qquad P(0) = I(0) = \{1\}.$$

Крок 1. Розглядаємо дуги, що виходять із вершини 1:

$$(1,2), \qquad (1,3).$$

Обчислюємо:

$$h_1 + c_{12} = 0 + 2 = 2, \qquad h_1 + c_{13} = 0 + 2 = 2.$$

Отже,

$$\theta_1 = 2.$$

Мінімум досягається на двох дугах, які заходять у різні вершини, тому

$$P(1) = \{2,3\}, \qquad h_2 = 2, \qquad h_3 = 2,$$

$$I(1) = \{1,2,3\}.$$

Крок 2. Розглядаємо дуги, що виходять із множини $I(1) = \{1,2,3\}$ до непозначених вершин:

$$(2,4), \qquad (3,4), \qquad (2,5), \qquad (3,5), \qquad (2,7).$$

Обчислюємо:

$$h_2 + c_{24} = 2 + 3 = 5, \qquad h_3 + c_{34} = 2 + 3 = 5,$$

$$h_2 + c_{25} = 2 + 4 = 6, \qquad h_3 + c_{35} = 2 + 4 = 6,$$

$$h_2 + c_{27} = 2 + 8 = 10.$$

Тому

$$\theta_2 = 5.$$

Мінімум досягається на дугах $(2,4)$ та $(3,4)$, але вони заходять в одну й ту саму вершину 4. Тому вибираємо одну з них, наприклад $(2,4)$. Отже,

$$P(2) = \{4\}, \qquad h_4 = 5,$$

$$I(2) = \{1, 2, 3, 4\}.$$

Крок 3. Розглядаємо дуги, що виходять із множини $I(2) = \{1, 2, 3, 4\}$ до непозначених вершин:

$$(2, 5), \qquad (3, 5), \qquad (4, 6), \qquad (2, 7).$$

Обчислюємо:

$$h_2 + c_{25} = 2 + 4 = 6, \qquad h_3 + c_{35} = 2 + 4 = 6,$$

$$h_4 + c_{46} = 5 + 2 = 7,$$

$$h_2 + c_{27} = 2 + 8 = 10.$$

Отже,

$$\theta_3 = 6.$$

Мінімум досягається на дугах $(2, 5)$ і $(3, 5)$, які заходять в одну й ту саму вершину 5. Вибираємо одну з них, наприклад $(2, 5)$. Тому

$$P(3) = \{5\}, \qquad h_5 = 6,$$

$$I(3) = \{1, 2, 3, 4, 5\}.$$

Крок 4. Розглядаємо дуги, що виходять із множини $I(3) = \{1, 2, 3, 4, 5\}$ до непозначених вершин:

$$(4, 6), \qquad (5, 6), \qquad (2, 7).$$

Обчислюємо:

$$h_4 + c_{46} = 5 + 2 = 7, \qquad h_5 + c_{56} = 6 + 2 = 8, \qquad h_2 + c_{27} = 2 + 8 = 10.$$

Отже,

$$\theta_4 = 7.$$

Мінімум досягається на дузі $(4, 6)$, тому

$$P(4) = \{6\}, \qquad h_6 = 7,$$

$$I(4) = \{1, 2, 3, 4, 5, 6\}.$$

Крок 5. Розглядаємо дуги, що виходять із множини $I(4) = \{1, 2, 3, 4, 5, 6\}$ до непозначених вершин:

$$(2, 7), \qquad (6, 7).$$

Обчислюємо:

$$h_2 + c_{27} = 2 + 8 = 10, \qquad h_6 + c_{67} = 7 + 3 = 10.$$

Отже,

$$\theta_5 = 10.$$

Мінімум досягається на двох дугах $(2, 7)$ і $(6, 7)$, які заходять в одну й ту саму вершину 7. Тому вибираємо одну з них, наприклад $(6, 7)$. Тоді

$$P(5) = \{7\}, \qquad h_7 = 10,$$

$$I(5) = \{1, 2, 3, 4, 5, 6, 7\}.$$

Крок 6. Розглядаємо дуги, що виходять із множини $I(5) = \{1,2,3,4,5,6,7\}$ до непозначених вершин:

$$(7,8).$$

Обчислюємо:

$$h_7 + c_{78} = 10 + 2 = 12.$$

Отже,

$$\theta_6 = 12.$$

Мінімум досягається на дузі $(7,8)$, тому

$$P(6) = \{8\}, \qquad h_8 = 12,$$

$$I(6) = \{1,2,3,4,5,6,7,8\}.$$

На цьому кроці нові вершини, досяжні з вершини 1, уже не з'являються. Вершина 9 залишається непозначеною. Підсумок кроків методу Мінті наведено в табл. 2.2.

Табл. 2.2: Підсумок кроків методу Мінті

Крок s	θ_s	$P(s)$	$I(s)$
0	–	$\{1\}$	$\{1\}$
1	2	$\{2,3\}$	$\{1,2,3\}$
2	5	$\{4\}$	$\{1,2,3,4\}$
3	6	$\{5\}$	$\{1,2,3,4,5\}$
4	7	$\{6\}$	$\{1,2,3,4,5,6\}$
5	10	$\{7\}$	$\{1,2,3,4,5,6,7\}$
6	12	$\{8\}$	$\{1,2,3,4,5,6,7,8\}$

Отже, позначки вершин мають вигляд

$$h_1 = 0, \quad h_2 = 2, \quad h_3 = 2, \quad h_4 = 5, \quad h_5 = 6, \quad h_6 = 7, \quad h_7 = 10, \quad h_8 = 12.$$

Вершина 9 позначки не отримала.

3. Побудова дерева найкоротших шляхів.

За вибраними дугами одержуємо дерево найкоротших шляхів

$$T = \{(1,2),\ (1,3),\ (2,4),\ (2,5),\ (4,6),\ (6,7),\ (7,8)\}.$$

Кінцевий граф із позначками та виділеним деревом найкоротших шляхів подано на рис. 2.4.

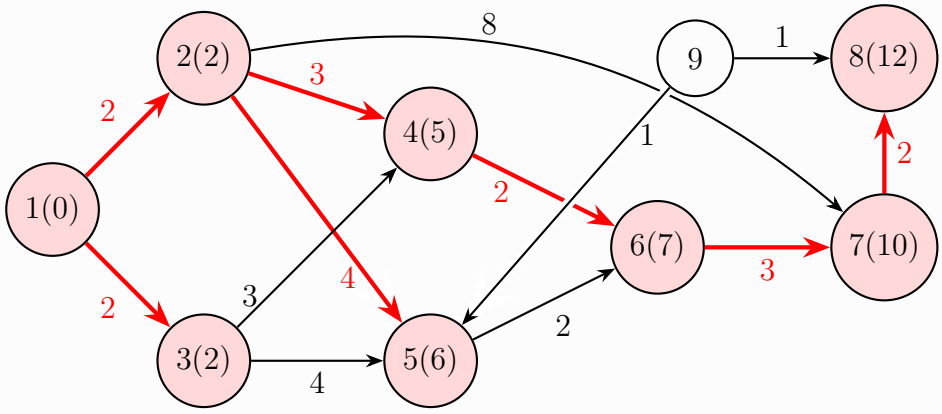

Рис. 2.4: Кінцевий граф із позначками та виділеним деревом найкоротших шляхів

4. Визначення недосяжних вершин.

Так. Вершина 9 є недосяжною з вершини 1, оскільки під час роботи алгоритму вона не отримує позначки. Це означає, що з вершини 1 не існує жодного шляху до вершини 9. Той факт, що з вершини 9 виходять дуги $(9, 5)$ і $(9, 8)$, не впливає на результат, оскільки алгоритм Мінті розглядає лише дуги, що виходять із уже позначених вершин.

5. Найкоротший шлях від вершини 1 *до вершини* 6*.*

У побудованому дереві маємо шлях

$$1 \to 2 \to 4 \to 6,$$

його довжина дорівнює

$$2 + 3 + 2 = 7.$$

6. Альтернативні найкоротші шляхи від вершини 1 *до вершини* 6*.*

Існує й альтернативний найкоротший шлях

$$1 \to 3 \to 4 \to 6,$$

який має ту саму довжину

$$2 + 3 + 2 = 7.$$

Отже, до вершини 6 існують два альтернативні найкоротші шляхи:

$$1 \to 2 \to 4 \to 6, \qquad 1 \to 3 \to 4 \to 6.$$

7. Найкоротший шлях від вершини 1 *до вершини* 7 *і порівняння маршрутів.*

У побудованому дереві маємо шлях

$$1 \to 2 \to 4 \to 6 \to 7,$$

його довжина дорівнює

$$2 + 3 + 2 + 3 = 10.$$

Порівняємо його з маршрутом, що містить дугу $(2, 7)$:

$$1 \to 2 \to 7.$$

Його довжина також дорівнює

$$2 + 8 = 10.$$

Крім того, існує ще один шлях тієї самої довжини:

$$1 \to 3 \to 4 \to 6 \to 7, \qquad 2 + 3 + 2 + 3 = 10.$$

Отже, до вершини 7 існує кілька альтернативних найкоротших шляхів довжини 10:

$$1 \to 2 \to 7, \qquad 1 \to 2 \to 4 \to 6 \to 7, \qquad 1 \to 3 \to 4 \to 6 \to 7.$$

8. Інтерпретація результатів у термінах маршрутизації пакетів.

З погляду маршрутизації пакетів у мережі дата-центру отримані позначки задають мінімальні затримки передавання даних від центрального сервера 1 до інших вузлів мережі. Дерево найкоротших шляхів задає один із можливих наборів оптимальних маршрутів. Наявність альтернативних найкоротших шляхів означає, що система має кілька рівноцінних маршрутів передавання даних, що може бути використано для балансування навантаження або підвищення надійності маршрутизації.

Задачі для самостійного розв'язування

Задача 2.1: Задача про найкоротший шлях

Для свого варіанта задано орієнтовану мережу у вигляді матриці

$$L = \begin{pmatrix} i_1 & i_2 & \dots & i_m \\ j_1 & j_2 & \dots & j_m \\ c_{i_1 j_1} & c_{i_2 j_2} & \dots & c_{i_m j_m} \end{pmatrix},$$

де кожний стовпчик задає одну дугу мережі: у першому рядку записано початок дуги, у другому — її кінець, а в третьому — довжину дуги.

Для кожного варіанта також задано початкову вершину s та вершину t, до якої потрібно відновити найкоротший шлях.

Потрібно:

1. Зобразити задану мережу графічно у вигляді зваженого орієнтованого графа.
2. Виконати метод Мінті для знаходження найкоротших шляхів від вершини s до всіх вершин мережі.
3. Для кожного кроку вказати множини $P(s)$ та $I(s)$, а також нові позначки вершин.
4. Побудувати дерево найкоротших шляхів з коренем у вершині s.
5. Відновити найкоротший шлях від вершини s до вершини t та знайти його довжину.

Варіанти:

1.1 Початкова вершина $s = 1$, кінцева вершина $t = 6$.

$$L = \begin{pmatrix} 1 & 1 & 2 & 3 & 3 & 4 & 5 \\ 2 & 3 & 4 & 4 & 5 & 6 & 6 \\ 3 & 1 & 2 & 5 & 2 & 3 & 4 \end{pmatrix}.$$

1.2 Початкова вершина $s = 1$, кінцева вершина $t = 6$.

$$L = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 4 & 5 \\ 2 & 3 & 4 & 5 & 5 & 6 & 6 \\ 2 & 4 & 3 & 6 & 2 & 2 & 3 \end{pmatrix}.$$

1.3 Початкова вершина $s = 1$, кінцева вершина $t = 5$.

$$L = \begin{pmatrix} 1 & 1 & 2 & 3 & 2 & 4 & 5 \\ 2 & 3 & 4 & 4 & 5 & 6 & 6 \\ 4 & 2 & 1 & 3 & 5 & 2 & 1 \end{pmatrix}.$$

1.4 Початкова вершина $s = 1$, кінцева вершина $t = 6$.

$$L = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 4 & 5 \\ 2 & 3 & 4 & 5 & 5 & 6 & 6 \\ 3 & 2 & 4 & 5 & 3 & 2 & 4 \end{pmatrix}.$$

1.5 Початкова вершина $s = 1$, кінцева вершина $t = 6$.

$$L = \begin{pmatrix} 1 & 1 & 2 & 3 & 4 & 2 & 5 \\ 2 & 4 & 3 & 5 & 6 & 5 & 6 \\ 2 & 6 & 1 & 4 & 2 & 3 & 2 \end{pmatrix}.$$

1.6 Початкова вершина $s = 1$, кінцева вершина $t = 5$.

$$L = \begin{pmatrix} 1 & 1 & 2 & 3 & 2 & 4 & 5 \\ 2 & 3 & 4 & 4 & 5 & 6 & 6 \\ 2 & 2 & 3 & 3 & 4 & 2 & 2 \end{pmatrix}.$$

1.7 Початкова вершина $s = 1$, кінцева вершина $t = 6$.

$$L = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 4 & 5 \\ 2 & 3 & 4 & 5 & 5 & 6 & 6 \\ 1 & 3 & 2 & 6 & 4 & 2 & 2 \end{pmatrix}.$$

1.8 Початкова вершина $s = 1$, кінцева вершина $t = 6$.

$$L = \begin{pmatrix} 1 & 1 & 2 & 3 & 2 & 4 & 5 \\ 2 & 4 & 3 & 5 & 5 & 6 & 6 \\ 2 & 5 & 1 & 3 & 4 & 2 & 3 \end{pmatrix}.$$

1.9 Початкова вершина $s = 1$, кінцева вершина $t = 6$.

$$L = \begin{pmatrix} 1 & 1 & 2 & 3 & 2 & 3 & 4 & 5 \\ 2 & 3 & 4 & 4 & 5 & 5 & 6 & 6 \\ 2 & 2 & 3 & 3 & 4 & 4 & 2 & 2 \end{pmatrix}.$$

1.10 Початкова вершина $s = 1$, кінцева вершина $t = 5$.

$$L = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 4 & 5 \\ 2 & 4 & 3 & 5 & 6 & 6 & 6 \\ 4 & 1 & 2 & 3 & 4 & 2 & 3 \end{pmatrix}.$$

1.11 Початкова вершина $s = 1$, кінцева вершина $t = 6$.

$$L = \begin{pmatrix} 1 & 1 & 2 & 3 & 4 & 2 & 5 \\ 2 & 4 & 3 & 5 & 6 & 5 & 6 \\ 3 & 5 & 2 & 4 & 1 & 3 & 2 \end{pmatrix}.$$

1.12 Початкова вершина $s = 1$, кінцева вершина $t = 6$.

$$L = \begin{pmatrix} 1 & 1 & 2 & 3 & 2 & 4 & 5 \\ 2 & 3 & 4 & 5 & 6 & 6 & 6 \\ 2 & 3 & 3 & 4 & 6 & 2 & 2 \end{pmatrix}.$$

1.13 Початкова вершина $s = 1$, кінцева вершина $t = 6$.

$$L = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 4 & 5 \\ 2 & 4 & 3 & 5 & 5 & 6 & 6 \\ 3 & 4 & 1 & 5 & 2 & 3 & 2 \end{pmatrix}.$$

1.14 Початкова вершина $s = 1$, кінцева вершина $t = 6$.

$$L = \begin{pmatrix} 1 & 1 & 2 & 3 & 2 & 4 & 5 & 3 \\ 2 & 3 & 4 & 5 & 6 & 6 & 6 & 6 \\ 2 & 2 & 3 & 4 & 5 & 2 & 3 & 6 \end{pmatrix}.$$

1.15 Початкова вершина $s = 1$, кінцева вершина $t = 6$.

$$L = \begin{pmatrix} 1 & 1 & 2 & 3 & 2 & 5 & 4 & 3 \\ 2 & 3 & 4 & 4 & 5 & 6 & 6 & 6 \\ 1 & 2 & 3 & 5 & 4 & 2 & 3 & 4 \end{pmatrix}.$$

Задача 2.2: Задача про найкоротший шлях для мереж, заданих ребрами

Для свого варіанта задано зважену мережу з множиною вершин

$$V = \{1, 2, \ldots, 10\},$$

у вигляді матриці

$$R = \begin{pmatrix} i_1 & i_2 & \ldots & i_m \\ j_1 & j_2 & \ldots & j_m \\ c_{i_1 j_1} & c_{i_2 j_2} & \ldots & c_{i_m j_m} \end{pmatrix},$$

де кожний стовпчик задає одне ребро мережі: у першому рядку записано один кінець ребра, у другому — другий кінець ребра, а в третьому — довжину ребра.

Вказівка. Під час розв'язування задачі кожне ребро $[i, j]$ слід замінити парою протилежно спрямованих дуг (i, j) та (j, i) однакової довжини c_{ij}.

Для кожного варіанта задано початкову вершину s та вершину t, до якої потрібно відновити найкоротший шлях.

Потрібно:

1. Зобразити задану мережу графічно у вигляді зваженого неорієнтованого графа.
2. Замінити кожне ребро парою протилежно спрямованих дуг однакової довжини.
3. Виконати метод Мінті для знаходження найкоротших шляхів від вершини s до всіх досяжних з неї вершин.
4. Для кожного кроку вказати множини $P(s)$, $I(s)$ та значення θ_s.
5. Побудувати дерево найкоротших шляхів з коренем у вершині s.
6. Визначити, чи існують вершини, недосяжні з вершини s.
7. Відновити найкоротший шлях від вершини s до вершини t, якщо він існує, та знайти його довжину.

Варіанти:

1.1 Початкова вершина $s = 1$, кінцева вершина $t = 9$.

$$R = \begin{pmatrix} 1 & 1 & 1 & 2 & 2 & 3 & 3 & 4 & 4 & 5 & 6 & 6 & 7 & 7 & 8 \\ 2 & 3 & 4 & 4 & 5 & 4 & 6 & 5 & 6 & 7 & 7 & 8 & 8 & 9 & 9 \\ 4 & 6 & 5 & 2 & 7 & 1 & 6 & 3 & 4 & 2 & 2 & 4 & 3 & 6 & 2 \end{pmatrix}.$$

1.2 Початкова вершина $s = 1$, кінцева вершина $t = 8$.

$$R = \begin{pmatrix} 1 & 1 & 1 & 2 & 2 & 3 & 3 & 4 & 4 & 5 & 6 & 6 & 7 & 8 & 7 \\ 2 & 3 & 4 & 4 & 5 & 4 & 6 & 5 & 6 & 7 & 7 & 8 & 8 & 9 & 9 \\ 3 & 5 & 7 & 3 & 4 & 2 & 5 & 6 & 3 & 2 & 4 & 3 & 2 & 4 & 5 \end{pmatrix}.$$

1.3 Початкова вершина $s = 1$, кінцева вершина $t = 9$.

$$R = \begin{pmatrix} 1 & 1 & 1 & 2 & 2 & 3 & 3 & 4 & 4 & 5 & 6 & 6 & 7 & 7 & 8 \\ 2 & 3 & 4 & 4 & 5 & 4 & 6 & 5 & 6 & 7 & 7 & 8 & 8 & 9 & 9 \\ 2 & 7 & 4 & 3 & 5 & 2 & 4 & 4 & 6 & 3 & 1 & 5 & 2 & 4 & 3 \end{pmatrix}.$$

1.4 Початкова вершина $s = 1$, кінцева вершина $t = 8$.

$$R = \begin{pmatrix} 1 & 1 & 1 & 2 & 2 & 3 & 3 & 4 & 4 & 5 & 6 & 6 & 7 & 7 & 8 \\ 2 & 3 & 4 & 4 & 5 & 4 & 6 & 5 & 6 & 7 & 7 & 8 & 8 & 9 & 9 \\ 5 & 4 & 6 & 1 & 6 & 3 & 7 & 2 & 5 & 4 & 2 & 3 & 3 & 5 & 2 \end{pmatrix}.$$

1.5 Початкова вершина $s = 1$, кінцева вершина $t = 7$.

$$R = \begin{pmatrix} 1 & 1 & 1 & 2 & 2 & 3 & 3 & 4 & 4 & 5 & 6 & 6 & 7 & 7 & 8 \\ 2 & 3 & 4 & 4 & 5 & 4 & 6 & 5 & 6 & 7 & 7 & 8 & 8 & 9 & 9 \\ 4 & 3 & 8 & 2 & 5 & 4 & 6 & 3 & 2 & 3 & 4 & 5 & 2 & 4 & 3 \end{pmatrix}.$$

1.6 Початкова вершина $s = 1$, кінцева вершина $t = 9$.

$$R = \begin{pmatrix} 1 & 1 & 1 & 2 & 2 & 3 & 3 & 4 & 4 & 5 & 6 & 6 & 7 & 7 & 8 \\ 2 & 3 & 4 & 4 & 5 & 4 & 6 & 5 & 6 & 7 & 7 & 8 & 8 & 9 & 9 \\ 6 & 2 & 5 & 2 & 4 & 3 & 5 & 5 & 2 & 2 & 3 & 4 & 1 & 5 & 2 \end{pmatrix}.$$

1.7 Початкова вершина $s = 1$, кінцева вершина $t = 8$.

$$R = \begin{pmatrix} 1 & 1 & 1 & 2 & 2 & 3 & 3 & 4 & 4 & 5 & 6 & 6 & 7 & 7 & 8 \\ 2 & 3 & 4 & 4 & 5 & 4 & 6 & 5 & 6 & 7 & 7 & 8 & 8 & 9 & 9 \\ 3 & 6 & 4 & 2 & 5 & 4 & 3 & 3 & 5 & 4 & 2 & 2 & 3 & 6 & 4 \end{pmatrix}.$$

1.8 Початкова вершина $s = 1$, кінцева вершина $t = 9$.

$$R = \begin{pmatrix} 1 & 1 & 1 & 2 & 2 & 3 & 3 & 4 & 4 & 5 & 6 & 6 & 7 & 7 & 8 \\ 2 & 3 & 4 & 4 & 5 & 4 & 6 & 5 & 6 & 7 & 7 & 8 & 8 & 9 & 9 \\ 2 & 5 & 6 & 4 & 3 & 1 & 6 & 2 & 4 & 5 & 2 & 3 & 2 & 4 & 3 \end{pmatrix}.$$

1.9 Початкова вершина $s = 1$, кінцева вершина $t = 8$.

$$R = \begin{pmatrix} 1 & 1 & 1 & 2 & 2 & 3 & 3 & 4 & 4 & 5 & 6 & 6 & 7 & 7 & 8 \\ 2 & 3 & 4 & 4 & 5 & 4 & 6 & 5 & 6 & 7 & 7 & 8 & 8 & 9 & 9 \\ 5 & 3 & 7 & 2 & 6 & 2 & 4 & 4 & 3 & 3 & 2 & 5 & 2 & 3 & 4 \end{pmatrix}.$$

1.10 Початкова вершина $s = 1$, кінцева вершина $t = 7$.

$$R = \begin{pmatrix} 1 & 1 & 1 & 2 & 2 & 3 & 3 & 4 & 4 & 5 & 6 & 6 & 7 & 7 & 8 \\ 2 & 3 & 4 & 4 & 5 & 4 & 6 & 5 & 6 & 7 & 7 & 8 & 8 & 9 & 9 \\ 4 & 4 & 6 & 1 & 5 & 3 & 5 & 2 & 4 & 4 & 2 & 3 & 3 & 5 & 2 \end{pmatrix}.$$

1.11 Початкова вершина $s = 1$, кінцева вершина $t = 9$.

$$R = \begin{pmatrix} 1 & 1 & 1 & 2 & 2 & 3 & 3 & 4 & 4 & 5 & 6 & 6 & 7 & 7 & 8 \\ 2 & 3 & 4 & 4 & 5 & 4 & 6 & 5 & 6 & 7 & 7 & 8 & 8 & 9 & 9 \\ 3 & 7 & 5 & 2 & 4 & 3 & 5 & 3 & 6 & 2 & 1 & 4 & 2 & 5 & 3 \end{pmatrix}.$$

1.12 Початкова вершина $s = 1$, кінцева вершина $t = 8$.

$$R = \begin{pmatrix} 1 & 1 & 1 & 2 & 2 & 3 & 3 & 4 & 4 & 5 & 6 & 6 & 7 & 7 & 8 \\ 2 & 3 & 4 & 4 & 5 & 4 & 6 & 5 & 6 & 7 & 7 & 8 & 8 & 9 & 9 \\ 2 & 6 & 4 & 3 & 5 & 2 & 4 & 4 & 3 & 3 & 2 & 4 & 1 & 4 & 2 \end{pmatrix}.$$

1.13 Початкова вершина $s = 1$, кінцева вершина $t = 9$.

$$R = \begin{pmatrix} 1 & 1 & 1 & 2 & 2 & 3 & 3 & 4 & 4 & 5 & 6 & 6 & 7 & 7 & 8 \\ 2 & 3 & 4 & 4 & 5 & 4 & 6 & 5 & 6 & 7 & 7 & 8 & 8 & 9 & 9 \\ 5 & 2 & 6 & 2 & 7 & 3 & 4 & 2 & 5 & 4 & 2 & 3 & 2 & 6 & 3 \end{pmatrix}.$$

1.14 Початкова вершина $s = 1$, кінцева вершина $t = 8$.

$$R = \begin{pmatrix} 1 & 1 & 1 & 2 & 2 & 3 & 3 & 4 & 4 & 5 & 6 & 6 & 7 & 7 & 8 \\ 2 & 3 & 4 & 4 & 5 & 4 & 6 & 5 & 6 & 7 & 7 & 8 & 8 & 9 & 9 \\ 4 & 5 & 3 & 2 & 6 & 4 & 5 & 3 & 2 & 4 & 3 & 2 & 3 & 5 & 4 \end{pmatrix}.$$

1.15 Початкова вершина $s = 1$, кінцева вершина $t = 7$.

$$R = \begin{pmatrix} 1 & 1 & 1 & 2 & 2 & 3 & 3 & 4 & 4 & 5 & 6 & 6 & 7 & 7 & 8 \\ 2 & 3 & 4 & 4 & 5 & 4 & 6 & 5 & 6 & 7 & 7 & 8 & 8 & 9 & 9 \\ 3 & 4 & 7 & 1 & 5 & 2 & 6 & 3 & 4 & 2 & 3 & 5 & 2 & 4 & 3 \end{pmatrix}.$$

Прикладні задачі

Задача (IT) 2.1: Навігація безпілотного автомобіля в розумному місті

IT-коментар

У системах автономної навігації задача про найкоротший шлях дає змогу визначати найшвидші або найвигідніші маршрути пересування між вузлами транспортної мережі з урахуванням поточного трафіку, затримок і обмежень дорожньої інфраструктури.

Транспортну мережу розумного міста[a] подано орієнтованим графом, де вершини — це перехрестя, а дуги — ділянки доріг. Довжина дуги відповідає часу проїзду цієї ділянки у секундах з урахуванням поточного трафіку, зафіксованого камерами відеоспостереження.

Нехай вершина 1 — поточне місцезнаходження безпілотного автомобіля. Потрібно визначити методом Мінті найшвидші маршрути до інших перехресть.

Мережа задається матрицею

$$L = \begin{pmatrix} 1 & 1 & 2 & 3 & 2 & 3 & 4 & 5 & 6 & 2 \\ 2 & 3 & 4 & 4 & 5 & 5 & 6 & 6 & 7 & 7 \\ 15 & 10 & 20 & 15 & 25 & 10 & 12 & 18 & 14 & 60 \end{pmatrix}.$$

Кожний стовпчик матриці задає одну дугу мережі: у першому рядку записано початок дуги, у другому — її кінець, а в третьому — час проїзду.

Потрібно:

1. зобразити транспортну мережу графічно у вигляді зваженого орієнтованого графа;
2. знайти методом Мінті позначки всіх вершин, досяжних із вершини 1, тобто мінімальний час прибуття до кожного перехрестя;
3. побудувати дерево найкоротших шляхів з коренем у вершині 1;
4. визначити, чи існують перехрестя, до яких неможливо прокласти маршрут, і пояснити, як алгоритм це виявляє;
5. знайти найшвидший маршрут від вершини 1 до вершини 6;
6. з'ясувати, чи існують альтернативні найшвидші маршрути до вершини 6, і якщо так, то виписати їх;
7. знайти найшвидший маршрут від вершини 1 до вершини 7 та порівняти маршрут транзитом через вершину 6 з прямим проїздом через ділянку $(2, 7)$;
8. інтерпретувати отримані результати в контексті роботи алгоритмів GPS-навігації та уникнення заторів.

[а] Smart City — концепція «розумного міста», у межах якої транспортна, комунальна та інформаційна інфраструктура інтегруються з цифровими системами керування та аналізу даних.

Задача (IT) 2.2: Маршрутизація запитів у мікросервісній архітектурі

IT-коментар

У мікросервісній архітектурі задача про найкоротший шлях дає змогу визначати найефективніші ланцюжки викликів між сервісами, мінімізуючи сумарний час відповіді та мережеві затримки.

Архітектуру бекенду веб-застосунку подано орієнтованим графом, де вершини відповідають окремим мікросервісам, а дуги — API-викликам[а] між ними. Вага дуги означає середній час обробки та мережевої затримки запиту в мілісекундах.

Нехай вершина 1 — це API Gateway[б], тобто точка входу всіх користувацьких запитів. Потрібно знайти найшвидші ланцюжки викликів до внутрішніх сервісів системи.

Мережа задається матрицею

$$L = \begin{pmatrix} 1 & 1 & 2 & 3 & 2 & 4 & 3 & 5 & 6 & 3 \\ 2 & 3 & 4 & 4 & 5 & 6 & 6 & 7 & 7 & 7 \\ 10 & 25 & 15 & 5 & 20 & 10 & 30 & 15 & 10 & 45 \end{pmatrix}.$$

Кожний стовпчик матриці задає одну дугу графа викликів: у першому рядку записано початок дуги, у другому — її кінець, а в третьому — вагу дуги.

Потрібно:

1. зобразити архітектуру мікросервісів у вигляді орієнтованого зваженого графа;
2. знайти методом Мінті позначки всіх вершин, що відповідають мінімальному часу відгуку від API Gateway до кожного сервісу;
3. побудувати дерево найкоротших шляхів з коренем у вершині 1;
4. визначити ізольовані сервіси, якщо вони є, і пояснити, що означає непозначена вершина в контексті мікросервісної архітектури;
5. знайти оптимальний ланцюжок викликів від 1 до мікросервісу 6;
6. перевірити наявність альтернативних шляхів однакової вартості до сервісу 6;
7. побудувати оптимальний шлях до бази даних (вершина 7) та порівняти виклик через сервіс 6 із довгим прямим запитом $(3, 7)$;
8. пояснити, як ці результати допомагають виявити вузькі місця у продуктивності бекенду.

[а] API (*Application Programming Interface*) — програмний інтерфейс, за допомогою якого одна програмна компонента взаємодіє з іншою.

[б] API Gateway — шлюз API, тобто єдина точка входу зовнішніх запитів до системи мікросервісів.

Задача (IT) 2.3: Пошук шляху для ігрового штучного інтелекту

IT-коментар

У задачах GameDev[а] алгоритми пошуку найкоротших шляхів використовують для побудови маршрутів неігрових персонажів, мінімізації витрат дії та вибору раціональної траєкторії руху в складному ігровому середовищі.

[а] GameDev (*Game Development*) — розроб відеоігор, включно з програмуванням ігрової логіки, штучного інтелекту, графіки та взаємодії з користувачем.

Ігрову локацію поділено на навігаційну сітку[а] (NavMesh), подану орієнтованим графом. Вершини — це ключові зони, а дуги — можливі переходи між ними. Довжина дуги визначає витрати очок дії[б] на подолання перешкод.

Вершина 1 — поточна позиція ігрового бота. Потрібно знайти найменш витратні маршрути до інших зон на карті.

Мережа задається матрицею

$$L = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 3 & 4 & 5 & 6 & 4 \\ 2 & 3 & 4 & 5 & 4 & 6 & 7 & 7 & 7 & 6 \\ 4 & 2 & 5 & 3 & 6 & 8 & 3 & 4 & 2 & 9 \end{pmatrix}.$$

Кожний стовпчик матриці задає одну дугу графа: у першому рядку записано початок дуги, у другому — її кінець, а в третьому — витрати очок дії.

Потрібно:

1. зобразити навігаційний граф графічно;
2. виконати кроки методу Мінті для знаходження мінімальних витрат очок дії до кожної зони;
3. побудувати дерево оптимальних шляхів переміщення бота;
4. пояснити математичну ознаку в алгоритмі Мінті, яка виявляє недосяжну зону;
5. прокласти оптимальний маршрут до зони 6;
6. виявити альтернативні маршрути до зони 6, якщо вони існують;
7. знайти шлях до укриття (вершина 7) та пояснити вибір алгоритмом обхідного шляху замість прямого переходу;
8. інтерпретувати застосування цього методу для побудови поведінки неігрових персонажів.

[а]NavMesh (*Navigation Mesh*) — спеціальна структура даних, що описує допустимі області переміщення персонажів в ігровому середовищі.

[б]Action Points — очки дії, тобто умовний ресурс, який персонаж витрачає на пересування або виконання дій.

Задача (IT) 2.4: Аналіз графа атак у кібербезпеці

IT-коментар

У кібербезпеці задача про найкоротший шлях дає змогу моделювати найшвидші або найменш витратні сценарії поширення атаки в мережі, що допомагає виявляти критичні вразливості та пріоритети захисту.

Мережу корпорації моделюють за допомогою графа атак[а]. Вершини — це комп'ютери, сервери та IoT-пристрої[б]. Орієнтована дуга означає наявність вразливості, що дозволяє зловмиснику перейти з одного вузла на інший. Вага дуги дорівнює оціночному часу в годинах або рівню зусиль, необхідному для успішного зламу.

Вершина 1 — скомпрометований робочий комп'ютер, тобто точка входу в мережу. Потрібно знайти найшвидші вектори атак на інші активи компанії.

Мережа задається матрицею

$$L = \begin{pmatrix} 1 & 1 & 2 & 3 & 3 & 4 & 4 & 5 & 6 & 2 \\ 2 & 3 & 4 & 5 & 4 & 6 & 7 & 6 & 7 & 7 \\ 8 & 12 & 15 & 10 & 20 & 25 & 10 & 5 & 15 & 50 \end{pmatrix}.$$

Кожний стовпчик матриці задає одну дугу графа атак: у першому рядку записано початок дуги, у другому — її кінець, а в третьому — вагу дуги.

Потрібно:

1. візуалізувати граф корпоративних вразливостей;
2. знайти методом Мінті позначки всіх вершин, що відповідають мінімальному часу компрометації кожного вузла;
3. побудувати дерево найкоротших векторів атак з коренем у вершині 1;

4. пояснити ситуацію, коли до певного сервера немає шляху в графі атак;
5. визначити найкоротший шлях компрометації системи керування (вершина 6);
6. перевірити наявність альтернативних сценаріїв атаки на вузол 6;
7. побудувати маршрут до критичного сервера баз даних (вершина 7) та порівняти атаку через проміжні вузли із прямою спробою зламу $(2, 7)$;
8. пояснити, як фахівці з безпеки можуть використовувати ці результати для пріоритезації закриття вразливостей.

[a]Attack Graph — граф атак, тобто модель можливих сценаріїв просування зловмисника мережею через наявні вразливості.

[б]IoT (*Internet of Things*) — інтернет речей, тобто мережа фізичних пристроїв, що обмінюються даними через цифрові канали зв'язку.

Задача (IT) 2.5: Оптимізація енергоспоживання в сенсорній мережі

IT-коментар

У бездротових сенсорних мережах задача про найкоротший шлях дає змогу знаходити маршрути передавання даних із мінімальними витратами енергії, що безпосередньо впливає на тривалість роботи автономних пристроїв.

Розглядається бездротова мережа датчиків[a]. Вершини відповідають автономним датчикам, а дуги — можливості передати радіосигнал. Довжина дуги дорівнює витратам енергії в мікроджоулях на передавання одного пакета даних між пристроями.

Вершина 1 — віддалений датчик, якому потрібно передати показники на інші вузли мережі та до базової станції. Потрібно знайти енергоефективні маршрути ретрансляції.

Мережа задається матрицею

$$L = \begin{pmatrix} 1 & 1 & 2 & 3 & 2 & 3 & 4 & 5 & 6 & 2 \\ 2 & 3 & 4 & 4 & 5 & 5 & 6 & 6 & 7 & 7 \\ 5 & 8 & 4 & 2 & 6 & 9 & 3 & 3 & 2 & 15 \end{pmatrix}.$$

Кожний стовпчик матриці задає одну дугу графа: у першому рядку записано початок дуги, у другому — її кінець, а в третьому — енергетичну вартість передавання.

Потрібно:

1. зобразити сенсорну мережу графічно;
2. розрахувати методом Мінті мінімальні витрати енергії для доставки пакета до кожної вершини;
3. побудувати дерево оптимальної маршрутизації даних;
4. вказати ознаку в алгоритмі, за якою визначається відсутність радіовидимості до певного вузла;
5. визначити енергоефективний шлях передавання до вузла 6;
6. виявити альтернативні рівноцінні шляхи до вузла 6, якщо вони існують;

7. знайти найменш витратний маршрут до базової станції (вершина 7) та порівняти передавання через транзитні вузли із прямим далеким радіоканалом $(2, 7)$;

8. обґрунтувати важливість знайденого дерева найкоротших шляхів для збереження заряду батарей в IoT-мережі.

[a]Wireless Sensor Network — бездротова сенсорна мережа, тобто сукупність автономних датчиків, які передають дані через радіоканали.

Питання для самоконтролю

1. Що називається зваженим орієнтованим графом?

2. Що називається шляхом у графі?

3. Як означається довжина шляху?

4. У чому полягає задача про найкоротший шлях на мережі?

5. Які прикладні інтерпретації може мати вага дуги в задачі про найкоротший шлях?

6. Що називається деревом найкоротших шляхів?

7. У чому полягає властивість оптимальності частин найкоротшого шляху?

8. Які обмеження накладаються на ваги дуг для коректного застосування методу Мінті?

9. Яка ідея лежить в основі методу Мінті?

10. Що називається позначеною вершиною в методі Мінті?

11. Який зміст має позначка h_i у методі Мінті?

12. Які дуги розглядаються на кожному кроці методу Мінті?

13. Як обчислюється кандидат на нову позначку вершини?

14. За яким правилом вибирається нова вершина для позначення?

15. Що є результатом роботи методу Мінті?

16. Як за побудованим деревом найкоротших шляхів відновити маршрут до заданої вершини?

17. Що означає ситуація, коли деяка вершина після завершення алгоритму залишається непозначеною?

18. У чому полягає зв'язок задачі про найкоротший шлях із задачею про оптимальний потік?

19. Як інтерпретується задача про найкоротший шлях як задача переміщення одиниці потоку в мережі?

20. У яких задачах інформатики та комп'ютерних наук використовується задача про найкоротший шлях?

Тема 3. Потоки на мережах: задача про максимальний потік на мережі

Короткі теоретичні відомості

Задача про максимальний потік на мережі є однією з центральних задач мережевої оптимізації та важливою складовою курсу «Дослідження операцій», оскільки поєднує елементи теорії графів, мережевого моделювання та комбінаторної оптимізації. На відміну від загальних методів математичного програмування, що працюють з усією системою обмежень одночасно, мережева структура задачі дає змогу застосовувати спеціалізований алгоритмічний підхід. Одним із класичних методів розв'язування є метод Форда–Фалкерсона, який спирається на графове подання мережі, локальні перетворення на дугах і покроковий пошук збільшувальних ланцюгів. Саме тому вивчення цієї задачі має не лише теоретичне, а й виразне прикладне значення.

Задача про максимальний потік природно виникає в ситуаціях, коли потрібно оцінити найбільший можливий обсяг ресурсу, що може бути переданий від джерела до стоку в орієнтованій мережі з обмеженими пропускними спроможностями дуг. На відміну від задачі про найкоротший шлях, де основну увагу зосереджено на виборі одного оптимального маршруту, у задачі про максимальний потік досліджується пропускна спроможність мережі в цілому. Саме тому ця модель має фундаментальне значення для транспортних, інформаційних, виробничих, енергетичних і логістичних систем, де важливо не лише знайти шлях передавання, а й оцінити, який обсяг ресурсу система здатна пропустити загалом.

Формально мережу подають у вигляді орієнтованого графа, у якому виділено джерело, стік і задано пропускні спроможності дуг. Потік у такій мережі має задовольняти дві основні умови: по-перше, величина потоку на кожній дузі не може перевищувати її пропускної спроможності, а по-друге, для кожної проміжної вершини має виконуватися умова збереження потоку, тобто сумарний потік, що входить у вершину, повинен дорівнювати сумарному потоку, що виходить із неї. За цих умов задача полягає у знаходженні такого допустимого потоку, для якого сумарний обсяг ресурсу, що виходить із джерела й надходить до стоку, є найбільшим.

Метод Форда–Фалкерсона ґрунтується на ідеї поступового нарощування допустимого потоку. Починаючи з деякого початкового потоку, зазвичай нульового, на кожному кроці алгоритму в мережі шукають збільшувальний ланцюг від джерела до стоку, уздовж якого поточний потік можна збільшити. Після знаходження такого ланцюга компоненти потоку перераховують: на прямих дугах потік збільшують, а на зворотних — за потреби зменшують. У результаті кожний крок алгоритму приводить до збільшення загального значення потоку. Процес триває доти, доки в мережі існує хоча б один збільшувальний ланцюг. Якщо жодного такого ланцюга більше немає, поточний потік є максимальним.

Важливість цієї теми полягає не лише у вивченні конкретного алгоритму, а й у розумінні глибокого зв'язку між максимальною пропускною спроможністю мережі та її вузькими місцями. Цей зв'язок виражає теорема Форда–Фалкерсона про максимальний потік і мінімальний розріз. Вона стверджує, що найбільший можливий потік від джерела до стоку дорівнює найменшій сумарній пропускній спроможності серед усіх розрізів, що відокремлюють джерело від стоку. Такий результат має принципове значення, оскільки показує, що обмеження на максимальний потік визначаються не локально окремими дугами, а глобальною структурою мережі.

Для майбутнього фахівця з комп'ютерних наук задача про максимальний потік має суттєве прикладне значення. Передусім вона лежить в основі моделей оцінювання пропускної спроможності телекомунікаційних мереж, балансування навантаження в хмарних інфраструктурах, маршрутизації даних і розподілу задач у розподілених обчисленнях.

Крім того, вона безпосередньо пов'язана із задачею про мінімальний розріз, що широко використовується в алгоритмах аналізу графів, комп'ютерному зорі та сегментації зображень. Нарешті, метод Форда–Фалкерсона є базою для подальшого вивчення швидших і ефективніших алгоритмів мережевої оптимізації, зокрема алгоритму Едмондса–Карпа, алгоритму Дініца та методу проштовхування–перемаркування.

Не менш важливо й те, що задача про максимальний потік наочно демонструє фундаментальну для алгоритмічної оптимізації ідею покрокового поліпшення допустимого розв'язку. Початковий потік може бути дуже далеким від оптимального, однак шляхом послідовного знаходження збільшувальних ланцюгів і коректного перерахунку значень на дугах алгоритм поступово наближає його до максимального. Для майбутніх фахівців з комп'ютерних наук такий підхід є надзвичайно корисним, оскільки він показує, як математично складна мережева проблема зводиться до чіткої алгоритмічної процедури. Опанування цієї теми створює міцне підґрунтя для розуміння складніших задач мережевої оптимізації та для застосування відповідних моделей в інформаційних системах.

IT-коментар: основні застосування задачі про максимальний потік

Задача про максимальний потік має багато важливих застосувань у комп'ютерних науках та інформаційних технологіях. Найтиповіші з них такі.

1. **Оцінювання пропускної спроможності комп'ютерних мереж.** У телекомунікаційних і комп'ютерних мережах вершини моделюють маршрутизатори, комутатори або сервери, а дуги — канали зв'язку між ними. Пропускна спроможність дуги відображає максимальний обсяг даних, який можна передати за одиницю часу. Задача про максимальний потік дає змогу оцінити граничний обсяг трафіку, який мережа здатна передати від джерела до стоку.

2. **Балансування навантаження в дата-центрах і хмарних інфраструктурах.** У розподілених обчислювальних системах потрібно визначити, який сумарний обсяг запитів, задач або потоків даних можна передати між вузлами інфраструктури з урахуванням обмежень на серверні ресурси й канали зв'язку. Модель максимального потоку дає змогу оцінити межі продуктивності системи та виявити вузькі місця її архітектури.

3. **Розподіл задач у розподілених системах.** У задачах планування виконання обчислень потік може інтерпретуватися як сукупність завдань, що передаються від джерел формування задач до обчислювальних вузлів або сервісів обробки. У такій постановці максимальний потік характеризує найбільший обсяг роботи, який система здатна виконати за заданих обмежень.

4. **Сегментація зображень і комп'ютерний зір.** У багатьох алгоритмах сегментації зображень задачу розбиття зображення на області зручно зводити до задачі про мінімальний розріз у графі, а отже — і до задачі про максимальний потік. Пікселі або групи пікселів інтерпретуються як вершини графа, а ваги дуг відображають подібність або зв'язок між ними. Такий підхід широко застосовується в аналізі зображень і відео.

5. **Аналіз надійності та вразливостей мережевої інфраструктури.** Задача про максимальний потік дає змогу не лише обчислити граничну пропускну спроможність системи, а й виявити її критичні ділянки через відповідний мінімальний розріз. У контексті кібербезпеки, телекомунікацій і проєктування

мереж це дає можливість знаходити потенційні вузькі місця, критичні канали та елементи, відмова яких найбільше впливає на функціонування системи.

Постановка задачі про максимальний потік

У задачі про максимальний потік розглядається орієнтована мережа з одним джерелом s і одним стоком t. Кожна дуга (i, j) має обмежену пропускну спроможність $d_{ij} \geq 0$. Потрібно знайти такий розподіл потоків x_{ij} на дугах, який дає змогу передати від джерела до стоку максимальну кількість ресурсу.

Означення 3.1: Допустимий потік на мережі

Потік

$$X = \{x_{ij}\}$$

називають *допустимим потоком*, якщо виконуються такі умови:

1. *Обмеження пропускної спроможності:* потік на кожній дузі не може бути від'ємним і не може перевищувати її пропускну спроможність:

$$0 \leq x_{ij} \leq d_{ij}, \quad (i, j) \in U.$$

2. *Збереження потоку:* для кожної проміжної вершини (крім s та t) сумарний потік, що виходить із неї, дорівнює сумарному потоку, що входить у неї:

$$\sum_{j\colon (i,j)\in U} x_{ij} - \sum_{k\colon (k,i)\in U} x_{ki} = 0, \quad i \in I,\ i \neq s,\ i \neq t.$$

Означення 3.2: Величина потоку

Величиною потоку називають сумарний потік, що виходить із джерела s:

$$v(X) = \sum_{j\colon (s,j)\in U} x_{sj}.$$

Для допустимого потоку ця сама величина дорівнює також сумарному потоку, що входить у стік t.

Означення 3.3: Максимальний потік та мінімальний розріз

Допустимий потік, для якого величина потоку є найбільшою серед усіх допустимих потоків, називають *максимальним потоком*.
Розрізом мережі $U(C)$ називають множину дуг, що виходять із підмножини вершин C, яка містить джерело s, у множину $I \setminus C$, що містить стік t:

$$U(C) = \{(i, j) \in U : i \in C,\ j \in I \setminus C\}.$$

Пропускну спроможність розрізу визначають формулою

$$d(C) = \sum_{(i,j)\in U(C)} d_{ij}.$$

Розріз із найменшою сумарною пропускною спроможністю серед усіх розрізів, що відділяють s від t, називають *мінімальним розрізом*.

Зв'язок задачі про максимальний потік із ЗЛП

Задача про максимальний потік є окремим випадком лінійного програмування. Справді, якщо за змінні взяти дугові потоки x_{ij}, то задачу можна записати у вигляді

$$L = \sum_{j\colon (s,j)\in U} x_{sj} \to \max$$

за умов

$$\sum_{j\colon (i,j)\in U} x_{ij} - \sum_{k\colon (k,i)\in U} x_{ki} = 0, \qquad i \in I,\ i \neq s,\ i \neq t,$$

$$0 \le x_{ij} \le d_{ij}, \qquad (i,j) \in U.$$

Отже, цільова функція і всі обмеження є лінійними відносно змінних x_{ij}, тому задача належить до класу задач лінійного програмування.

Специфіка цієї задачі полягає в тому, що її обмеження мають *мережеву структуру обмежень*: кожна змінна входить лише в рівняння для початку і кінця відповідної дуги. Саме завдяки цій структурі для задачі про максимальний потік можна застосовувати не лише загальні методи ЗЛП, а й спеціалізовані мережеві алгоритми, зокрема метод Форда–Фалкерсона, які в багатьох випадках працюють значно ефективніше.

Теорема 3.1: Теорема Форда–Фалкерсона

Для будь-якої мережі з джерелом s і стоком t має місце рівність

$$\max_{X\in\mathcal{F}} v(X) = \min_{C\colon\ s\in C,\, t\notin C} d(C),$$

де $\mathcal{F}$ — множина всіх допустимих потоків на мережі, $v(X)$ — величина потоку X, а $d(C)$ — пропускна спроможність розрізу, породженого множиною вершин C. Отже, величина максимального потоку від джерела s до стоку t дорівнює пропускній спроможності мінімального розрізу, що відділяє s від t.

Зауваження 3.1: Практичний зміст теореми

Теорема Форда–Фалкерсона показує, що найбільший можливий потік у мережі визначається її "вузьким місцем", тобто розрізом із найменшою сумарною пропускною спроможністю. Саме тому задача про максимальний потік тісно пов'язана із задачею про мінімальний розріз.

Для знаходження максимального потоку використовують метод Форда–Фалкерсона[2]. Це ітераційна процедура, яка на кожному кроці шукає *збільшувальний ланцюг* від джерела до стоку в *залишковій мережі* і виконує перерахунок потоку уздовж нього. Для побудови такого ланцюга застосовують **метод позначок**. Позначка вершини j має вигляд $j(N_j, \theta_j)$, де N_j — індекс попередньої вершини, з якої вершина j була досягнута (зі знаком "+" у випадку прямої дуги і зі знаком "−" у випадку зворотної дуги), а θ_j — найбільша величина, на яку можна збільшити потік від джерела до вершини j уздовж побудованого ланцюга.

[2]Метод названо на честь Лестера Р. Форда-молодшого та Делберта Р. Фалкерсона. Класичний виклад методу подано в праці L. R. Ford Jr., D. R. Fulkerson *Maximal Flow through a Network* (1956). Попередній виклад результатів містився також у RAND-меморандумі 1954 року з тією самою назвою.

Алгоритми

Алгоритм методу Форда–Фалкерсона (метод позначок)

Початковий етап. Задають початковий допустимий потік

$$X^0 = \{\, x_{ij}^0 \colon (i,j) \in U\}.$$

Найчастіше як початковий беруть нульовий потік:

$$x_{ij}^0 = 0, \qquad (i,j) \in U.$$

Нехай на деякій ітерації задано поточний потік

$$X^n = \{\, x_{ij}^n \colon (i,j) \in U\}.$$

Крок 1. Розстановка позначок.

1.1. Позначають джерело s позначкою

$$s(0,\infty)$$

і утворюють початкову множину позначених вершин

$$I^{(1)} = \{\, s(0,\infty)\}.$$

Джерело вважають позначеним і нерозглянутим.

1.2. Нехай після r-го кроку побудовано множину позначених вершин

$$I^{(r)} = \{\ldots,\, i(N_i,\theta_i),\, \ldots\}.$$

Вибирають будь-яку позначену нерозглянуту вершину $i \in I^{(r)}$ з позначкою $i(N_i,\theta_i)$ і будують множини:

$$J^-(r) = \big\{\, j \notin I^{(r)} \colon (i,j) \in U,\ d_{ij} - x_{ij}^n > 0\big\},$$

$$J^+(r) = \big\{\, j \notin I^{(r)} \colon (j,i) \in U,\ x_{ji}^n > 0\big\}.$$

Тут $J^-(r)$ є множиною вершин, до яких можна перейти по прямих дугах, а $J^+(r)$ — множиною вершин, до яких можна перейти по зворотних дугах.

1.3. Для кожної вершини $j \in J^-(r)$ присвоюють позначку

$$j(+i,\theta_j), \qquad \theta_j = \min\big(\theta_i,\ d_{ij} - x_{ij}^n\big).$$

Для кожної вершини $j \in J^+(r)$ присвоюють позначку

$$j(-i,\theta_j), \qquad \theta_j = \min\big(\theta_i,\ x_{ji}^n\big).$$

1.4. Позначимо через

$$P^{(r+1)} = J^-(r) \cup J^+(r)$$

множину вершин, позначених на поточному кроці. Тоді нова множина позначених вершин має вигляд

$$I^{(r+1)} = I^{(r)} \cup P^{(r+1)}.$$

Після цього вершину i вважають розглянутою.

1.5. Процедуру повторюють доти, поки або не буде позначено стік t, або не залишиться позначених нерозглянутих вершин.

Крок 2. Перевірка умови зупинки.

- Якщо стік t **не отримав позначки**, то збільшувального ланцюга не існує. Алгоритм завершується, а поточний потік X^n є максимальним. Множина всіх позначених вершин
$$C^* = I^{(r)}$$
породжує мінімальний розріз $U(C^*)$.

- Якщо стік t **отримав позначку**
$$t(N_t, \theta_t),$$
то переходять до зміни потоку.

Крок 3. Зміна потоку.
Починаючи зі стоку t, рухаються у зворотному напрямку до джерела s, орієнтуючись на першу компоненту позначок N_j.

- Якщо вершину j було досягнуто по прямій дузі (i, j), тобто $N_j = +i$, то покладають
$$x_{ij}^{n+1} = x_{ij}^n + \theta_t.$$

- Якщо вершину j було досягнуто по зворотній дузі (j, i), тобто $N_j = -i$, то покладають
$$x_{ji}^{n+1} = x_{ji}^n - \theta_t.$$

Усі інші компоненти потоку залишають без змін. Після цього одержують новий потік
$$X^{n+1} = \{ x_{ij}^{n+1} \colon (i, j) \in U \},$$
усі позначки стирають і переходять до нової ітерації.

Зауваження 3.2: Залишкова мережа і критично важливі дуги

Під час роботи методу Форда–Фалкерсона фактично розглядають не лише початкову мережу, а й *залишкову мережу*. Для поточного потоку X^n у залишковій мережі:

- прямій дузі (i, j) відповідає залишкова пропускна спроможність
$$r_{ij} = d_{ij} - x_{ij}^n,$$
якщо $r_{ij} > 0$;

- зворотній дузі (j, i) відповідає можливість зменшити вже наявний потік на дузі (i, j), причому її залишкова пропускна спроможність дорівнює
$$r_{ji} = x_{ij}^n,$$
якщо $x_{ij}^n > 0$.

Саме в залишковій мережі шукають збільшувальний ланцюг від джерела s до стоку t. Якщо такого ланцюга немає, то поточний потік є максимальним.

Серед дуг збільшувального ланцюга особливу роль відіграють *критично важливі дуги* — дуги, для яких залишкова пропускна спроможність є найменшою на цьому ланцюзі. Саме вони визначають величину

$$\theta = \min r_{ij},$$

на яку можна збільшити потік уздовж знайденого ланцюга. Після перерахунку потоку принаймні одна з таких дуг стає насиченою у випадку прямої дуги або спорожнілою у випадку зворотної дуги, що і зумовлює перехід до нової ітерації алгоритму.

Зауваження 3.3: Перевірка правильності розв'язку

Для перевірки знайденого розв'язку слід переконатися, що величина побудованого максимального потоку, тобто сума потоків, які виходять із джерела s, дорівнює пропускній спроможності знайденого мінімального розрізу $U(C^*)$. При цьому всі дуги розрізу, що виходять із множини C^*, мають бути насиченими:

$$x_{ij} = d_{ij}, \qquad (i,j) \in U(C^*),$$

а дуги, що входять у C^* із множини $I \setminus C^*$, повинні мати нульовий потік:

$$x_{ji} = 0.$$

Типові задачі

Задача 3.1: Побудова максимального потоку

Для заданої мережі з джерелом у вершині 1 та стоком у вершині 5 знайти максимальний потік методом Форда–Фалкерсона (методом позначок). Мережу подано на рис. 3.1.

Потрібно:

1. побудувати задачу лінійного програмування, що відповідає задачі знаходження максимального потоку на мережі;
2. записати початковий нульовий потік;
3. на кожній ітерації вказати множини $I^{(r)}$, $J^-(r)$, $J^+(r)$;
4. побудувати збільшувальні ланцюги та знайти величини приросту потоку θ;
5. продовжити алгоритм до одержання максимального потоку;
6. знайти один із максимальних потоків і побудувати відповідний мінімальний розріз.

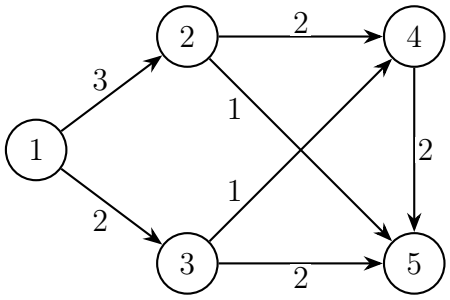

Рис. 3.1: Початкова мережа для задачі про максимальний потік

Розв'язання.

1. Побудова задачі лінійного програмування.

Позначимо через

$$x_{12},\ x_{13},\ x_{24},\ x_{25},\ x_{34},\ x_{35},\ x_{45}$$

потоки на відповідних дугах мережі. Тоді задачу про максимальний потік для мережі на рис. 3.1 можна записати як задачу лінійного програмування.

Якщо ввести змінну b — величину потоку, то маємо модель

$$b \to \max$$

за умов балансу:

$$x_{12} + x_{13} = b,$$
$$x_{24} + x_{25} - x_{12} = 0,$$
$$x_{34} + x_{35} - x_{13} = 0,$$
$$x_{45} - x_{24} - x_{34} = 0,$$
$$x_{25} + x_{35} + x_{45} = b,$$

та обмежень на пропускні спроможності:

$$0 \le x_{12} \le 3, \qquad 0 \le x_{13} \le 2,$$
$$0 \le x_{24} \le 2, \qquad 0 \le x_{25} \le 1,$$
$$0 \le x_{34} \le 1, \qquad 0 \le x_{35} \le 2, \qquad 0 \le x_{45} \le 2.$$

Виключивши змінну b, одержуємо еквівалентну задачу лінійного програмування:

$$L = x_{12} + x_{13} \to \max$$

за умов

$$x_{24} + x_{25} - x_{12} = 0,$$
$$x_{34} + x_{35} - x_{13} = 0,$$
$$x_{45} - x_{24} - x_{34} = 0,$$
$$0 \le x_{12} \le 3, \qquad 0 \le x_{13} \le 2,$$
$$0 \le x_{24} \le 2, \qquad 0 \le x_{25} \le 1,$$
$$0 \le x_{34} \le 1, \qquad 0 \le x_{35} \le 2, \qquad 0 \le x_{45} \le 2.$$

Отже, задача про максимальний потік для даної мережі є задачею лінійного програмування зі спеціальною мережевою структурою.

2. Початковий нульовий потік.
Починаємо з нульового потоку

$$X^0 = \{\, x_{ij}^0 = 0 \colon (i,j) \in U\}.$$

3–5. Ітерації методу Форда–Фалкерсона, побудова збільшувальних ланцюгів і одержання максимального потоку.
Ітерація 1.
Позначаємо джерело: $1(0,\infty)$, $I^{(1)} = \{\, 1(0,\infty)\}$.
Для вершини 1 маємо $J^-(1) = \{2,3\}$, $J^+(1) = \varnothing$, бо

$$d_{12} - x_{12}^0 = 3 > 0, \qquad d_{13} - x_{13}^0 = 2 > 0.$$

Одержуємо позначки: $2(+1,3)$, $3(+1,2)$.
Тоді

$$I^{(2)} = \{\, 1(0,\infty),\ 2(+1,3),\ 3(+1,2)\}.$$

Розглядаючи вершину 2, маємо $J^-(2) = \{4,5\}$, $J^+(2) = \varnothing$, оскільки

$$d_{24} - x_{24}^0 = 2 > 0, \qquad d_{25} - x_{25}^0 = 1 > 0.$$

Одержуємо $4(+2,2)$, $5(+2,1)$.
Стік 5 позначено, тому знайдено збільшувальний ланцюг

$$1 \to 2 \to 5, \qquad \theta_1 = 1.$$

Змінюємо потік:

$$x_{12}^1 = 1, \qquad x_{25}^1 = 1.$$

Ітерація 2.
Знову позначаємо джерело:

$$1(0,\infty), \qquad I^{(1)} = \{\, 1(0,\infty)\}.$$

Для вершини 1:

$$J^-(1) = \{2,3\}, \qquad J^+(1) = \varnothing,$$

бо

$$d_{12} - x_{12}^1 = 2 > 0, \qquad d_{13} - x_{13}^1 = 2 > 0.$$

Одержуємо

$$2(+1,2), \qquad 3(+1,2).$$

Далі для вершини 2:

$$J^-(2) = \{4\}, \qquad J^+(2) = \varnothing,$$

оскільки

$$d_{24} - x_{24}^1 = 2 > 0, \qquad d_{25} - x_{25}^1 = 0.$$

Отже,

$$4(+2,2).$$

Для вершини 4:

$$J^-(4) = \{5\}, \qquad J^+(4) = \varnothing,$$

бо

$$d_{45} - x^1_{45} = 2 > 0.$$

Тому

$$5(+4, 2).$$

Знайдено ланцюг

$$1 \to 2 \to 4 \to 5, \qquad \theta_2 = 2.$$

Змінюємо потік:

$$x^2_{12} = 3, \qquad x^2_{24} = 2, \qquad x^2_{45} = 2.$$

Ітерація 3.

Позначаємо джерело:

$$1(0, \infty), \qquad I^{(1)} = \{\, 1(0, \infty)\}.$$

Тепер

$$J^-(1) = \{3\}, \qquad J^+(1) = \varnothing,$$

оскільки

$$d_{12} - x^2_{12} = 0, \qquad d_{13} - x^2_{13} = 2 > 0.$$

Отже,

$$3(+1, 2).$$

Для вершини 3:

$$J^-(3) = \{4, 5\}, \qquad J^+(3) = \varnothing,$$

бо

$$d_{34} - x^2_{34} = 1 > 0, \qquad d_{35} - x^2_{35} = 2 > 0.$$

Оскільки стік можна позначити безпосередньо, беремо

$$5(+3, 2).$$

Знайдено ланцюг

$$1 \to 3 \to 5, \qquad \theta_3 = 2.$$

Змінюємо потік:

$$x^3_{13} = 2, \qquad x^3_{35} = 2.$$

Перевірка умови зупинки.

Для потоку X^3:

$$d_{12} - x^3_{12} = 0, \qquad d_{13} - x^3_{13} = 0.$$

Отже,

$$J^-(1) = \varnothing, \qquad J^+(1) = \varnothing.$$

Збільшувального ланцюга більше не існує, тому потік X^3 є максимальним.

Його величина:

$$v(X^*) = x^3_{12} + x^3_{13} = 3 + 2 = 5.$$

6. Один із максимальних потоків і мінімальний розріз.

Один із мінімальних розрізів:

$$C^* = \{1\}, \qquad U(C^*) = \{(1, 2), (1, 3)\},$$

$$d(C^*) = 3 + 2 = 5 = v(X^*).$$

Отже,

$$v(X^*) = 5.$$

Один із максимальних потоків має вигляд

$$x_{12} = 3, \quad x_{13} = 2, \quad x_{24} = 2, \quad x_{25} = 1, \quad x_{35} = 2, \quad x_{45} = 2,$$

а на решті дуг потік дорівнює нулю: $x_{34} = 0$.

Кінцевий потік для цієї задачі подано на рис. 3.2.

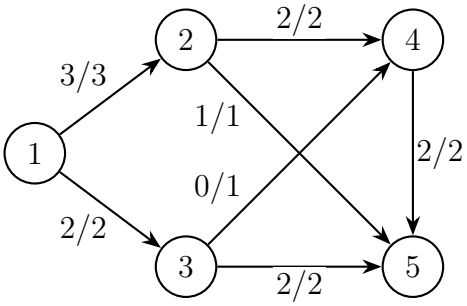

Рис. 3.2: Кінцевий потік у мережі (у форматі x_{ij}/d_{ij})

Задача 3.2: Максимальний потік, залишкова мережа і зворотна дуга

Для заданої мережі з джерелом у вершині 1 та стоком у вершині 4 знайти максимальний потік методом Форда–Фалкерсона (методом позначок). Мережу подано на рис. 3.3.

Потрібно:

1. побудувати початковий нульовий потік;
2. після першої ітерації побудувати залишкову мережу;
3. показати, що в залишковій мережі існує збільшувальний ланцюг, який містить зворотну дугу;
4. виконати перерахунок потоку;
5. вказати максимальний потік і мінімальний розріз.

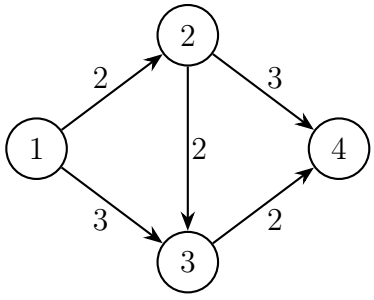

Рис. 3.3: Початкова мережа

Розв'язання.

1. Побудова початкового нульового потоку.

Починаємо з нульового потоку

$$X^0 = \{\, x_{ij}^0 = 0 \colon (i,j) \in U\}.$$

2–4. Побудова залишкової мережі, знаходження збільшувального ланцюга зі зворотною дугою та перерахунок потоку.

Ітерація 1.

Позначаємо джерело:

$$1(0,\infty), \qquad I^{(1)} = \{\, 1(0,\infty)\}.$$

Для вершини 1:

$$J^-(1) = \{2,3\}, \qquad J^+(1) = \varnothing.$$

Одержуємо

$$2(+1,2), \qquad 3(+1,3).$$

Розглядаємо вершину 2. Маємо

$$J^-(2) = \{3,4\}, \qquad J^+(2) = \varnothing,$$

бо

$$d_{23} - x_{23}^0 = 2 > 0, \qquad d_{24} - x_{24}^0 = 3 > 0.$$

Для реалізації зворотної дуги на наступному кроці зручно спочатку провести потік ланцюгом

$$1 \to 2 \to 3 \to 4.$$

Його пропускна спроможність:

$$\theta_1 = \min\{2,2,2\} = 2.$$

Змінюємо потік:

$$x_{12}^1 = 2, \qquad x_{23}^1 = 2, \qquad x_{34}^1 = 2.$$

Отже,

$$v(X^1) = 2.$$

Потік після першої ітерації подано на рис. 3.4.

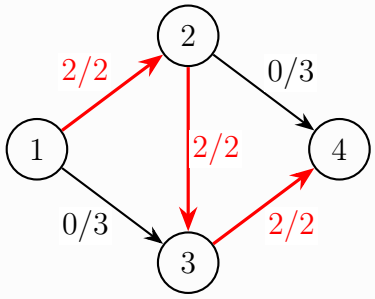

Рис. 3.4: Потік після першої ітерації

Залишкова мережа після першої ітерації.

Для потоку X^1 маємо, зокрема, такі ненульові залишкові пропускні спроможності:

$$r_{13} = 3, \qquad r_{24} = 3, \qquad r_{32} = x^1_{23} = 2.$$

Отже, в залишковій мережі існує збільшувальний ланцюг

$$1 \to 3 \to 2 \to 4,$$

де перехід $3 \to 2$ є зворотним до дуги $(2, 3)$. Цей ключовий ланцюг у залишковій мережі подано на рис. 3.5.

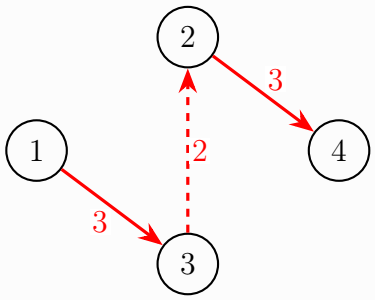

Рис. 3.5: Ключовий збільшувальний ланцюг у залишковій мережі: $1 \to 3 \to 2 \to 4$

Ітерація 2.

Позначаємо джерело:

$$1(0, \infty), \qquad I^{(1)} = \{\, 1(0, \infty)\}.$$

Для вершини 1:

$$J^-(1) = \{3\}, \qquad J^+(1) = \varnothing,$$

бо

$$d_{13} - x^1_{13} = 3 > 0, \qquad d_{12} - x^1_{12} = 0.$$

Тому

$$3(+1, 3).$$

Для вершини 3:

$$J^-(3) = \varnothing, \qquad J^+(3) = \{2\},$$

оскільки

$$x^1_{23} = 2 > 0, \qquad d_{34} - x^1_{34} = 0.$$

Отже,

$$2(-3, 2).$$

Для вершини 2:

$$J^-(2) = \{4\}, \qquad J^+(2) = \varnothing,$$

бо

$$d_{24} - x^1_{24} = 3 > 0.$$

Тому

$$4(+2, 2).$$

Знайдено ланцюг

$$1 \to 3 \to 2 \to 4, \qquad \theta_2 = 2.$$

Виконуємо перерахунок потоку:

$$x_{13}^2 = 2, \qquad x_{24}^2 = 2, \qquad x_{23}^2 = 0.$$

Компоненти

$$x_{12}^2 = 2, \qquad x_{34}^2 = 2$$

не змінюються. Отже,

$$v(X^2) = x_{12}^2 + x_{13}^2 = 2 + 2 = 4.$$

Перевірка умови зупинки.
Для потоку X^2 з джерела 1 можна перейти до вершини 3, бо

$$d_{13} - x_{13}^2 = 1 > 0,$$

але далі рух до стоку неможливий, оскільки

$$d_{34} - x_{34}^2 = 0, \qquad x_{23}^2 = 0.$$

Отже, стік 4 більше недосяжний у залишковій мережі, тому потік є максимальним.

5. Максимальний потік і мінімальний розріз.
Один із мінімальних розрізів:

$$C^* = \{1, 3\}, \qquad U(C^*) = \{(1, 2), (3, 4)\},$$

$$d(C^*) = 2 + 2 = 4 = v(X^*).$$

Отже,

$$v(X^*) = 4.$$

Один із максимальних потоків має вигляд

$$x_{12} = 2, \qquad x_{13} = 2, \qquad x_{24} = 2, \qquad x_{34} = 2,$$

а на решті дуг потік дорівнює нулю:

$$x_{23} = 0.$$

Кінцевий потік для цієї задачі подано на рис. 3.6.

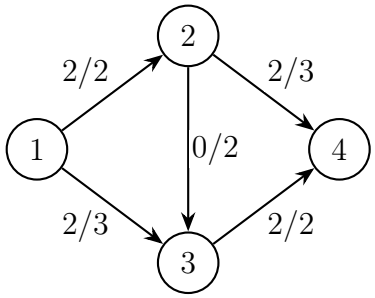

Рис. 3.6: Кінцевий потік у мережі (у форматі x_{ij}/d_{ij})

Задача 3.3: Розгортання оновлень прошивки в мережі IoT-пристроїв

Центральний сервер оновлень 1 передає пакети прошивки до цільового IoT-шлюзу 6 через проміжні вузли $2, 3, 4, 5$. Кожна дуга мережі має пропускну спроможність, що показує максимальну кількість пакетів оновлення за одну хвилину. Потрібно знайти максимальний потік пакетів від вузла 1 до вузла 6 методом Форда–Фалкерсона (методом позначок). Мережу подано на рис. 3.7.

Потрібно:

1. побудувати початковий нульовий потік;
2. знайти максимальний потік;
3. вказати мінімальний розріз;
4. інтерпретувати результат як максимальну інтенсивність розгортання оновлень.

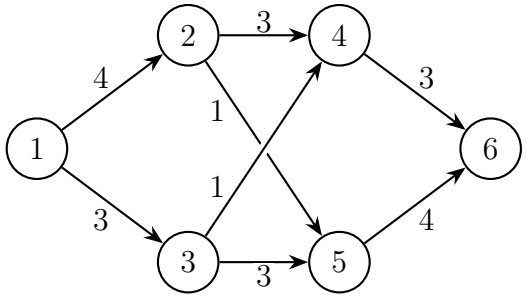

Рис. 3.7: Мережа розгортання оновлень

Розв'язання.

1. Побудова початкового нульового потоку.
Починаємо з нульового потоку

$$X^0 = \{\, x_{ij}^0 = 0 \colon (i,j) \in U \}.$$

2. Знаходження максимального потоку.
Метод позначок приводить до трьох послідовних збільшувальних ланцюгів:

$$1 \to 2 \to 4 \to 6, \qquad \theta_1 = 3,$$

$$1 \to 3 \to 5 \to 6, \qquad \theta_2 = 3,$$

$$1 \to 2 \to 5 \to 6, \qquad \theta_3 = 1.$$

Після трьох ітерацій маємо потік

$$x_{12} = 4, \qquad x_{13} = 3, \qquad x_{24} = 3, \qquad x_{25} = 1,$$

$$x_{35} = 3, \qquad x_{46} = 3, \qquad x_{56} = 4.$$

На решті дуг потік дорівнює нулю:

$$x_{34} = 0.$$

Отже,

$$v(X^*) = x_{12} + x_{13} = 4 + 3 = 7.$$

Оскільки обидві дуги, що виходять із джерела 1, насичені,

$$x_{12} = d_{12} = 4, \qquad x_{13} = d_{13} = 3,$$

то збільшувального ланцюга більше не існує. Тому знайдений потік є максимальним.

3. Мінімальний розріз.
Один із мінімальних розрізів: $C^* = \{1\}$, $U(C^*) = \{(1,2),(1,3)\}$,

$$d(C^*) = 4 + 3 = 7 = v(X^*).$$

4. Інтерпретація результату.
Знайдений максимальний потік 7 означає, що мережа здатна передавати не більш як 7 пакетів оновлення за хвилину від центрального сервера до цільового шлюзу без перевищення пропускних спроможностей каналів.

Відповідь.

$$v(X^*) = 7.$$

Кінцевий потік для цієї задачі подано на рис. 3.8.

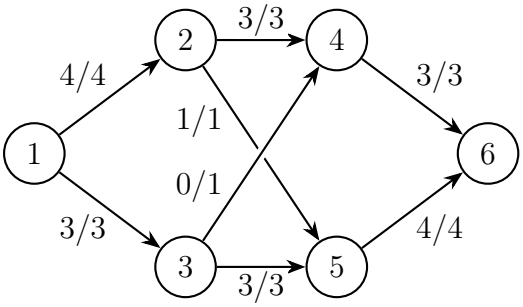

Рис. 3.8: Кінцевий потік у мережі розгортання оновлень (у форматі x_{ij}/d_{ij})

Задачі для самостійного розв'язування

Задача 3.1: Максимальний потік на мережі

Для свого варіанта задано мережу з джерелом у вершині 1 та стоком у вершині 5. Мережу задано матрицею

$$C = \begin{pmatrix} i_1 & i_2 & \dots & i_m \\ j_1 & j_2 & \dots & j_m \\ d_{i_1 j_1} & d_{i_2 j_2} & \dots & d_{i_m j_m} \end{pmatrix},$$

де кожний стовпчик задає одну дугу мережі: у першому рядку записано початок дуги, у другому — її кінець, а в третьому — пропускну спроможність.

Потрібно:

1. побудувати граф мережі;
2. записати початковий нульовий потік;
3. знайти максимальний потік методом Форда–Фалкерсона (методом позначок), указуючи на кожній ітерації множини $I^{(r)}$, $J^-(r)$, $J^+(r)$, збільшувальний ланцюг і величину приросту потоку θ;

4. записати один із максимальних потоків;
5. побудувати мінімальний розріз і перевірити рівність між величиною максимального потоку та пропускною спроможністю мінімального розрізу.

Варіанти:

1. $C = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 3 & 4 \\ 2 & 3 & 4 & 5 & 4 & 5 & 5 \\ 4 & 2 & 3 & 1 & 1 & 2 & 2 \end{pmatrix}$.

2. $C = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 3 & 4 \\ 2 & 3 & 4 & 5 & 4 & 5 & 5 \\ 3 & 3 & 2 & 2 & 1 & 2 & 3 \end{pmatrix}$.

3. $C = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 3 & 4 \\ 2 & 3 & 4 & 5 & 4 & 5 & 5 \\ 5 & 2 & 2 & 1 & 2 & 1 & 2 \end{pmatrix}$.

4. $C = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 3 & 4 \\ 2 & 3 & 4 & 5 & 4 & 5 & 5 \\ 4 & 3 & 3 & 1 & 1 & 2 & 2 \end{pmatrix}$.

5. $C = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 3 & 4 \\ 2 & 3 & 4 & 5 & 4 & 5 & 5 \\ 3 & 4 & 2 & 2 & 2 & 1 & 3 \end{pmatrix}$.

6. $C = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 3 & 4 \\ 2 & 3 & 4 & 5 & 4 & 5 & 5 \\ 5 & 3 & 2 & 1 & 1 & 3 & 2 \end{pmatrix}$.

7. $C = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 3 & 4 \\ 2 & 3 & 4 & 5 & 4 & 5 & 5 \\ 4 & 2 & 1 & 2 & 2 & 3 & 2 \end{pmatrix}$.

8. $C = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 3 & 4 \\ 2 & 3 & 4 & 5 & 4 & 5 & 5 \\ 3 & 3 & 3 & 1 & 2 & 2 & 1 \end{pmatrix}$.

9. $C = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 3 & 4 \\ 2 & 3 & 4 & 5 & 4 & 5 & 5 \\ 4 & 4 & 2 & 1 & 1 & 3 & 2 \end{pmatrix}$.

10. $C = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 3 & 4 \\ 2 & 3 & 4 & 5 & 4 & 5 & 5 \\ 5 & 2 & 3 & 2 & 1 & 2 & 1 \end{pmatrix}$.

11. $C = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 3 & 4 \\ 2 & 3 & 4 & 5 & 4 & 5 & 5 \\ 3 & 5 & 2 & 1 & 2 & 3 & 2 \end{pmatrix}$.

12. $C = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 3 & 4 \\ 2 & 3 & 4 & 5 & 4 & 5 & 5 \\ 4 & 3 & 1 & 2 & 2 & 2 & 3 \end{pmatrix}$.

13. $C = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 3 & 4 \\ 2 & 3 & 4 & 5 & 4 & 5 & 5 \\ 5 & 4 & 2 & 1 & 1 & 2 & 3 \end{pmatrix}$.

14. $C = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 3 & 4 \\ 2 & 3 & 4 & 5 & 4 & 5 & 5 \\ 3 & 2 & 3 & 2 & 1 & 3 & 2 \end{pmatrix}$.

15. $C = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 3 & 4 \\ 2 & 3 & 4 & 5 & 4 & 5 & 5 \\ 4 & 5 & 2 & 1 & 2 & 2 & 1 \end{pmatrix}$.

Задача 3.2: Максимальний потік, залишкова мережа і зворотна дуга

Для свого варіанта задано мережу з джерелом у вершині 1 та стоком у вершині 6. Мережу задано матрицею

$$C = \begin{pmatrix} i_1 & i_2 & \dots & i_m \\ j_1 & j_2 & \dots & j_m \\ d_{i_1 j_1} & d_{i_2 j_2} & \dots & d_{i_m j_m} \end{pmatrix},$$

де кожний стовпчик задає одну дугу мережі: у першому рядку записано початок дуги, у другому — її кінець, а в третьому — пропускну спроможність.

Потрібно:

1. побудувати граф мережі;
2. записати початковий нульовий потік;
3. виконати першу ітерацію методу Форда–Фалкерсона;
4. побудувати залишкову мережу після першої ітерації;
5. знайти в залишковій мережі збільшувальний ланцюг, який містить зворотну дугу;
6. виконати перерахунок потоку;
7. перевірити, чи є отриманий потік максимальним;
8. записати один із максимальних потоків і вказати його величину;
9. побудувати мінімальний розріз і перевірити рівність між величиною максимального потоку та пропускною спроможністю мінімального розрізу.

Варіанти:

1. $C = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 4 & 5 \\ 2 & 3 & 4 & 5 & 4 & 6 & 6 \\ 3 & 2 & 3 & 3 & 2 & 3 & 3 \end{pmatrix}$.

2. $C = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 4 & 5 \\ 2 & 3 & 4 & 5 & 4 & 6 & 6 \\ 4 & 1 & 4 & 3 & 2 & 4 & 2 \end{pmatrix}$.

3. $C = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 4 & 5 \\ 2 & 3 & 4 & 5 & 4 & 6 & 6 \\ 4 & 2 & 4 & 4 & 3 & 4 & 5 \end{pmatrix}$.

4. $C = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 4 & 5 \\ 2 & 3 & 4 & 5 & 4 & 6 & 6 \\ 5 & 2 & 5 & 3 & 4 & 5 & 3 \end{pmatrix}$.

5. $C = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 4 & 5 \\ 2 & 3 & 4 & 5 & 4 & 6 & 6 \\ 5 & 3 & 5 & 4 & 4 & 5 & 5 \end{pmatrix}$.

6. $C = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 4 & 5 \\ 2 & 3 & 4 & 5 & 4 & 6 & 6 \\ 3 & 1 & 3 & 2 & 2 & 3 & 2 \end{pmatrix}$.

7. $C = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 4 & 5 \\ 2 & 3 & 4 & 5 & 4 & 6 & 6 \\ 6 & 2 & 6 & 5 & 3 & 6 & 4 \end{pmatrix}$.

8. $C = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 4 & 5 \\ 2 & 3 & 4 & 5 & 4 & 6 & 6 \\ 4 & 2 & 4 & 2 & 2 & 4 & 3 \end{pmatrix}$.

9. $C = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 4 & 5 \\ 2 & 3 & 4 & 5 & 4 & 6 & 6 \\ 5 & 1 & 5 & 4 & 2 & 5 & 2 \end{pmatrix}$.

10. $C = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 4 & 5 \\ 2 & 3 & 4 & 5 & 4 & 6 & 6 \\ 6 & 3 & 6 & 5 & 4 & 6 & 6 \end{pmatrix}$.

11. $C = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 4 & 5 \\ 2 & 3 & 4 & 5 & 4 & 6 & 6 \\ 4 & 1 & 4 & 2 & 3 & 4 & 4 \end{pmatrix}$.

12. $C = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 4 & 5 \\ 2 & 3 & 4 & 5 & 4 & 6 & 6 \\ 5 & 2 & 5 & 5 & 3 & 5 & 4 \end{pmatrix}$.

13. $C = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 4 & 5 \\ 2 & 3 & 4 & 5 & 4 & 6 & 6 \\ 3 & 1 & 3 & 3 & 2 & 3 & 3 \end{pmatrix}$.

14. $C = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 4 & 5 \\ 2 & 3 & 4 & 5 & 4 & 6 & 6 \\ 6 & 2 & 6 & 4 & 5 & 6 & 3 \end{pmatrix}$.

15. $C = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 4 & 5 \\ 2 & 3 & 4 & 5 & 4 & 6 & 6 \\ 4 & 2 & 4 & 5 & 2 & 4 & 5 \end{pmatrix}$.

Прикладні задачі

Задача (IT) 3.1: Передавання відеопотоку в мережі доставки контенту

IT-коментар

У мережах доставки контенту (CDN)[a] задача максимального потоку дає змогу оцінити граничний обсяг відеотрафіку, який можна передати від центрального сервера до регіонального вузла без перевантаження окремих каналів. Така модель також допомагає виявити вузькі місця мережі та визначити, які канали слід модернізувати насамперед для підвищення якості потокового передавання.

[a] CDN (*Content Delivery Network*) — мережа доставки контенту, тобто розподілена система серверів для швидкого передавання цифрового вмісту користувачам.

Мережу доставки контенту подано орієнтованим графом, де вершини — це сервери та проміжні вузли маршрутизації, а дуги — канали передавання даних. Пропускна спроможність дуги відповідає максимальному обсягу відеоданих, який можна передати цим каналом за одну секунду.

Нехай вершина 1 — центральний сервер, а вершина 6 — регіональний вузол-одержувач. Потрібно методом Форда–Фалкерсона визначити максимальний потік відеоданих від сервера 1 до вузла 6.

Мережа задається матрицею

$$C = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 3 & 4 & 5 & 4 & 5 \\ 2 & 3 & 4 & 5 & 4 & 5 & 6 & 6 & 5 & 4 \\ 40 & 30 & 20 & 15 & 10 & 25 & 20 & 30 & 10 & 5 \end{pmatrix}.$$

Кожний стовпчик матриці задає одну дугу мережі: у першому рядку записано початок дуги, у другому — її кінець, а в третьому — пропускну спроможність каналу в Мбіт/с.

Потрібно:

1. зобразити мережу графічно у вигляді орієнтованого графа з пропускними спроможностями;
2. записати початковий нульовий потік;
3. методом Форда–Фалкерсона побудувати послідовність збільшувальних ланцюгів;
4. для кожної ітерації знайти величину приросту потоку θ;
5. визначити максимальний потік від вершини 1 до вершини 6;
6. побудувати один мінімальний розріз мережі;
7. інтерпретувати знайдений максимальний потік як максимальну пропускну спроможність CDN.

Задача (IT) 3.2: Розподіл задач у хмарній обчислювальній системі

IT-коментар

У хмарних обчисленнях модель максимального потоку використовується для оцінювання максимальної кількості задач, які система може передати від вузлів доступу до серверів обробки за одиницю часу.

Хмарну обчислювальну систему подано орієнтованим графом, де вершини — це шлюзи доступу, балансувальники навантаження, проміжні вузли та обчислювальні сервери, а дуги — канали передавання задач. Пропускна спроможність дуги відповідає максимальній кількості задач, яку можна передати за одну хвилину.

Нехай вершина 1 — центральний вузол приймання запитів, а вершина 7 — група обчислювальних серверів. Потрібно методом Форда–Фалкерсона визначити максимальну інтенсивність потоку задач від вузла 1 до вузла 7.

Мережа задається матрицею

$$C = \begin{pmatrix} 1 & 1 & 1 & 2 & 2 & 3 & 3 & 4 & 5 & 6 & 5 \\ 2 & 3 & 4 & 5 & 6 & 5 & 6 & 6 & 7 & 7 & 6 \\ 50 & 40 & 30 & 20 & 25 & 15 & 20 & 25 & 30 & 40 & 10 \end{pmatrix}.$$

Кожний стовпчик матриці задає одну дугу мережі: у першому рядку записано початок дуги, у другому — її кінець, а в третьому — пропускну спроможність у задачах за хвилину.

Потрібно:

1. побудувати граф мережі;
2. визначити методом Форда–Фалкерсона максимальний потік від вершини 1 до вершини 7;
3. записати всі проміжні потоки після кожної ітерації;
4. знайти мінімальний розріз мережі;
5. вказати дуги, які є критично важливими для збільшення потоку;
6. пояснити, які канали є вузькими місцями системи;
7. інтерпретувати результат як максимальну продуктивність хмарної інфраструктури.

Задача (IT) 3.3: Резервне копіювання даних у розподіленій мережі зберігання

IT-коментар

У системах резервного копіювання задача максимального потоку дає змогу оцінити, який найбільший обсяг даних можна передати між дата-центрами за обмежений час, не перевищуючи пропускних спроможностей каналів.

Мережу резервного копіювання подано орієнтованим графом, де вершини — це цен-

три зберігання даних і проміжні вузли передавання, а дуги — канали зв'язку між ними. Пропускна спроможність дуги відповідає максимальному обсягу даних, який можна передати цим каналом за одну годину.

Нехай вершина 1 — основний дата-центр, а вершина 6 — резервний центр зберігання. Потрібно визначити методом Форда–Фалкерсона максимальний обсяг даних, який можна передати за одну годину з вершини 1 до вершини 6.

Мережа задається матрицею

$$C = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 3 & 4 & 5 & 4 \\ 2 & 3 & 4 & 5 & 4 & 5 & 6 & 6 & 5 \\ 6 & 8 & 5 & 3 & 4 & 6 & 7 & 5 & 2 \end{pmatrix}.$$

Кожний стовпчик матриці задає одну дугу мережі: у першому рядку записано початок дуги, у другому — її кінець, а в третьому — пропускну спроможність у ТБ/год.

Потрібно:

1. зобразити мережу у вигляді орієнтованого графа;
2. знайти максимальний потік від вершини 1 до вершини 6;
3. побудувати хоча б один мінімальний розріз мережі;
4. перевірити виконання рівності між величиною максимального потоку та пропускною спроможністю мінімального розрізу;
5. визначити, які канали мають бути модернізовані насамперед для збільшення швидкості резервного копіювання;
6. інтерпретувати отриманий результат у контексті відмовостійкості системи зберігання.

Задача (IT) 3.4: Маршрутизація пакетів у програмно-керованій мережі

IT-коментар

У програмно-керованих мережах (SDN)[a] задача максимального потоку дає змогу оцінити межу пропускної спроможності магістрального сегмента і виявити канали, що обмежують інтенсивність маршрутизації пакетів.

[a] SDN (*Software-Defined Networking*) — програмно-керована мережа, тобто мережа, у якій керування передаванням даних логічно відокремлено від мережевого обладнання.

Програмно-керовану мережу (SDN) подано орієнтованим графом, де вершини — це маршрутизатори, а дуги — канали зв'язку між ними. Пропускна спроможність дуги відповідає максимальній кількості пакетів, яку можна передати через канал за одну мілісекунду.

Нехай вершина 1 — вхідний маршрутизатор, а вершина 7 — вихідний маршрутизатор магістрального сегмента. Потрібно визначити методом Форда–Фалкерсона максимальний потік пакетів від вершини 1 до вершини 7.

Мережа задається матрицею

$$C = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 3 & 4 & 5 & 6 & 4 & 5 \\ 2 & 3 & 4 & 5 & 5 & 6 & 7 & 7 & 7 & 5 & 6 \\ 12 & 10 & 7 & 5 & 6 & 5 & 8 & 6 & 5 & 3 & 4 \end{pmatrix}.$$

Кожний стовпчик матриці задає одну дугу мережі: у першому рядку записано початок дуги, у другому — її кінець, а в третьому — пропускну спроможність у пакетах за мілісекунду.

Потрібно:

1. побудувати граф мережі;
2. знайти максимальний потік від вершини 1 до вершини 7;
3. виписати збільшувальні ланцюги, які використовуються в алгоритмі;
4. визначити, які дуги стають насиченими після завершення алгоритму;
5. знайти мінімальний розріз мережі;
6. пояснити, як знайдений потік характеризує максимально можливу інтенсивність маршрутизації пакетів.

Задача (IT) 3.5: Потік даних у системі розподіленої відеоаналітики

IT-коментар

У системах розподіленої відеоаналітики задача максимального потоку дає змогу оцінити, який найбільший потік кадрів можна передати від вузлів збору даних до центрального сервера в режимі реального часу.

Систему розподіленої відеоаналітики подано орієнтованим графом, де вершини — це вузли збору відеоданих, сервери попередньої обробки, вузли розпізнавання та центральний сервер прийняття рішень. Дуги відповідають каналам передавання даних між цими вузлами. Пропускна спроможність дуги показує максимальну кількість відеокадрів, які можна передати за одну секунду.

Нехай вершина 1 — вузол збору відеоданих, а вершина 8 — центральний сервер прийняття рішень. Потрібно визначити методом Форда–Фалкерсона максимальний потік кадрів від вершини 1 до вершини 8.

Мережа задається матрицею

$$C = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 3 & 4 & 5 & 5 & 6 & 7 & 4 & 6 \\ 2 & 3 & 4 & 5 & 5 & 6 & 7 & 7 & 8 & 8 & 8 & 5 & 7 \\ 18 & 15 & 10 & 6 & 9 & 8 & 10 & 7 & 4 & 8 & 12 & 3 & 2 \end{pmatrix}.$$

Кожний стовпчик матриці задає одну дугу мережі: у першому рядку записано початок дуги, у другому — її кінець, а в третьому — пропускну спроможність у кадрах за секунду.

Потрібно:

1. побудувати граф мережі;

2. знайти максимальний потік від вершини 1 до вершини 8;
3. визначити мінімальний розріз мережі;
4. знайти критично важливі дуги для збільшення потоку;
5. встановити, чи збільшиться максимальний потік, якщо пропускну спроможність дуги $(5, 8)$ підвищити на 3;
6. інтерпретувати отримані результати в контексті роботи системи відеоаналітики в реальному часі.

Питання для самоконтролю

1. Сформулюйте означення мережі з джерелом і стоком. Що називають пропускною спроможністю дуги?

2. Що називають допустимим потоком на мережі? Які умови має задовольняти допустимий потік?

3. Що називають величиною потоку? Чому для допустимого потоку сума потоків, що виходять із джерела, дорівнює сумі потоків, що входять у стік?

4. Сформулюйте задачу про максимальний потік на мережі.

5. Що називають розрізом мережі, який відділяє джерело від стоку? Як визначається пропускна спроможність розрізу?

6. Що називають мінімальним розрізом мережі?

7. Сформулюйте теорему Форда–Фалкерсона про максимальний потік і мінімальний розріз.

8. У чому полягає ідея методу Форда–Фалкерсона?

9. Що називають збільшувальним ланцюгом? Які дуги можуть входити до збільшувального ланцюга?

10. Що називають залишковою мережею? Як визначаються залишкові пропускні спроможності для прямої та зворотної дуг?

11. Що означає позначка вершини $j(N_j, \theta_j)$ у методі позначок?

12. Як виконують перерахунок потоку після знаходження збільшувального ланцюга?

13. Які дуги називають критично важливими на збільшувальному ланцюгу? Яку роль вони відіграють у зміні потоку?

14. За якою ознакою встановлюють, що поточний потік уже є максимальним?

15. Як за множиною позначених вершин побудувати мінімальний розріз після завершення алгоритму?

16. Як перевірити правильність знайденого максимального потоку?

Список використаних джерел

[1] Бартіш М. Я., Дудзяний І. М. Дослідження операцій. Частина 1. Лінійні моделі : підручник. Львів : Вид. центр ЛНУ імені Івана Франка, 2007. 168 с.

[2] Бартіш М. Я., Дудзяний І. М. Дослідження операцій. Частина 2. Алгоритми оптимізації на графах. Львів : Вид. центр ЛНУ імені Івана Франка, 2007. 120 с.

[3] Дзюбан І. Ю., Жиров О. Л., Охріменко О. Г. Методи дослідження операцій. Київ : ІВЦ "Видавництво 'Політехніка"', 2005. 108 с.

[4] Зайченко Ю. П. Дослідження операцій : підручник. 7-ме вид., перероблл. та допов. Київ : Вид. дім "Слово", 2006. 816 с.

[5] Іксанов О. М., Шевченко В. І. Потоки на мережах (курс "Дослідження операцій") : навч. посіб. Київ : Наук. вид-во "ТВіМС", 2010. 46 с.

[6] Іксанов О. М., Шевченко В. І. Транспортна задача, її властивості та методи розв'язування (курс "Дослідження операцій") : навч. посіб. Київ : Наук. вид-во "ТВіМС", 2010. 84 с.

[7] Катренко А. В. Дослідження операцій : підручник. 3-тє вид., стер. Львів : "Магнолія 2006", 2006. 350 с.

[8] Ларіонов Ю. І., Левикін В. М., Хажмурадов М. А. Дослідження операцій в інформаційних системах : навч. посіб. 2-ге вид. Харків : Компанія СМІТ, 2005. 364 с.

[9] Наконечний С. І., Савіна С. С. Математичне програмування : навч. посіб. Київ : КНЕУ, 2003. 452 с.

[10] Нефьодов Ю. М., Балицька Т. Ю. Методи оптимізації в прикладах і задачах : навч. посіб. Київ : Кондор, 2011. 324 с.

[11] Попов Ю. Д., Тюптя В. І., Шевченко В. І. Методи оптимізації : навч. електрон. посіб. для студ. спец. "Прикладна математика", "Інформатика", "Соціальна інформатика". Київ : Електрон. б-ка ф-ту кібернетики КНУ імені Тараса Шевченка, 2003. 215 с.

[12] Попов Ю. Д., Тюптя В. І., Шевченко В. І. Методичні рекомендації до виконання лабораторних робіт з методів оптимізації на персональних комп'ютерах. Київ : Електрон. б-ка ф-ту кібернетики КНУ імені Тараса Шевченка, 2003. 53 с.

[13] Шевченко В. І., Тюптя В. І., Іксанов О. М. Методична розробка до проведення практичних занять з лінійного програмування. Київ : Електрон. б-ка ф-ту кібернетики КНУ імені Тараса Шевченка, 2003. 98 с.

[14] Ahuja R. K., Magnanti T. L., Orlin J. B. Network Flows: Theory, Algorithms, and Applications. Upper Saddle River : Prentice Hall, 1993.

[15] Hillier F. S., Lieberman G. J. Introduction to Operations Research. 10th ed. New York : McGraw-Hill Education, 2015.

[16] Taha H. A. Operations Research: An Introduction. 10th ed. Boston : Pearson, 2017.

Предметний покажчик