Тема 2. Окремі випадки застосування методів мережевого планування в управлінні бізнес-процесами

План

1. Мережеві математичні моделі у задачах управління бізнес-процесами
2. Основні поняття та визначення теорії графів
3. Задача мінімізації мережі
4. Задача про найкоротший шлях
5. Задача про максимальний потік

2.1. Мережеві математичні моделі у задачах управління бізнес-процесами

Значну кількість практичних задач з управління бізнес-процесами, математично можна описати, як задачі лінійного програмування. Для даного типу оптимізаційних задач, існує універсальний алгоритм їхнього вирішення – це симплексний метод.

Однак, достатньо велика кількість задач лінійного програмування при їхньому вирішенні симплексним методом, вимагає наявності значних обчислювальних потужностей й може займати тривалий час, що не виправдовує застосування даного методу.

Сукупність таких задач може бути математично описана й вирішена більш раціонально в рамках спеціальних теорій. Вони розробляють ефективні алгоритми

вирішення цих задач, об'єднаних в деякі однотипні класи. Таким чином, існує ряд практичних задач з управління бізнес-процесами, які зручно представляти, наприклад, у вигляді графічних структур (мережевих моделей).

Наприклад, можна формалізувати процес прийняття рішень з функціонування виробничої системи, транспортування продукції, передачі інформації тощо. Перераховані задачі можуть бути сформульовані й вирішені у вигляді задач лінійного програмування. Однак, у зв'язку з величезною кількістю змінних й обмежень, пряме застосування симплексного методу в мережевих задачах є недоцільним. Особлива структура таких задач дозволяє розробити більш ефективні алгоритми їхнього вирішення.

Розглянемо детальніше типові приклади спеціальних ЗЛП й методи їхнього розв’язання, а саме: модифікацію транспортної задачі й її постановку на мережевих моделях (графах).

Транспортна задача й її можливі економічні інтерпретації – це лише одна з багатьох задач, які можуть бути сформульовані й вирішені за допомогою мережевих моделей.

Приклад 1. Задача мінімізації мережі.

Дана оптимізаційна задача виникає в різних сферах управління бізнес-процесами, коли необхідно визначити максимально коротке з'єднання двох й більше заданих об'єктів (наприклад, найкоротший маршрут між обраними містами, розташування

двох елементів з мінімальною відстанню на електронній схемі). *Розглянемо нижче змістовну постановку типової задачі мінімізації мережі.*

Припустимо, в деякому місті, населення якого перевищило 1 мільйон осіб, планується будівництво метрополітену, який повинен з'єднати центр міста й п'ять його районів підземною залізничною колією. Оцінивши можливі варіанти будівництва тунелів, були обрані найбільш ефективні, з точки зору переслідуваної мети. Маршрути прокладання тунелів повинні бути обрані таким чином, щоб мета будівництва досягалась за критерієм мінімальних витрат на будівництво, або мінімальної довжини тунелів. Й щоб кожен з районів міста був напряму з'єднаний підземною залізничною колією з його центром, або через інші райони.

Приклад 2. Задача про найкоротший шлях.

Оптова компанія здійснює продаж товарів через торгівельну мережу магазинів. Відомі всі можливі маршрути доставки товарів зі складу цієї компанії в кожен з магазинів, а також транспортні витрати на кожний маршрут (або, наприклад, довжина маршрутів). Для того щоб скоротити загальні витрати (або, сумарний кілометраж) на доставку товарів до магазинів, керівництву компанії слід обрати таку множину маршрутів, яка би дозволяла доставляти товари зі складу в кожен з магазинів безпосередньо, або через інші магазини, щоб мінімізувати витрати компанії.

Приклад 3. Задача про максимальний потік в мережі.

Дана задача виникає кожного разу, коли через певну мережу пропускається будь-який матеріальний, фінансовий, або інформаційний потік. При цьому, необхідно знайти такий розподіл елементів потоку по наявних каналах зв'язку, щоб за одиницю часу передавався його максимальний обсяг.

Наприклад, газова компанія з Азербайджану підписала контракт на постачання нафтопродуктів до Німеччини трубопроводом. Транспортування нафтопродуктів можливе через трубопроводи та паромні станції, що розташовані на території України. Пропускна здатність трубопроводів на різних ділянках також є різною. Перед керівництвом компанії постає питання: який максимально можливий обсяг нафтопродуктів можна транспортувати по існуючій мережі в одиницю часу?

Аналіз цих прикладів показує, що оптимізаційні мережеві задачі можуть бути описані наступними типами моделей:

1) мінімізація мережі (випадок 1);
2) знаходження найкоротшого шляху (випадок 2);
3) визначення максимальної пропускної здатності (випадок 3).

Всі розглянуті приклади мережевих задач можуть бути сформульовані й вирішені, як задачі лінійного програмування. Однак, у зв'язку з величезною кількістю змінних й обмеженістю обчислювальних можливостей, пряме застосування симплексного методу є недоцільним. Особлива структура цих задач дозволяє

створювати більш ефективні алгоритми, які в більшості випадків базуються на теорії лінійного програмування.

Аналіз графічних структур дозволяє знаходити оптимальні рішення цих задач. Одним з таких представлень є мережа (граф). Теорія, яка розвиває алгоритми розв'язання задач на мережах (або графах), називається теорією графів. У загальному випадку, граф – це пара множин, елементи однієї з яких називаються вершинами, а інші – ребрами (або дугами).

2.2. Основні поняття та визначення теорії графів

Звичайний граф (мережа) G = (V, E) – це впорядкована пара множин, рис. 1, скінченної непорожньої множини V, елементи якої називаються **вершинами графу** G й довільної підмножини E $\subseteq$ <V$\times$V>, елементи якої називаються **ребрами** цього графа (мережі).

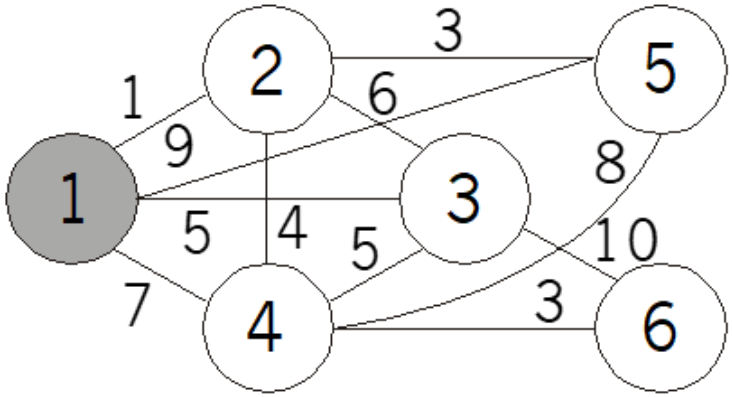


Рис. 1. Приклад звичайного графу

Основними властивостями звичайного графу є:

1. Множина ребер є обмеженою;
2. Ребра графу є неорієнтованими;
3. В графі відсутні петлі, тобто ребра виду $l = (v1, v1)$;
4. Граф G не містить кратних ребер (тобто $(v1, v2) = (u1, u2)$, якщо $v1 = u1$ та $v2 = u2$);

Два крайніх випадки звичайних n-вершинних графів:

1. Безреберний граф, де $E = \varnothing$;
2. Повний граф, де $E = <V \times V>$, тобто будь-які дві його вершини є суміжними.

Далі, на основі рис. 1, розглянемо основні поняття й визначення теорії графів.

Послідовність вершин й ребер графу $v0\ (v0, v1)\ v1\ (v1, v2)\ v2\ \ldots\ vn$ називається **маршрутом**, що з'єднує вершини $v0$ та vn.

Маршрут називається **ланцюгом,** якщо всі його ребра є різними (не повторюються):

1 (1, 2) 2 (2, 3) 3 (3, 4) 4 (4, 2) 2 (2, 5) 5.

Ланцюг називається **простим**, якщо всі його вершини є різними (не повторюються):

1 (1, 4) 4 (4, 5) 5.

Ланцюг, в якому початкова вершина *v0* співпадає з кінцевою вершиною *vn* й всі ребра є різними, називається **циклом**:

1 (1, 2) 2 (2, 3) 3 (3, 4) 4 (4, 2) 2 (2, 5) 5 (5, 1) 1.

Цикл називається **простим**, якщо всі його вершини є різними, за винятком початкової *v0* й кінцевої *vn*, які є однаковими:

1 (1, 4) 4 (4, 2) 2 (2, 5) 5 (5, 1) 1.

Граф (мережа) називається **зв'язаним,** якщо будь-які дві його неспівпадаючі вершини з'єднані маршрутом (граф на рис. 1 є зв'язаним). В іншому випадку, він є незв’язаним, рис. 2.

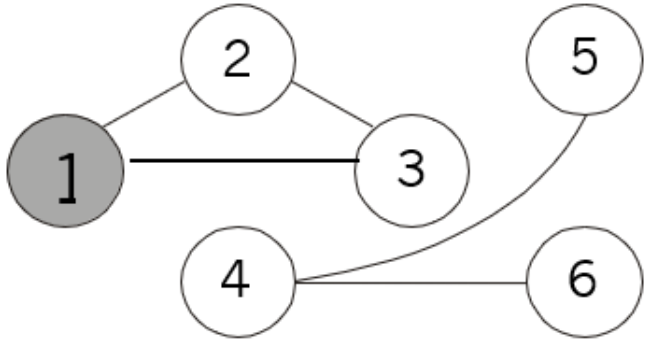


Рис. 2. Приклад незв’язаного графу

Ребро, після видалення якого граф зі зв’язаного перетворюється на незв’язаний, називається **мостом**.

Граф називається **зваженим**, якщо кожному з його ребер відповідає певне число ω(i), яке називається **вагою ребра**.

Тоді, під **вагою графу** розуміють суму ваг всіх його ребер, тобто:

$\boldsymbol{\omega(G) = \sum \omega(i)}$.

Зв’язаний граф, який не містить циклів, називається **деревом**, рис. 3.

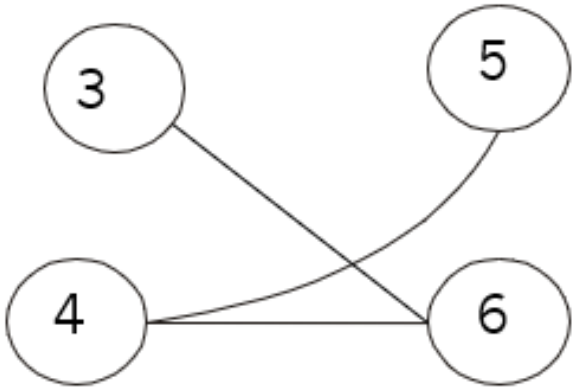


Рис. 3. Приклад дерева

Орієнтований граф – це пара множин (V, A), де V – множина вершин; A – множина орієнтованих ребер, які називаються **дугами**. Приклад орієнтованого графу наведений на рис. 4.

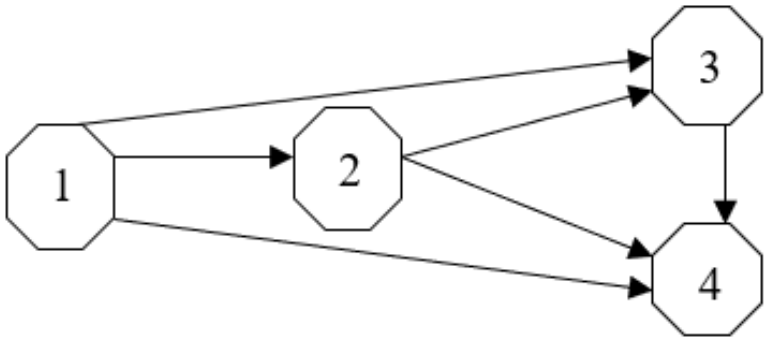


Рис. 4. Орієнтований граф

Якщо a = (v1, v2) – це дуга, то вершини *v1* й *v2* називаються її **початком** та **кінцем** відповідно.

Якщо кожна дуга графу має певну вагу, то такий граф називається **орієнтованим зваженим**.

2.3. Задача мінімізації мережі

Задача мінімізації мережі полягає в пошуку ребер, які з'єднують між собою всі вузли мережі (тобто, всі вершини графу) й мають мінімальну загальну довжину, або вагу. Очевидно, що оптимальне вирішення цієї задачі не повинно містити циклів.

Для її вирішення існують ефективні алгоритми, що застосовуються до довільного зв'язного графа.

Алгоритм Краскала:

1. Будуємо граф $T_1 = O_n + l_1$, приєднавши ребро l_1 мінімальної ваги до порожнього графа O_n на множині вершин V графа G. Якщо таких ребер декілька, тобто $\omega(l_1) = \omega(l_2) = min\ \omega(l)$, тоді обирається будь-яке з цих ребер. Це вказує на неоднозначність рішення (оптимальних рішень може бути декілька).

2. Якщо ми вже маємо побудований граф T_i, причому $i < (n - 1)$, то будуємо граф $T_{i+1} = T_i + l_{i+1}$, де l_{i+1} – ребро графу G, яке має мінімальну вагу серед ребер, що не входять до складу T_i й не утворюють з ними циклів.

3. В іншому випадку, якщо $i = (n - 1)$, то алгоритм завершує свою роботу. Тобто, ми побудували граф мінімальної ваги, який охоплює всі вершини.

Даний алгоритм на кожному кроці будує ациклічний граф, який на останньому кроці стає зв’язаним.

Розглянемо ще один алгоритм для вирішення задачі про мінімізацію мережі.

Алгоритм Прима:

1. Будуємо граф $T_1 = O_n + l_1$, приєднавши ребро l_1 мінімальної ваги до порожнього графа O_n на множині вершин V графа G. Якщо таких ребер декілька, тобто $\omega(l_1) = \omega(l_2) = min\ \omega(l)$, тоді обирається будь-яке з цих ребер. По аналогії, це вказує на неоднозначність рішення (оптимальних рішень може бути декілька).

2. Якщо ми вже маємо побудований граф T_i, причому $i < (n - 1)$, то будуємо граф $T_{i+1} = T_i + l_{i+1}$, де l_{i+1} – ребро графу G мінімальної ваги, яке додається до однієї з вершин цього графу T_i.

3. В іншому випадку, якщо $i = (n - 1)$, то алгоритм завершує свою роботу. Тобто, ми побудували граф мінімальної ваги, який охоплює всі вершини.

На відміну від алгоритму Краскала, алгоритм Прима будує на кожному кроці зв'язний ациклічний граф.

Примітка

У деяких ситуаціях доводиться вирішувати задачу з максимізації мережі, будуючи зв’язаний граф не мінімальної, а максимальної ваги. В таких випадках, алгоритми Краскала й Прима також можуть використовуватись. Необхідно лише всюди замінити максимальні ваги ребер на максимальні.

Приклад 4. Телекомунікаційна компанія планує створити кабельну мережу для обслуговування п'яти нових будинків, рис. 6.

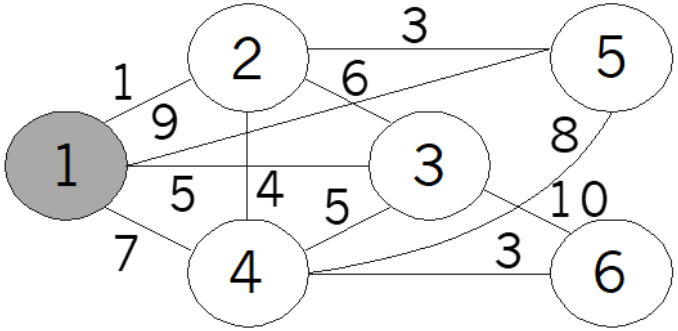


Рис. 6. Схема можливої прокладки кабелів

Цифри на ребрах вказують на довжину кабелю (км), що з'єднує відповідні вершини графу. Вузол №1 представляє собою ретранслятор сигналу, а решта вузлів (№2 – №6) відповідають п'яти новим будівлям.

Необхідно знайти такі ребра, які з’єднають всі вершини графу між собою, причому, вага графу повинна бути мінімальною. Таким чином, маємо задачу мінімізації мережі.

Рішення

Алгоритм Краскала:

Упорядкуємо ребра графу, рис. 6, в порядку зростання їхніх ваг:

Ребра графу	Ваги ребер	Додаємо (+), або не додаємо (–) ребра
(1,2)	1	+
(4,6)	3	+
(2,5)	3	+
(2,4)	4	+
(1,3)	5	+ (–)
(3,4)	5	– (+)
(2,3)	6	–
(1,4)	7	–
(4,5)	8	–
(1,5)	9	–
(3,6)	10	–

Знаком «+» в таблиці позначені ті ребра, які ми включили до складу графу. Одне з ребер (1, 3), або (3, 4) можуть бути включені до даного графу.

Таким чином, мінімальна довжина кабелю, що з'єднує ретранслятор сигналу з новозбудованими будівлями, дорівнює:

$\omega = 1 + 3 + 3 + 4 + 5 = 16$ (км).

Алгоритм Прима:

Логічним є розпочати вирішення даної задачі з вершини №1, де розташований ретранслятор сигналу.

T_1: V_1 ={1, 2}, E_1={(1, 2)} (множина вершин та множина ребер, відповідно);

$T_2 = T_1 \cup (2, 5)$: V_2 ={1, 2, 5}, E_2 = {(1, 2), (2, 5)};

$T_3 = T_2 \cup (2,4)$: V_3 = {1, 2, 4, 5}, E_3 = {(1, 2), (2, 5), (2, 4)};

$T_4 = T_3 \cup (4,6)$: V_4 = {1,2,4,5,6}, E_2 = {(1, 2), (2, 5), (2, 4), (4, 6)};

$$T_5 = \begin{cases} T_4 \cup (1,3)\text{: } V_5 = \{1,2,3,4,5,6\},\ E_5 = \{(1,2), (2,5), (2,4), (4,6), (1,3)\} \\ T_4 \cup (3,4)\text{: } V_5 = \{1,2,3,4,5,6\},\ E_5 = \{(1,2), (2,5), (2,4), (4,6), (3,4)\} \end{cases}$$

Оскільки $i = (n - 1) = 6 - 1 = 5$, то ітераційний процес побудови зв'язаного графу завершується, рис. 7.

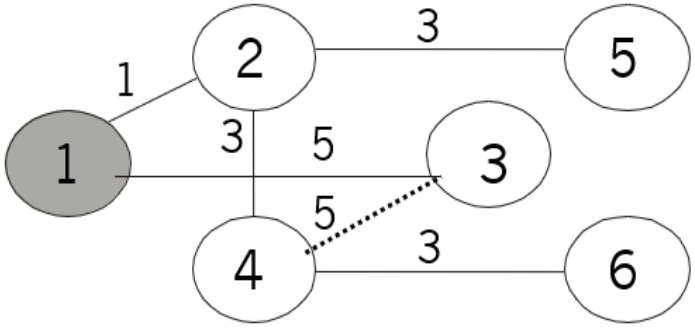


Рис. 7. Вирішення задачі мінімізації мережі

Таким чином, оптимальна мережа підключень нових будинків до ретранслятору, показана на рис. 7. Для її практичної реалізації, необхідний кабель мінімальної довжини у 16 км.

2.4. Задача про найкоротший шлях

Задача про найкоротший шлях зазвичай формулюється на орієнтованих графах.

Нехай $G=(V,A)$ — орієнтований зважений граф. ***Задача на найкоротший шлях*** полягає в тому, щоб знайти шлях мінімальної ваги, що з'єднує задані початкову і кінцеву вершини графа G , за умови, що існує хоча б один шлях.

Початкова і кінцева вершини позначаються відповідно s і t; (s,t)-шлях мінімальної ваги буде називатися ***найкоротшим (s,t)-шляхом.***

Розглянемо випадок, коли ваги всіх дуг графіка невід'ємні, тобто ($\omega(l)\geq 0$, $\forall l \in A$). Існує ефективний алгоритм побудови найкоротшого шляху в графі з невід'ємними вагами дуг, запропонований Е. Дейкстрою в 1959 році.

Суть алгоритму. На кожній ітерації цього алгоритму кожна вершина v графа G має мітку $l(v)$, яка може бути *постійною* або *тимчасовою*. У першому випадку $l(v)$ — вага найкоротшого (s,v) шляху. Якщо *мітка* $l(v)$ тимчасова, то $l(v)$ — вага найкоротшого (s,v)-шлях, який проходить тільки через вершини з постійними мітками. Таким чином, мітка часу $l(v)$ є оцінкою ваги найкоротшого (s,v) шляху зверху вниз, і ставши постійною на деякій ітерації, вона залишається такою до кінця

роботи алгоритму. Крім $l(v)$, кожна вершина v графа G, за винятком θs, пов'язана з іншою міткою - $\theta(v)$. — це номер вершини, що передує v у (s,v)-шляху, який має найменшу вагу серед усіх (s,v)-шляхів, що проходять через вершини, які отримали постійні мітки до цього часу. Після того, як вершина t отримала постійну мітку, легко використовувати мітки $\theta(v)$ для вказівки послідовності вершин, які складають найкоротший (s,v)-шлях.

Перед початком першої ітерації алгоритму вершина S має константну мітку $l(s)=0$, а l міток всіх інших вершин дорівнюють нескінченності і ці мітки є тимчасовими. Загальна ітерація алгоритму виглядає наступним чином. Нехай p — вершина, яка отримала постійну мітку $l(p)$ на попередній ітерації. $\in$Часові позначки для зменшення (якщо можливо) цих часових позначок. тут $G(p)$ — множина всіх вершин, які мають часові мітки і є суміжними з вершиною p. Мітки $l(v)$ вершини $\in v$ $G(p)$ замінюються на $l(p)+\omega(p,v)$, якщо виявляється, що $l(v)>l(p)+\omega(p,v)$. У цьому випадку ми говоримо, що вершина v отримала свою *мітку* l від вершини p, і припускаємо $\theta(v)=p$.

Якщо $l(v) \le l(p) + \omega(p,v)$, то мітки θ і l вершини v в цій ітерації не змінюються. Алгоритм завершується, коли *мітка* $l(t)$ стає постійною. *Тоді* $l(t)$ – вага найкоротшого (s,t)-шляху, який позначається P^*. Цей шлях визначається за допомогою θ міток наступним чином: $P^* = (c,..., \theta^3(m), \theta^2(m), \theta(m), m)$.

Алгоритм Дейкстри:

1. Поставте $l(s)=0$ і вважайте цю мітку постійною. Поставте $l(v)=\infty$, $\forall v \in V$, $v\, s \neq$ і вважайте ці мітки тимчасовими. Поставте $p = s$.

2. $\forall v \in G(r)$ з позначками часу для виконання:

а) якщо $l(v) > l(p) + \omega(p,v)$, *то* $l(p) = l(p) + \omega(p,v)$; $\theta(в) = p$;

б) в іншому випадку $l(v)$ і $\theta(v)$ не повинні змінюватися.

3. Нехай V' — множина вершин з часовими мітками l. Знайти вершину v^* таку, що $l(v^*)=min(v), v \in V'$. Вважайте, що мітка $l(v^*)$ є постійною міткою *вершини* v^*.

4. $p = в^*$. Якщо $p = m$, то переходимо до пункту 5. В іншому випадку переходимо до пункту 2.

5. $P^* = (c,..., \theta^3(m), \theta^2(m), \theta(m), m)$ - найкоротший шлях. Кінець.

Зауваження:

1. Алгоритм Дейкстри застосовний до змішаних графів і, зокрема, до неорієнтованих графів. Для цього достатньо розглянути кожне неорієнтоване ребро графа uv, що має вагу $\omega(u,v)$ як пару дуг (u,v) і (v,u) однакової ваги.

2. Якщо крок 4 модифікувати так, що алгоритм закінчується тільки після того, як всі вершини отримали постійні мітки, то він побудує найкоротші шляхи від вершини s до кожної з інших вершин.

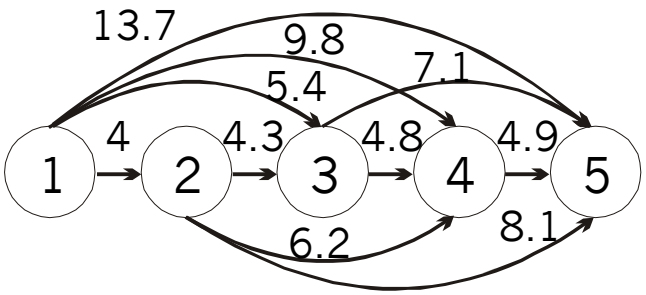
Приклад: Проблема із заміною обладнання.

Змістовна постановка проблеми

Компанія з прокату автомобілів планує замінити автопарк на найближчі п'ять років. Автомобіль повинен прослужити не менше одного року, перш ніж компанія поставить питання про його заміну. У таблиці вказана вартість заміни авто (в тисячах доларів) в залежності від часу заміни і кількості років, протягом яких машина перебувала в експлуатації.

Побудова математичної моделі

Це завдання може бути представлено в мережі наступним чином (див. Рис. 8). Щороку присвоюється певний вузол. Довжина дуги, що з'єднує два вузли, дорівнює відповідній вартості заміщення з таблиці. Задача полягає в тому, щоб знайти найкоротший шлях між вузлами 1 і 5.



Рік	1	2	3	4	5
1		4.0	5.4	9.8	13.7
2			4.3	6.2	8.1
3				4.8	7.1
4					4.9

Рис. 8. Завдання по заміні обладнання (автопарку)

Розв'язання: Розв'язуємо цю задачу за допомогою побудованої моделі за алгоритмом Дейкстри:

1) $l(1) = 0, 1^*, l(2) = \ldots = l(5) = \infty$;

2) $\left.\begin{array}{l} l(2)=4,\ \theta(2)=1 \\ l(3)=5{,}4,\ \theta(3)=1 \\ l(4)=9{,}8,\ \theta(4)=1 \\ l(5)=13{,}7,\ \theta(5)=1 \end{array}\right\} \Rightarrow \min\limits_{v=2,3,4,5} l(v)=l(2),\ 2^*,\ \theta(2)=1;$

3) $\left.\begin{array}{l} l(3)=5{,}4,\ \theta(3)=1 \\ l(4)=9{,}8,\ \theta(4)=1 \\ l(5)=12{,}1,\ \theta(5)=2 \end{array}\right\} \Rightarrow \min\limits_{v=3,4,5} l(v)=l(3),\ 3^*,\ \theta(3)=1;$

4) $\left.\begin{array}{l} l(4)=9{,}8,\ \theta(4)=1 \\ l(5)=12{,}1,\ \theta(5)=2 \end{array}\right\} \Rightarrow \min(l(4),l(5))=l(4),\ 4^*,\ \theta(4)=1;$

5) $l(5)=12{,}1,\quad \theta(5)=2,\ 5^*.$

$P^*= (1, 2, 5)$ – найкоротший шлях від вершини 1 до вершини 5. Його довжина буде мінімальною і складе 12,1.

Відповідь: Таким чином, оптимальним рішенням є 1→2 5→з вартістю $12,1 тис. Це означає, що кожен автомобіль замінюється через два роки, а через п'ять років списується.

2.5. Задача про максимальний потік

Нехай $G = (V, A)$ - орієнтований граф (мережа).

$S' \in V$ називається ***джерелом***, якщо немає дуг, що закінчуються на S'.

$S \in V$ називається ***стоком,*** якщо немає дуг, що починаються в точці S.

Припустимо, *що є рівно одне джерело* S/ *і один стік* S *в* G.

Функція φ: $A \to R+$ - це ***ємність***, тобто ємність R+. $\varphi(l)$ - ємність дуги l, яка визначає максимальну величину потоку, що проходить через дугу l.

Потоком на графіку є функція μ: $A \to R+$, яка має властивості:

1) $\forall l \in A$: $0 \le \mu(l) \le \varphi(l)$, де $\mu(l)$ — потік через дугу l;

2) $\forall y \in B$: $y \neq c/$ і $y \neq c$: $\sum_{e\,вх.ви} \mu(\lambda) = \sum_{e\,исх.изи} \mu(\lambda)$.

Величина потоку називається числом $\rho(\mu) = \sum_{e\,исх.из\,s'} \mu(\lambda)$.

Теорема: У графіку $G = (V, A)$ сумарний потік по всіх дугах, що вийшли з джерела, дорівнює сумарному потоку по всіх дугах, що увійшли в потік, тобто:

$$\sum_{e\,исх.из\,s'} \mu(\lambda) = \sum_{e\,вх.в\,s''} \mu(\lambda)$$

Формулювання задачі максимальної витрати:

Нехай *у* $G = (V, A)$ є одне джерело і один стік. φ: $A \to R+$ - це смуга пропускання дуг цього графіка. $M = \{\mu\}$ - це множина всіх потоків через графік G, $\rho(\mu)$ є значенням потоку.

Серед усіх потоків через графік G знайдіть витрату максимального значення, тобто:

$$\rho(\mu) \to \max, \quad где\ \mu \in M\ .$$

Алгоритм Форда-Фулкерсона (знаходження максимального потоку)

1. Нехай φ— ($л$)>0, $\forall л \in A$; $\mu_0(л)$=0; $K=1$.
2. Знайдіть довільний спрямований шлях від S' до S на графіку $G=(V\ \varphi_{k,Ak,k})$ і побудуйте μ_k: $A \to R+$ за наступним правилом.

Виберіть номер $\alpha_k = \min\limits_{\lambda \in пути} \varphi_k(\lambda)$ і побудуйте графік потоку, що пройшов через дуги обраного шляху в цій ітерації:

$$\mu_к(\lambda) = \begin{cases} \mu_{k-1}(\lambda) + \alpha_к, & если \quad \lambda \in пути, \\ \mu_{к-1}(\lambda), & если \quad \lambda \notin пути. \end{cases}$$

Після цього перераховуємо залишкову пропускну здатність дуг:

$$\varphi_{к+1}(\lambda) = \begin{cases} \varphi_к(\lambda) - \alpha_к, & если \quad \lambda \in пути, \\ \varphi_к(\lambda), & если \quad \lambda \notin пути. \end{cases}$$

3. Дуги, для яких $\varphi_{k+1}(\lambda)=\varnothing$ видаляються з $Gk+1=(V,A_{k+1},k\varphi_{+1})$.

4. Якщо на кроці *k* в графі *G* неможливо знайти спрямований шлях від S' до *S*, то зупинка. Таким чином, $\mu_k(\lambda)$ — шуканий потік, а $\rho_{\max} = \mu_0(\lambda) + \sum \alpha_k$ – *величина потока.*

В іншому випадку перейдемо до пункту 2.

Зауваження

Якщо в графі $G = (V, A, \varphi)$ є два джерела S'_1 і S'_2, то додаємо до множини вершин графа вигадану вершину S' і дуги (S', S'_1) і (S', S'_2); припускаємо:

$$\varphi(S', S'_1) = \sum_{e \text{ исх. из } S'_1} \varphi(\lambda) \quad , \quad \varphi(S', S''_1) = \sum_{e \text{ исх. из } S'_2} \varphi(\lambda) .$$

Те ж саме справедливо і для двох мийок S_1 і S_2. Для перетвореного графіка застосовний алгоритм Форда-Фулкерсона для знаходження максимального потоку в мережі.

Поняття мережевого перетину пов'язане з поняттям потоку. Під ***ділянкою мережі*** розуміють сукупність дуг мережі Р *(А),* які мають таку властивість: будь-який шлях від джерела до потоку буде проходити хоча б по одній дузі ділянки.

Розмір ділянки С(П) дорівнює сумі смуг пропускання входять до нього дуг: $C(P) = \sum_{e \in P(A)} \varphi(\lambda)$. Природно, необхідно знайти розріз мінімального розміру.

У мережі величина максимального потоку дорівнює значенню мінімального зрізу, т. Е.

$$\max \rho(\mu) = \min C(P).$$

Приклад

Змістовна постановка проблеми

Двоє користувачів інтернету обмінюються один з одним якоюсь інформацією. Їх робочі станції пов'язані між собою через мережу серверів. Більш того, передача інформації можлива за допомогою різних комбінацій серверів і каналів зв'язку, що з'єднують їх.

Яка максимально можлива кількість інформації за одиницю часу може бути передана по даній мережі від одного користувача до іншого, якщо відомо кількість інформації, яку здатний передати кожен канал зв'язку.

Побудова математичної моделі

Припустимо, що користувачі (S', S) з'єднані через 5 серверів (вузлів мережі) каналами зв'язку (дугами). Припустимо, що відомо також кількість інформації, яку кожен канал здатний передати за одиницю часу (смуги пропускання - це числа вище дуг).

Тоді початкову задачу можна формалізувати у вигляді моделі мережі, показаної на рисунку 9:

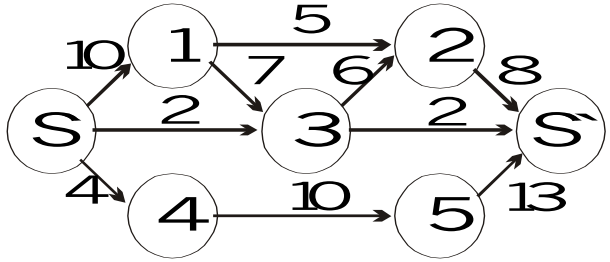


Рис. 9 Математическая модель задачи об информационном потоке в сети Internet

Умовно приймемо вимогу односпрямованості інформаційного потоку в даній мережі, наприклад, $S' \to S$ (оптимальна схема передачі інформації $S \to S'$ буде, очевидно, такою ж).

Ця проблема може бути вирішена як задача максимального потоку. Скористаємося алгоритмом Форда-Фулкерсона для знаходження максимального потоку в мережі.

Рішення:

Знайти довільний спрямований шлях, що з'єднує вузли S' і S.

1) $З' \to 1 \to 3 \to 2 \to С$.

Виберіть мінімальну пропускну здатність з усіх смуг пропускання дуг, що входять в обраний шлях: $\alpha_1 = \min\{10, 7, 6, 8\} = 6$.

Це величина потоку, що проходить по цьому шляху. Запишіть α у дужках 1 = 6 над дугами, включеними в цей шлях, а решту пропускної здатності за дужками запишіть як різницю між пропускною здатністю на попередньому кроці та значенням пропущеного потоку $\alpha_1 = 6$.

Дуга, яка має нульову залишкову смугу пропускання, виключається з розгляду. Далі шукаємо якийсь інший шлях, що з'єднує S' і S, і так далі, поки не вдасться знайти орієнтований шлях від S' до S.

Процес вирішення цієї задачі представлений нижче, а проілюстрований на малюнку 10.

2) $S' \to 4 \to 5 \to S''$, $\alpha_2 = \min\{4, 10, 13\} = 4$;

3) $S' \to 1 \to 2 \to S''$, $\alpha_3 = \min\{4, 5, 2\} = 2$;

4) $S' \to 3 \to S''$, $\alpha_4 = \min\{2, 2\} = 2$.

Після четвертого кроку алгоритм завершує свою роботу, так як шляхів, що з'єднують вузли S' і S, більше немає.

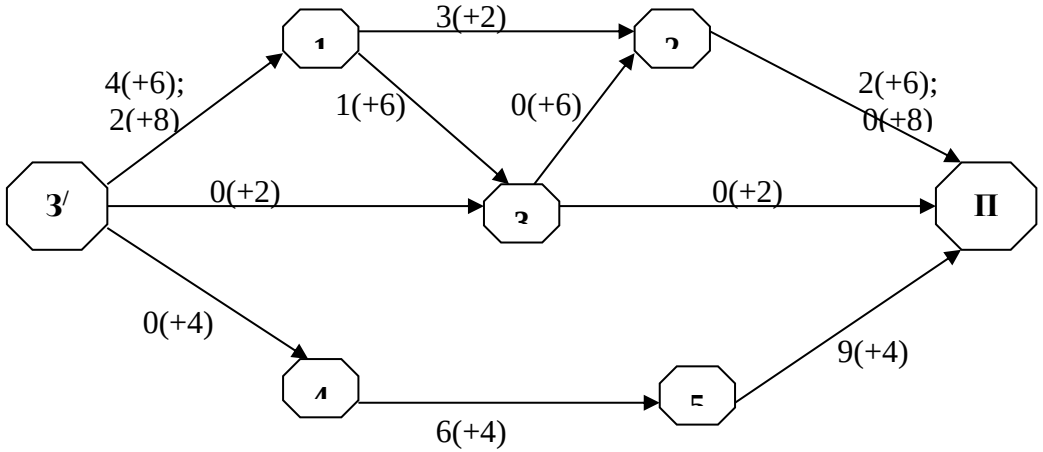


Рис. 10. Вирішення проблеми максимального потоку

Максимальне значення потоку:

$\rho_{max}= \alpha_1+\alpha_2+\alpha_3+\alpha_4= 6+4+2+2= 14,$

А сама максимальна витрата (оптимальне рішення задачі) представлена на малюнку 11.

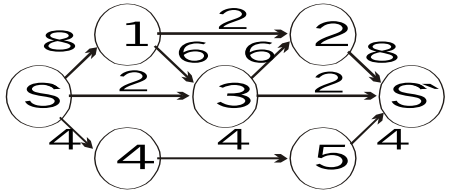


Рис. 11 Оптимальное решение задачи об информационном потоке в сети Internet

Над дугами записуються значення потоку, що пройшов через кожну з них.

Таким чином, аналізуючи оптимальне рішення цієї задачі, приходимо до наступного висновку: максимальна кількість інформації, яка може бути передана за одиницю часу по даній інформаційній мережі від робочої станції S' до робочої станції S (або в зворотному напрямку), дорівнює 14 одиницям обсягу інформації.

Задача максимального потоку може бути сформульована у вигляді задачі лінійного програмування (LPP). Однак вирішення мережевих проблем за допомогою симплексного методу не є доцільним. З іншого боку, вивчення формулювань мережевих завдань як ПЛП допомагає виявити моделі ЛП, які на перший погляд не є мережевими, але які можуть бути зведені до мережевих або безпосередньо, або з

деякими модифікаціями. Перевага такого підходу полягає в тому, що при використанні мережевої постановки ефективність обчислень може значно зрости.

Побудуємо математичну лінійну модель задачі максимального потоку.

Змінні моделі

Нехай x_{ij} — потік, що пройшов через дугу *(i; j)*, тобто змінні лінійної моделі відображаються на кожну дугу мережевої моделі.

Цільова функція

Цільова функція моделі повинна формалізувати мету задачі – потік, що вийшов з джерела, повинен бути максимальним. Сумарний потік, що вийшов з джерела $S/$ $\sum_j X_{S/j}$ за введеними змінними, дорівнює , де підсумовування здійснюється по всіх дугах, починаючи від джерела $S/$. Тоді цільову функцію можна записати у вигляді:

$$\rho = \sum_j X_{S/j} \to \max$$

Система обмежень

За техніко-економічним змістом введених змінних на них повинні бути накладені наступні обмеження.

1. *Загальний потік, що покинув джерело $S/$, дорівнює загальному потоку, що надійшов у потік* S. Математично це можна записати так:

$$\sum_{(S^{/};\, j)} X_{S^{/} j} = \sum_{(i;\, S^{//})} X_{iS^{//}}$$

2. Для будь-якої проміжної вершини мережі k сумарний потік, що увійшов в неї, дорівнює загальному потоку, який вийшов з неї. Математично це записується так:

$$\sum_{(i;\, k)} X_{ik} = \sum_{(k;\, j)} X_{kj}$$

3. З визначення поняття потоку в мережі випливає наступна умова:

$$0 \le X_{ij} \le \varphi_{ij}$$

де φ_{ij} - несуча здатність дуги *(i; j).*

Таким чином, математична лінійна модель задачі максимального потоку буде виглядати наступним чином:

$$\rho = \sum_{j} X_{S'j} \rightarrow \max$$

$$\begin{cases} \sum\limits_{(S';\, j)} X_{S'j} = \sum\limits_{(i;\, S'')} X_{iS''} \\ \sum\limits_{(i;\, k)} X_{ik} = \sum\limits_{(k;\, j)} X_{kj} \\ 0 \le X_{ij} \le \varphi_{ij} \end{cases}$$