**УДК 519.178**

**О. Ю. Бережний, В. В. Козлов, В. Р. Судаков**

**Оптимізація транспортних перевезень**

Розглянуто основні складові транспортних перевезень, як частини виробництва, а також основні функції та завдання логістичних служб. Проаналізовано основні принципи оптимізації перевезень, що можуть допомогти визначити найбільш раціональний маршрут для зниження транспортних витрат. Як наслідок, для підвищення конкурентоспроможності був запропонований алгоритм Дейкстри – алгоритм на графах, що здатен сформувати оптимальний маршрут серед запропонованих напрямків руху.

# Введення

Сьогодні, при ринковому типу економіки, основним критерієм успішності підприємства є його конкурентоспроможність.

Конкурентоспроможність підприємства можна визначити як здатність підприємства створювати, виробляти і продавати товари та послуги, характеристики яких привабливіші, ніж в аналогічних продуктах конкурентів. Висока конкурентоспроможність – це випуск продукції, співвідношення ціни та якості якої буде найоптимальнішим та найпривабливішим на ринку.

Одним з методів підвищення конкурентоспроможності є зменшення ціни без втрати якісних характеристик продукції. Ціна продукції складається з таких аспектів як витрати на вироблення продукції (собівартість), зберігання продукції, реклама, транспортування до місця реалізації, та багато інших.

Транспортні перевезення є невід’ємною складовою майже будь-якого виробництва.

Значення прискорення доставки вантажів полягає не тільки в скороченні часу на перевезення певних матеріальних цінностей суспільства, а і в тому, що це є важливим резервом підвищення продуктивності праці, та, відповідно, її результативності [1].

Саме через прагнення підприємства до високої конкурентоспроможності питання зниження ціни на транспортні перевезення дуже актуальне. Оптимізація маршрутів перевезень товарів дозволяє знизити транспортні витрати, а отже знизити кінцеву ціну товарів без втрати їх якісних характеристик.

З цього випливає, що першочерговим завданням логістичної служби будь-якого підприємства є визначення найбільш раціонального маршруту.

Основні принципи оптимізації перевезень:

* принцип оптимальності: найкоротший маршрут дозволяє знизити витрати на паливо, амортизацію транспорту;
* принцип адекватного використання ресурсів: вартість на оптимізацію маршрутів транспортних перевезень не повинна перевищувати прибуток від оптимізації;
* принцип відповідальності: розраховувати на певні результати від оптимізації слід лише при відповідальному ставленні та безумовному виконанні усіх рекомендацій співробітниками компанії.

# Поняття і сутність оптимізації транспортних перевезень

Маршрут – напрямок руху транспортного засобу при здійсненні перевезень між початковим і кінцевим пунктом.

Оптимізація – процес надання будь-чому найвигідніших характеристик, співвідношень.

Транспорт – сукупність засобів, призначених для переміщення людей, вантажів, продукції з одного місця в інше.

Транспорт має фундаментальне значення для економіки та суспільства. Мобільність є життєвою необхідністю для ринку та якості життя громадян. Транспорт, у свою чергу, створює умови для зростання економіки та працевлаштування населення [2].

Процес оптимізації транспортних перевезень складається з:

* аналізу поточного стану існуючої транспортної системи підприємства;
* об’єктивної оцінки дієздатності, продуктивності транспортної системи підприємства, виявлення її недоліків і переваг;
* складання списку дій, відповідно до проведеного аналізу, направлених на вирішення проблемних питань, серед яких визначаються дії першочергового порядку, тобто ті, які необхідно впровадити найближчим часом, та дії поступового впровадження – іншими словами, проблеми відсортовуються за ступенем гостроти.
* впровадження створених рекомендацій, враховуючи поточні матеріальні і фізичні ресурси підприємства, та зіставляючи їх з списком необхідних дій.

Основні етапи створення і інтегрування оптимізації транспортних перевезень:

1‑й етап – організаційно-підготовчий: зводяться до мінімуму перевантаження з одного транспортного засобу в інший, з однієї тари для транспортування – в іншу; впроваджується облік типу машини для кращої вантажомісткості; зводяться до мінімуму витрати на перевезення товару неналежної якості;

2‑й етап – аналізується поточний стан замовлень та завантаженість транспортних засобів, приймаються рішення, щодо розширення чи оновлення транспортного парку, проводиться налагодження розгалуженої мережі пунктів зберігання товару – складів;

3‑й етап – інтегрується до робочого складу співробітників людина чи навіть відділ (в залежності від обсягів компанії), посадовими обов'язками якої буде вибір найвигіднішого маршруту, тобто маршруту який вимагатиме найменшу витрату ресурсів на перевезення;

4‑й етап – введення контролю за роботою налаштованої системи. Мета цього етапу – постійний нагляд і стимулювання шляхом впровадження інновацій в роботу транспортної гілки підприємства для підтримання системи на працездатному рівні і швидкого реагування на можливі зміни.

# Формування оптимального маршруту

Для знаходження оптимального варіанту маршруту застосовують математичні моделі, засновані на використанні в якості вхідної інформації транспортні мережі, що відображає транспортні зв'язки між пунктами відправлення та призначення вантажів. З усіх математичних об'єктів графи займають провідне місце в якості формальних моделей реальних систем. Граф являє собою сукупність скінченного числа точок, які називаються вершинами графа, і попарно з'єднують деякі з цих вершин лініями, що називають ребрами або дугами графа. Використання графових моделей у вирішенні оптимізаційних задач зберігає наочність і змістовність описуваних об'єктів і дозволяє будувати легкі для сприйняття людиною формальні алгоритми обробки цих моделей.

Таким чином, транспортні мережі зручно представити у вигляді графа.

Транспортна мережа враховує тільки ту частину дорожньої мережі, по якій можливо організувати відповідні перевезення, тобто враховуються обмеження за станом вулиць (доріг), односторонній рух, обмеження на рух вантажного транспорту, на повну масу транспортного засобу, дозволені навантаження на вісь та інші.

Транспортна мережа може бути представлена тільки зв’язаним графом.

Моделювання транспортної мережі починають з розміщення вершин. Вершини привласнюють пунктам завантаження і відвантаження вантажу, центрам великих житлових кварталів, відокремлених населених пунктів. Вершини, що мають між собою транспортне сполучення, пов'язують ребрами або (в разі однобічного зв'язку) орієнтованими дугами.

Кожному ребру зіставляють критерій вигідності, який визначається не тільки витратами часу, а тією метою, яку необхідно досягти при вирішенні задачі оптимального варіанту перевезень. Найбільш часто в якості критерію приймається мінімум сумарного пробігу, адже при однакових умовах руху на всіх ділянках маршруту план, оптимальний по пробігу, буде оптимальним за витратами часу і вартості. Крім того, в якості критерію вигідності так само можуть бути використані такі показники: платність або завантаженість доріг, частота перетину дороги з населеними пунктами і так далі.

При побудові графа слід вибирати раціональне число вершин. З одного боку, число вершин має бути якомога більше. З іншого боку, чим більше число вершин – тим транспортна мережа буде складнішою, тобто визначення найкоротших відстаней потребує тривалого часу.

Для зниження розмірності задачі і прискорення розрахунків для транспортних мереж великих міст або районів застосовують мікро- і макрорайонування. При мікрорайонуванні транспортної мережі в якості вершин використовують не перетин вулиць і конкретні пункти відправлення та призначення, а центри мікрорайонів (районів отримання або призначення вантажів). Макрорайонування транспортної мережі полягає в розбитті її на окремі підмережі, розрахунки за якими виконуються окремо, а потім об'єднуються для отримання загального результату. При змінах дорожньої обстановки проводиться перерахунок окремої підмережі, в якій відбулися зміни, а не по всій транспортній мережі.

Задача про найкоротший шлях є однією з найважливіших класичних задач теорії графів. Значимість даної задачі визначається її широким практичним застосуванням. Так, завдання пошуку найкоротшого шляху на графі широко використовується для знаходження шляхів між фізичними об'єктами на картографічних сервісах, в GPS-навігаторах, при визначенні найменшої відстані в мережі доріг.

Існує ряд широко відомих алгоритмів для її вирішення, наприклад, алгоритми перебору, алгоритм найближчого сусіду, генетичний алгоритм та інші.

Якщо вирішувати завдання пошуку найкоротшого шляху перебором всіх замкнутих шляхів, що зв'язують вершини, то доведеться покроково перевірити всі можливі маршрути. Метод є надзвичайно неефективним при великій кількості пунктів.

Алгоритм найближчого сусіда – один серед найпростіших методів розв'язку задачі пошуку найкоротшого шляху. Алгоритм роботи зводиться до наступного: вершини графу послідовно включаються в маршрут, при цьому кожна чергова вершина, що включається, повинна бути найближчою до останньої обраної вершини серед усіх, які ще не включені до маршруту. Недоліком алгоритму є те, що він припиняє роботу, коли знайдено розв'язок, та не намагається оптимізувати його.

Генетичний алгоритм – алгоритм пошуку найкоротшого шляху, який виконується шляхом випадкового підбору, поєднання варіацій шуканих параметрів через використання механізмів, аналогічних істотному відбору в природі. Він є різновидом еволюційних обчислень, за допомогою яких вирішуються оптимізаційні задачі з використанням методів природної еволюції. Недоліками алгоритму є: погана масштабованість під складність вирішуємої проблеми – тобто під велику кількість значень; тенденція зведення до локального мінімуму, чи навіть до випадкового значення замість вирішення завдання.

До ефективних алгоритмів пошуку можна віднести такі алгоритми як Дейкстри, мурашиний, імітації відпалу, генетичний алгоритм.

Найбільш відомий алгоритм рішення задачі пошуку найкоротших шляхів з однієї вершини носить ім'я алгоритм Дейкстри. Цей алгоритм застосовується лише до графів з невід'ємними вагами. Оскільки в більшості додатків ця умова виконується, таке обмеження ніяк не сприяє зниженню популярності алгоритму Дейкстри[3, с 387]. Через це далі буде розглянуто саме його.

# Алгоритм Дейкстри

## Опис алгоритму

Алгоритм Дейкстри – алгоритм на графах, відкритий Дейкстрою. Знаходить найкоротший шлях від однієї вершини графа до всіх інших вершин.

Дана концепція досі активно використовується в багатьох додатках для відтворення маршрутів на картах.

Заведемо масив (набір) елементів *d[ ]*, в якому для кожної вершини *v* будемо зберігати поточну довжину *d[v]* найкоротшого шляху з *s* (початкової вершини) в *v* (кінцеву вершину). Спочатку запишемо, що *d[s] = 0*, а для всіх інших вершин ця довжина дорівнює нескінченності (1):

(1)

Слід зазначити, що при реалізації на комп’ютері зазвичай в якості нескінченності вибирають максимально можливе для запису число, визначаючи обмеження на вхідні дані в документації до програми.

Також, для кожної вершини *v* будемо зазначати, позначена вона вже чи ні, тобто запишемо ще масив *u[ ]*. Спочатку всі вершини не позначені, тобто (2):

(2)

Сам алгоритм Дейкстри складається з *n* ітерацій. На черговій ітерації визначається вершина *v* з найменшою величиною *d[v]* серед ще не помічених, а саме (3):

(3)

Зазначимо, що на першій ітерації буде вибрана стартова вершина *s*.

Обрана таким чином вершина *v* відзначається поміченої. Далі, на поточній ітерації, з вершини *v* виробляються релаксації: проглядаються всі ребра (*v*, *t*), які виходять з вершини *v*, і для кожної такої вершини *t* алгоритм намагається поліпшити значення *d[t]*. Нехай довжина поточного ребра дорівнює *length*, тоді у вигляді коду релаксація виглядає як (4):

(4)

На цьому поточна ітерація закінчується, алгоритм переходить до наступної ітерації (знову вибирається вершина з найменшою величиною *d*, з неї виробляються релаксації, і т.п.). При цьому, в решті-решт, після *n* ітерацій, всі вершини графа стануть поміченими, і алгоритм свою роботу завершує. Знайдені значення *d[v]* і є шукані довжини найкоротших шляхів з *s* в *v*.

Варто зауважити, що, якщо не всі вершини графа досяжні з вершини *s*, то значення *d[v]* для них так і залишаться нескінченністю. Зрозуміло, що на кількох останніх ітераціях алгоритму будуть якраз вибиратись ці вершини, але ніякої корисної роботи виробляти ці ітерації не будуть (оскільки нескінченна відстань не зможе прорелаксувати інші, навіть теж нескінченні відстані). Тому як тільки в якості обраної вершини береться вершина з нескінченною відстанню алгоритм можна відразу зупиняти. Це збереже час і ресурси на вирішення задачі.

Зрозуміло, що потрібно знати не тільки довжини найкоротших шляхів, а й отримати самі шляхи до цих вершин. Для цього достатньо так званого «масиву предків» – масиву *p[ ]*, в якому для кожної вершини *v ≠ s* зберігається номер вершини *p[v]*, що є передостанньою в найкоротшому шляху до вершини *v*. Найкоротший шлях можна буде відновити просто щоразу беручи предка від поточної вершини, поки ми не прийдемо в стартову вершину *s* – так ми отримаємо шуканий найкоротший шлях, але записаний в зворотному порядку. Отже, найкоротший шлях *P* до вершини *v* дорівнює (5):

(5)

Будувати масив предків досить просто: при кожній успішній релаксації, тобто коли з обраної вершини *v* відбувається поліпшення відстані до деякої вершини *t*, ми записуємо, що предком вершини *t* є вершина *v* (6):

(6)

## Доведення істинності алгоритму

Основне твердження, на якому заснована коректність алгоритму Дейкстри, наступне. Стверджується, що після того як будь-яка вершина *v* стає поміченої, поточна відстань до неї *d[v]* вже є найкоротшою, і, відповідно, більше не буде змінюватись.

Доказ будемо виробляти за індукцією. Для першої ітерації справедливість твердження очевидна – для вершини *s* маємо *d[s] = 0*, що і є довжиною найкоротшого шляху до неї. Нехай тепер це твердження виконано для всіх попередніх ітерацій, тобто всіх уже помічених вершин. Доведемо, що воно не порушується після виконання поточної ітерації. Нехай *v* – вершина, обрана на поточній ітерації, тобто вершина, яку алгоритм збирається помітити. Доведемо, що *d[v]* дійсно дорівнює довжині найкоротшого шляху до неї (позначимо цю довжину через *l[v]*).

Розглянемо найкоротший шлях *P* до вершини *v*. Зрозуміло, цей шлях можна розбити на два шляхи: *P1*, що складається тільки з позначених вершин (як мінімум стартова вершина *s* буде в цьому шляху), і інша частина шляху *P2* (вона теж може включати помічені вершини, але починається обов'язково з непомічених). Позначимо через *p* першу вершину шляху *P2*, а через *q* – останню вершину шляху *P1*.

Доведемо спочатку твердження для вершини *р*, тобто доведемо рівність *d[p] = l[p]*. Однак, це практично очевидно, бо на одній з попередніх ітерацій була вибрана вершина *q* і виконана релаксація з неї. Оскільки (за вибором самої вершини *p*) найкоротший шлях до *p* дорівнює найкоротшому шляху до *q* з додаванням ребра *(p, q)*, то при виконанні релаксації з *q* величина *d[p]* дійсно встановлюється в відповідне значення.

Внаслідок невід'ємності вартостей ребер довжина шляху найкоротшого шляху *l[p]* (а вона, за вищезазначеним доказом дорівнює *d[p]*) не перевищує довжину *l[v]* найкоротшого шляху до вершини *v*. З огляду на те, що *l[v] ≤ d[v]* (бо алгоритм Дейкстри не міг знайти більш коротшого шляху, чим це можливо), в результаті отримуємо співвідношення (7):

(7)

Оскільки *p* і *v* – вершини невідмічені, то так як на поточній ітерації була обрана саме вершина *v*, а не вершина *p*, то отримаємо іншу нерівність (8):

(8)

З цих двох нерівностей випливає рівність *d[p] = d[v]*, а тоді, з відшуканих до цього співвідношень, отримаємо (9):

(9)

Що й потрібно було довести.

## Приклад розв’язування задачі на графі

Для кращого засвоєння далі наведено приклад розв’язку задачі на складеному випадковим чином графі (рис 3.1).

Завдання: необхідно визначити найкоротшу відстань від пункту 1 (1‑ша вершина) до пункту 5 (5‑та вершина).

Зазначимо умовні позначення: еліпси – це населені пункти (тобто вершини), лінії – це шляхи між ними. В еліпсах зазначено номери населених пунктів (вершин), над ребрами позначено кількість кілометрів – їх вага (довжина шляху, у км).

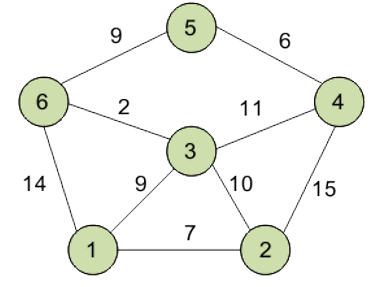


Рисунок 3.1 – Вихідний граф

Запуск (ініціалізація). Поблизу кожної вершини позначається червона мітка – довжина найкоротшого шляху вершини 1 в цю вершину. Приймається, що мітка вершини 1 дорівнює 0, а мітки інших вершин – нескінченності, яку, умовно, для кращого сприйняття конкретної задачі, позначимо 10000. Невідвіданими позначаються всі вершини графа (рис 3.2).

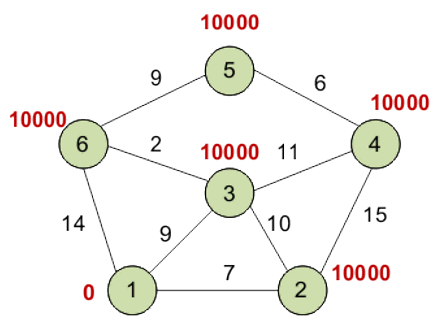


Рисунок 3.2 – Візуалізація ініціалізації графу розв’язання

Крок 1. Мінімум серед міток має 1‑ша вершина. Прилеглими до 1‑ї вершини є вершини за номерами 2, 3 і 6. Переглядаємо прилеглі вершини по черзі.

2‑га вершина – перша прилегла 1‑ї вершини, бо до неї мінімальна довжина шляху. Довжина шляху в неї через 1‑шу вершину є сумою мінімальної відстані до 1‑ї вершини – тобто значенням її мітки, і довжини ребра, що йде з 1‑ї в 2‑гу вершини, тобто 0 + 7 = 7. Результат значно менше поточної мітки 2‑ї вершини (10000) – а, відповідно, нова мітка 2‑ї вершини буде дорівнювати 7 (рис 3.3).

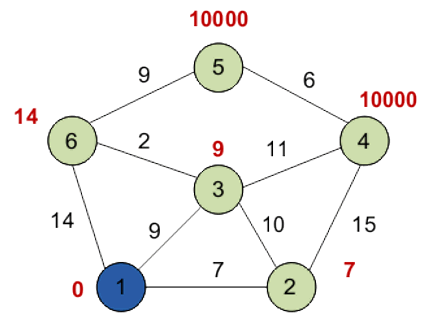


Рисунок 3.3 – Візуалізація 1‑го кроку розв’язання

Аналогічно до другої вершини знаходимо довжини шляхів для всіх інших прилеглих вершин (вершини 3 і 6). Після перегляду всіх прилеглих вершин 1‑ї вершини поточна мінімальна відстань до 1‑ї вершини приймається остаточною і не підлягає перегляду. Вершина 1 зазначається як відвідана.

Крок 2. Повторюється 1 крок алгоритму. Знаходимо «найближчу» з невідвіданих прилеглих вершин. Це 2‑га вершина з значенням 7. Спробуємо зменшити мітки прилеглих вершин обраної вершини, завдяки проходу до них через 2‑гу вершину. Прилеглими до 2‑ї вершини є вершини 1, 3 і 4.

1‑ша вершина вже переглянута. Наступна прилегла вершина 2‑ї вершини – вершина 3, бо має мінімум серед позначок з вершин, що є невідвіданими. Якщо йти в неї через 2, то довжина такого шляху дорівнюватиме 17 (7 + 10 = 17). Враховуючи, що поточна мітка 3‑ї вершини дорівнює 9, а 9 < 17, відповідно мітка не змінюється (рис 3.4).

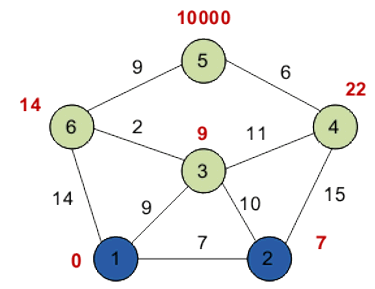


Рисунок 3.4 – Візуалізація 2‑го кроку розв’язання

Ще одна з прилеглих вершин 2‑ї вершини – вершина 4. Якщо йти в неї через 2‑гу, то вага такого шляху буде дорівнює 22 (7 + 15 = 22). Оскільки 22 < 10000, встановлюємо мітку 4‑ї вершини рівній 22.

Всі вершини прилеглі до 2‑ї вершини переглянуті, помічаємо її як відвідану.

Крок 3. Повторюємо крок алгоритму, для вершини 3. Після виконання алгоритму отримаємо наступні результати (рис 3.5).

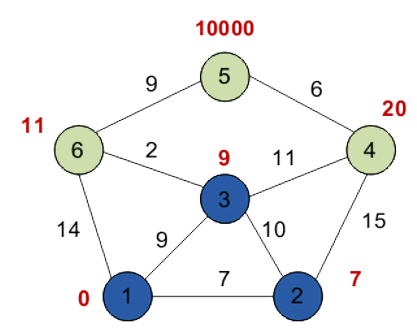


Рисунок 3.5 – Візуалізація 3‑го кроку розв’язання

Далі все робиться аналогічно доти, доки граф не буде повністю перевірений(рис 3.6 – рис 3.8).

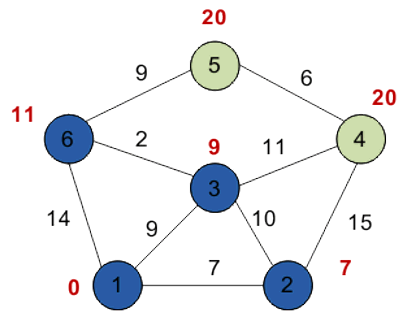


Рисунок 3.6 – Візуалізація 4‑го кроку розв’язання

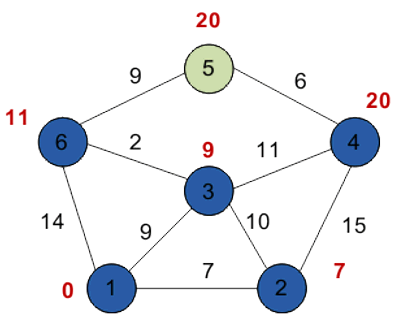


Рисунок 3.7 – Візуалізація 5‑го кроку розв’язання

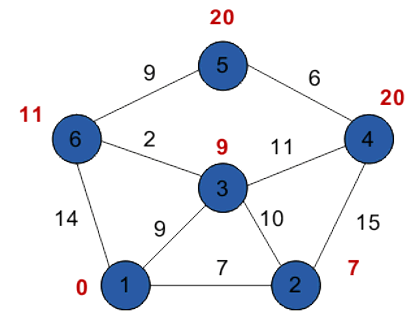


Рисунок 3.8 – Візуалізація 6‑го кроку розв’язання

Оскільки алгоритм повністю завершив роботу – можемо визначити, що найкоротшим шляхом з вершини 1 у вершину 5 буде шлях через вершини 1 ‑ 3 ‑ 6 ‑ 5, бо саме цей шлях дасть мінімальну вагу дуг, що дорівнює 20.

Окрім вже розглянутого способу отримання найкоротшого шляху після відпрацювання алгоритму є більш простий, проте менш ефективний за швидкодією спосіб – спосіб безпосереднього перебору значень. Розглянемо вершини з кінця. Звернемо увагу на кінцеву вершину (вершину 5). Для всіх вершин, з якою вона пов'язана, знаходимо довжину шляху, як різницю довжини шляху кінцевої вершини і ваги відповідного ребра.

Вершина 5 має довжину шляху 20. Вона пов'язана з вершинами 6 і 4.

Для вершини 6 отримаємо вагу 20 – 9 = 11 (збіглося).

Для вершини 4 отримаємо вагу 20 – 6 = 14 (не збіглося).

У випадку, коли в результаті ми отримаємо значення, яке збігається з довжиною шляху розглянутої вершини (в даному випадку – вершини 6), можемо стверджувати, що саме з неї був здійснений перехід в кінцеву вершину. Відзначаємо цю вершину на шуканому шляху.

Далі визначаємо ребро, через яке ми потрапили в вершину 6 і так поки не дійдемо до початку.

При збіганні значень декількох вершин можна брати будь-яку з них – кілька шляхів матимуть однакову довжину.

## Поради при реалізації алгоритму Дейкстри

Для зберігання ваг графа можна користуватися квадратною матрицею. У заголовках рядків і стовпців знаходяться вершини. А ваги дуг розміщуються у внутрішніх осередках таблиці. Граф не містить петель, тому на головній діагоналі матриці містяться нульові значення.

# Висновки

Сучасний стан ринкової економіки і виробництва вимагає прийняття рішень, направлених на оптимізацію виробництва без втрати якості кінцевого продукту. Одним з напрямів впровадження оптимізаційних рішень на підприємства є завдання з мінімізації транспортних витрат на перевезення вантажу.

Запропоновані методи оптимізації допоможуть виробити максимально ефективну стратегію транспортних перевезень на підприємстві, що сприятиме зниженню кінцевої ціни продукції та підвищенню конкурентоздатності підприємства, а, відповідно, й його розвитку.

# Перелік посилань

1. Литвиненко С.Л. Науково-методичні засади виробничо-логістичної діяльності підприємств транспорту / Литвиненко С.Л., Яновський П.О., Нестеренко Г.І., Габріелова Т.Ю. – К.: Кондор, 2017. – 262с.
2. Біла книга Європейської Комісії – план розвитку єдиного європейського транспортного простору – на шляху до конкурентоспроможності та ресурсоефективної транспортної системи, – Люксембург, 2011. – 28 с.
3. Левітін А.В. Алгоритми: введення в розробку і аналіз, пер. з англ. Трігуб С.Г., 2006. – 576 с.