ISSN 2074-5893

ПМ
ММ

До 100-річчя
Дніпровського
національного університету
імені Олеся Гончара
(1918-2018)

ПИТАННЯ ПРИКЛАДНОЇ МАТЕМАТИКИ І МАТЕМАТИЧНОГО МОДЕЛЮВАННЯ

2017

ISSN 2074-5893

Міністерство освіти і науки України
Дніпровський національний університет
ім. Олеся Гончара

ПИТАННЯ ПРИКЛАДНОЇ МАТЕМАТИКИ І МАТЕМАТИЧНОГО МОДЕЛЮВАННЯ

Збірник наукових праць

Випуск 17

До 100-річчя
Дніпровського
національного університету
імені Олеся Гончара
(1918-2018)

Дніпро
2017

УДК 004, 517, 519, 539, 51-76
ББК 22.12я431+22.311я431+22.176я431+22.18я431
П 32

Надруковано за рішенням вченої ради
Дніпровського національного університету ім. Олеся Гончара

П 32 **Питання прикладної математики і математичного моделювання [Текст]**: зб. наук. пр. / редкол.: О. М. Кісельова (відп. ред.) [та ін.]. – Д., 2017. – Вип. 17. – 279 с.

Рецензенти:
д-р фіз.-мат. наук, проф. **В. В. Лобода**
д-р фіз.-мат. наук, проф. **С. Б. Вакарчук**

У збірнику вміщено результати фундаментальних досліджень та практичних розробок із проблем математичного і програмного забезпечення інтелектуальних систем. Статті присвячено питанням математичного моделювання складних прикладних систем і розробці ефективних обчислювальних методів та алгоритмів їх розв’язання, а також оптимізації, чисельних методів, функціонального аналізу, математичної фізики.

Призначений для науковців, викладачів ВНЗ, може бути корисним аспірантам і студентам.

Сборник содержит результаты фундаментальных исследований и практических разработок по проблемам математического и программного обеспечения интеллектуальных систем. Статьи посвящены вопросам математического моделирования сложных прикладных систем и разработке эффективных вычислительных методов и алгоритмов их решения, а также оптимизации, численных методов, функционального анализа, математической физики.

Предназначен для ученых, преподавателей вузов, может быть полезен аспирантам и студентам.

УДК 004, 517, 519, 539, 51-76
ББК 2.12я431+22.311я431+22.176я431+22.18я431

ISSN 2074-5893

УДК 519.85

П.І. Стецюк*, О.В. Міца, О.В. Стрелюк***, О.В. Фесюк***

**Інститут кібернетики імені В.М. Глушкова НАН України, Київ*

***Ужгородський національний університет МОН України*

****Вінницький національний технічний університет МОН України*

ТРАНСПОРТНА ЗАДАЧА З ОБМЕЖЕННЯМИ НА ПРОПУСКНІ СПРОМОЖНОСТІ ПРОМІЖНИХ ПУНКТІВ

Сформульовано транспортну ЛП-задачу (задачу лінійного програмування) для знаходження оптимального плану перевезень з використанням проміжних пунктів з обмеженнями на їх пропускну спроможність. Наведено її AMPL-код та два демонстраційні приклади. Побудовано задачу негладкої оптимізації для знаходження оптимальних двоїстих змінних, які відповідають частині обмежень в ЛП-задачі.

Сформулирована транспортная ЛП-задача (задача линейного программирования) для нахождения оптимального плана перевозок с использованием промежуточных пунктов с ограничениями на их пропускные способности. Приведен ее AMPL-код и два демонстрационных примера. Построена задача негладкой оптимизации для нахождения оптимальных двойственных переменных, отвечающих части ограничений в ЛП-задаче.

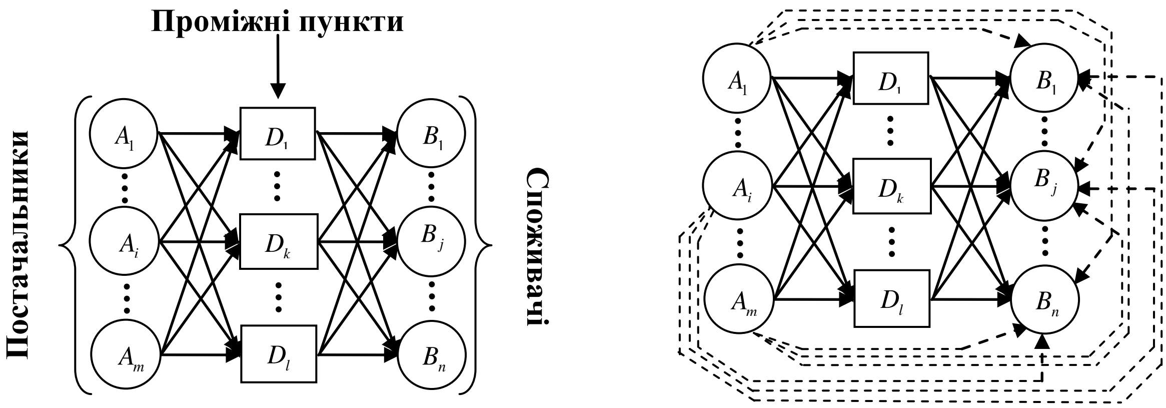
The transport LP-problem (the linear programming problem) to find the optimal transportation plan using intermediate points with restrictions on their capacity is formulated. It is AMPL-code and two demonstration examples are given. The problem of nonsmooth optimization for finding optimal dual variables corresponding to some of the constraints in the LP-problem is constructed.

Ключові слова: двохетапна транспортна задача, ЛП-задача, AMPL, gurobi, двоїста задача, методи негладкої оптимізації.

Вступ. У класичному формулюванні двохетапної транспортної задачі [1] допускається, що вантаж перевозиться від постачальників до споживачів тільки через проміжні пункти (див. рис. 1а). В якості проміжних пунктів можуть виступати посередницькі фірми та різного роду сховища (склади). При певних умовах алгоритм розв'язання такої транспортної задачі поділяється на два етапи: спочатку знаходять оптимальний план перевезень від постачальників до проміжних пунктів (посередників), а потім — від посередників до споживачів. У загальному випадку для розв'язання двохетапної транспортної задачі застосовують спеціалізовані алгоритми [1,2].

У статті розглядається узагальнення двохетапної транспортної задачі на випадок, якщо вантаж перевозиться від постачальників до споживачів не тільки через проміжні пункти, але і напряму. Схема функціонування перевезень вантажу для нашої задачі наведена на рис. 1б. Цю задачу називатимемо транспортною ЛП - задачею (задачею лінійного програмування). В ній буде враховуватися, що пропускна спроможність проміжних пунктів обмежена.

Задачі такого типу актуальні для підприємств, які займаються доставкою вантажу від постачальника до споживача і мають мережу пунктів зберігання вантажу. Наприклад, це можуть бути національні служби доставки товарів,



а) б)

Рис.1. Система «постачальники – проміжні пункти – споживачі»: а) для двохетапної транспортної задачі; б) для транспортної LP-задачі

які мають широку мережу пунктів прийому/видачі товарів та кілька проміжних складів на території країни. Їм потрібно мінімізувати витрати при доставці товару з однієї частини країни в іншу перевантаживши товар у відповідному пункті. Також, це можуть бути агропідприємства, які орендують землю для вирощування продукції в різних частинах країни, зберігають продукцію в власних чи орендованих сховищах та мають кілька пунктів переробки продукції. Агропідприємствам потрібно зібраний урожай розподілити по сховищах, враховуючи що сховище повинно бути або достатньо завантажене, або не використовуватися взагалі. Пункти переробки також мають невеликі склади, які забезпечують їх роботу, тому одна частина продукції може бути доставлена напряму, а друга частина продукції повинна підвозитись з проміжних пунктів тимчасового зберігання.

Обчислювальні складнощі, що виникають при розв’язанні транспортних ЛП-задач, пов'язані з їх великою розмірністю. У той же час ці задачі мають ряд властивостей, близьких до транспортної задачі, що дозволяє розробляти спеціальні методи їх розв’язання, які зазвичай більш ефективні, ніж загальні методи лінійного програмування. Зокрема, ці властивості використовуються при побудові схем декомпозиції, які в поєднанні з методами негладкої оптимізації дозволяють отримати максимальний обчислювальний ефект [3]. Для алгоритмів на основі схеми декомпозиції легко замінити лінійні цільові функції на нелінійні, наприклад, дробово-лінійні, яким присвячена книга [4].

Матеріал статті викладено в наступному порядку. У розділі 1 описано формулювання транспортної ЛП-задачі, наведено її властивості та зв’язок з двохетапною транспортною задачею. У розділі 2 наведено опис задачі на мові моделювання AMPL [5] та результати розрахунків програмою **gurobi**. У розділі 3 сформульована двоїста задача до транспортної ЛП-задачі, яка може бути розв'язана за допомогою алгоритмів оптимізації негладких функцій.

1. Транспортна ЛП-задача та її властивості. Нехай в m пунктах постачання $A_1, A_2, ..., A_m$ є $a_1, a_2, ..., a_m$ одиниць продукції, яку потрібно перевезти до n споживачів $B_1, B_2, ..., B_n$, задовольнивши їх потреби $b_1, b_2, ..., b_n$. Для транспортування продукції від постачальників до споживачів можна задіяти l проміжних пунктів $D_1, D_2, ..., D_l$ з мінімальними $d_1^{low}, d_2^{low}, ..., d_l^{low}$ та максимальними $d_1^{up}, d_2^{up}, ..., d_l^{up}$ пропускними спроможностями ($d_k^{low} \le d_k^{up}, k = 1, ..., l.$). Витрати на перевезення одиниці продукції з пункту постачання A_i до проміжного пункту D_k позначимо c_{ik}, з проміжного пункту D_k до споживача B_j – c_{kj}, з пункту постачання A_i до споживача B_j – c_{ij}. Кількість продукції, яка перевозиться від постачальника A_i до проміжного пункту D_k, позначимо x_{ik}, від проміжного пункту до споживача – y_{kj}, від постачальника до споживача – z_{ij}.

Транспортна ЛП задача має такий вигляд: знайти

$$f^* = \min_{x,y,z} \left\{ f(x,y,z) = \sum_{i=1}^{m}\sum_{k=1}^{l} c_{ik} x_{ik} + \sum_{k=1}^{l}\sum_{j=1}^{n} c_{kj} y_{kj} + \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij} z_{ij} \right\} \quad (1)$$

за обмежень

$$\sum_{k=1}^{l} x_{ik} + \sum_{j=1}^{n} z_{ij} = a_i, \quad i = 1, \ldots, m, \quad (2)$$

$$\sum_{k=1}^{l} y_{kj} + \sum_{i=1}^{m} z_{ij} = b_j, \quad j = 1, \ldots, n, \quad (3)$$

$$\sum_{i=1}^{m} x_{ik} - \sum_{j=1}^{n} y_{kj} = 0, \quad k = 1, \ldots, l, \quad (4)$$

$$d_k^{low} \le \sum_{j=1}^{n} y_{kj} \le d_k^{up}, \quad k = 1, \ldots, l, \quad (5)$$

$$x_{ik} \ge 0, \; y_{kj} \ge 0, \; z_{ij} \ge 0, \; i = 1, \ldots, m, \, k = 1, \ldots, l, \; j = 1, \ldots, n. \quad (6)$$

Вона є задачею лінійного програмування, містить $m \times l + l \times n + m \times n$ змінних x_{ik}, y_{kj}, z_{ij} та $m + 3l + n$ обмежень загального виду, тобто не враховуючи обмежень (6) – умов на невід'ємність усіх змінних. Тут цільова функція (1) задає сумарні витрати на транспортування продукції від постачальників до споживачів, враховуючи транспортування через проміжні пункти. Обмеження (2) означає транспортування усієї продукції $a_1, a_2, ..., a_m$ із пунктів постачання до проміжних пунктів та споживачів. Обмеження (3) означає, що споживачам потрібно доставити необхідну кількість продукції $b_1, b_2, ..., b_n$, яка може бути доставлена як з пунктів постачання так і з проміжних пунктів. Обмеження (4) задає умови на те, щоб вся продукція яка приходить від постачальників до кожного проміжного пункту, була обовязково відправлена споживачам. Обмеження (5) визначає умови на пропускні здатності проміжних пунктів. Вони записані для вихідних сумарних потоків із проміжних пунктів, але зміст задачі не зміниться, якщо їх записати для вхідних потоків.

Задача (1)–(6) відноситься до закритих задач транспортного типу, тобто вся продукція постачальників повинна бути доставлена споживачам, не залишаючи нічого в проміжних пунктах. Це диктує умови на сумісність системи обмежень (2)–(6) – системи лінійних рівностей та лінійних нерівностей. Справедливим є наступне твердження.

Лема 1. Система обмежень (2)–(6) є несумісною, якщо виконується одна із наступних умов: 1) $\sum_{i=1}^{m} a_i \neq \sum_{j=1}^{n} b_j$, 2) $\sum_{k=1}^{l} d_k^{low} > \sum_{j=1}^{n} b_j$, 3) $\sum_{k=1}^{l} d_k^{low} > \sum_{i=1}^{m} a_i$.

Доведення. Доведення проведемо методом від супротивного. Приймемо, що існують такі $\bar{x} = \left\{ \bar{x}_{ik}, i = 1,\ldots,m, k = 1,\ldots,l \right\}$, $\bar{y} = \left\{ \bar{y}_{kj}, k = 1,\ldots,l, j = 1,\ldots,n \right\}$ та $\bar{z} = \left\{ \bar{z}_{ij}, i = 1,\ldots,m, j = 1,\ldots,n \right\}$, які задовільняють систему обмежень (2)–(6).

Якщо всі рівності із (2) просумувати за індексом $i = 1,\ldots,m$, а всі рівності із (3) просумувати за індексом $j = 1,\ldots,n$, то отримаємо такі рівності

$$\sum_{i=1}^{m}\sum_{k=1}^{l} \bar{x}_{ik} + \sum_{i=1}^{m}\sum_{j=1}^{n} \bar{z}_{ij} = \sum_{i=1}^{m} a_i, \tag{7}$$

$$\sum_{j=1}^{n}\sum_{k=1}^{l} \bar{y}_{jk} + \sum_{j=1}^{n}\sum_{i=1}^{m} \bar{z}_{ij} = \sum_{j=1}^{n} b_j. \tag{8}$$

Віднявши від рівності (7) рівність (8) та змінивши порядок індексів в сумах, отримаємо наступну рівність

$$\sum_{k=1}^{l}\sum_{i=1}^{m} \bar{x}_{ik} - \sum_{k=1}^{l}\sum_{j=1}^{n} \bar{y}_{jk} = \sum_{i=1}^{m} a_i - \sum_{j=1}^{n} b_j. \tag{9}$$

Якщо всі рівності із (4) просумувати за індексом $k = 1,\ldots,l$, то отримаємо таку рівність

$$\sum_{k=1}^{l}\sum_{i=1}^{m} \bar{x}_{ik} - \sum_{k=1}^{l}\sum_{j=1}^{n} \bar{y}_{kj} = 0. \tag{10}$$

За умови $\sum_{i=1}^{m} a_i \neq \sum_{j=1}^{n} b_j$ із рівності (9) випливає $\sum_{k=1}^{l}\sum_{i=1}^{m} \bar{x}_{ik} - \sum_{k=1}^{l}\sum_{j=1}^{n} \bar{y}_{kj} \neq 0$, що протирічить рівності (10). Це протиріччя доводить, що вектори $\bar{x}$, $\bar{y}$ та $\bar{z}$ не можуть бути допустимими для обмежень (2)–(5), а значить при виконанні умови 1) система обмежень (1)–(6) є несумісною.

Якщо усі нерівності із (4), які відповідають нижнім границями на пропускні спроможності вузлів, просумувати за індексом $k = 1,\ldots,l$, то отримаємо наступну нерівність

$$\sum_{k=1}^{l} d_k^{low} \leq \sum_{k=1}^{l}\sum_{j=1}^{n} \bar{y}_{kj}. \tag{11}$$

Із рівності (8), враховуючи що компоненти $\overline{z} = \left\{ \overline{z}_{ij}, i = 1,\ldots,m, j = 1,\ldots,n \right\}$ є невід'ємними, випливає справедливість такої нерівності

$$\sum_{j=1}^{n}\sum_{k=1}^{l}\overline{y}_{jk} \le \sum_{j=1}^{n} b_j. \tag{12}$$

Нерівності (11) та (12) можна об'єднати в одну двосторонню нерівність

$$\sum_{k=1}^{l} d_k^{low} \le \sum_{k=1}^{l}\sum_{j=1}^{n}\overline{y}_{kj} \le \sum_{j=1}^{n} b_j. \tag{13}$$

З нерівності (13) отримуємо $\sum_{k=1}^{l} d_k^{low} \le \sum_{j=1}^{n} b_j$, що протирічить $\sum_{k=1}^{l} d_k^{low} > \sum_{j=1}^{n} b_j$. Це означає, що вектори $\overline{x}$, $\overline{y}$ та $\overline{z}$ не можуть бути допустимими для обмежень (2)–(6). Тому при виконанні умови 2) система обмежень (2)–(6) є несумісною.

Враховуючи, що із рівності (10) маємо $\sum_{k=1}^{l}\sum_{i=1}^{m}\overline{x}_{ik} = \sum_{k=1}^{l}\sum_{j=1}^{n}\overline{y}_{kj}$, то нерівність (11) можна переписати в такому вигляді

$$\sum_{k=1}^{l} d_k^{low} \le \sum_{k=1}^{l}\sum_{i=1}^{m}\overline{x}_{ik}. \tag{14}$$

Із рівності (7), враховуючи що компоненти $\overline{z} = \left\{ \overline{z}_{ij}, i = 1,\ldots,m, j = 1,\ldots,n \right\}$ є невід'ємними, випливає справедливість такої нерівності

$$\sum_{i=1}^{m}\sum_{k=1}^{l}\overline{x}_{ik} \le \sum_{i=1}^{m} a_i. \tag{15}$$

Нерівності (14) та (15) також можна записати як двосторонню нерівність

$$\sum_{k=1}^{l} d_k^{low} \le \sum_{k=1}^{l}\sum_{i=1}^{m}\overline{x}_{ik} \le \sum_{i=1}^{m} a_i. \tag{16}$$

З нерівності (16) випливає $\sum_{k=1}^{l} d_k^{low} \le \sum_{i=1}^{m} a_i$, що протирічить $\sum_{k=1}^{l} d_k^{low} > \sum_{i=1}^{m} a_i$. Тому при виконанні умови 3) вектори $\overline{x}$, $\overline{y}$ та $\overline{z}$ також не можуть бути допустимими для системи (2)–(6). Це означає, що система обмежень (2)–(6) є несумісною за умови 3). Лема доведена.

Цікавим є окремий випадок задачі (1)–(6), коли вона розпадається на дві незалежні класичні транспортні підзадачі. Йому відповідає наступне твердження.

Лема 2. Якщо виконується умова $\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j = \sum_{k=1}^{l} d_k^{low}$, то задача (1)–(6) розпадається на дві незалежні транспортні підзадачі:

$$f_1^* = \min_{x \ge 0}\left\{ f(x) = \sum_{i=1}^{m}\sum_{k=1}^{l} c_{ik}x_{ik} \right\}: \ \sum_{k=1}^{l} x_{ik} = a_i, i = 1,\ldots,m, \ \sum_{i=1}^{m} x_{ik} = d_k^{low}, k = 1,\ldots,l, \tag{17}$$

$$f_2^* = \min_{x \ge 0}\left\{ f(y) = \sum_{k=1}^{l}\sum_{j=1}^{n} c_{kj}y_{kj} \right\}: \ \sum_{k=1}^{l} y_{kj} = b_j, j = 1,\ldots,n, \ \sum_{j=1}^{n} y_{kj} = d_k^{low}, k = 1,\ldots,l. \tag{18}$$

При цьому оптимальні значення цільових функцій в задачі (1)–(6) та задачах (17), (18) задовільняють умові $f^* = f_1^* + f_2^*$.

Доведення леми 2 наводити не будемо, а відмітимо тільки те, що воно враховує, що при виконанні умови $\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j = \sum_{k=1}^{l} d_k^{low}$ змінні z_{ij} стають нульовими. Якщо у задачі (1)–(6) забрати змінні z_{ij} то вона переходить в двоетапну транспортну задачу з обмеженнями на пропускні спроможності проміжних пунктів. Ця задача має наступний вигляд: знайти:

$$f_A^* = \min_{x,y,z}\left\{ f(x,y,z) = \sum_{i=1}^{m}\sum_{k=1}^{l} c_{ik}x_{ik} + \sum_{k=1}^{l}\sum_{j=1}^{n} c_{kj}y_{kj} \right\} \tag{19}$$

за обмежень

$$\sum_{k=1}^{l} x_{ik} = a_i, \quad i = 1,\ldots,m, \tag{20}$$

$$\sum_{k=1}^{l} y_{kj} = b_j, \quad j = 1,\ldots,n, \tag{21}$$

$$\sum_{i=1}^{m} x_{ik} - \sum_{j=1}^{n} y_{kj} = 0, \quad k = 1,\ldots,l, \tag{22}$$

$$d_k^{low} \le \sum_{j=1}^{n} y_{kj} \le d_k^{up}, \quad k = 1,\ldots,l, \tag{23}$$

$$x_{ik} \ge 0, \ y_{kj} \ge 0, \ i = 1,\ldots,m, k = 1,\ldots,l, \ j = 1,\ldots,n, \tag{24}$$

Задача (19)–(24) містить $m \times l + l \times n$ змінних та $m + 3l + n$ обмежень загального виду. Якщо з неї забрати обмеження (23) – умови на пропускні спроможності проміжних пунктів, то задача (19)–(24) переходить в відому двоетапну транспортну задачу [1, с. 69].

2. AMPL-код для транспортної ЛП-задачі. Нижче наведемо опис задачі (1)–(6) на мові алгебраїчного моделювання AMPL [5] та опишемо результати розрахунків за його допомогою для двох прикладів – приклад 5.4 з [2, с. 232] та його незначна модифікація, яка пов'язана з використанням нижніх меж на пропускні спроможності проміжних пунктів.

Приклад 1. Виробниче об'єднання складається з трьох філіалів: A_1, A_2, A_3, які виготовляють однорідну продукцію в обсягах відповідно 1000, 1500 та 1200 одиниць на місяць. Ця продукція відправляється на два склади D_1 і D_2 місткістю відповідно 2500 та 1200 од., а потім – до п'яти споживачів B_1, B_2,

…, B_5, попит яких становить відповідно 900, 700, 1000, 500 і 600 од. Вартості перевезень одиниці продукції (в умовних одиницях) від виробників на склади, а потім – зі складів до споживачів наведені в табл. 1а та 1б.

Крім того, за індивідуальними контрактами можливі безпосередні поставки продукції з першого філіалу до другого споживача, а також з третього філіалу – до четвертого споживача. Вартість транспортування одиниці продукції за транзитним маршрутом A_1B_2 дорівнює 3 ум. од., а за маршрутом A_3B_4 – 4 ум. од. Перевезення продукції зі складу на склад недопустимі.

Таблиця 1

Вартість перевезення 1000 т бензину, ум. од.

Виробник	Склад	
	D_1	D_2
A_1	2	8
A_2	3	5
A_3	1	4

а)

Склад	Споживач				
	B	B	B	B	B
D_1	1	3	8	5	4
D_1	2	4	5	3	1

б)

AMPL-код для опису ЛП-задачі (1)–(6) має наступний вигляд:

```
param m>=2; #Кількість постачальників
param l>=1; #Кількість проміжних пунктів
param n>=2; #Кількість споживачів
#Вартості перевезення одиниці продукції:
param cik{i in 1..m, k in 1..l} >= 0; #від A до D
param ckj{k in 1..l, j in 1..n} >= 0; #від D до B
param cij{i in 1..m, j in 1..n} >= 0; #від A до B
#Інші дані
param a{i in 1..m} >= 0; #Продукція в A
param b{j in 1..n} >= 0; #Потреби B
param d_low{k in 1..l} >= 0; #Нижня пропускна спроможність D
param d_up{k in 1..l} >= 0; #Верхня пропускна спроможність D
#Невідомі (продукція, яку потрібно перевезти)
var x{i in 1..m, k in 1..l}  >=0; #від A до D
var y{k in 1..l, j in 1..n}  >=0; #від D до B
var z{i in 1..m, j in 1..n}  >=0; #від A до B
minimize f_opt: #Мінімізувати витрати на перевезення продукції:
sum{i in 1..m, k in 1..l} cik[i,k]*x[i,k]+ #від A до D
sum{k in 1..l, j in 1..n} ckj[k,j]*y[k,j]+ #від D до B
sum{i in 1..m, j in 1..n} cij[i,j]*z[i,j]; #від A до B
subject to #за обмежень
con2 {i in 1..m}: #перевезення продукції з A до D і B
sum{k in 1..l}x[i,k]+sum{j in 1..n}z[i,j]=a[i];
con3 {j in 1..n}: #задоволення потреб B з A і D
sum{k in 1..l}y[k,j]+sum{i in 1..m}z[i,j]=b[j];
```

```
con4 {k in 1..l}: #всю продукцію потрібно доставити в B
sum{i in 1..m}x[i,k]-sum{j in 1..n}y[k,j]=0;
con5a {k in 1..l}: #нижня пропускна спроможність D
sum{j in 1..n}y[k,j]>=d_low[k];
con5b {k in 1..l}: #верхня пропускна спроможність D
sum{j in 1..n}y[k,j]<=d_up[k];
```

Якщо цей AMPL-код доповнити необхідними даними про задачу (розглянемо нижче), то його можна використовувати для розв'язання транспортних ЛП-задач за допомогою стандартного програмного забезпечення для розв'язання задач лінійного програмування. Це можна зробити як за допомогою тих програм NEOS-сервера [6] із розділу «Linear Programming», для яких підтримуються вхідні формати даних на AMPL, так і за допомогою комерційних або з вільним доступом версій AMPL.

Для проведення розрахунків ми вибрали демо-версію AMPL 3.1.0 для операційної системи на базі Linux з 64-розрядним процесором*. Вона працює з обмеженою кількістю змінних: для задач лінійного програмування це 500 змінних та 500 обмежень. Для розв'язання задач лінійного програмування в цю версію включена відома програма gurobi, для використання якої до AMPL-коду з описом ЛП-задачі (1)–(6) потрібно додати оператор:

```
option solver gurobi; #Вибираємо розв'язувач
```

Розрахунки проводилися на персональному комп'ютері з операційною системою Ubuntu 14.04 LTS, процесором AMD Athlon(tm) X2 Dual-Core QL-64 × 2 / 2.1 ГГц та 3 Гб оперативної пам'яті. Для опису вхідних даних задачі (1)–(6) для прикладу 1 використано наступний AMPL-код:

```
data; #Блок вхідних даних
#Задаємо:
param m := 3; #Кількість постачальників A
param l := 2; #Кількість проміжних пунктів D
param n := 5; #Кількість споживачів B
#Витрати на перевезення одиниці продукції:
param cik: 1 2 := #від A (↓) до D (→)
        1 2 8
        2 3 5
        3 1 4;
param ckj: 1 2 3 4 5 := #D (↓) до B (→)
        1 1 3 8 5 4
        2 2 4 5 3 1;
param cij:  1  2  3  4  5 := #від A (↓) до B (→)
        1 20  3 20 20 20
        2 20 20 20 20 20
        3 20 20 20  4 20;
param a:= #Продукція в A
1 1000 2 1500 3 1200; #A_1, A_2, A_3
param b:= #Потреби B
1 900 2 700 3 1000 #B_1, B_2, B_3
```

* Безкоштовні версії програми знаходяться за посиланням: http://ampl.com/try-ampl/download-a-free-demo/

```
4 500 5 600; #B_4, B_5
#Межі на пропускні спроможності D
param d_low:= #Нижня
1 0 2 0; #D_1, D_2
param d_up:= #Верхня
1 2500 2 1200; #D_1, D_2
solve; #Розв'язати задачу (1) – (6)
#Роздрукувати значення цільової функції, час розв'язку
display f_opt, _solve_time;
#Роздрукувати значення оптимальних змінних
display x; display y; display z;
```

Тут, для того, щоб не враховувалися перевезення між постачальниками та споживачами, зв'язки між якими відсутні, витрати на перевезення одиниці продукції між ними вибрані рівними 20, які є достатньо великими у порівнянні зі значеннями витрат із табл. 1.

Програма gurobi 7.0.2 знайшла оптимальний розв'язок за 6 симплексних ітерацій. Він збігається з розв'язком з [2, с.235] і представлений на рис. 2. Значення цільової функції дорівнює 20 000 умовних одиниць.

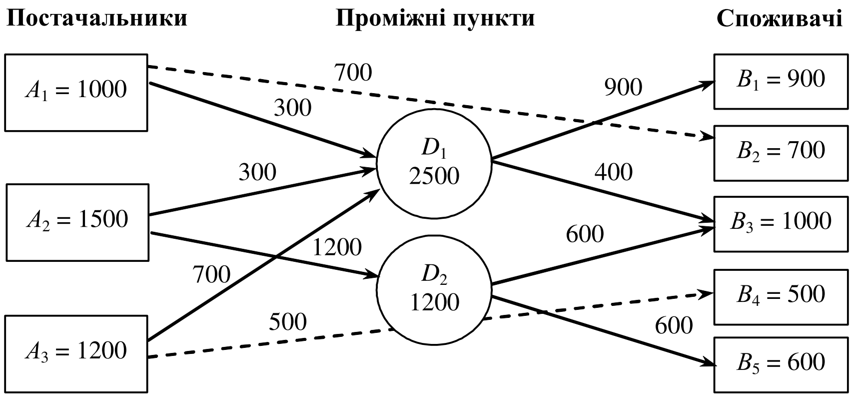


Рис. 2. Оптимальний план перевезень продукції для прикладу 1

Для отриманого розв'язку склад D_1 завантажений на 1300 одиниць з 2500 можливих, що складає приблизно половину його завантаження. Тому модифікуємо приклад 1 наступним чином.

Приклад 2. *Все теж саме, що і в прикладі 1, але склад* D_1 *має нижню межу на пропускну спроможність* $d_1^{low} = 1500$*, що складає 3/5 від його максимального завантаження* $d_1^{up} = 2500$.

Для прикладу 2 вхідні дані в AMPL-коді потребують єдиної зміни, щоб врахувати нові обмеження на пропускну здатність першого проміжного пункту. Програма gurobi 7.0.2 знайшла оптимальний розв'язок за 8 симплексних ітерацій. Йому відповідає значення цільової функції, яке дорівнює 20 200 умовних одиниць, та оптимальний план перевезень, представлений на рис. 3.

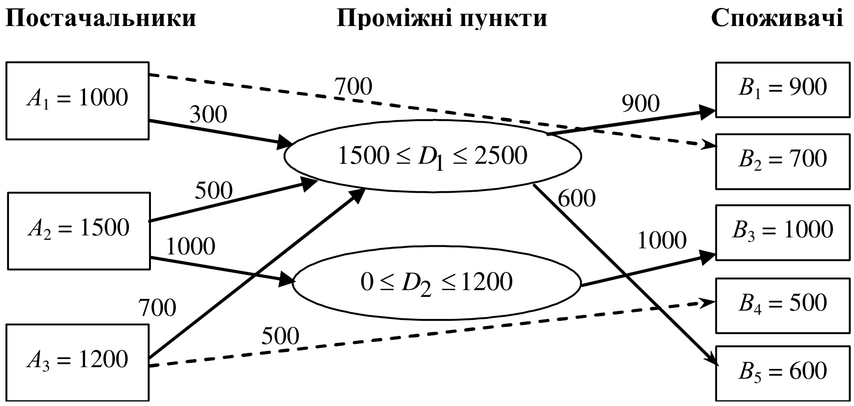


Рис. 3. Оптимальний план перевезень продукції для прикладу 2

З другого розв'язку бачимо, що у порівнянні з першим відбувся значний перерозподіл потоків та об'ємів транспортування продукції по них. В першому випадку з філіалу A_2 до складу D_1 перевезено 300 одиниць продукції, а до складу D_2 – 1200 одиниць, в другому відповідно 500 і 1000 одиниць. При транспортуванні продукції зі складів до споживачів кількість поставок зменшилась з чотирьох до трьох та значно перерозподілилися об'єми поставок. Так, наприклад, в першому випадку зі складу D_2 споживачам B_3 та B_5 доставлено по 600 одиниць продукції, а в другому – 1000 одиниць доставлено споживачу B_3, а поставка для B_5 відсутня. Оскільки, частина продукції від постачальника A_2 доставлена споживачу B_3 через проміжний пункт D_2, що є його фактичним завантаженням, то доцільно розглянути можливість прямої поставки з A_2 до B_3. Це може бути пов'язано з тим, що перевантаження продукції на складах має власні витрати, які не враховані в задачі.

3. Двоїстий алгоритм для задачі (1)–(6). Для розв'язання задачі (1)–(6) можна використовувати методи негладкої оптимізації у сполученні зі схемою декомпозиції за обмеженнями [7,3]. Вона пов'язана з лагранжевою релаксацією за частиною обмежень та дозволяє враховувати специфічні особливості матриці обмежень (2)–(6), а саме те, що обмеження (2)–(5) пов'язанні з трьома транспортними задачами: задача «постачальники–споживачі», задача «постачальники–проміжні пункти» та задача «проміжні пункти–споживачі». Наприклад, якщо знайдені сумарні потоки через проміжні пункти, то задачу (1)–(6) можна замінити класичною транспортною задачею, де новими постачальниками будуть постачальники та проміжні пункти, а новими споживачами – проміжні пункти та споживачі. Тому очевидно, що одними з обмежень, які підлягають лагранжевій релаксації будуть обмеження (4), які зв'язують між собою постачальників і споживачів через проміжні пункти.

В результаті використання схеми декомпозиції за обмеженнями отримуємо негладку двоїсту задачу, вигляд якої будуть визначати ті обмеження, які вибрано для лагранжевої релаксації. Справедливе наступне твердження.

Лема 3. Для задачі (1)–(6) мінімальне значення цільової функції $f^* = \psi^*$, де ψ^* є розв'язком задачі негладкої оптимізації

$$\psi^* = \psi(u^*, v^*) = \max_{u,v} \psi(u,v) \tag{25}$$

в якій увігнута негладка функція $\psi(u,v)$ від змінних $u = (u_1, \ldots, u_m)$ та $v = (v_1, \ldots, v_m)$ визначена наступним чином:

$$\psi(u,v) = -\sum_{j=1}^{m} b_j u_j +$$

$$+\min_{x,y,z}\left\{\sum_{i=1}^{m}\sum_{k=1}^{l}(c_{ik}+v_k)x_{ik} + \sum_{k=1}^{l}\sum_{j=1}^{m}(c_{kj}+u_j-v_k)y_{kj} + \sum_{i=1}^{m}\sum_{j=1}^{n}(c_{ij}+u_j)z_{ij}\right\} \tag{26}$$

за обмежень

$$\sum_{k=1}^{l} x_{ik} + \sum_{j=1}^{n} z_{ij} = a_i, \quad i = 1, \ldots, m, \tag{27}$$

$$d_k^{low} \le \sum_{j=1}^{n} y_{kj} \le d_k^{up}, \quad k = 1, \ldots, l, \tag{28}$$

$$x_{ik} \ge 0,\ y_{kj} \ge 0,\ z_{ij} \ge 0,\ i = 1, \ldots, m, k = 1, \ldots, l,\ j = 1, \ldots, n. \tag{29}$$

Задача (25) є двоїстою задачею, яка отримана із ЛП-задачі (1)–(6) за допомогою лагранжевої релаксації обмежень (3) та (4), яким відповідають множники Лагранжа $u = (u_1, \ldots, u_m)$ та $v = (v_1, \ldots, v_m)$. Кількість змінних в задачі (25) дорівнює $n + l$ та буде невеликою, коли є небагато проміжних пунктів та споживачів. Для її розв'язання можна використовувати субградієнтні методи з перетворенням простору [7,8], наприклад, r-алгоритми, які дозволяють отримати максимальний обчислювальний ефект.

Для розв'язання задачі (25) обчислювальна складність двоїстого алгоритму визначається обраним методом негладкої оптимізації та методом розв'язання задачі лінійного програмування (26)–(29) з $nm + ml + nl$ змінними x_{ik}, y_{kj}, z_{ij}. Розв'язок останньої використовується для обчислення значення функції ψ та її суперградієнта в точці (u,v). Задача (26)–(29) розпадається на $m + l$ незалежних підзадач лінійного програмування з одним обмеженням та невід'ємними значеннями змінних. Для її розв'язання потрібно m раз знайти мінімальні елементи в одновимірних масивах довжин n та l, і l раз знайти мінімальний елемент в одновимірному масиві довжини m.

Відмітимо, що двоїстий алгоритм для розв'язання задачі (25) може бути зорієнтовано на випадок великих m (десятки тисяч постачальників) і порівняно невеликих k (сотні проміжних пунктів) та n (тисячі споживачів). Такі розміри представляють проблеми при розв'язанні транспортних ЛП-задач за допомогою стандартного програмного забезпечення. Тому субградієнтні алгоритми з перетворенням простору є перспективними для розв'язання задачі (25), так як вони здатні значно значно збільшити розміри ефективно вирішуваних задач вигляду (1)–(6) по відношенню до того, що дозволяє або симплекс-метод, або метод внутрішніх точок. Відмітимо, що для двоїстого алгоритму легко лінійну функцію витрат замінити на квадратичну сепарабельну функцію, при цьому виникають підзадачі, які розглядалися в [9].

Висновки. В статті сформульовано транспортну ЛП-задачу для знаходження оптимального плану перевезень від постачальників до споживачів, використовуючи проміжні пункти з обмеженнями на їх пропускну спроможність. Наведено властивості задачі та показано, що її частковий випадок є двохетапною транспортною задачею з обмеженнями на пропускну спроможність проміжних пунктів. Розроблено AMPL-код для опису транспортної ЛП-задачі та показано, що його можна використовувати для задач з невеликою кількістю змінних на персональному комп'ютері (для задач великих розмірів можна використовути NEOS-сервер). Наведено демонстраційні приклади, які дозволяють зрозуміти – як розв'язувати задачі такого типу за допомогою наявного програмного забезпечення. Побудована задача негладкої оптимізації для знаходження оптимальних двоїстих змінних, які відповідають частині обмежень в транспортній ЛП-задачі. Вона дозволяє на основі методів негладкої оптимізації розробляти двоїсті алгоритми, які за відповідних умов будуть значно ефективнішими за стандартне програмне забезпечення.

Наведена транспортна ЛП-задача є актуальною для агропідприємств, які займаються розподіленням та доставкою вирощеної продукції для продажу або переробки на власних потужностях, наприклад Миронівського хлібопродукту (МХП). В якості проміжних пунктів тут виступають власні та орендовані елеватори (зерносховища), а споживачі мають потужності по переробці продукції. Якщо задачу (1)–(6) адаптувати до мережевої форми з неповними транспортними зв'язками, то її можна використовувати при пошуку оптимальних планів перевезення вирощеної продукції для продажу або переробки на власних потужностях МХП. Оскільки пункти постачання, зберігання та переробки МХП географічно розташовані по всій території України, то задачі (1)–(6) великих розмірів є дуже доречними.

Транспортна ЛП-задача та її модифікації можуть бути використані студентами вищих навчальних закладів для розробки власних програмних проектів, пов'язаних з теорією та методами оптимізації. Наведені AMPL-коди при відповідному їх доопрацюванні у поєднанні з програмою **gurobi** планується використати при вивченні навчальної дисципліни «Мережеві задачі оптимізації» магістрантами спеціальностей 8.122 «Комп'ютерні науки та інформаційні технології» та 8.121 «Інженерія програмного забезпечення» ДВНЗ «Ужгородський національний університет» [10].

Бібліографічні посилання

1. **Карагодова, О.О.** Дослідження операцій: Навч. посіб. [Текст] / О.О. Карагодова, В.Р. Кігель, В.Д. Рожок – К.:Центр учбової літератури, 2007. – 256 с.
2. **Наконечний, С.І.** Математичне програмування: Навч. посіб. [Текст] / С.І. Наконечний, С.С. Савіна – К.: КНЕУ, 2003. – 452 с.
3. **Михалевич, В.С.** Оптимизационные задачи производственно-транспортного планирования [Текст] / В.С. Михалевич, В.А. Трубин, Н.З. Шор. – М.: Наука, 1986. – 264 с.

4. **Соломон, Д.И.** Дробное программирование и недифференцируемая оптимизация [Текст] / Д.И. Соломон – Кишинэу: Эврика, 2010. – 556 с.
5. **Fourer, R.** AMPL, A Modeling Language for Mathematical Programming [Text] / R. Fourer, D. Gay, B. Kernighan.– Belmont: Duxburry Press, 2003. – 517 p.
6. **NEOS Solver** [Electronic resourse]: https://neos-server.org/neos/solvers/.
7. **Шор, Н.З.** Методы минимизации недифференцируемых функций и их приложения [Текст] / Н.З. Шор – К.: Наук. думка, 1979. – 199 с.
8. **Стецюк, П.И.** Методы эллипсоидов и *r*-алгоритмы [Текст] / П.И. Стецюк – Кишинэу: Эврика, 2014. – 488 с.
9. **Стецюк, П.И.** Двойственный алгоритм решения задачи сепарабельного квадратичного программирования с одним ограничением и границами на переменные [Текст] / П.И. Стецюк, А.В. Фесюк // Питання прикладної математики і математичного моделювання. – Дніпропетровськ: Дніпропетровський національний університет ім. Олеся Гончара. – 2015. – С.191–200.
10. **Стецюк, П.І.** Мережні інформаційні технології: методичні рекомендації до вивчення курсу [Текст] / П.І. Стецюк., О.В. Міца., В.І. Пецко. – Ужгород: Видавництво УжНУ «Говерла», 2014. – 65 с.

Надійшла до редколегії: 11.02.2017

ЗМІСТ

Наукове видання

ПИТАННЯ ПРИКЛАДНОЇ МАТЕМАТИКИ І МАТЕМАТИЧНОГО МОДЕЛЮВАННЯ

Збірник наукових праць

Випуск 17

Українською, російською та англійською мовами

Свідоцтво про державну реєстрацію
державного засобу масової інформації
серія КВ № 5713 від 24.12.2001 р.

Комп'ютерна верстка *Н.Є. Яцечко*

Підписано до друку 16.08.17. Формат 60×84 $\frac{1}{16}$. Папір друкарський.
Друк плоский. Гарнітура Times New Roman. Ум. друк. арк. 13,5. Ум. фарбовідб. 13,5. Обл.-вид. арк. 12,6. Вид. № 1788. Тираж 100 пр. Зам. № 161.

ПП «Ліра ЛТД», вул. Погребняка, 25, м. Дніпро, 49010.
Свідоцтво про внесення до Державного реєстру
серія ДК № 188 від 19.09.2000 р. Фактична адреса: вул. Наукова, 5