УДК 004.023

ТРАНСПОРТНА ЗАДАЧА ЗА КРИТЕРІЄМ ЧАСУ

Селезньова О.О.
Національний технічний університет України «Київський політехнічний інститут»
Кафедра автоматизованих систем обробки інформації та управління
E-mail: ok7ana@ua.fm

Анотація

Селезньова О.О. Транспортна задача за критерієм часу. *В статі розглянуто транспортну задачу, у якій метою є мінімізація загального часу перевезення продукції. Основною частиною застосованого алгоритму розв'язання задачі є побудова розвантажувального циклу. У статті досліджувались різні правила його побудови.*

Загальна постановка проблеми

В прикладних задачах, метою яких є знаходження плану перевезень, часто виникає така ситуація, коли первинним критерієм пошуку є не вартість перевезень, а час, адже існують такі перевезення, коли можна знехтувати витратами на перевезення заради збереження продукції, наприклад, швидкопсувного продукту і доставки його до місця призначення. Так виникає транспортна задача за критерієм часу, яка відноситься до класу мінімаксних задач оптимізації.

Постановка задачі

Задано m пунктів постачання A_i $(i=\overline{1,m})$ з відповідними об`ємами запасів $a_1, a_2, ..., a_m$ одиниць продукції та n споживачів B_j $(j=\overline{1,n})$, потреби яких становлять відповідно $b_1, b_2, ..., b_n$, причому $\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j$. Позначимо через x_{ij} — обсяг продукції, що перевозиться від i-го $(i=\overline{1,m})$ постачальника j-му $(j=\overline{1,n})$ споживачеві. Витрати часу на здійснення перевезень від постачальника A_i до споживача B_j становлять величину t_{ij}, при цьому вважається, що час не залежить від обсягів перевезень x_{ij}. Необхідно знайти план перевезень, що задовольняє наступним умовам:

$$\sum_{j=1}^{n} x_{ij} = a_i,\ i=\overline{1,m},$$

$$\sum_{i=1}^{m} x_{ij} = b_j,\ j=\overline{1,n}$$

і час $T = \max\limits_{(ij)/x_{ij}>0} \{t_{ij}\}$, який витрачатиметься на всі перевезення, був би мінімальним.

Математичну модель, яка має нелінійну та недиференційовану цільову функцію, можна звести до лінійної шляхом введення mn бульових змінних. Нехай

$$y_{ij} = \begin{cases} 1, x_{ij} > 0, \\ 0, x_{ij} = 0,\ i=\overline{1,m},\ j=\overline{1,n}. \end{cases}$$

Така сукупність змінних дозволить нам позбавитися умов “якщо $x_{ij} > 0$ ” і замінити цільову функцію $\max\limits_{(ij)/x_{ij}>0} \{t_{ij}\} \to \min$ на систему

$$\begin{cases} \max\limits_{ij} \{t_{ij} y_{ij}\} \to \min, \\ x_{ij} \le M y_{ij}, \quad i = \overline{1,m},\ j = \overline{1,n}, \\ y_{ij} \in B, \quad i = \overline{1,m},\ j = \overline{1,n}, \end{cases}$$

де M – достатньо велике число, таке, що нерівність $x_{ij} \le M, \quad i = \overline{1,m},\ j = \overline{1,n}$ виконується на усій множині допустимих рішень. Щоб позбавитися від мінімаксності цільової функції, введемо змінну

$$y = \max\limits_{ij} \{t_{ij} y_{ij}\}.$$

Це дозволить нам перейти до задачі пошуку мінімуму лінійної функції однієї змінної:

$$\begin{cases} y \to \min, \\ y \ge t_{ij} y_{ij}, \quad i = \overline{1,m},\ j = \overline{1,n}. \end{cases}$$

Таким чином математична модель вихідної задачі прийме вигляд:

$$\begin{aligned} & y \to \min \\ & y \ge t_{ij} y_{ij}, \quad i = \overline{1,m},\ j = \overline{1,n}, \\ & x_{ij} \le M y_{ij}, \quad i = \overline{1,m},\ j = \overline{1,n}, \\ & \sum_j x_{ij} = a_i, \quad i = \overline{1,m}, \\ & \sum_i x_{ij} = b_i, \quad j = \overline{1,n}, \\ & x_{ij} \ge 0, \quad i = \overline{1,m},\ j = \overline{1,n}, \\ & y_{ij} \in B, \quad i = \overline{1,m},\ j = \overline{1,n}. \end{aligned}$$

Розв’язання задачі

Отже, вихідна задача може бути зведена до лінійної задачі цілочисельного програмування з числом змінних $2mn+1$ (з них mn змінних - булеві) та числом обмежень $2mn+m+n$. Така розмірність отриманої задачі не дозволяє для задач великої розмірності за прийнятний час отримувати оптимальний розв’язок. Для подібних задач прийнято використовувати наближені або евристичні алгоритми.

З урахуванням того, що обмеження цієї задачі співпадають з обмеженнями класичної ТЗЛП природно використати прийоми методу потенціалів [1]. Основними етапами цього методу є: знаходження початкового розв’язку; вибір клітини (маршруту), що розвантажується; побудова розвантажувального циклу (РЦ) для обраної клітини; перехід до нового розв’язку. Виходячи з цього, головною операцією методу є побудова розвантажувального циклу, який дозволяє на кожній ітерації зменшувати загальний час перевезень за рахунок розвантаження клітин (маршрутів), перевезення якими займає найбільше часу. На відміну від транспортної задачі лінійного програмування, для клітини, яка розвантажується, може бути побудовано декілька розвантажувальних циклів [2]. Від того, який саме метод побудови РЦ буде обрано, залежатиме швидкість і якість знаходження розв’язку. Робота присвячена дослідженню особливостей побудови РЦ.

Алгоритм розв'язання задачі можна описати у вигляді такої послідовності кроків.

КРОК 1. Знаходження початкового розв'язку

Знайти допустимий розв`язок задачі x_{ij}, $i=\overline{1,m}, j=\overline{1,n}$.

КРОК 2. Вибір маршруту, що розвантажується

2.1. Знайти $T=\max\limits_{ij/x_{ij}>0}\{t_{ij}\}$. Якщо існують такі клітини, в яких $t_{ij}>T$, то викреслити їх, у подальшому ми їх не розглядаємо.

2.2. Обрати клітину (st), для якої виконується $t_{st}=T$. Для цієї клітини будуємо РЦ.

КРОК 3. Побудова розвантажувального циклу

Перебираючи клітини зліва-направо та зверху-вниз (або зправа-наліво та знизу-вверх – напрямок обирається в залежності від місця розташування розвантажуваної клітини), згідно алгоритму бектрекінгу, побудувати розвантажувальний цикл. В процесі побудови циклу до нього включаються як заняті так і вільні клітини за умови, що всім клітинам (ij) з непарними номерами (вважаючи першою клітину (st)) відповідає $x_{ij}>0$, а з парними номерами – $t_{ij}<T$ (сформувати $I^{н}$ – множину індексів з непарними номерами та I^{n} – множину індексів з парними номерами).

Якщо для клітини (st) не вдалося побудувати РЦ, то зупинити обчислення.

КРОК 4. Перехід до нового розв'язку

4.1. Для побудованого циклу знайти: $\Delta=\min\limits_{(ij)\in I^{n}}\{x_{ij}\}$. Перемістити це число по циклу: $x_{ij}=x_{ij}-\Delta$, де $(ij)\in I^{н}$ та $x_{ij}=x_{ij}+\Delta$, де $(ij)\in I^{n}$.

4.2. Якщо $\Delta=x_{st}$ (клітина (st) розвантажена повністю), то вона в подальшому не розглядається (викреслюється).

4.3. Перейти до кроку 2.

Дослідження задачі

Найбільш трудомістка частина алгоритму – це побудова розвантажувального циклу. У випадку, коли цикл має вигляд прямокутника, кількість розвантажувальних циклів для клітини обмежена зверху величиною $O(mn)$, а у випадку довільного прямокутника – $O(2^{\min(m,n)})$. Кількість ітерацій, яку потрібно виконати, щоб отримати розв`язок задачі, залежить від кількості клітин, які потрібно розвантажити.

Отже, основною проблемою при знаходження розв`язку транспортної задачі за критерієм часу є організація циклу, який може будуватися різними способами. Виходячи з правил побудови розвантажувального циклу, можна зробити висновок, що РЦ може мати вигляд прямокутника або багатокутника. Природно припустити, що якщо в прямокутному розвантажувальному циклі, вершина, протилежна до розвантажувальної, буде мати максимально можливе значення (тобто буде такою, щоб на даній ітерації якомога більше розвантажити клітину), то такий підхід до побудови розвантажувального циклу дозволить знайти розв`язок за меншу кількість ітерацій. При цьому необхідно перебрати усі можливі варіанти РЦ для обраної клітини. Отже маємо визначити, яким саме способом доцільно будувати РЦ. Нехай:

- ПРЦ1 – правило побудови РЦ «перший прямокутник, що розвантажує клітину (st)»;
- ПРЦ2 – правило побудови РЦ «кращий з усіх знайдених прямокутних РЦ»;
- ПРЦ3 – правило побудови РЦ «перший багатокутник, що розвантажує клітину».

Як показали дослідження, на задачах малої розмірності всі три правила побудови РЦ дають оптимальний розв`язок і значної відмінності між ними не спостерігається. Більш суттєві відмінності в ефективності роботи між даними правилами відслідковуються на задачах великої розмірності (рис. 1). Тож, якщо використовується ПРЦ2, то відповідний алгоритм в 62,5% задач дає кращі показники по кількості зроблених ітерацій, ніж алгоритм, що працює за способом ПРЦ1.

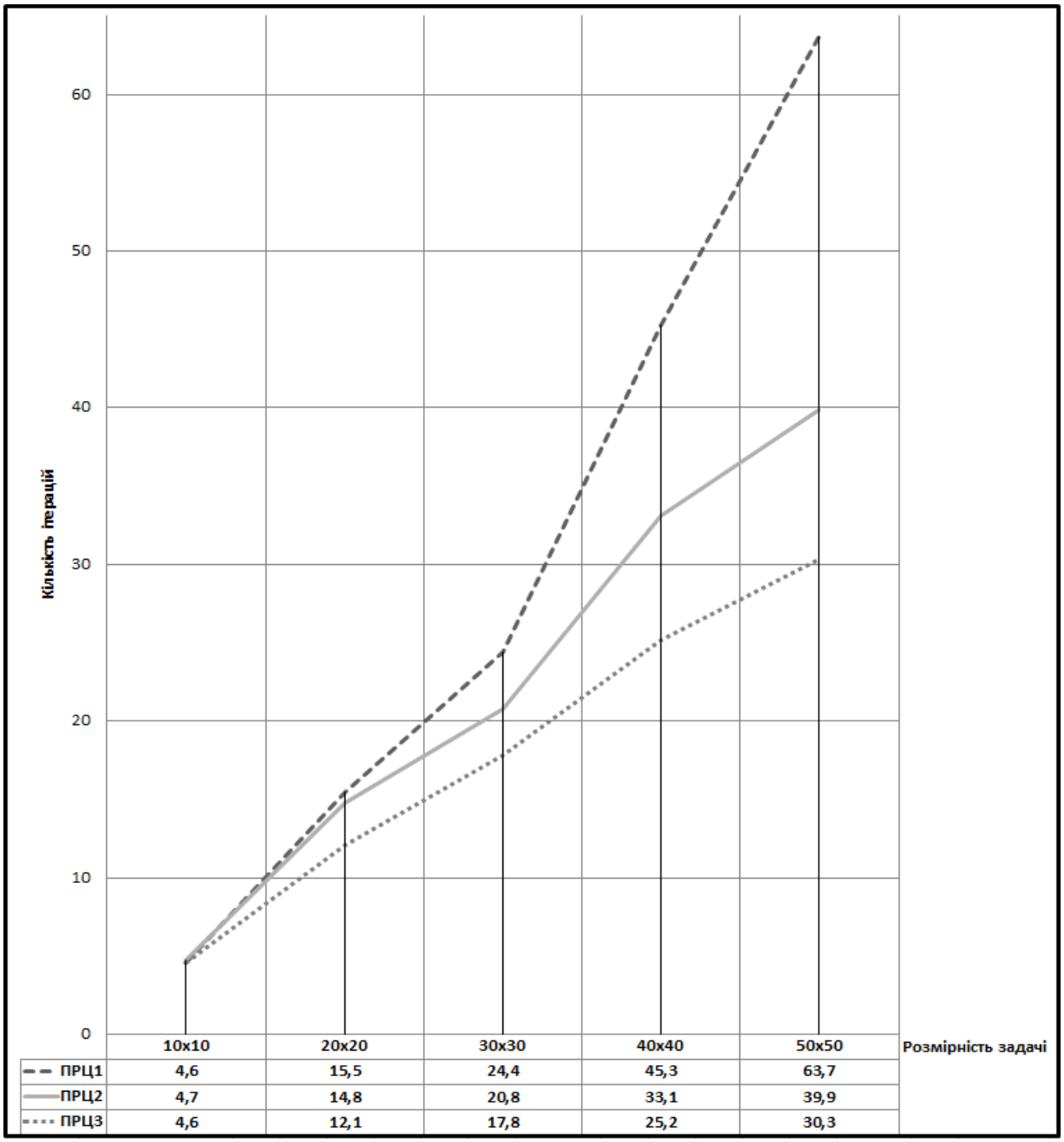

Рисунок 1 – Графік залежності кількості ітерацій, необхідних для знаходження розв’язку задачі, від розмірності задачі

По проведеним дослідженням, щодо ефективності алгоритму за ПРЦ3, як видно з рисунку 1, можна сказати, що він робить меншу кількість ітерацій, ніж інші. Це зумовлено тим, що багатокутник на одній ітерації розвантажує набагато більше клітин, в той час, коли

при побудові циклу прямокутником за одну ітерацію змінюється лише 4 клітини, а відповідно й ітерацій буде зроблено більше, аби повністю розвантажити необхідну клітину.

Ефективність побудови РЦ, окрім кількості зроблених ітерацій при побудові циклу, також може характеризуватися часом, затраченим на будову цього циклу. Найбільше часу на знаходження рішення витрачає алгоритм, в якому закладено побудову РЦ довільним прямокутником (ПРЦ1), а найменшу – цикл багатокутником (ПРЦ3) (рис. 2).

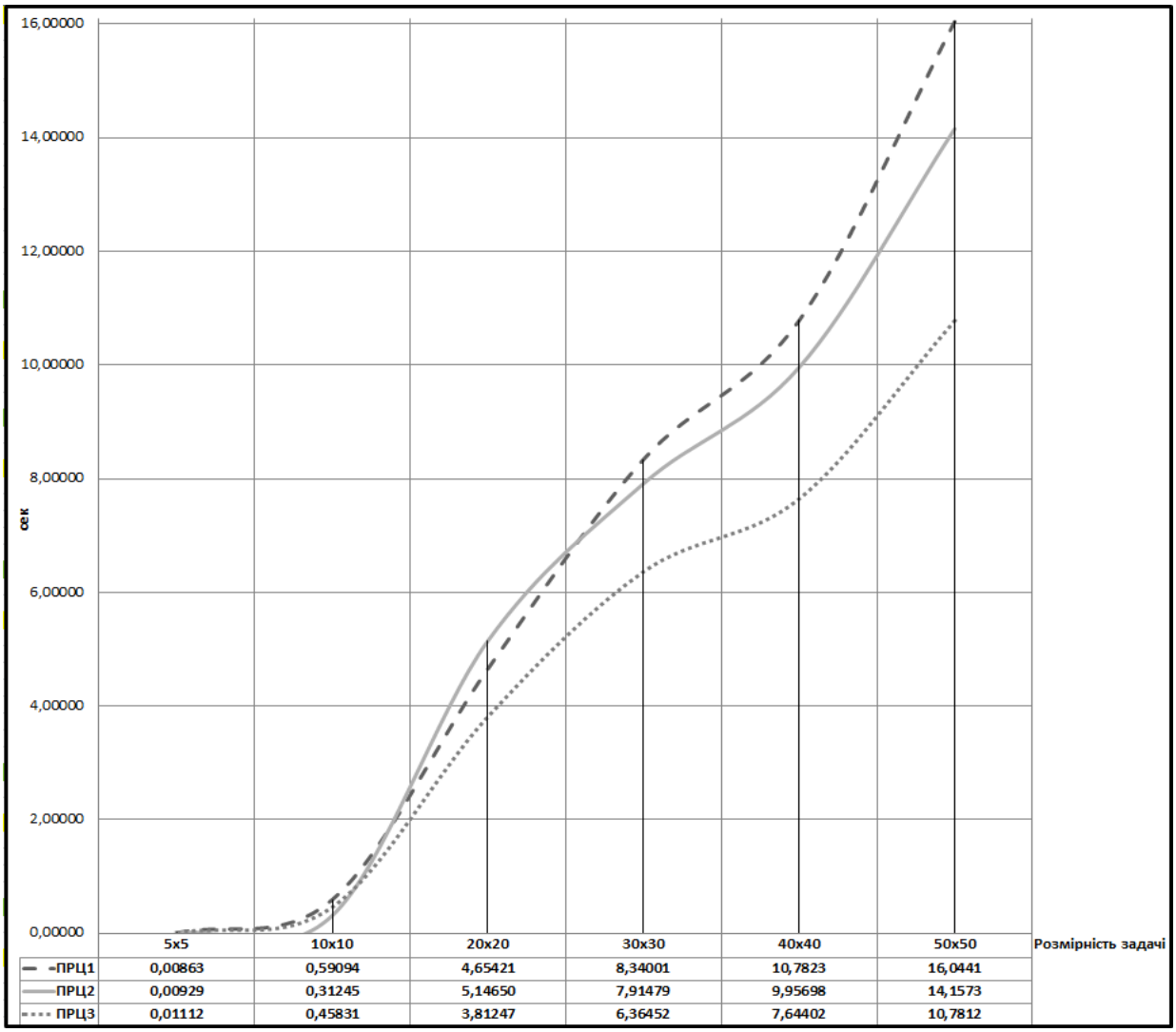

Рисунок 2 – Графік залежності часу знаходження розв’язку від способу побудови циклу

Висновок

Було розглянуто транспортну задачу за критерієм часу. Також були описані особливості розвантажувального циклу та наведено дослідження щодо оцінки ефективності використання різних способів побудови РЦ, які показали що найбільш ефективно, з огляду на кількість ітерації та час, використовувати для побудови розвантажувальний цикл, побудований за ПРЦ3.

Література

1. Калихман И.Л. Линейная алгебра и программирование. – М.: Высшая школа, 1967. – 426 с.

2. Таха Х А. Введение в исследование операций. – М.: Вильямс, 2005. – 912 с.