УДК 519.854

С. В. Чупов (Ужгородський нац. ун-т)

МОДИФІКАЦІЇ АЛГОРИТМУ ПОШУКУ ЛЕКСИКОГРАФІЧНОГО МАКСИМУМУ МНОЖИНИ

This article deals with the issues of improving the efficiency of the algorithms for lexicographical maximum set searching determined by a system of linear inequalities with integrant coefficients and Boolean variables. We propose new search algorithms for finding the lexicographic maximum set as well as the analysis of their work compared to standard search algorithms.

В роботі розглядаються питання підвищення ефективності алгоритму пошуку лексикографічного максимуму множини, яка визначається системою лінійних нерівностей з невід'ємними коефіцієнтами та булевими змінними. Пропонуються нові алгоритми пошуку лексикографічного максимуму множини. Здійснюється аналіз ефективності їх роботи в порівнянні із стандартним алгоритмом пошуку.

1. Постановка задачі. Базовою складовою у методах лексикографічного пошуку оптимального розв'язку задачі дискретної оптимізації є методи пошуку лексикографічних екстремумів множин, що є множиною допустимих значень задачі або її підмножинами [3]. Від ефективності методів пошуку лексикографічних екстремумів множини в значній мірі залежить загальна ефективність пошуку оптимального розв'язку задачі дискретної оптимізації.

В даній роботі досліджується питання підвищення ефективності пошуку лексикографічного максимуму множини, що визначається системою лінійних нерівностей з невід'ємними коефіцієнтами та булевими змінними. Зазначимо, що така множина є множиною допустимих значень для багатьох прикладних задач булевої оптимізації, зокрема для задачі про багатовимірний булевий ранець.

Розглянемо задачу: знайти лексикографічний максимум множини X^D, $x^{max} = max^L X^D$, де

$$X^D = \{x \in B^n \mid Ax \le b,\ a_{ij} \ge 0,\ b_i > 0, i = 1, \dots, m, j = 1, \dots, n\}, \qquad (1)$$

$$B^n = \underbrace{\{0,1\} \times \cdots \times \{0,1\}}_{n}.$$

2. Стандартний алгоритм пошуку лексикографічного максимуму множини $\mathbf{X^D}$. Стандартний алгоритм пошуку лексикографічного максимуму множини X^D полягає в наступному [2]:

Алгоритм ABLexMax1.

1-*ий крок*. Розв'язуємо задачу:

$$x_1^1 = max\left(x_1 \mid x \in X^D,\ x_j = 0, j = 2, \dots, n\right) \qquad (2)$$

Знаходження розв'язку задачі (2) здійснюємо за правилом: $x_1^1 = Min(1, \lfloor \delta_1 \rfloor)$, де $\delta_1 = min\left\{\frac{b_i}{a_{i1}} \;\middle|\; a_{i1} > 0,\ i = 1, \dots, m\right\}$. $x_j^1 = 0, j = 2, \dots, n$.

k-ий крок $(1 < k \le n)$. Розв'язуємо задачу:

$$x_k^k = max\left(x_k \mid x \in X^D, x_j = x_j^{k-1}, j = 1, \ldots, k-1,\ x_j = 0, j = k+1, \ldots, n\right) \quad (3)$$

Знаходження розв'язку задачі (3) здійснюємо за правилом: $x_j^k = x_j^{k-1}$, $j = 1, 2, \ldots, k-1$, $x_k^k = Min(1, \lfloor \delta_k \rfloor)$, де $\delta_k = min\left\{\frac{\beta_i^k}{a_{ik}} \;\middle|\; a_{ik} > 0,\ i = 1, \ldots, m\right\}$, $\beta_i^k = b_i - \sum_{j \in J^k} a_{ij}$, $J^k = \left\{j \in \{1, \ldots, k-1\} \mid x_j^{k-1} = 1\right\}$, $i = 1, \ldots, m$, $x_j^k = 0, j = k+1, \ldots, n$.

Виконавши n кроків за алгоритмом $ABLexMax1$ буде отриманий розв'язок x^n. Значення координат розв'язку x^n отримуються послідовно, починаючи з першої координати, шляхом розв'язання скалярних задач максимізації (2) та (3). Тому, якщо x^{max}– лексикографічний максимум множини X^D, для довільного $p (1 \le p \le n)$ не може виконуватись $x_j^{max} = x_j^n, j = 1, \ldots, p-1$, $x_p^{max} > x_p^n$, що, за означенням лексикографічного впорядкування, означає $x^{max} >^L x^n$. Таким чином довели:

Теорема 1. *Розв'язок x^n, отриманий в результаті застосування алгоритму $ABLexMax1$ є лексикографічним максимумом множини* (2).

Реалізація алгоритму $ABLexMax1$ представлена на рис. 1. В подальшому будуть використовуватись наступні позначення: $from$ $(0 \le from < n)$ – визначає номер кроку $k > 0$ з якого починається робота алгоритму $ABLexMax1$, при цьому $k = from + 1$. Слід зазначити, що коли $from = 0$, тоді будуть виконані усі кроки алгоритму $ABLexMax1$, тобто буде знайдено лексикографічний максимум множини (2). Якщо ж $from > 0$, тоді алгоритм $ABLexMax1$ може бути використаний для пошуку лексикографічного максимуму множини $\left\{x \in X^D \mid x \le^L \overline{x}\right\}$, де $\overline{x} = (\overline{x}_1, \ldots, \overline{x}_{from-1}, 0, 1, \ldots, 1)$.

$$delta_i = \begin{cases} b_i,\ from = 0, \\ b_i - \sum_{j \in J} a_{ij}, J = \{j \in \{1, 2, \ldots, from\} \mid x_j = 1\},\ from > 0, \end{cases}$$

$$i = 1, \ldots, m.$$

```
for (int j = from + 1; j ≤ n; j++){
  double δ_j = 1.0;
  for (int i = 1; i ≤ m; i++){
    if (a_ij > 0){
      δ_j = Min(dlt, delta_i/a_ij);
    }
  }
  x_j = Min(1, ⌊δ_j⌋);
  if (x_j > 0){
    for (int i = 1; i ≤ m; i++){
      delta_i −= a_ij;
    }
  }
}
```

Рис. 1. Стандартний метод пошуку лексикографічного максимуму множини

3. Перша модифікація алгоритму ABLexMax1 .

Визначення на кожному кроці $k (k > 0)$ значення $\delta_k = min\left\{\frac{\beta_i^k}{a_{ik}} \;\middle|\; a_{ik} > 0,\ i = 1, \ldots, m\right\}$ еквівалентно відшуканню найменшого допу-

стимого розв'язку системи нерівностей $\begin{cases} \beta_1^k \le a_{1k}x_k, \\ \dots \\ \beta_m^k \le a_{mk}x_k. \end{cases}$ Враховуючи той факт, що значення змінної x_k — булеве, вона може бути рівною 1 лише при сумісності системи

$$\begin{cases} \beta_1^k \le a_{1k}, \\ \dots \\ \beta_m^k \le a_{mk}. \end{cases} \tag{4}$$

Якщо хоча б для одного $p(1 \le p \le m)$ отримаємо $\beta_p^k < a_{pk}$, тоді змінна x_k може приймати лише нульове значення. Отже виконання кроку $k(k > 0)$ за алгоритмом $ABLexMax1$ еквівалентно встановленню сумісності системи (4).

Теорема 2. *Розв'язання задач скалярної максимізації* (2) *та* (3)*, для кожного* $1 \le k \le n$*, еквівалентно встановленню сумісності системи* (4)*.*

Процедура відповідної модифікації алгоритму $ABLexMax1$ зображена на рис. 2. При цьому використовуються наступні позначення:

$oldJ$ – номер координати значення якої максимізувалося на попередньому кроці;

$xOldJ$ – значення, що було отримано на попередньому кроці;

$$dlt = Min\left(1, \lfloor \delta_j \rfloor\right), \ j = from + 1, \dots, n.$$

```
int oldJ = from;
int xOldJ = 0;
for (int j = from + 1; j ≤ n; j++){
    int dlt = 1;
    if (xOldJ == 1){
        for (int i = 1; i ≤ m; i++){
            delta_i -= a_{i,oldJ};
            if (dlt > 0 ∧ a_ij > 0 ∧ delta_i < a_ij) dlt = 0;
        }
    }
    else{
        for (int i = 1; i ≤ m; i++){
            if (a_ij > 0 ∧ delta_i < a_ij){
                dlt = 0;
                break;
            }
        }
    }
    x_j = dlt;
    oldJ = j;
    xOldJ = dlt;
}
if (xOldJ == 1){
    for (int i = 1; i ≤ m; i++){
        delta_i -= a_{i,oldJ};
    }
}
```

Рис. 2. 1-й модифікований метод пошуку лексикографічного максимуму множини

4. Друга модифікація алгоритму ABLexMax1.

В другій модифікації також використовується теорема 2 для послідовного визначення значень окремих координат, але, при цьому, дослідження здійснюється окремо за кожним обмеженням системи. Процедура другої модифікації алгоритму $ABLexMax1$ зображена на рис. 3.

```
int beginJ = from + 1;
int oldJ = −1;
bool changeDelta = false;
do{
    int maxJ = beginJ;
    int countI = 0;
    if (changeDelta){
        for (int i = 1; i ≤ m; i++){
            delta_i −= a_{i,oldJ};
            int j = maxJ;
            while (j ≤ n ∧ delta_i < a_ij) j += 1;
            if (j > maxJ){
                maxJ = j;
                countI += 1;
            }
        }
    }
    else{
        for (int i = 1; i ≤ m; i++){
            int j = maxJ;
            while (j ≤ n ∧ delta_i < a_ij) j += 1;
            if (j > maxJ){
                maxJ = j;
                if (maxJ > n) break;
                countI += 1;
            }
        }
    }
    if (maxJ > n) break;
    changeDelta = false;
    if (countI = 0){
        changeDelta = true;
        x_maxJ = 1;
        oldJ = maxJ;
        maxJ += 1;
    }
    beginJ = maxJ;
} while (true);
```

Рис. 3. 2-й модифікований метод пошуку лексикографічного максимуму множини

Нехай пошук здійснюється починаючи з координати $p > 0$. Визначимо $p_i = min\left\{j \in \{p, \ldots, n\} \mid \beta_i^k \geq a_{ij}\right\}$, $i = 1, \ldots, m$ та індекс $maxJ = \max\{p_1, \ldots, p_m\}$. Якщо $maxJ = p_1 = \cdots = p_m$, тоді системи нерівностей (4) для $k = p, \ldots, maxJ - 1$ є несумісними, але система (4) для $k = maxJ$ – сумісна. Отже, на підставі теореми 2, можна твердити, що $x_j^{max} = 0$, $j = p, \ldots, maxJ - 1$, $x_{maxJ}^{max} = 1$. При цьому подальший пошук будемо здійснювати починаючи з координати $maxJ+1$.

Якщо ж не всі значення $p_1, \ldots, p_m$ рівні між собою, тоді це означає, що системи нерівностей (4) для $k = p, \ldots, maxJ$ є несумісними, тобто, за теоремою 2, $x_j^{max} = 0,\ j = p, \ldots, maxJ$ і подальший пошук будемо здійснювати починаючи з координати $maxJ$.Таким чином здійснено $maxJ - p + 1$ кроків стандартного алгоритму $ABLexMax1$.

Теорема 3. *Пошук значення $maxJ$, починаючи з координати $p > 0$, еквівалентний виконанню $maxJ - p + 1$ кроків стандартного алгоритму $ABLexMax1$.*

5. Аналіз ефективності методів пошуку лексикографічного максимуму множини $\mathbf{X^D}$. Для аналізу ефективності методів пошуку лексикографічного максимуму множини (2) використовувались серії тестових задач Chu-Beasley про багатовимірний булевий ранець з відомої бібліотеки OR–Library (`people.brunel. ac.uk/~mastjjb/jeb/orlib mknapinfo.html`). У цих задачах цікавим є те, що вони розбиті на групи відповідно до щільності ненульових значень в оптимальному розв'язку. Під щільністю значень розв'язку будемо розуміти відношення кількості ненульових значень до загальної кількості координат розв'язку, що виражене у відсотках. Робота методів аналізувалася на тестових задачах в яких кількість змінних становила 500. Вони, умовно, розбиті на три групи зі щільністю ненульових значень 25%, 50% та 75% в кожній групі відповідно. Кожна з цих груп складається з трьох підгруп в яких кількість обмежень задачі дорівнює 5, 10 та 30 відповідно. Кожна задача розв'язувалась три рази алгоритмом лексикографічного пошуку [1,3] з використанням трьох різних методів пошуку лексикографічного максимуму множини, розглянутих вище. Кількість звертань до методу пошуку лексикографічного максимуму множини в алгоритмі фіксувалася. І після 10000000 звертань робота алгоритму переривалася та фіксувався час його роботи. Розв'язувалось по 5 задач з кожної підгрупи після чого обчислювався середній час їх роботи. Результати дослідження ефективності роботи трьох методів пошуку лексикографічного максимуму множини для кількості обмежень, що рівна 5, 10 та 30 та різних щільностях значень розв'язку представлена на рис. 4., рис 5. та рис 6. відповідно.

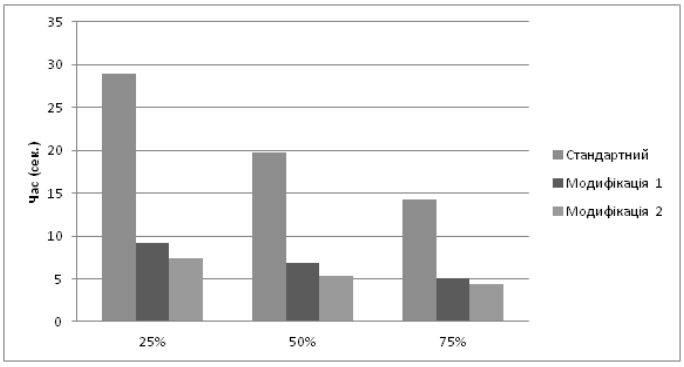

Рис. 4. Середній час для $m = 5$

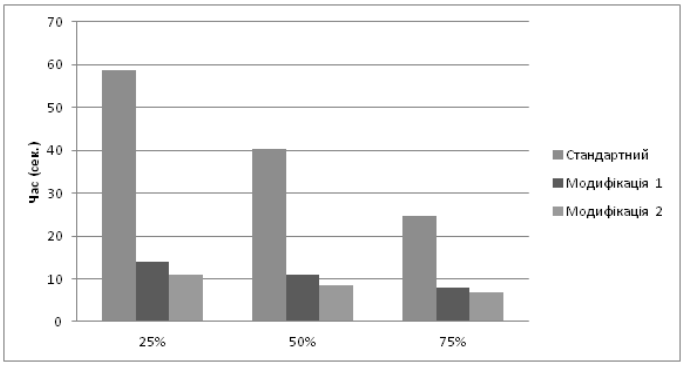

Рис. 5. Середній час для $m = 10$

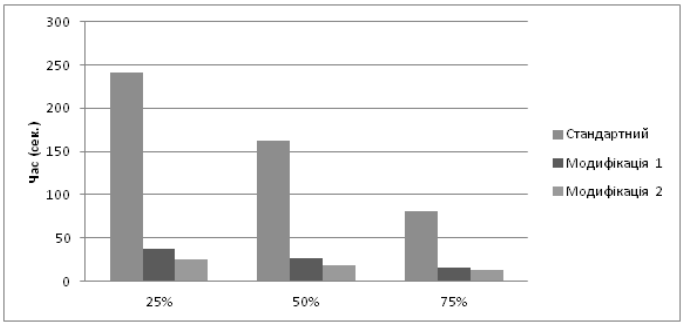

Рис. 6. Середній час для $m = 30$

З наведених діаграм видно, що найкращий (найменший) час показала друга модифікація методу. Якщо через $\hat{t}_2$ позначити середній час роботи другої модифікації алгоритму пошуку лексикографічного максимуму множини (2), через $\hat{t}_s$ – середній час роботи стандартного методу і через $\hat{t}_1$ – середній час роботи першої модифікації алгоритму, тоді ефективність роботи другої модифікації алгоритму в порівнянні з іншими двома методами відображена в наступній таблиці.

	25%		50%		75%	
m	$\hat{t}_s$	$\hat{t}_1$	$\hat{t}_s$	$\hat{t}_1$	$\hat{t}_s$	$\hat{t}_1$
5	$3,91 * \hat{t}_2$	$1,23 * \hat{t}_2$	$3,66 * \hat{t}_2$	$1,27 * \hat{t}_2$	$3,20 * \hat{t}_2$	$1,15 * \hat{t}_2$
10	$5,37 * \hat{t}_2$	$1,28 * \hat{t}_2$	$4,82 * \hat{t}_2$	$1,30 * \hat{t}_2$	$3,57 * \hat{t}_2$	$1,17 * \hat{t}_2$
30	$9,61 * \hat{t}_2$	$1,47 * \hat{t}_2$	$8,66 * \hat{t}_2$	$1,42 * \hat{t}_2$	$6,23 * \hat{t}_2$	$1,15 * \hat{t}_2$

Список використаної літератури

1. *Чупов С.В.* Стохастичний алгоритм лексикографічного пошуку розв'язку булевої задачі про ранець// Праці VII міжнародної школи-семінару "Теорія прийняття рішень"(29 вересня-4 жовтня 2014 р. м. Ужгород). – Ужгород: Ужгородський нац. ун-т, 2014. – с. 263-264.
2. *Чупов С.В.* Алгоритм пошуку цілочислового лексикографічного максимуму множини// Науковий вісник Ужгород. держ. ун-ту: Сер. "Математика". – Ужгород, 1997. – Вип. 2. – С. 122-123.
3. *Червак Ю.Ю., Гренджа В.І., Чупов С.В.* Лексикографічна оптимізація в дискретному програмуванні// Міжнар. конф. "Питання оптимізації обчислень"(Київ, 6-9 жовтня 1997р.). – Праці/ НАН України. – Ін-т кібернетики ім. В.М. Глушкова, 1997. – С. 317-319.

Одержано 07.09.2015