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