Компьютерная Математика

УДК 519.854

С.В. ЧУПОВ

ЭФФЕКТИВНЫЕ АЛГОРИТМЫ ПОИСКА ЛЕКСИКОГРАФИЧЕСКОГО МИНИМУМА МНОЖЕСТВА

Рассматриваются вопросы повышения эффективности работы алгоритма поиска лексикографического минимума множества, которое определяется системой линейных неравенств с неотрицательными коэффициентами и булевыми переменными. Предлагаются новые алгоритмы поиска лексикографического минимума множества. Проводится анализ эффективности их работы по сравнению со стандартным алгоритмом поиска.

Введение. Основным конструктивным элементом в алгоритмах лексикографического поиска оптимального решения задачи дискретной оптимизации есть поиск лексикографического экстремума множества, которое, как правило, является множеством допустимых значений задачи или его подмножеством [1, 2]. Повышение эффективности работы алгоритмов поиска лексикографических экстремумов множества повлияет и на общую эффективность поиска оптимального решения задачи дискретной оптимизации, поскольку каждый шаг в методе лексикографического поиска оптимального решения задачи представляет собой последовательность поисков лексикографических экстремумов множеств.

Постановка задачи. Определим понятие лексикографического упорядочения векторов следующим образом [1]: вектор $x \in R^n$ – лексикографически положительный вектор, $x >^L 0$ ($>^L$ – знак отношения «лексикографически больше»), если первая по порядку отличная от нуля координата этого вектора положительна; вектор $x \in R^n$ лексикографически больше вектора $y \in R^n$, $x >^L y$, если вектор $(x - y)$ – лексикографически положительный, $x - y >^L 0$. Равенство векторов определяется обычным образом. При таком упорядочении любые два вектора одной размерности сравнимы между собой. На основе лексикографического упорядочения векторов, определяется понятие лексикографического минимума множества. Вектор $x^{\min}$ из некоторого множества $G \subset R^n$ называется лексикографическим минимумом этого множества, $x^{\min} = \min^L G$, если любой вектор из множества G лексикографически не меньше чем $x^{\min}$, т. е. для каждого допустимого вектора x из множества G выполняется условие $x \geq^L x^{\min}$ ($\geq^L$ – знак отношения «лексикографически не меньше»). Далее рассматривается задача поиска лексикографического минимума множества X^D, $x^{\min} = \min^L X^D$, где

$$X^D = \left\{ x \in B^n \middle| Ax \geq b, a_{ij} \geq 0, b_i > 0, i = 1,...,m,\ j = 1,...,n \right\},$$

$$B^n = \underbrace{\{0,1\} \times \ldots \times \{0,1\}}_{n}.B^n. \qquad (1)$$

Стандартный алгоритм поиска лексикографического минимума множества X^D. Формальная схема стандартного алгоритма поиска лексикографического минимума множества X^D состоит в следующем [1, 2]:

Алгоритм *ABLexMin*1

<u>*Шаг 1*</u>. Решаем скалярную задачу минимизации:

$$x_1^1 = \min\left(x_1 \middle| x \in X^D, x_j = 1, j = 2,...,n \right) \qquad (2)$$

Отыскание решения задачи (2) осуществляем по правилу:

$$x_1^1 = \max\left(0, \lceil \delta_1 \rceil\right),$$

где $\delta_1 = \max\left\{ \frac{\beta_i^1}{a_{i1}} \middle| a_{i1} > 0, i = 1,...,m \right\}$, $\beta_i^1 = b_i - \sum_{j=2}^{n} a_{ij}, i = 1,...,m$,

$$x_j^1 = 1,\ j = 2,...,n..$$

<u>*Шаг k*</u>. $(1 < k \leq n)$. Решаем скалярную задачу минимизации:

$$x_k^k = \min\left(x_k \middle| x \in X^D, x_j = x_j^{k-1},\ j = 1,...,k-1, x_j = 1,\ j = k+1,...,n \right) \qquad (3)$$

Отыскание решения задачи (3) осуществляем по правилу:

$$x_j^k = x_j^{k-1}, j = 1,...,k-1,,$$

$$x_k^k = \max(0, \lceil \delta_k \rceil),$$

где $\delta_k = \max\left\{ \frac{\beta_i^k}{a_{ik}} \middle| a_{ik} > 0, i = 1,...,m \right\}$, $\beta_i^k = b_i - \sum_{\substack{j=1 \\ x_j^{k-1}=1}}^{k-1} a_{ij} - \sum_{j=k+1}^{n} a_{ij}$, $i = 1,2,...,m,$

$$x_j^k = 1, j = k+1,...,n.$$

После реализации n шагов алгоритма ***ABLexMin*1** будет получено решение x^n.

Теорема 1. Решение x^n, полученное в результате использования алгоритма ***ABLexMin*1**– лексикографический минимум $x^{\min}$ множества (1).

Доказательство. Значения координат решения x^n вычисляются последовательно, начиная с первой координаты, путем решения скалярных задач минимизации (2) и (3). Таким образом, для произвольного индекса $p (1 \le p \le n)$ значение $x_p^n = x_p^p$ есть наименьшим при фиксированных значениях x_j^n, $j = 1,..., p-1$. Поэтому, если $x^{\min}$ – лексикографический минимум множества (1), то для произвольного индекса $p (1 \le p \le n)$ не может иметь место $x_j^{\min} = x_j^n$, $j = 1,..., p-1$, $x_p^{\min} = x_p^n$, что по определению лексикографического упорядочения, означает $x^{\min} <^L x^n$.

Теорема 2. Если множество (1) не пусто, тогда $\delta_k \le 1$, $k = 1,..., n$.

Доказательство. Отметим, что если множество (1) не пусто тогда, по крайней мере, вектор $\overline{1} = (\underbrace{1,...,1}_{n})$ должен принадлежать этому множеству. Это означает, что должны выполняться неравенства $\sum_{j=1}^{n} 1 * a_{ij} \ge b_i \quad i = 1,..., m.$

Имеем $a_{i1} + \sum_{j=2}^{n} a_{ij} \ge b_i \Leftrightarrow b_i - \sum_{j=2}^{n} a_{ij} \le a_{i1}$, $i = 1, 2,..., m$. Если $a_{i1} > 0$, тогда $\dfrac{b_i - \sum_{j=2}^{n} a_{ij}}{a_{i1}} \le 1$, $i = 1,..., m$. Учитывая, что $\beta_i^1 = b_i - \sum_{j=2}^{n} a_{ij}, i = 1,..., m,$ пусть индекс p такой, что $\dfrac{\beta_p^1}{a_{p1}} = \max\left\{ \dfrac{\beta_i^1}{a_{i1}}, a_{i1} > 0, i = 1, 2,...m \right\} \le 1$. Но в формальной схеме алгоритма ***ABLexMin*1** это максимальное значение определяется как δ_1. Таким образом, получили, что $\delta_1 \le 1$. Если $k > 1$, тогда, по аналогии с вышеизложенным, получим, что для того что бы значение δ_k было бы не больше 1, вектор $(x_1^{k-1},..., x_{k-1}^{k-1}, \underbrace{1,...,1}_{n-k+1})$ должен принадлежать X^D, но этот вектор является решением предыдущей задачи. Значит он должен быть допустимым.

Реализация алгоритма ***ABLexMin*1** показана на рис. 1. В дальнейшем будут использоваться следующие обозначения:

$from (0 \le from < n)$ – определяет номер шага алгоритма $k > 0$ с которого начинается работа алгоритма ***ABLexMin*1**, при этом $k = from + 1$. Следует отметить, что когда $from = 0$, тогда будут выполнены все n шагов алгоритма ***ABLexMin1***, т. е. будет найден лексикографический минимум множества (1).

Если же $from > 0$, тогда алгоритм ***ABLexMin*1** может быть использован для поиска лексикографического минимума множества $\left\{ x \in X^D \middle| x \geq^L \bar{x} \right\}$, где $\bar{x} = (\bar{x}_1, ..., \bar{x}_{from-1}, 1, 0, ..., 0)$. $delta_i = \begin{cases} \beta_i^1, \ from = 0 \\ \beta_i^{from+1}, \ from > 0 \end{cases}$, $i = 1, ..., m$.

```
for (int j = from + 1; j ≤ n; j + +){
double δ_j = 0.0;
  for (int i = 1; i ≤ m; i + +){
    if (a_ij > 0){
      δ_j = Max(δ_j, (delta_i + a_ij)/a_ij);
    }
  }
x_j = Max(0, ⌈δ_j⌉);
  if (x_j = 0){
    for (int i = 1; i ≤ m; i + +){
      delta_i += a_ij;
    }
  }
}
```

РИС. 1. Стандартный метод поиска лексикографического минимума множества (1)

Первая модификация алгоритма *ABLexMin*1. Определение на каждом шаге $k(k > 0)$ значения $\delta_k = \max\left\{ \frac{\beta_i^k}{a_{ik}} \middle| a_{ik} > 0, i = 1, ..., m \right\}$ эквивалентно отысканию наибольшего допустимого решения системы неравенств $\begin{cases} \beta_1^k \leq a_{1k} x_k \\ \cdots \\ \beta_m^k \leq a_{mk} x_k \end{cases}$.

Учитывая то, что значение переменной x_k – булево, она может быть равной нулю только при выполнении условий (4):

$$\begin{cases} \beta_1^k \leq 0 \\ \cdots \\ \beta_m^k \leq 0 \end{cases}. \tag{4}$$

Если хотя бы для одного $p(1 \leq p \leq m)$ получим что $\beta_p^k > 0,$ тогда переменная x_k может иметь значение только равное 1. Таким образом выполнение шага $k(k > 0)$ алгоритмом ***ABLexMin*1** эквивалентно определению не положительности для всех значений $\beta_i^k,\ \ i = 1,\ldots,m$ из системы (4), т. е. определению ее совместности. Исходя из вышеизложенного, следует теорема.

Теорема 3. Решение задач скалярной минимизации (2) и (3), для каждого $k = 1,...,n,$ эквивалентно определению совместности системы (4).

Использование теоремы 3 позволяет построить новую модификацию алгоритма поиска лексикографичкского мимнимума множества (1). Процедура соответствующей модификации алгоритма ***ABLexMin*1** показана на рис. 2. При этом используются следующие обозначения:

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

РИС. 2. 1-й модифицированный метод поиска лексикографического минимума множества (1) *oldJ* – номер координаты, значение которой минимизировалось на предыдущем шаге;

xOldJ – значение координаты, полученное на предыдущем шаге;

$$dlt = \max\left(0, \lceil \delta_k \rceil\right), \;\; k = from + 1, ..., n.$$

Следует отметить, что из определения $delta_i$, $i = 1,...,m$, на каждом шаге минимизации переменной $x_k (k > from)$, следуют равенства $delta_i = \beta_i^k - a_{ik}$, $i = 1,...,m$. На основании теоремы 3, значение переменной x_k может быть равным 0 только в случае совместности системы:

$$\begin{cases} \beta_1^k \le 0 \\ \cdots \\ \beta_m^k \le 0 \end{cases} \Leftrightarrow \begin{cases} \beta_1^k - a_{1k} + a_{1k} \le 0 \\ \cdots \\ \beta_m^k - a_{mk} + a_{mk} \le 0 \end{cases} \Leftrightarrow \begin{cases} delta_1 \le -a_{1k} \\ \cdots \\ delta_m \le -a_{mk} \end{cases}. \quad (5)$$

Если хотя бы для одного $p(1 \le p \le m)$ получим, что $delta_p > -a_{pk}$, тогда переменная x_k может принимать только единичное значение. Это значит, что выполнение шага $k(k > 0)$ алгоритмом ***ABLexMin*1**, также эквивалентно определению совместности системы (5).

Вторая модификация алгоритма *ABLexMin1*.

Во второй модификации также используется теорема 3 для обоснования последовательной минимизации значений координат решения, но, при этом, исследование проводится отдельно по каждому ограничению системы $Ax \ge b$. Процедура второй модификации алгоритма ***ABLexMin1*** показана на рис. 3.

Предположим что поиск лексикографического минимума осуществляется начиная с координаты $p > 0$. Определим значение $p_i = \min\left\{ j \in \{p,...,n\} \middle| \, | \, delta_i \le -a_{ij} \right\}$, $i = 1,...,m$ и индекс $mJ = \max\{p_1,..., p_m\}$. Если $mJ = p_1 = \cdots = p_m$, тогда системы неравенств (5) для $k = p,..., mJ - 1$ являются несовместными, но система (5) для $k = mJ$ – совместна. Таким образом, на основании теоремы 3, можно утверждать, что $x_j^{\min} = 1, j = p,..., mJ - 1$, $x_{mJ}^{\min} = 0$. При этом дальнейший поиск будем осуществлять начиная с координаты $mJ + 1$. Если же не все значения $p_1,..., p_m$ равны между собой, тогда это означает, что системы неравенств (5) для $k = p,..., mJ$ являются несовместными, т. е., по теореме 3, $x_j^{\min} = 1, j = p,..., mJ$ и дальнейший поиск будем осуществлять начиная с координаты mJ. В результате такого подхода будет проведено $mJ - p + 1$ шагов стандартного алгоритма ***ABLexMin*1**. Исходя из вышеизложенного, теорема доказана.

Теорема 4. Поиск значения mJ, начиная с координаты $p > 0$, эквивалентен выполнению $mJ - p + 1$ шагов стандартного алгоритма $\boldsymbol{ABLexMin}\mathbf{1}$.

```
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 = 0;
    oldJ = maxJ;
    maxJ += 1;
  }
  beginJ = maxJ;
} while (true);
```

РИС. 3. 2-й модифицированный метод поиска лексикографического минимума множества (1)

Анализ эффективности методов поиска лексикографического минимума множества (1). Для анализа эффективности работы методов поиска лексикографического минимума множества (1) использовались серии тестовых линейных булевых задач минимизации, которые генерировались по принципу, описанному в [3]. Множества допустимых значений в этих задачах формировались так, чтобы они имели структуру множества (1). Особенностью данных тестовых задач есть то, что они разбиты на группы соответственно плотности ненулевых значений в оптимальном решении. Это означает, что все допустимые решения задачи имеют приблизительно одинаковую плотность. Под плотностью значений решения будем понимать отношение количества ненулевых значений к общему количеству переменных задачи, выраженное в процентах. Работа алгоритмов анализировалась на тестовых задачах, в которых количество переменных равнялась 250. Они, условно, разбиты на три группы с плотностью ненулевых значений 25 %, 50 % и 75 % в каждой группе соответственно. Каждая из этих групп состоит из 10 подгрупп, в которых количество ограничений задачи равняется 5, 10, 15, 20, 25, 30, 35, 40, 45 и 50. Каждая тестовая задача решалась три раза алгоритмом лексикографического поиска [2] с использованием трех разных методов поиска лексикографического минимума множества, рассмотренных выше. Количество обращений к алгоритму поиска лексикографического минимума множества в основном алгоритме фиксировалась. После 10000000 обращений работа основного алгоритма прерывалась и фиксировалось время его работы. Решалось по 5 задач из каждой подгруппы, после чего подсчитывалось среднее время их работы.

Из полученных результатов можно сделать вывод, что наилучшее (наименьшее) время показала вторая модификация алгоритма. Если $\hat{t}_2$ обозначить среднее время работы второй модификации алгоритма поиска лексикографического минимума множества (1) в секундах, $\hat{t}_s$ – среднее время работы стандартного метода и $\hat{t}_1$ – среднее время работы первой модификации алгоритма, тогда эффективность работы всех трех методов поиска по отношению ко второй модификации алгоритма приведена в таблице.

ТАБЛИЦА. Результаты решения тестовых задач

	25 %			50 %			75 %		
m	$\hat{t}_s$	$\hat{t}_1$	$\hat{t}_2$	$\hat{t}_s$	$\hat{t}_1$	$\hat{t}_2$	$\hat{t}_s$	$\hat{t}_1$	$\hat{t}_2$
5	4,25 $\hat{t}_2$	1,30 $\hat{t}_2$	4,69	5,35 $\hat{t}_2$	1,32 $\hat{t}_2$	5,64	6,40 $\hat{t}_2$	1,37 $\hat{t}_2$	7,67
10	4,45 $\hat{t}_2$	1,24 $\hat{t}_2$	6,42	6,14 $\hat{t}_2$	1,40 $\hat{t}_2$	7,91	7,14 $\hat{t}_2$	1,43 $\hat{t}_2$	10,83
15	5,99 $\hat{t}_2$	1,25 $\hat{t}_2$	8,35	6,90 $\hat{t}_2$	1,30 $\hat{t}_2$	9,38	8,27 $\hat{t}_2$	1,34 $\hat{t}_2$	12,52
20	5,32 $\hat{t}_2$	1,34 $\hat{t}_2$	10,99	7,86 $\hat{t}_2$	1,47 $\hat{t}_2$	12,22	9,32 $\hat{t}_2$	1,41 $\hat{t}_2$	15,09
25	5,33 $\hat{t}_2$	1,25 $\hat{t}_2$	11,87	7,61 $\hat{t}_2$	1,40 $\hat{t}_2$	13,62	9,62 $\hat{t}_2$	1,56 $\hat{t}_2$	16,01
30	5,59 $\hat{t}_2$	1,30 $\hat{t}_2$	13,26	8,16 $\hat{t}_2$	1,37 $\hat{t}_2$	15,97	10,18 $\hat{t}_2$	1,63 $\hat{t}_2$	18,74
35	6,25 $\hat{t}_2$	1,18 $\hat{t}_2$	15,33	8,47 $\hat{t}_2$	1,26 $\hat{t}_2$	16,61	10,07 $\hat{t}_2$	1,38 $\hat{t}_2$	19,64
40	5,63 $\hat{t}_2$	1,24 $\hat{t}_2$	16,94	7,73 $\hat{t}_2$	1,32 $\hat{t}_2$	18,97	10,55 $\hat{t}_2$	1,73 $\hat{t}_2$	22,28
45	5,89 $\hat{t}_2$	1,23 $\hat{t}_2$	18,50	8,87 $\hat{t}_2$	1,41 $\hat{t}_2$	19,95	10,89 $\hat{t}_2$	1,60 $\hat{t}_2$	24,37
50	5,55 $\hat{t}_2$	1,19 $\hat{t}_2$	19,07	8,43 $\hat{t}_2$	1,29 $\hat{t}_2$	19,72	11,38 $\hat{t}_2$	1,50 $\hat{t}_2$	23,84
Среднее	**5,43** $\hat{t}_2$	**1,25** $\hat{t}_2$		**8,66** $\hat{t}_2$	**1,35** $\hat{t}_2$		**9,38** $\hat{t}_2$	**1,50** $\hat{t}_2$	

Выводы. Построены и исследованы два новых метода поиска лексикографического минимума множества, которое определяется системой линейных неравенств с неотрицательными коэффициентами. Робота алгоритмов протестирована на тестовом наборе задач, которые генерировались по правилам, изложенным в [3]. Анализ результатов экспериментальных расчетов показал, что вторая модификация является лучшей по сравнению со стандартным методом и первой модификацией. В среднем, вторая модификация алгоритма поиска лексикографического минимума множества, в зависимости от структуры множества допустимых значений, работает в 5 – 10 раз быстрее, нежели стандартный метод. По сравнению с первой модификацией вторая эффективнее по времени на 25 % – 50 %.

С.В. Чупов

ЕФЕКТИВНІ АЛГОРИТМИ ПОШУКУ ЛЕКСИКОГРАФІЧНОГО МІНІМУМУ МНОЖИНИ

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

S.V. Chupov

EFFICIENT ALGORITHMS FOR LEXICOGRAPHICAL MINIMUM OF A SET SEARCHING

The issues of improving the efficiency of the algorithms for lexicographical minimum of a set searching, which is determined by a system of linear inequalities with nonnegative coefficients and Boolean variables, is considered. We propose new search algorithms for finding the lexicographic minimum of a set as well as the analysis of their efficiency compared to the standard search algorithms.

1. *Червак Ю.Ю., Гренджа В.І., Чупов С.В.* Лексикографічна оптимізація в дискретному програмуванні // Міжнар. конф. Питання оптимізації обчислень, (Київ, 6 – 9 жовтня 1997 р. – Пр. НАН України. – Ін-т кібернетики імені В.М. Глушкова, 1997. – С. 317 – 319.
2. *Червак Ю.Ю., Гренджа В.І., Кузка О.І., Куцик О.В., Чупов С.В.* Система алгоритмів лексикографічного пошуку в дискретному програмуванні // Науковий вісник Ужгород. держ. ун-ту: Сер. «Математика». – Ужгород, 1994. – Вип. 1. – С. 122 – 129.
3. *Сергиенко И.В., Шило В.П.* Задачи дискретной оптимизации: проблемы, методы решения, исследования. – Киев: Наук. думка, 2003. – 264 с.

Получено 22.10.2015

Об авторе:

Чупов Сергей Викторович,
кандидат физико-математических наук,
доцент кафедры системного анализа и теории оптимизации
Ужгородского национального университета.
E-mail: sergey.chupov@gmail.com