ДИСКРЕТНЫЙ АНАЛИЗ И ИССЛЕДОВАНИЕ ОПЕРАЦИЙ
Октябрь—декабрь 2004. Серия 1. Том 11, № 4, 36–43

УДК 519.854

ОБ ОДНОМ ОБОБЩЕНИИ МИНИМАКСНОЙ ЗАДАЧИ О НАЗНАЧЕНИЯХ*)

Н. И. Глебов

Для некоторого обобщения минимаксной задачи о назначениях, представляющего собой транспортную задачу с минимаксным критерием и ограниченными целочисленными переменными, предложен алгоритм, который в общем случае является псевдополиномиальным. В случае несбалансированной минимаксной задачи о назначениях алгоритм полиномиален, а при определенном соотношении параметров задачи его оценка трудоемкости линейным образом зависит от размерности задачи.

Рассматриваемая задача представляет собой обобщение известной минимаксной задачи о назначениях или, более точно, транспортную задачу с минимаксным критерием и ограниченными целочисленными переменными:

$$\max_{i,j} f_{ij}(x_{ij}) \quad \longrightarrow \quad \min_{x} \tag{1}$$

при ограничениях:

$$\sum_{j=1}^{m} x_{ij} = a_i, \quad i = 1, 2, \dots, n, \tag{2}$$

$$\sum_{i=1}^{n} x_{ij} \leqslant b_j, \quad j = 1, 2, \dots, m, \tag{3}$$

$$0 \leqslant x_{ij} \leqslant d_{ij}, \quad x_{ij} - \text{целое}, \quad i = 1, 2, \dots, n; \ \ j = 1, 2, \dots, m, \tag{4}$$

где a_i, b_j, d_{ij} — натуральные числа и $f_{ij}(\cdot)$ — монотонно неубывающие функции.

Обычная минимаксная задача о назначениях соответствует случаю $a_i = b_j = d_{ij} = 1$, $f_{ij}(0) = 0$ и $n = m$. Ссылки на посвященные ей работы можно найти в [2, 3].

*) Исследование выполнено при финансовой поддержке Российского фонда фундаментальных исследований (проект 02–01–01153) и гранта ведущих научных школ РФ (проект НШ–313.2003.1).

Ниже предлагается алгоритм получения точного решения рассматриваемой задачи, который в общем случае является псевдополиномиальным. В случае $a_i = b_j = d_{ij} = 1$ и $f_{ij}(x_{ij}) = c_{ij}x_{ij}$ $(i = 1, 2, \dots, n;$ $j = 1, 2, \dots, m)$ алгоритм оказывается полиномиальным и имеет оценку трудоемкости $O(n^3 + nm)$, так что при $n = O(\sqrt{m})$ трудоемкость алгоритма сравнима по порядку с числом элементов матрицы (c_{ij}) размера $n \times m$.

Алгоритм решения задачи.

Для описания алгоритма нам понадобяться некоторые понятия и обозначения.

Определение 1. Матрицу $x = (x_{ij})$ будем называть *частично допустимым решением* задачи, если ее элементы удовлетворяют ограничениям (2) и (4). Если при этом они удовлетворяют также и ограничению (3), то x будет называться *допустимым решением.*

Определение 2. *Дефектом* частично допустимого решения x будем называть величину $D(x) = \sum\limits_{j=1}^{m}(\sum\limits_{i=1}^{n} x_{ij} - b_j)^+$, где $a^+ = \max\{0, a\}$.

Очевидно, что $0 \leqslant D(x) \leqslant \sum\limits_{i=1}^{n} a_i$ и частично допустимое решение x является допустимым решением задачи (1)–(4), если $D(x) = 0$.

Через F_0 обозначим $\min\limits_{x} \max\limits_{ij} f_{ij}(x_{ij})$, где минимум берется по всем допустимым решениям x. Таким образом, для оптимальности допустимого решения x необходимо и достаточно, чтобы при любых i и j выполнялось неравенство $f_{ij}(x_{ij}) \leqslant F_0$.

Необходимыми условиями существования допустимых решений, а следовательно, и разрешимости задачи (1)–(4), являются неравенства

$$\sum_{j=1}^{m} d_{ij} \geqslant a_i, \quad i = 1, 2, \dots, n.$$

Для существования частично допустимых решений эти условия являются не только необходимыми, но и достаточными. В связи с этим в дальнейшем они наряду с неравенством $\sum\limits_{j=1}^{m} b_j \geqslant \sum\limits_{i=1}^{n} a_i$ будут предполагаться выполненными.

Общая идея алгоритма. Начиная с некоторого частично допустимого решения x, удовлетворяющего условию

$$\max_{i,j} f_{ij}(x_{ij}) \leqslant F_0, \tag{5}$$

будет производиться последовательное преобразование x так, чтобы, во-первых, вновь получаемое частично допустимое решение удовлетворяло условию (5) и, во-вторых, новое частично допустимое решение имело меньший дефект. В случае разрешимости задачи это обеспечит в конечном счете получение искомого решения, т. е. допустимого решения, удовлетворяющего условию оптимальности (5).

Все действия алгоритма в промежутке между двумя последовательными преобразованиями частично допустимого решения (и от момента получения начального частично допустимого решения до первого преобразования этого решения) будем называть *этапом.*Таким образом, число этапов не превосходит $\sum\limits_i a_i$, поскольку при каждом преобразовании решения его дефект уменьшается на единицу.

В процессе работы алгоритма имеется следующая информация:

i1) Частично допустимое решение x, удовлетворяющее условию (5).

i2) Величины $b'_j = b_j - \sum\limits_i x_{ij}$, $j = 1, 2, \ldots, m$ и $D(x) = \sum\limits_j (-b'_j)^+$.

i3) Множество (или список) R номеров свободных и/или помеченных столбцов. (j-й столбец называется *свободным*, если $b'_j > 0$. Некоторым столбцам, для которых $b'_j = 0$, могут быть приписаны метки. Такие столбцы считаются помеченными. В качестве метки столбца, обозначаемой посредством $\nu(j)$, выступает номер некоторой "обработанной" строки i, при этом $x_{ij} > 0$.)

i4) Множество Q номеров обработанных строк и список S номеров i таких необработанных строк, что $\{j \in R \mid x_{ij} < d_{ij}\} \neq \varnothing$.

i5) Номера $j(i),\ i \in Q \cup S$, характеризующиеся следующими свойствами: $j(i) \in R$ и $f_{ij(i)}(x_{ij(i)} + 1) = \min\{f_{ij}(x_{ij} + 1) \mid j \in R,\ x_{ij} < d_{ij}\}$.

Замечание 1. Для каждой обработанной строки i имеем:

а) $f_{ij(i)}(x_{ij(i)} + 1) \leqslant F_0$ (доказательство см. ниже);

б) если $x_{ij} > 0$, то $j \in R$.

Крупноблочная схема алгоритма.

1) Начальная подготовка. (Основное назначение блока — построение начального частично допустимого решения x, удовлетворяющего условию (5)).

2) Если $D(x) = 0$, то Конец (x — оптимальное решение), иначе перейти к п. 3.

3) Подготовительная работа (очередного этапа).

4) Выделение множества строк (для предполагаемой обработки).

5) Определение возможности выполнения преобразования частично допустимого решения x. Если ответ положительный, то переход к п. 7,

в противном случае — переход к п. 6.

6) Обработка строк и переход к п. 4.

7) Преобразование x и переход к п. 2.

Поблочное описание алгоритма.

Блок 1 (*начальная подготовка*). Здесь производится построение начального частично допустимого решения x. Для этого значения элементов x_{ij} i-й строки матрицы x определяются независимо от других строк по следующему правилу: начальные значения элементов полагаются равными нулю, а затем производится последовательное увеличение на единицу одной из компонент x_{il}, удовлетворяющей условиям $x_{il} < d_{il}$ и

$$f_{il}(x_{il}+1) = \min\{f_{ij}(x_{ij}+1) \mid x_{ij} < d_{ij}\}.$$

Указанная процедура должна выполняться, пока имеет место неравенство $\sum\limits_j x_{ij} < a_i$. Построенное таким способом частично допустимое решение x при каждом значении $i = 1, 2, \dots, n$ удовлетворяет условию

$$\max\{f_{ij}(x_{ij}) \mid x_{ij} > 0\} \leqslant \min\{f_{ij}(x_{ij}+1) \mid x_{ij} < d_{ij}\}.$$

Кроме того, как будет показано ниже, оно удовлетворяет также условию (5).

В этом же блоке производится вычисление величин b'_j и $D(x)$.

Блок 2. Действия, выполняемые в этом логическом блоке, вполне очевидны. Напомним лишь, что при равенстве дефекта нулю частично допустимое решение является допустимым, а при выполнении условия (5) оно будет и оптимальным решением задачи (1)–(4).

Блок 3 (*подготовительная работа*). В этом блоке определяются информационные объекты, перечисленные в пунктах i3)–i5). На данной стадии вычислений множество Q пусто, а множество R состоит из номеров всех свободных столбцов (обработанных строк и помеченных столбцов пока нет).

Блок 4 (*выделение множества строк*). В случае $S \neq \varnothing$ из списка S выделяется часть S^0, состоящая из строк i таких, что

$$f_{ij(i)}(x_{ij(i)}+1) = \min\{f_{kj}(x_{kj}+1) \mid x_{kj} < d_{kj},\ k \in S, j \in R\}.$$

В случае $S = \varnothing$ задача (1)–(4) неразрешима, так как ограничения (2)–(4) несовместны.

Блок 5. Здесь осуществляется попытка выбора такой строки $i_0 \in S^0$, что $x_{i_0 j} > 0$ и $b'_j < 0$ при некотором j. Наличие такой строки позволяет произвести нужное (т. е. уменьшающее дефект и сохраняющее условие (5)) преобразование частично допустимого решения x.

Блок 6 (*обработка строк*). Как следует из описания крупноблочной схемы, этот блок вступает в работу, когда $b'_j \geqslant 0$ при любых $i \in S^0$ и $j \in \Pi_i$, где $\Pi_i = \{j \mid x_{ij} > 0\}$.

Сначала последовательно просматриваются все строки из списка S^0 и для каждой такой строки i_0 все столбцы $j \in \Pi_{i_0}$. Если очередной просматриваемый столбец не входит в текущее множество R, то производится его пометка. (В этом и состоит обработка строки i_0). Сама процедура пометки столбца j_0 заключается в следующем:

1) номер j_0 вносится в R;

2) метка $\nu(j_0)$ полагается равной i_0;

3) для каждого номера $i \notin Q \cup S^0$ такого, что $x_{ij_0} < d_{ij_0}$, выполняется либо а), либо в):

а) если $i \in S$ и $f_{ij_0}(x_{ij_0} + 1) < f_{ij(i)}(x_{ij(i)} + 1)$, то положить $j(i) := j_0$;

в) если $i \notin S$, то положить $j(i) := j_0$ и занести номер i в список S.

Далее из списка S удаляются все строки, принадлежащие списку S^0.

Блок 7 (*преобразование матрицы* x). Работа блока происходит в условиях, когда предварительно (в блоке 5) уже выбраны строка $i_0 \in S^0$ и столбец j_0 такие, что $x_{i_0 j_0} > 0$ и $b'_{j_0} < 0$.

По номеру i_0 однозначно определяются последовательности $i_0, \ldots, i_{k-1}$ и $j_1, \ldots, j_k$ такие, что $b'_{j_k} > 0$ $(k \geqslant 1)$, $j_s = j(i_{s-1})$, где $s = 1, 2, \ldots, k$, а $i_s = \nu(j_s)$, $s = 1, 2, \ldots, k-1$. Затем производится преобразование матрицы x и величин b'_j, $D(x)$:

$$x_{i_s j_s} := x_{i_s j_s} - 1; \quad x_{i_s j_{s+1}} := x_{i_s j_{s+1}} + 1; \;\; s = 0, 1, \ldots, k-1,$$
$$b'_{j_0} := b'_{j_0} + 1; \quad b'_{j_k} := b'_{j_k} - 1; \quad D(x) := D(x) - 1.$$

В результате получается новое частично допустимое решение с меньшим дефектом, удовлетворяющее условию (5).

Замечание 2. В описании блока 1 утверждалось, что получаемое в этом блоке частично допустимое решение x в случае разрешимости задачи удовлетворяет условию (5). В самом деле, пусть x^* — оптимальное решение задачи. Если $x_{ij} \leqslant x^*_{ij}$, то $f_{ij}(x_{ij}) \leqslant f_{ij}(x^*_{ij}) \leqslant F_0$. Если $x_{ij} > x^*_{ij}$, то в этом случае в силу ограничения (2) должен существовать номер l такой, что $x_{il} < x^*_{il}$. Тогда

$$f_{ij}(x_{ij}) \leqslant \max\{f_{is}(x_{is}) \mid x_{is} > 0\} \leqslant \min\{f_{is}(x_{is} + 1) \mid x_{is} < d_{is}\} \leqslant$$
$$\leqslant f_{il}(x_{il} + 1) \leqslant f_{il}(x^*_{il}) \leqslant F_0.$$

Замечание 3. Преобразование частично допустимого решения x, производимое в блоке 7, затрагивает лишь некоторые элементы из x, расположенные в строках из $Q \cup S^0$. При этом среди элементов i-й строки может увеличиться на единицу лишь элемент $x_{ij(i)}$ и уменьшиться в этом случае на единицу один элемент этой строки. Учитывая также способы формирования множества Q и номеров $j(i)$, для доказательства того, что новое частично допустимое решение будет удовлетворять условию (5), для каждого $i \in S^0$ достаточно показать справедливость неравенства

$$f_{ij(i)}(x_{ij(i)} + 1) \leqslant F_0.$$

Далее используются обозначения $Z_k = \{1, 2, \ldots, k\}$ и $\overline{Q}$, $\overline{R}$ — для дополнений множеств Q и R относительно множеств Z_n и Z_m соответственно.

Рассмотрим ситуацию, которая возникает сразу после срабатывания блока 4. Прежде всего заметим, что

$$\sum\{x_{ij} \mid (i,j) \in Q \times \overline{R}\} = 0, \quad \sum\{x_{ij} \mid (i,j) \in Z_n \times \overline{R}\} > \sum\{b_j \mid j \in \overline{R}\}.$$

Следовательно,

$$\sum\{x_{ij} \mid (i,j) \in Z_n \times \overline{R}\} = \sum\{x_{ij} \mid (i,j) \in \overline{Q} \times \overline{R}\} > \sum\{b_j \mid j \in \overline{R}\}.$$

Пусть x^* — оптимальное решение задачи (1)–(4). Тогда

$$\begin{aligned}
&\sum\{b_j \mid j \in \overline{R}\} \geqslant \sum\{x^*_{ij} \mid (i,j) \in Z_n \times \overline{R}\} \geqslant \sum\{x^*_{ij} \mid (i,j) \in \overline{Q} \times \overline{R}\},\\
&\sum\{x_{ij} \mid (i,j) \in \overline{Q} \times \overline{R}\} > \sum\{x^*_{ij} \mid (i,j) \in \overline{Q} \times \overline{R}\},\\
&\sum\{a_i \mid i \in \overline{Q}\} - \sum\{x_{ij} \mid (i,j) \in \overline{Q} \times R\}\\
&\qquad\qquad > \sum\{a_i \mid i \in \overline{Q}\} - \sum\{x^*_{ij} \mid (i,j) \in \overline{Q} \times R\},\\
&\sum\{x_{ij} \mid (i,j) \in \overline{Q} \times R\} < \sum\{x^*_{ij} \mid (i,j) \in \overline{Q} \times R\}.
\end{aligned}$$

Из последнего неравенства непосредственно вытекает существование такой пары номеров $(i,j) \in \overline{Q} \times R$, что $x_{ij} + 1 \leqslant x^*_{ij}$. (Отсюда, в частности, следует, что $S \neq \varnothing$, так что в случае $S = \varnothing$ задача не имеет оптимального решения, т. е. неразрешима).

Пусть $x_{pq} + 1 \leqslant x^*_{pq}$, $(p,q) \in \overline{Q} \times R$. Тогда для $i \in S^0$ имеем

$$f_{ij(i)}(x_{ij(i)} + 1) = \min\{f_{kj}(x_{kj} + 1) \mid x_{kj} < d_{kj}, (k,j) \in \overline{Q} \times R\} \leqslant$$

$$\leqslant f_{pq}(x_{pq}+1) \leqslant f_{pq}(x^*_{pq}) \leqslant F_0.$$

Замечание 4. Основным назначением блока 3 является приведение в надлежащий исходный вид информации, перечисленной в пунктах i3)–i5). Если в начале работы алгоритма естественным было считать множества обработанных строк и помеченных столбцов пустыми, то на последующих этапах столь же естественным (и целесообразным с точки зрения сокращения вычислений) было бы учитывать при формировании R и Q имеющуюся на текущий момент информацию и сохранить некоторые строки (столбцы) как обработанные (помеченные). С этой целью проведем необходимый анализ используемой информационной структуры.

Рассмотрим ориентированный граф $G = (V, E)$ с множеством вершин

$$V = \{(i,j) \mid 1 \leqslant i \leqslant n,\ 1 \leqslant j \leqslant m\}$$

и множеством дуг

$$E = \{((i,j),(i,j')) \mid x_{ij} > 0 \text{ и } j' = j(i) \neq j\} \\ \cup \{((i,j),(i',j)) \mid j = j(i) \text{ и } i' = \nu(j)\}.$$

Каждая компонента связности этого графа, отличная от одновершинной, представляет собой ориентированное (от листьев к корню) дерево. Любой максимальный путь в таком дереве имеет вид:

$$(i_0,j_0) \to (i_0,j_1) \to (i_1,j_1) \to (i_1,j_2) \to \cdots \to (i_{k-1},j_{k-1}) \to (i_{k-1},j_k),$$

возможно, без вершины (i_0,j_0). При этом все строки с номерами $i_1,\ldots,$ i_{k-1} являются обработанными, все столбцы с номерами $j_1,\ldots,j_{k-1}$ — помеченными и $b'_{j_k} > 0$. Если к тому же оказывается, что $i_0 \in S^0$ и $b'_{j_0} < 0$, то указанный путь может быть использован для преобразования матрицы x.

После выполнения преобразования матрицы x некоторые из этих путей могут перестать удовлетворять необходимым условиям. Но так будет обстоять дело, вообще говоря, не со всеми максимальными путями и это можно попытаться использовать для экономии вычислений, производя соответствующим образом корректировку множеств Q и R.

Пусть (i_{k-1},j_k) — последняя вершина пути, использованного при выполнении преобразования x, а $T(i_{k-1},j_k)$ — дерево с корнем (i_{k-1},j_k), являющееся компонентой связности графа G. Будем говорить, что дерево T *накрывает* строку i (соответственно столбец j), если в нем имеется вершина вида (i,j') (соответственно (i',j)).

В том случае, когда $b'_{j_k} > 0$ (после преобразования x), из множеств R и Q удаляются все помеченные столбцы и обработанные строки, накрытые деревом $T(i_{k-1}, j_k)$. В случае $b'_{j_k} = 0$ при корректировке Q и R необходимо переводить в разряд необработанных (непомеченных) все строки (столбцы), накрытые не только деревом $T(i_{k-1}, j_k)$, но и всеми другими деревьями с корневыми вершинами, расположенными в столбце j_k. Кроме того, из R следует удалить столбец j_k, поскольку он перестает быть свободным.

Выполненная указанным способом корректировка множества R может привести к нарушению условия "$\Pi_i \subset R$" для обработанных строк. Поэтому заново производится пометка всех удаленных из R столбцов j таких, что $x_{ij} > 0$ и $i \in Q$.

После того, как уточнены множества R и Q, определяются номера $j(i)$ для всех строк i, попавших в список S.

Замечание 5. В общем случае предложенный алгоритм псевдополиномиален (при полиномиальной вычислимости функций f_{ij}.) В случае $a_i = 1$ и $f_{ij}(x_{ij}) = c_{ij}x_{ij}$ алгоритм является полиномиальным и имеет оценку трудоемкости $O(n^3 + nm)$, которая может быть обоснована тем же способом, что и аналогичная оценка из [1]. Таким образом, при $n = O(\sqrt{m})$ трудоемкость алгоритма сравнима по порядку с числом элементов матрицы (c_{ij}) размера $n \times m$.

ЛИТЕРАТУРА

1. **Диниц Е. А.** О решении двух задач о назначении // Исследования по дискретной оптимизации. М.: Наука, 1976. С. 333–348.
2. **Burkard R. E.** Selected topics on assignment problems // Discrete Appl. Math. 2002. V. 123, N 1–3. P. 257–302.
3. **Pferschy U.** Solution methods and computational investigations for the linear bottleneck assignment problem // Computing. 1997. V. 59, N 3. P. 237–258.

Адрес автора:
Институт математики
им. С. Л. Соболева СО РАН,
пр. Академика Коптюга, 4,
630090 Новосибирск, Россия.

Статья поступила
15 июня 2004 г.