

УДК 519.87+519.854

## ЗАДАЧА О ВЫБОРЕ ЦЕН НА ПРОДУКЦИЮ ПРИ УСЛОВИИ ОБЯЗАТЕЛЬНОГО УДОВЛЕТВОРЕНИЯ СПРОСА\*)

*В. Т. Дементьев, Ю. В. Шамардин*

Рассматривается следующая задача. Пусть известны список пунктов производства некоторого продукта и перечень его потребителей, обладающих разными покупательными способностями. Производитель назначает на каждом предприятии свою цену на выпускаемый продукт, но таким образом, чтобы каждый потребитель имел возможность закупки продукта хотя бы в одном пункте производства. Требуется найти цены на продукт, при которых суммарный доход производителя максимален. Показано, что задача NP-трудна. Найдены случаи ее полиномиальной разрешимости. В общей ситуации предложены алгоритм приближенного решения и способ вычисления верхней оценки оптимума задачи.

### Введение

В моделях исследования операций, имеющих экономическое содержание, ценовые или затратные параметры, как правило, предполагаются заданными. Основное внимание уделяется поиску наилучших в том или ином смысле объемов производства, планов поставок, размещений пунктов производства, номенклатур изделий и т. п. Процессы ценообразования в условиях рыночных отношений выдвигают новые задачи, целью которых является нахождение цен на продукцию, наилучших для ее производителя. В данной статье рассматривается одна из простейших моделей такого вида. В идейном и методологическом отношении эта модель родственна моделям из [2, 4], в которых использовалось варьирование ценовых характеристик. Перейдем к содержательному описанию задачи.

Пусть известен список пунктов производства некоторого продукта (или, коротко, предприятий) и задан перечень потребителей этого

---

\*) Исследование выполнено при финансовой поддержке Российского фонда фундаментальных исследований (проект 02-01-00977).

продукта. Каждое предприятие имеет «неограниченный» объем производства, т. е. способно удовлетворить спрос любого числа потребителей. Производитель, владеющий всеми предприятиями, вправе устанавливать в каждом пункте производства любую цену на выпускаемый продукт.

Относительно каждого потребителя известны объем спроса, множество предприятий, доступных данному потребителю, и ценовой порог, характеризующий покупательную способность потребителя. Он выбирает из всех доступных ему предприятий то, на котором цена продукта минимальна, но делает покупку только в случае, если эта минимальная цена не превышает его ценового порога.

Предполагается, что производитель обязан устанавливать такие цены, чтобы весь потребительский спрос был удовлетворен.

В этих условиях требуется найти цены продукта на каждом предприятии, при которых суммарный доход производителя максимален.

Формальная постановка задачи и некоторые ее свойства приведены в разделе 1. В разделах 2 и 3 излагаются частные случаи задачи, в которых удается построить алгоритмы полиномиальной сложности. Для общей ситуации в разделе 4 предлагаются алгоритм приближенного решения и способ вычисления верхней оценки оптимума задачи.

## 1. Постановка задачи

Введем обозначения:

$I = \{1, \dots, n\}$  — множество пунктов производства (предприятий);

$J = \{1, \dots, m\}$  — множество потребителей;

$I_j$  — множество предприятий из списка  $I$ , доступных потребителю  $j \in J$ ;

$b_j, d_j$  — объем спроса и ценовой порог потребителя  $j$ ;

$x_i$  — цена единицы продукта на предприятии  $i \in I$ ;

$x = (x_1, \dots, x_n)$  — вектор цен, выбираемых производителем;

$\varphi_j(x) = \min\{x_i \mid i \in I_j\}$  — минимальная цена продукта среди предприятий, доступных потребителю  $j$ .

Предполагается, что  $b_j > 0$ ,  $d_j > 0$  и  $I_j \neq \emptyset$  при любом  $j \in J$ .

В принятых обозначениях требуется найти максимум суммарного дохода производителя

$$\Phi(x) = \sum_{j \in J} b_j \varphi_j(x) \rightarrow \max_x \quad (1)$$

при ограничениях

$$\varphi_j(x) \leq d_j, \quad j \in J, \quad (2)$$

$$x_i \geq 0, \quad i \in I. \quad (3)$$

Введем ряд обозначений, используемых ниже при анализе сформулированной задачи:

$J_i = \{j \in J \mid i \in I_j\}$  — список потребителей, которым доступно предприятие  $i \in I$ ;

$D_i = \{d_j \mid j \in J_i\}$  — совокупность значений их ценовых порогов;

$X$  — множество векторов  $x = (x_1, \dots, x_n)$ , удовлетворяющих (2) и имеющих компоненты  $x_i \in D_i$ ,  $i \in I$ ;

$\Phi^*$  — оптимум в задаче (1)–(3).

Если при некотором  $i \in I$  множество  $J_i$  пусто, то можно взять произвольное значение  $x_i$  и удалить предприятие  $i$  из списка  $I$ . Поэтому далее предполагается, что  $J_i \neq \emptyset$  при любом  $i \in I$ .

Будем считать, что потребители занумерованы в порядке неубывания их ценовых порогов, т. е.

$$d_1 \leq d_2 \leq \dots \leq d_m. \quad (4)$$

В задаче (1)–(3) переменные  $x_i$  принимают любые неотрицательные значения, но можно ограничиться рассмотрением только множества  $X$ , о чем говорит следующая

**Лемма 1.** *Справедливо равенство*

$$\Phi^* = \max_{x \in X} \Phi(x). \quad (5)$$

**Доказательство.** Обозначим через  $\Phi'$  значение максимума в правой части (5). Поскольку  $\Phi^* \geq \Phi'$ , достаточно показать, что  $\Phi^* \leq \Phi'$ .

Пусть  $x = (x_1, \dots, x_n)$  — произвольный вектор, удовлетворяющий (2) и (3). Положим  $y = (t, x_2, \dots, x_n)$ . Значение  $t$  выберем из условия максимума функции

$$\psi(t) = \Phi(y)$$

при ограничениях

$$\varphi_j(y) \leq d_j, \quad j \in J, \quad t \geq 0. \quad (6)$$

Используя обозначения

$$\alpha_j = \min\{x_i \mid i \in I_j \setminus \{1\}\}, \quad j \in J, \quad K = \{j \in J_1 \mid \alpha_j > d_j\},$$

получаем

$$\psi(t) = t \sum_{j \in K} b_j + \sum_{j \in J_1 \setminus K} b_j \min\{t, \alpha_j\} + \sum_{j \in J \setminus J_1} b_j \alpha_j. \quad (7)$$

Если  $K = \emptyset$ , то ограничения (6) принимают вид  $t \geq 0$  и с учетом (4) максимум функции (7) достигается, в частности, при  $t = d_l$ , где  $l = \max\{j \in J_1\}$ . Если  $K \neq \emptyset$ , то ограничения (6) преобразуются к виду

$0 \leq t \leq d_k$ , где  $k = \min\{j \in K\}$ , и максимум функции  $\psi(t)$  достигается при  $t = d_k$ . Следовательно,  $\psi(x_1) \leq \psi(t)$  и  $\Phi(x) \leq \Phi(y)$ .

Применяя аналогичную процедуру для корректировки компонент  $x_2, \dots, x_n$ , получаем вектор  $z \in X$  такой, что

$$\Phi(x) \leq \Phi(z) \leq \Phi'.$$

В силу произвольности вектора  $x$  имеем  $\Phi^* \leq \Phi'$ . Лемма 1 доказана.

Вычислительную сложность задачи (1)–(3) выясняет

**Теорема 1.** *Задача (1)–(3) является NP-трудной.*

**ДОКАЗАТЕЛЬСТВО.** Достаточно показать, что к (1)–(3) сводится следующая NP-трудная задача о покрытии множества [3].

Пусть заданы множество  $K = \{1, \dots, k\}$  и семейство его подмножеств  $K_1, \dots, K_n$ , объединение которых равно  $K$ . Положим  $I = \{1, \dots, n\}$ , и пусть  $L \subseteq I$ . Множество  $P = \{K_i \mid i \in L\}$  называется *покрытием* множества  $K$ , если объединение множеств  $K_i$ ,  $i \in L$ , равно  $K$ . Требуется найти покрытие  $P^*$  минимальной мощности.

Множество  $I$ , фигурирующее в задаче о покрытии, возьмем в качестве списка предприятий. Число потребителей выберем равным  $m = k + n$ . Их список обозначим через  $J = K \cup \{k + 1, \dots, m\}$ . Объем спроса  $b_j$  принимается равным 1 при любом  $j \in J$ . Ценовой порог  $d_j$  равен 1, если  $j \leq k$ , и равен 2, если  $j > k$ . Множество  $I_j$  предприятий, доступных потребителю  $j \in J$ , выберем следующим образом. Полагаем  $I_j = \{i \in I \mid j \in K_i\}$ , если  $j \leq k$ , и  $I_j = \{j - k\}$ , если  $j > k$ . Отметим, что введенные выше множества  $J_i$  в данной задаче имеют вид  $J_i = K_i \cup \{i + k\}$ ,  $i \in I$ .

Пусть  $x^* = (x_1^*, \dots, x_n^*)$  — оптимальное решение задачи (1)–(3). По лемме 1 можно считать, что  $x^* \in X$ , т. е. компоненты вектора  $x^*$  равны 1 или 2. Положим  $L^* = \{i \in I \mid x_i^* = 1\}$ . С учетом ограничений (2) получаем

$$\Phi^* = \Phi(x^*) = k + |L^*| + 2(n - |L^*|) = k + 2n - |L^*|$$

и  $P^* = \{K_i \mid i \in L^*\}$  является покрытием множества  $K$ .

Пусть  $P = \{K_i \mid i \in L\}$  — произвольное покрытие множества  $K$ . Рассмотрим вектор  $x = (x_1, \dots, x_n)$  с компонентами  $x_i = 1$ , если  $i \in L$ , и  $x_i = 2$ , если  $i \in I \setminus L$ . Ограничения (2) заведомо выполняются при  $j > k$  и справедливы при  $j \leq k$ , поскольку  $P$  — покрытие множества  $K$ . Вычисляя разность между  $\Phi^*$  и величиной  $\Phi(x) = k + 2n - |L|$ , получаем

$$\Phi^* - \Phi(x) = |L| - |L^*| \geq 0,$$

так как  $\Phi^* \geq \Phi(x)$ . В силу произвольности  $P$  покрытие  $P^*$  минимально. Искомое сведение получено. Теорема 1 доказана.

## 2. Сведение к задаче на графе

Задачу (1)–(3) запишем в эквивалентном виде, включив ограничения (2) в функционал. Выберем достаточно большое число  $A$ , например

$$A = \sum_{j \in J} b_j d_j,$$

и введем функции  $\psi_j(x)$ ,  $j \in J$ , полагая  $\psi_j(x) = b_j \varphi_j(x)$ , если  $\varphi_j(x) \leq d_j$ , и  $\psi_j(x) = -A$ , если  $\varphi_j(x) > d_j$ . Учитывая лемму 1, нетрудно убедиться в том, что задача (1)–(3) эквивалентна следующей задаче: найти максимум функции

$$\Psi(x) = \sum_{j \in J} \psi_j(x) \quad (8)$$

при ограничениях

$$x_i \in D_i, \quad i \in I. \quad (9)$$

Предположим, что каждый потребитель имеет не более двух доступных ему предприятий, т. е.

$$|I_j| \leq 2, \quad j \in J.$$

Это условие позволяет ввести в рассмотрение граф  $G = (I, E)$  с множеством вершин  $I$  и множеством ребер  $E$ , состоящий из всех двухэлементных множеств  $I_j$ ,  $j \in J$ . Вершинам и ребрам графа  $G$  поставим в соответствие функции

$$\begin{aligned} f_i(x_i) &= \sum_{j|I_j=\{i\}} \psi_j(x), \quad i \in I, \\ f_{ik}(x_i, x_k) &= \sum_{j|I_j=\{i,k\}} \psi_j(x), \quad \{i,k\} \in E. \end{aligned}$$

В этих обозначениях задача (8), (9) принимает вид: найти максимум функции

$$F(x) = \sum_{i \in I} f_i(x_i) + \sum_{\{i,k\} \in E} f_{ik}(x_i, x_k) \quad (10)$$

при ограничениях

$$x_i \in D_i, \quad i \in I. \quad (11)$$

Рассмотрим задачу (10), (11) в предположении, что граф  $G$  является два-деревом или подграфом некоторого два-дерева.

Класс два-деревьев определяется индуктивно следующим образом:

а) полный граф с тремя вершинами есть два-дерево;

б) граф, который получается из два-дерева в результате выбора двух смежных вершин  $u$ ,  $v$  и добавления вершины  $w$  и ребер  $\{u, w\}$ ,  $\{v, w\}$ , является два-деревом.

В любом два-дереве или его подграфе имеется вершина степени не более 2. На этом свойстве основан следующий алгоритм, который отвечает «да», если граф является два-деревом или его подграфом, и отвечает «нет» в противном случае.

**Шаг 1.** Положить  $I' = I$ ,  $E' = E$  и  $G' = (I', E')$ .

**Шаг 2.** В графе  $G'$  выбрать вершину  $i \in I'$  минимальной степени  $d$ .

**Шаг 3.** Если  $d = 0$ , то положить  $I' = I' \setminus \{i\}$ .

Если  $d = 1$  и  $u$  — вершина, смежная с  $i$  в  $G'$ , то положить  $I' = I' \setminus \{i\}$ ,  $E' = E' \setminus \{i, u\}$ .

Если  $d = 2$  и  $u, v$  — вершины, смежные с  $i$  в  $G'$ , то положить

$$I' = I' \setminus \{i\}, \quad E' = (E' \cup \{u, v\}) \setminus (\{i, u\} \cup \{i, v\}).$$

Если  $d \geq 3$ , то алгоритм заканчивает работу с ответом «нет».

**Шаг 4.** Если  $I' = \emptyset$ , то алгоритм заканчивает работу с результатом «да». В противном случае следует вернуться к шагу 2.

**Конец**

Положим  $p = \max\{|D_i| \mid i \in I\}$ .

**Теорема 2.** Если граф  $G = (I, E)$  является два-деревом или его подграфом, то задача (10), (11) решается методом динамического программирования за время  $O(nr^3)$  при объеме памяти  $O(nr^2)$ .

**ДОКАЗАТЕЛЬСТВО.** Без потери общности граф  $G$  можно считать два-деревом, добавляя к нему в случае необходимости новые ребра и ставя им в соответствие нулевые функции. Вершинные функции  $f_i(x_i)$ ,  $i \in I$ , фигурирующие в (10), сложим с реберными функциями, выбирая произвольное ребро  $\{i, k\}$ , инцидентное вершине  $i$ , и заменяя  $f_{ik}(x_i, x_k)$  на сумму  $f_i(x_i) + f_{ik}(x_i, x_k)$ . В результате задача (10), (11) принимает вид: найти максимум функции

$$F(x) = \sum_{\{i,k\} \in E} f_{ik}(x_i, x_k) \quad (12)$$

при ограничениях

$$x_i \in D_i, \quad i \in I. \quad (13)$$

В силу определения два-дерева в графе  $G = (I, E)$  найдется вершина  $i \in I$  степени 2. Это позволяет редуцировать задачу (12), (13) следующим образом. Обозначим через  $k$  и  $l$  вершины, смежные с  $i$ . Вычислим функцию

$$g_i(x_k, x_l) = \max_{x_i \in D_i} \{f_{ik}(x_i, x_k) + f_{il}(x_i, x_l)\}, \quad x_k \in D_k, \quad x_l \in D_l. \quad (14)$$

Ребру  $\{k, l\} \in E$  поставим в соответствие сумму  $f_{kl}(x_k, x_l) + g_i(x_k, x_l)$ . Из графа  $G$  удаляем вершину  $i$  и ребра  $\{i, k\}$ ,  $\{i, l\}$ . В результате приходим

к задаче того же вида, что и (12), (13), но меньшей размерности. Редуцированный граф  $G$  по-прежнему является два-деревом, поэтому вновь можно применить описанный шаг. Продолжая этот процесс, приходим к заключительному шагу, когда граф  $G$  состоит из двух вершин, например  $i, k$ , и ребра  $\{i, k\}$ , которому соответствует некоторая функция  $f_{ik}(x_i, x_k)$ . Расчет величины

$$\Phi^* = \max\{f_{ik}(x_i, x_k) \mid x_i \in D_i, x_k \in D_k\} \quad (15)$$

завершает прямой ход динамического программирования.

В процессе прямого хода наряду с функциями  $g_i(x_k, x_l)$  запоминаются *условно-оптимальные решения*, т. е. те значения переменных, на которых достигаются экстремумы в (14), (15). Решение исходной задачи (12), (13) восстанавливается обратным ходом по найденным условно-оптимальным решениям.

Оценим сложность описанной вычислительной схемы, полагая, что длительность выполнения арифметической операции равна 1. Отметим, что два-дерево с  $n = |I|$  вершинами имеет  $2n - 3$  ребер. Поэтому число всех функций, рассчитываемых в процессе прямого хода, не превосходит  $O(n)$ . Каждая функция требует  $O(p^2)$  ячеек памяти для хранения ее значений. Вычисление одного значения составляет  $O(p)$  арифметических операций. Обратный ход осуществляется за  $O(n)$  действий.

Следовательно, время работы алгоритма не превосходит  $O(np^3)$ , а необходимый объем памяти составляет не более  $O(np^2)$ . Теорема 2 доказана.

### 3. Свойство связности

Рассмотрим еще один полиномиально разрешимый случай задачи (1)–(3).

Пусть  $J' \subseteq J$ . Множество  $J'$  называется *связным*, если оно состоит из последовательных элементов, т. е. имеет вид  $J' = \{k, k+1, \dots, l\}$ .

**Теорема 3.** Если множества  $J_i, i \in I$ , являются связными и выполняется (4), то задача (1)–(3) решается за время  $O(nt \min\{n, t\})$ .

**ДОКАЗАТЕЛЬСТВО.** Пусть  $x' = (x'_1, \dots, x'_n)$  — произвольное оптимальное решение задачи (1)–(3). По лемме 1 можно считать, что  $x' \in X$ . Обозначим через  $S$  множество элементов  $s \in I_1$  таких, что

$$|J_s| = \min_{i \in I_1} |J_i|.$$

Выберем произвольный элемент  $l \in S$  и сформируем вектор  $x^* = (x_1^*, \dots, x_n^*)$ , полагая  $x_l^* = d_1, x_s^* — произвольное число, не меньшее  $d_1$ ,$

если  $s \in S \setminus \{l\}$ , и  $x_i^* = x'_i$ , если  $i \in I \setminus S$ . Покажем, что  $x^*$  — оптимальное решение задачи (1)–(3).

В силу ограничений (2) найдется элемент  $p \in I_1$  такой, что  $x'_p = d_1$ . Из связности множеств  $J_l$ ,  $J_p$  и неравенства  $|J_l| \leq |J_p|$  следует, что  $J_l \subseteq J_p$ . Поэтому  $\varphi_j(x^*) = \varphi_j(x') = d_1$ , если  $j \in J_l$ , и заведомо  $\varphi_j(x^*) = \varphi_j(x') \leq d_j$ , если  $j \in J \setminus J_l$ . Следовательно, вектор  $x^*$  удовлетворяет ограничениям (2) и (3), справедливо равенство  $\Phi(x^*) = \Phi(x') = \Phi^*$  и  $x^*$  — оптимальное решение.

Доказанный факт позволяет перейти от задачи (1)–(3) к задаче того же вида с множеством предприятий  $I' = I \setminus S$  и множеством потребителей  $J' = J \setminus J_l$ . В редуцированной задаче множества  $J'_i = J' \cap J_i$ ,  $i \in I'$ , являются связными и выполняется (4). Поэтому к ней можно применить тот же прием и продолжить процесс последовательного нахождения компонент оптимального решения.

Изложенное выше является обоснованием следующего алгоритма расчета оптимального решения  $x^* = (x_1^*, \dots, x_n^*)$  задачи (1)–(3).

**Шаг 1.** Положить  $J' = J$  и  $x_i^* = d_m$ ,  $i \in I$ .

**Шаг 2.** В множестве  $J'$  выбрать минимальный элемент  $k$  и найти элемент  $l \in I_k$  из условия

$$|J' \cap J_l| = \min_{i \in I_k} |J' \cap J_i|.$$

**Шаг 3.** Положить  $x_l^* = d_k$ ,  $J' = J' \setminus J_l$ .

**Шаг 4.** Если  $J' = \emptyset$ , то алгоритм заканчивает работу. В противном случае следует перейти к шагу 2.

**Конец**

Время работы алгоритма определяется в основном шагом 2. Однократное выполнение шага 2 требует  $O(nm)$  арифметических операций, длительности которых принимаются равными 1. Число обращений к шагу 2 не более  $\min\{n, m\}$ . Поэтому в целом время работы алгоритма не превосходит  $O(nm \min\{n, m\})$ . Теорема 3 доказана.

#### 4. Приближенное решение и верхняя оценка оптимума

Рассмотрим задачу (1)–(3) в общем случае. Теорема 1 оставляет мало надежды на полиномиальную разрешимость задачи. Ниже описываются алгоритм нахождения приближенного решения и способ вычисления верхней оценки оптимума  $\Phi^*$ .

Предлагаемый алгоритм строит приближенное решение  $x' = (x'_1, \dots, x'_n)$  по следующим шагам.

**Шаг 1.** Положить  $J' = J$  и  $x'_i = d_m$ ,  $i \in I$ .



**Шаг 2.** В множестве  $J'$  выбрать минимальный элемент  $k$  и найти элемент  $l \in I_k$  из условия

$$l = \arg \max_{i \in I_k} \left\{ d_k \sum_{j \in J' \cap J_i} b_j + \sum_{j \in J' \setminus J_i} b_j d_j \right\}.$$

**Шаг 3.** Положить  $x'_l = d_k$  и  $J' = J' \setminus J_l$ .

**Шаг 4.** Если  $J' = \emptyset$ , то алгоритм заканчивает работу. В противном случае следует вернуться к шагу 2.

**Конец**

Описанный алгоритм аналогичен алгоритму, приведенному в доказательстве теоремы 3, и имеет ту же временную сложность  $O(nm \min\{n, m\})$ .

Перейдем к вычислению верхней оценки величины  $\Phi^*$ . Пусть  $u, v$  — целые числа и  $0 \leq u < v \leq m$ . Положим  $J(u, v) = \{u + 1, u + 2, \dots, v\}$ , введем вспомогательную функцию

$$h(u, v) = \max_{i \in I_{u+1}} \left\{ d_{u+1} \sum_{j \in J(u, v) \cap J_i} b_j + \sum_{j \in J(u, v) \setminus J_i} b_j d_j \right\}$$

и обозначим через  $H^*$  оптимум следующей задачи о «ближайшем соседе»: найти минимум суммы

$$\sum_{k=1}^p h(z_{k-1}, z_k)$$

при ограничениях

$$z_0 = 0 < z_1 < \dots < z_p = m, \quad p = 1, \dots, m. \quad (16)$$

Покажем, что  $H^*$  есть верхняя оценка величины  $\Phi^*$ .

Учитывая условие (4), нетрудно проверить, что при любом  $x \in X$  справедливо неравенство

$$\sum_{j \in J(u, v)} b_j \varphi_j(x) \leq h(u, v).$$

Поэтому для любого вектора  $z = (z_1, \dots, z_p)$ , удовлетворяющего (16), получаем

$$\Phi(x) = \sum_{k=1}^p \sum_{j \in J(z_{k-1}, z_k)} b_j \varphi_j(x) \leq \sum_{k=1}^p h(z_{k-1}, z_k).$$

Следовательно, в силу произвольности выбора  $x$  и  $z$  имеем  $\Phi^* \leq H^*$ .

Величина  $H^*$  вычисляется за время  $O(m^2)$  алгоритмом динамического программирования [1].

**ЛИТЕРАТУРА**

1. **Береснев В. Л., Гимади Э. Х., Дементьев В. Т.** Экстремальные задачи стандартизации. Новосибирск: Наука, 1978.
2. **Горбачевская Л. Е.** Алгоритмы и сложность решения двухуровневых задач стандартизации с коррекцией дохода // Дискрет. анализ и исслед. операций. Сер. 2. 1998. Т. 5, № 2. С. 20–33.
3. **Гэри М., Джонсон Д.** Вычислительные машины и труднорешаемые задачи. М.: Мир, 1982.
4. **Ларин Р. М., Пяткин А. В.** Двухуровневая биматричная игра с регуляризкой выигрыша // Дискрет. анализ и исслед. операций. Сер. 2. 2000. Т. 7, № 2. С. 54–59.

Адрес авторов:

Институт математики  
им. С. Л. Соболева СО РАН,  
пр. Академика Коптюга, 4,  
630090 Новосибирск, Россия.  
E-mail: orlab@math.nsc.ru

Статья поступила  
18 июня 2002 г.