ДИСКРЕТНЫЙ АНАЛИЗ И ИССЛЕДОВАНИЕ ОПЕРАЦИЙ
Январь—июнь 2000. Серия 2. Том 7, № 1, 3–8

УДК 519.87+519.854.33

ЗАДАЧА РАЗМЕЩЕНИЯ ПУНКТОВ ПРОИЗВОДСТВА НА ДВА-ДЕРЕВЕ С ОГРАНИЧЕННЫМИ ПРОПУСКНЫМИ СПОСОБНОСТЯМИ КОММУНИКАЦИЙ*)

И. П. Вознюк

Рассмотрена задача о наилучшем размещении пунктов производства в вершинах сети с ограниченными пропускными способностями коммуникаций. Показано, что если сеть является два-деревом, то задача решается методом динамического программирования за время $O(nb^4)$ при объеме памяти $O(nb^2)$, где n — число вершин сети, b — суммарный объем спроса.

Введение

В данной статье продолжается начатое в [3] изучение следующей задачи. Задана сеть коммуникаций, связывающая пункты потребления некоторого продукта и возможные места его производства. Известны пропускные способности коммуникаций сети и объемы спроса потребителей продукта. Требуется разместить пункты производства таким образом, чтобы суммарные затраты на открытие предприятий и транспортные расходы по доставке продукта от производителей к потребителям были минимальны.

Задача с неограниченными пропускными способностями коммуникаций на сетях специального вида рассмотрена в ряде публикаций. Для случае, когда сетью является дерево, в работах [5, 7] найден полиномиальный алгоритм временной сложности $O(n^3)$, где n — число пунктов потребления. В [4, 6] предложены алгоритмы для решения этой же задачи на дереве с временной сложностью $O(n^2)$ и $O(nm)$ соответственно, где m — число возможных пунктов размещения производства. Задача на внешнепланарном графе решается за время $O(n^3m)$ [1]. В работе [8] представлен алгоритм для задачи на параллельно-последовательной

*) Работа выполнена при финансовой поддержке Федеральной целевой программы «Интеграция» (код проекта 274).

© 2000 Вознюк И. П.

сети с временной сложностью $O(n^4)$. Позже [2] для более общего случая задачи, а именно для два-древесной сети, был построен алгоритм с временной сложностью $O(nm^3)$.

В работе [3] задача с ограниченными пропускными возможностями коммуникаций на сети в виде дерева решена методом динамического программирования с временной сложностью $O(nb^2)$, где b — суммарный объем спроса. В настоящей статье этот же метод применяется в более общей ситуации, когда сеть является два-деревом. В этом случае задача решается за время $O(nb^4)$.

1. Постановка задачи

Рассмотрим сеть коммуникаций, представленную графом $G = (N, E)$. Вершины из множества $N = \{1, \dots, n\}$ обозначают пункты потребления некоторого продукта и возможные места его производства. Ребра из E соответствуют коммуникациям сети, по которым происходит доставка продукта от производителей к потребителям. Известны объем спроса b_i и затраты g_i^0 на размещение производства в пункте $i \in N$. Каждая коммуникация $e \in E$ характеризуется стоимостью c_e транспортировки по ней единицы продукта и максимальным количеством a_e продукта, который можно перевезти по этой коммуникации.

Требуется указать те пункты производства из N, при открытии которых удовлетворяется спрос на производимый продукт и суммарные затраты на открытие предприятий и доставку продукта минимальны.

Введем вспомогательные обозначения и переменные величины:

P_{ij} — множество всех простых путей, ведущих в графе G из вершины i в вершину j;

Q_e — множество таких троек (i, j, p), что путь p проходит через ребро e и $p \in P_{ij}$;

$g_{ij}^p = \sum\limits_{e \in p} c_e$ — стоимость транспортировки единицы продукта по пути $p \in P_{ij}$;

x_i — переменная выбора, равная единице, если в пункте $i \in N$ открывается предприятие, и равная нулю в противном случае;

x_{ij}^p — количество продукта, поставляемого из пункта производства i в пункт потребления j по пути $p \in P_{ij}$.

В принятых обозначениях задача формулируется следующим образом: найти минимум суммарных затрат

$$\sum_{i \in N} g_i^0 x_i + \sum_{i,j \in N} \sum_{p \in P_{ij}} g_{ij}^p x_{ij}^p \tag{1}$$

по всем переменным x_i, x_{ij}^p при условиях

$$\sum_{i\in N}\sum_{p\in P_{ij}} x_{ij}^p = b_j, \quad j\in N, \tag{2}$$

$$\sum_{(i,j,p)\in Q_e} x_{ij}^p \leqslant a_e, \quad e\in E, \tag{3}$$

$$0\leqslant x_{ij}^p \leqslant b_j x_i, \quad p\in P_{ij}, \quad i,j\in N, \tag{4}$$

$$x_i\in\{0,1\}, \quad i\in N. \tag{5}$$

Равенство (2) обеспечивает удовлетворение спроса в пункте $j\in N$. Соотношение (3) учитывает ограничение на пропускную способность ребра $e\in E$. Неравенства (4) не позволяют поставлять продукт из пункта i, если в нем не размещено производство. Постановку (1)–(5) будем кратко называть *задачей размещения.*

Числа a_e, b_i считаем неотрицательными и целыми. Суммарный объем спроса обозначим через $b=\sum_{i\in N} b_i$. Отметим используемое ниже легко доказываемое утверждение.

Лемма. *Существует такое оптимальное решение задачи* (1)–(5), *что*

1) *каждый пункт производства не является транзитной вершиной для потоков продукта из других предприятий;*

2) *все потоки продукта, идущие через данное ребро, имеют одно и то же направление.*

2. Задача размещения на два-дереве

Класс два-деревьев определяется индуктивно следующим образом:

а) полный граф на трех вершинах 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.

Теорема. *Задача размещения на два-дереве решается методом динамического программирования за время $O(nb^4)$ при объеме памяти $O(nb^2)$.*

Доказательство. Рассмотрим ребро $e=\{i,j\}$ исходной сети. Через y_{ij} обозначим суммарный поток продукта, идущего по ребру e. В силу леммы достаточно учитывать потоки, имеющие одно и то же направление. Условимся считать $y_{ij}>0$, если поток продукта направлен от вершины j к вершине i, и $y_{ij}\leqslant 0$, если поток идет в обратном направлении. Можно анализировать только целые значения величины y_{ij}, поскольку пропускные способности коммуникаций и объемы спроса — целые числа.

Через $[-b, b]$ обозначим целочисленный сегмент и через $B = \{(u, v) \mid u, v \in [-b, b]\}$ — соответствующий целочисленный квадрат. Отметим, что величина любого допустимого потока, проходящего через данную вершину по всем инцидентным с ней ребрам, лежит в $[-b, b]$.

Поясним основную идею применения метода динамического программирования к обсуждаемой задаче размещения. Поскольку граф G является два-деревом, в нем есть вершины степени 2. Пусть i — вершина степени 2, смежная с вершинами j и k. Если фиксировать величины u и v, равные суммам потоков

$$u = y_{ji} + y_{jk}, \quad v = y_{ki} + y_{kj},$$

то фрагмент, состоящий из вершины i и ребер $\{i, j\}$, $\{i, k\}$, $\{j, k\}$, «изолируется» от остальной сети и может быть рассмотрен отдельно. Минимум суммарных затрат, связанных с выделенным фрагментом при данных $(u, v) \in B$, обозначим через $S^i_{jk}(u, v)$. Из графа G удалим вершину i и ребра $\{i, j\}$, $\{i, k\}$. Ребру $\{j, k\}$ поставим в соответствие функцию S^i_{jk}.

В результате этой операции, которую назовем *свертыванием треугольника*, получаем два-дерево с меньшим числом вершин. Применяя к нему операцию свертывания треугольника и продолжая этот процесс, приходим к ситуации, когда в графе остаются только две вершины. Расчет минимума суммарных затрат, соответствующих этому случаю, завершает прямой ход динамического программирования.

Перейдем к более подробному изложению. Для единообразия записи каждому ребру $e = \{i, j\} \in E$ исходного графа $G = (N, E)$ вначале поставим в соответствие функцию $S^0_{ij}(y_{ij}, y_{ji})$, $(y_{ij}, y_{ji}) \in B$, полагая

$$S^0_{ij}(y_{ij}, y_{ji}) = c_e |y_{ij}|,$$

если совместны условия

$$y_{ij} + y_{ji} = 0, \quad |y_{ij}| \leqslant a_e,$$

и $S^0_{ij}(y_{ij}, y_{ji}) = \infty$, если эти условия несовместны. Величина $S^0_{ij}(y_{ij}, y_{ji})$ равна транспортным расходам на перемещение продукта по ребру $\{i, j\}$. Ее можно трактовать как результат операции свертывания треугольника с фиктивной нулевой вершиной.

Рассмотрим общий шаг прямого хода динамического программирования. В текущем графе выберем произвольную вершину i степени 2, смежную с вершинами j и k. Пусть ребрам $\{i, j\}$, $\{i, k\}$ и $\{j, k\}$ соответствуют рассчитанные на предыдущих шагах функции S^α_{ij}, S^β_{ik} и S^γ_{jk}, где $\alpha, \beta, \gamma \in N \cup \{0\}$ — номера вершин ранее свернутых треугольников. Через $S^i_{jk}(Y_{jk}, Y_{kj})$ обозначим минимум суммарных затрат, связанных с выделенным треугольным фрагментом, при фиксированных потоках

$(Y_{jk}, Y_{kj}) \in B$ продукта, идущего через вершины j и k по инцидентным с ними ребрам всех ранее свернутых треугольников.

С целью экономии вычислений расчет функции S^i_{jk} проводится в два этапа. Сначала вычисляется вспомогательная функция $R^i_{jk}(z_{ji}, z_{ki})$, $(z_{ji}, z_{ki}) \in B$, по формуле

$$R^i_{jk}(z_{ji}, z_{ki}) = \min_{x_i, w_{ij}, w_{ik}} \{g^0_i x_i + S^\alpha_{ij}(w_{ij}, z_{ji}) + S^\beta_{ik}(w_{ik}, z_{ki})\}, \qquad (6)$$

где минимум берется при ограничениях

$$w_{ij} + w_{ik} = b_i, \text{ если } x_i = 0; \quad (w_{ij}, w_{ik}) \in B, \quad x_i \in \{0, 1\}.$$

Величина $R^i_{jk}(z_{ji}, z_{ki})$ равна минимуму суммарных затрат, связанных с вершиной i и ребрами $\{i, j\}$, $\{i, k\}$. После расчета функции R^i_{jk} получаем

$$S^i_{jk}(Y_{jk}, Y_{kj}) = \min_{(z_{ji}, z_{ki}) \in B} \{R^i_{jk}(z_{ji}, z_{ki}) + S^\gamma_{jk}(Y_{jk} - z_{ji}, Y_{kj} - z_{ki})\}. \qquad (7)$$

После расчета функции S^i_{jk} из текущего графа удаляются вершина i и ребра $\{i, j\}$, $\{i, k\}$. Ребру $\{j, k\}$ ставится в соответствие новая функция S^i_{jk}, что и завершает операцию свертывания треугольника.

Рассмотрим заключительный шаг прямого хода, когда граф состоит из вершин i, j и ребра $\{i, j\}$, которому соответствует функция S^α_{ij}. Прямой ход заканчивается расчетом величины S^*, равной экстремуму в исходной задаче:

$$S^* = \min_{x_i, x_j, Y_{ij}, Y_{ji}} \{g^0_i x_i + g^0_j x_j + S^\alpha_{ij}(Y_{ij}, Y_{ji})\}. \qquad (8)$$

Минимум ищется при ограничениях

$$Y_{ij} = b_i, \text{ если } x_i = 0; \ Y_{ji} = b_j, \text{ если } x_j = 0; \ (Y_{ij}, Y_{ji}) \in B, \ x_i, x_j \in \{0, 1\}.$$

В процессе прямого хода наряду с функциями R^i_{jk} и S^i_{jk} запоминаются *условно-оптимальные решения*, т. е. те значения переменных, на которых достигаются экстремумы в (6)–(8). Решение исходной задачи 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 на n вершинах имеет $2n - 3$ ребер. Поэтому число всех функций, рассчитываемых в процессе прямого хода, не превосходит $O(n)$. Каждая функция требует $O(b^2)$ ячеек памяти для хранения ее значений. Вычисление одного значения функции требует $O(b^2)$ арифметических операций. Обратный ход осуществляется за $O(n)$ действий.

Следовательно, время работы алгоритма не превосходит $O(nb^4)$, а необходимый объем памяти составляет не более $O(nb^2)$. Теорема доказана.

Теорема остается верной, если в задаче размещения граф G является подграфом два-дерева. Таковы, в частности, параллельно-последовательные графы [2, 8]. Поэтому задача размещения на параллельно-последовательной сети решается с теми же оценками сложности, что и в доказанной теореме.

В заключение выражаю признательность моему научному руководителю Э. Х. Гимади и Ю. В. Шамардину за помощь при написании статьи.

ЛИТЕРАТУРА

1. **Агеев А. А.** Графы, матрицы и простейшая задача размещения // Управляемые системы: Сб. науч. тр. Новосибирск: Ин-т математики СО АН СССР, 1989. Вып. 29. С. 3–11.
2. **Агеев А. А.** Полиномиальный алгоритм решения задачи размещения на последовательно-параллельной сети // Управляемые системы: Сб. науч. тр. Новосибирск: Ин-т математики СО АН СССР, 1990. Вып. 30. С. 3–16.
3. **Вознюк И. П.** Задача размещения на сети с ограниченными пропускными способностями коммуникаций // Дискрет. анализ и исслед. операций. Сер. 2. 1999. Т. 6, № 1. С. 3–11.
4. **Гимади Э. Х.** Эффективный алгоритм решения задачи размещения с областями обслуживания, связными относительно ациклической сети // Управляемые системы: Сб. науч. тр. Новосибирск: Ин-т математики СО АН СССР, 1989. Вып. 23. С. 12–23.
5. **Трубин В. А.** Эффективный алгоритм решения задачи размещения на сети в форме дерева // Докл. АН СССР. 1976. Т. 231, № 3. С. 547–550.
6. **Billionet A., Costa M.-C.** Solving the uncapacitated plant location problem on trees // Discrete Applied Math. 1994. V. 49, N 1–3. P. 51–59.
7. **Kolen A.** Solving covering problems and the uncapacitated plant location on the trees // European J. Oper. Res. 1983. V. 12, N 3. P. 266–278.
8. **Hassin R., Tamir A.** Efficient algorithm for optimization and selection on series-parallel graphs // SIAM J. Algebraic Discrete Methods. 1986. V. 7, N 3. P. 379–389.

Адрес автора:

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

Статья поступила
18 февраля 1999 г.,
переработанный вариант —
9 февраля 2000 г.