ВикипедиЯ

Венгерский алгоритм

Материал из Википедии — свободной энциклопедии

Венгерский алгоритм — алгоритм оптимизации, решающий задачу о назначениях за полиномиальное время (см. исследование операций). Он был разработан и опубликован Харолдом Куном в 1955 году. Автор дал ему имя «венгерский метод» в связи с тем, что алгоритм в значительной степени основан на более ранних работах двух венгерских математиков (Кёнига и Эгервари).

Джеймс Манкрес в 1957 году заметил, что алгоритм является (строго) полиномиальным. С этого времени алгоритм известен также как **алгоритм Куна — Манкреса** или **алгоритм Манкреса решения задачи о назначениях**. Временная сложность оригинального алгоритма была $O(n^4)$, однако Эдмондс и Карп (а также Томидзава независимо от них) показали, что его можно модифицировать так, чтобы достичь времени выполнения $O(n^3)$. Форд и Фалкерсон распространили метод на общие транспортные задачи. В 2006 году было обнаружено, что Якоби нашёл решение задачи о назначениях в XIX веке и опубликовал его в 1890 году на латыни[1].

Содержание

Объяснение на примере

Предположим, есть три работника: Иван, Пётр и Андрей. Нужно распределить между ними выполнение трёх видов работ (которые мы назовём A, B, C), каждый работник должен выполнять только одну разновидность работ. Как это сделать так, чтобы потратить наименьшую сумму денег на оплату труда рабочих? Сначала необходимо построить матрицу стоимостей работ.

	A	B	C
Иван	10.000 руб.	20.000 руб.	30.000 руб.
Пётр	30.000 руб.	30.000 руб.	30.000 руб.
Андрей	30.000 руб.	30.000 руб.	20.000 руб.

Венгерский алгоритм, применённый к приведённой выше таблице, даст нам требуемое распределение: Иван выполняет работу A, Пётр — работу B, Андрей — работу C.

Постановка задачи

Дана неотрицательная матрица размера $n \times n$, где элемент в i-й строке и j-м столбце соответствует стоимости выполнения j-го вида работ i-м работником. Нужно найти такое соответствие работ работникам, чтобы расходы на оплату труда были наименьшими. Если цель состоит в нахождении назначения с наибольшей стоимостью, то решение сводится к решению только что сформулированной задачи путём замены каждой стоимости C на разность между максимальной стоимостью и C.[2]

Основные идеи

Алгоритм основан на двух идеях:

- если из всех элементов некой строки или столбца вычесть одно и то же число y, общая стоимость уменьшится на y, а оптимальное решение не изменится;
- если есть решение нулевой стоимости, оно оптимально.

Алгоритм ищет значения, которые надо вычесть из всех элементов каждой строки и каждого столбца (разные для разных строк и столбцов), такие, что все элементы матрицы останутся неотрицательными, но появится нулевое решение.

Определения

Алгоритм проще описать, если сформулировать задачу, используя двудольный граф. Дан полный двудольный граф $G=(S, T; E)$ с n вершинами, соответствующими работникам (S), и n вершинами, соответствующими видам работ (T); стоимость каждого ребра $c(i, j)$ неотрицательна. Требуется найти совершенное, или полное паросочетание с наименьшей стоимостью.

Будем называть функцию $y\colon (S \cup T) \to \mathbb{R}$ **потенциалом**, если $y(i) + y(j) \leq c(i,j)$ для каждых $i \in S, j \in T$. **Значение потенциала** y равно $\sum_{v \in S \cup T} y(v)$. Нетрудно заметить, что стоимость любого совершенного паросочетания не меньше, чем значение любого потенциала. Венгерский метод находит полное паросочетание и потенциал с одинаковой стоимостью/значением, что доказывает оптимальность обеих величин. Фактически он находит совершенное паросочетание **жёстких** рёбер: ребро ij называется жёстким для потенциала y, если $y(i) + y(j) = c(i,j)$. Подграф жёстких рёбер будем обозначать как $\mathbf{G_y}$. Стоимость полного паросочетания в G_y (если оно существует) равна значению y.

Алгоритм в терминах двудольных графов

Алгоритм хранит в памяти потенциал y и ориентацию (задание направления) каждого жёсткого ребра, обладающую тем свойством, что рёбра, направленные от T к S образуют паросочетание, которое мы обозначим M. Ориентированный граф, состоящий из жёстких рёбер с заданной ориентацией, мы обозначаем $\overrightarrow{\mathbf{G_y}}$. Таким образом, в любой момент есть три типа рёбер:

- нежёсткие (и не принадлежащие M)
- жёсткие, но не принадлежащие M
- жёсткие и принадлежащие M

Изначально y везде равно 0, и все рёбра направлены от S к T (таким образом, M пусто). На каждом шаге или модифицируется y так, что увеличивается множество вершин Z, определённое в следующем абзаце, или изменяется ориентация, чтобы получить паросочетание с большим количеством рёбер; всегда остаётся верным, что все рёбра из M являются жёсткими. Процесс заканчивается, если M — совершенное паросочетание.

Пусть на каждом шаге $R_S \subseteq S$ и $R_T \subseteq T$ составляют множество вершин, не инцидентных рёбрам из M (то есть R_S — множество вершин из S, в которые не входит ни одно ребро, а R_T — множество вершин из T, из которых не исходит ни одно ребро). Обозначим через Z множество вершин, достижимых из R_S в $\overrightarrow{G_y}$ (оно может быть найдено поиском в ширину).

Если $R_T \cap Z$ не является пустым, это значит, что есть хотя бы один путь в $\overrightarrow{G_y}$ из R_S в R_T. Выбираем любой из таких путей, и изменяем ориентацию всех его ребёр на обратную. Размер паросочетания увеличится на 1.

> Для доказательства отметим два очевидных факта:
>
> - На выбранном пути рёбра из S в T чередуются с рёбрами из T в S. Это следует из двудольности графа.
> - Первая вершина пути принадлежит S, а последняя — T. Следовательно, первое и последнее его рёбра направлены из S в T.
>
> По определению $\overrightarrow{G_y}$, отсюда следует, что на выбранном пути рёбра, принадлежащие и не принадлежащие M чередуются, причём первое и последнее рёбра не принадлежат M, то есть путь является повышающим, откуда и следует доказываемое утверждение.

Если $R_T \cap Z$ пусто, положим $\Delta := \min\{c(i,j) - y(i) - y(j) : i \in Z \cap S, j \in T \setminus Z\}$. Δ положительна, потому что нет жёстких рёбер между $Z \cap S$ и $T \setminus Z$.

> В самом деле, пускай такое ребро (i, j) есть. Поскольку $i \in Z$, но $j \notin Z$, вершина i достижима из R_S в $\overrightarrow{G_y}$, а вершина j недостижима. Следовательно, $\overrightarrow{G_y}$ не может содержать ребра (i, j). Следовательно, $(i,j) \in M$, откуда $i \notin R_S$. Поскольку i достижима из R_S в $\overrightarrow{G_y}$, существует путь в i из какой-то вершины, принадлежащей R_S. Рассмотрим последнее ребро этого пути. Обозначим его (k, i). Поскольку k достижима из R_S в $\overrightarrow{G_y}$, а j недостижима, $k \neq j$. Поскольку $(k,i) \in \overrightarrow{G_y}$, $(i,k) \in M$. Следовательно, i инцидентна сразу двум рёбрам из M: (i,j) и (i,k), что невозможно, так как M — паросочетание. Противоречие.

Увеличим y на Δ на вершинах из $Z \cap S$ и уменьшим y на Δ на вершинах, входящих в $Z \cap T$. Новый y остаётся потенциалом.

> В самом деле, для любого ребра (i, j), $i \in S, j \in T$:
>
> - если $i \notin Z$, $j \notin Z$, то c(i, j)-y(i)-y(j) не меняется, потому что не меняются ни y(i), ни y(j)
> - если $i \in Z$, $j \in Z$, то c(i, j)-y(i)-y(j) не меняется, потому что y(i) увеличивается на Δ, а y(j) на столько же уменьшается
> - если $i \notin Z$, $j \in Z$, разность c(i, j)-y(i)-y(j) увеличивается на Δ, следовательно, остаётся неотрицательной
> - если $i \in Z$, $j \notin Z$, разность c(i, j)-y(i)-y(j) уменьшается на Δ, но всё равно остаётся неотрицательной, потому что Δ — наименьшая из таких разностей.

Граф $\overrightarrow{G_y}$ меняется, но, несмотря на это, содержит M.

>

> В самом деле, чтобы исключить из $\overrightarrow{G_y}$ некое ребро (i, j), $i \in S, j \in T$, надо сделать его нежёстким, то есть повысить разность c(i, j)-y(i)-y(j). Как мы видели, разность повышается только если $i \notin Z$, $j \in Z$, иными словами, i недостижима из R_S, а j достижима. Но в таком случае ребро (j, i) не может принадлежать $\overrightarrow{G_y}$, следовательно, ребро не принадлежит M.

Ориентируем новые рёбра от S к T. По определению Δ, множество Z вершин, достижимых из R_S, увеличится (при этом число жёстких рёбер вовсе не обязательно возрастёт).

> Для доказательства этого утверждения сначала докажем, что ни одна вершина не пропадёт из Z. Пусть $V \in Z$. Тогда существует путь из некоей вершины, принадлежащей R_S, в V. Все вершины на этом пути достижимы из R_S, то есть принадлежат Z. Каждое рёбро на этом пути инцидентно двум вершинам из Z. Как мы видели выше, для таких рёбер разность c(i, j)-y(i)-y(j) не меняется. Значит, все рёбра пути останутся жёсткими, и V по-прежнему будет достижима из R_S. Теперь докажем, что хотя бы одна вершина добавится к Z. Рассмотрим ребро, на котором достигается минимум $\min\{c(i,j) - y(i) - y(j) : i \in Z \cap S, j \in T \setminus Z\}$. Для этого ребра, разность c(i, j)-y(i)-y(j) обнулится, следовательно, оно станет жёстким и будет направлено из S в T, то есть от i к j. Поскольку $i \in Z$, j также станет достижимым из R_S, то есть добавится к Z.

Повторяем эти шаги до тех пор, пока M не станет совершенным паросочетанием; в этом случае оно даёт назначение с наименьшей стоимостью. Время выполнения этой версии алгоритма равно $O(n^4)$: M дополняется n раз, а в стадии, когда M не меняется, может быть не более n изменений потенциала (так как Z увеличивается каждый раз). Время, необходимое на изменение потенциала, равно $O(n^2)$.

Матричная интерпретация

Для n работников и работ, дана матрица $n \times n$, задающая стоимость выполнения каждой работы каждым работником. Найти минимальную стоимость выполнения работ, такую что каждый работник выполняет ровно одну работу, а каждую работу выполняет ровно один работник.

В дальнейшем мы под **назначением** понимаем соответствие между работниками и работами, имеющее нулевую стоимость, после того как мы произвели трансформации, влияющие лишь на общую стоимость работ.

Прежде всего запишем задачу в матричной форме:

$$\begin{bmatrix} a1 & a2 & a3 & a4 \\ b1 & b2 & b3 & b4 \\ c1 & c2 & c3 & c4 \\ d1 & d2 & d3 & d4 \end{bmatrix}$$

где a, b, c, d — работники, которые должны выполнить работы 1, 2, 3, 4. Коэффициенты a1, a2, a3, a4 обозначают стоимость выполнения работником «a» работ 1, 2, 3, 4 соответственно. Аналогичный смысл имеют остальные символы. Матрица квадратная, поэтому каждый работник может выполнить только одну работу.

Шаг 1

Уменьшаем элементы построчно. Находим наименьший из элементов первой строки (a1, a2, a3, a4), и вычитаем его из всех элементов первой строки. При этом хотя бы один из элементов первой строки обнулится. То же самое выполняем и для всех остальных строк. Теперь в каждой строке матрицы есть хотя бы один ноль.

Иногда нулей уже достаточно, чтобы найти назначение. Пример показан в таблице. Красные нули обозначают назначенные работы.

0	a2'	0	a4'
b1'	b2'	b3'	0
0	c2'	c3'	c4'
d1'	0	d3'	d4'

При большом количестве нулей для поиска назначения (нулевой стоимости) можно использовать алгоритм нахождения максимального паросочетания двудольных графов, например алгоритм Хопкрофта — Карпа. Кроме того, если хотя бы в одном столбце нет нулевых элементов, то назначение невозможно.

Шаг 2

Часто на первом шаге нет назначения, как, например, в следующем случае:

0	a2'	a3'	a4'
b1'	b2'	b3'	0
0	c2'	c3'	c4'
d1'	0	d3'	d4'

Задача 1 может быть эффективно (за нулевую стоимость) выполнена как работником a, так и работником c, зато задача 3 не может быть эффективно выполнена никем.

В таких случаях мы повторяем шаг 1 для столбцов и вновь проверяем, возможно ли назначение.

Шаг 3

Во многих случаях мы достигнем желаемого результата уже после шага 2. Но иногда это не так, например:

0	a2'	a3'	a4'
b1'	b2'	b3'	0
0	c2'	c3'	c4'
d1'	0	0	d4'

Если работник d выполняет работу 2, некому выполнять работу 3, и наоборот.

В таких случаях мы выполняем процедуру, описанную ниже.

Сначала, используя любой алгоритм поиска максимального паросочетания в двудольном графе, назначаем как можно больше работ тем работникам, которые могут их выполнить за нулевую стоимость. Пример показан в таблице, назначенные работы выделены красным.

0	a2'	a3'	a4'
b1'	b2'	b3'	0
0	c2'	c3'	c4'
d1'	0	0	d4'

Отметим все строки без назначений (строка 1). Отметим все столбцы с нулями в этих строках (столбец 1). Отметим все строки с «красными» нулями в этих столбцах (строка 3). Продолжаем, пока новые строки и столбцы не перестали отмечаться.

×				
0	a2'	a3'	a4'	×
b1'	b2'	b3'	0	
0	c2'	c3'	c4'	×
d1'	0	0	d4'	

Теперь проводим линии через все отмеченные столбцы и неотмеченные строки.

×				
0	a2'	a3'	a4'	×
b1'	b2'	b3'	0	
0	c2'	c3'	c4'	×
d1'	0	0	d4'	

Все эти действия преследовали лишь одну цель: провести наименьшее количество линий (вертикалей и горизонталей) так, чтобы покрыть все нули. Можно было воспользоваться любым другим методом вместо описанного.

Шаг 4

Из непокрытых линиями элементов матрицы (в данном случае это a2', a3', a4', c2', c3', c4') найти наименьший. Вычесть его из всех не отмеченных строк и прибавить ко всем пересечениям отмеченных строк и столбцов. Так, например, если наименьший элемент из перечисленных равен a2', мы получим

×				
0	0	a3'-a2'	a4'-a2'	×
b1'+a2'	b2'	b3'	0	
0	c2'-a2'	c3'-a2'	c4'-a2'	×
d1'+a2'	0	0	d4'	

Повторять процедуру (шаги 1-4) до тех пор, пока назначение не станет возможным.

Библиография

- R.E. Burkard, M. Dell'Amico, S. Martello: *Assignment Problems*. SIAM, Philadelphia (PA.) 2009. ISBN 978-0-89871-663-4
- Harold W. Kuhn, «The Hungarian Method for the assignment problem», *Naval Research Logistics Quarterly*, **2**:83—97, 1955. Kuhn's original publication.
- Harold W. Kuhn, «Variants of the Hungarian method for assignment problems», *Naval Research Logistics Quarterly*, **3**: 253—258, 1956.
- J. Munkres, «Algorithms for the Assignment and Transportation Problems», *Journal of the Society for Industrial and Applied Mathematics*, **5**(1):32—38, 1957 March.
- M. Fischetti, «Lezioni di Ricerca Operativa», Edizioni Libreria Progetto Padova, Italia, 1995.
- R. Ahuja, T. Magnanti, J. Orlin, «Network Flows», Prentice Hall, 1993.

Примечания

1. (недоступная ссылка)politechnique.fr (http://www.lix.polytechnique.fr/~ollivier/JACOBI/jacobiEngl.htm)
2. Архивированная копия (http://www.ams.jhu.edu/~castello/362/Handouts/hungarian.pdf) (недоступная ссылка — *история* (https://web.archive.org/web/*/http://www.ams.jhu.edu/~castello/362/Handouts/hungarian.pdf)). Проверено 11 февраля

2010. Архивировано (https://web.archive.org/web/20110719220036/http://www.ams.jhu.edu/~castello/362/Handouts/hungarian.pdf) 19 июля 2011 года.

Ссылки