

Задача об оптимальном расписании (Scheduling Problem)

Зотов Алексей, 497

28 ноября 2016 г.

Формулировка задачи

Имеется множество работ J и множество машин M . Также задана функция $p : J \times M \rightarrow \mathbb{R}_+$. Значение $p(i, j) = p_{ij}$ означает время выполнения i -ой работы на j -ой машине.

Требуется найти распределение работ по машинам, так чтобы время выполнения всех работ было минимально. Формально, требуется построить функцию $x : J \times M \rightarrow \{1, 0\}$ такую, что:

$$\sum_{j \in M} x_{ij} = 1, \quad \forall i \quad (1)$$

$$\max_{j \in M} \sum_i x_{ij} p_{ij} \rightarrow \min \quad (2)$$

NP - полнота

Теорема. Задача об оптимальном расписании является **NP** - полной.
Здесь рассматривается измененный вариант задачи:

$$\max_{j \in M} \sum_i x_{ij} p_{ij} \leq k \quad (3)$$

Доказательство.

1. SCHEDULING \in NP

Действительно, сертификатом будет являться значения функции x на множестве J . Полиномиально вычисляется искомый функционал.

2. Рассмотрим задачу SUBSETSUM.

Дано множество A , определена весовая функция $s : A \rightarrow \mathbb{N}$ на элементах множества. Необходимо найти подмножество $A' \subseteq A$, такое что $\sum_{a \in A'} s(a) = \sum_{a \in A \setminus A'} s(a)$.

Легко свести данную задачу к задаче о расписании, взяв $J = A$ - множество элементов как множество работ, $M = \{1, 2\}$ - два подмножества в задаче о разбиении как две машины, $p_{ij} = s(a_i)$ - вес элемента как сложность выполнения работы и $k = 0$, так как нам нужно точное разбиение. Тогда, решив задачу о расписании, мы получим решение задачи о разбиении.

Покажем теперь, что **SUBSETSUM NP** - трудна, что завершит доказательство теоремы.

Утверждение. SUBSETSUM \in NPH.

Доказательство утверждения. Сведем **NP** полную задачу о покрытии ребрами 3-дольного 3-однородного гиперграфа (**3-dimensional matching**, **3DM**) к задаче **SUBSETSUM**.

Пусть $W = \{w_1, \dots, w_q\}$, $X = \{x_1, \dots, x_q\}$, $Y = \{y_1, \dots, y_q\}$, $V = W \cup X \cup Y$ - вершины и $M \subseteq W \times X \times Y$, $|M| = k$ - ребра гиперграфа из задачи **3DM**.

Будем считать, что $\forall v \in V \exists e \in M : v \in e$, то есть для каждой вершины существует ребро, которое

ее содержит. Если это не так, то ответ на задачу отрицательный, и это можно вычислить за полиномиальное время.

Требуется построить множество A и размеры $s(a)$ всех его элементов так, чтобы было выполнено:

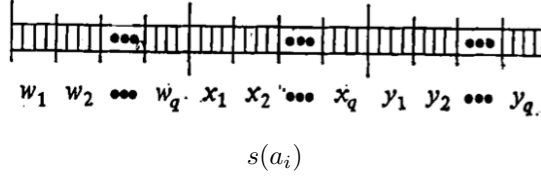
$$\exists A' \in A \sum_{a \in A'} s(a) = \sum_{a \in A' \setminus A} s(a) \iff \exists J \subseteq \{1 \dots k\}, m_j \in M, j \in J : \bigsqcup_J m_j = V \quad (4)$$

то есть, что в A можно найти подмножество, равное по весу своему дополнению, тогда и только тогда, когда M содержит искомое покрытие.

Множество A будет состоять из $k + 2$ элементов. Первые k элементов множества A будут $\{a_i : 1 \leq i \leq k\}$, вес элемента $s(a_i)$ будет определяться по m_i . Построим двоичную запись числа $s(a_i)$ по $m_i = (w_{f(i)}, x_{g(i)}, y_{h(i)})$. Для записи $s(a_i)$ будем использовать ровно $3qp$ бит, где $p = \lceil \log_2(k+1) \rceil$. Каждой из трех компонент m_i соответствует один подотрезок длины qp в двоичной записи числа $s(a_i)$, который состоит из q блоков длины p , каждый блок соответствует своему $1 \leq i \leq q$. В числе $s(a_i)$ правые концы зон, соответствующих $w_{f(i)}, x_{g(i)}, y_{h(i)}$ равны 1, остальные биты равны 0:

$$s(a_i) = 2^{p(3q-f(i))} + 2^{p(2q-g(i))} + 2^{p(q-h(i))} \quad (5)$$

$s(a_i)$ имеет длину не больше $3pq$, и может быть построено за полиномиальное время.



Заметим, что если просуммировать содержимое одной зоны всех элементов множества $\{a_i : 1 \leq i \leq k\}$, то результат не будет превосходить $2^p - 1$. Следовательно, при суммировании по любому подмножеству, никогда не придется переносить единицы из одной зоны (p бит) в соседнюю.

Положим :

$$B = \sum_{j=0}^{3q-1} 2^{pj} \quad (6)$$

B - число, в двоичной записи которого в правом конце каждой зоны стоит 1. Тогда для любого подмножества $A' \subseteq \{a_i : 1 \leq i \leq k\}$, соотношение:

$$\sum_{a \in A'} s(a) = B \quad (7)$$

выполняется тогда и только тогда $M' = \{m_i : a_i \in A'\}$ - решение **3DM**. Последние два элемента b_1 и b_2 множества A такие, что выполнено:

$$s(b_1) = 2 \left(\sum_{i=1}^k s(a_i) \right) - B \quad (8)$$

$$s(b_2) = \left(\sum_{i=1}^k s(a_i) \right) + B \quad (9)$$

Двоичные записи $s(b_1)$ и $s(b_2)$ имеют длину не более $3pq + 1$ и могут быть построены за полиномиальное время.

Предположим, что имеется подмножество $A' \subseteq A$, такое, что:

$$\sum_{a \in A'} s(a) = \sum_{a \in A' \setminus A} s(a) \quad (10)$$

Тогда каждая из этих сумм должна быть равна $\frac{1}{2} \sum_{a \in A} s(a) = 2 \sum_{i=1}^k s(a_i)$. Также $s(b_1) + s(b_2) = 3 \left(\sum_{i=1}^k s(a_i) \right)$, значит одно из множеств A' или $A' \setminus A$ содержит b_1 и не содержит b_2 . Значит остальные

элементы этого подмножества из $\{a_i : 1 \leq i \leq k\}$ и сумма их весов в точности равна B . Тогда, по сделанному ранее замечанию, этому подмножеству соответствует $M' \subseteq M$ являющееся решением задачи **3DM**.

Обратно, если задано покрытие $M' \subseteq M$, являющееся решением задачи **3DM**, то $b_1 \cup a_i : m_i \in M'$ - искомое множество A' .

Утверждение доказано, и тем самым доказана теорема об **NP** - полноте задачи о расписании.

Полиномиальное приближение

Найдем полиномиальное приближенное решение для частного случая задачи об оптимальном расписании, когда мощности всех машин одинаковы, то есть $p_{ij} = p_i$.

Рассмотрим алгоритм **LPTR** (Longest Processing Time Rule):

1. Отсортируем работы в порядке невозрастания сложности работы p_i , то есть так, что $p_i \geq p_{i+1} \quad \forall i$.
2. Пройдем все работы в данном порядке, назначая на i -м шаге данную работу той машине, которая завершит обработку уже назначенных ей задач раньше всех остальных машин.

Теорема. LPTR - имеет коэффициент аппроксимации $\frac{4}{3}$ для задачи об оптимальном расписании, то есть $\text{LPTR}(x) \leq \frac{4}{3} \text{OPT}(x) \quad \forall x$ - входные данные, $\text{OPT}(x)$ - оптимальное решение.

◁ Пусть S - расписание, результат работы **LPTR** алгоритма, $w(S)$ - время окончания работ. Пусть l - работа, которая завершится последней. Можно считать, что l - это номер последней работы. Если это не так, то докажем для $J' = \{1 \dots l\}$. Тогда для новой и исходной задачи алгоритм выдаст расписание с одним и тем же временем окончания работы, то есть $w(S') = w(S)$, в то время как оптимальное время в исходной задаче может быть только больше либо равно оптимальному времени в новой задаче ($\text{OPT} \geq \text{OPT}'$). Значит, доказав аппроксимацию для работы на J' , автоматически докажем и для работы на J .

Пусть t_l - время начала выполнения работы l . Тогда в этот момент времени все машины заняты (одна или несколько могли освободиться в данный момент), иначе если какая-то машина была свободна, то t_l было бы меньше (в момент времени $t = 0$ все машины считаются занятыми). Значит все время t_l все m машин работали непрерывно:

$$t_l \leq (\sum_{j \in M} p_j - p_l)/m \leq \text{OPT} - \frac{p_l}{m} \quad (11)$$

$$w(S) = t_l + p_l \leq \text{OPT} + p_l(1 - \frac{1}{m}) \quad (12)$$

Лемма. Если $p_{\min} > \frac{\text{OPT}}{3}$, то $w(S) = \text{OPT}$.

Доказательство леммы. В условиях леммы в оптимальном расписании будет не более двух задач на одной машине. Покажем что LPTR алгоритм также не назначит более двух задач на одну машину. Всего работ не более чем $2m$. Предположим в S на какой-то машине не менее 3х работ. Пусть k - первая работа которая стала 3-ей на какой-то машине. Тогда есть и работа j , которая единственна на своей машине в S и не единственна в оптимальном. Заметим, что в LPTR j будет рассмотрена раньше k , иначе k была бы распределена на свободную машину. Тогда $p_j < 2\text{OPT}/3$. С другой стороны, $p_j > 2p_{\min} > 2\text{OPT}/3$, иначе работа k могла бы быть распределена на машину к работе j . Значит LPTR назначает не более двух работ на машину. Можно найти такое оптимальное расписание S_2 , что $a_i \geq a_j \quad i \leq j$ и $b_i \leq b_j \quad i \leq j$, где a_i и b_i - время обработки первой и второй работ соответственно на i -ой машине. Именно такое расписание и выдаст LPTR алгоритм.

Завершим доказательство теоремы.

Если $p_{\min} > \frac{\text{OPT}}{3}$, то S - оптимальное расписание. Иначе $p_l \leq \frac{\text{OPT}}{3}$ и по (12) получаем :

$$w(S) \leq \text{OPT} + \frac{\text{OPT}}{3}(1 - \frac{1}{m}) \leq \frac{4}{3} \text{OPT} \quad (13)$$

Что завершает доказательство. ▷

Список литературы

- [1] <http://research.microsoft.com/en-us/um/people/dechakr/Courses/CO454/Lectures/lecture12.pdf>
- [2] М.Гэри, Д.Джонсон "Вычислительные машины и труднорешаемые задачи". М.: Мир, 1982.