Задача об оптимальном расписании

Batyr Ishanov

26 ноября 2022 г.

1 Постановка задачи

Имеется множество работ J и множество машин M. Также задана функция $p : J \times M \to R_+$. Значение p_{ij} означает время выполнение i-й работы на j-й машине.

Требуется построить распределение работ по машинам так, чтобы все работы были выполнены и чтобы конечное время выполнения всех работ было минимально.

Рассмотрим частный случай задачи, когда производительности всех машин одинаковы, то есть $p_{ij} = p_i$.

Язык $\{(J, m, k) |$ на m одинаковых машинах выполнятся все задачи из J за время $\leq k\}$ будем называть PARALLEL-SCHEDULING.

2 NP-полнота

Доказательство **NP**-полноты состоит из двух пунктов: док-ва принадлежности к **NP** и док-ва **NP**-сложности.

2.1 Принадлежность к NP

Утв. *PARALLEL-SCHEDULING* $\in$ **NP**

Док-во. Сертификат - распределение задач по машинам, его проверка выполняется не более чем за полиномиальное время (суммируем время задач для каждой машины и проверяем, что оно $\leq k$). ■

2.2 NP-сложность

SUBSET-SUM$= \{(n_1, ..., n_l, N) \mid \exists \alpha \in \{0, 1\}^l \ \sum \alpha_i n_i = N\}$

Доказывать **NP**-сложность будем через сведение SUBSET-SUM к нашему языку.

Утв. *SUBSET-SUM -* **NP***-полный*

Док-во. Доказательство возьмем отсюда. ■

Утв. *SUBSET-SUM* $\leq_p$ *PARALLEL-SCHEDULING*

Док-во. Приведем функцию сводимости f. Пользоваться будем идеей, что SUBSET-SUM является частным случаем PARALLEL-SCHEDULING для 2 машин. Опишем f для 3х случаев:

1. $\sum_i n_i = 2N$

   Тогда $(n_1, ..., n_l, N) \longmapsto ((n_1, ..., n_l), 2, N)$. Если у нас есть подмножество $\{n_{i_1}, ..., n_{i_m}\}$ такое, что $\sum_{j=1}^{m} n_{i_j} = N$, то мы может дать эти задачи одной машине, а все остальные - другой. Каждая машина справится за N времени, значит, и суммарное время будет N. Если же такого подмножества нет, то какой-то машине достанется задач на $\geq N + 1$ времени, а значит, и суммарное время будет $\geq N + 1$.

2. $\sum_i n_i < 2N$

   Тогда $(n_1, ..., n_l, N) \longmapsto ((n_1, ..., n_l, 2N - \sum_i n_i), 2, N)$. Заметим, что сумма длительностей всех задач все так же $2N$, поэтому рассуждение работает точно так же, как в предыдущем пункте.

3. $\sum_i n_i > 2N$

   Тогда $(n_1, ..., n_l, N) \longmapsto ((n_1, ..., n_l, \sum_i n_i - 2N), 2, \sum_i n_i - N)$. Если у нас есть подмножество $\{n_{i_1}, ..., n_{i_m}\}$ такое, что $\sum_{j=1}^{m} n_{i_j} = N$, то мы может дать задачи $\{n_{i_1}, ..., n_{i_m}, \sum_i n_i - 2N)\}$ одной машине, а все остальные - другой. Если же такого подмножества нет, то какой-то машине достанется задач на $\geq \sum_i n_i - N + 1$ времени, а значит, и суммарное время будет $\geq \sum_i n_i - N + 1$. ■

3 Алгоритм вычисления

Алгоритм заключается в следующем:

1. Отсортируем список задач по убыванию длительности

2. Будем последовательно раздавать по задаче из списка наименее загруженной машине

Утв. *Данный алгоритм дает $\frac{4}{3}$-приближение*
Док-во. Будем д-ть от противного - попробуем привести контрпример.

Обозначим оптимальный ответ за t^*, ответ алгоритма за t, а длительность каждой задачи за q_i (всего задач n, а машин - m).

Для начала заметим, что мы можем рассматривать контрпример, в котором только одна задача заканчивается в последний момент t. Если есть другие задачи, заканчивающиеся в момент t, мы можем их просто выкинуть и получить контрпример поменьше (оптимальный ответ при выкидывании задач может только улучшиться, а вот наш не изменится).

Обозначим $\tau = t^* - q_n$. Тогда

$$m\tau \leq \sum_{i \neq n} q_i \tag{1}$$

(это верно, т.к. если бы какая-то машина заканчивала раньше τ, то мы бы могли дать последнюю задачу ей и улучшить ответ).

Заметим также, что

$$t^* \geq \frac{1}{m} \sum_{i=1}^{n} q_i \tag{2}$$

(даже если бы могли расщепить каждую задачу на единичные, получили бы ответ не лучше этого).

По предположению: $\frac{t}{t^*} > \frac{4}{3}$.
Но также $\frac{t}{t^*} = \frac{\tau + q_n}{t*} = \frac{\tau}{t^*} + \frac{q_n}{t^*} \leq^{1} \frac{\sum_{i \neq n} q_i}{mt^*} + \frac{q_n}{t^*} \leq \frac{\sum_{i=1}^{n} q_i}{mt^*} + \frac{q_n}{t^*} \leq^{2} 1 + \frac{q_n}{t^*}$

Тогда получаем: $1 + \frac{q_n}{t^*} > \frac{4}{3} \Rightarrow q_n > \frac{t^*}{3}$

А это значит, что в нашем контрпримере каждой машине достается ≤ 2 задач.

Давайте возьмем какое-нибудь распределение задач (≤ 2 задач на машину) и доведем его до оптимального 2-умя операциями:

1. Поменять местами 2 задачи в пределах одной машины. Очевидно, что этой операцией мы никак не изменим ответ.

2. Пусть в 1-ой машине 1-ая задача заканчивается позже 1-ой задачи во второй машине, то же самое для 2-ых задач. 2-ая операция - поменять 2-ые задачи местами. Этой операцией мы точно улучшаем ответ.

Заметим, что с помощью 2-ух этих операций (а их, очевидно, конечное кол-во), мы получим оптимальный ответ для случая, когда каждая машина берет ≤ 2 задач. Но наш алгоритм будет действовать так же (с точностью до одинаковых по времени задач), значит, ответ будет тем же.

Получили, что $t^* = t \Rightarrow$ противоречие. ■

Список литературы

[Gra69] L.R. Graham. Bounds on multiprocessing timing anomalies. pages 423–425, 1969.

[IFM] Neerc IFMO. NP-полнота задачи о сумме подмножества.

[WS11] David P. Williamson and David B. Shmoys. The design of approximation Algorithms. pages 39–43, 2011.