Задачи на параллельных машинах

Задача $P \mid pmtn \mid C_{\max}$

Имеется m одинаковых машин и n работ. Любая работа может выполняться на любой машине. Прерывания работ разрешены. Требуется найти расписание с минимальным временем завершения всех работ.

Нижняя оценка

$$LB = \max\left\{\max_{i=1,\dots,n} p_i;\ \frac{1}{m}\sum_{i=1}^{n} p_i\right\}.$$

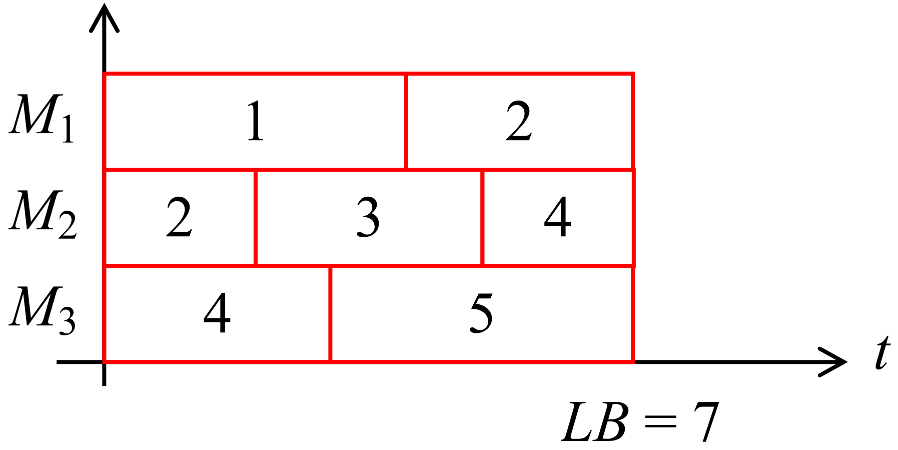
Если найдем расписание с $C_{\max} = LB$, то получим оптимальное решение.

Алгоритм: возьмем произвольный список работ и будем «загружать» машины последовательно: сначала первую машину в интервале времени $[0, LB]$, «отрезая» часть последней работы, если она не вошла целиком и, перенося эту часть на следующую машину в интервале $[0, LB]$ и т.д. Получим некоторое расписание.

Пример:

$m = 3, n = 5$

i	1	2	3	4	5
p_i	4	5	3	5	4



Так как $LB \geq \max\limits_{i=1,\ldots,n} p_i$, то в каждый момент времени работа выполняется не более чем на одной машине. Значит, полученное расписание является допустимым, и $C_{\max} = LB$, то есть получили оптимальное расписание.

Задача $P \mid pmtn, r_i \mid L_{\max}$

Имеется m одинаковых машин и n работ. Для каждой работы заданы время поступления на обслуживание $r_i \geq 0$ и директивный срок окончания $d_i \geq r_i$.

Требуется найти расписание с минимальной задержкой $L_{\max} = \max\limits_{i=1,\dots,n} (c_i - d_i)$, где c_i — время окончания работы i.

Для решения этой задачи сначала научимся отвечать на вопрос: ∃ **ли расписание с $L_{\max} \leq L$?** А затем методом дихотомии найдем минимальное значение L, для которого такое расписание существует.

Пусть L задано. Если расписание существует, то работа i выполняется в интервале $[r_i, c_i]$ и $c_i - d_i \leq L$, то есть $c_i \leq L + d_i = d_i^L$ и можно считать, что работа i выполняется в интервале $[r_i, d_i^L]$. Для того, чтобы ответить на вопрос «∃ ли расписание с прерываниями, где каждая работа выполняется в своем временном окне», нам потребуется задача о потоке в сети, которая может быть решена симплекс–методом.

Задача о потоке в сети

Задана сеть $G = (V, E, s, t)$ с одним источником s и одним стоком t.

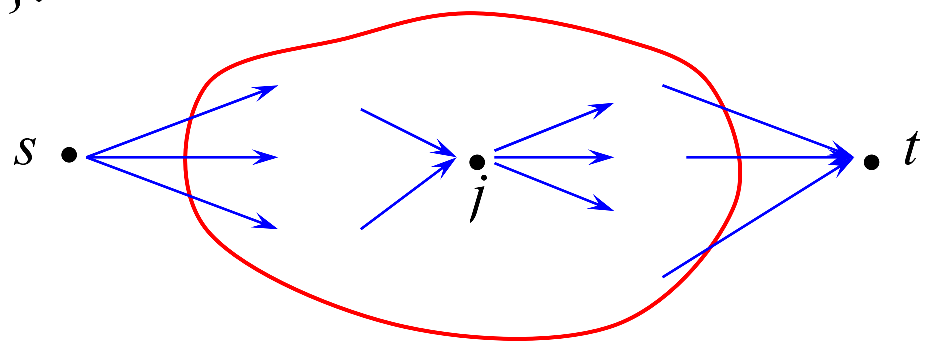
Сеть G есть ориентированный ациклический граф. Каждой дуге $(ij) \in E$ приписан вес $c_{ij} \geq 0$ — пропускная способность дуги.

Определение. ***Потоком*** в сети G называется функция $F : E \to R$ на дугах, которая удовлетворяет условиям на пропускные способности

$$0 \leq f_{ij} \leq c_{ij}, \quad (ij) \in E$$

и сохраняет поток в каждой внутренней вершине $v \in V \setminus \{s, t\}$:

$$\sum_{\substack{i \in V \\ (ij) \in E}} f_{ij} = \sum_{\substack{i \in V \\ (ji) \in E}} f_{ji}, \quad \forall j \in V \setminus \{s, t\}.$$



Легко проверить, что $\sum\limits_{\substack{i \in V \\ (si) \in E}} f_{si} = \sum\limits_{\substack{i \in V \\ (it) \in E}} f_{it}$.

Определение. Величина $M(f) = \sum\limits_{\substack{i \in V \\ (si) \in E}} f_{si}$ называется ***мощностью потока***.

Задача состоит в том, чтобы найти поток максимальной мощности.

Заметим, что это задача линейного программирования. Множество допустимых решений задачи не пусто, т.к. решение $f_{ij} = 0,\ \forall (ij) \in E$, является допустимым решением задачи. Целевая функция ограничена сверху, следовательно, оптимальное решение существует.

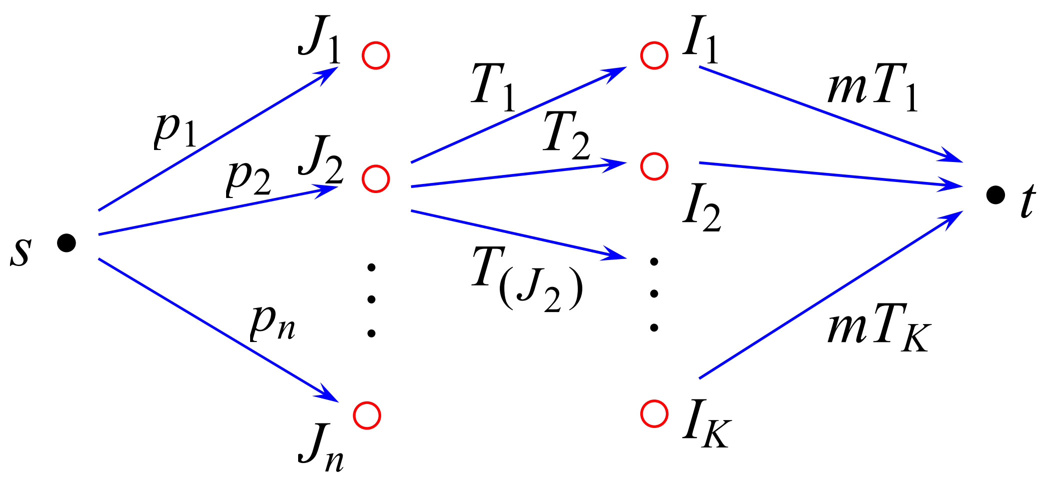
Покажем, как с помощью задачи о потоке в сети найти расписание, удовлетворяющее временным окнам.

Сольем два массива r_i, d_i^L и упорядочим их значения

$$t_1 < t_2 < \ldots < t_k, \quad k \le 2n.$$

Рассматриваем только различные значения r и d.

Построим интервалы $I_k = [t_k, t_{k+1}]$, длины $T_k = t_{k+1} - t_k$ и рассмотрим сеть $G = (V, E, s, t)$:



Дуга (i, k) принадлежит E, если работа J_i может выполняться в интервале I_k, т.е. $I_k \subseteq [r_i, d_i^L]$. Каждой дуге приписан вес, как показано на рисунке.

Решим задачу о максимальном потоке в этой сети. Получим $\max M(F)$. Сравним эту величину с $\sum_{i=1}^{n} p_i$. Если они равны, то искомое расписание существует, если нет, то есть $\sum p_i > M(F)$, то такого расписания не существует.

Пусть имеет место равенство. Тогда сохранение потока в каждой вершине J_i дает:

$$\sum_k f_{ik} = p_i, \quad \forall i = 1,\ldots,n$$

и величины f_{ik} определяют расписание работы J_i.

Сохранение потока в вершине I_k: $\sum_i f_{ik} \le mT_k, \quad \forall k = 1,\ldots,K$, гарантирует, что m машин справятся со всеми работами в интервале T_k.

Задача $Q \mid pmtn \mid C_{\max}$

Имеется m машин со скоростями $s_1 \geq s_2 \geq \ldots \geq s_m$ и n работ с трудоемкостью $p_1 \geq p_2 \geq \ldots \geq p_n$. Время выполнения $p_{ij} = p_i / s_j$. Разрешаются прерывания.

Найти расписание с минимальным временем выполнения всех работ.

Сначала найдем нижнюю оценку на $C_{\max}$, затем построим расписание с $C_{\max} = LB$, то есть оптимальное!

Положим $P_i = p_1 + p_2 + \ldots + p_i$, $S_j = s_1 + s_2 + \ldots + s_j$.

Предполагаем, что $n \geq m$, иначе выбрасываем $(m - n)$ самых медленных машин.

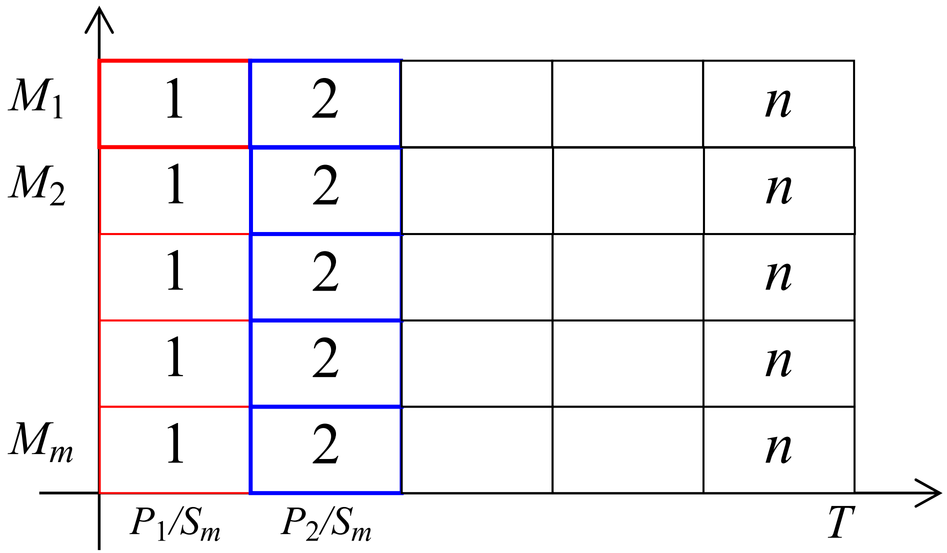
Если хотим выполнить все работы в интервале $[0, T]$, то $P_n \leq S_m T$.

Аналогично, $P_j \leq S_j T$, $\forall j = 1, \ldots, m - 1$, так как P_j / S_j есть нижняя граница для выполнения работ $j' = 1, \ldots, j$. Таким образом,

$$LB = \max\{ \max_{j=1,\ldots,m-1} P_j / S_j ; P_n / S_m \}$$

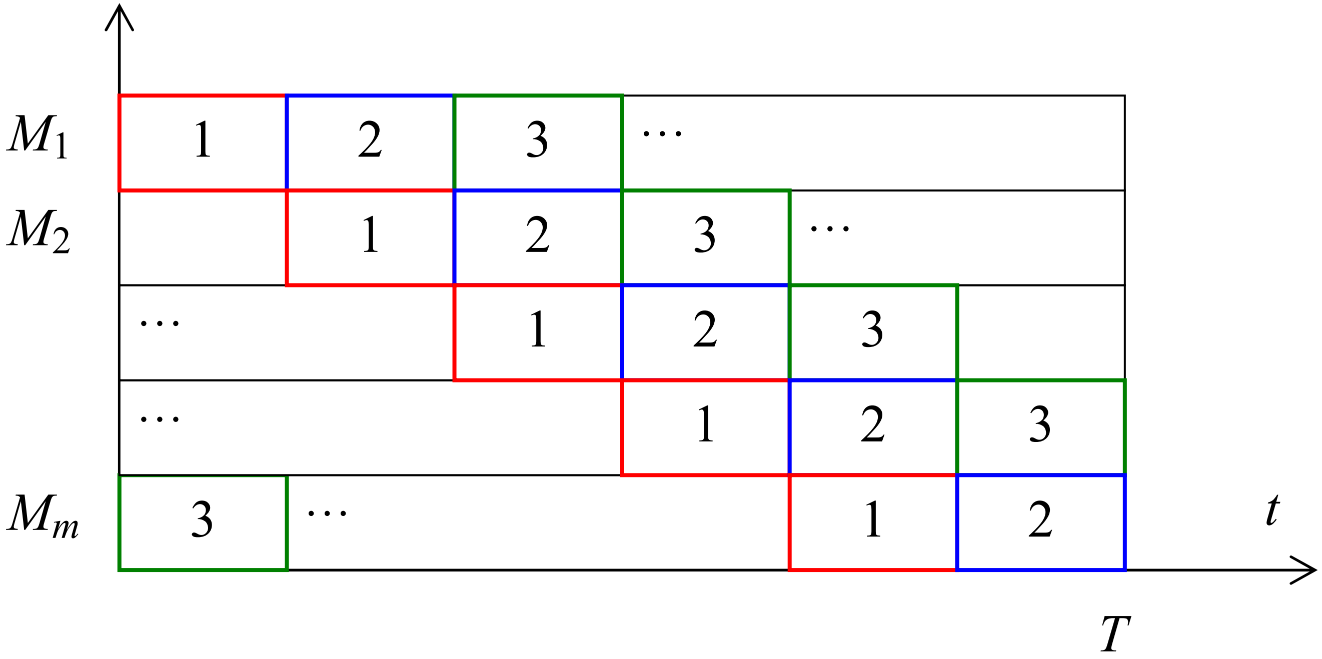
есть нижняя граница для $C_{\max}$.

Предположим, что $C_{\max} = T$ и до момента T ни одна машина не простаивала. Тогда $T = P_n / S_m$. Если бы можно было выполнить работу сразу на всех машинах, то расписание легко построить:



Но это расписание легко переделать так, чтобы работа не была одновременно на нескольких машинах.

Так как $n \geq m$, то сдвинем работы по времени следующим образом:



Совместное выполнение работ

Это оптимальное расписание, если $p_i = p_j \; \forall \; i \; j = 1,\dots, n$. Если $p_i = p$, то мы умеем строить оптимальное расписание и оно обладает тем свойством, что ни одна машина не простаивает до завершения всех работ.

Теперь рассмотрим общий случай разных p_i. В нем работы с большими длительностями ставятся на самые быстрые машины до тех пор, пока их длительность не сократится настолько, что совпадет с длительностью других работ, образуя группу одинаковых работ, а группу мы умеем расписывать оптимально.

Обозначим через $p_i(t)$ часть работы i, которая еще не выполнена к моменту t. Наш алгоритм будет двигаться по t и в некоторых моментах времени s останавливаться для переназначения работ по машинам.

Алгоритм Level

1. t := 0
2. While ∃ работа с $p(t) > 0$ do

   2.1. Assign (t)

   2.2. $t_1 := \min\{s > t \mid \exists$ работа завершающаяся в момент $s\}$

   2.3. $t_2 := \min\{s > t \mid \exists\ ij,\ p_i(t) > p_j(t)$ и $p_i(s) = p_j(s)\}$

   2.4. $t = \min\ \{\ t_1, t_2\}$

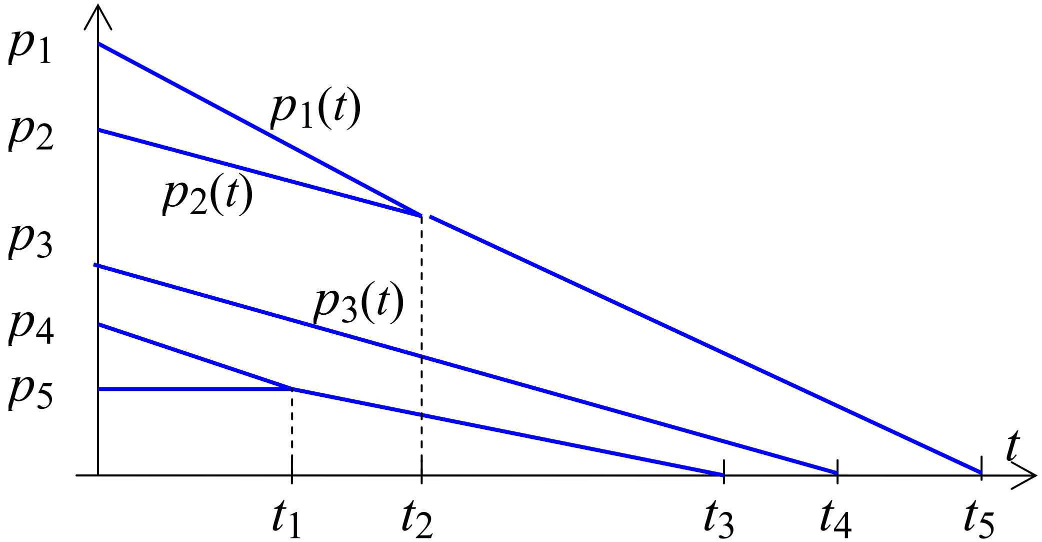
3. Восстановить расписание работ.

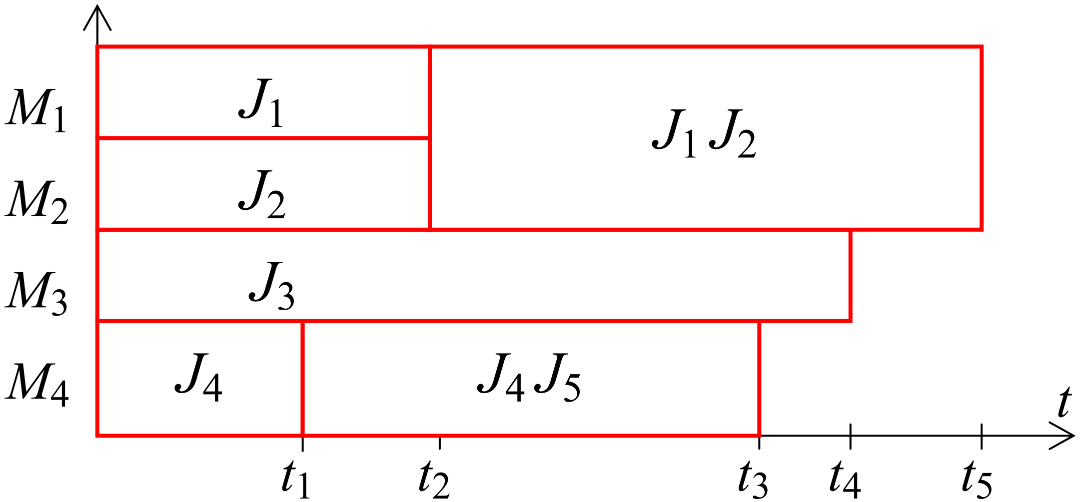
Процедура Assign (t) производит назначение работ по машинам.

Процедура Assign

1. $J := \{j \mid p_j(t)>0\}$.
2. $M := \{M_1,\ldots, M_m\}$.
3. Всем $j\in J$ и $M_i\in M$ присвоить статус «свободен».
4. While $\exists$ (свободные работы) и (свободные машины) do
    - 4.1. Найти множество $I \subseteq J$ всех работ с $p_i(t) = \max\limits_{k\in J} p_k(t)$.
    - 4.2. $k := \min\{|M|, |I|\}$.
    - 4.3. Назначить работы из I на k самых быстрых машин из M для совместной обработки.
    - 4.4. $J := J \setminus I$.
    - 4.5. Исключить k самых быстрых машин из M.

Пример Имеется 5 работ и 4 машины





В момент 0 начали выполняться 4 работы.

В момент t_1 $p_5 = p_4(t_1)$ и далее они выполняются вместе на машине 4.

В момент t_2 $p_1(t_2) = p_2(t_2)$ и далее они выполняются вместе на двух самых быстрых машинах.

Заметим, что кривая $p_1(t)$ всегда остается выше всех других, $p_2(t)$ не ниже $p_i(t)$, $i > 2$ и т.д., т.е.

$$p_1(t) \geq p_2(t) \geq \ldots \geq p_n(t), \;\; \forall t \geq 0,$$

и процедура Assign (t) назначает работы на машины именно в этом порядке

Теорема. Алгоритм Level строит оптимальное расписание.

Доказательство. Достаточно убедится в том, что алгоритм закончит работу в момент

$$t = LB = \max\left\{ \max_{j=1}^{m-1} P_j / S_j \,;\; P_n / S_m \right\}.$$

1. Если к моменту завершения всех работ ни одна машина не простаивала, то $t = P_n / S_m$ и решение оптимально.

2. Пусть машины завершили свою работу в разное время. Тогда $f_1 \geq f_2 \geq \ldots \geq f_m$, f_i — время остановки машины i, и хотя бы одно неравенство строгое, т.е.

$$T = f_1 = f_2 = \ldots = f_j > f_{j+1} \quad \text{и} \quad j < m$$

Но работы, заканчивающиеся в момент T, должны были начаться в $t = 0$, и тогда $T = P_j / S_j$.

Убедимся в том, что все работы, заканчивающиеся в момент T, начали выполняться в $t = 0$. Предположим, что это не так, т.е. $\exists$ работа i, которая началась в момент $t_i > 0$. Тогда в $t = 0$ начались другие работы: 1, 2, …, m и

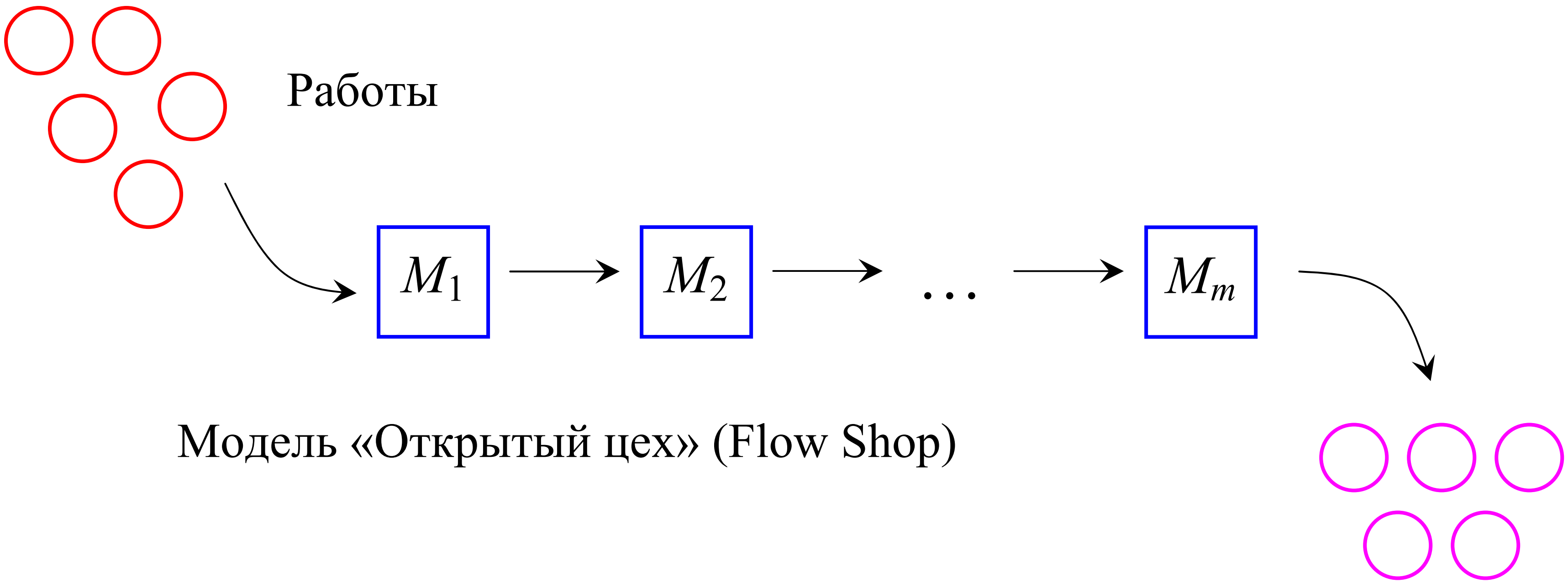
$$p_1(0) \geq p_2(0) \geq \ldots \geq p_m(0) > p_i(0).$$

Все машины работали до $t \leq t_i$, а в момент t_i мы получим $p_1(t_i) = p_2(t_i) = \ldots = p_m(t_i) = p_i(0)$ и далее они выполнялись вместе, т.е. машины не простаивали. Это противоречит предположению. ■

Задача $F \,||\, C_{\max}$

Имеется n работ, каждая из которых должна пройти обработку последовательно на всех машинах $M_1, M_2,\ldots, M_m$, т.е. каждая работа состоит из m операций и для всех работ порядок выполнения операций один и тот же.

Требуется найти расписание выполнения работ за наименьшее время.

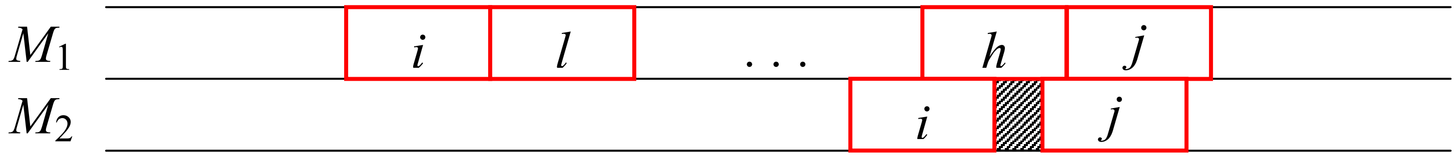


Модель «Открытый цех» (Flow Shop)

Теорема. Существует оптимальное расписание для задачи $F \| C_{\max}$, обладающее следующими свойствами:

1. Последовательности выполнения работ на первой и второй машинах одинаковы.
2. Последовательности выполнения работ на последней и предпоследней машинах одинаковы.

Доказательство. 1. Пусть утверждение не верно и среди всех оптимальных расписаний выберем такое, что последовательности на M_1 и M_2 совпадают для первых k работ, $k < n$ и k — максимально.

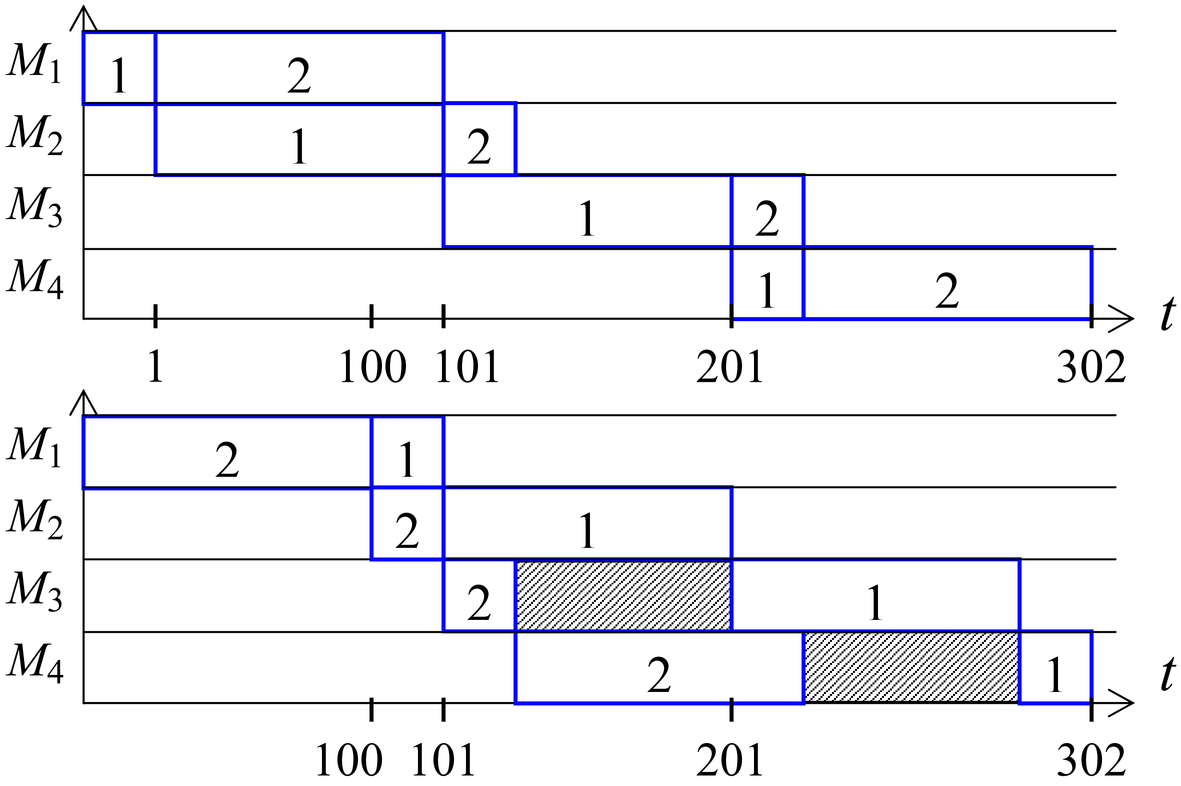


Обозначим эту k-ю работу через J_i. На первой машине за ней идет J_l. На второй машине — J_j. Если на первой машине поставим J_j между J_i и J_l, то длина расписания не изменится, но k увеличится. Получили противоречие. Второе утверждение доказывается аналогично. ■

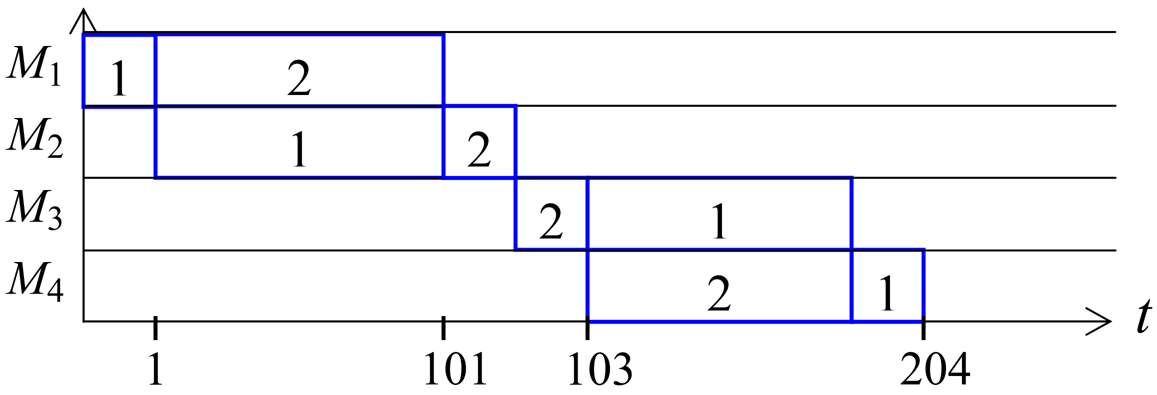
Следствие. При $m \le 3$ существует оптимальное решение задачи $F \parallel C_{\max}$ с одинаковым порядком выполнения работ на всех машинах.

Контрпример для $m = 4$.

$n = 2$ и длительности операций для J_1 и J_2 задаются векторами: (1, 100, 100, 1) и (100, 1, 1, 100). Если порядки выполнения работ на всех машинах одинаковы, то их всего два: (J_1, J_2) или (J_2, J_1). Тогда в обоих случаях $C_{\max} = 302$.



Оптимальное решение $C_{\max} = 204$



Порядок выполнения работ на M_2 и M_3 разный. Если при $m > 3$ искать решение в виде одной перестановки, то отношение перестановочного оптимума и глобального оптимума может достигать величины $0{,}5\sqrt{m}$.

Задача Джонсона $F2 \,||\, C_{\max}$

Пусть заданы перестановка Π, определяющая порядок выполнения работ на двух машинах. Соответствующее расписание представим в виде сетевого графика:

$$
\begin{tikzcd}
M_1 & \Pi_1 \arrow[r] \arrow[d] & \Pi_2 \arrow[r] \arrow[d] & \Pi_3 \arrow[r] \arrow[d] & \cdots \arrow[r] & \Pi_n \arrow[d] & \\
M_2 & \Pi_1 \arrow[r] & \Pi_2 \arrow[r] & \Pi_3 \arrow[r] & \cdots \arrow[r] & \Pi_n & \text{сток}
\end{tikzcd}
$$

Каждой вершине приписан вес равный длительности выполнения соответствующей операции. Заметим, что при заданной перестановке Π время окончания всех работ ($C_{\max}$) равно длине максимального пути из источника в сток. Этот путь на некоторой работе s переходит от M_1 к M_2.

Пусть a_j — время выполнения работы j на M_1,

b_j — время выполнения работы j на M_2.

Тогда $$C_{\max}(\Pi) = \max_{1 \le s \le n} \left(\sum_{k=1}^{s} a_{\Pi_k} + \sum_{k=s}^{n} b_{\Pi_k} \right).$$

Теорема. Перестановка $\Pi^0 = (1,\ 2,\ \dots\ ,\ n)$ оптимальна, если существует номер q такой, что

1. $a_j \le b_j,\ \ j = 1,\dots,q$ и $a_1 \le a_2 \le \dots \le a_q$;

2. $a_j \ge b_j,\ \ j = q+1,\dots,n$ и $b_{q+1} \ge b_{q+2} \ge \dots \ge b_n$;

или более наглядно

$$\begin{array}{ccccccccc}
a_1 \le & a_2 \le & \dots \le & a_q & a_{q+1} & a_{q+2} & \dots & a_n \\
\wedge\!| & \wedge\!| & & \wedge\!| & |\!\vee & |\!\vee & & |\!\vee \\
b_1 & b_2 & \dots & b_q & b_{q+1} \ge & b_{q+2} \ge & \dots \ge & b_n
\end{array}$$

Доказательство. Пусть Π — произвольная перестановка и

$C_{\max}(\Pi,s)=\sum_{k=1}^{s}a_{\Pi_k}+\sum_{k=s}^{n}b_{\Pi_k}$. Поскольку $C_{\max}(\Pi)=\max_{s}C_{\max}(\Pi,s)$, то

достаточно показать, что для всякого s найдется номер r такой, что

$$C_{\max}(\Pi^0,s)\le C_{\max}(\Pi,r).$$

В случае $s\le q$ выберем r так чтобы $\Pi_r\in\{s,\ \ldots,\ q\}\subset\{\Pi_r,\ \Pi_{r+1},\ldots,\ \Pi_n\}$.

Для этого достаточно в перестановке Π найти работу из $\{s,\ \ldots,\ q\}$ с наименьшей позицией. Эту работа обозначили через Π_r. Тогда

$$C_{\max}(\Pi^0,s)=\sum_{k=1}^{s-1}a_k+a_s+\sum_{k=s}^{q}b_k+\sum_{k=q+1}^{n}b_k=a_s+\sum_{k\in\overline{s,q}}b_k+\sum_{k\notin\overline{s,q}}\min(a_k,b_k),$$

где $\overline{s,q}=\{s,\ldots,q\}$.

Для перестановки Π величина $C_{\max}(\Pi, r)$ представима в виде

$$C_{\max}(\Pi, r) = a_{\Pi_r} + \sum_{k \in \overline{s,q}} b_k + \sum_{k \notin \overline{s,q}} c_k ,$$

где $c_k \geq \min(a_k, b_k)$. Второе слагаемое содержит только величины b_k, так как $\{s, \ldots, q\} \subset \{\Pi_r, \Pi_{r+1}, \ldots, \Pi_n\}$. Из условия $\Pi_r \in \{s, \ldots, q\}$ получаем $a_{\Pi_r} \geq a_s$ откуда и следует нужное неравенство.

В случае $s > q$ выбираем r так, чтобы $\Pi_r \in \{q+1, \ldots, s\} \subset \{\Pi_1, \ldots, \Pi_r\}$.

Остальная часть доказательства проводится аналогично. ■