Parte 2

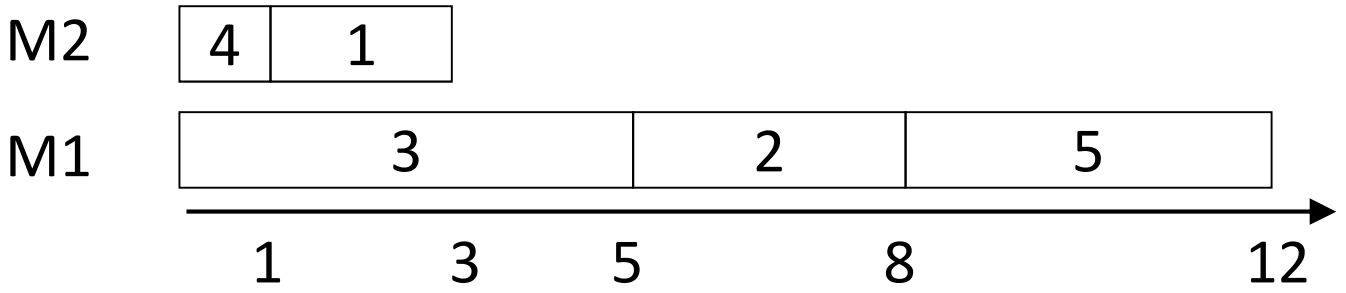
Problemi con macchine parallele

Esempio

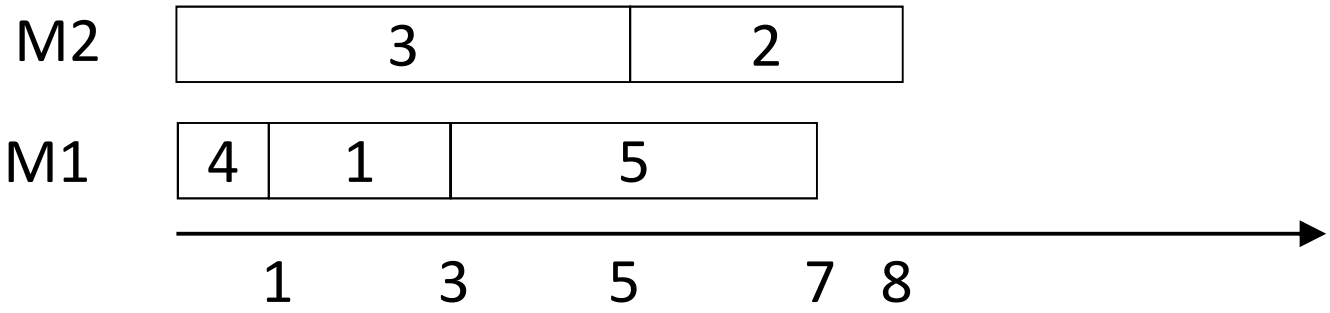
job	1	2	3	4	5
p_j	2	3	5	1	4

2 macchine

Assegnando {2,3,5} a M1 e {1,4} a M2



Assegnando {1,4,5} a M1 e {2,3} a M2



$R_m//C_{max}$

Algoritmo 2-approssimato basato sulla PL

$R_m//C_{max}$: formulazione PLM

variabili decisionali: $x_{ij} = 1$ se job j assegnato alla macchina i e 0 altrimenti, $C_{max} \in \Re$

$$\min \quad C_{\max}$$

$$\sum_{i=1}^{m} x_{ij} = 1, \qquad j = 1,\ldots,n$$

$$\sum_{j=1}^{n} p_{ij} x_{ij} \le C_{\max}, \qquad i = 1,\ldots,m$$

$$x_{ij} \in \{0,1\} \qquad i = 1,\ldots,m; j = 1,\ldots,n$$

Algoritmo 2-approssimato

1. Risolviamo il rilassamento lineare, ottenuto sostituendo $x_{ij} \in \{0,1\}$ con $x_{ij} \geq 0$

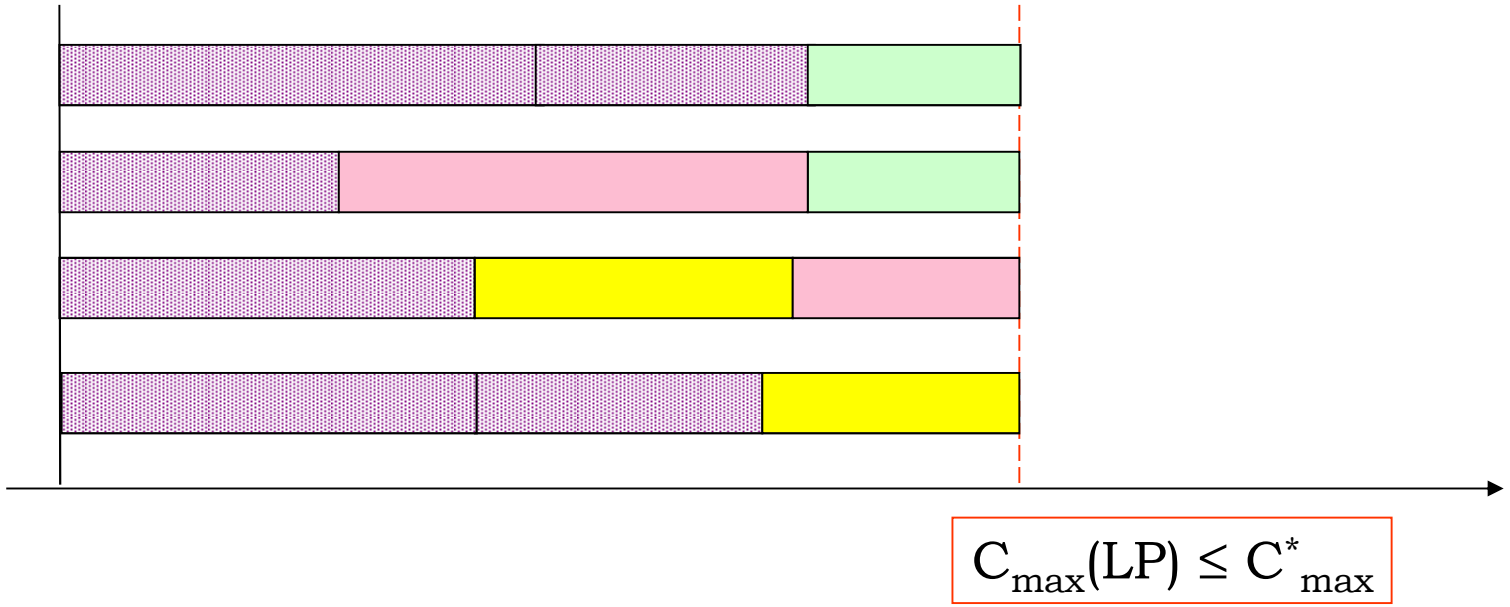
- il numero di variabili non nulle in una soluzione di base è minore o uguale al numero $m + n$ di righe della matrice.

- Essendo C_{max} sempre positivo, al più $m + n - 1$ variabili x_{ij} sono positive. Inoltre, ciascun job ha almeno una variabile positiva ad esso associata.

- Quindi, al più $m - 1$ job sono stati "frazionati" su due o più macchine.

- il valore della soluzione $\mathrm{C_{max}(LP)}$ è una limitazione inferiore del valore ottimo

Algoritmo 2-approssimato

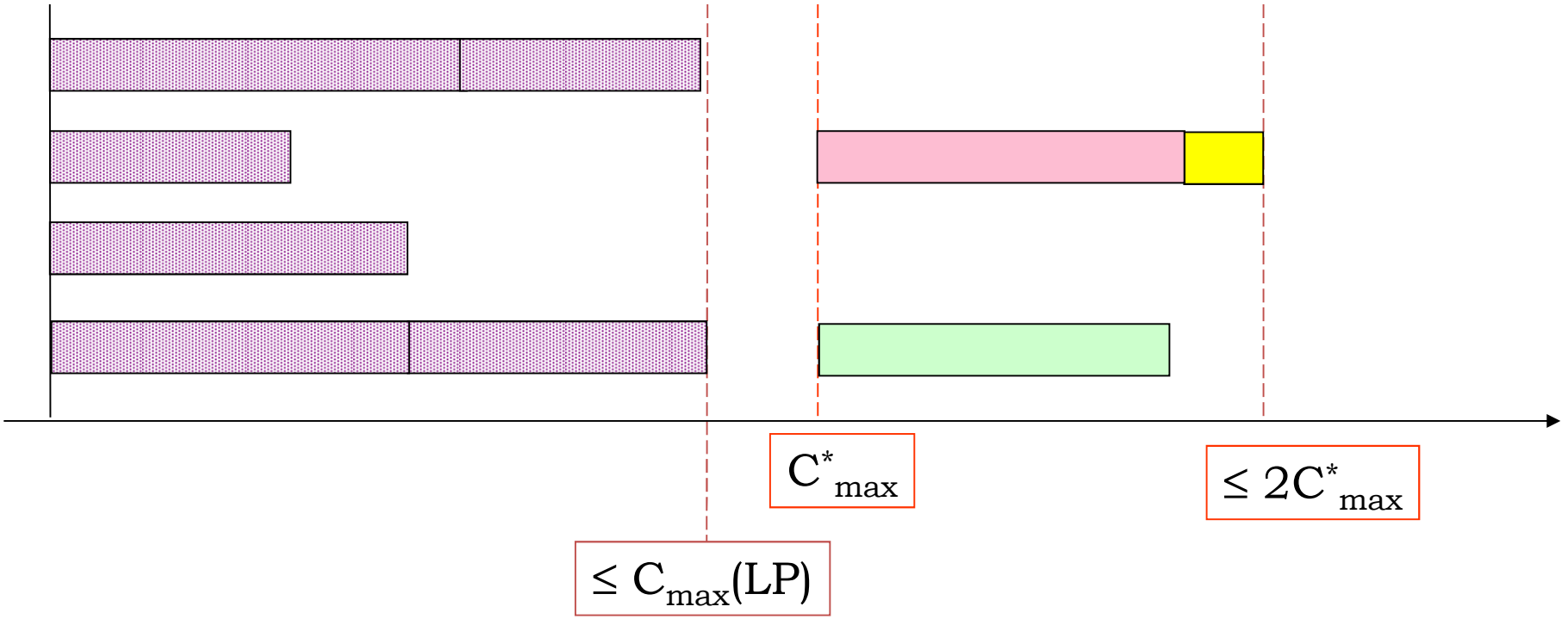


$C_{max}(LP) \leq C^*_{max}$

2. Costruiamo lo schedule in due fasi:

- job assegnati in modo intero dall'LP: mantengono tale assegnamento.
- job frazionari (al più $m - 1$): per enumerazione totale calcoliamo uno schedule ottimo in tempo $\mathbf{O(m^{(m-1)})}$

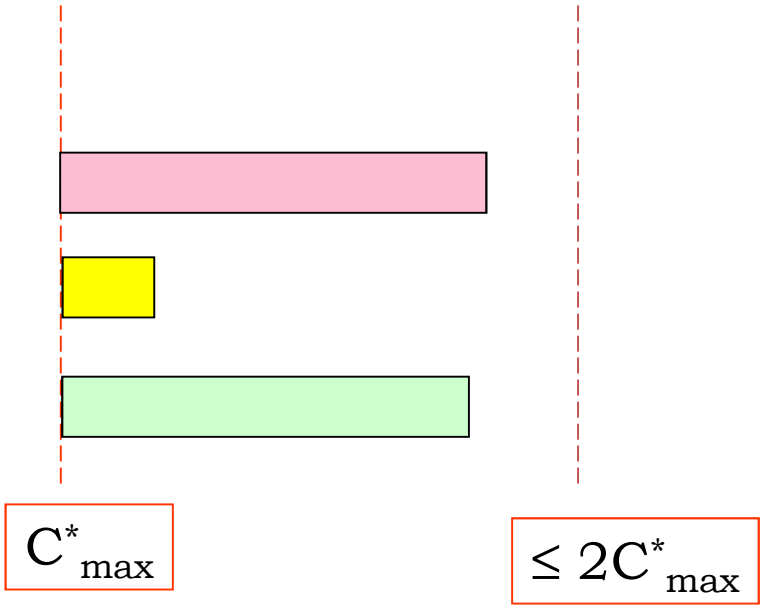
Algoritmo 2-approssimato



- lo schedule parziale dei job interi ha durata $\leq C_{max}(LP) \leq C^*_{max}$ (job grigi)
- lo schedule parziale dei job frazionari ha durata $\leq C^*_{max}$

3. Concatenando i due schedule parziali si ottiene una soluzione pari ad al più $2C^*_{max}$

caso speciale: $P_m//C_{max}$



2. lo schedule ottimo dei job frazionari è: $m-1$ macchine ricevono **1 job** (qualunque) ed una rimane scarica.

Quindi, la complessità dell'algoritmo si riduce a quella della soluzione di un problema di PL

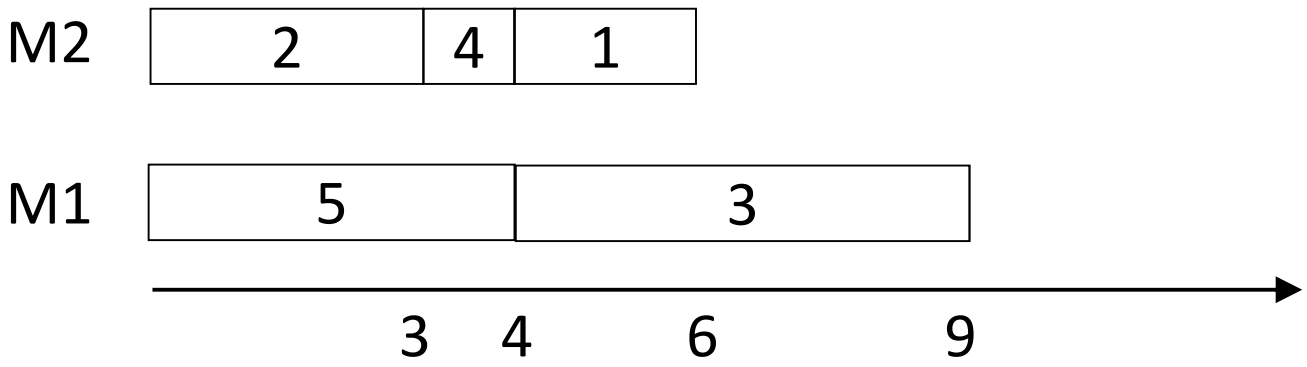
$P_m//C_{max}$

algoritmi approssimati

List Scheduling

List Scheduling (LS): data una (qualsiasi) sequenza fissata dei job, ogni volta che una macchina i si libera, il successivo job della lista viene processato su i

(5,2,4,3,1)



List Scheduling

indichiamo con $C_{\max}^{LS}$, $C_{\max}^{*}$ il valore della soluzione calcolata da LS e l'ottimo rispettivamente

Teorema: per qualunque sequenza di job, risulta:

$$C_{\max}^{LS} \leq (2 - 1/m) C_{\max}^{*}$$

Dimostrazione. Sia k l'ultimo job ad essere completato nello schedule LS e $s_k = C_k - p_k$ il suo istante di inizio.

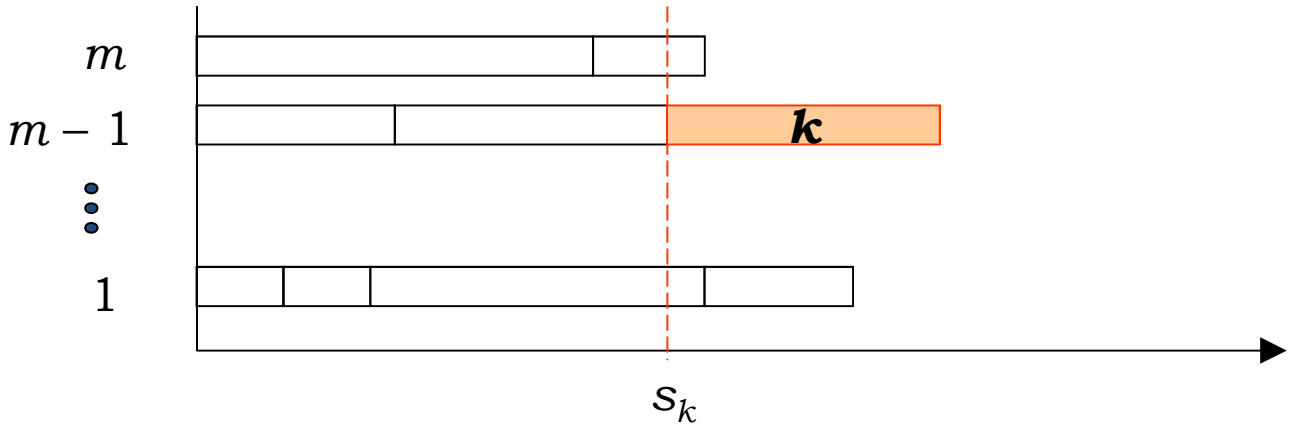
Valgono le seguenti tre relazioni:

(a) $C_{\max}^{*} \geq p_k$ k deve essere processato

LS: dimostrazione

(b) $C^*_{\max} \geq (1/m)\sum_{j=1}^{n} p_j$ macchine perfettamente bilanciate

Inoltre, nessuna macchina può essere ferma prima di s_k



Quindi,

(c) $s_k \leq (1/m)\sum_{j \neq k} p_j$ macchine perfettamente bilanciate senza il job k

LS: dimostrazione

Da cui:

$$C_{\max}^{LS} = C_k = s_k + p_k \leq (1/m)\sum_{j\neq k}^{n} p_j + p_k =$$

(c)

$$= (1/m)\sum_{j=1}^{n} p_j + (1-1/m)p_k \leq$$

$$\leq C_{\max}^{*} + (1-1/m)C_{\max}^{*} = (2-1/m)C_{\max}^{*}$$

(b)+(a)

□

Regola LPT

Longest Processing Time (LPT) first: ordina i job per tempi di processamento non crescenti

Assumiamo che $p_1 \geq p_2 \geq \ldots \geq p_n$

Teorema: per qualunque istanza di $P_m//C_{max}$, detta C_{max}^{LPT} la soluzione calcolata da LPT, risulta

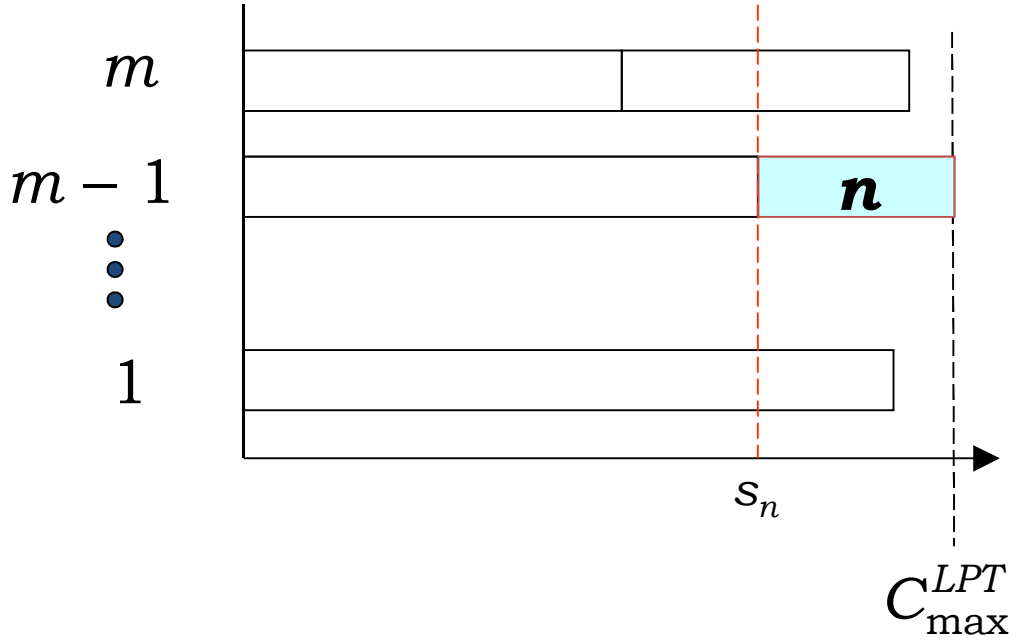
$$C_{max}^{LPT} \leq (4/3 - 1/3m) C_{max}^{*}$$

Dimostrazione. Per contraddizione. Assumiamo che esistano controesempi, cioè istanze di $P_m//C_{max}$ per cui

$$C_{max}^{LPT} / C_{max}^{*} > (4/3 - 1/3m)$$

LPT: dimostrazione

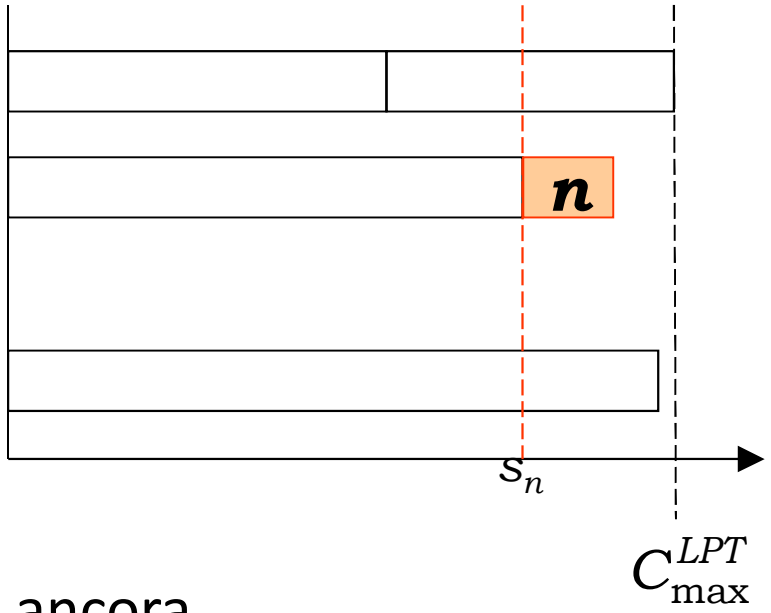
Consideriamo, fra essi, il controesempio con il minimo numero n di job.
Per questo vale la seguente proprietà: nello schedule LPT il job n (il più corto) è l'ultimo ad iniziare e anche **l'ultimo a terminare**.



LPT: dimostrazione

Infatti, se così non fosse, la rimozione di n produrrebbe una nuova istanza per cui:

- $C_{\max}^{LPT}$ non cambia
- $C_{\max}^{*}$ non cresce



Quindi, per la nuova istanza si ha ancora

$$C_{\max}^{LPT} / C_{\max}^{*} > (4/3 - 1/3m)$$

ed ha un numero inferiore di job, contraddicendo la minimalità del controesempio.

LPT: dimostrazione

Quindi, per il più piccolo controesempio, risulta $s_n = C_{\max}^{LPT} - p_n$

In tale istante, tutte le macchine sono occupate, quindi

$$C_{\max}^{LPT} - p_n \leq 1/m \left(\sum_{j=1}^{n-1} p_j\right)$$

da cui $C_{\max}^{LPT} \leq p_n + 1/m\left(\sum_{j=1}^{n-1} p_j\right) = p_n(1-1/m) + 1/m\left(\sum_{j=1}^{n} p_j\right)$

inoltre: $C_{\max}^{*} \geq (1/m)\sum_{j=1}^{n} p_j$

quindi: $C_{\max}^{LPT} \leq (1-1/m)\, p_n + C_{\max}^{*}$

Dimostrazione

ricordando l'ipotesi: $$\frac{4}{3}-\frac{1}{3m}<\frac{C_{\max}^{LPT}}{C_{\max}^{*}}\leq\frac{p_n(1-1/m)}{C_{\max}^{*}}+1$$

cioè:

$$\frac{1}{3}-\frac{1}{3m}<\frac{p_n(1-1/m)}{C_{\max}^{*}}$$

$$(1-1/m)C_{\max}^{*}<3p_n(1-1/m)$$

infine:

$$C_{\max}^{*}<3p_n$$

quindi, per il più piccolo controesempio, uno schedule ottimo può assegnare al più due job a ciascuna macchina (quindi, $n\leq 2m$).

LPT:dimostrazione

In questo caso, uno schedule ottimo ha la seguente struttura. Sia $n = 2m - h$. Assegnamo i job 1, 2, …, h da soli su h macchine; assegnamo alle rimanenti $m - h$ macchine le seguenti coppie di job: $(h + 1, n)$, $(h + 2, n - 1)$, $(h + 3, n - 2)$, etc.

D'altro canto, questo è proprio lo schedule ottenuto da LPT.

Quindi $C_{\max}^{LPT} / C_{\max}^{*} = 1$, contraddizione.

□

il bound è stretto

job	1	2	3	4	5	6	7	8	9
p_j	7	7	6	6	5	5	4	4	4