June 28th, 201

454

R|pmtn|Cmax and O|pmtn|Cmax:

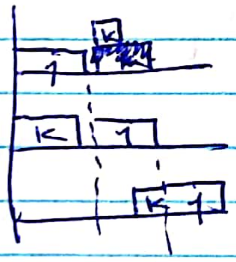
O: Open shop environment

- Each job j consists of m operations; operation i of job j has to be scheduled on m/c i, and takes P_{ij} time to complete. Operations of a job can be performed in any order, but at any point of time, at most one ~~job~~ operation of a job can be ~~running~~ processed.
- The completion time of a job = time by which all operations of the job have completed.

So,

$C_{max} :=$ max completion time of an ~~operation~~ job
= max completion time of an operation

Ex:



Must spend $\sum_{i=1}^{m} P_{ij}$ ~~pro~~ time processing the operations of job j.

O|pmtn|Cmax:

1 $C^*_{max} \geq$ max $\left\{ \max_{i=1,\dots,m} \underbrace{\sum_{j=1}^{n} P_{ij}}_{2}, \max_{j=1,\dots,n} \sum_{i=1}^{m} P_{ij} \right\}$

2 m/c i must spend this much time processing the ith operation of jobs.

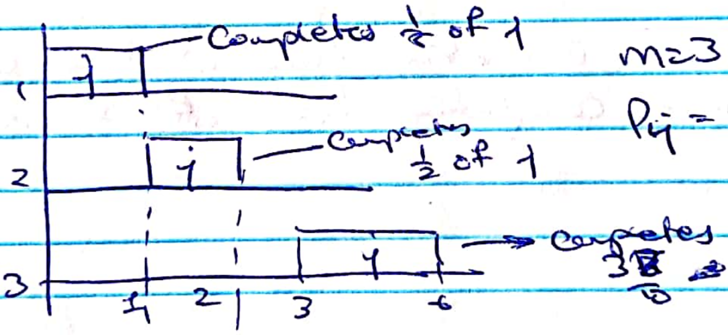
Call this LB

Theorem 1.
We can efficiently compute a schedule for $O|pmtn|C_{max}$ (with at most $mn+m+n$ preemptions) with makespan $= LB$; and hence is an opt schedule.

Theorem 2:
$R|pmtn|C_{max} \le_p O|pmtn|C_{max}$

Completion time of j = time when j is processed to the extent of 1.

Ex:



$m=3$,
$p_{1j}=5,\ p_{2j}=2,\ p_{3j}=10$

So, $\frac{1}{2}+\frac{1}{5}+\frac{3}{10}=1$.

Proof] (Thm 2).
Will write down an LP-relaxation for $R|pmtn|C_{max}$. ~~Variables~~ x.
Let variables x_{ij}: fraction of job j scheduled on m/c i
$\forall i=1,\dots,m,\ \forall j=1,\dots,n.$

→ cont'd

[Proof] (con't)

(LP)

min C_{max} → Variable to denote makespan

s.t.

(1) $\sum_{i=1}^{m} x_{ij} = 1 \quad \forall$ jobs j. (All jobs completed to the extent of 1)

(2) $\sum_{j=1}^{n} p_{ij} x_{ij} \le C_{max} \quad \forall$ m/c i.

(3) $\sum_{i=1}^{m} p_{ij} x_{ij} \le C_{max} \quad \forall$ jobs j

$C_{max}, x_{ij} \ge 0 \quad \forall i,j$

Fact! $OPT_{LP} \le OPT_{R|pmtn|C_{max}}$.

Reduction: Find an opt. sol'n (x^{LP}, C_{max}^{LP}) to (LP). Create following input for $O|pmtn|C_{max}$

For every job j, create an operation on i with length $p_{ij} x_{ij}$ $\forall i = 1, \ldots, m$

Then,

1) LB for open-shop Instance is $\le C_{max}^{LP}$ (follows from (2), (3))
2) Theorem 1 produces an open-shop schedule S of makespan = LB $\le C_{max}^{LP}$
3) S is feasible for $R|pmtn|C_{max}$.

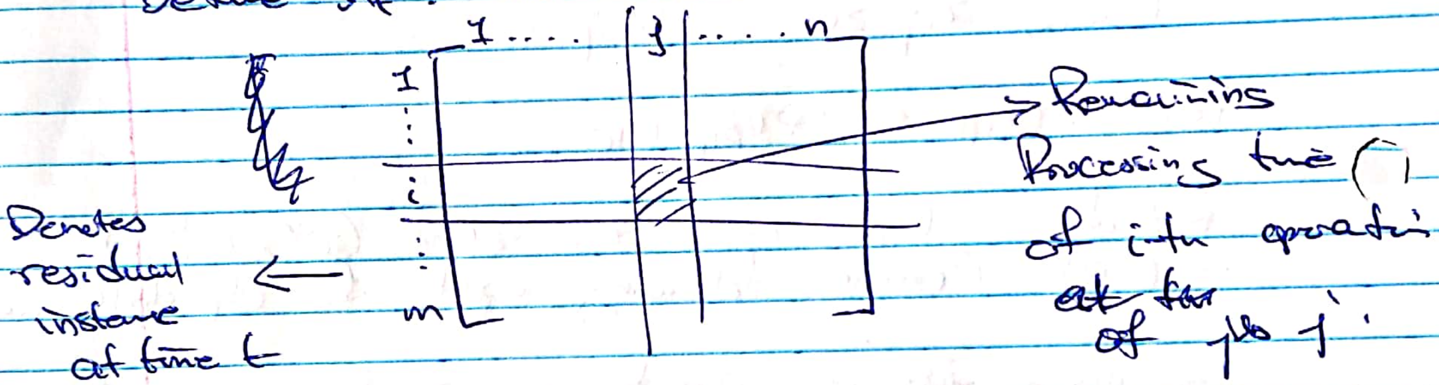
On m/c i, we spend $p_{ij}x_{ij}$ time on job j, so since $\sum_{i=1}^{m} x_{ij} = 1 \ \forall j$, we completely process each job. The $p_{ij}x_{ij}$ time units spent on i for j do not overlap with the $p_{i'j}x_{i'j}$ time spent on some other machine i' for j.

→

[Proof] (cont'd)
[Every mlc is processing ≤ 1 job at any time]
So, 1) + 2) + 3) $\Rightarrow$ S optimal for $R|pmtn|C_{max}$.

[Proof] (of Thm 1).

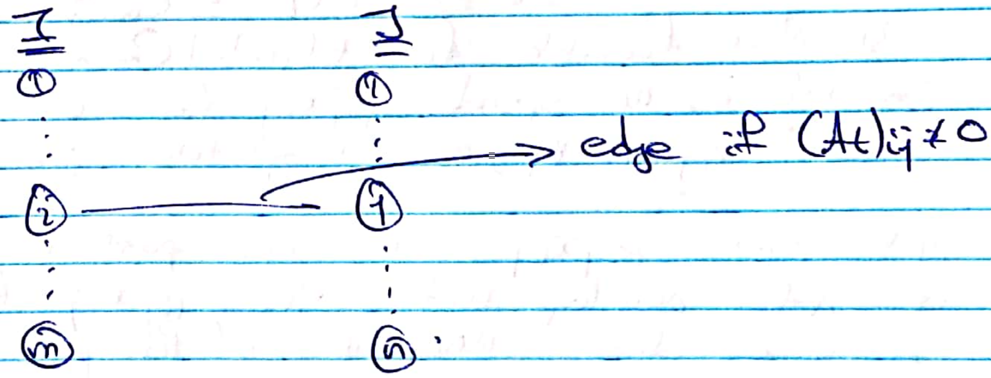
Suppose we have built a schedule up to time t.
Define A_t : $m \times n$ matrix



Let Δ_t = max row/col sum in A_t
(At $t=0$, $\Delta_0 = (B)$)

With A_t, have a corresponding bipartite graph

G_t:



[Proof] (con't)

Define :

$I_t = \{ i \in \{1, \dots, m\} : \text{row-sum for } i = \Delta_t \}$

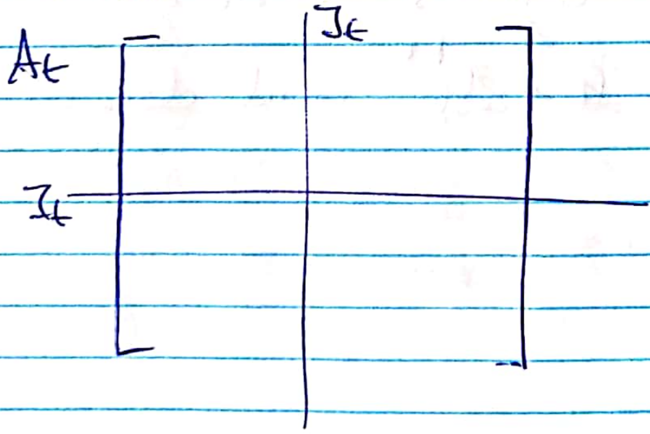
$J_t = \{ j \in \{1, \dots, n\} : \text{col-sum for } j = \Delta_t \}$.

Observations:

1) At any time t, the assignment of (operations of) jobs to m/c's forms a matching in G_t
2) $\Delta_t \geq \Delta_0 - t$.
3) After time t, it takes $\geq \Delta_t$ units to complete all operations, so $C_{max} \geq t + \Delta_t \geq \Delta_0$.

So to get $C_{max} = \Delta_0$ we must have $\Delta_t = \Delta_0 - t$ and we can take any Δ_t time units after t to complete all operations.

$\Rightarrow$ At time $t+1$, we must have $\Delta_{t+1} = \Delta_t - 1$



To get $\Delta_{t+1} = \Delta_t - 1$, we must pick an $(I_t \cup J_t)$ - perfect matching in G_t

Algorithm:
Starting at time $t=0, 1, \ldots,$
- We pick an $(I_t \cup J_t)$-perfect matching M in G_t
- Run M for 1 time unit (assuming all operation lengths are infinite)
- Update $A_t, \Delta_t, I_t, J_t, t$

This will return a schedule of makespan Δ_0.

2 Q's:
- Why should even an $(I_t \cup J_t)$ ~~...~~ perfect matching exist

What prevents M from being an $(I_{t+1} \cup J_{t+1})$ perfect matching?
- The sets I_t and J_t could change, i.e. $I_{t+1} \neq I_t$ and $J_{t+1} \neq J_t$. (But, $I_{t+1} \supseteq I_t$ and $J_{t+1} \supseteq J_t$)
- An edge of M I_t ~~$\cup J_t$~~ could disappear from G_t