July 12th, 2021

454

Recall:
R||Cmax:
- Given target makespan D, solve

(LP_D)

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

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

$$x \geq 0, \quad x_{ij} = 0 \text{ if } p_{ij} > D$$

Let x be a feasible solution to (LP_D)

Show: x can be rounded to obtain schedule of makespan $\leq 2D$.

Rounding Algo:

→ we can think about these n_i copies as placeholders for jobs (slots)

Define $n_i = \lceil \sum_{j=1}^{n} x_{ij} \rceil \quad \forall i = 1, \ldots, m$

Create bipartite graph w/ a node for each job, nodes (i,c) for all $c = 1, \ldots, m$ and $c = 1, \ldots, n_i$.
Let:
- J = job nodes and
- $C_i = \{(i,c) : c = 1, \ldots, n_i\}$.

Determine edges b/w J and C_i by defining $w_{j,(i,c)} \quad \forall j = 1, \ldots, n \quad \forall c = 1, \ldots, n_i$

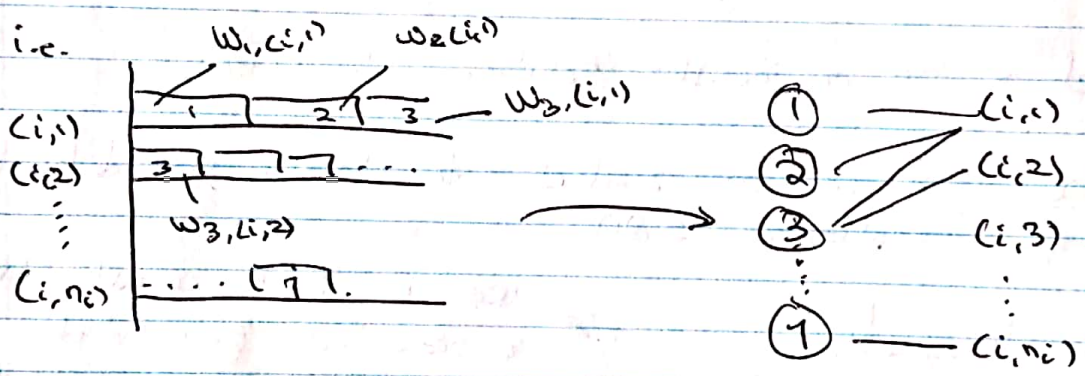
and create an edge $(j, (i,c))$ if $w_{j,(i,c)} > 0$

Claim

How do we define $w_{j,(i,c)}$?

Fix an m/c i. Order jobs $J_i = \{j : x_{ij} > 0\}$ in decreasing p_{ij} order. "Pack" the x_{ij}'s on the n_i-copies $(i,1), \ldots, (i,n_i)$ by considering jobs in this order and packing them on a copy up to boundary T. (McNaughton's Wraparound Rule).

Define $w_{j,(i,c)}$ = amount of j assigned to copy c



Algorithm Cont'd

Find a J-perfect matching in the bipartite graph that we create. If we include edge $(j,(i,c))$ in our matching, that corresponds to scheduling j on m/c i.

Show: Schedule has makespan $\leq 2D$

(We will use $w_{j,(i,c)}$ to show this)

July 12th, 202

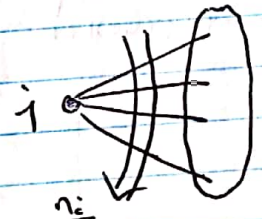
Properties of $w_{k,c}$:

Fix i, $\{w_{j,(i,c)}\}_{\substack{j=1,\dots,n \\ c=1,\dots,n_i}}$ satisfy following properties:

1) $\forall j$, $\sum_{c=1}^{n_i} w_{j(i,c)} = x_{ij}$

$\Rightarrow \{w_{j,(i,c)}\}_{\substack{j=1,\dots,n \\ i=1,\dots,m \\ c=1,\dots,n_i}}$ gives a fractional J-perfect matching in our bipartite graph

i.e.



$\Rightarrow$ each job node sees a total weight of 1

$\sum_{c=1}^{n_i} w_{j(i,c)} = x_{ij}$

And each machine copy (i,c) sees a total weight ≤ 1, since we pack x_{ij}'s up to a bound of 1.

2) By one of our facts on bipartite matching our bipartite graph has a J-perfect matching M.

3) Consider $c > 1$. Consider any j with $w_{j,(i,c)} > 0$ and any k with $w_{k,(i,c-1)} > 0$.
Then $P_{ij} \leq P_{ik}$.
Also if $w_{j,(i,c)} > 0$, then $\sum_k w_{k,(i,c)} = 1$.
So, for any $c > 1$, any j with $w_{j,(i,c)} > 0$:

$$P_{ij} \leq \sum_k w_{k,(i,c-1)} P_{ik}. \quad (*)$$

Finishing up the proof: ...

Claim: For any $i=1,\dots,m$, $\sum_{j:(i,j)\in M} p_{ij} \le 2D$. → Recall M is our 3-perfect matching

under matching M

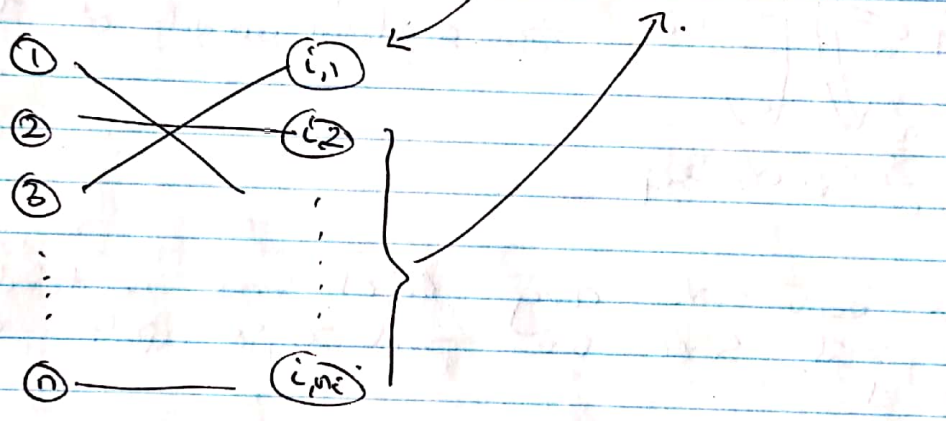
i.e. M contains an edge $(j,(i,c))$ for some $c=1,\dots,n_i$.

Proof:

Total load on $i = \sum_{c=1}^{n_i} \sum_{j:(j,(i,c))\in M} p_{ij}$.

$= \sum_{(j,(i,1))\in M} p_{ij} + \sum_{c=2}^{n_i} \sum_{j:(j,(i,c))\in M} p_{ij}$

i.e.



We want to bound:

$\sum_{(j,(i,1))\in M} p_{ij} + \sum_{c=2}^{n_i} \sum_{j:(j,(i,c))\in M} p_{ij}$

$\le \max_{j:(j,(i,1))\in M} p_{ij}$; $j:(j,(i,1))$ is an edge

iff $w_{j,(i,1)} > 0$.

$\Rightarrow x_{ij} > 0$.

$\Rightarrow p_{ij} \le D$

$\Rightarrow \le D + \sum_{c=2}^{n_i} \sum_{j:(j,(i,c))\in M} p_{ij}$

(1)

Cont'd

Proof: (cont'd)

$$\leq D + \sum_{c=2}^{n_i} \sum_{j:(i,c,j)\in M} P_{ij}$$

Only 1 P_{ij} exists since we have a matching

Using (*), we get:

$$\leq D + \sum_{c=2}^{n_i} \left(\sum_k w_{k,(i,c-1)} P_{ik} \right)$$

$$= \sum_k \sum_{c=2}^{n_i} P_{ik} w_{k,(i,c-1)}$$

$$= \sum_k P_{ik} \sum_{c=1}^{n_i - 1} w_{k,(i,c)}$$

$\leq x_{ik}$ → Since $\sum_{c=1}^{n_i} w_{k,(i,c)} = x_{ik}$, but we are excluding a copy.

$$\leq D + \sum_k P_{ik} x_{ik}.$$

$\leq D$, one of the constraints in (LP_2)

$$\leq 2D.$$

□.

Shop Scheduling:

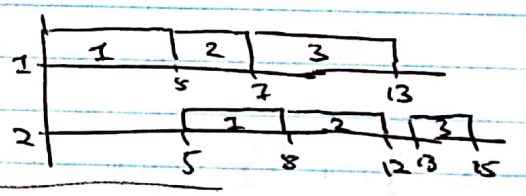
Flow Shop ($F|\cdot|\cdot$)

Each job j consists of m operations where the ith operation has to be processed on m/c i, and takes p_{ij} time. There is a fixed ordering $1, \ldots, m$ of m/c's s.t $\forall$ jobs j, operation 1 must be completed on m/c 1, operation 2 —"— 2, and so on.

Ex:

m/c's \ Jobs	1	2	3
1	5	2	6
2	3	4	2

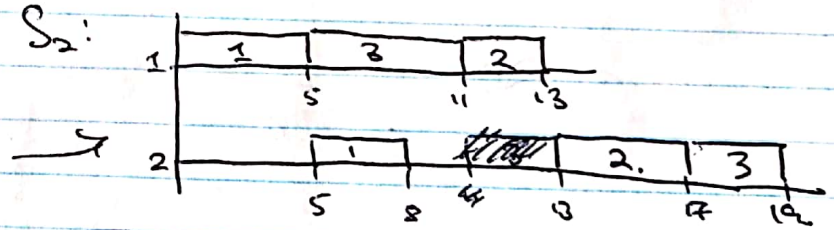
S_1:



$C_{max} = 15$

On different m/c's, Ordering of jobs can be different

S_2:



$C_{max} = 14$

Both S_1, S_2 are feasible.

S_1 is called a permutation schedule. (We follow the same sequence of jobs across different m/c's)

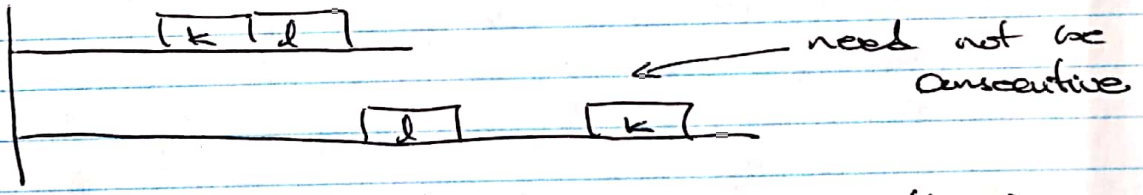
Def'n (Permutation Schedule)
All jobs are processed in same order on all m/c's

F2||Cmax:

Lemma: There is always an optimal schedule that is a permutation schedule.

Proof:
This follows from an interchange argument
Let S^*: optimal schedule that is not a permutation schedule.
Then we can find 2 jobs k, l s.t.
- k comes immediately before l on m/c I.
- l is processed before k (but need not be consecutive)

i.e.

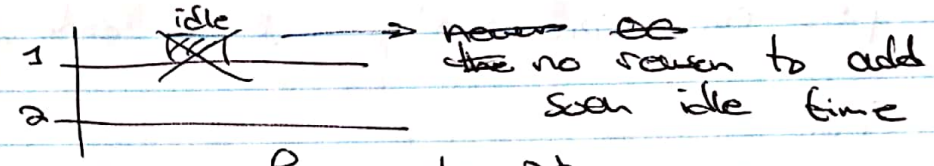


Interchanging l, k on m/c I still gives us a feasible schedule with no greater makespan. So, after a finite # of interchanges, this yields an optimal schedule that is a permutation schedule.

$O(n^2)$.

□

On mc 1, we can always push all idle time towards the end.



(and similarly for mc 2, we can push idle time towards the beginning) → both as important.