Announcements

- Welcome back! I hope you all had a great break!!

  Prelim 2 regrade deadline: Friday 4/10

- Mon/Tue April 6-7 section no quiz, on divide and conquer
- HW8 divide and conquer, due Friday April 10

Plan for the remainder of the semester

- HW9-11 due Fridays April 17, 24, May 1
- Cumulative Final, May 9th
- 1-on-1 appointments possible, see signup here (also posted on Ed)

Dealing with NP-hard problems

optimization

Coping with hardness

- Run algorithm to find true optimum
  may take longer than polynomial time
  see more next week

- Algorithms run in polynomial time
  finds "decent" solution (may not be optimal)
  approximation algorithm, next

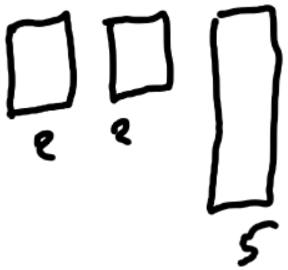
- Find poly time algorithm
  & win $1M

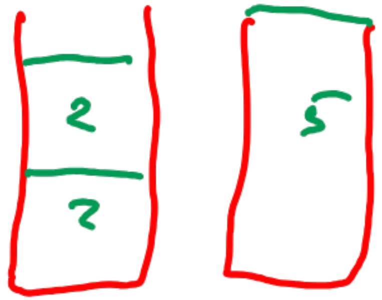
Job scheduling and makespan

m jobs   job j takes time t_j

n machines : assign jobs to machine

$n=2$ jobs times → 2, 2, 5





$A(i)$ = set jobs assigned to machine i

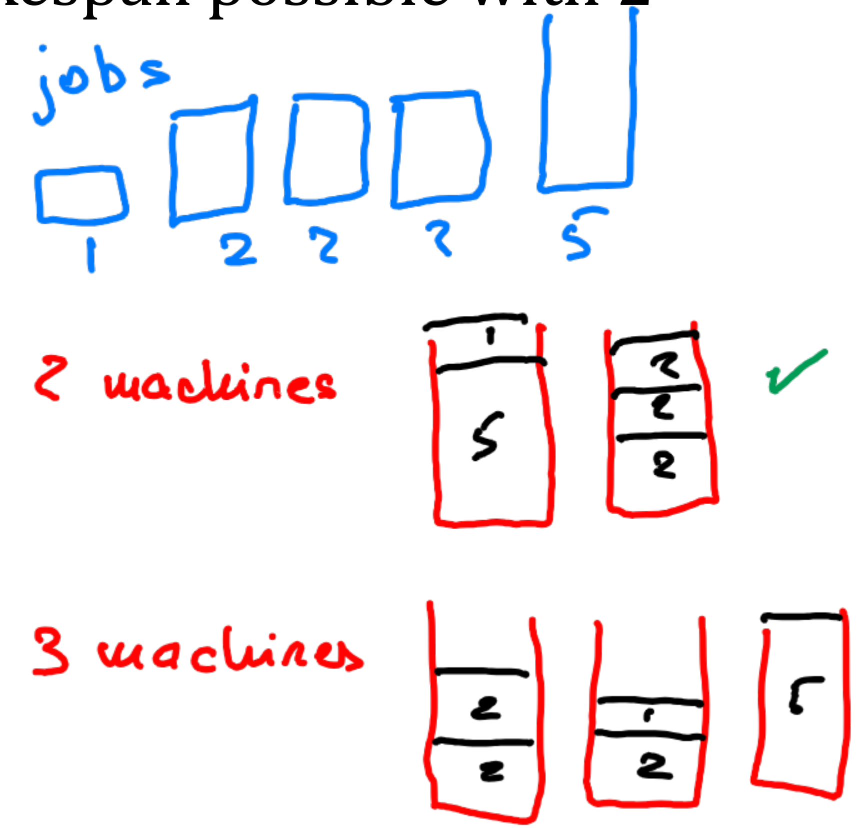
time for machine i $\sum_{j \in A(i)} t_j$

makespan $T = \max_i T_i$

Join by Web **PollEv.com/evatardos772**

5 jobs with times 1, 2, 2, 2, 5. what is the minimum makespan possible with 2 or 3 machines

A. With 2 machines OPT=5, with 3 machines ~~OPT=4~~
B. With 2 machines OPT=6, with 3 machines ~~OPT=4~~
C. With 2 machines OPT=5, with 3 machines OPT=5
D. With 2 machines OPT=6, with 3 machines OPT=5
E. With 2 machines OPT=7, with 3 machines OPT=5



Makespan NP-hard

Partition $\leq_p$ makespan optimization

$w_1 \dots w_n$ & $\exists ?$ subset A $\sum_{i \in A} w_i = \sum_{i \notin A} w_i$

$n = 2$ machine, $t_i = w_i$

Is makespan $\frac{1}{2} \sum_i w_i$ possible?

Greedy algorithm ideas

Sort jobs in decreasing order

$T_i = 0$ for all i

For $j = 1 \dots n$

put job j

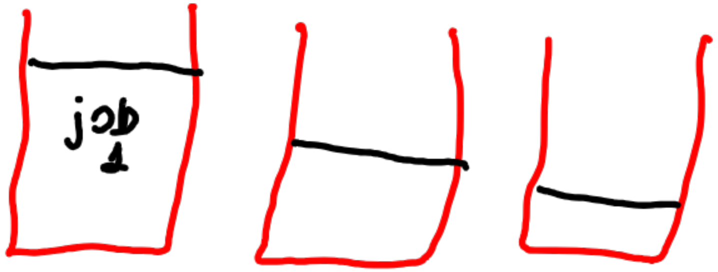
on machine i

least loaded so far } select $i = \arg\min_k T_k$

$T_i = T_i + t_j$

jobs times $t_1 \dots t_m$

machines



Plan to prove guarantee on makespan

Recall Opt = min possible makespan is NP-hard

T_{alg} = makespan of algorithms result

Goal: $T_{alg} \overset{?}{\leq} Opt$

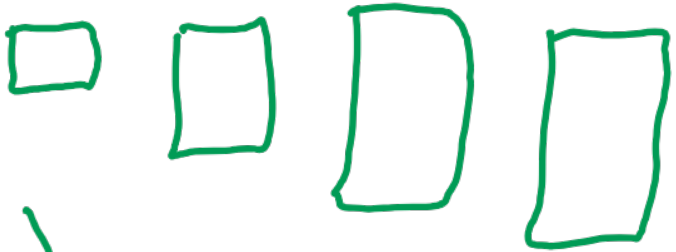
need: lower bound $X \leq Opt$

prove $T_{alg} \leq 2X$

implies $T_{alg} \leq 2X \leq 2\,Opt$

What we know about optimal makespan

job $t_1 \cdots t_m$



1. $Opt \geq \frac{1}{n} \sum_{j=1}^{m} t_j$ (equal if possible to equally divide)

2. $Opt \geq \max_j t_j$

Hope: can we compare greedy outcome to these bounds

Approximation guarantee

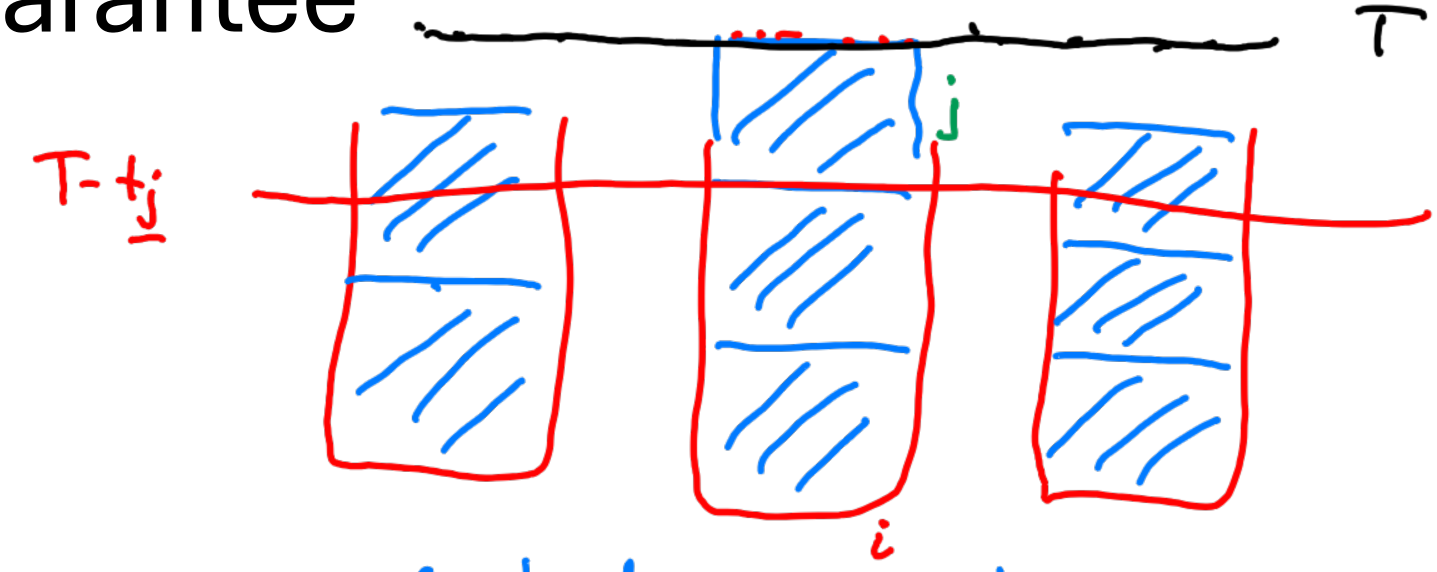
$T = \max_i T_i$

assume machine i is most loaded

consider last job j on machine i

$\Rightarrow \quad T_k \geq T - t_j$

all machines k



Sort decreasing t_j

in this order

add job to least loaded machine

two bounds we know

(2) $\text{Opt} \geq \max_j t_j$ (1) $\text{Opt} \geq \frac{1}{n} \sum_j t_j$

Approximation guarantee

Using these:

we know $T_k \geq T - t_j$ all k *

$$\sum_{k=1}^{n} T_k = \sum_{k} \sum_{\ell \in A(k)} t_\ell = \sum_{\ell} t_\ell$$

summing

$$\sum_{k} T_k \geq nT - nt_j$$

$$\Rightarrow \quad \sum_{\ell} t_\ell \geq nT - nt_j$$

$$\Rightarrow \quad T \leq \frac{1}{n} \sum_{\ell} t_\ell + t_j \leq \mathrm{Opt} + \mathrm{Opt}$$

bounds (1) + (2)

Theorem: Greedy guarantee $\leq 2\,\mathrm{Opt}$ 2-approximation

Approximation guarantee

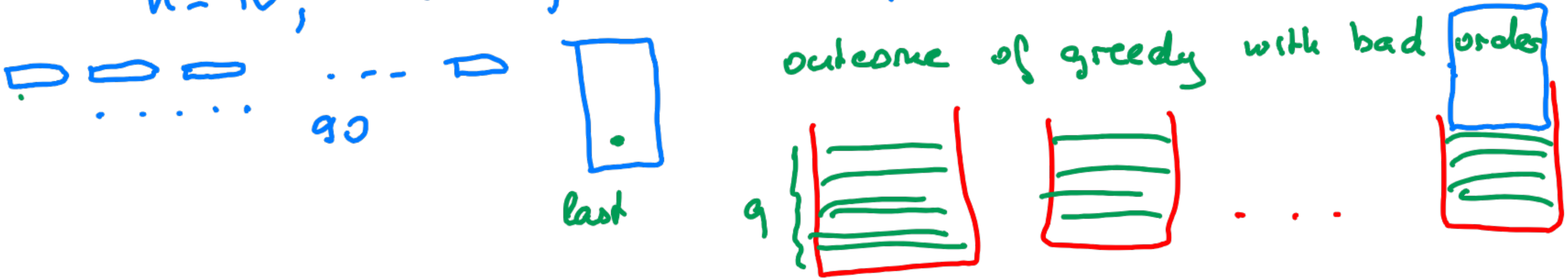
Observe: (i) our proof did not use sorting

without sorting 2 is best possible

bad example

job : time $t_1 = 10$

$n = 10$, other jobs size 1, 90 of these



Without sorting 2-approx is best bound for our algorithm