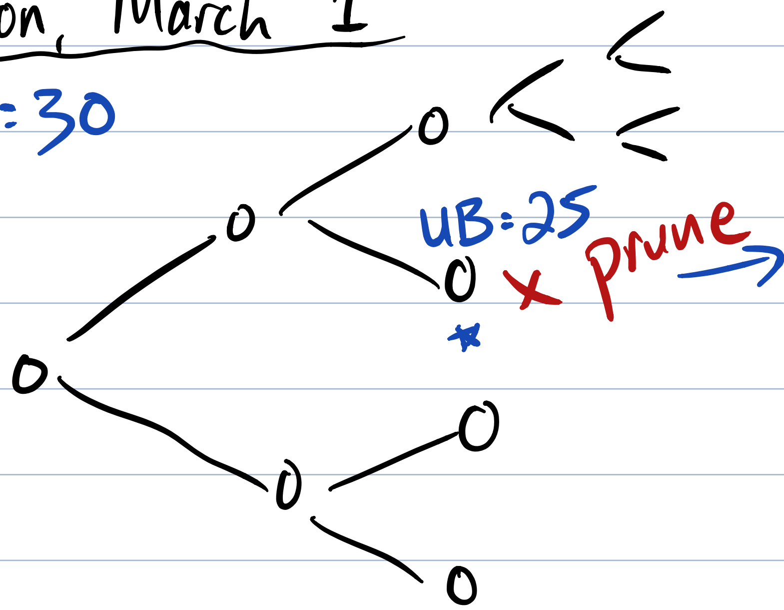
Mon, March 1

sol = 30



I don't know how good we can do from here, but we def. can't beat 25.

Mathematical Foundations

(1) "making decisions to build partial solutions"

⇒ really splitting the search space into disjoint parts (subspaces)

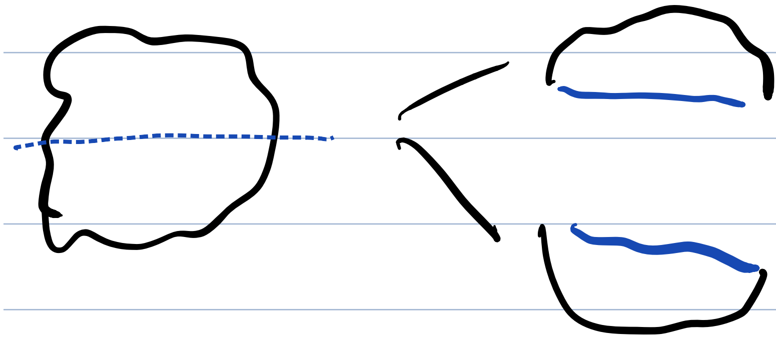
↳ no overlap

Ex: Knapsack - Item 1 is in or out

{all subsets of items}

→ {subsets containing 1}

and {subsets not containing 1}



{subsets containing 1}
↳ {subsets containing 1 and 2}
and {subsets containing 1 and not containing 2}

This is called <u>branching</u>.

(2) For any subspace S we create, we need to be able to compute bound(S), some upper bound on the best score possible for any solution in S.

<u>Notes</u>: * We're phrasing this all for maximization.
* bound(S) has to be an <u>upper</u> bound. Lower bounds are easy but useless.

<u>Ex</u>: Problem #5 : Job Assignment
You have n tasks that need to be done and n workers. Each worker can do 1 task. Each task has a different cost to complete depending on which

worker does it. Goal: Assign workers to tasks, minimizing cost.

		tasks		
workers	1	2	3	4
A	3	5	2	2
B	6	8	10	8
C	2	6	4	9
D	10	4	7	5

greedy?
task 1: pick cheapest
2: cheapest rem.
$2+4+2+8=16$
$2+6+4+4=16$

Search space: All assignments of workers to tasks.

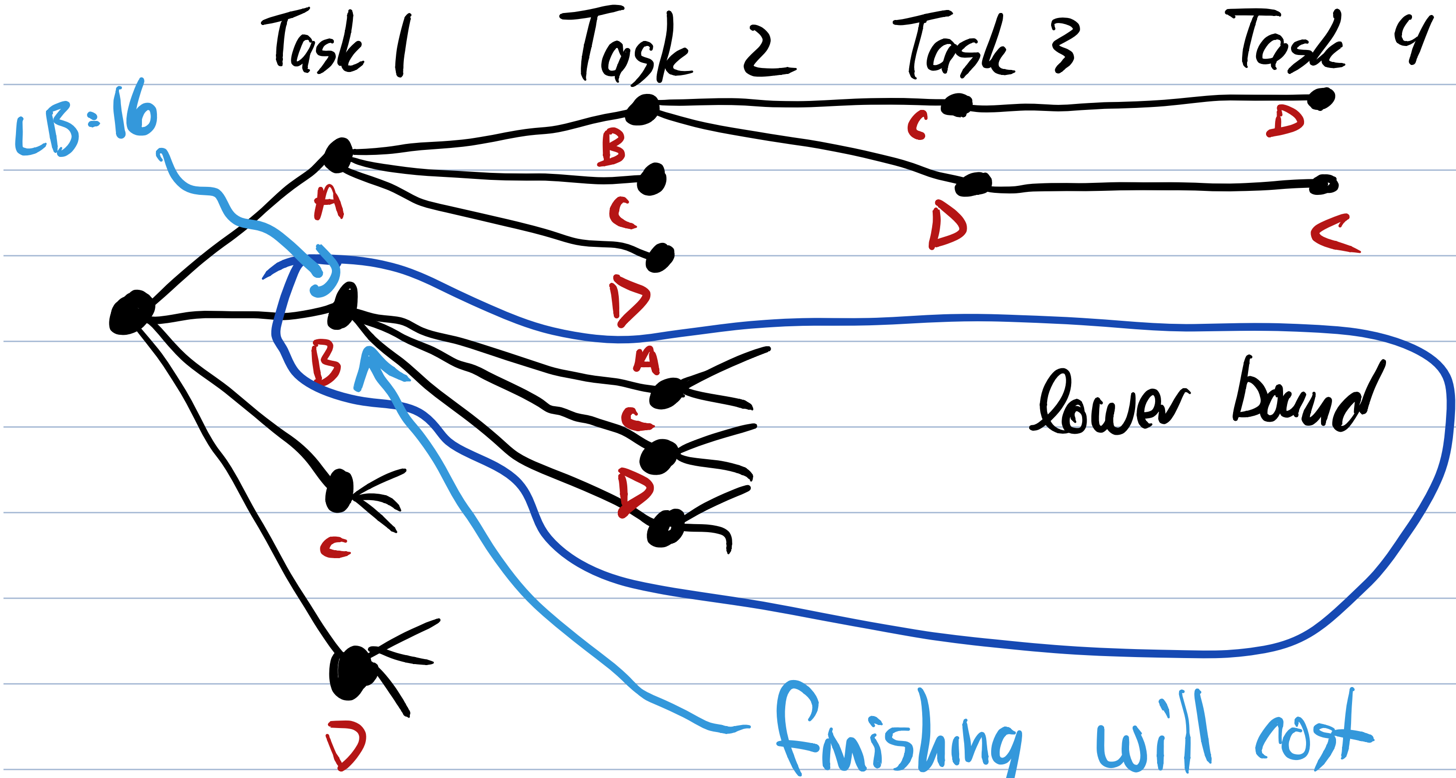
How big? Worker A: 4 tasks
Worker B: 3 tasks
Worker C: 2 tasks
Worker D: 1 task

$\underbrace{}_{24}$

In general: $n! = n\cdot(n-1)\cdot(n-2)\cdots3\cdot2\cdot1.$
$(\sim n^n)$

No constraints! Backtracking = Brute Force
(1) Branching (2) Bound
↳ Pick which worker does a certain

task.



Bounding

* Suppose you've already picked worker B to do task 1.

* What is a lower bound on the best you can finish? (Has to be easier than actually solving the whole problem!)

	1	2	3	4	
A	3	5	2	2	2
B	6	3	10	8	
C	2	6	4	9	+ 4
D	10	4	7	5	+ 4
	8 = 4 + 2 + 2				10

One possibility: every task will cost

at least the minimum remaining cost of available workers.

$8

Alternate: every worker will incur a cost at least their minimum remaining $10

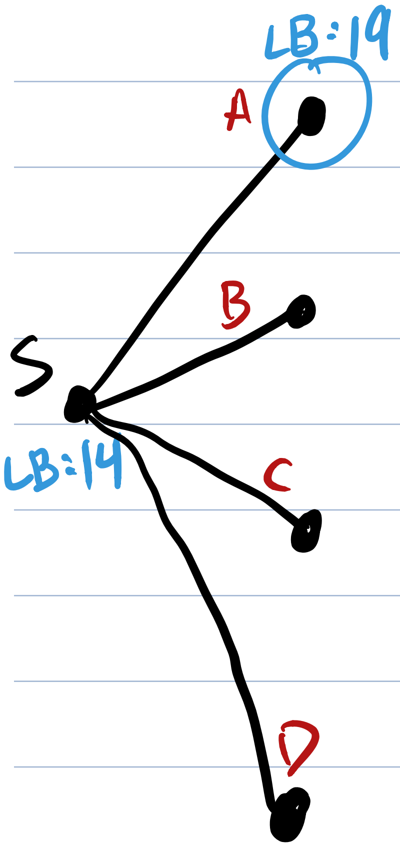
So, finishing will cost at least 10.

Our lower bound is:

max(sum of smallest cost in each remaining row,
sum of smallest cost in each remaining col)

$(= \max(10, 8) = 10)$

best sol: ∞



13 = 4 4 5

	1	2	3	4
A	3	~~5~~	~~2~~	~~2~~
B	~~6~~	8	10	8
C	~~2~~	6	4	9
D	~~10~~	4	7	5

$$\begin{array}{r} 8 \\ 4 \\ 4 \\ \hline 16 \end{array}$$

max(16, 13) = 16