Wednesday, March 1, 2023
Lecture #19
MSSC 6000

## Announcements

* HW 3 due Wed, March 8, 11:59pm
* Office Hours today are cancelled
* Makeup office hours tomorrow (Thursday), 2pm-3pm on Teams (same link)
* Midterm Exam, Wed, March 8 in class
* Friday March 10 schedule? ↳ 10am - 10:50am
* PEP 8 Song

## Topic 8 - Branch and Bound

Recall that our problems usually have two considerations:

(1) Constraints that <u>must</u> be satisfied

ex: capacity of the knapsack
no conflicts of meetings in WIS

(2) A value/score that we want to maximize or minimize among all candidates in the search space that satisfy the constraints.

Some problems are only about constraints
* Sudoku
* NFL scheduling

Some problems don't have constraints and are only about scores – depends on how you define your search space
(minimum spanning tree)

Backtracking boils down to:
* If you build your solutions a bit at a time, you can detect early if the constraints are violated and rule out a chunk of the search space all at once.

This never considered value/score.

Branch and Bound is just backtracking with an extra way to rule out a partial solution.
(Assume maximization for now.)

* If I've already seen a complete solution with a score of X, and the partial solution I'm now building has no way of being completed that beats a score of X, then prune it (stop expanding that partial solution).

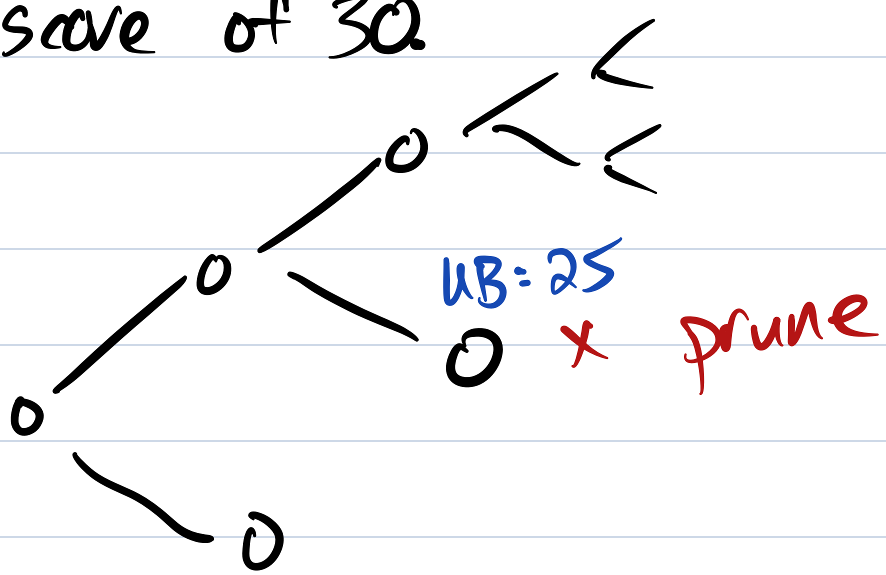
There's no way to know exactly the best score you can do on completing a partial solution — if you could do that quickly, just do it and you've solved the problem.

Need: A way to get an upper bound on the best you could do when completing a given partial solution.

"I don't know how good I can do, but I know for sure I can't do better than X."

## Generic Picture

Have in hand a solution with a score of 30.




Hard part: how to compute an upper bound

## Ex: Job Assignment Problem

You have n tasks that need to be done and n workers. Each task has a different cost to complete depending on which worker does it. Each worker can do 1 task. Goal: minimize total cost.

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

4 · 3 · 2 · 1

Many applications:
→ Drivers picking up passengers
→ Shipments from mines to factories

* Search Space: All assignments of workers to tasks.

How big? $n!$ $(4! = 4 \cdot 3 \cdot 2 \cdot 1 = 24)$

Constraints? None, every candidate is valid.
Backtracking is useless (equivalent to brute force)

Two things to describe

(1) Branching (2) Bounding

↓

how we're going to build the partial solutions

* Pick which worker does a certain task