Announcements:
1. Midterm 2 on April 3, Tuesday
2. Conflict " " " 8, Monday
   (Form fill up deadline today at 2 pm)

Greedy Alg.:

For solving optimization problems.

Incrementally build set.
Making "local" optimal choice/greedy choice
in every step/iteration.

Adv: Simple & Fast

Disadv: They can be wrong!
Post & correctness is needed.

Ex.1: Interval / Job scheduling.

Given n intervals (jobs) \([s_1, t_1], [s_2, t_2], \ldots, [s_n, t_n]\)

\(start\ time \leftrightarrow finish\ time\).
Find maximum # non-overlapping intervals.

Example:

\[ \text{OPT: 3} \]

Idea 1: Pick job of earliest start time

\{3,1,4\} fails!

Idea 2: Pick job that has min # intersections

\{5,2,13\} yay!

Idea 3: Pick smallest job fails!

Idea 4: Earliest end time - works!
Greedy Alg'm:

1. repeat
2. pick \([s_i, t_i]\) of smallest \(t\) among
   the intervals left. Add this to the sol'n
   0(\(n\))
3. resume all intervals intersecting \(\{s_i, t_i\}\)
4. until no intervals left.

Running Time: \(O(n^2)\)

Better Run time: Sort in ascending order
    end times \(O(n \log n)\)
    \(+ \ O(n) \text{ work.} \)
    \(= O(n \log n). \)

Proof of correctness:

\(I^*\) be \(I\).

let \(I^*\) be an opt. sol'n (unknown)

let the first interval in \(I^*\) be \([s^*, t^*]\)

let the first interval in \(I\) be \([s_i, t_i]\).

We know that (by choice of greedy Alg'm)

\(*\quad *\quad *\quad *\quad *\quad *\quad *\quad \ldots\)
\[ I^* : \quad \therefore \quad \text{do not intersect} \]

\[ \Rightarrow [s_i, t_i] \text{ does not intersect any interval in} \]

\[ \bigcup_{i=1}^{k^*} [s_i, t_i]^* \]

\[ i^* \leftarrow i^* \setminus \{[s_i, t_i]^*\} \cup \{[s_i, t_i]\} \]

is

a feasible soln. \quad |i^*| = |i^*|

(exchange argument) & it is also optimum.

Remove \([s_i, t_i]\) & all its intersecting intervals ↓

& Repeat  → Solution.

Red: Doesn't work on weights. (DP).

**Ex.2:** Job scheduling to minimize the total wait time.

Given \(n\) jobs with processing times \(P_1, P_2, P_3, \ldots, P_n \geq 0\)

Want to find an ordering of the jobs
want to find an ordering of the jobs that minimized the total wait time.

\[ \text{cost} = 0 + p_1 + (p_1 + p_2) + \ldots + (p_1 + p_2 + \ldots + p_n) \]

Ordering that minimizes the cost.

<table>
<thead>
<tr>
<th>Jobs</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>( c(i) )</td>
<td>3</td>
<td>4</td>
<td>1</td>
<td>8</td>
<td>2</td>
</tr>
</tbody>
</table>

cost \((3, 4, 1, 8, 2)\) = \(0 + 3 + (3+4) + (3+4+1) + (5+4+1+8)\)

\[ = 34 \]

cost \((3, 4, 1, 2, 8)\) = \(0 + 3 + (3+4) + (3+4+1) + (5+4+1+2)\)

\[ = 28 \]

cost \((3, 1, 4, 2, 8)\) = \(0 + 3 + \ldots \)

\[ = 25 \]

I claim: Order in increasing order
Greedy Algorithm: Order in increasing order as processing time.

\[ \text{cost}(1, 2, 3, 4, 5, 8) = 0 + 1 + (1+2) + (1+2+3) + (1+2+3+4) = 20 \]

Correctness Proof:

Let \( p_1^*, p_2^*, \ldots, p_i^*, \ldots, p_n^* \) is the optimal ordering.

\[ p_1^* \text{ in sorted order} \Rightarrow \exists i: p_i^* > p_{i+1}^* \]

\[
\text{opt-cost} = 0 + p_1^* + (p_1^* + p_2^*) + \ldots + (p_1^* + p_2^* + \ldots + p_i^* + p_{i+1}^*)
\]

Consider new ordering: \( p_1^*, p_2^*, \ldots, p_i^*, p_1^*, p_2^*, \ldots, p_n^* \)

New cost: \[ 0 + p_1^* + (p_1^* + p_2^*) + \ldots + (p_1^* + p_2^* + \ldots + p_i^* + p_{i+1}^*) + (p_1^* + p_2^* + \ldots + p_{i-1}^* + p_i^*) \]

\[ \text{opt-cost} - \text{new-cost} = p_i^* - p_{i+1}^* > 0 \quad (\because p_i^* > p_{i+1}^*) \]

\[ \Rightarrow \text{opt-cost} > \text{new-cost} \]

Contradiction to optimality of \( p_1^*, \ldots, p_n^* \)
Extends to weighted problem:

We are also given \( w_i \geq 0 \) \( \forall i \) together with \( p_i \) (processing time).

\[
\text{cost} = w_1 (p_1) + w_2 (p_1 + p_2) + w_3 (p_1 + p_2 + p_3) + \ldots + w_{n-1} (p_1 + \ldots + p_{n-1})
\]

Find ordering that minimizes the cost.

Intuition:
- Increasing order \( \mathcal{O} \) \( p_i \)
- Decreasing order \( \mathcal{O} \) \( w_i \) (weight/priority)

Greedy Alg:\m: Increasing order \( \mathcal{O} \) \( \frac{p_i}{w_i} \)

Correctness pf: (Similar to unweighted)
Opt: \( P_1^*, P_2^*, \ldots, P_i^*, P_{i+1}, \ldots, P_n^* \)

\[ \frac{P_i^*}{w_i^*} > \frac{P_{i+1}^*}{w_{i+1}^*} \]

After swap: \( P_1^*, P_2^*, \ldots, P_i^*, P_{i+1}, P_1^*, P_2^*, P_{i+2}, \ldots, P_n^* \)

Opt-cost: \( w_i^*(0) + w_i^*(P_i^*) + \ldots + w_{i+1}^*(P_i^* + \ldots + P_{i+1}) - w_{i+1}^*(P_i^* + \ldots + P_{i+1}) + w_{i+2}^*(P_i^* + \ldots + P_{i+1}) + \ldots \)

New-cost (after swap):

\[ w_i^*(0) + w_i^*(P_i^*) + \ldots + w_{i+1}^*(P_i^* + \ldots + P_{i+1}) + w_i^*(P_i^* + \ldots + P_{i+1}) + w_{i+2}^*(P_i^* + \ldots + P_{i+1}) + \ldots \]

Opt-cost - New-cost = \( w_{i+1}^* P_i^* - w_i^* P_{i+1}^* > 0 \)

\[ \frac{P_i^*}{w_i^*} > \frac{P_{i+1}^*}{w_{i+1}^*} \]