Tuesday, April 5, 2022 10:50 AM

Greedy Algorithms:

To solve some optimization problems,
incrementally building sol^n
at each step, what seems best "locally".

Adv: Simple & fast

Disadv: May not be correct
If correct, then needs a proof.

Ex 1: Interval scheduling.

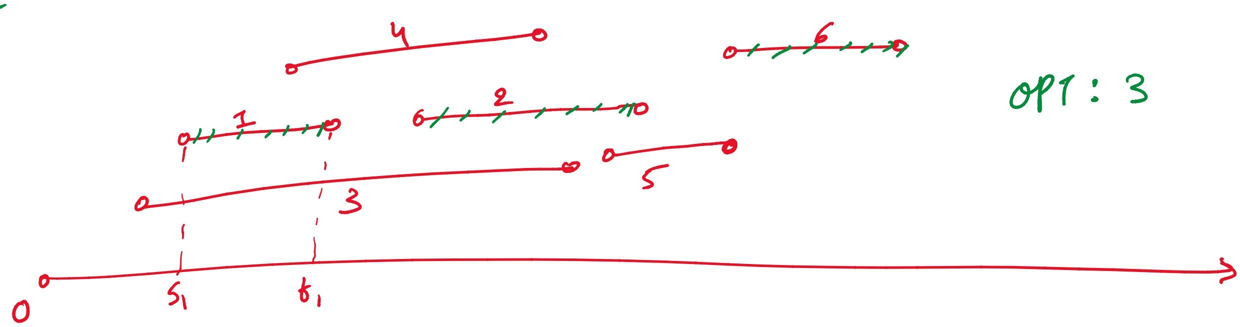
Given n intervals $[s_1, f_1]$ $[s_2, f_2]$, . . . , $[s_n, f_n]$

start time job 1 → finish time

Find the largest subset of non-overlapping intervals.

↓ maximum # of jobs

eg.



OPT : 3

idea 1 (Greedy): Pick job w/ earliest start time ($\min_i s_i$)

$\{3, 5\} \rightarrow 2$ jobs fails!

idea 2 (Greedy): Pick the smallest job ($\min_i (f_i - s_i)$)

$\{5, 1\}$ fails!

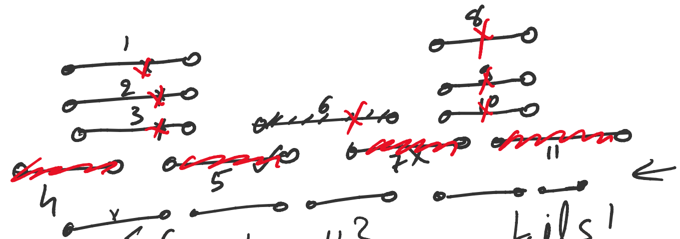
$\{5, 1\}$ fails!

idea 3 : Pick the job that overlaps w/ minimum # of other jobs.

$\{6, 2, 1\}$ yay!

OPT : 4

Greedy : 3



$\{6, 4, 11\}$ fails!

idea 4 : Pick job w/ earliest finish time ($\min_i t_i$)

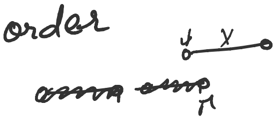
bingo! WORKS.

Greedy Alg'm : → Greedy sol'n.

1. repeat {
2. Pick $[s_i, t_i]$ w/ smallest t_i.
3. Remove $[s_i, t_i]$ & all intervals overlapping with it.
4. } Until intervals left.
5. Output picked intervals.

Running Time : Naive $O(n^2)$

Better: Sort in increasing order of t_i. 1 Scan.



$O(n \log n)$.

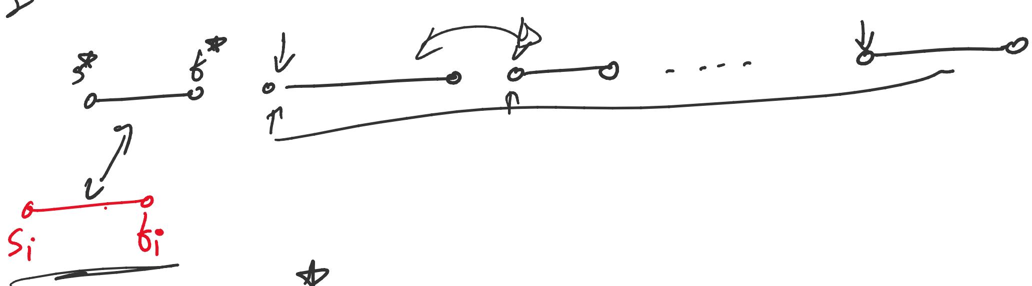
Correctness Proof: $I = \{[s_1, t_1], \ldots, [s_n, t_n]\}$

OPT sol'n (unknown).

Correctness Proof: $I = \{[\ldots]\}$

→ Let $I^* \subseteq I$ be an opt sol'n (unknown).
Let $[s^*, f^*]$ be the left most interval in I^*.
Let $[s_i, f_i]$ be the first interval picked by Greedy Alg'm.

I^*:



Know: $f_i \le f^*$

$$I^* - \{[s^*, f^*]\} \cup \{[s_i, f_i]\}$$ is feasible.

AND has the same # intervals as I^*.

exchange argument. → Reset $I^* \leftarrow I^* - \{[s^*, f^*]\} \cup \{[s_i, f_i]\}$

is an optimal sol'n.

Remove $[s_i, f_i]$ & all intervals overlapping with it.
smaller instance
Repeat the argument → Induction. □

idea 5 : Pick latest start time. $(\max_i s_i)$.

Does not extend to weighted.

Rmk: Does not extend to weighted.
(use DP).

Ex2: Job scheduling to minimize average wait time.

Given n jobs of processing times $P_1, P_2, \ldots, P_n$.
Find an ordering that minimizes total wait time.

ordering: $P_1, P_2, \ldots, P_n$

$$\text{cost} = \underline{0} + P_1 + (P_1+P_2) + \cdots + (P_1+P_2+\cdots+P_{n-1})$$
(total wait time) $= \underline{(n-1)}P_1 + (n-2)P_2 + \cdots + (n-i)P_i + \cdots + P_{n-1}$

jobs	1	2	3	4	5
Process Time	3	4	1	8	2

order: 3, 4, 1, 8, 2

$$\text{cost} = 0 + 3 + (3+4) + (3+4+1) + (3+4+1+8)$$
$$= 34$$

↓

better order: 3, 4, 1, 2, 8

$$\text{cost} = 0 + 3 + \underline{(3+4)} + (3+4+1) + (3+4+1+2)$$
$$= 28$$

↓

even better: 3, 1, 4, 2, 8

$$\text{cost} = 0 + 3 + \underline{(3+1)} + (3+1+4) + (3+1+4+2)$$
$$= 25$$

⋮

best: 1, 2, 3, 4, 8

$$\ldots + (1+2) + (1+2+3) + (1+2+3+4)$$

best: 1, 2, 3, 4, 8

cost $= 0 + 1 + (1+2) + (1+2+3) + (1+2+3+4)$

$= 20$

Greedy Alg'm: Order in increasing p_i.

Correctness Proof:

Let $\underline{p_1^*, p_2^*, \dots, p_n^*}$ be the optimal order.

Suppose it is not sorted.

Then $\exists i: \; p_i^* > p_{i+1}^*$

Swap i & $i+1$ to get a new order. $\left(p_1^*, \dots, p_{i-1}^*, p_{i+1}^*, p_i^*, p_{i+2}^*, \dots, p_n^*\right)$

old cost (opt cost) $= \cancel{0} + \cancel{p_1^*} + (p_1^* + p_2^*) + \dots + (\cancel{p_1^* + p_2^* + \dots + p_{i-1}^*}) +$
$\underline{(p_1^* + p_2^* + \dots + p_{i-1}^*} + \underline{p_i^*}) + (p_1^* + p_2^* \cancel{+ \dots + p_{i-1}^* + p_i^*} + p_{i+1}^*) + \dots$
$\dots + (p_1^* \cancel{+ \dots + p_{n-1}^*})$

New cost (after swap) $= \cancel{0} + \cancel{p_1^*} + \dots + (p_1^* \cancel{+ \dots + p_{i-1}^*}) +$
$(\underline{p_1^* + \dots + p_{i-1}^*} + \underline{p_{i+1}^*}) + (p_1^* + \cancel{\dots + p_{i-1}^* + p_{i+1}^*} + p_i^*) + \dots$
$+ (p_1^* \cancel{+ \dots + p_{n-1}^*})$

New cost $-$ old cost $= p_{i+1}^* - p_i^* < 0 \quad (\because p_i^* > p_{i+1}^*)$

$\Downarrow$

we get a better sol'n than OPT!

Contradiction. $\square$

Note: Works for weighted version.

w_i for job i.

Note: Works for weighted version.

also, Given weight w_i for job i.

want to minimize weighted total wait time

$$cost = w_1 0 + w_2(P_1) + w_3(P_1+P_2) + \cdots + w_n(P_1 + \cdots + P_{n-1})$$

Greedy strategy: want increasing P_i
& decreasing w_i

Sort in increasing order of $\cancel{P_i - w_i}$
P_i / w_i ✓

Correctness Pf: (similar)

OPT: $P_1^*, P_2^*, \ldots, P_i^*, P_{i+1}^*, \ldots, P_n^*$

after swap: $P_1^*, P_2^*, \ldots, P_{i+1}^*, P_i^*, P_{i+2}^*, \ldots, P_n^*$

$$\text{Old cost}: \cancel{w_1^* 0} + \cdots + w_i^*(\cancel{P_1^* + \cdots + P_{i-1}^*}) + \underline{w_{i+1}^*}(\underline{\cancel{P_1^* + \cdots + P_{i-1}^*}} + \underline{P_i^*}) + \cancel{w_{i+2}^*(P_1^* + \cdots + P_i^* + P_{i+1}^*)} + \cdots + \cancel{w_n^*(P_1^* + \cdots + P_{n-1}^*)}$$

$$\text{New cost}: \cancel{w_1^* 0} + \cdots + w_{i+1}^*(\cancel{P_1^* + \cdots + P_{i-1}^*}) + w_i^*(\underline{\cancel{P_1^* + \cdots + P_{i-1}^*}} + P_{i+1}^*) + \cancel{w_{i+2}^*(P_1^* + \cdots + P_{i+1}^* + P_i^*)} + \cdots + \cancel{w_n^*(P_1^* + \cdots + P_{n-1}^*)}$$

$$\text{New cost} - \text{old cost} = w_i^* P_{i+1}^* - w_{i+1}^* P_i^* < 0$$

$$\Uparrow$$

$$\frac{P_i^*}{w_i^*} > \frac{P_{i+1}^*}{w_{i+1}^*} \qquad \square$$

$$\frac{r_i}{w_i^*} > \frac{r_{i+1}}{w_{i+1}^*} \qquad \square$$