MSSC 6000
Feb 4, 2022

Announcements:
→ HW 1 due next Wed on D2L

Lecture 3 - Greedy Algorithms (continued)

Throughout the course we're going to learn about a catalogue of problems that model all kinds of real world problems that you might face.

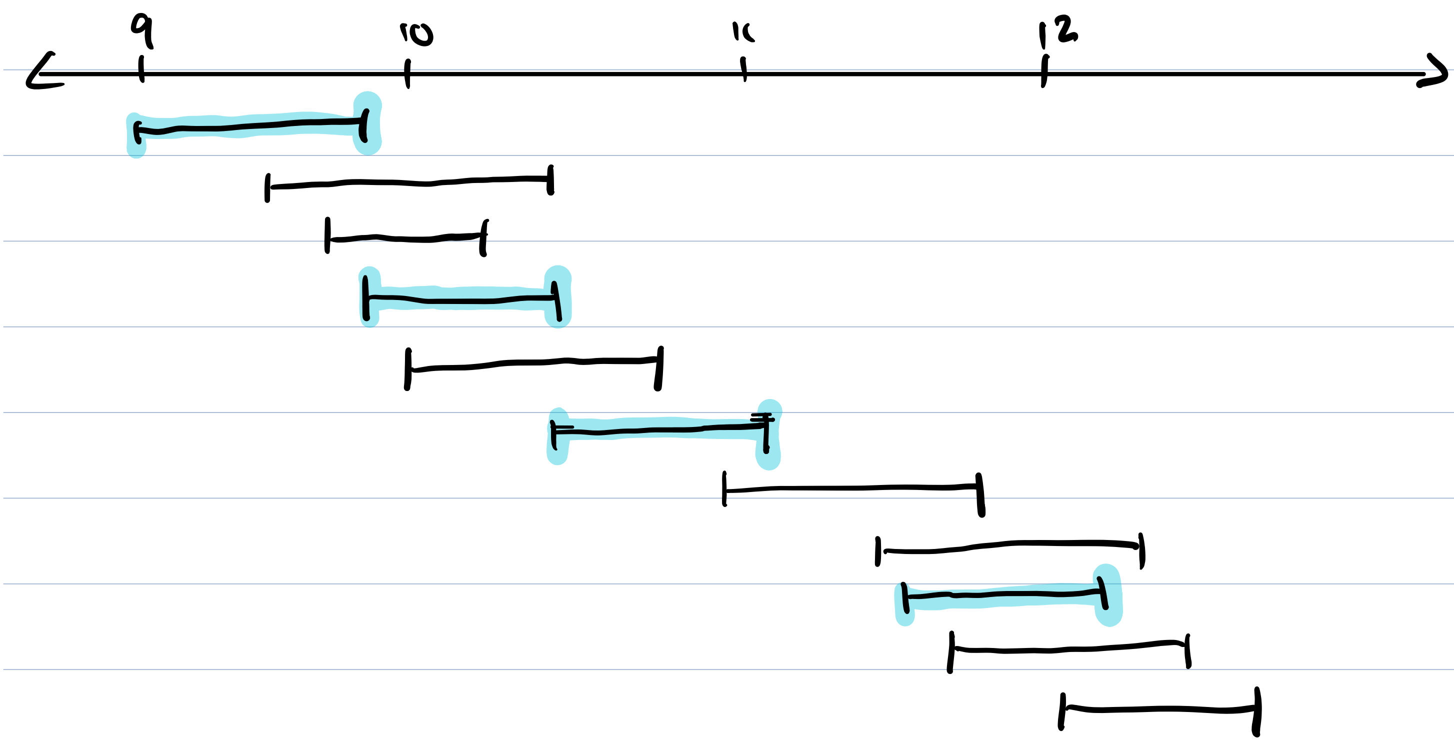
Problem #1: Interval Scheduling
(Algorithm Design, by Kleinberg and Tardos)

Suppose you are in charge of a conference room that a lot of people want to use to hold meetings. A bunch of people tell you the times they want to book the room for, and your goal is to accomodate as many groups as possible.

We want to maximize the # of bookings. (2)

Ex: **Reservations:**

9am - 9:50am	10:30 - 11:15
9:30 - 10:30	11:00 - 11:50
9:45 - 10:15	11:30 - 12:15
9:50 - 10:30	11:35 - 12:10
10 - 10:50	11:40 - 12:20
	12:00 - 12:30



Maximize the # of meetings. (4)

Formal setup:

- n requests
- each request has a start time s_i and a finish time f_i (real #s)

and $s_i < f_i$.

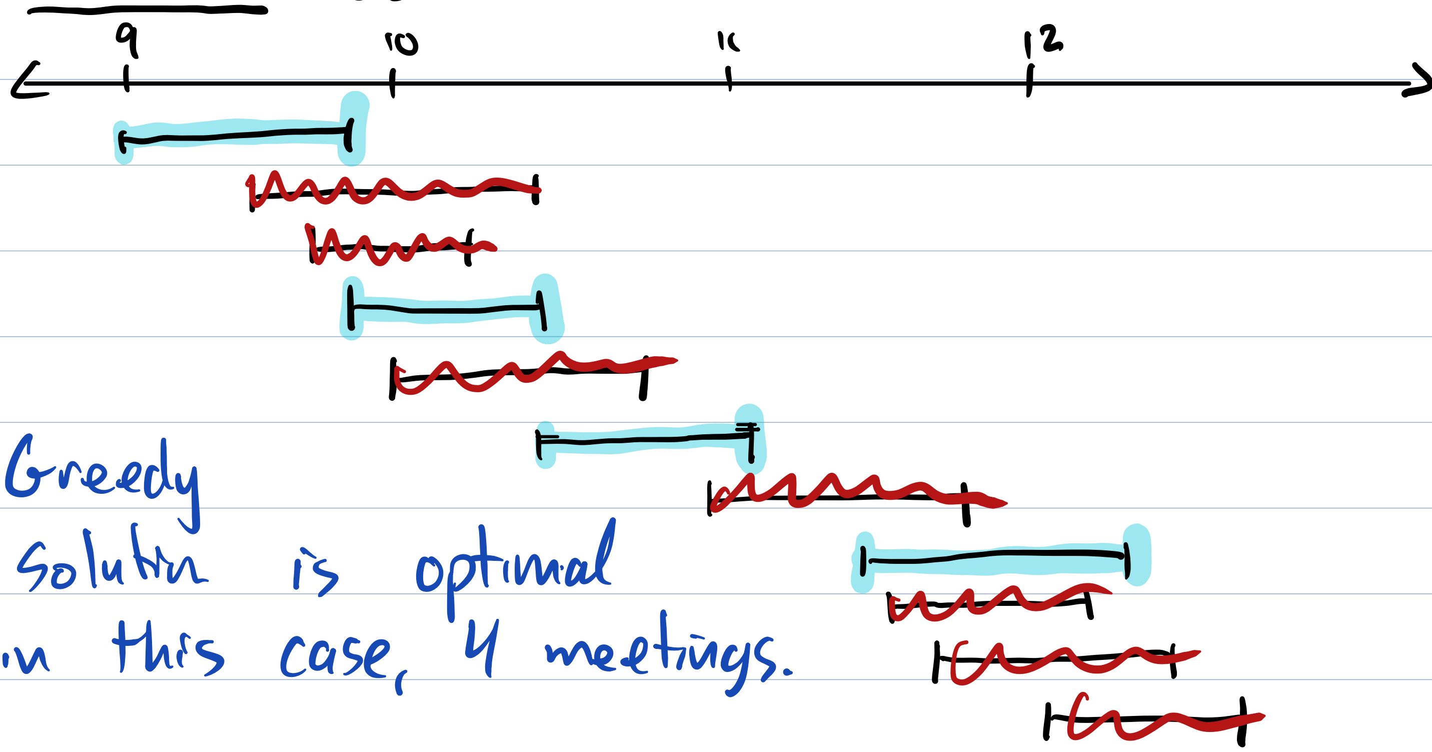
Goal: Find a <u>maximal</u> size subset of non overlapping requests (if requests i and j are both chosen, then: $f_i \le s_j$ or $f_j \le s_i$)

Let's think about possible greedy approaches.

General idea:

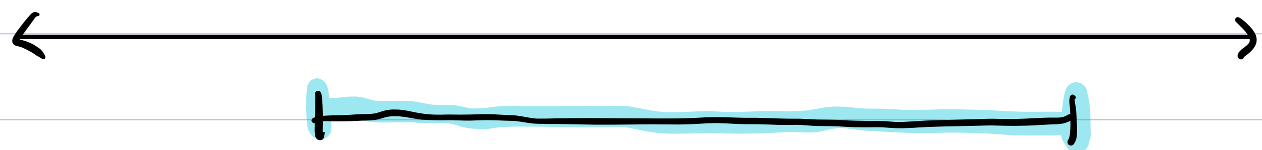
* decide on a rule for which meeting is "best"
* pick it, eliminate conflicts
* repeat until no meetings are left

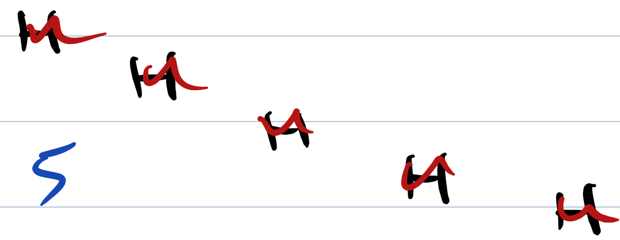
<u>Idea #1:</u> best = earliest start time



Greedy solution is optimal in this case, 4 meetings.

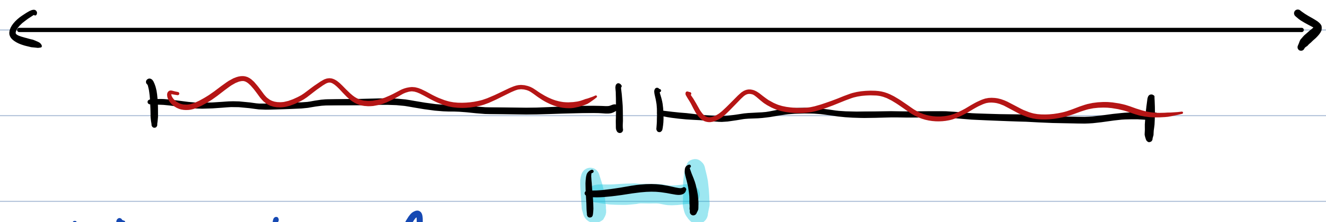
Is it always optmal? Can we break it?





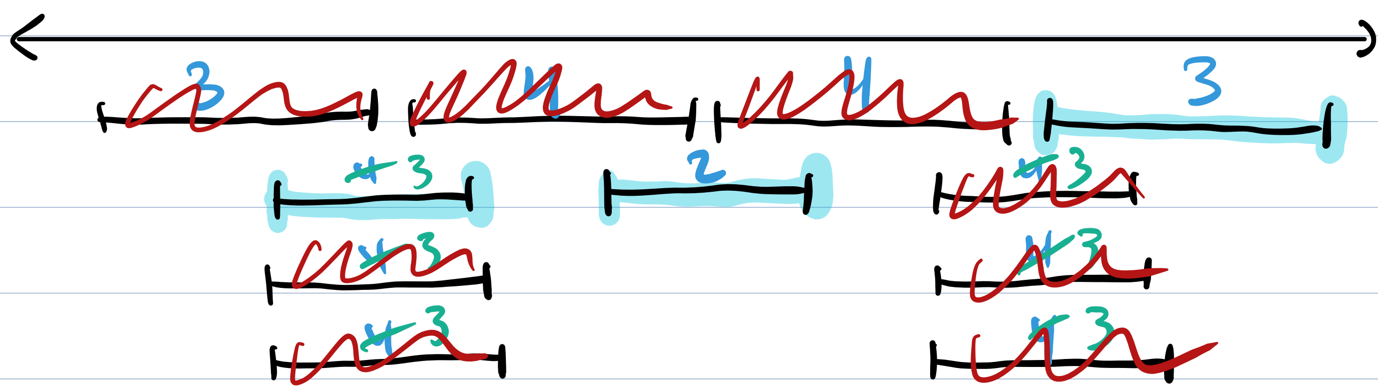
Not optimal
1, instead of 5

Idea #2: best = "shortest"



Not optimal.
1, instead of 2.

Idea #3: best = "least conflicts"



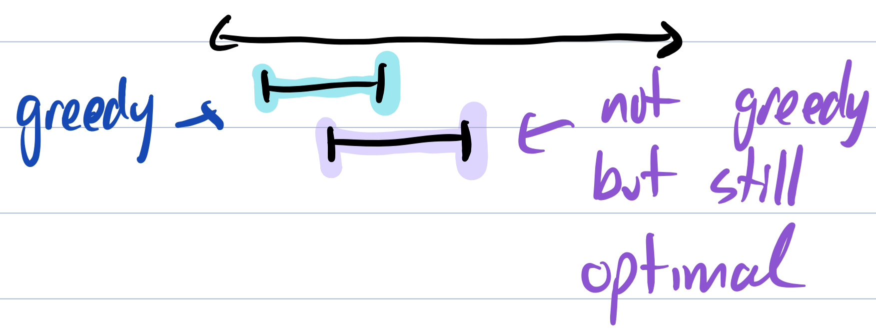
sol = 3

Not optimal, 3 instead of 4.

Idea #4: best = "earliest end time"
Can we break it? No.
This greedy algorithm is guaranteed to give an ~~the~~ optimal solution.

Algorithm:
Let R be the set of requests.
Let A be the empty set.
while R is nonempty:
    Find the request with the earliest end time
    Add it to A
    Remove it from R and remove all other requests that conflict with it
A is the solution.

Theorem: The greedy alg described above produces an optimal solution.



A common strategy when proving that your greedy algo. is optimal is showing that the answer it produces <u>stays ahead</u> of any optimal solution.

<u>Proof</u>: Let R be a set of requests.
Let A be the output solution from our greedy algorithm.
Let $\mathcal{O}$ be an arbitrary optimal solution.
We want to show $|A| = |\mathcal{O}|$.

$|A| \leq |\mathcal{O}|$ is obvious because we assumed $\mathcal{O}$ was optimal.
So we need to show $|A| \geq |\mathcal{O}|$.

Suppose the requests in A are:
$A = \{(s_1, f_1), (s_2, f_2), \dots, (s_k, f_k)\}$
and in $\mathcal{O}$:
$\mathcal{O} = \{(s_1', f_1'), (s_2', f_2'), \dots, (s_m', f_m')\}$
and assume we have listed them in order:

$s_1 < f_1 \leq s_2 < f_2 \leq \dots$

$s_1' < f_1' \leq s_2' < f_2' \leq \dots$

Note that $k \leq m$.

We'll now show that A "stays ahead" of $\mathcal{O}$:

$$f_r \leq f'_r, \quad \text{for } r = 1, \ldots, k.$$

In English, the r^{th} meeting of A finishes before the r^{th} task of $\mathcal{O}$.

We'll prove this by induction.

Base case: $r = 1$, $f_1 \leq f_1'$

This is true because we defined our greedy algo. to start by picking the earliest ending time.