Friday, Jan 27, 2023
Lecture #5
MSSC 6000

Announcements

* HW 1 assigned today, due Mon. Feb 6 11:59pm on D2L
* Monday O.H. may change to 1pm - 2pm

Topic 3 - Greedy Algorithms (continued)

Ex: Giving change

Suppose you owe \$3.27 and pay with \$20.

$$20 - 3.27 = 16.73$$

~~100~~	~~50~~	~~20~~	~~10~~	~~5~~	~~1~~	~~0.25~~	~~0.1~~	~~0.05~~	~~0.01~~
			1	1	1	2	2		3
			6.73	1.73	0.73	0.23	0.03		0

= 10 bills/coins

Is this the optimal solution for \$16.73?

Yes, you can't form \$16.73 with 9 or fewer bills/coins.

Is this greedy algorithm optimal?

Theorem: For the US currency denominations listed above, the cashier's algorithm is optimal.

↳ fewest bills / coins

What would be some other reasonable def. of optimal?

* least weight of the bills/coins
* constraints on # of certain coins
* most bills/coins

Proof that the Cashier's Algo. always gives the fewest # of coins/bills:

To simplify, we'll just assume the

denominations are 1, 5, 10, 25 cents.

Suppose we want to make change for x cents.

fewest coins

Lemma: The optimal solution will have ≤ 4 pennies.

Proof: If an optimal solution had ≥ 5 pennies, we could swap 5 of them for a nickel, giving an even better solution.

Lemma: The optimal solution will have ≤ 1 nickel. (some kind of proof)

Lemma: The optimal solution will have $\#N + \#D \leq 2$

\# nickels   \# dimes

Proof: 2N: bad, 1D is better
1N + 2D: bad, 1Q is better

0N+3D: bad, 1Q+1N is better

Main proof: (proof by induction)

Base Case: $x = 0$ cents
Algo says 0 coins, clearly optimal ✓

Now assume we're making change for x cents, and that the Cashier's Algo gives an optimal solution for all inputs $< x$ cents.

Case 1: $x < 5$, only use pennies, clearly optimal ✓

Case 2: $5 \leq x < 10$
By our lemmas, the optimal sol. will use ≤ 4 pennies, therefore must use 1 nickel.
Solution = 1N + (optimal sol for $x-5$ cents)

Case 3: $10 \le x < 25$: lemmas say we must use 1D (5)

Solution $= 1D + \left(\text{optimal sol for } x-10 \text{ cents}\right)$

$\left(\begin{array}{l} x = 22 \text{ cents} \\ 1D + (\text{optimal sol for } 12 \text{ cents}) \end{array}\right)$

Case 4: $x \ge 25$: lemmas say we must use 1Q

(why? if not, we'd have >4 P or >1 N or 1N + 2D or 3D)

Solution $= 1Q + \left(\text{optimal sol. for } x-25 \text{ cents}\right)$. ∎

Ex: US postage denominations

1, 2, 3, 5, 10, 20, 35, 36, 55, 65, 75, 95, 100, 200, 500, 795, 1000, 2635

~~73¢: Cashier's Algo:~~
~~65¢ + 5¢ + 3¢ 3 stamps~~

72¢: Cashier's Algo:
65¢ + 5¢ + 2¢ = 3 stamps
Better: 36¢ + 36¢ = 2 stamps

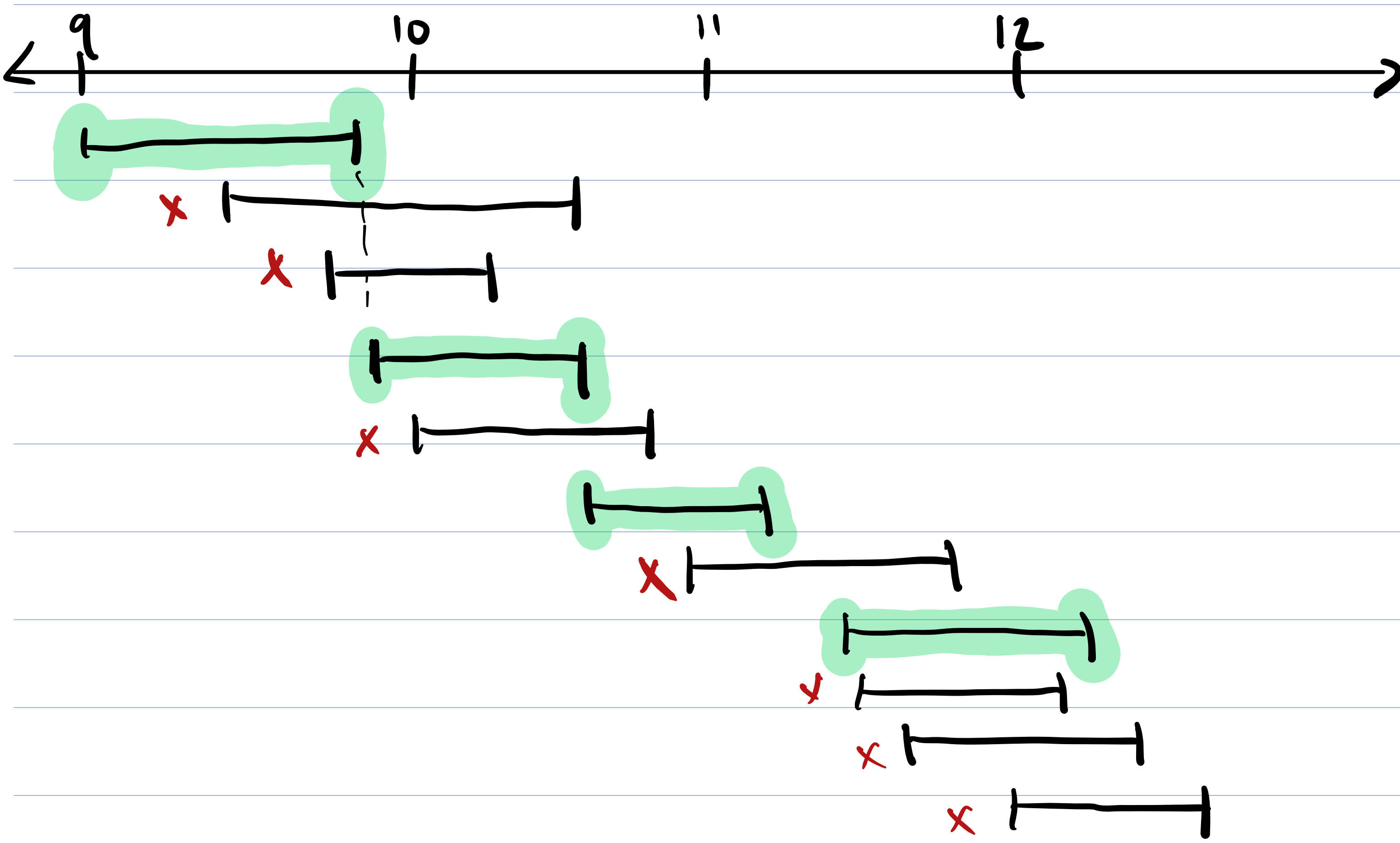
With these denominations, C.A. is <u>not</u> optimal.

<u>Problem #1</u>: Interval Scheduling
(Algorithm Design, by Kleinberg + Tardos)

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

Ex: Requested times

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



What is the largest # of meetings that we can book?

Best = 4 meetings, there are many

Formal setup:

- n requests
- each request has a start time s_i and a finish time f_i. s_i and f_i can be real #s and $s_i < f_i$.

Goal: Find a maximal size subset of the requests with no overlapping meetings.

(No overlapping meeting means that if request # i and request # j are both included, $s_j \geq f_i$ or $s_i \geq f_j$)

Let's think about possible greedy approaches.

General idea:

* decide on a rule for which meeting is "best"
* pick it, eliminate conflicts, repeat

Idea #1: best = overlaps with the fewest other meetings

9