Learning Goals

- Describe typical properties of greedy algorithms ✓
- Describe scheduling problem + terms "Objective Function", "Completion Time", "Optimization Problem", "Scoring Function", "Optimal order", "Greedy Order" ✓
- Create counterexample to a greedy ordering ✓

Announcements

- Drop-in Evening Hours (good time? reminders?) Thurs.
- Prog. Assignment 1 Rough Draft Due Sun. (Eclipse, command line)
- Smith's Drop-in Hours
- PS1 Feedback

Exit Tickets

Greedy Algorithm (informal def): an alg that sequentially contstructs a solution through a series of myopic (short-sighted = local = not global = not thinking about future) decisions

Typical properties

- Easy to create
- Easy to analyze runtime
- Not optimal
- Hard to prove correct

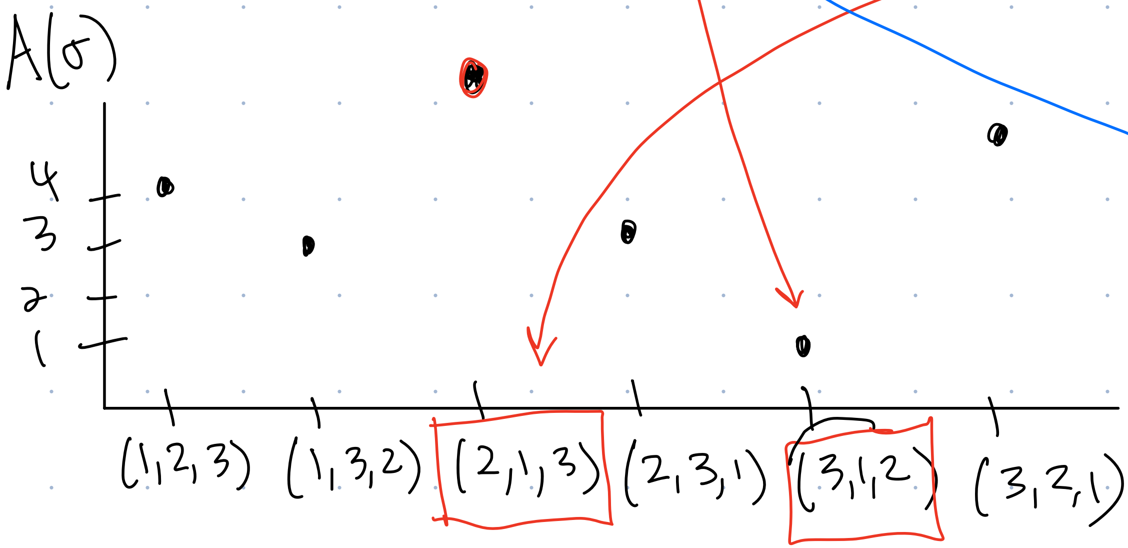
Ordering Problem (Abstract)

"objective function"

Input: n items
Output: Order σ that <u>minimizes</u> (or maximizes) $A(\sigma)$

A: orders $\rightarrow$ #

ex (3 items)



"optimization problem"

<u>Brute Force</u>: Evaluate $A(\sigma)$ for every ordering $\Omega(n!)$
+ find the min-value ordering

<u>Greedy</u>:
- Design a simple scoring function f: item $\rightarrow$ score
- Evaluate score for each item $\Omega(n)$
- $O(n\log n)$ Return σ that puts items in increasing/decreasing score order

Scheduling Tasks

Input: n tasks: time, weight

larger means more important

task	1	2	3
time	3	4	2
weight	5	1	2

Output: Ordering σ that minimizes

$$A(\sigma) = \sum_{i=1}^{n} w_i \cdot C_i(\sigma)$$

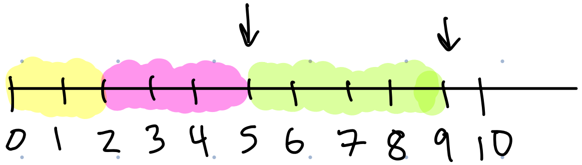
Applications: CPU, hw scheduling

$C_i(\sigma)$ = completion time of task i given ordering σ

ex:

task	1	2	3
time	3	4	2

$\sigma = (3,1,2)$



$C_1(3,1,2) = 5$

$C_2(3,1,2) = 9$

$C_3(3,1,2) = 2$

What is $C_3(2,1,3)$?

A) 2 B) 3 C) 7 $\boxed{\text{D) } 9}$

Calculate $A(\sigma)$ + determine best order: (Brute force)

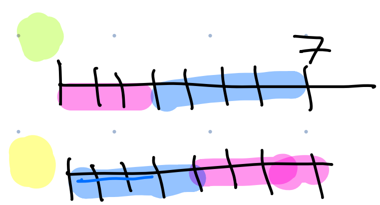
task	1	2
time	3	4
weight	5	1

Order	(1,2)	(2,1)
$A(\sigma)$	22	39

Order	(1,2)	(2,1)
$C_1(\sigma)$	3	7
$C_2(\sigma)$	7	4

$5 \cdot 3 + 1 \cdot 7$

$= 22$



$$\boxed{A(\sigma) = \sum_{i=1}^{n} w_i \cdot C_i(\sigma)}$$

What is this objective function optimizing? What time/weight jobs is it prioritizing?

<u>Output:</u> Ordering σ that minimizes

$$A(\sigma) = \sum_{i=1}^{n} w_i C_i(\sigma)$$

Creating Scoring Functions + Counterexamples

Creating a greedy algorithm Easy

Creating a good greedy algorithm Hard

To create:

$f(i) = t_i$ (increasing of f)

task	1	2
time	3	4
weight	5	1
f	3	4

"Greedy Ordering"

→ (1, 2)

Yay! Correct!

But will this greedy ordering always be correct?

Try to find a counterexample:

- think extreme
- create tasks that are similar in score but diff otherwise

$f(i) = t_i$ (increasing)

task	1	2
time	1	2
weight	1	100
f	1	2

Greedy Ordering → (1,2)

Order	(1,2)	(2,1)
$A(\sigma)$	300	203

Optimal Ordering: (2,1)

Group Exercise

Create counterexamples for scoring functions

A) $f(i) = w_i$ (decreasing) B) $f(i) = w_i - t_i$ (decreasing)

A)

task	1	2
time	1000	1
weight	2	1
f	2	1

Greedy $\rightarrow (1,2)$

Brute Force

Order	(1,2)	(2,1)
$A(\sigma)$	3001	2002

Optimal: $(2,1)$

B)

task	1	2
time	2	100
weight	1	98
f	-1	-2

$\rightarrow (1,2)$

Brute Force:

Order	(1,2)	(2,1)
$A(\sigma)$	9998	9902

Optimal $(2,1)$

One approach to designing greedy alg:

$w_i - t_i$ (decr)
w/t (decr)
$w^2 - 2t$ (dec)

- Create several reasonable scoring functions
- test, try to create counterex.
- No counterex. try prove correct, or just use (heuristic)

Thm: Ordering jobs by decreasing value of $f(i) = w_i/t_i$ is optimal for minimizing $A(\sigma) = \sum_i w_i C_i(\sigma)$ if w_i/t_i are all distinct.

Pf: [Exchange Argument → Type of Pf by Contradiction]
WLOG relabel tasks so $w_1/t_1 > w_2/t_2 > w_3/t_3 \cdots > w_n/t_n$
so greedy ordering $\sigma = (1, 2, 3, \dots n)$. Assume for contradiction the greedy ordering σ is not the optimal ordering. Then $\exists\, \sigma^*$ that is optimal and $\sigma^* \neq \sigma$.