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)

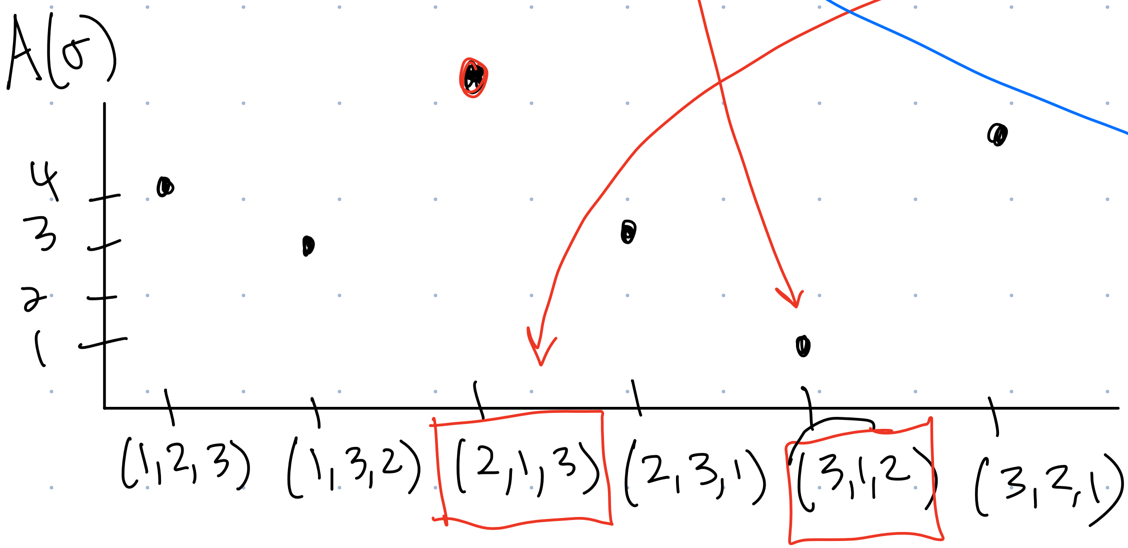
"objective function"

Input: n items

Output: Order σ that minimizes (or maximizes) $A(\sigma)$

A: orders $\to$ #

ex (3 items)



"optimization problem"

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

Greedy:
- Design a simple scoring function f: item $\to$ 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

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

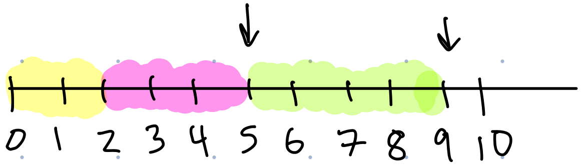
Applications: CPU, hw scheduling

$C_i(\sigma)$ = 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)

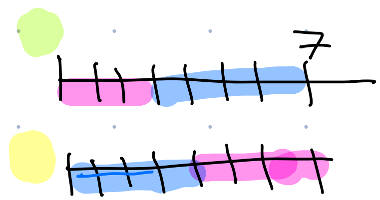
task	1	2
time	3	4
weight	5	1

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"

$\rightarrow (1, 2)$

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 → (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

→ (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 <u>reasonable</u> scoring functions
- test, try to create counterex.
- No counterex. try prove correct, or just use (heuristic)

<u>Thm</u>: Ordering jobs by $f = \frac{w_i}{t_i}$ (decreasing) is optimal for minimizing $A(\sigma) = \sum_i w_i C_i(\sigma)$ if w_i/t_i are all distinct.

<u>Pf</u>: [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 $\sigma^* \neq \sigma$.

Since $\sigma^* \neq \sigma$, there must be tasks in σ^* that are next to each other but out of numerical order

$$\sigma^* = (\ldots, y, b, \ldots) \quad \text{but } b < y$$

ex: $\sigma^* = (3, 1, 2)$.

What is b, y in this example?

A) $\boxed{b = 1 \quad y = 3}$

B) $b = 3$
$y = 1$

C) $b = 2$
$y = 3$

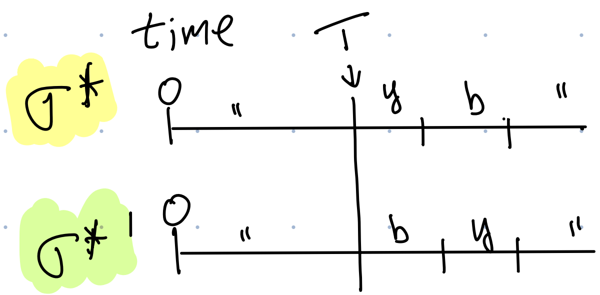
D) No unique b, y

Let $\sigma^{*\prime}$ be the same sequence as σ^* but with b, y exchanged to be in correct order

$$\sigma^* = (\ldots\ldots, y, b, \ldots\ldots) \qquad \sigma^* = (3, 1, 2)$$

$$\sigma^{*\prime} = (\ldots\ldots, b, y, \ldots\ldots) \qquad \sigma^{*\prime} = (1, 3, 2)$$

What is $A(\sigma^*) - A(\sigma^{*\prime}) < 0$ $\left(A(\sigma) = \sum_{i=1}^{n} w_i C_i(\sigma) \right)$



w_y, w_b

t_y, t_b

$\sigma^*(i) =$ i^{th} job in ordering σ^*

$$A(\sigma^*) = w_{\sigma^*(1)} C_{\sigma^*(1)} + w_{\sigma^*(2)} C_{\sigma^*(2)} + \ldots w_y \underbrace{C_y(\sigma^*)}_{(T+t_y)} + w_b \underbrace{C_b(\sigma^*)}_{(T+t_y+t_b)} + \ldots$$

$$- \; A(\sigma^{*\prime}) = w_{\sigma^{*\prime}(1)} C_{\sigma^{*\prime}(1)} + w_{\sigma^{*\prime}(2)} C_{\sigma^{*\prime}(2)} + \ldots w_b \underbrace{C_b(\sigma^{*\prime})}_{(T+t_b)} + w_y \underbrace{C_y(\sigma^{*\prime})}_{(T+t_b+t_y)} + \ldots$$

$$= w_b t_y - w_y t_b$$

$$\frac{0}{t_y t_b} > \frac{A(\sigma^*) - A(\sigma^{*'})}{t_y t_b} = \frac{w_b t_y - w_y t_b}{t_y t_b}$$ Divide both sides by $t_y t_b$

$$0 > \quad \text{"} \quad = \frac{w_b}{t_b} - \frac{w_y}{t_y} > 0$$

But $b < y$ so by our greedy ordering $\frac{w_b}{t_b} > \frac{w_y}{t_y}$

$$0 > \frac{w_b}{t_b} - \frac{w_y}{t_y}$$

$$0 < \frac{w_b}{t_b} - \frac{w_y}{t_y} \rightarrow \text{Contradiction!}$$

We had said that $A(\sigma^*)$ was optimal so it must have the smallest A value, but now we've shown that $\sigma^{*'}$ has a smaller A value, a contradictions.

Thus, our assumption that σ (our greedy ordering was not optimal was incorrect, and so σ is optimal.

$\square$

Structure of Exchange Proof

1. Assume greedy ordering σ (ordering from f) is not optimal
2. There must be some other ordering that is optimal σ^*
3. Modify σ^* by exchanging 2 elements to make it more like σ (called $\sigma^{*'}$)
4. Show $\sigma^{*'}$ is better (in A value) than σ^*

contradiction

What is the runtime of our greedy scheduling algorithm?

A) $O(1)$ B) $O(n)$ C) $O(n \log n)$ D $O(n^2)$

Where did we use w_i/t_i are all distinct?

Schedule(n):

$O(n)$ { • For $i \leftarrow 1$ to n:
  Calculate w_i/t_i (score)

+

$O(n \log n)$ { • Sort by score

Recall

$A(\sigma^*) - A(\sigma^{*\prime}) = w_b t_y - w_y t_b$. Divide both sides by $t_y \cdot t_b$:

$$\frac{A(\sigma^*) - A(\sigma^{*\prime})}{t_y t_b} = \frac{w_b}{t_b} - \frac{w_y}{t_y}.$$

But $b < y$, so $\frac{w_b}{t_b} \geq \frac{w_y}{t_y}$, so

$$\frac{A(\sigma^*) - A(\sigma^{*\prime})}{t_y t_b} \geq 0,$$ and $t_y, t_b > 0$, so

$A(\sigma^*) - A(\sigma^{*\prime}) \geq 0$. We said

$A(\sigma^*)$ was optimal so it must have the smallest A-value, but now we've shown $\sigma^{*\prime}$ has a smaller A-value, a contradiction.

Thus our assumption that σ was not optimal was incorrect, and σ must be optimal.

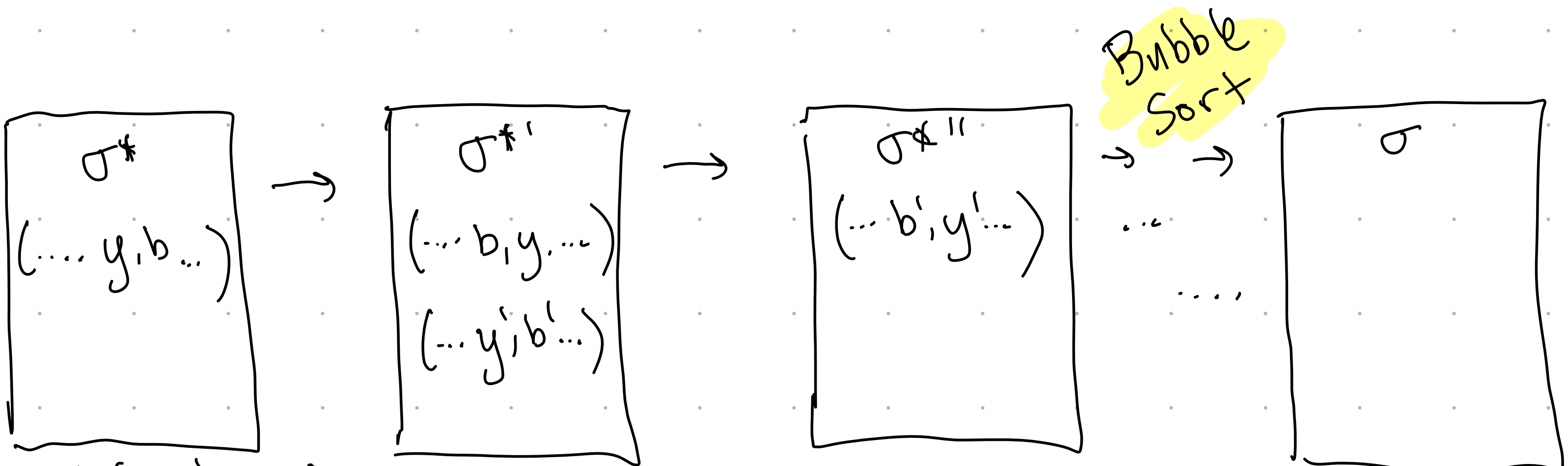
Lose contradiction!

Thm: Ordering jobs by decreasing value of 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 sketch: Choose some relabelling of tasks so that

$$w_1/t_1 \geq w_2/t_2 \geq w_3/t_3 \cdots \geq w_n/t_n$$

We call $\sigma=(1,2,3,\ldots,n)$ this greedy ordering. Let σ^* be any other ordering.



$$A(\sigma^*) \geq A(\sigma^{*\prime}) \geq A(\sigma^{*\prime\prime}) \cdot \geq A(\sigma)$$

We have shown for every other ordering $A(\sigma^*) \geq A(\sigma)$, so σ must be optimal.

What is the runtime of our greedy scheduling algorithm?

A) $O(1)$ B) $O(n)$ C) $O(n \log n)$ D $O(n^2)$

Work backwards to find optimal scoring function

Thm: Ordering jobs by decreasing value of $\overset{f}{\cancel{w_i/t_i}}$ 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 so $\overset{f_1}{\cancel{w_1/t_1}} > \overset{f_2}{\cancel{w_2/t_2}} > \overset{f_3}{\cancel{w_3/t_3}}$ so greedy ordering is $\sigma = (1, 2, 3, \dots, n)$. Assume for contradiction that σ is not optimal. Then $\exists\, \sigma^*$ that is optimal.

Since $\sigma^* \neq \sigma$, there must be tasks b, y in σ^* that are next to each other but out of order

$$\sigma^* = (\dots\, y, b \dots) \quad b < y$$

ex: $n = 3$

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

$\sigma^* = (\boxed{\underset{y}{3}, \underset{b}{1}}, 2)$

Let $\sigma^{*\prime}$ be the same sequence as σ^*, but with b, y exchanged to be in the correct order

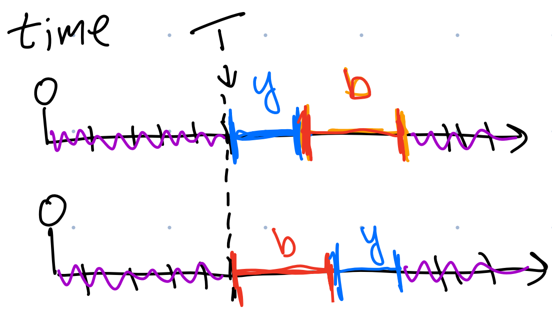
$$\sigma^* = (\dots \; y, b \; \dots)$$

$$\sigma^{*\prime} = (\dots \; b, y \; \dots)$$

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

$$\sigma^{*\prime} = (1, 3, 2)$$

What is $A(\sigma^*) - A(\sigma^{*\prime})$? $\left(A(\sigma) = \sum_{i=1}^{n} w_i C_i(\sigma) \right)$



σ^*

$\sigma^{*\prime}$

$\sigma^*(i) =$ i^{th} job in ordering σ^*

$$A(\sigma^*) = \cancel{w_{\sigma^*(1)} C_{\sigma^*(1)}} + \cancel{w_{\sigma^*(2)} C_{\sigma^*(2)}} + \dots w_y(\cancel{T} + \cancel{t_y}) + w_b(\cancel{T} + t_y + \cancel{t_b}) + \cancel{\dots}$$

$$-A(\sigma^{*\prime}) = \cancel{w_{\sigma^{*\prime}(1)} C_{\sigma^{*\prime}(1)}} + \cancel{w_{\sigma^{*\prime}(2)} C_{\sigma^{*\prime}(2)}} + \dots w_b(\cancel{T} + \cancel{t_b}) + w_y(\cancel{T} + \cancel{t_y} + t_b) + \cancel{\dots}$$

$$= w_b t_y - w_y t_b$$

$A(\sigma^*) - A(\sigma^{*\prime}) = w_b t_y - w_y t_b$. Divide both sides by $t_y \cdot t_b$:

$$\frac{A(\sigma^*) - A(\sigma^{*\prime})}{t_y t_b} = \frac{w_b}{t_b} - \frac{w_y}{t_y}.$$

But $b < y$, so $\cancel{\frac{w_b}{t_b} > \frac{w_y}{t_y}}$, so

$$\frac{A(\sigma^*) - A(\sigma^{*\prime})}{t_y t_b} > 0,$$

and $t_y, t_b > 0$, so

$A(\sigma^*) - A(\sigma^{*\prime}) > 0$. We said $A(\sigma^*)$ was optimal so it must have the smallest A-value, but now we've shown $\sigma^{*\prime}$ has a smaller A-value, a contradiction.

Thus our assumption that σ was not optimal was incorrect, and σ must be optimal.