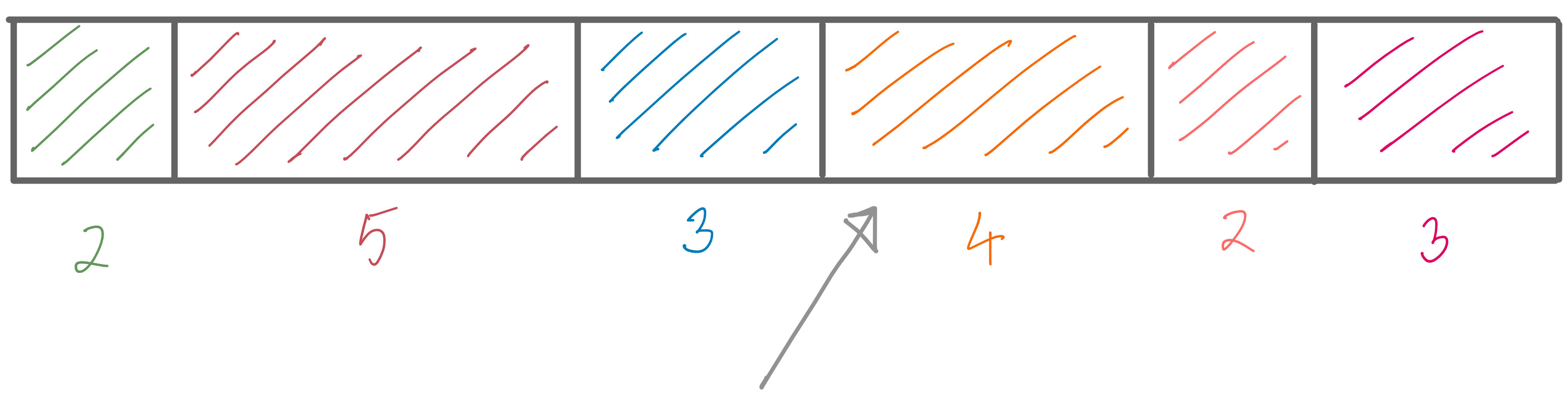
Advanced Algorithms (W1, P1) — Greedy Algorithms

Storing Files on Tape. I

We have n files that we want to store on a magnetic tape.

files $F_1, F_2, \ldots, F_n$ of lengths $L_1, L_2, \ldots, L_n$, respectively.



cost of access $= 2+5+3+4$

Storing Files on Tape. I

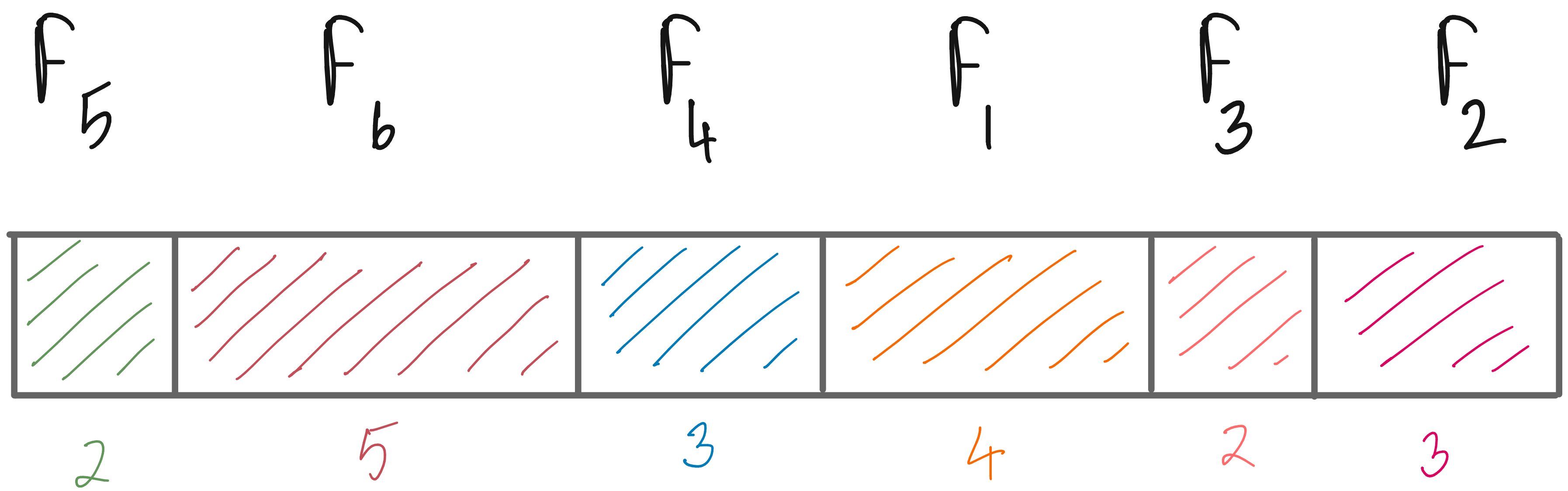
Expected cost of reading a random file

$$\mathbb{E}(\text{cost}) = \sum_{k=1}^{n} \frac{\text{cost}(k)}{n}$$

$$= \frac{1}{n} \sum_{k=1}^{n} \sum_{i=1}^{k} L_i$$

ADVANCED ALGORITHMS (W1, P1) Greedy Algorithms

Storing Files on Tape. I



$$\underline{\ell_5} + \underline{\ell_5 + \ell_6} + \underline{\ell_5 + \ell_6 + \ell_4} + \underline{\ell_5 + \ell_6 + \ell_4 + \ell_1}$$

$$+ \underline{\ell_5 + \ell_6 + \ell_4 + \ell_1 + \ell_3} + \underline{\ell_5 + \ell_6 + \ell_4 + \ell_1 + \ell_3 + \ell_2}$$

Storing Files on Tape. I

When files are sorted according to π :

Expected cost of reading a random file

$$\mathbb{E}(\text{cost}) = \sum_{k=1}^{n} \frac{\text{cost}(k)}{n}$$

$$= \frac{1}{n} \sum_{k=1}^{n} \sum_{i=1}^{k} L_{\pi(i)}$$

* $\pi(i)$ denotes the index of the i^{th} file from the left stored on the tape

Storing Files on Tape. I

Question

Which order minimizes the expected cost ?

A strategy that "feels" natural :

Store the files in increasing order of length

Storing Files on Tape. I

Lemma. $E(\text{cost})$ is minimized when $L_{\pi(i+1)} \geq L_{\pi(i)} \; \forall i$.

Suppose π is opimal & $\underbrace{\exists\, i \text{ s.t. } L_{\pi(i+1)} < L_{\pi(i)}.}_{\text{for the sake of contradiction}}$

Modify π by swapping $i, i+1$.

Before

$\overbrace{\ell_{\pi(1)} + \cdots + \ell_{\pi(i)}}^{\text{cost}(\pi(i))}$

$\underbrace{\ell_{\pi(1)} + \cdots + \ell_{\pi(i)} + \ell_{\pi(i+1)}}_{\text{cost}(\pi(i+1))}$

After

$\overbrace{\ell_{\pi(1)} + \cdots + \ell_{\pi(i+1)}}^{\text{cost}(\pi(i+1))}$

$\underbrace{\ell_{\pi(1)} + \cdots + \ell_{\pi(i+1)} + \ell_{\pi(i)}}_{\text{cost}(\pi(i))}$

Storing Files on Tape. II

S'pose now that we are also given information about the frequencies $\underbrace{\text{with which these files are accessed}}_{F_i \text{ will be accessed } f_i \text{ times}}$.

$$\text{Total cost using } \pi = \sum_{k=1}^{n} f_{\pi(k)} \cdot \Big\{ \underbrace{\sum_{i=1}^{k} \ell_{\pi(i)}}_{\text{cost of } \pi(k)} \Big\} = \sum_{k=1}^{n} \sum_{i=1}^{k} f_{\pi(k)} \cdot \ell_{\pi(i)}$$

Storing Files on Tape. II

Question

Which order minimizes the total cost?

If all file lengths are equal : sort by ↓ frequencies

← more frequent → less frequent

If all frequencies are equal : sort by ↑ file lengths

← smaller files → larger files

Storing Files on Tape. II

higher values have priority

In general, sort by the ratio f/ℓ

lower values have priority

Lemma. Total cost (π) is minimized when

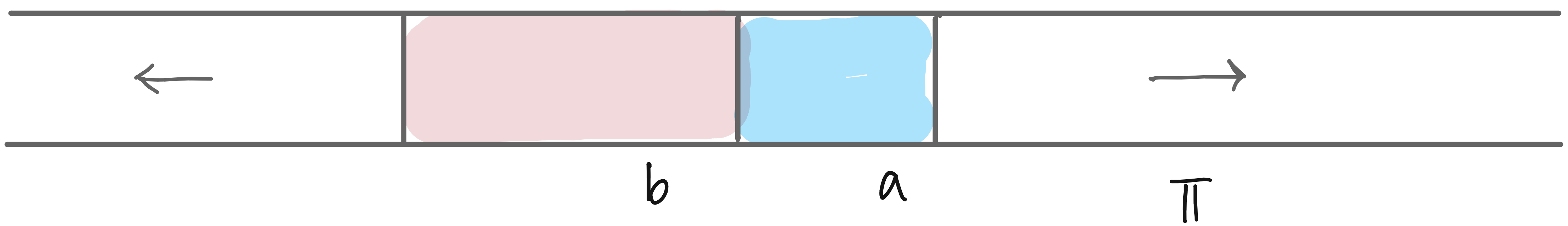
$$\frac{\ell_{\pi(i)}}{f_{\pi(i)}} \leq \frac{\ell_{\pi(i+1)}}{f_{\pi(i+1)}} \quad \text{for all } i$$

Storing Files on Tape. II

$\pi(i) = a \qquad \pi(i+1) = b$

Proof.

S'pose π is optimal & $\ell_a / f_a > \ell_b / f_b$



Swap these Files

Storing Files on Tape. II

$$\text{cost}(\pi) = \cdots + f_a \ell_a + \cdots + f_b l_a + f_b l_b + \cdots$$

$$\text{cost}(\pi') = \cdots + f_b l_b + \cdots + f_a l_b + f_a l_a + \cdots$$

$$\text{net change} = f_a l_b - f_b l_a$$

Recall that : $\underbrace{\frac{\ell_a}{f_a} > \frac{\ell_b}{f_b}}_{\text{by assumption}} \Rightarrow f_b \ell_a > f_a l_b$

$\Rightarrow f_a l_b - f_b l_a < 0$,

a net reduction in cost!

Scheduling Classes.

The Setting

n classes in a semester

(start time, finish time)

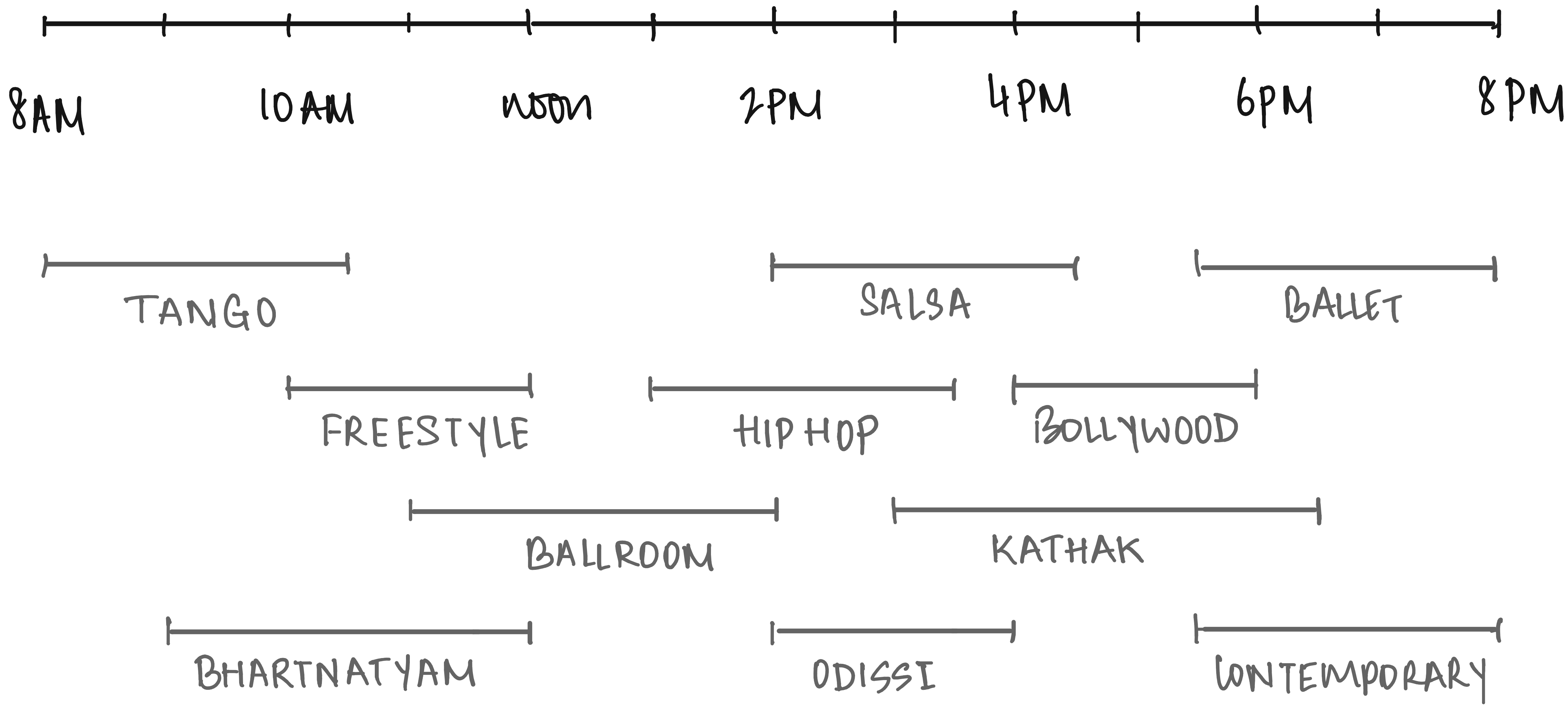
s_p f_p

(Also, all classes happen on Saturdays)

ADVANCED ALGORITHMS (W1, P3)

Greedy Algorithms

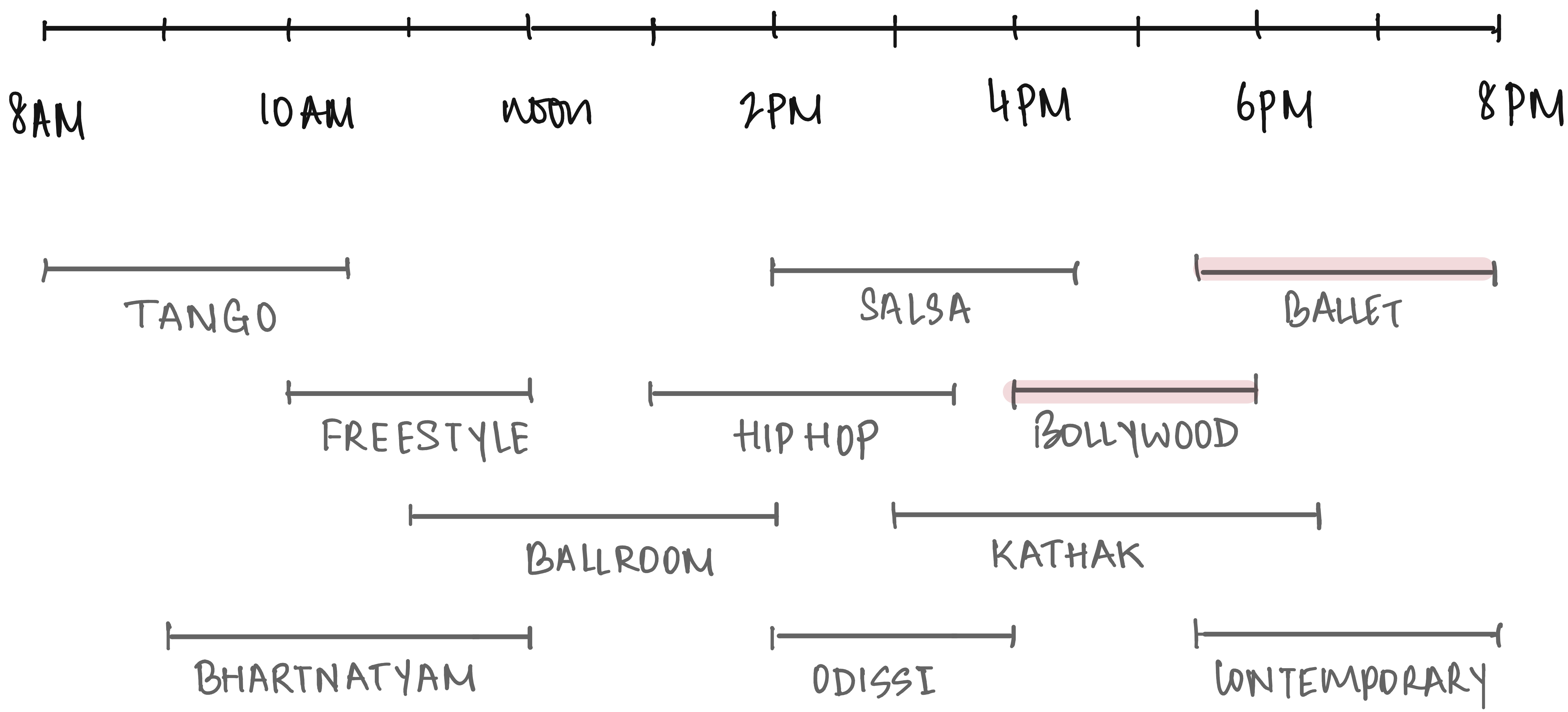
Scheduling Classes.



ADVANCED ALGORITHMS (W1, P3)

Greedy Algorithms

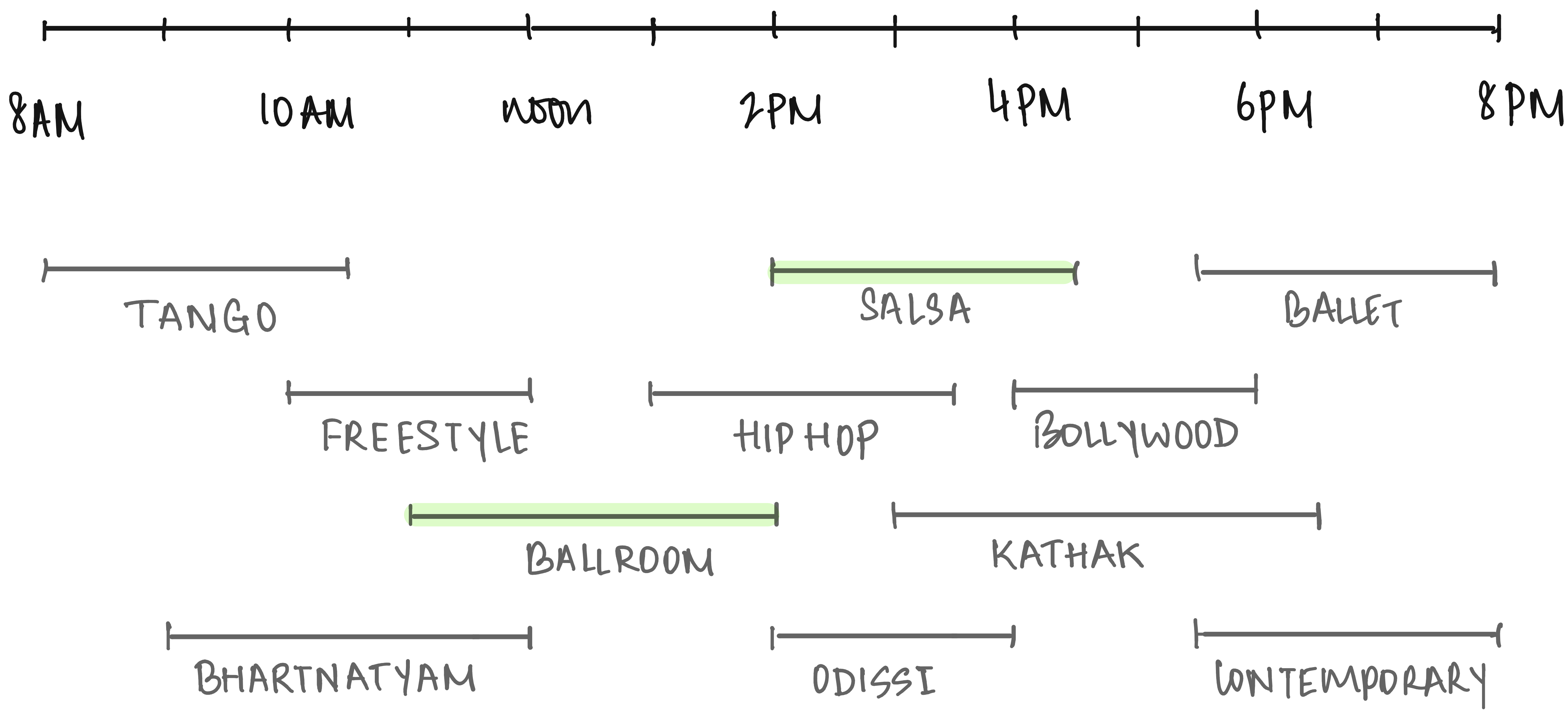
Scheduling Classes.



ADVANCED ALGORITHMS (W1, P3)

Greedy Algorithms

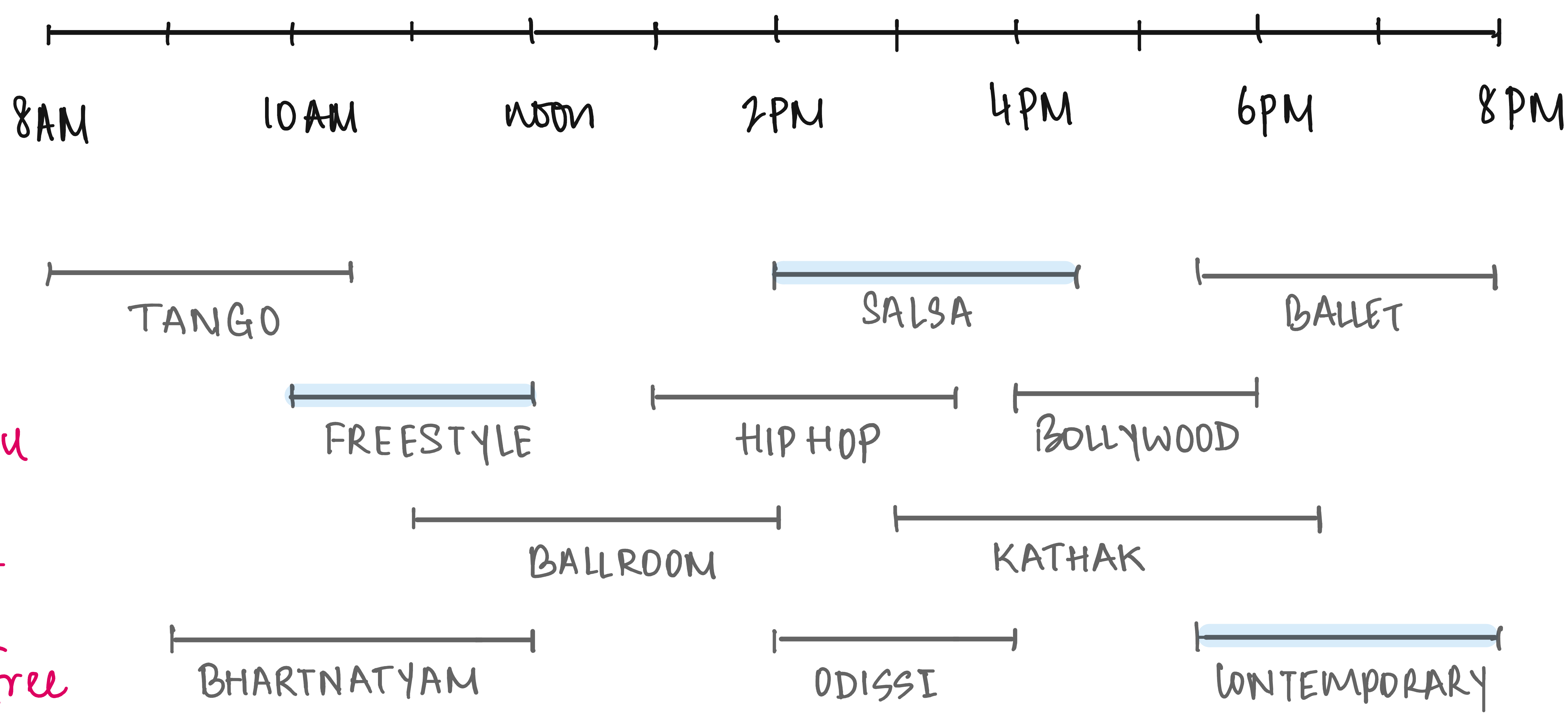
Scheduling Classes.



ADVANCED ALGORITHMS (W1, P3)

Greedy Algorithms

Scheduling Classes.

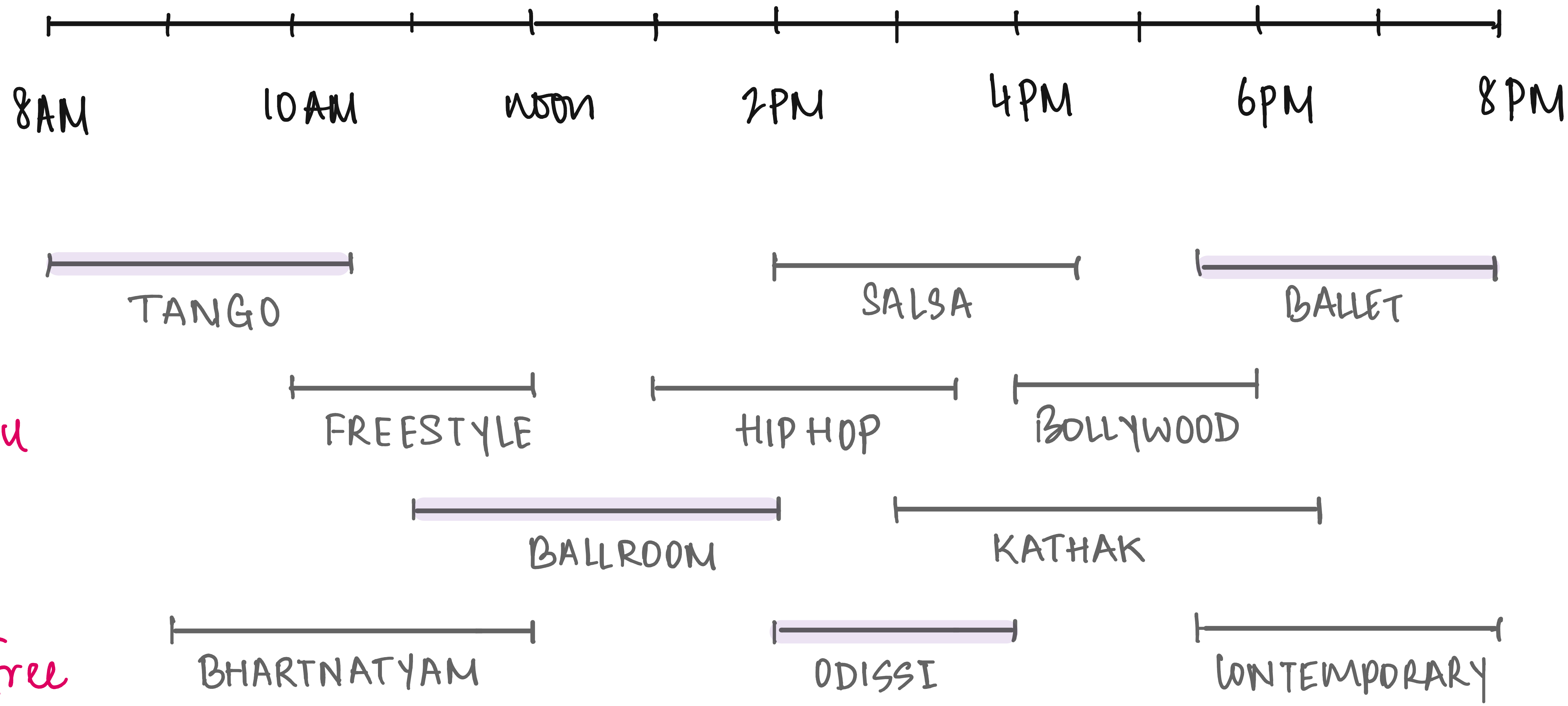


Can you find 4 conflict-free classes?

ADVANCED ALGORITHMS (W1, P3)

Greedy Algorithms

Scheduling Classes.



Can you find 5 conflict-free classes?

Advanced Algorithms (W1, P3)

Greedy Algorithms

Scheduling Classes.

$s_p \geq f_q$ or $f_p \leq s_q$

$\forall p, q$

The Goal

Find a largest conflict-free collection of classes.

$\{ \ldots, p, q, \ldots \}$

Scheduling Classes.

Possible Greedy Choices → Shortest class

Possible Greedy Choices → class with fewest overlaps

Possible Greedy Choices → Class that begins first

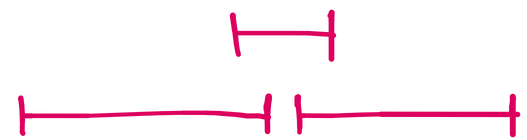
ADVANCED ALGORITHMS (W1, P3)

Greedy Algorithms

Scheduling Classes.

Possible Greedy Choices →

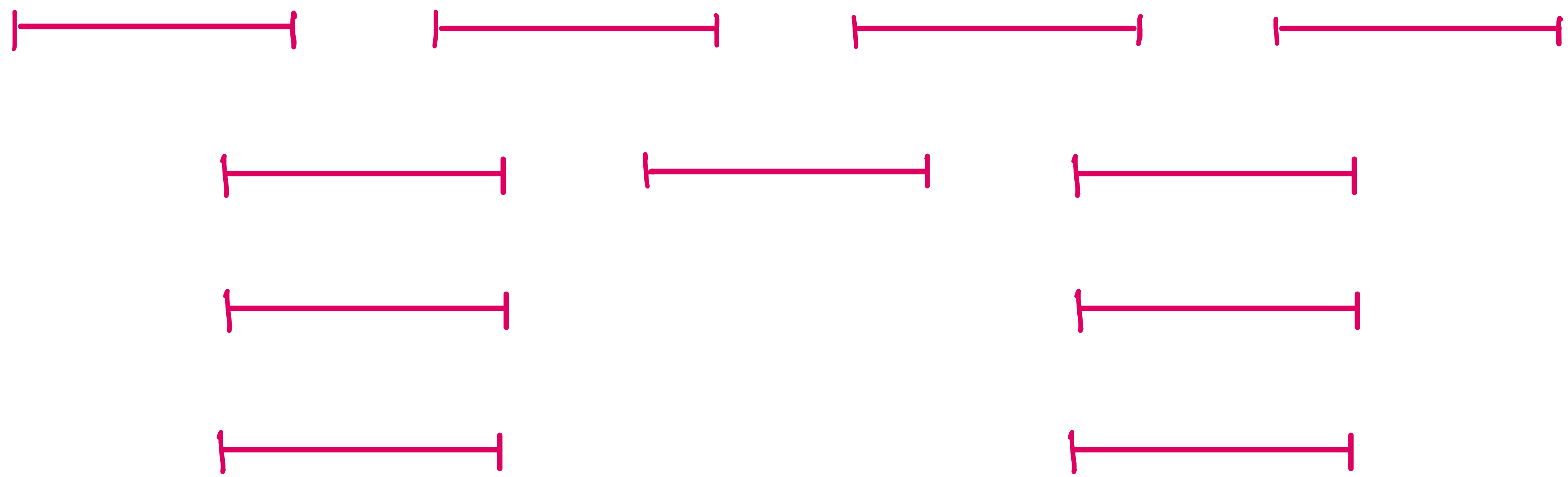
Shortest Class



Class with fewest overlaps

Class that begins first

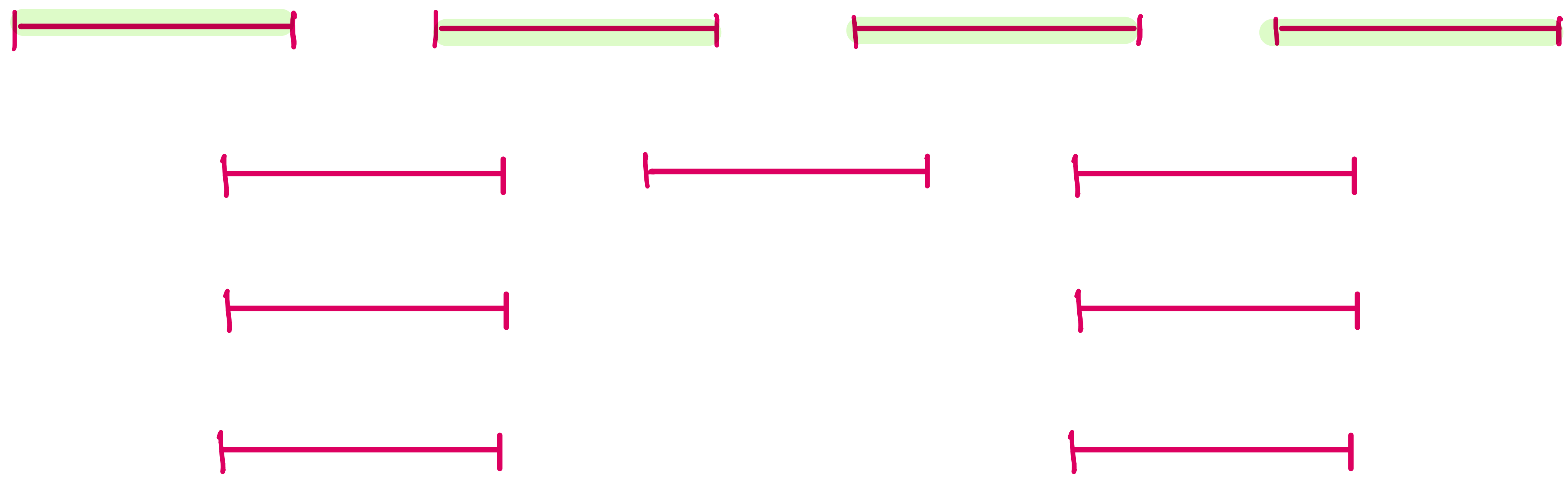
Scheduling Classes.



ADVANCED ALGORITHMS (W1, P3)

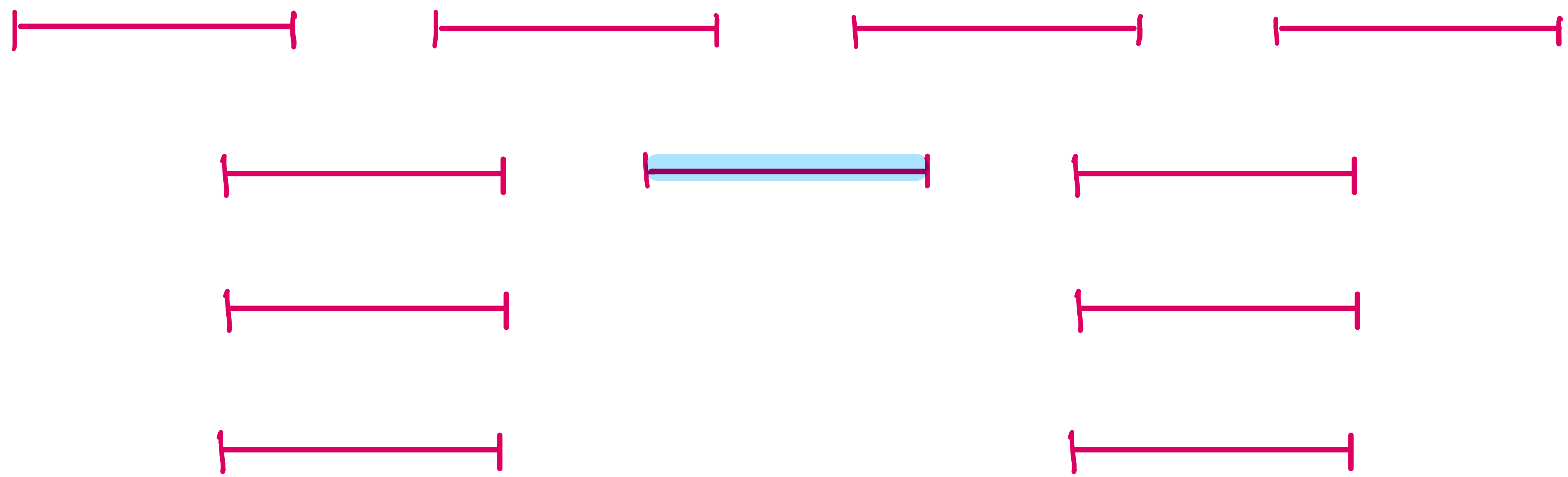
Greedy Algorithms

Scheduling Classes.



Opt = 4

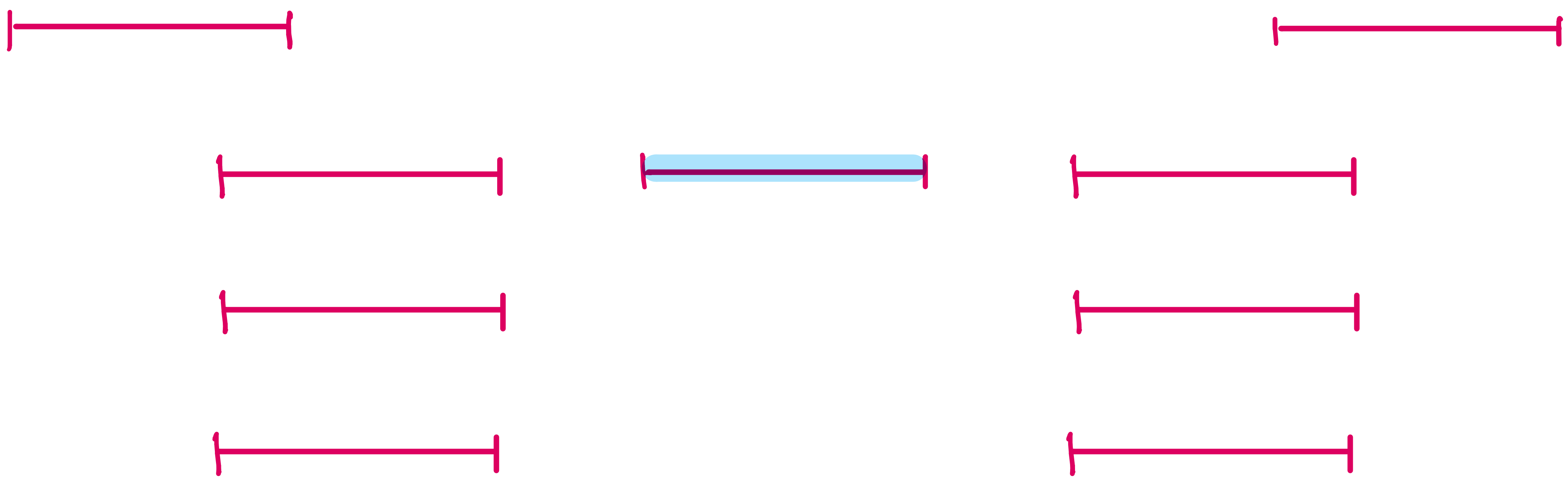
Scheduling Classes.



ADVANCED ALGORITHMS (W1, P3)

Greedy Algorithms

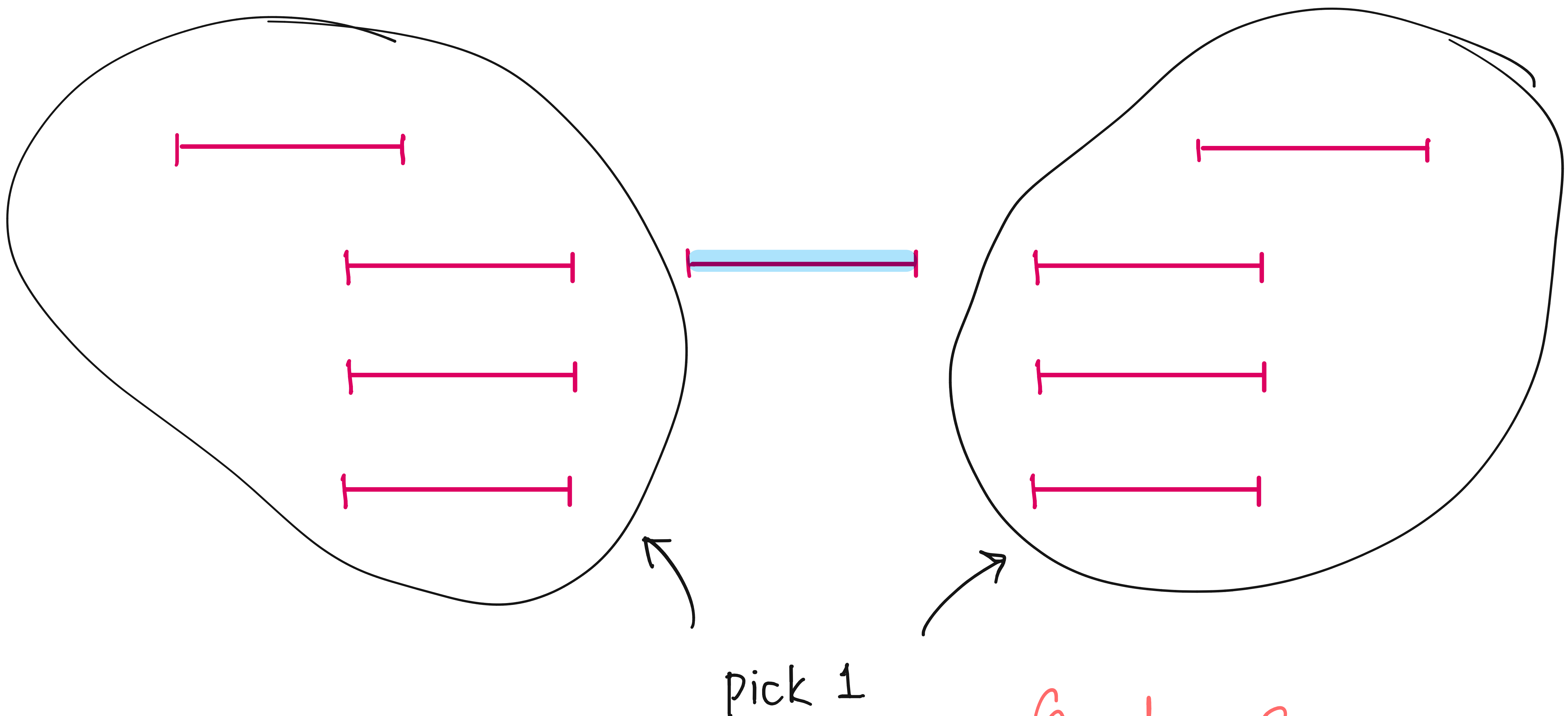
Scheduling Classes.



ADVANCED ALGORITHMS (W1, P3)

Greedy Algorithms

Scheduling Classes.



Greedy = 3

Scheduling Classes.

Takeaway

Intuitively appealing greedy strategies may not actually work!

Scheduling Classes.

The Greedy Approach
that does work:

Pick the class that ends first.

Scheduling Classes : II

Greedy Scheduler

Sort classes by finish time.

Count = 0, $X = \emptyset$

while a class is still available :

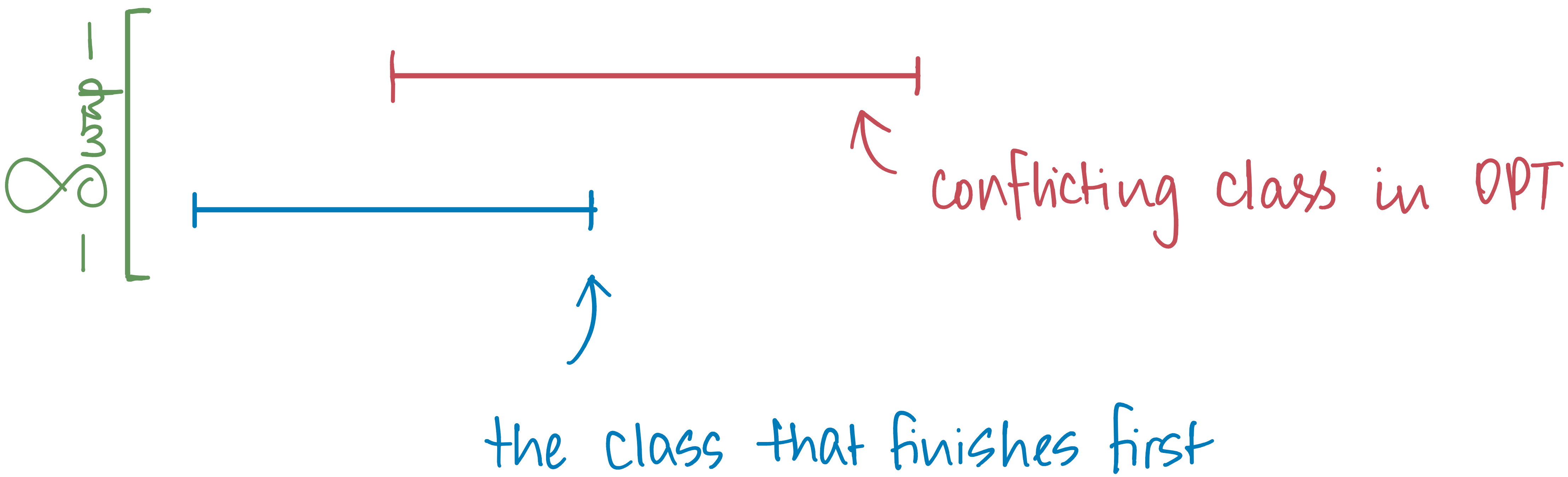
add the first class to X & remove it

C

remove all classes that conflict with C.

Scheduling Classes: II

Lemma. At least one optimal conflict-free collection of classes includes the one that finishes first

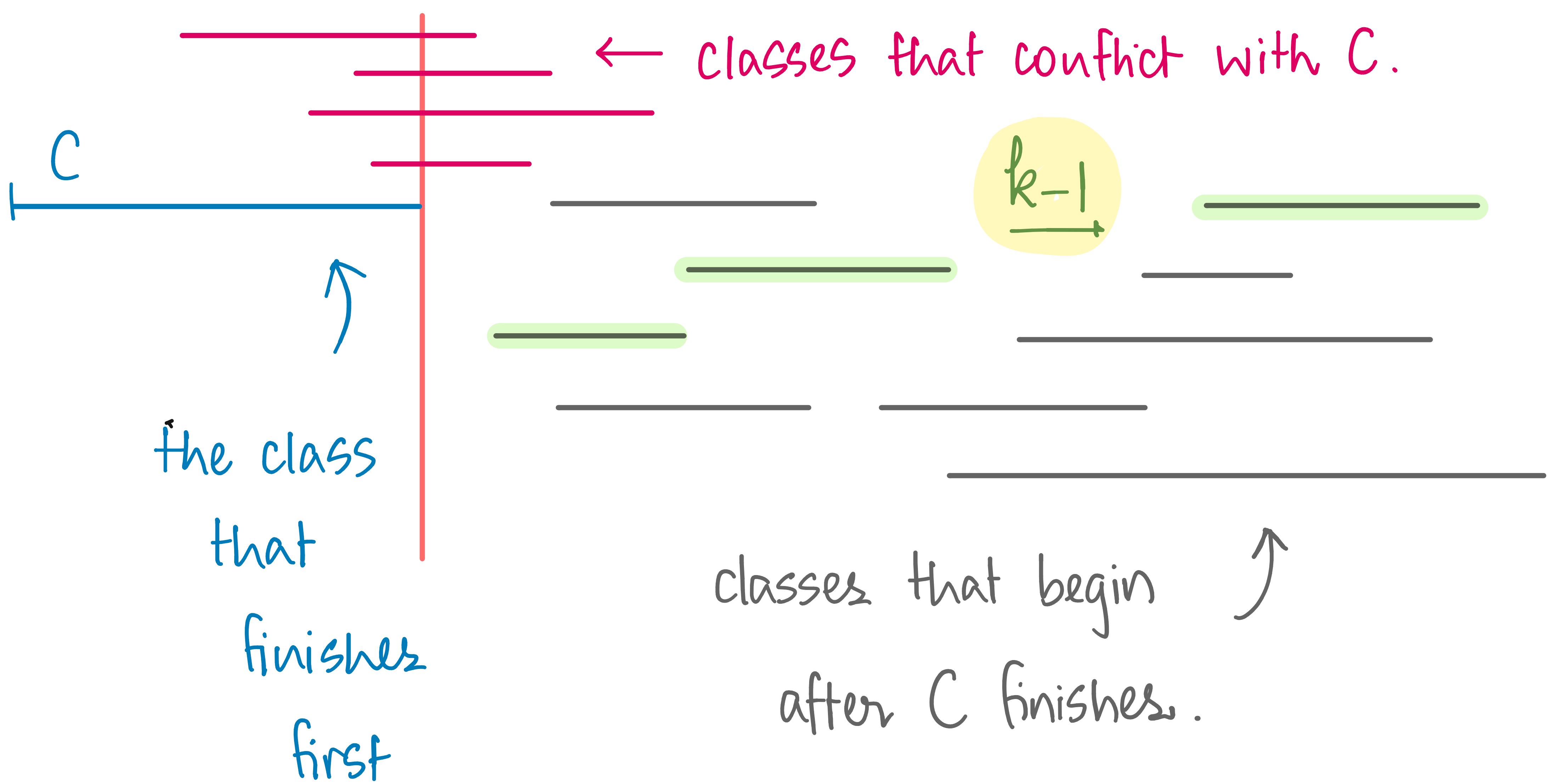


Scheduling Classes : II

Theorem. The greedy schedule is optimal

Recall:

∃ an optimal schedule that contains C. w/ k classes.

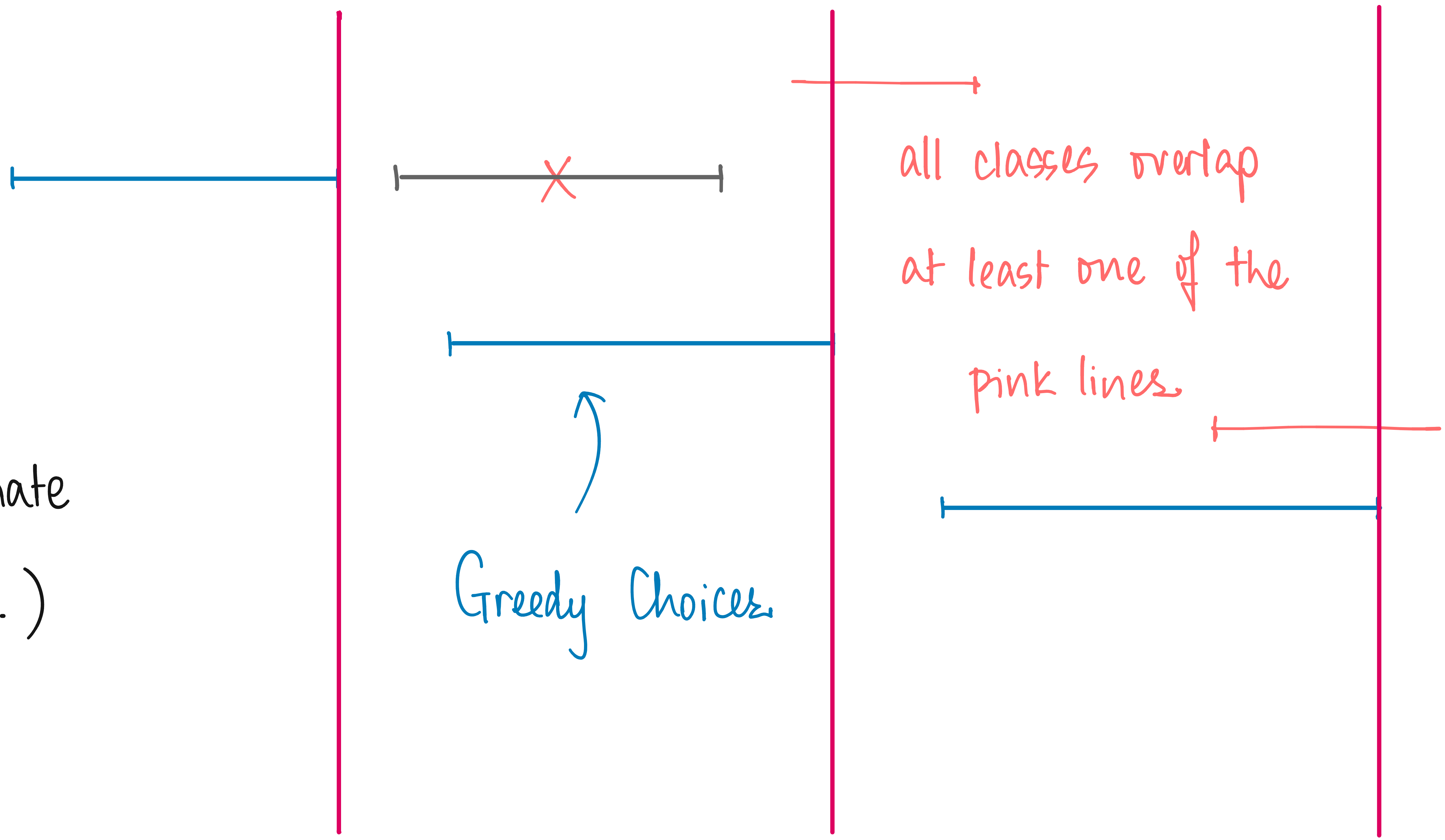


ADVANCED ALGORITHMS (W1, P4)

Greedy Algorithms

Scheduling Classes: II

(An alternate proof.)



Stable Matchings I

Two groups of "agents"

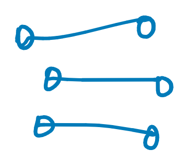
→ doctors, hospitals

→ students, colleges

→ jobs, applicants

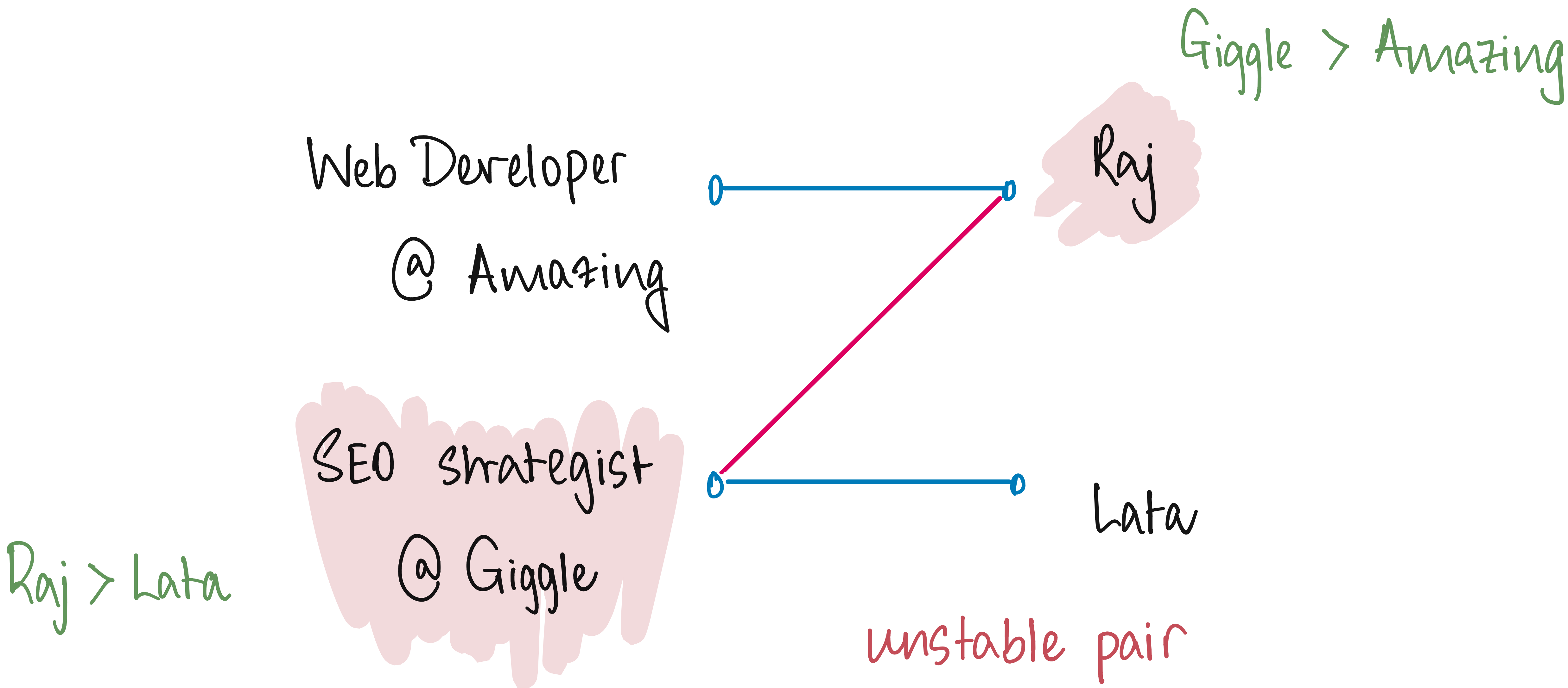
→ men, women

who have preferences rankings over each other need to be matched with each other.



How?

Stable Matchings I



unstable pair

AKA blocking pair

Stable Matchings I

i/p → n men & n women.

all men rank all the women & vice-versa.

GOAL. Find a matching that minimizes the # of blocking pairs.

A matching w/ no blocking pairs is called a **STABLE MATCHING**.

Stable Matchings I

Idea Start with any matching.

While $\exists$ a blocking pair :

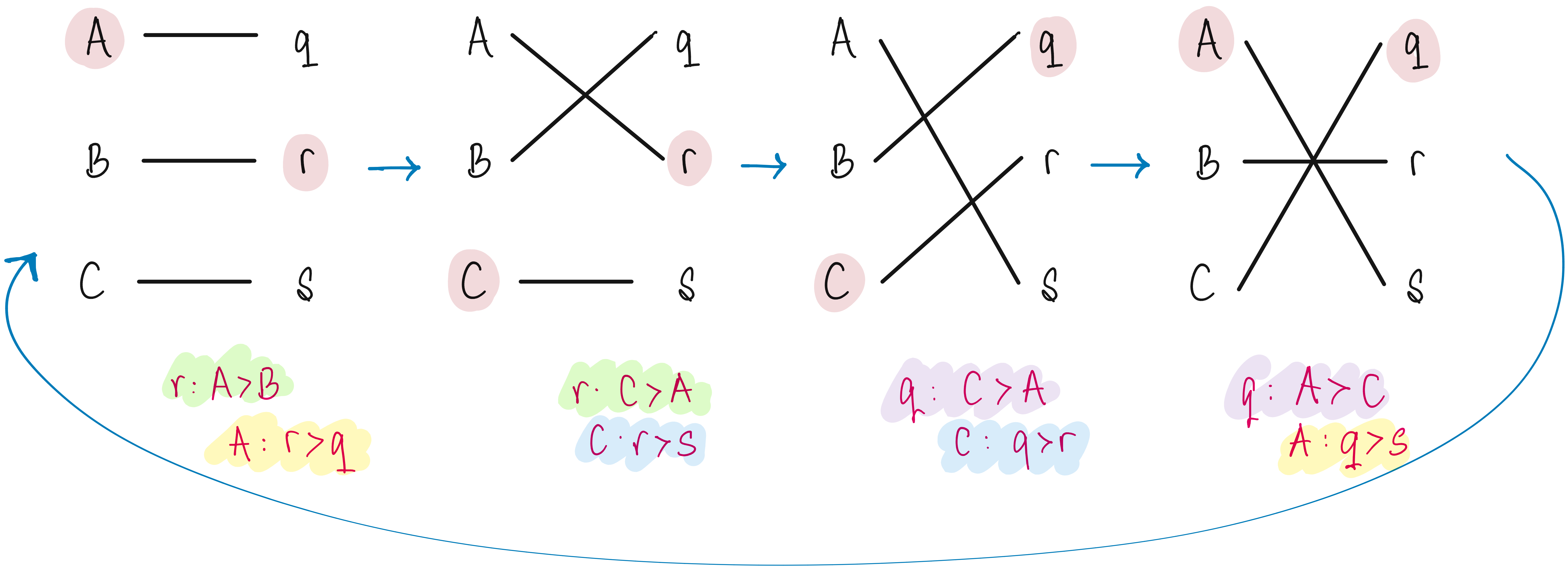
rematch in their favor

to $\downarrow$ # of blocking pairs by 1

& hopefully (?!) not create new ones.

Stable Matchings I

This greed is never-ending!



Stable Matchings II

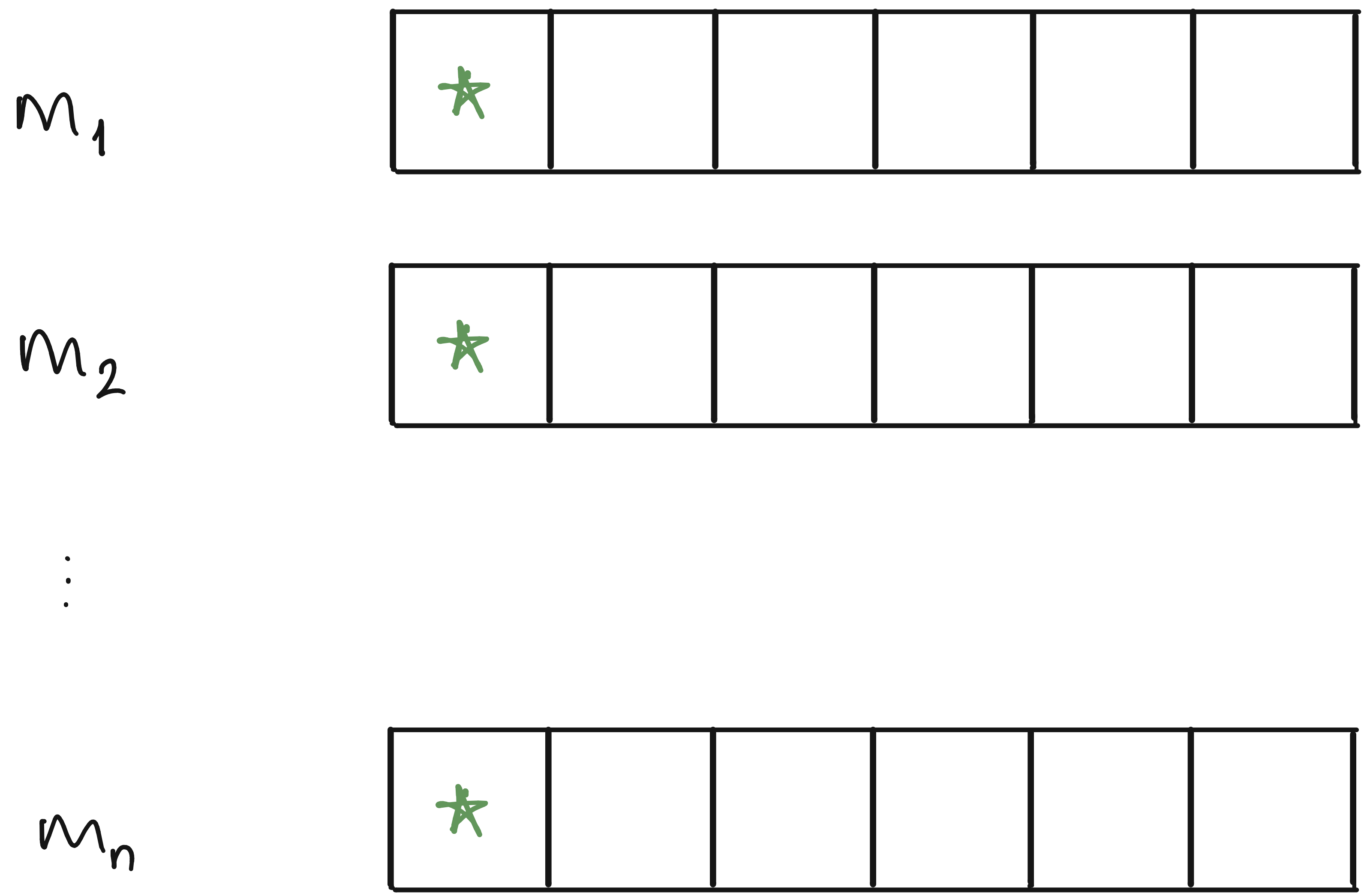
Another greedy approach (that works)

→ men propose in rank order

& women engage with the best offer.

Stable Matchings II

Men's perspective



all "single" men "propose" to the highest-ranked woman who has not rejected them yet.

Stable Matchings II

$W_i \rightsquigarrow$ no proposals $\rightarrow$ nothing to do

at least 1

$W_i \rightsquigarrow$ multiple proposals $\rightarrow$ engage w/ the best*

from W_i's perspective

* even if it means breaking off an existing engagement.

Stable Matchings II

$m_k > m_j$

$m_k \rightsquigarrow w_i \longleftrightarrow m_j$ $\rightsquigarrow$ goes back to being single

if w_i does not accept m_k's proposal,

(w_i, m_k) will be a blocking pair

$\downarrow$

will match with someone "worse" than w_i

Stable Matchings II

q	$C > B > A$		A	$q > s > r$
r	$A > C > B$		B	$q > r > s$
s	$A > B > C$		C	$s > r > q$

ADVANCED ALGORITHMS (W1, P6) Greedy Algorithms

Stable Matchings II

A, B

q C > B > A

r A > C > B

s A > B > C

C

A → q > s > r

B → q > r > s

C → s > r > q

Stable Matchings II

q	C > B > A		A	s > r
r	A > C > B		B	q > r > s
s	A > B > C		C	s > r > q

qB , sC

ADVANCED ALGORITHMS (W1, P6)

Greedy Algorithms

Stable Matchings II

	q	C > B > A	A →	s > r
	r	A > C > B	B	q > r > s
A	s	A > B > C	C	s > r > q

qB, sC

Stable Matchings II

q	C > B > A	A	s > r
r	A > C > B	B	q > r > s
s	A > B > C	C	r > q

qB, ~~sC~~, sA

Stable Matchings II

q	C > B > A	A	s > r
c r	A > C > B	B	q > r > s
s	A > B > C	C	→ r > q

qB, ~~sC~~, sA

Stable Matchings II

q	C > B > A	A	s > r
r	A > C > B	B	q > r > s
s	A > B > C	C	r > q

qB, ~~sC~~, sA, rC

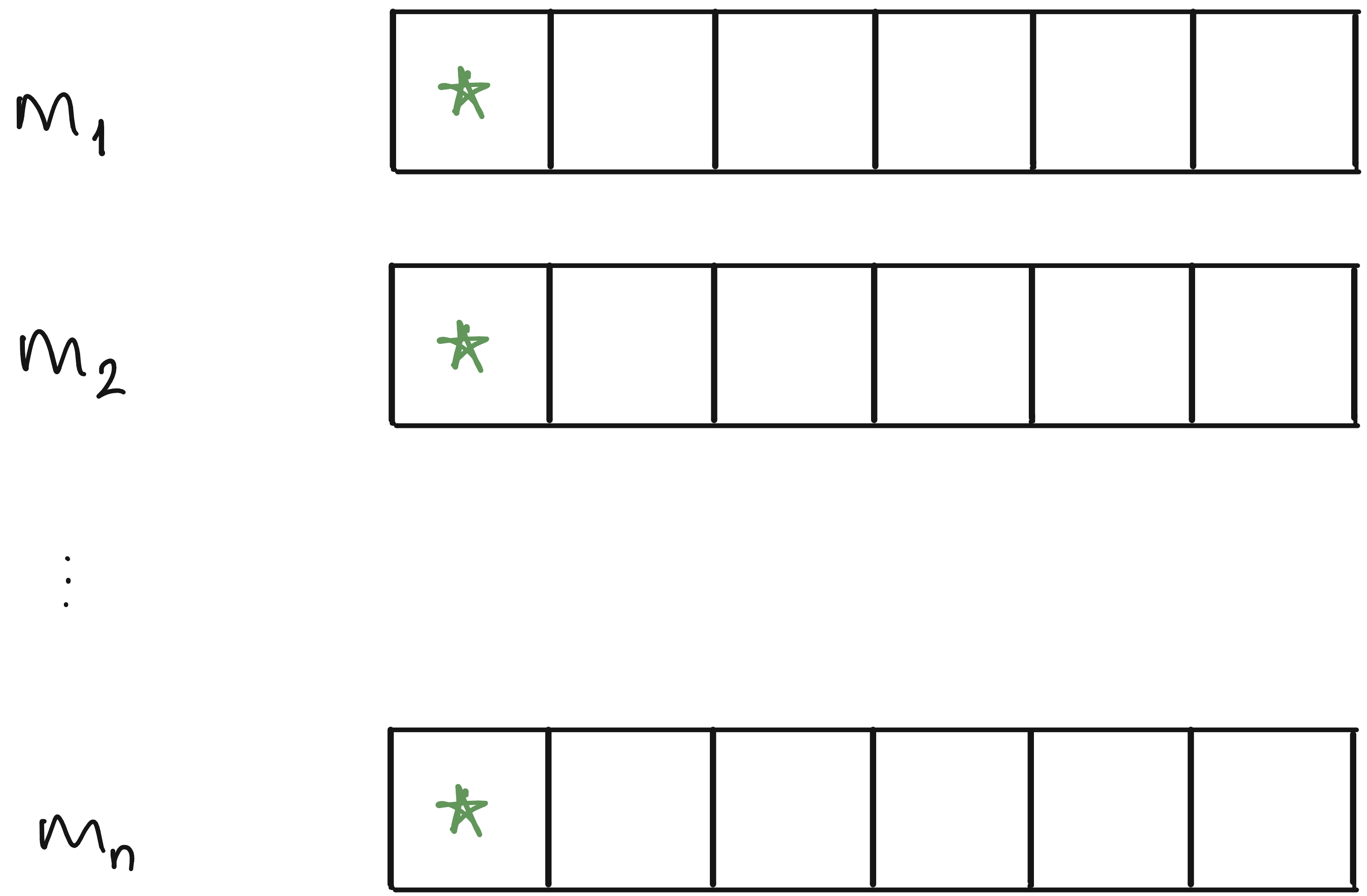
Stable Matchings III

Correctness of Deferred Acceptance

terminates
in a stable
matching

Stable Matchings III

Men's perspective



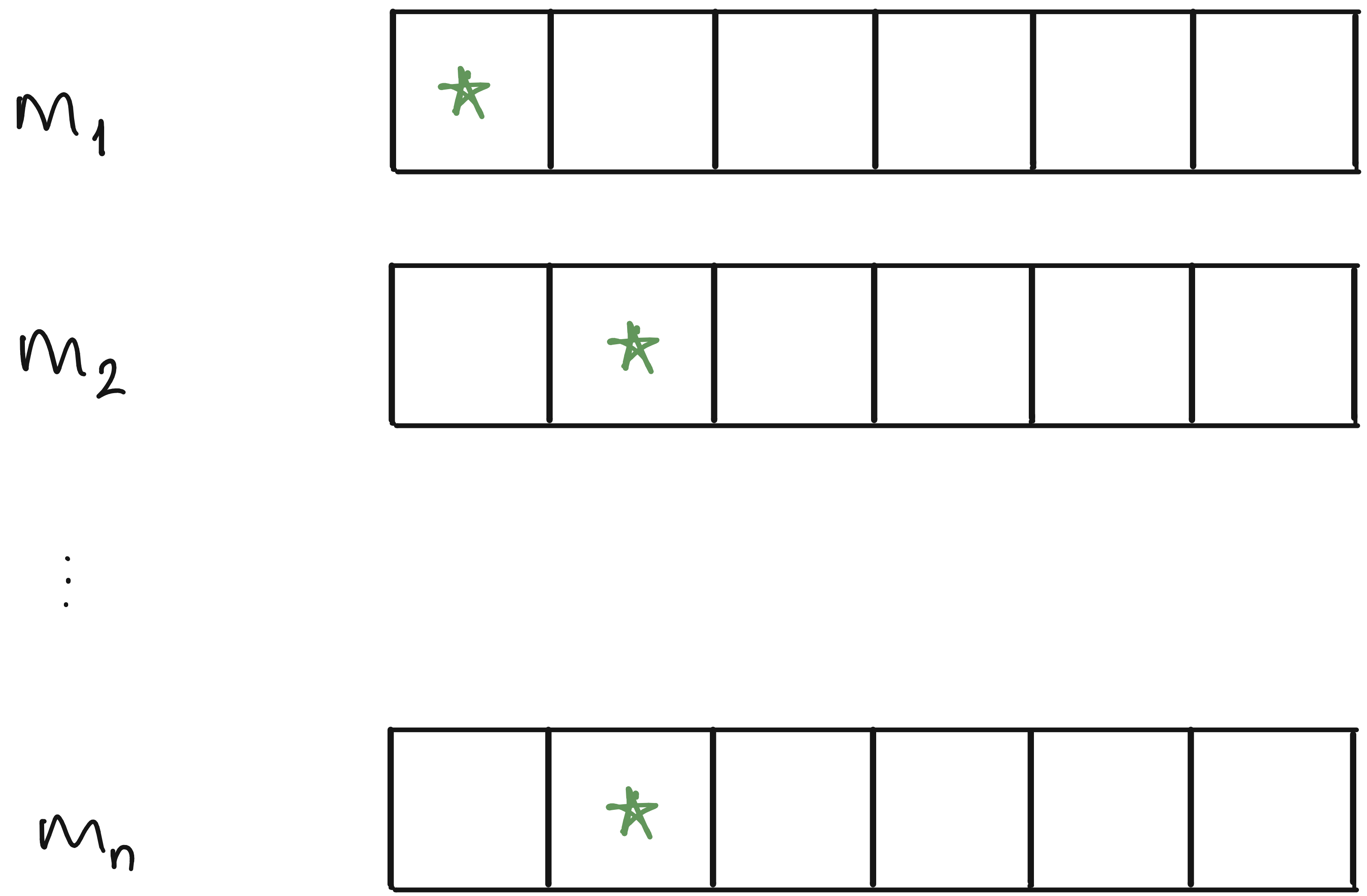
all "single" men "propose" to the highest-ranked woman who has not rejected them yet.

ADVANCED ALGORITHMS (W1, P7)

Greedy Algorithms

Stable Matchings III

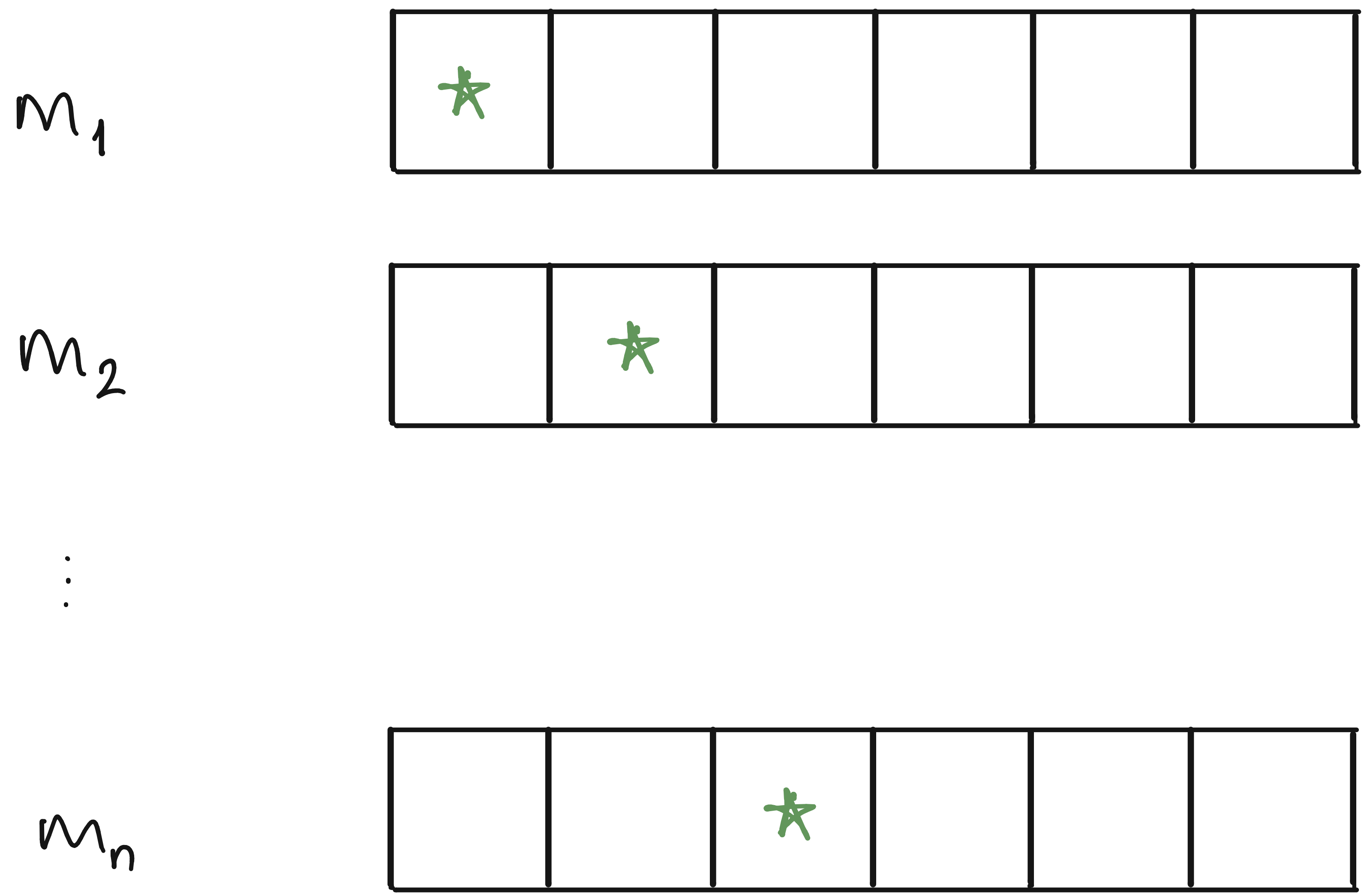
Men's perspective



all "single" men "propose" to the highest-ranked woman who has not rejected them yet.

Stable Matchings III

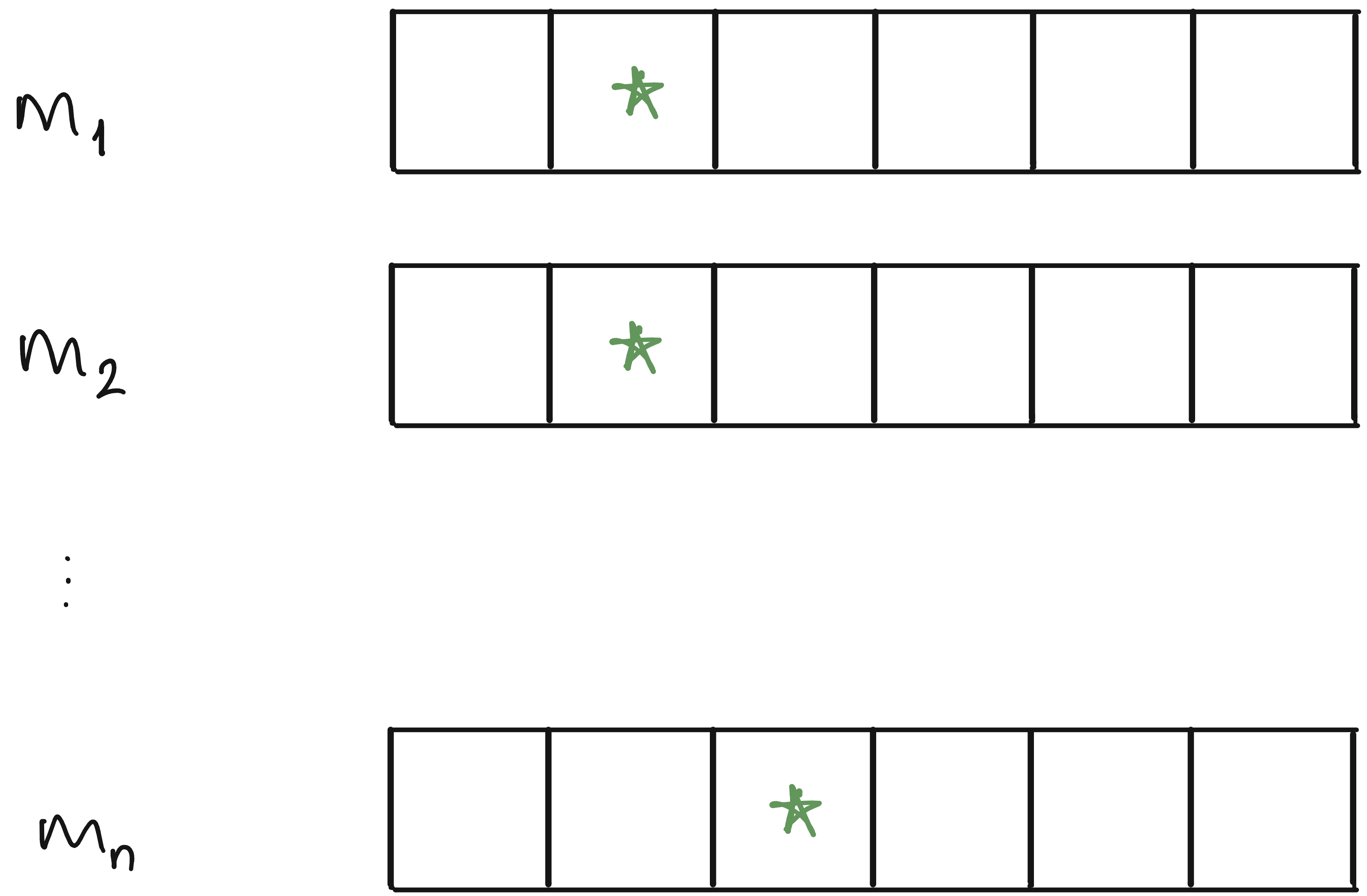
Men's perspective



all "single" men "propose" to the highest-ranked woman who has not rejected them yet.

Stable Matchings III

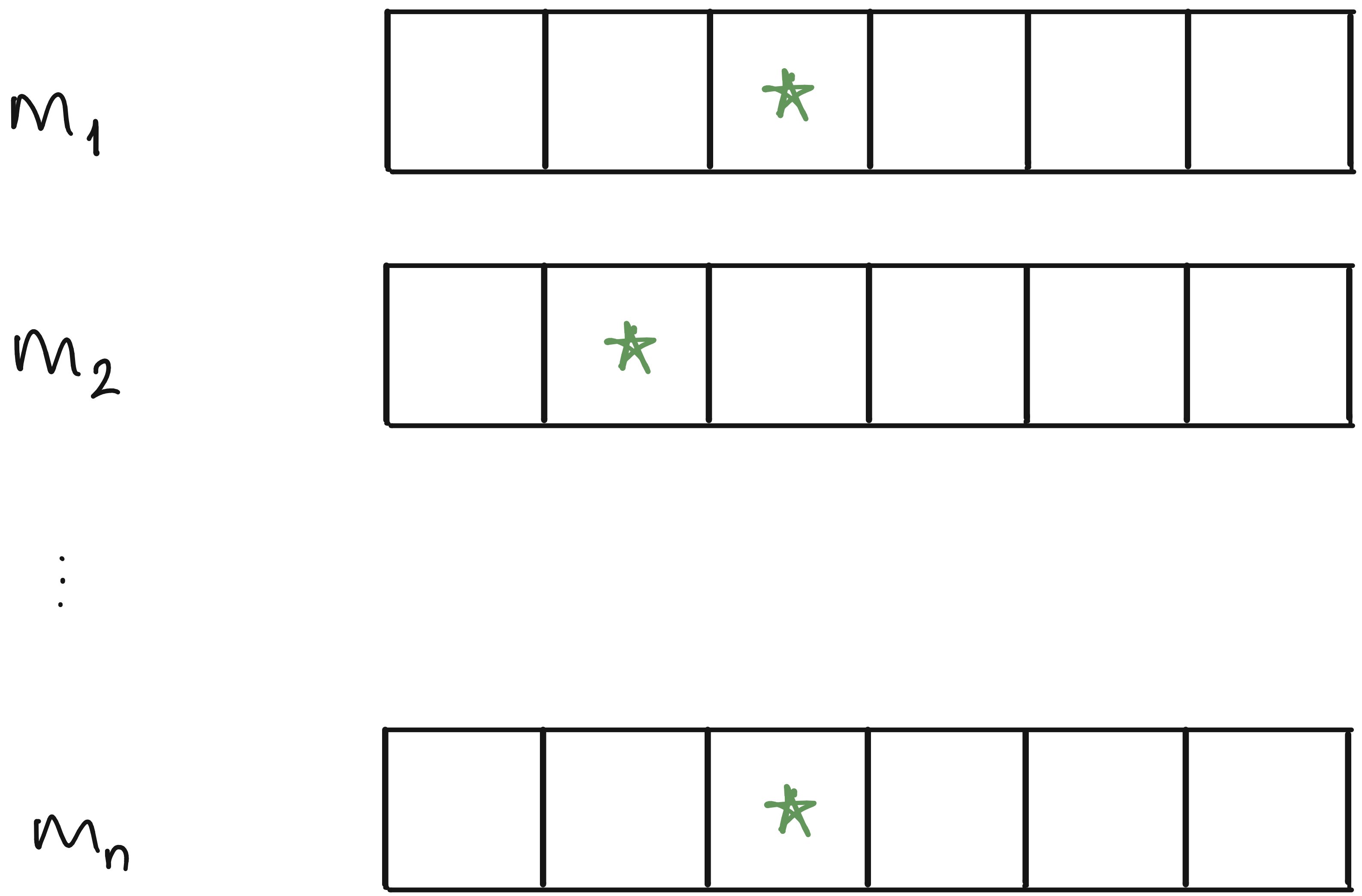
Men's perspective



all "single" men "propose" to the highest-ranked woman who has not rejected them yet.

Stable Matchings III

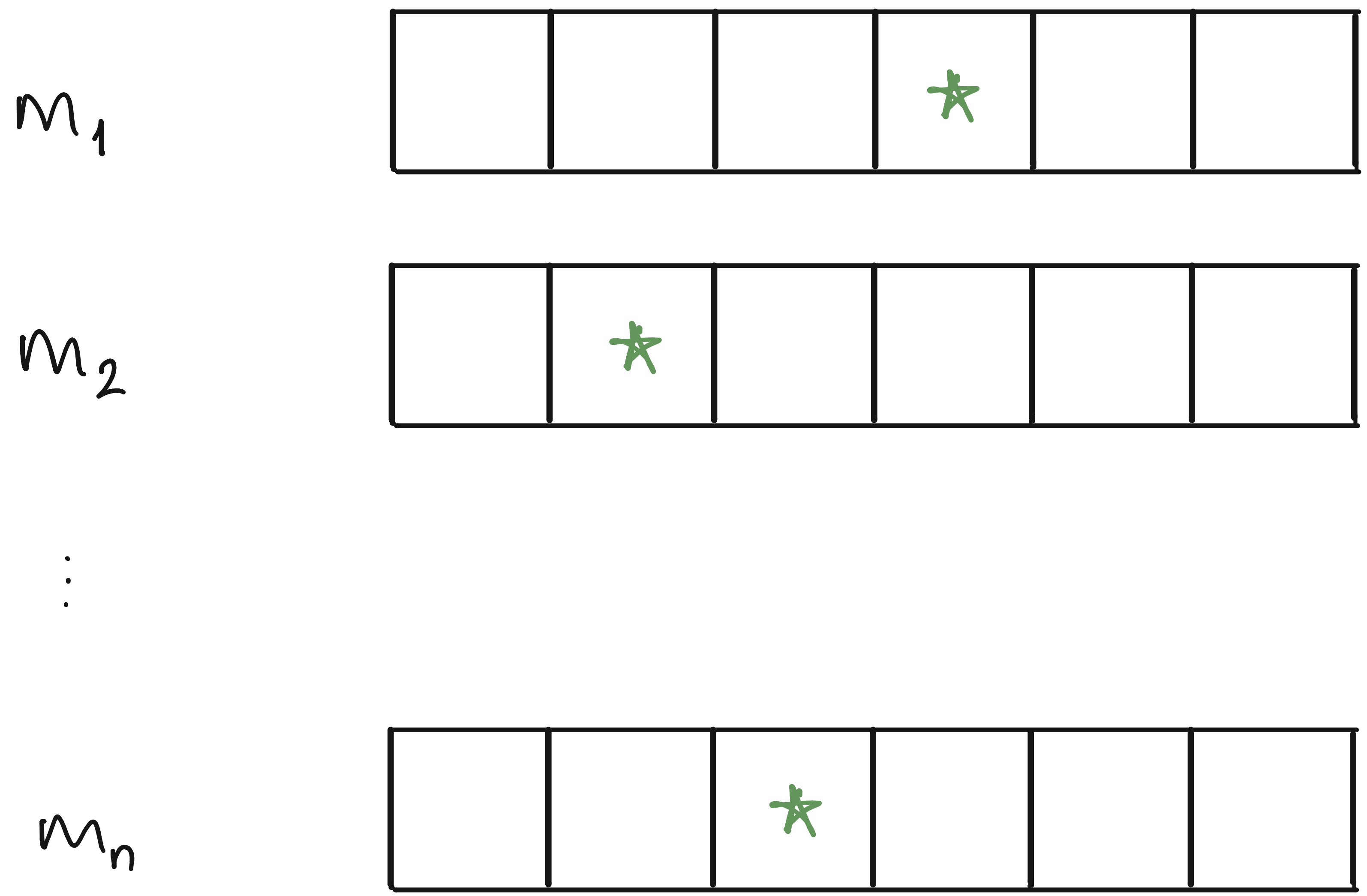
Men's perspective



all "single" men "propose" to the highest-ranked woman who has not rejected them yet.

Stable Matchings III

Men's perspective



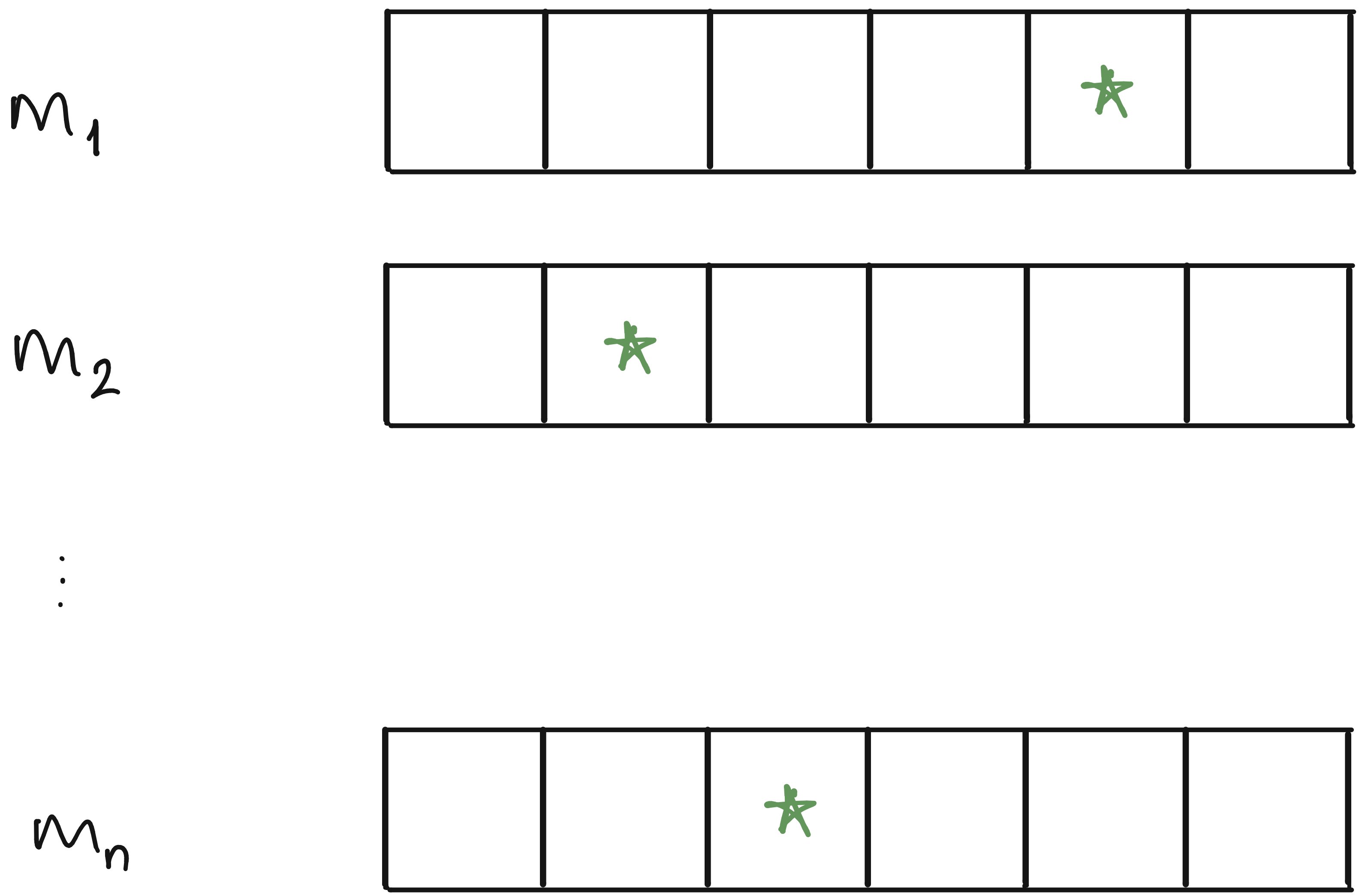
all "single" men "propose" to the highest-ranked woman who has not rejected them yet.

Advanced Algorithms (W1, P7)

Greedy Algorithms

Stable Matchings III

Men's perspective



all "single" men "propose" to the highest-ranked woman who has not rejected them yet.

Stable Matchings III

Termination

A man never proposes to the same woman more than once.

$\#$ proposals $\leq n^2$.

Stable Matchings III

The output is a matching

No man is single at the end

A man is matched w/ ≤ 1 woman at any stage of the algorithm.

Stable Matchings III

The output is stable

$w_7 > w_3$

$m_1 > m_5$

m_1 —— w_3

m_5 —— w_7

m_1 proposed to w_7 before w_3. w_7 eventually rejected m_1.