Greedy algorithm design paradigm

- It contructs a solution by considering one step at at a time
- At each step it chooses the locally best solution
- In some cases it contructs a globally best solution by repeatedly choosing the locally best option.

Advantages vs challenges

Advantages: simplicity :- Easy to describe
Efficiency :- efficiently implemented

challenges: Hard to design: once you have the right greedy approach design greedy algorithm is easy.

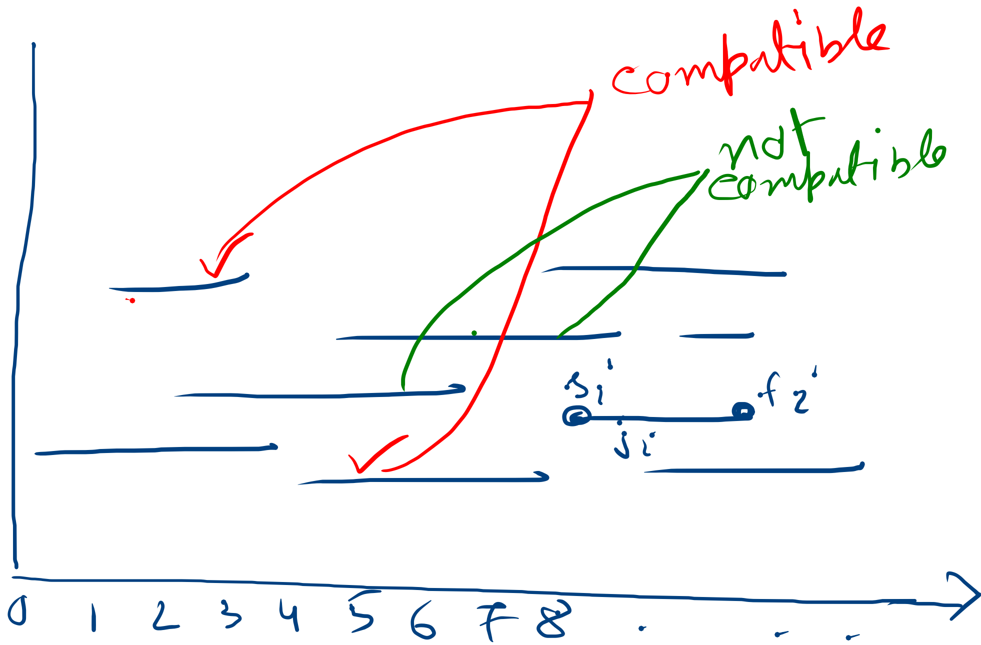
Hard to verify: The correctness often requires critical arguments.

Activity selection Problem / interval scheduling problem

maximum independent set problem in intervals given on a line.

Input: n jobs $J = \{j_1, j_2, \ldots, j_n\}$

each job $j_i \in J$, start time s_i, finish time f_i



Feasible solution:-

Two jobs are compatible if they do not overlap.

Find a subset of jobs that are mutually compatible

objective: maximise the size of the mutually compatible set of jobs.

Pick the job which finishes earlier.

Greedy rule.

- Select jobs one after another using some rule

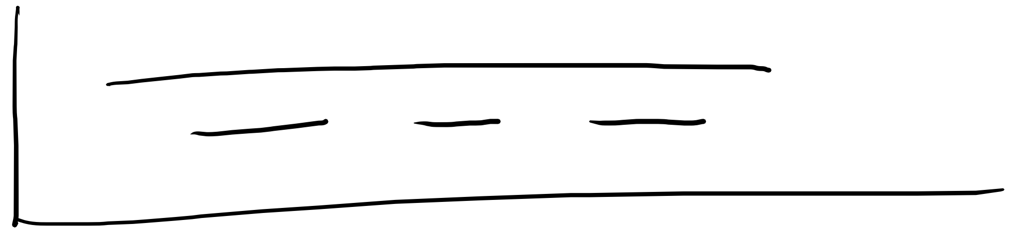
Rule 1 → Earliest start time

Rule 2 :- Smallest job first

Rule 3 :- smallest conflict job first

Rule 4: Earliest finish time

Rule 1: not optimum



Rule 2:

H. W.

Rule 3:

H. W.

Rule 4: Earliest finish time

Greedy rule:

- Initially J be the set of jobs and A ~~be~~ an empty set
- While J is not empty
    - choose a job $j \in J$ that has the smallest finish time
    - Add j to A
    - Delete all jobs from J that are not compatible with j

- Return A

$$O(n\log n + n)$$
$$= O(n\log n)$$

correctness :-

claim: A is a feasible solution.

Proof straightforward.

claim: A is optimum

A ← set of jobs return by the algorithm

opt ← largest set of pairwise non-overlapping jobs.

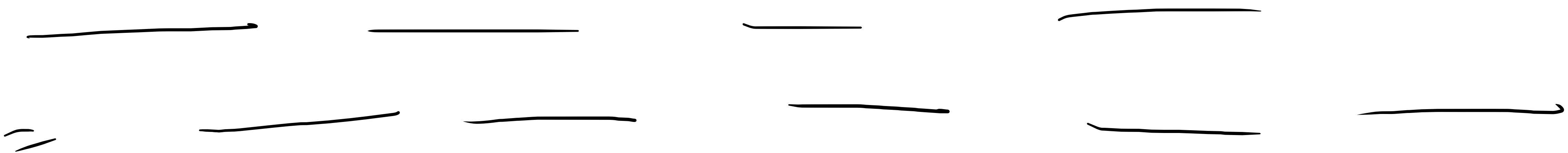
what we have to prove ?

A must ~~be~~ as large as opt

$$|A| = |opt|$$

$A = \{A_1, A_2, \ldots, A_K\}$
$Opt = \{O_1, O_2, \ldots, O_m\}$ Assume A and Opt are sorted.

A =
Opt =



Question: what is the relation between K & m

Answer: $K \leq m$

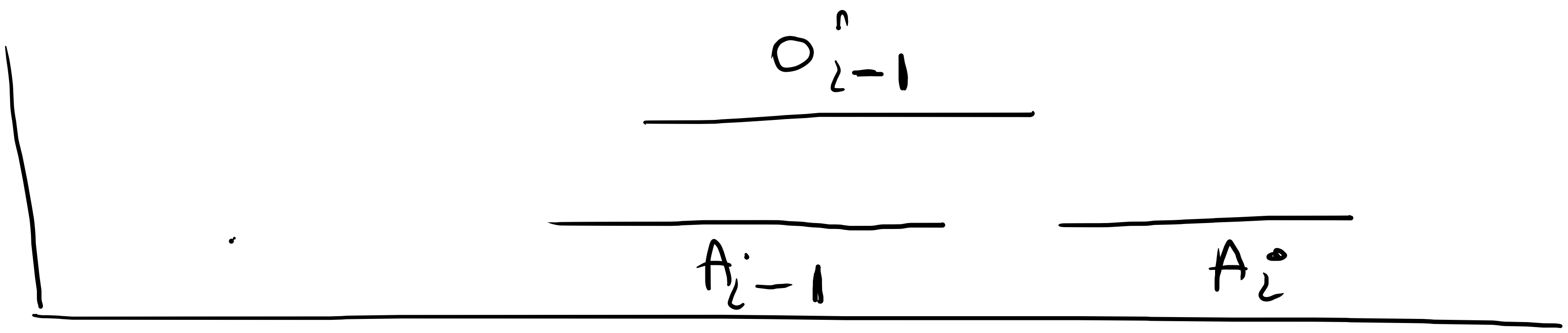
$$\boxed{\text{our aim:- } \cancel{K} = m}$$

claim for every $i \le K$
A_i finishes not later than O_i

Proof Prove using induction

Base case:- $i=1$ it is true A_1 finishes not later than O_1

I.H: A_{i-1} finishes not later than O_{i-1}



If O_i finishes before A_i then it would overlap with A_{i-1} that means it overlap with O_{i-1}

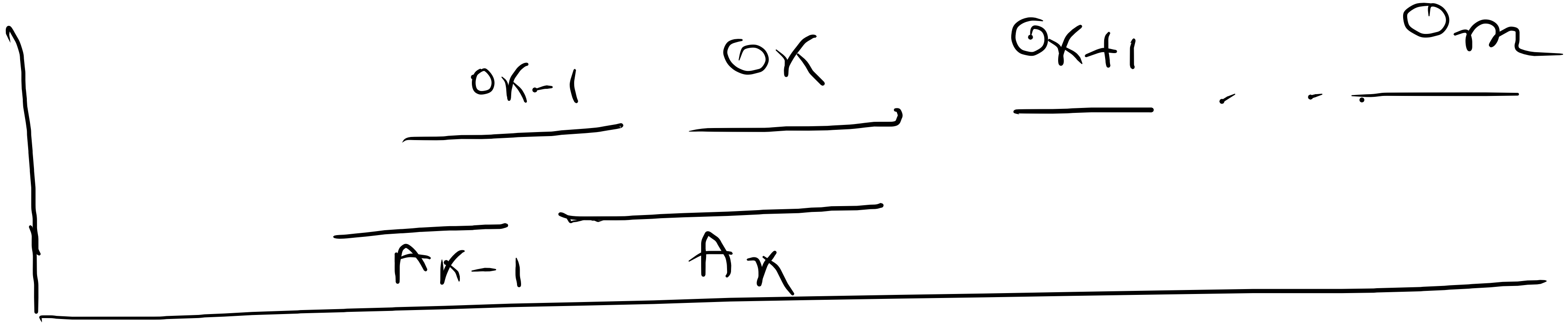
$\Rightarrow\!\Leftarrow$

Main claim: A is the optimum solution.

Proof: we need to ~~prove~~ $K = m$.

If $K = m$ we are done.

Assume that $K < m$.



O_{K+1} starts after O_K and consequently A_K

we could add O_{K+1} in A and obtain a bigger solution. $\Rightarrow : \Leftarrow$