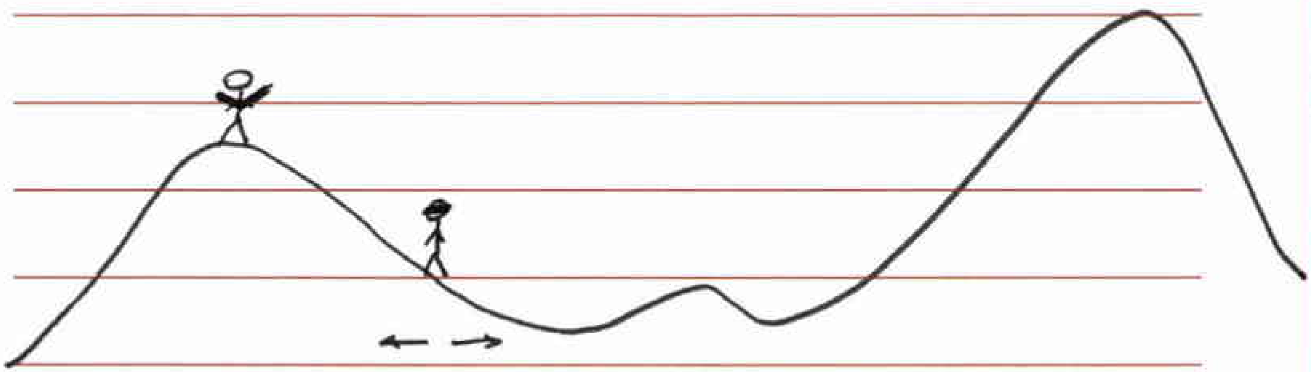
Greedy Part 1

Interval Scheduling
Fractional knapsack



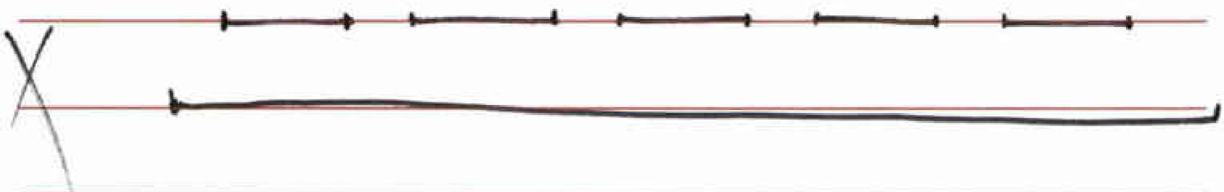
Interval Scheduling

Input: Set of requests $\{1..n\}$

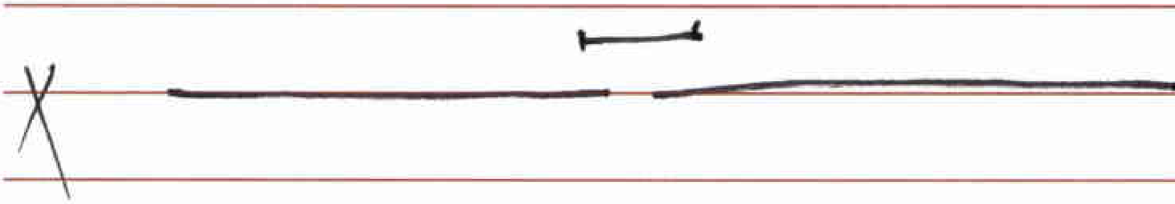
i^{th} request starts at $s(i)$ and ends at $f(i)$

Objective: to find the largest compatible subset of these requests.

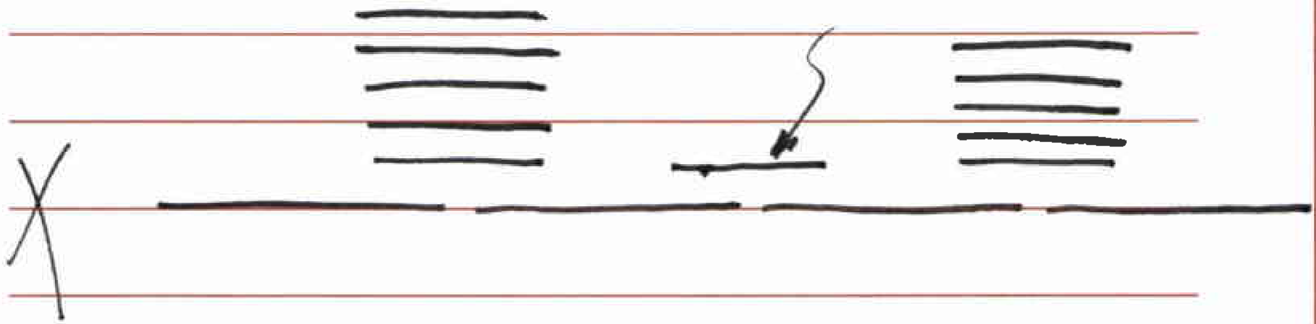
Try #1 "Earliest start time"



Try #2 "Smallest request first"



try #3 "smallest no. of overlaps"



Try #4 "Earliest finish time"

Solution:

Initially R is the complete set of requests & A is empty

While R is not empty

choose a request $i \in R$ that has the smallest finish time

Add request i to A

Delete all requests from R that are not compatible with request i

end while

Return A

Proof of correctness

✓ ① Show A is a compatible set

② Show A is an opt. set.

Say A is of size $\underline{k}$

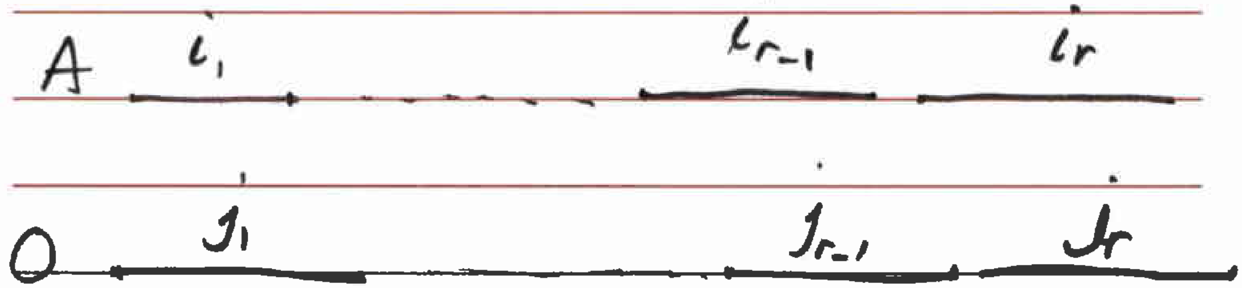
Say there is an opt. set O
we will prove that $|A| = |O|$

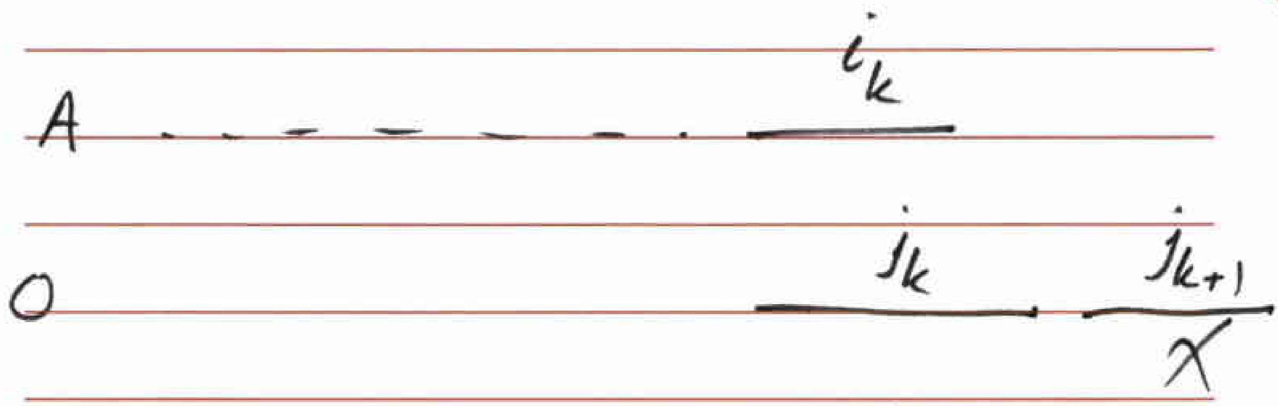
requests in A: $i_1 \dots i_k$

requests in O: $j_1 \dots j_m$

Prove that for all indices $r \leq k$

$\underline{\text{we have } f(i_r) \leq f(j_r)}$





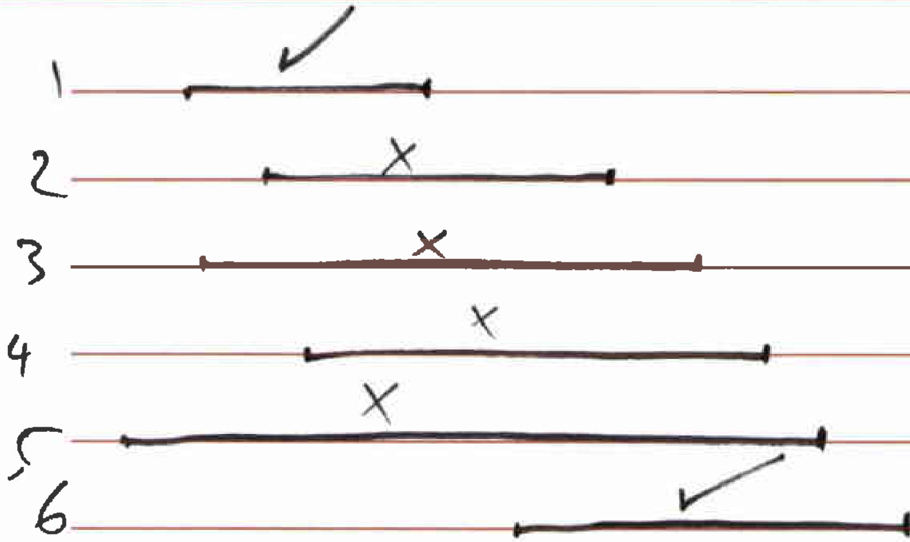
contradiction!

$\Rightarrow |A| = |O|$

Implementation

$O(n \log n)$
- sort requests in order of finish time and label in this order
  $f(i) \leq f(j)$ where $i < j$

$O(n)$
- select requests in order of increasing $f(i)$ always selecting the first the iterate through the intervals in order until reaching the first interval for which $s(j) \geq f(i)$



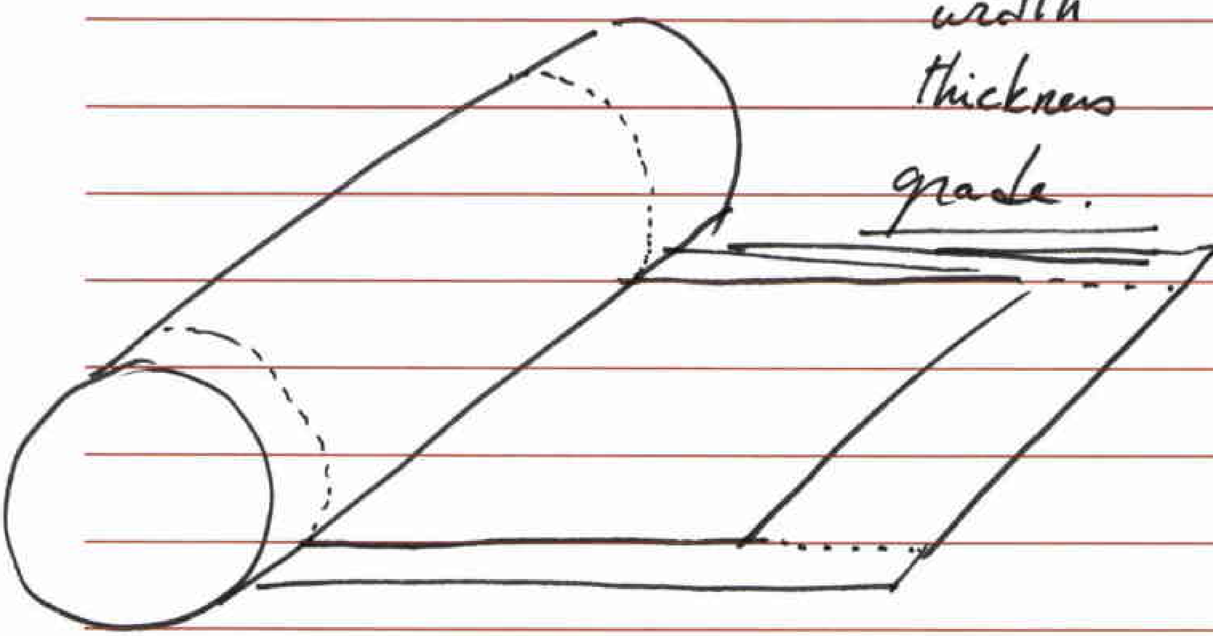
overall complexity $= O(n \lg n) + O(n)$

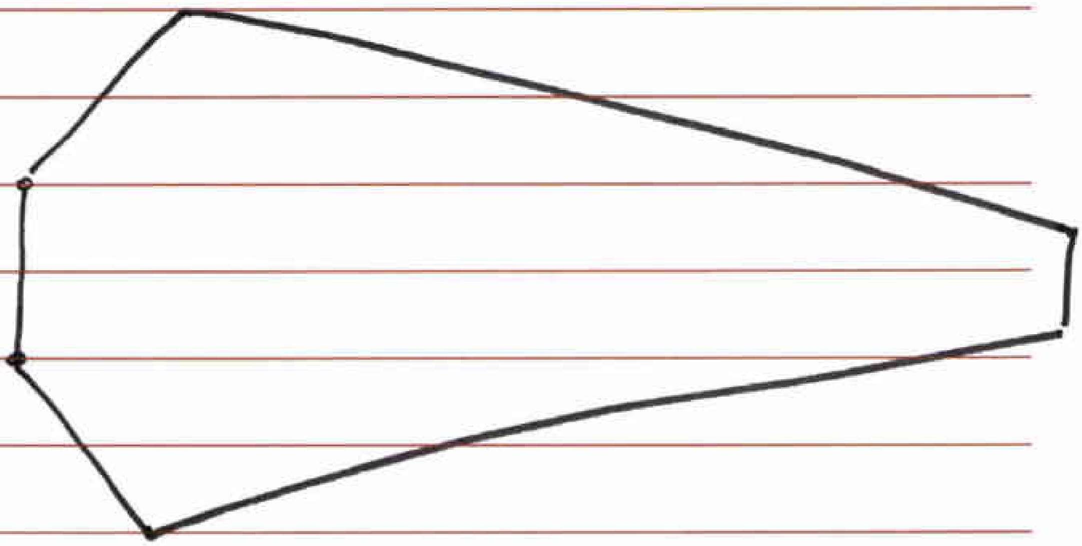
$= O(n \lg n)$

steel sheets

orders indicated

Qty
width
thickness
grade.





Fractional knapsack

knapsack has a weight capacity of W

we are given as input a set of n objects with weight w_i and value v_i.

Objective: Fill up the knapsack to its weight capacity such that the value of items in knapsack is maximized.

Ex. knapsack weight cap : 10

item#'s	1	2	3	4	5
values	10	20	15	2	8
weight	4	10	5	1	2
ratio of value/weight	2.5	2	3	2	4

Sorted list: 5, 3, 1, 2, 4

value of opt. sol. = $8 + 15 + 3/4 * 10 = 30.5$

Scheduling to Minimize lateness

- Requests can be scheduled at any time
- Each request has a deadline
- Notation $L_i = f(i) - d_i$

  lateness for ~~a~~ request $\underline{i}$

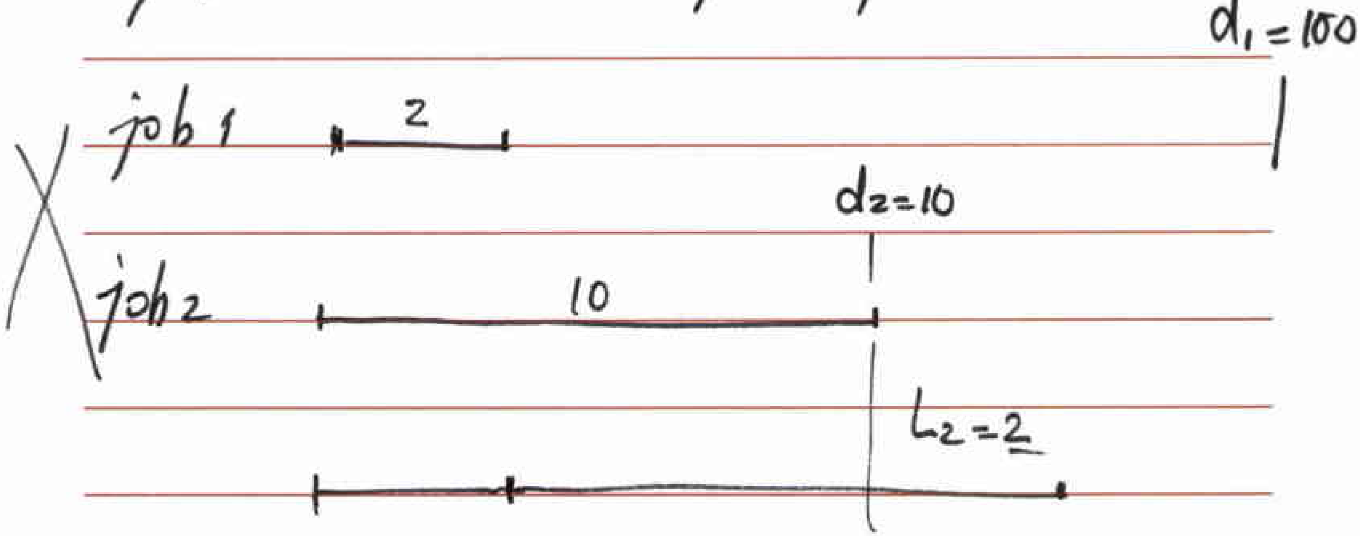
Goal: Minimize the Max. lateness

$$L = Max_i L_i$$

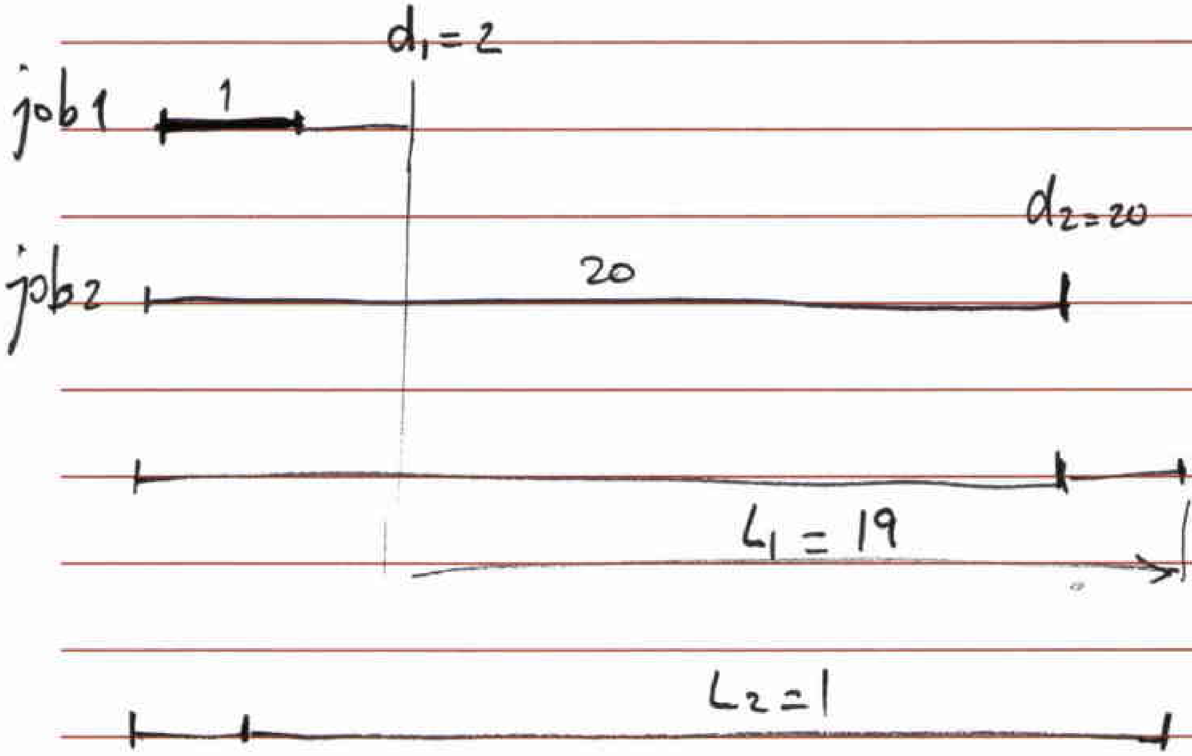
Sol. 1 job $\underline{1}$ late by $\underline{5}$ hrs,
job $\underline{2}$ late by $\underline{6}$ hrs.

Sol. 2 job 1 late by $\underline{0}$ hrs
job 2 late by $\underline{7}$ hrs.

try #1 "shortest jobs first"



try #2 "smallest slack time first"

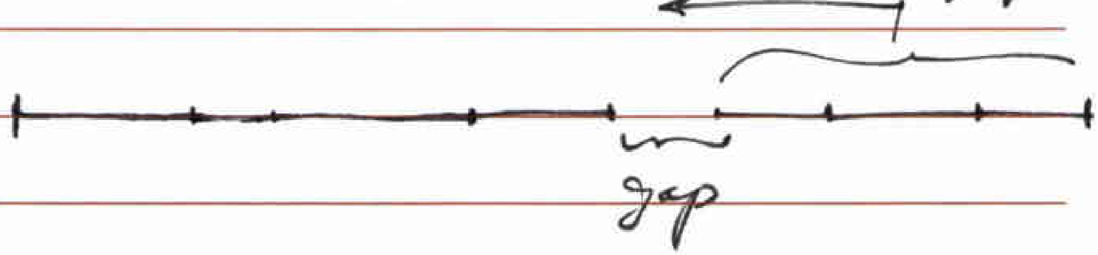


try #3 "Earliest Deadline first"

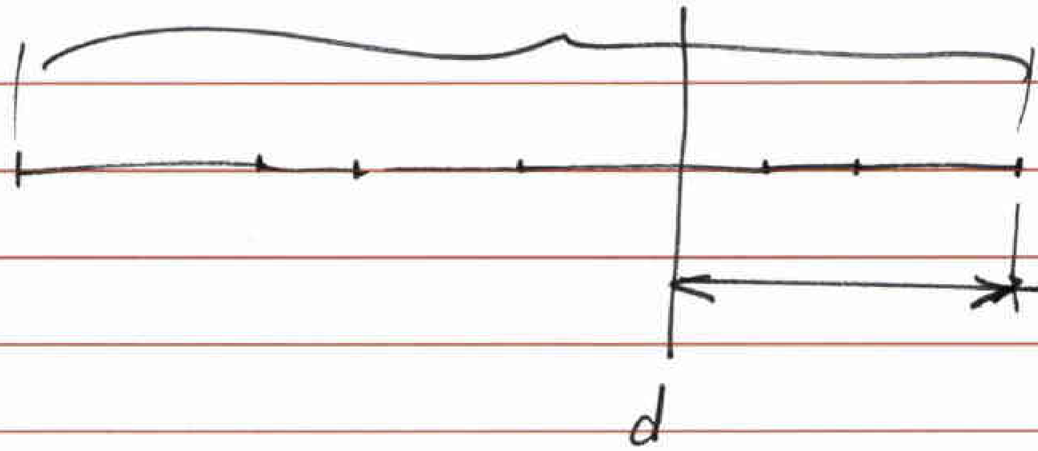
Solution: sched. jobs in order of their deadline w/o any gaps between jobs.

Proof of Correctness

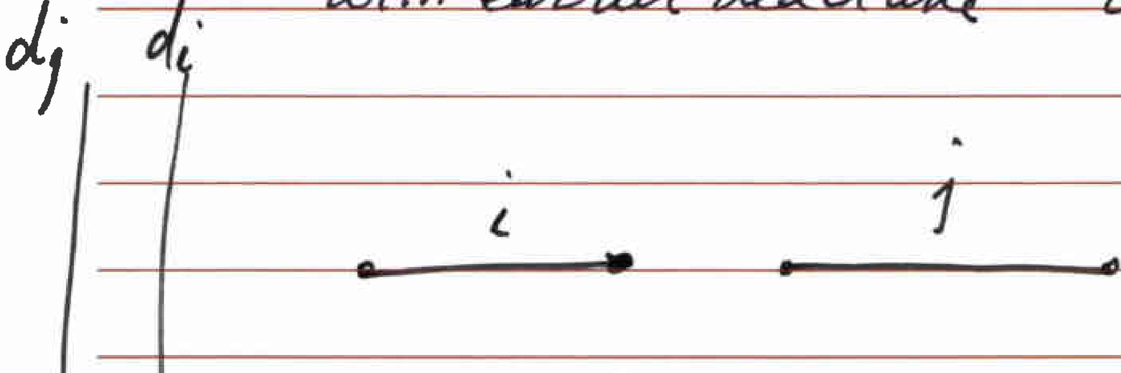
① There is an opt. solution w/ no gaps ✓



② jobs w/ identical deadlines
Can be sched'd in any order
without affecting Max. lateness



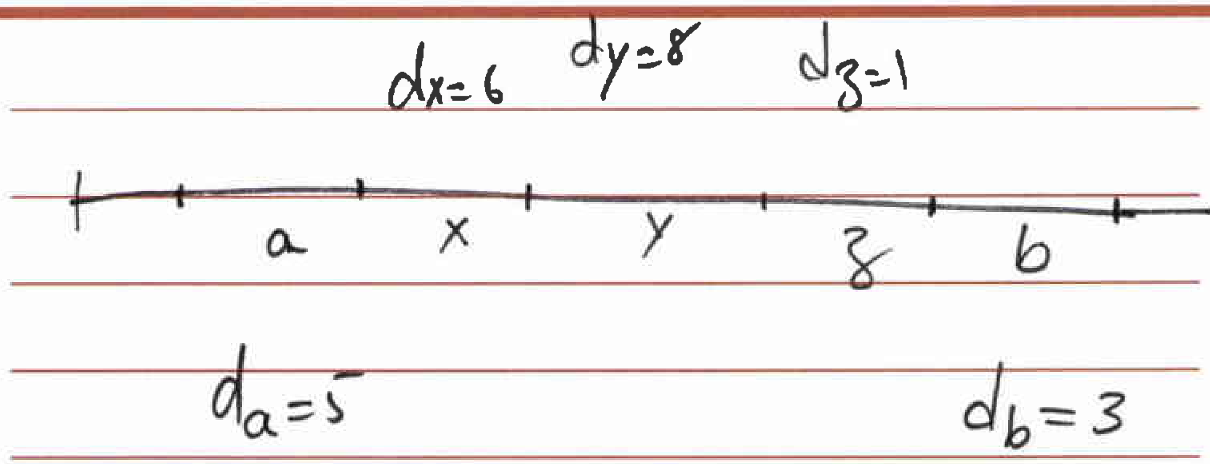
③ <u>Def.</u> Sched. A' has an <u>inversion</u>
if a job i w/ deadline d_i
is scheduled before job j
with earlier deadline $d_j < d_i$

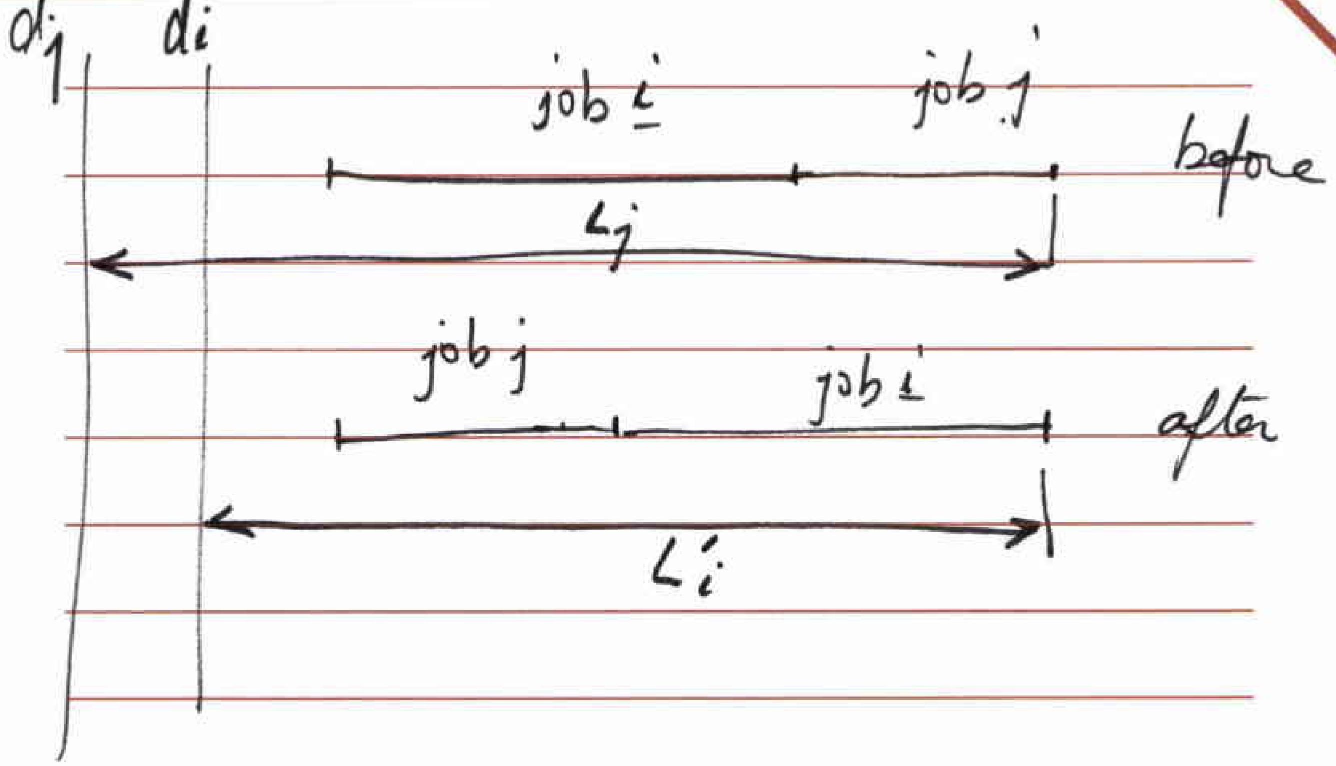


observation: By def. A has
no inversions.

④ All sched's w/ no inversions and no idle time have the same Max. lateness.

⑤ There is an optimal schedule that has no inversions and no idle time.





(6) - Proved that there exists an opt. sched. w/ no inv's & no idle time

- Proved that all sched's w/ no inv. & no idle time have the same Max. lateness

- Our greedy alg. produces one such sol. $\Rightarrow$ it will be an opt. sol.