# COL 351:
# Analysis and Design of Algorithms

**Wednesday, 11 August 2021**

# Today's Lecture

**Job-Scheduling Problem**
(can be solved using Greedy Strategy)

# Job Scheduling

Motivational Problem: Watch maximum number of show on New Year Eve

There are $n$ shows coming on Television on various channels.

Each show has a start-time and end-time, however, different shows overlap.

**GOAL :** Design a strategy to **watch maximum number of shows**. ← max count
(A show if watched must be played on Television from starting till end).

# Job Scheduling

## Formal Definition

start time

end time

Given: A collection $n$ jobs, $\{J_1 = [s_1, t_1], \ldots, J_n = [s_n, t_n]\}$.
A single server.

Constraint: If job $J_i$ is scheduled on server, then it occupies the server for time-interval $[s_i, t_i]$

Aim: Find a maximum subset $A$ of $J_{set} = \{J_1, \ldots, J_n\}$ of non-overlapping jobs.

set of all jobs

9 jobs

Example:

$J_1 = (1.5,6),\ J_2 = (8,9.5),\ J_3 = (3,5),\ J_4 = (6.5,9),\ J_5 = (2,3.5),\ J_6 = (5,9.5),\ J_7 = (1,5.5),\ J_8 = (8,8.5),\ J_9 = (4,7.5)$

start

end

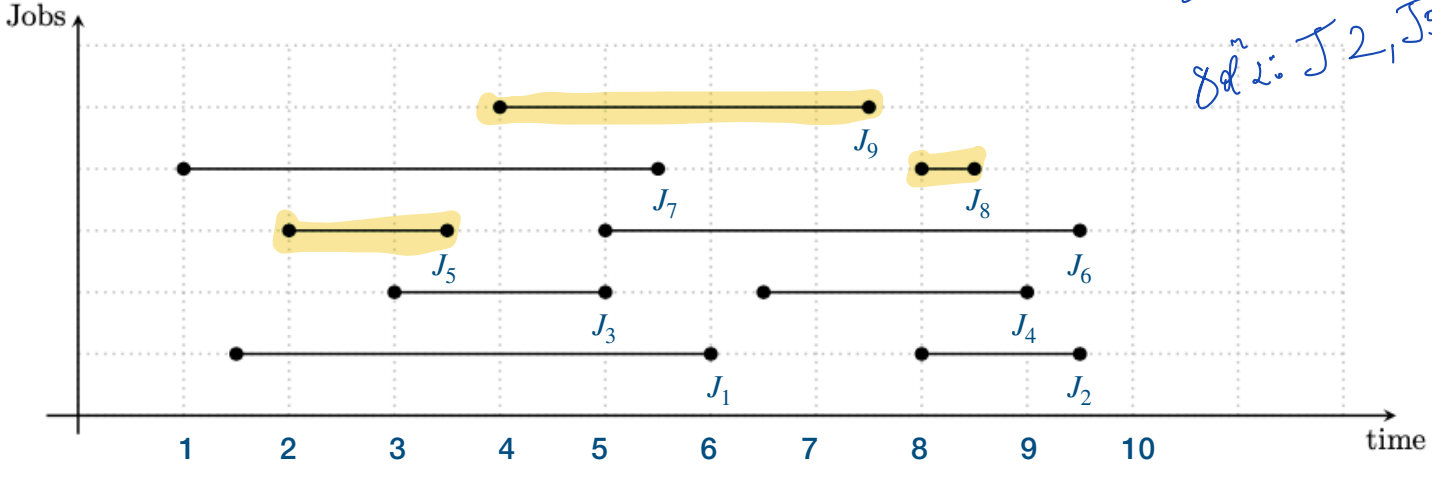
# Pictorial Representation

Example:

$J_1 = (1.5,6)$, $J_2 = (8,9.5)$, $J_3 = (3,5)$, $J_4 = (6.5,9)$, $J_5 = (2,3.5)$, $J_6 = (5,9.5)$, $J_7 = (1,5.5)$, $J_8 = (8,8.5)$, $J_9 = (4,7.5)$

Sol 1: J5 J8 J9

Sol 2: J2, J5, J9



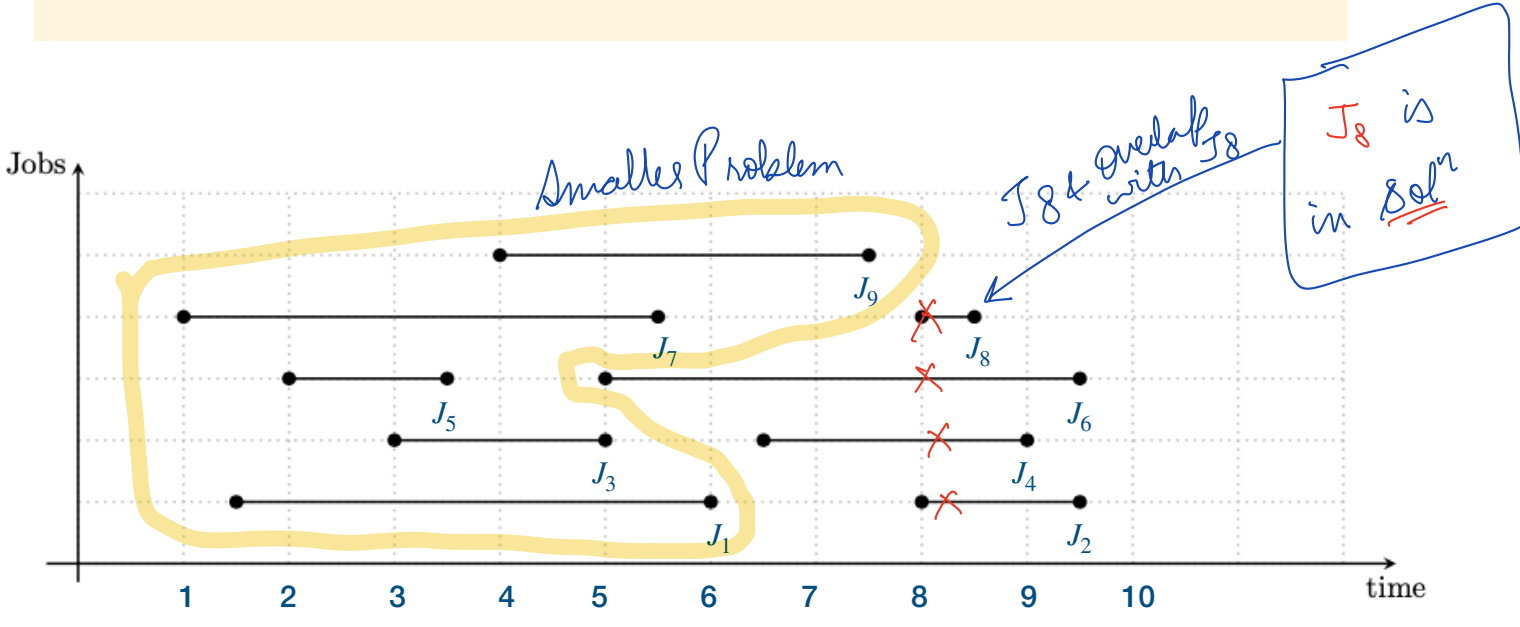

Kuldeep — Ends early.

Rajdeep — All possible sets — $\Omega(2^n)$

# Greedy Approach

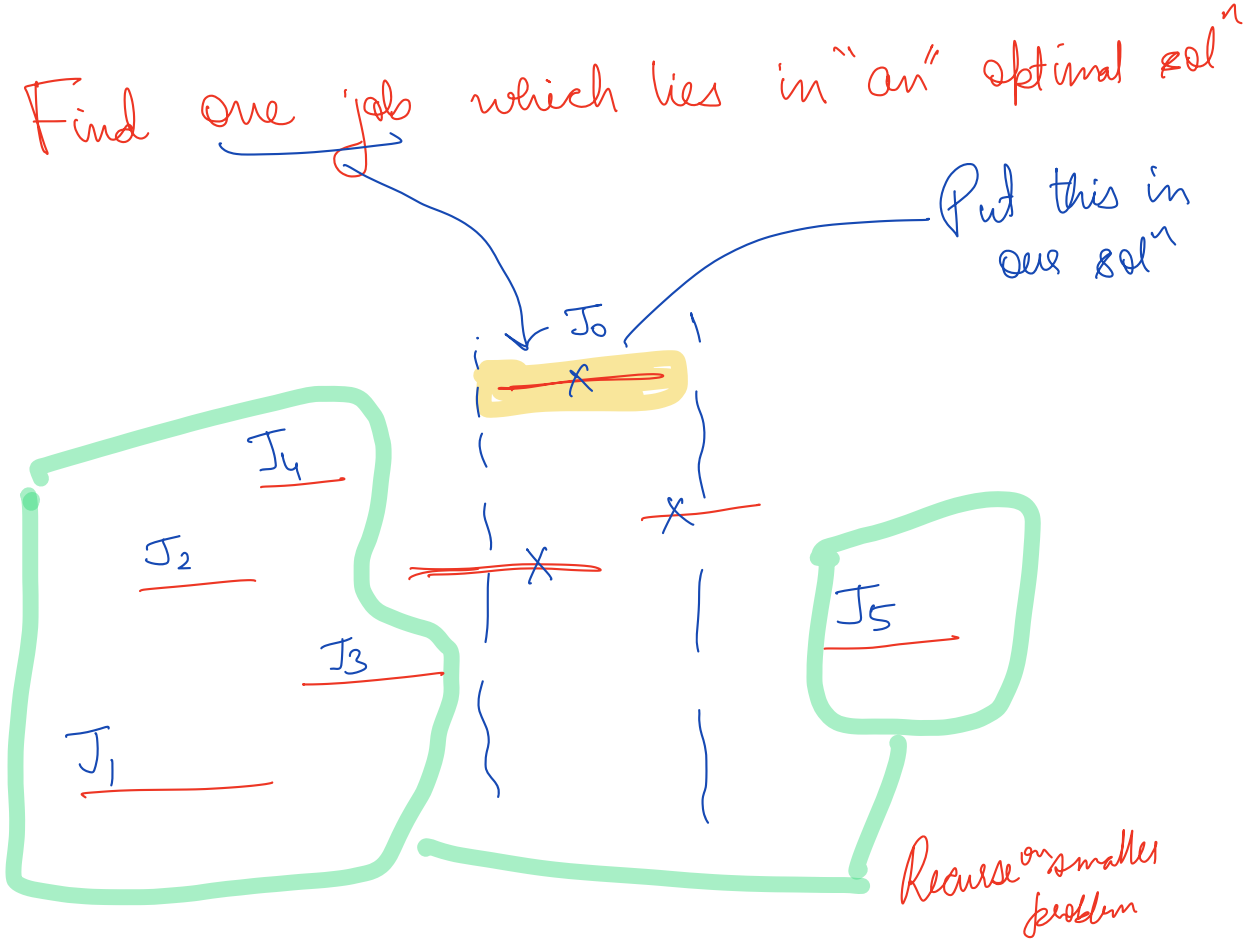
find one element that lie in soln ⇒ Reduce the problem size

GRREDY APPROACH: Just find one job $J_i$ that lies in an optimal solution.



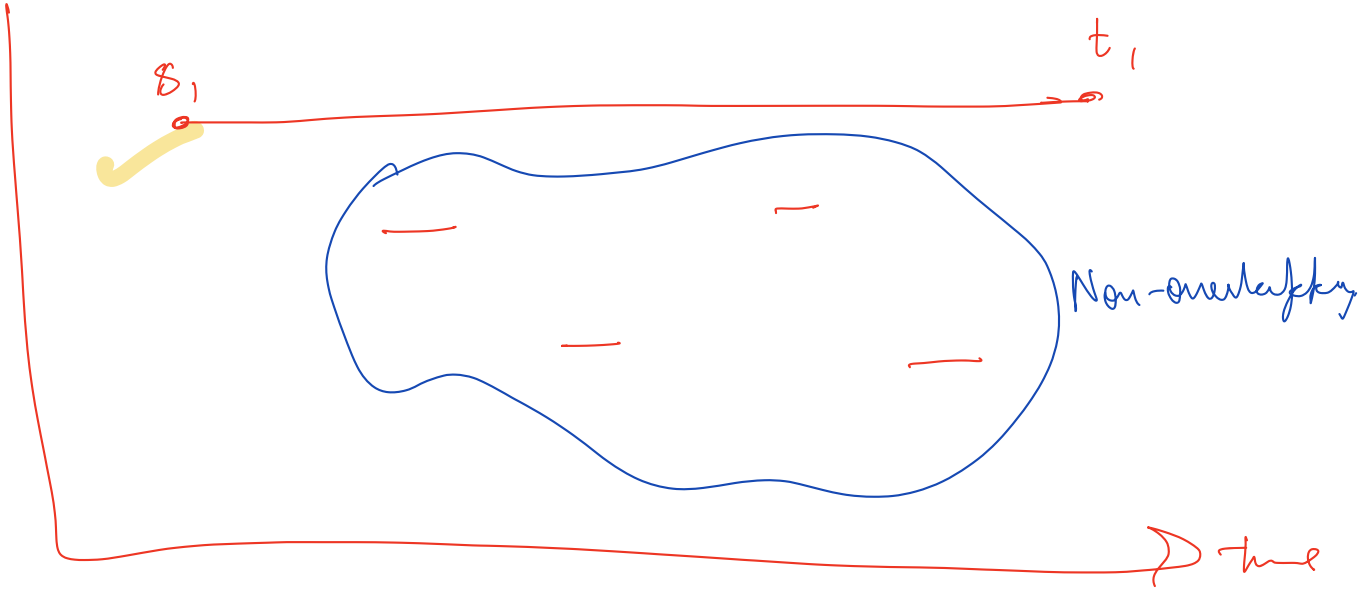
# Idea

Find one job which lies in "an" optimal sol$^n$



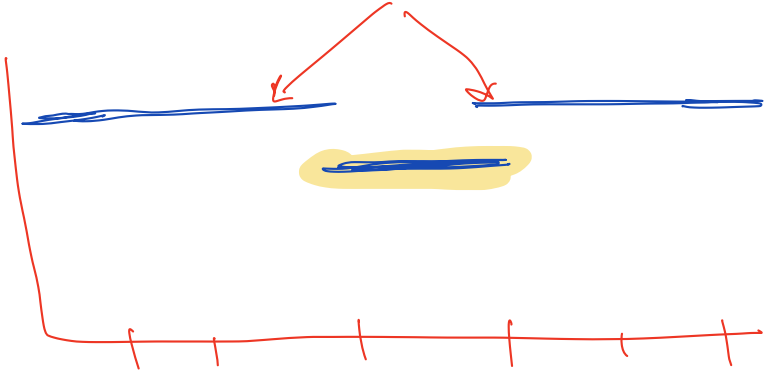
# Which Greedy Strategy works?

Ques. Can we select a job that **arrives first**, i.e., has smallest $s_i$? NO



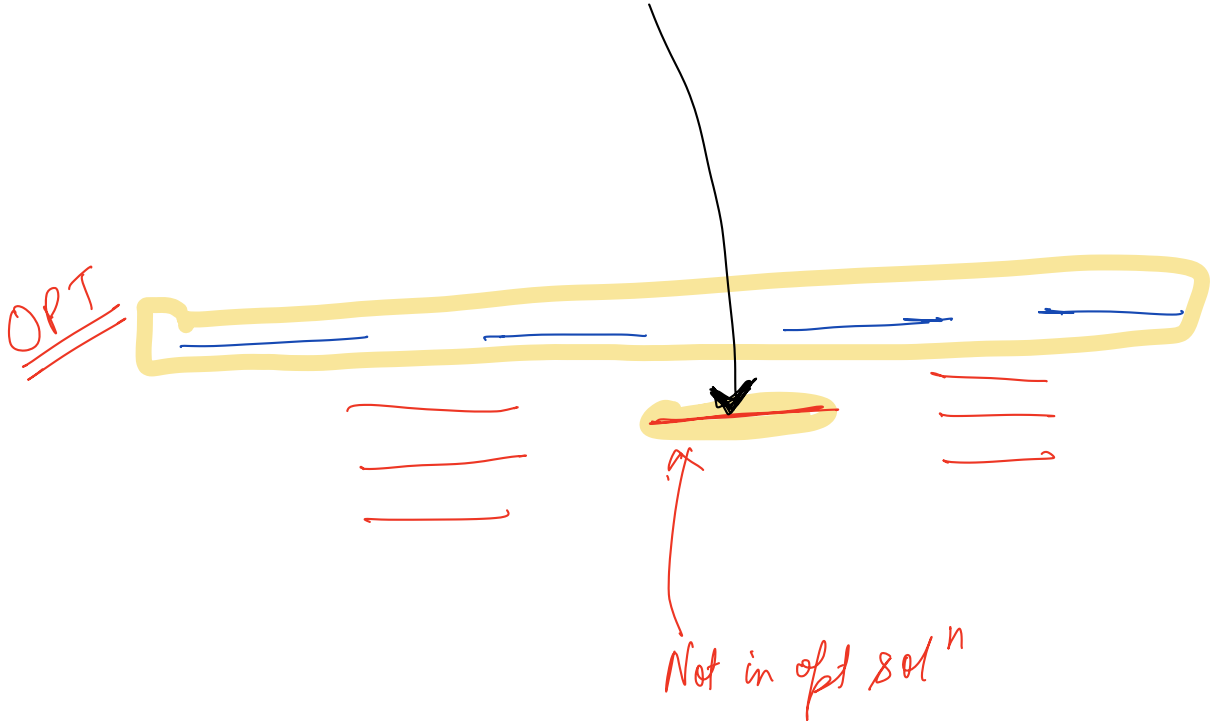
# Which Greedy Strategy works?

Ques. Can we select a job with **smallest duration**, i.e., with smallest value of $(t_i - s_i)$? NO



# Which Greedy Strategy works?

Ques. Can we select a job with **minimum overlap**?



# Which Greedy Strategy works?

Ques. Can we select a job that **finishes earliest**, i.e., with smallest value of $t_i$?

Rajdeep : ∃ opt soln that contains

# Which Greedy Strategy works?

Lemma: Let $J_0 \in J_{set}$ be job with earliest finish time.

Then there exists a optimal (i.e. of maximum possible size) subset of non-overlapping jobs in $J_{set}$, say $A_0$, that contains $J_0$

Proof:

Nalin

① Let $A^*$ be any opt soln and in $A^*$ let $J_1$ be job with earliest finishing time

② $(A^* \setminus J_1) \cup \{J_0\}$ has non-overlapping jobs

③ $|A^*| = |(A^* \setminus J_1) \cup \{J_0\}| \Rightarrow (A^* \setminus J_1) \cup \{J_0\}$ proves

# Which Greedy Strategy works?

Lemma: Let $J_0 \in J_{set}$ be job with earliest finish time.

Then there exists a optimal (i.e. of maximum possible size) subset of non-overlapping jobs in $J_{set}$, say $A_0$, that contains $J_0$

Proof: Let $A \subseteq J_{set}$ be largest set of non-overlapping jobs. Further, let

- $a_0$ be the job in $A$ with earliest finish time.

Then, jobs in $\left((A \backslash \{a_0\}) \cup \{J_0\}\right)$ are non-overlapping. (Why?)

As size of $\left((A \backslash \{a_0\}) \cup \{J_0\}\right)$ and $A$ is identical, $\left((A \backslash \{a_0\}) \cup \{J_0\}\right)$ is an opt solution containing $J_0$.

# Algorithm

Trivial Implementation = $O(|A| * n)$

Running $|A|$ times

$A = \phi$

$A = \{J_5\}$

$A = \{J_5, J_9\}$

$A = \{J_5, J_9, J_8\}$

Set $A = \phi$.

While $J_{set} \neq \phi$:

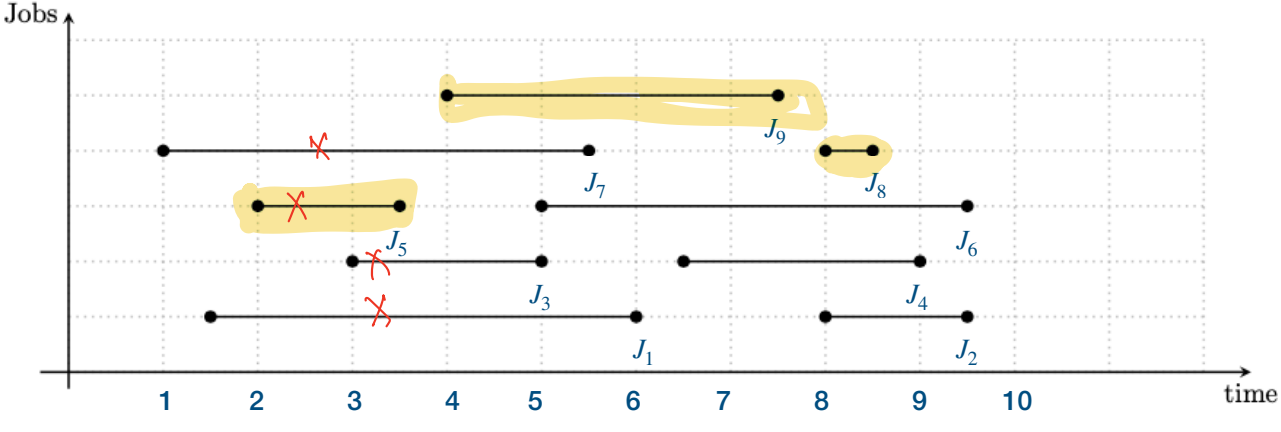
$O(n)$ time

Find a job $J_0 \in J_{set}$ with earliest finish time, and **add** it to set A.

Remove $J_0$ and all those jobs that overlap with $J_0$ from $J_{set}$.

Return A.

Better Implementation => Sort according to $t_i$



# Algorithm

Set $A = \phi$.

While $J_{set} \neq \phi$:

Find a job $J_0 \in J_{set}$ with earliest finish time, and **add** it to set A.

Remove $J_0$ and all those jobs that overlap with $J_0$ from $J_{set}$.

Return A.

} decreasing the size of |J set| by at least 1.

Q: [Why algo terminates?

Implementation time?

How to prove correctness?

Proof: While loop runs at most n number of times.

# Algorithm Correctness

Overlap $J_0$ = { Set of all jobs overlapping with $J_0$ }

Theorem 1: Let $J_0 \in J_{set}$ be job with earliest finish time, and $J'_{set} = J_{set} \setminus \text{Overlap}(J_0)$.

Then

$$OPT(J_{set}) = OPT(J'_{set}) + 1.$$

① $opt(J_{set}) \geq opt(J'_{set}) + 1$

Proof

Lemma: ∃ opt soln containing $J_0$

① Suppose $A'$ is opt soln of $J'_{set}$

② $A' \cup J_0$ is non-overlapping. (By def$^n$ of $J'_{set}$)

③ $|A' \cup J_0| = 1 + |A'| = 1 + opt(J'_{set})$

$\Rightarrow opt(J_{set}) \geq 1 + opt(J'_{set})$

② $opt(J'_{set}) \geq opt(J_{set}) - 1$

Proof

Lemma: ∃ opt soln containing $J_0$

① Let $A$ be opt sol of $J_{set}$ having $J_0$

② $(A \setminus J_0)$ — non overlapping jobs

subset of $J'_{set}$

③ $opt(J'_{set}) \geq |A \setminus J_0|$

$= |A| - 1$

$= opt(J_{set}) - 1$

# Exercises

- Design an algorithm to find an optimal scheduling when you are given two servers.

  $k$ = 2 servers.

- Design an algorithm to check whether a collection of $n$ given jobs has a unique optimal scheduling, with respect to one fixed server.

Next class
Graphs

H.W. Suppose $J_{set} = (J_1 = [s_1, t_1], \ldots, J_n = [s_n, t_n])$

and $t_1 \le t_2 \le \ldots \le t_{n-1} \le t_n$.

Task: Design $O(n)$ time algo for opt-soln.

Examples

Server

A. Pentes

2PM - 3:30PM

4PM - 5PM

3PM - 4PM

Cult-Fest

Audi - Server