CS 381 – FALL 2019

Week 5.3, Friday, Sept 20

Homework 3 Due: September 23rd , 2019 @ 11:59PM on Gradescope
Homework 2: Solutions available on Piazza
Midterm 1: September 25 (evening)

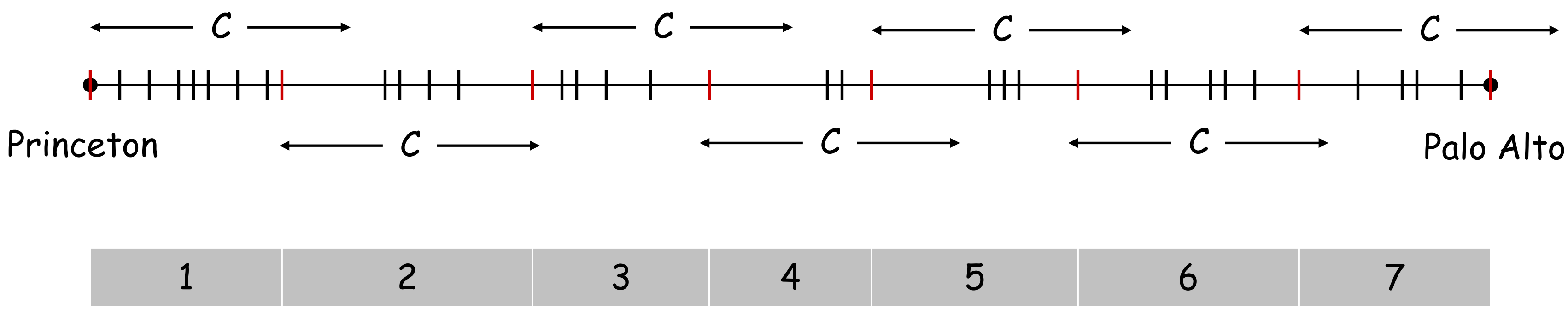
Selecting Breakpoints

Selecting Breakpoints

Selecting breakpoints.

- Road trip from Princeton to Palo Alto along fixed route.
- Refueling stations at certain points along the way.
- Fuel capacity = C.
- Goal: makes as few refueling stops as possible.

Greedy algorithm. Go as far as you can before refueling.



Selecting Breakpoints: Greedy Algorithm

Truck driver's algorithm.

```
Sort breakpoints so that: 0 = b0 < b1 < b2 < ... < bn = L

S ← {0}        ← breakpoints selected
x ← 0          ← current location

while (x ≠ bn)
   let p be largest integer such that bp ≤ x + C
   if (bp = x)
      return "no solution"
   x ← bp
   S ← S ∪ {p}
return S
```

Implementation. $O(n \log n)$

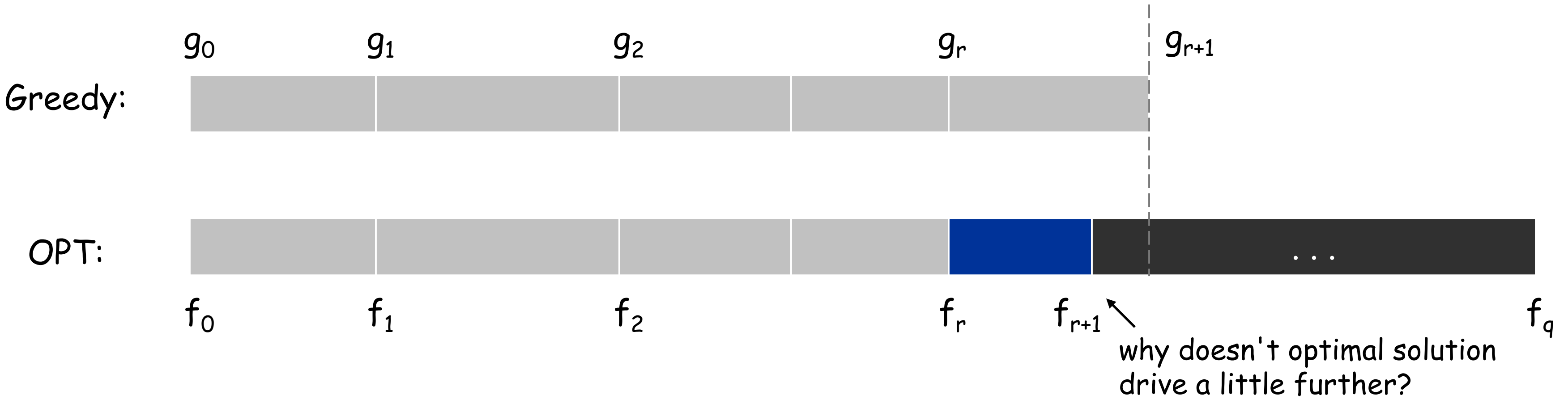
- Use binary search to select each breakpoint p.

Selecting Breakpoints: Correctness

Theorem. Greedy algorithm is optimal.

Pf. (by contradiction)

- Assume greedy is not optimal, and let's see what happens.
- Let $0 = g_0 < g_1 < \ldots < g_p = L$ denote set of breakpoints chosen by greedy.
- Let $0 = f_0 < f_1 < \ldots < f_q = L$ denote set of breakpoints in an optimal solution with $f_0 = g_0$, $f_1 = g_1$, . . . , $f_r = g_r$ for largest possible value of r.
- Note: $g_{r+1} > f_{r+1}$ by greedy choice of algorithm.

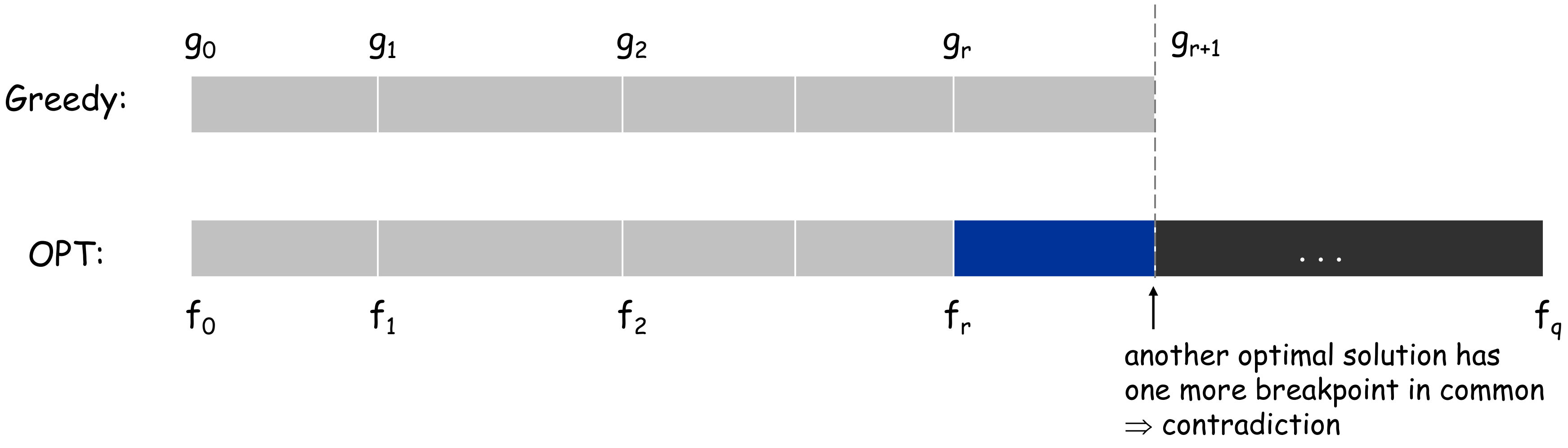


Selecting Breakpoints: Correctness

Theorem. Greedy algorithm is optimal.

Pf. (by contradiction)

- Assume greedy is not optimal, and let's see what happens.
- Let $0 = g_0 < g_1 < \ldots < g_p = L$ denote set of breakpoints chosen by greedy.
- Let $0 = f_0 < f_1 < \ldots < f_q = L$ denote set of breakpoints in an optimal solution with $f_0 = g_0$, $f_1 = g_1$, . . . , $f_r = g_r$ for largest possible value of r.
- Note: $g_{r+1} > f_{r+1}$ by greedy choice of algorithm.

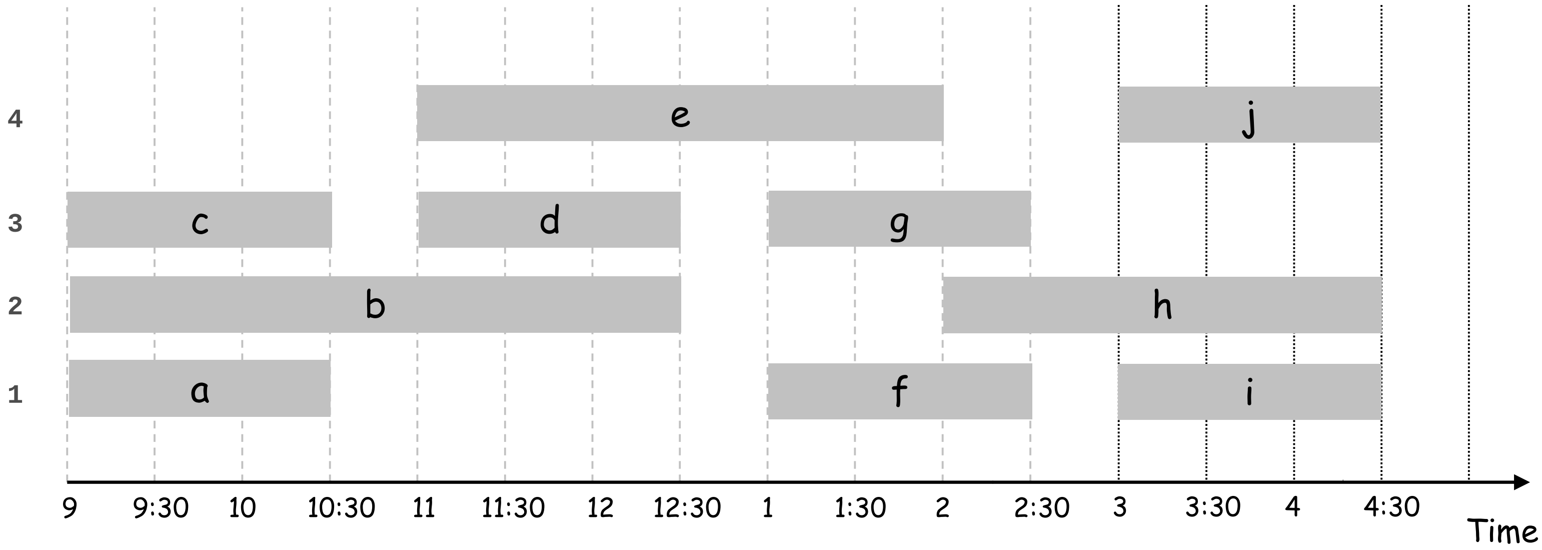


4.1 Interval Partitioning

Interval Partitioning

Interval partitioning.
- Lecture j starts at s_j and finishes at f_j.
- Goal: find minimum number of classrooms to schedule all lectures so that no two occur at the same time in the same room.

Ex: This schedule uses 4 classrooms to schedule 10 lectures.

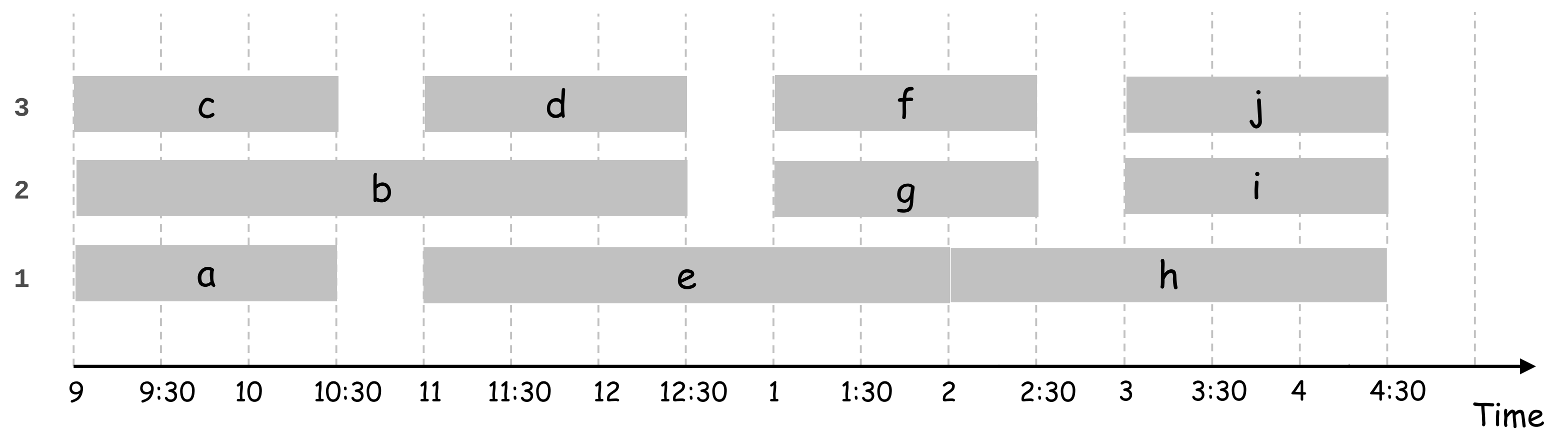


Interval Partitioning

Interval partitioning.

- Lecture j starts at s_j and finishes at f_j.
- Goal: find minimum number of classrooms to schedule all lectures so that no two occur at the same time in the same room.

Ex: This schedule uses only 3.



Interval Partitioning: Lower Bound on Optimal Solution

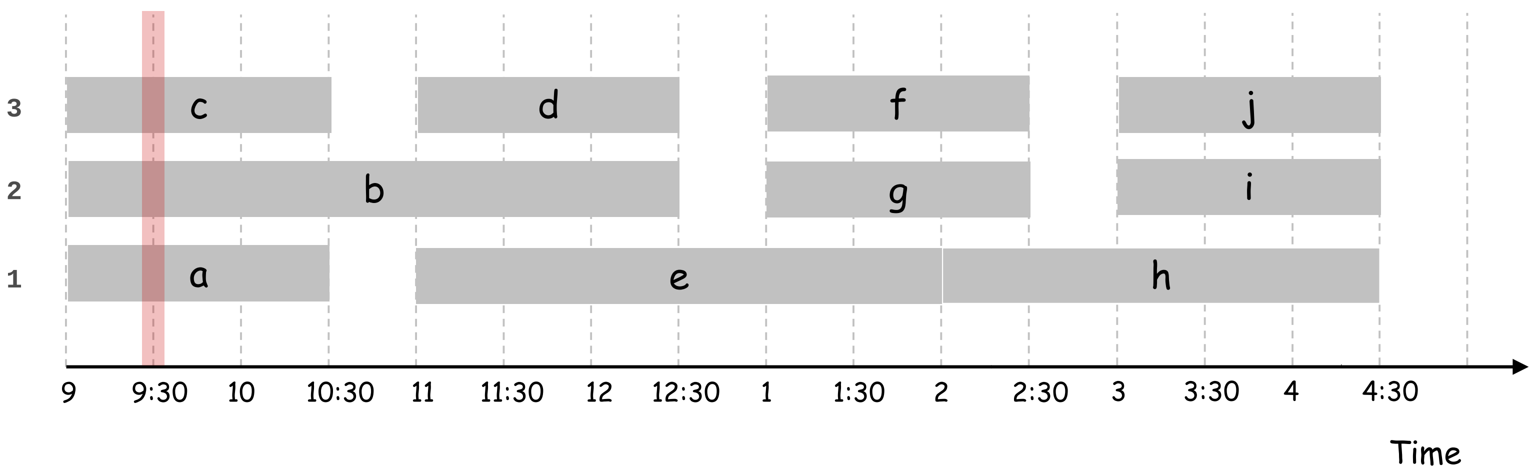
Def. The depth of a set of open intervals is the maximum number that contain any given time.

Key observation. Number of classrooms needed $\geq$ depth.

Ex: Depth of schedule below = 3 $\Rightarrow$ schedule below is optimal.

a, b, c all contain 9:30

Q. Does there always exist a schedule equal to depth of intervals?



Interval Partitioning: Greedy Algorithm

Greedy algorithm. Consider lectures in increasing order of start time: assign lecture to any compatible classroom.

```
Sort intervals by starting time so that s1 ≤ s2 ≤ ... ≤ sn.
d ← 0    ← number of allocated classrooms

for j = 1 to n {
   if (lecture j is compatible with some classroom k)
      schedule lecture j in classroom k
   else
      allocate a new classroom d + 1
      schedule lecture j in classroom d + 1
      d ← d + 1
}
```

Implementation. O(n log n).

- For each classroom k, maintain the finish time of the last job added.
- Keep the classrooms in a priority queue.
  - Quickly find the classroom with earliest finish time

Interval Partitioning: Greedy Analysis

Observation. Greedy algorithm never schedules two incompatible lectures in the same classroom.

Theorem. Greedy algorithm is optimal.

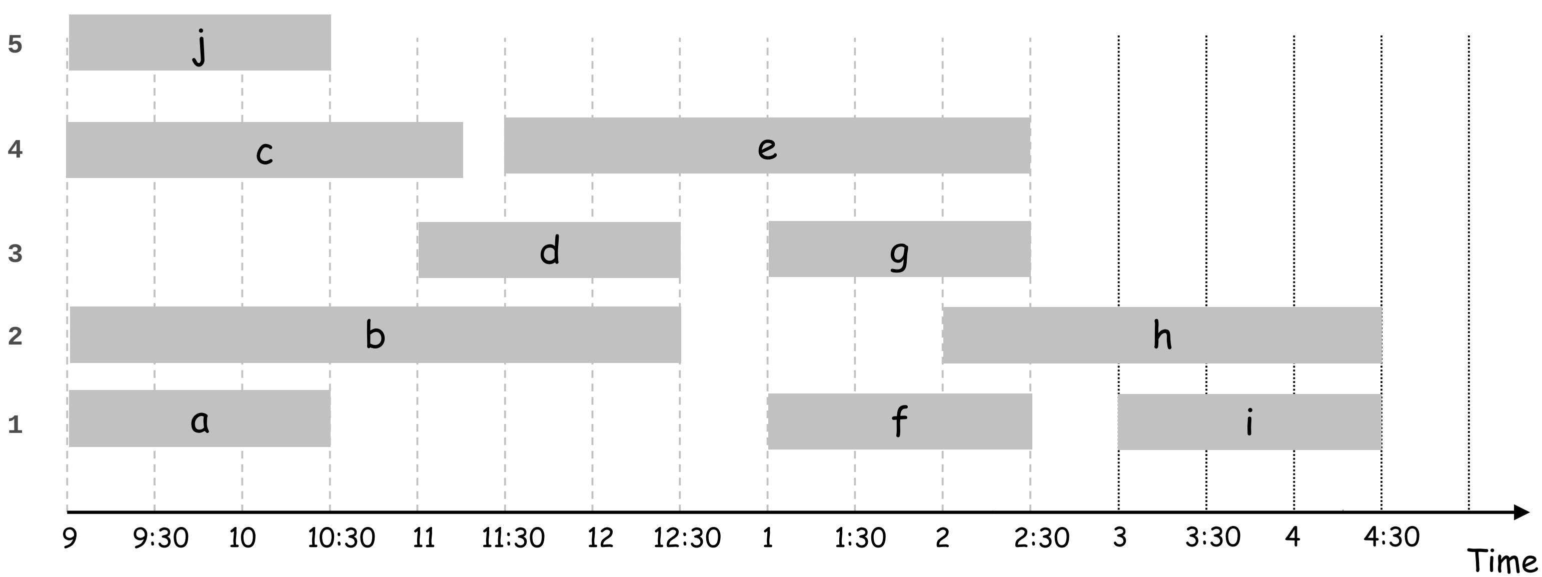
Pf.

- Let d = number of classrooms that the *greedy* algorithm allocates.
- Classroom d is opened because we needed to schedule a job, say j, that is incompatible with all d-1 other classrooms.
- These d jobs (including j) each end after s_j.
- Since we sorted by start time, all these incompatibilities are caused by lectures that start no later than s_j.
- Thus, we have d lectures overlapping at time $s_j + \varepsilon$.
- Key observation $\Rightarrow$ all schedules use $\geq$ d classrooms. ▪

Clicker Question

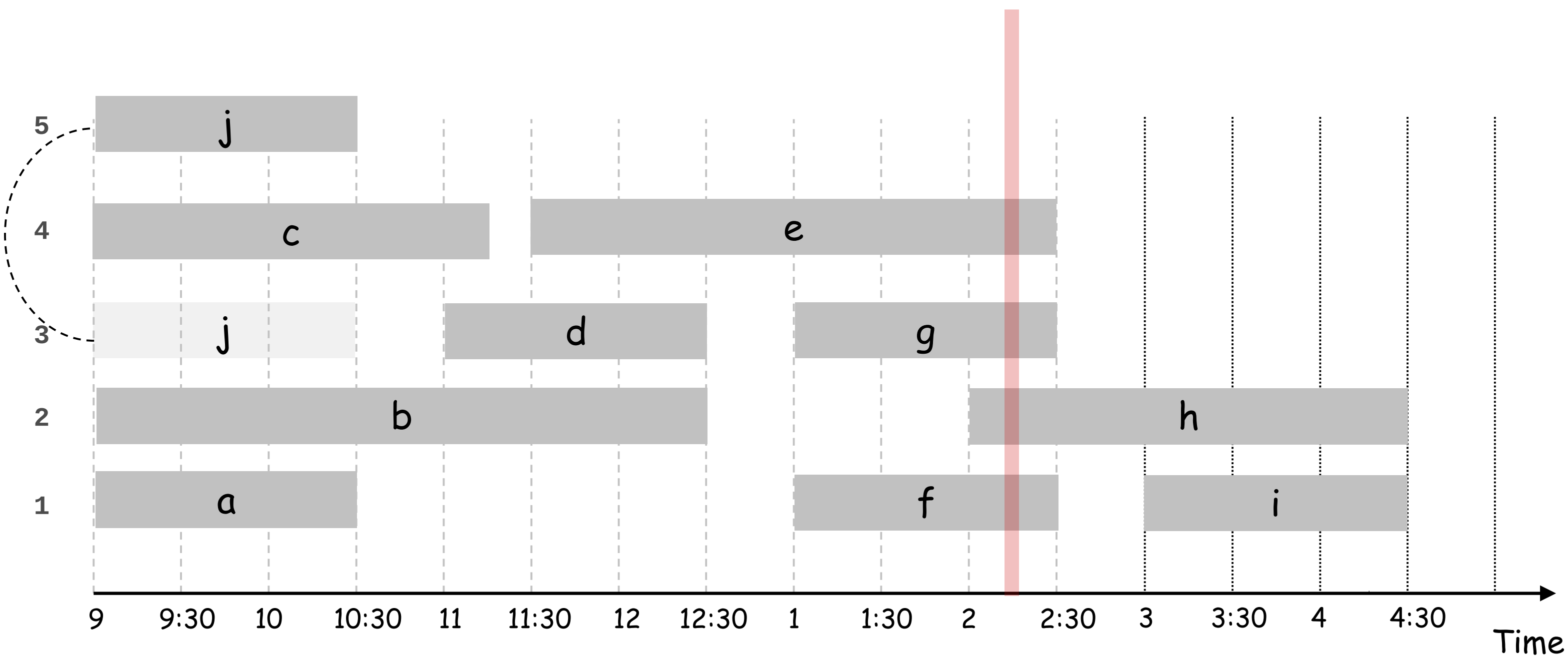
Consider the interval partitioning instance (below). The current schedule uses 5 classrooms. How many classrooms are required in the *optimal* solution?

A. 6 **B.** 5 **C.** 4 **D.** 3 **E.** ∞ suffices



Consider the interval partitioning instance (below). The current schedule uses 5 classrooms. How many classrooms are required in the *optimal* solution?

A. 6 **B.** 5 **C.** 4 **D.** 3 **E.** ∞ suffices



Greedy Analysis Strategies

Exchange argument. Gradually transform any solution to the one found by the greedy algorithm without hurting its quality.
Example: Interval Scheduling, Minimizing Lateness (inversions)

Structural. Discover a simple "structural" bound asserting that every possible solution must have a certain value. Then show that your algorithm always achieves this bound.
Example: Interval Partitioning (Depth of Schedule)

Greedy algorithm stays ahead. Show that after each step of the greedy algorithm, its solution is at least as good as any other algorithm's.
Example: Offline Caching, Dijkstra (shortest path for explored set)

Other greedy algorithms. Kruskal, Prim, Huffman, …