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.
Q. Is there a schedule that requires less classrooms?
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 example = 3 $\Rightarrow$ schedule is optimal, because 3 rooms.

\[
\text{a, b, c all contain 9:30}
\]

**Q.** Does there always exist a schedule equal to depth of intervals?
If so, how to obtain such a schedule? If not, why not?
(1 min for both)
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 $s_1 \leq s_2 \leq \ldots \leq s_n$. 

\[
\text{d } \leftarrow \text{ 0 \quad \text{number of allocated classrooms}}
\]

\[
\text{for} \quad j = 1 \text{ to } n \{
\begin{align*}
\text{if} \quad (\text{lecture } j \text{ is compatible with some classroom } k) \\
\quad \text{schedule lecture } j \text{ in classroom } k \\
\text{else} \\
\quad \text{allocate a new classroom } d + 1 \\
\quad \text{schedule lecture } j \text{ in classroom } d + 1 \\
\quad \text{d } \leftarrow \text{ d + 1}
\end{align*}
\}
\]

Q. What is the worst-case running time?
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 \( s_1 \leq s_2 \leq \ldots \leq s_n \).

d \leftarrow 0 \quad \text{number of allocated classrooms}

\begin{verbatim}
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 \leftarrow d + 1
}
\end{verbatim}

Q. What is the worst-case running time?
Q. How to implement finding a free classroom?
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 \( s_1 \leq s_2 \leq \ldots \leq s_n \).

\[
d \leftarrow 0 \quad \text{number of allocated classrooms}
\]

\[
\text{for } j = 1 \text{ to } n \{} \\
\quad \text{if (lecture } j \text{ is compatible with some classroom } k) \\
\quad \quad \text{schedule lecture } j \text{ in classroom } k \\
\quad \text{else} \\
\quad \quad \text{allocate a new classroom } d + 1 \\
\quad \quad \text{schedule lecture } j \text{ in classroom } d + 1 \\
\quad \quad d \leftarrow d + 1 \\
\}\]

Implementation. [Q. How to implement finding a free room?]

- For each classroom \( k \), maintain the finish time of the last job added.
- Keep the classrooms in a priority queue.
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 $s_1 \leq s_2 \leq \ldots \leq s_n$.
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 (operations of $O(\log n)$).
Interval Partitioning: Greedy Analysis

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

Observation 2. Greedy algorithm never schedules two incompatible lectures in the same classroom.

Theorem. Greedy algorithm for interval partitioning is optimal.

Pf. (using a structural bound)
Interval Partitioning: Greedy Analysis

Key observation. Number of classrooms needed ≥ depth.

Observation 2. Greedy algorithm never schedules two incompatible lectures in the same classroom.

Theorem. Greedy algorithm for interval partitioning is optimal.

Pf. (using a structural bound)

To prove: Greedy allocates d classrooms ⇒ we need at least d.
Interval Partitioning: Greedy Analysis

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

Observation 2. Greedy algorithm never schedules two incompatible lectures in the same classroom.

Theorem. Greedy algorithm for interval partitioning is optimal.

Pf. (using a structural bound)

To prove: Greedy allocates $d$ classrooms $\Rightarrow$ we need at least $d$.

Q. Why did classroom $d$ open?
Key observation. Number of classrooms needed ≥ depth.

Observation 2. Greedy algorithm never schedules two incompatible lectures in the same classroom.

Theorem. Greedy algorithm for interval partitioning is optimal.

Pf. (using a structural bound)

To prove: Greedy allocates d classrooms ⇒ we need at least d.

- Classroom d opened because lecture j incompatible with d-1 others.

Q. How many incompatibilities at time $s_j + \varepsilon$? (with $\varepsilon > 0$ very small)
Interval Partitioning: Greedy Analysis

Key observation. Number of classrooms needed \( \geq \) depth.

Observation 2. Greedy algorithm never schedules two incompatible lectures in the same classroom.

Theorem. Greedy algorithm for interval partitioning is optimal.

Pf. (using a structural bound)

To prove: Greedy allocates \( d \) classrooms \( \Rightarrow \) we need at least \( d \).

- Classroom \( d \) opened because lecture \( j \) incompatible with \( d-1 \) others.
- 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 \( s_j + \varepsilon \) (with \( \varepsilon \geq 0 \) very small).
Interval Partitioning: Greedy Analysis

Key observation. Number of classrooms needed \( \geq \) depth.

Observation 2. Greedy algorithm never schedules two incompatible lectures in the same classroom.

Theorem. Greedy algorithm for interval partitioning is optimal.

Pf. (using a structural bound)

To prove: Greedy allocates \( d \) classrooms \( \Rightarrow \) we need at least \( d \).

- Classroom \( d \) opened because lecture \( j \) incompatible with \( d-1 \) others.
- 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 \( s_j + \epsilon \) (with \( \epsilon \geq 0 \) very small).

- Using the key observation (ie number of classrooms needed \( \geq \) depth) \( \Rightarrow \) we know that all schedules use \( \geq d \) classrooms. 

This proof can be found on pages 123-125.