Lecture 9 – Greedy algorithms
Reading: KT Section 4.1

Greedy algorithms
- An algorithm is greedy if it builds up a solution in small steps,
  - choosing a decision at each step myopically to optimize some underlying criterion.
- One can often design many different greedy algorithms for the same problem,
  - each one locally, incrementally optimizing some different measure on its way to a solution.
- When a greedy algorithm succeeds in solving a nontrivial problem optimally,
  - it typically implies something interesting and useful about the structure of the problem itself

Interval scheduling
- Job \( j \) starts at \( s_j \) and finishes at \( f_j \).
- Two jobs are compatible if they don’t overlap.
- Goal: find maximum subset of mutually compatible jobs.
Interval scheduling: greedy algorithms

- How should we pick the next job to schedule?
- Earliest start time
- Earliest finish time
- Shortest interval
- Fewest conflicts

Interval Partitioning

**Earliest-Finish-Time-First** \( (n, s_1, s_2, \ldots, s_n, f_1, f_2, \ldots, f_n) \)

**SORT** jobs by finish time so that \( f_1 \leq f_2 \leq \ldots \leq f_n \)

\( A \leftarrow \emptyset \) \quad \text{set of jobs selected}

\( \text{FOR } j = 1 \text{ TO } n \)

\( \text{If job } j \text{ is compatible with } A \)

\( A \leftarrow A \cup \{ j \} \)

**RETURN** \( A \)

Let’s analyze!

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 lectures occur at the same time in the same room.

Ex. This schedule uses 4 classrooms to schedule 10 lectures.
Greedy algorithm

Greedy template. Consider lectures in some natural order. Assign each lecture to an available classroom (which one?), allocate a new classroom if none are available.

- counterexample for earliest finish time
  3
  2
  1

- counterexample for shortest interval
  2
  1

- counterexample for fewer conflicts
  2
  1

Lower bound on optimal solution

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

Key observation. Number of classrooms needed ≥ depth.

Q. Does minimum number of classrooms needed always equal depth?
A. Yes! Moreover, earliest-start-time-first algorithm finds a schedule whose number of classrooms equals the depth.

**EARLIEST-START-TIME-FIRST (n, s_1, s_2, ..., s_n, f_1, f_2, ..., f_n)**

*SORT* lectures by start time so that \( s_1 \leq s_2 \leq \ldots \leq s_n \).

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

FOR \( j = 1 \) TO \( n \)

If lecture \( j \) is compatible with some classroom
   Schedule lecture \( j \) in any such classroom \( k \).

ELSE
   Allocate a new classroom \( d + 1 \).
   Schedule lecture \( j \) in classroom \( d + 1 \).
   \( d \leftarrow d + 1 \)

RETURN schedule.