Greedy Algorithms
### Problem

Interval scheduling: Given a set of \( n \) intervals of the form \((S(i), F(i))\), find the largest subset of non-overlapping intervals.

### Algorithm

**GreedySchedule**
- Initialize \( R \) to contain all intervals
- While \( R \) is not empty
  - Choose an interval \((S(i), F(i))\) from \( R \) that has the smallest value of \( F(i) \)
  - Delete all intervals in \( R \) that overlaps with \((S(i), F(i))\)

- Running time?
Greedy Algorithms

Interval scheduling

Problem

Interval scheduling: Given a set of $n$ intervals of the form $(S(i), F(i))$, find the largest subset of non-overlapping intervals.

Algorithm

GreedySchedule
- While $R$ is not empty
- Choose an interval $(S(i), F(i))$ from $R$ that has the smallest value of $F(i)$
- Delete all intervals in $R$ that overlaps with $(S(i), F(i))$

- Running time? $O(n \log n)$
Problem

Job scheduling: You are given $n$ jobs and you are supposed to schedule these jobs on a machine. Each job $i$ consists of a duration $T(i)$ and a deadline $D(i)$. The lateness of a job w.r.t. a schedule is defined as $\max(0, F(i) - D(i))$, where $F(i)$ is the finishing time of job $i$ as per the schedule. The goal is to minimise the maximum lateness.
Greedy Algorithms

Job scheduling

Problem

**Job scheduling:** You are given $n$ jobs and you are supposed to schedule these jobs on a machine. Each job $i$ consists of a duration $T(i)$ and a deadline $D(i)$. The *lateness* of a job w.r.t. a schedule is defined as $\max(0, F(i) - D(i))$, where $F(i)$ is the finishing time of job $i$ as per the schedule. The goal is to minimise the maximum lateness.

- Greedy strategies
  - Smallest jobs first.
Problem

Job scheduling: You are given \( n \) jobs and you are supposed to schedule these jobs on a machine. Each job \( i \) consists of a duration \( T(i) \) and a deadline \( D(i) \). The lateness of a job w.r.t. a schedule is defined as \( \max(0, F(i) - D(i)) \), where \( F(i) \) is the finishing time of job \( i \) as per the schedule. The goal is to minimise the maximum lateness.

- Greedy strategies
  - Smallest jobs first.
  - Earliest deadline first.

Algorithm

GreedyJobSchedule

- Sort the jobs in non-decreasing order of deadlines and schedule the jobs on the machine in this order.
**Algorithm**

GreedyJobSchedule

- Sort the jobs in non-decreasing order of deadlines and schedule the jobs on the machine in this order.

**Claim 1**: There is an optimal schedule with no idle time (time when the machine is idle).

**Definition**

A schedule is said to have inversion if there are a pair of jobs \((i, j)\) such that

1. \(D(i) < D(j)\), and
2. Job \(j\) is performed before job \(i\) as per the schedule.

**Claim 2**: There is an optimal schedule with no idle time and no inversion.
Claim 2: There is an optimal schedule with no idle time and no inversion.

Proof sketch of Claim 2

- Consider an optimal schedule $O$. First, if there is any idle time, we obtain another optimal schedule $O_1$ without the idle time.
- Suppose $O_1$ has inversions. Consider one such inversion $(i, j)$.

Claim 2.1: If an inversion exists, then there exists a pair of adjacently scheduled jobs $(m, n)$ such that the schedule has an inversion w.r.t. $(m, n)$.
Claim 2: There is an optimal schedule with no idle time and no inversion.

Proof sketch of Claim 2

- Consider an optimal schedule $O$. First, if there is any idle time, we obtain another optimal schedule $O_1$ without the idle time.
- Suppose $O_1$ has inversions. Consider one such inversion $(i, j)$.
- Claim 2.1: If an inversion exists, then there exists a pair of adjacently scheduled jobs $(m, n)$ such that the schedule has an inversion w.r.t. $(m, n)$.

Claim 2.2: If a schedule has an inversion w.r.t. adjacently scheduled jobs $(m, n)$, then exchanging $m$ and $n$ does not increase the maximum lateness.
End