Lecture 18:
• Greedy Algorithms: Proof Techniques

March 30, 2020
The movie *Wall Street*, however, is not.
Greedy Algorithms

• What’s a greedy algorithm?
  • I know it when I see it
  • Roughly, an algorithm that builds a solution myopically and never looks back (compare to DP)
  • Typically, make a single pass over the input (e.g. Kruskal)

• Why care about greedy algorithms?
  • Greedy algorithms are the fastest and simplest algorithms imaginable, and sometimes they work!
  • Sometimes make useful heuristics when they don’t
  • Simplicity makes them easy to adapt to different models
Interval Scheduling
(Weighted) Interval Scheduling

- **Input:** $n$ intervals $(s_i, f_i)$ with values $v_i$
- **Output:** a compatible schedule $S$ with the largest possible total value
  - A schedule is a subset of intervals $S \subseteq \{1, \ldots, n\}$
  - A schedule $S$ is compatible if no two $i, j \in S$ overlap
  - The total value of $S$ is $\sum_{i \in S} v_i$
(Unweighted) Interval Scheduling

- **Input:** \( n \) intervals \((s_i, f_i)\)
- **Output:** a compatible schedule \( S \) with the largest possible size
  - A schedule is a subset of intervals \( S \subseteq \{1, ..., n\} \)
  - A schedule \( S \) is compatible if no two \( i, j \in S \) overlap
Possibly Greedy Rules

• Choose the shortest interval first

• Choose the interval with earliest start first

• Choose the interval with earliest finish first
Greedy Algorithm: Earliest Finish First

- Sort intervals so that $f_1 \leq f_2 \leq \cdots \leq f_n$
- Let $S$ be empty
- For $i = 1, \ldots, n$:
  - If interval $i$ doesn’t create a conflict, add $i$ to $S$
- Return $S$
Greedy Stays Ahead

• How do we know we found an optimal schedule

• “Greedy Stays Ahead” strategy
  • We’ll show that at every point in time, the greedy schedule does better than any other schedule
Greedy Stays Ahead

• Let $G = \{i_1, ..., i_r\}$ be greedy’s schedule
• Let $O = \{j_1, ..., j_s\}$ be some optimal schedule
• **Key Claim:** for every $t = 1, ..., r$, $f_{i_t} \leq f_{j_t}$
Greedy Stays Ahead

• Let $G = \{i_1, \ldots, i_r\}$ be greedy’s schedule
• Let $O = \{j_1, \ldots, j_s\}$ be some optimal schedule
• **Key Claim:** for every $t = 1, \ldots, r$, $f_{i_t} \leq f_{j_t}$
Greedy Stays Ahead

• Let $G = \{i_1, \ldots, i_r\}$ be greedy’s schedule
• Let $O = \{j_1, \ldots, j_s\}$ be some optimal schedule
• Key Claim: for every $t = 1, \ldots, r$, $f_{i_t} \leq f_{j_t}$
Greedy Stays Ahead

• Let $G = \{i_1, \ldots, i_r\}$ be greedy’s schedule
• Let $O = \{j_1, \ldots, j_s\}$ be some optimal schedule
• **Key Claim:** for every $t = 1, \ldots, r$, $f_{i_t} \leq f_{j_t}$
Minimum Lateness Scheduling
Minimum Lateness Scheduling

- **Input:** \( n \) jobs with length \( t_i \) and deadline \( d_i \)
  - Simplifying assumption: all deadlines are distinct
- **Output:** a minimum-lateness schedule for the jobs
  - Can only do one job at a time, no overlap
  - The lateness of job \( i \) is \( \max\{f_i - d_i, 0\} \)
  - The lateness of a schedule is \( \max_i\{\max\{f_i - d_i, 0\}\} \)
Possible Greedy Rules

• Choose the shortest job first \((\min t_i)\)?

• Choose the most urgent job first \((\min d_i - t_i)\)?
Greedy Algorithm: Earliest Deadline First

• Sort jobs so that $d_1 \leq d_2 \leq \cdots \leq d_n$
• For $i = 1, \ldots, n$:
  • Schedule job $i$ right after job $i - 1$ finishes
Exchange Argument

• $G = \text{greedy schedule}, \ O = \text{optimal schedule}$

• Exchange Argument:
  • We can transform $O$ to $G$ by exchanging pairs of jobs
  • Each exchange only reduces the lateness of $O$
  • Therefore the lateness of $G$ is at most that of $O$
Exchange Argument

• $G = \text{greedy schedule, } O = \text{optimal schedule}$

• Observation: the optimal schedule has no gaps
  • A schedule is just an ordering of the jobs, with jobs scheduled back-to-back
Exchange Argument

- $G =$ greedy schedule, $O =$ optimal schedule

- We say that two jobs $i, j$ are inverted in $O$ if $d_i < d_j$ but $j$ comes before $i$
  - Observation: greedy has no inversions
Exchange Argument

• We say that two jobs \( i, j \) are inverted in \( O \) if \( d_i < d_j \) but \( j \) comes before \( i \)

• Claim: the optimal schedule has no inversions
  • Step 1: suppose \( O \) has an inversion, then it has an inversion \( i, j \) where \( i, j \) are consecutive
Exchange Argument

• We say that two jobs $i, j$ are inverted in $O$ if $d_i < d_j$ but $j$ comes before $i$

• Claim: the optimal schedule has no inversions
  • Step 1: suppose $O$ has an inversion, then it has an inversion $i, j$ where $i, j$ are consecutive
  • Step 2: if $i, j$ are a consecutive jobs that are inverted then flipping them only reduces the lateness
Exchange Argument

• If $i, j$ are a consecutive jobs that are inverted then flipping them only reduces the lateness
Exchange Argument

- We say that two jobs $i, j$ are inverted in $O$ if $d_i < d_j$ but $j$ comes before $i$

- Claim: the optimal schedule has no inversions
  - Step 1: suppose $O$ has an inversion, then it has an inversion $i, j$ where $i, j$ are consecutive
  - Step 2: if $i, j$ are a consecutive jobs that are inverted then flipping them only reduces the lateness

- $G$ is the unique schedule with no inversions, $O$ is the unique schedule with no inversions, $G = O$