Lecture 7

• Greedy Algorithms cont’d
Review

• In a DFS tree of an undirected graph, can there be an edge \((u,v)\)
  – where \(v\) is an ancestor of \(u\)? (“back edge”)
  – where \(v\) is a sibling of \(u\)? (“cross edge”)

• Same questions with a directed graph?

• Same questions with a BFS tree
  – directed?
  – undirected?
Interval Scheduling Problem

- Job $j$ starts at $s_j$ and finishes at $f_j$.
- Two jobs are **compatible** if they do not overlap.
- **Find**: maximum subset of mutually compatible jobs.

![Interval Scheduling Diagram](image-url)
Greedy: Counterexamples

- for earliest start time
- for shortest interval
- for fewest conflicts
Formulating Algorithm

- Arrays of start and finishing times
  - \( s_1, s_2, \ldots, s_n \)
  - \( f_1, f_2, \ldots, f_n \)

- Input length?
  - \( 2n = \Theta(n) \)
Greedy Algorithm

• **Earliest finish time:** ascending order of \( f_j \).

```plaintext
Sort jobs by finish times so that \( f_1 \leq f_2 \leq \ldots \leq f_n \).

\[
A \leftarrow \emptyset \quad \Delta \text{ Set of selected jobs}
\]

\[
\text{for } j = 1 \text{ to } n \{ \\
\quad \text{if (job } j \text{ compatible with } A) \\
\quad \quad A \leftarrow A \cup \{j\}
\}
\]

\[
\text{return } A
\]
```

• **Implementation.** \( O(n \log n) \) time; \( O(n) \) space.
  
  – Remember job \( j^* \) that was added last to \( A \).
  
  – Job \( j \) is compatible with \( A \) if \( s_j \geq f_{j^*} \).
Running time: \( O(n \log n) \)

\begin{align*}
O(n \log n) & \\
O(1) & \\
n \times O(1) & \\
\end{align*}

Sort jobs by finish times so that 
\[ f_1 \leq f_2 \leq \ldots \leq f_n. \]

\begin{align*}
A & \leftarrow \text{(empty)} \quad \triangle \text{Queue of selected jobs} \\
j* & \leftarrow 0 \\
\text{for } j = 1 \text{ to } n \{ \\
\quad \text{if } (f_{j*} \leq s_j) \\
\quad \quad \text{enqueue}(j \text{ onto } A) \\
\} \\
\text{return } A
\end{align*}
Analysis: Greedy Stays Ahead

• Theorem. Greedy algorithm is optimal.

• Proof strategy (by contradiction):
  – Suppose greedy is not optimal.
  – Consider an optimal strategy… which one?
    • Consider the optimal strategy that agrees with the greedy strategy for as many initial jobs as possible
  – Look at first place in list where optimal strategy differs from greedy strategy
  – Show a new optimal strategy that agrees more w/ greedy

A. Smith; based on slides by E. Demaine, C. Leiserson, S. Raskhodnikova, K. Wayne
• Theorem. Greedy algorithm is optimal.

• Pf (by contradiction): Suppose greedy is not optimal.
  – Let $i_1, i_2, \ldots i_k$ denote set of jobs selected by greedy.
  – Let $j_1, j_2, \ldots j_m$ be set of jobs in the optimal solution with $i_1 = j_1, i_2 = j_2, \ldots, i_r = j_r$ for the largest possible value of $r$.

```
Greedy:       OPT:                                 job $i_{r+1}$ finishes before $j_{r+1}$
i_1           j_1                                  ↓
i_2           j_2                                  ↓
i_r           j_r                                  ↓
i_{r+1}       j_{r+1}                                  ↓
```

why not replace job $j_{r+1}$ with job $i_{r+1}$?
Analysis: Greedy Stays Ahead

• Theorem. Greedy algorithm is optimal.

• Pf (by contradiction): Suppose greedy is not optimal.
  – Let $i_1, i_2, \ldots, i_k$ denote set of jobs selected by greedy.
  – Let $j_1, j_2, \ldots, j_m$ be set of jobs in the optimal solution with
    $i_1 = j_1, i_2 = j_2, \ldots, i_r = j_r$ for the largest possible value of $r$.

Greedy: $i_1 \quad i_2 \quad i_r \quad i_{r+1}$

OPT: $j_1 \quad j_2 \quad j_r \quad i_{r+1}$

job $i_{r+1}$ finishes before $j_{r+1}$

↑ solution still feasible and optimal, but contradicts maximality of $r$. 

A. Smith; based on slides by E. Demaine, C. Leiserson, S. Raskhodnikova, K. Wayne
Interval Partitioning Problem

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

• **E.g.**: 10 lectures are scheduled in 4 classrooms.
Interval Partitioning

• Lecture \( j \) starts at \( s_j \) and finishes at \( f_j \).

• **Find**: minimum number of classrooms to schedule all lectures so that no two occur at the same time in the same room.

• **E.g.**: Same lectures are scheduled in 3 classrooms.
Lower Bound

• **Definition.** 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.

• **E.g.:** Depth of this schedule = 3 \( \Rightarrow \) this schedule is optimal.

\[
\begin{array}{cccccccccc}
9 & 9:30 & 10 & 10:30 & 11 & 11:30 & 12 & 12:30 & 1 & 1:30 & 2 \\
3 & c &  &  &  &  &  &  &  &  &  \\
2 &  & b &  &  &  &  &  &  &  &  \\
1 & a &  &  & e &  &  & h &  &  &  \\
\end{array}
\]

- a, b, c all contain 9:30

• **Q:** Is it always sufficient to have number of classrooms = depth?

9/10/2008

*A. Smith; based on slides by E. Demaine, C. Leiserson, S. Raskhodnikova, K. Wayne*
Greedy Algorithm

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

Sort intervals by starting time so that \( s_1 \leq s_2 \leq \ldots \leq s_n \).

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

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

• Implementation. \( O(n \log n) \) time; \( O(n) \) space.
  – For each classroom, maintain the finish time of the last job added.
  – Keep the classrooms in a priority queue (main loop \( n \log(d) \) time)
• **Observation.** Greedy algorithm never schedules two incompatible lectures in the same classroom.

• **Theorem.** Greedy algorithm is optimal.

• **Proof:** Let $d =$ number of classrooms allocated by greedy.
  – Classroom $d$ is opened because we needed to schedule a lecture, say $j$, that is incompatible with all $d-1$ last lectures in other classrooms.
  – These $d$ lectures each end after $s_j$.
  – Since we sorted by start time, they 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. ▪
Scheduling to minimize lateness

A. Smith; based on slides by E. Demaine, C. Leiserson, S. Raskhodnikova, K. Wayne
Scheduling to Minimizing Lateness

Minimizing lateness problem.

- Single resource processes one job at a time.
- Job $j$ requires $t_j$ units of processing time and is due at time $d_j$.
- If $j$ starts at time $s_j$, it finishes at time $f_j = s_j + t_j$.
- Lateness: $\ell_j = \max \{0, f_j - d_j\}$.
- Goal: schedule all jobs to minimize maximum lateness $L = \max \ell_j$.

Ex:

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_j$</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>4</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>$d_j$</td>
<td>6</td>
<td>8</td>
<td>9</td>
<td>9</td>
<td>14</td>
<td>15</td>
</tr>
</tbody>
</table>

lateness = 2  lateness = 0  max lateness = 6

<table>
<thead>
<tr>
<th>$d_3 = 9$</th>
<th>$d_2 = 8$</th>
<th>$d_6 = 15$</th>
<th>$d_1 = 6$</th>
<th>$d_5 = 14$</th>
<th>$d_4 = 9$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>
Minimizing Lateness: Greedy Algorithms

Greedy template. Consider jobs in some order.

- [Shortest processing time first] Consider jobs in ascending order of processing time $t_j$.

- [Earliest deadline first] Consider jobs in ascending order of deadline $d_j$.

- [Smallest slack] Consider jobs in ascending order of slack $d_j - t_j$. 
Minimizing Lateness: Greedy Algorithms

Greedy template. Consider jobs in some order.

- [Shortest processing time first] Consider jobs in ascending order of processing time $t_j$.

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_j$</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>$d_j$</td>
<td>100</td>
<td>10</td>
</tr>
</tbody>
</table>

  counterexample

- [Smallest slack] Consider jobs in ascending order of slack $d_j - t_j$.

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_j$</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>$d_j$</td>
<td>2</td>
<td>10</td>
</tr>
</tbody>
</table>

  counterexample