**CS Lunch**

Michelle Oraa Ali  
Tien Dao  
Vladislava Paskova  
Nan Zhuang  
Surabhi Sharma

Wednesday, 12:15 PM  
Kendade 307

---

**Midterm 2!**

- Monday, November 21  
- In class  
- Covers Greedy Algorithms  
- Closed book

---

**Dynamic Programming “Recipe”**

- Solve subproblems, remember the solutions in an array  
- Gradually build up solution to bigger problems based on subproblem solutions
Interval Scheduling
(Yes, this is an old problem!)

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

Interval Scheduling: Greedy Solution

- Sort jobs by earliest finish time.
- Take each job provided it's compatible with the ones already taken.

b, e, h
Weighted Interval Scheduling

- Job \( j \) starts at \( s_j \), finishes at \( f_j \), and has weight or value \( v_j \).
- Two jobs compatible if they don't overlap.
- Goal: find maximum weight subset of mutually compatible jobs.

<table>
<thead>
<tr>
<th>Time</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Optimal Substructure

- \( \text{OPT}(j) \) = value of optimal solution to the problem consisting of job requests 1, 2, ..., \( j \).
- Case 1: \( \text{OPT} \) selects job \( j \).
- Case 2: \( \text{OPT} \) does not select job \( j \).

\[
\text{OPT}(j) = \begin{cases} 
0 & \text{if } j = 0 \\
\max \left\{ v_j \times \text{OPT}(p_j), \text{OPT}(j-1) \right\} & \text{otherwise} 
\end{cases}
\]