Goals

• Practice Dynamic Programming Set-up
• Describe multi-exchange arguments

Test Info

• All topics up to 3/5 except QuickSort
• All PSets up to 4
• Tuesday: review. Topics → Canvas Discussion until Sunday
• Wed @ noon → Fri @ 6 pm
• Reserve 3 hours
• No class 3/14
• 1 page, single sided hand written notes

Quiz on Canvas

No PSet 5
Scheduling Goal:

Minimize $\sum_{j=1}^{n} w_j c_j \Leftarrow \text{"Objective function"}$

Greedy strategy: Order by largest $\frac{w_i}{t_i}$

Q: If we get rid of assumption that all $\frac{w_i}{t_i}$ are unique, what changes? Which change(s) make the proof fail?
Scheduling Goal:

Minimize \( \sum_{j=1}^{n} w_j c_j \) \( \Rightarrow \) "Objective function"

Greedy strategy: Order by largest \( w_i / t_i \)

Q: If we get rid of assumption that all \( w_i / t_i \) are unique, why does our exchange argument fail?

A) There might not be a unique solution
B) Objective function might not decrease from \( \sigma^+ \) to \( \sigma^* \)
C) We can't create an ordering such that \( w_1 / t_1 > w_2 / t_2 > \cdots > w_n / t_n \)

All are true, but B ruins the proof.

Recall, we get a contradiction b/c \( \sigma^k = (\ldots, k, j, \ldots) \)

where \( j < k \Rightarrow \frac{w_j}{t_j} > \frac{w_k}{t_k} \Rightarrow w_j t_k > w_k t_j \)

\( \Rightarrow A_{\sigma^k} - A_{\sigma^*} = w_j t_k - w_k t_j > 0 \)
Now, if $w_i/t_i$ not unique:

$$j \rightarrow k \Rightarrow \frac{w_j}{t_j} > \frac{w_k}{t_k} \Rightarrow w_j t_k > w_k t_j$$

$$\Rightarrow A_{\sigma^*} - A_{\sigma^*'} = w_j t_k - w_k t_j > 0$$

New Idea:
Still use **EXCHANGE** argument, but now need more exchanges.

Choose some ordering $s.t.$

$$w_1/t_1 \geq w_2/t_2 \geq w_3/t_3 \cdots \geq w_n/t_n$$

Let $\sigma$ be strategy using this ordering.

Let $\sigma'$ be any other strategy.

We will show $A_{\sigma'} \geq A_{\sigma}$, which means $\sigma$ is optimal.

This procedure is bubble sort!
Conclusion: $A_\sigma \geq A_\sigma$, so greedy strategy is optimal.

What is the runtime of this greedy algorithm?

A) $O(1)$  B) $O(n)$  C) $O(n \log n)$  D) $O(n^2)$
Conclusion: $A_{\sigma} \geq A_{\tau}$, so greedy strategy is optimal.

What is the runtime of this greedy algorithm?

A) $O(1)$  B) $O(n)$  C) $O(n \log n)$  D) $O(n^2)$

$O(n \log n)$. Need to sort $n$ items, requires $O(n \log n)$ time, bubble sort is only imagined to happen as part of proof. It doesn't actually occur.
To Create D.P. (dynamic programming) algorithm:

1. Think of form of optimal solution.
   (ii)
   • WMIS on line: \( v_n \in S \) or \( v_n \notin S_n \)

2. Optimal soln in terms of smaller soln? (For each case)

   (i)
   
   (ii)

3. Create recurrence for objective function

   \[
   A(k) = \max \left\{ A(k-1), A(k-2) + w_k \right\}
   \]

   (i) 
   (ii)

4. Store \( A \) in an array using for-loop

5. Work backwards through array to reconstruct optimal solution
Knapsack Problem $K(v_1, ..., v_n, w_1, ..., w_n, W)$

Input: $n$ items, each has
- value $v_i \geq 0$
- size $w_i \geq 0$

Capacity $W$

Output: A subset $S \subseteq \{1, 2, ..., n\}$ that maximizes $\sum_{i \in S} v_i = V(S)$ and satisfies $\sum_{i \in S} w_i \leq W$

Applications
- Cargo trucks
- Investments $K_i \left[ \begin{array}{c} v_i = \text{expected yield} \% \\ w_i = \text{min investment amount} \\ W = \text{amount to invest} \end{array} \right.$

Notation
- Let $K_{ij,r}$ be subproblem on 1st $i$ items, with capacity $r$.
- Solution: satisfies constraints
- Optimal solution: satisfies constraints and maximizes value.
1. Form of optimal solution:

Let $S$ be the optimal solution to $K_{n,w}$

Two options:

(i) $n \notin S$

(ii) $n \in S$

2. Relate to optimal solution of subproblem

(i) If $S$ is optimal solution to $K_{n,w}$ and $n \notin S$, then __ is optimal solution to $K_{-,-}$

Pf: For contradiction assume __ is not optimal solution to $K_{-,-}$ ....

(ii) If $S$ is optimal solution to $K_{n,w}$ and $n \in S$, then __ is optimal solution to $K_{-,-}$.

Pf: For contradiction assume __ is not optimal solution to $K_{-,-}$ ....
(i) If $S$ is the optimal soln to $K_n,w$ and $n \notin S$, then $S$ is optimal soln to $K_{n-1},w$.

Pf: For contradiction, suppose $S$ is not opt soln for $K_{n-1},w$. Then there exists an optimal solution $S'$ for $K_{n-1},w$ with $V(S') > V(S)$. But $S'$ is also a solution to $K_n,w$ with $V(S') > V(S)$ and $n \notin S'$, so $S$ is not the optimal solution, a contradiction.

(ii) If $S$ is opt. soln for $K_n,w$ and $n \in S$, then $S \setminus \{n\}$ is opt soln for $K_{n-1},w-w_n$.

Pf: Suppose for contradiction, $S \setminus \{n\}$ is not optimal for $K_{n-1},w-w_n$. Then there exists an optimal solution $S'$ for $K_{n-1},w-w_n$ s.t. $V(S') > V(S \setminus \{n\})$. But then $S' \cup \{n\}$ is a soln to $K_n,w$, and $V(S' \cup \{n\}) > V(S)$, so $S$ is not opt for $K_n,w$, a contradiction.