19.5

Maximum Weight Subset of Elements: Cardinality and Beyond
Picking $k$ elements to maximize total weight

1. Given $n$ items each with non-negative weights/profits and integer $1 \leq k \leq n$.
2. Goal: pick $k$ elements to maximize total weight of items picked.

<table>
<thead>
<tr>
<th></th>
<th>$e_1$</th>
<th>$e_2$</th>
<th>$e_3$</th>
<th>$e_4$</th>
<th>$e_5$</th>
<th>$e_6$</th>
</tr>
</thead>
<tbody>
<tr>
<td>weight</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>4</td>
<td>3</td>
<td>2</td>
</tr>
</tbody>
</table>

$k = 2$: 

$k = 3$: 

$k = 4$: 
**Greedy Template**

\[
N \text{ is the set of all elements } X \leftarrow \emptyset \\
(* \ X \text{ will store all the elements that will be picked } *)
\]

while \( |X| < k \) and \( N \) is not empty do

\[ \text{choose } e_j \in N \text{ of maximum weight} \]

\[ \text{add } e_j \text{ to } X \]

\[ \text{remove } e_j \text{ from } N \]

return the set \( X \)

**Remark:** One can rephrase algorithm simply as sorting elements in decreasing weight order and picking the top \( k \) elements but the above template generalizes to other settings a bit more easily.

**Theorem 19.1.**

Greedy is optimal for picking \( k \) elements of maximum weight.
Greedy Template

\[ N \text{ is the set of all elements } X \leftarrow \emptyset \]
(* \( X \) will store all the elements that will be picked *)

while \( |X| < k \) and \( N \) is not empty do
  choose \( e_j \in N \) of maximum weight
  add \( e_j \) to \( X \)
  remove \( e_j \) from \( N \)

return the set \( X \)

**Remark:** One can rephrase algorithm simply as sorting elements in decreasing weight order and picking the top \( k \) elements but the above template generalizes to other settings a bit more easily.

**Theorem 19.1.**

Greedy is optimal for picking \( k \) elements of maximum weight.
A more interesting problem

1. Given $n$ items $N = \{e_1, e_2, \ldots, e_n\}$. Each item $e_i$ has a non-negative weight $w_i$.
2. Items partitioned into $h$ sets $N_1, N_2, \ldots, N_h$. Think of each item having one of $h$ colors.
3. Given integers $k_1, k_2, \ldots, k_h$ and another integer $k$
4. Goal: pick $k$ elements such that no more than $k_i$ from $N_i$ to maximize total weight of items picked.

<table>
<thead>
<tr>
<th></th>
<th>$e_1$</th>
<th>$e_2$</th>
<th>$e_3$</th>
<th>$e_4$</th>
<th>$e_5$</th>
<th>$e_6$</th>
<th>$e_7$</th>
</tr>
</thead>
<tbody>
<tr>
<td>weight</td>
<td>9</td>
<td>5</td>
<td>4</td>
<td>7</td>
<td>5</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

$N_1 = \{e_1, e_2, e_3\}$, $N_2 = \{e_4, e_5\}$, $N_3 = \{e_6, e_7\}$

$k = 4$, $k_1 = 2$, $k_2 = 1$, $k_3 = 2$
Greedy Template

\[ N \text{ is the set of all elements } X \leftarrow \emptyset \]
(* \( X \) will store all the elements that will be picked *)

\[ \text{while } N \text{ is not empty do} \]
\[ N' = \{ e_i \in N \mid X \cup \{ e_i \} \text{ is feasible} \} \]
\[ \text{if } N' = \emptyset \text{ then break} \]
\[ \text{choose } e_j \in N' \text{ of maximum weight} \]
\[ \text{add } e_j \text{ to } X \]
\[ \text{remove } e_j \text{ from } N \]

\[ \text{return the set } X \]

**Theorem 19.2.**

Greedy is optimal for the problem on previous slide.

Proof: exercise after class.

Special case of general phenomenon of Greedy working for maximum weight independent set in a matroid. Beyond scope of course.
Greedy Template

\[ N \text{ is the set of all elements } X \leftarrow \emptyset \]
(* X will store all the elements that will be picked *)

while \( N \) is not empty do

\[ N' = \{ e_i \in N \mid X \cup \{ e_i \} \text{ is feasible} \} \]

if \( N' = \emptyset \) then break

choose \( e_j \in N' \) of maximum weight

add \( e_j \) to \( X \)

remove \( e_j \) from \( N \)

return the set \( X \)

Theorem 19.2.

Greedy is optimal for the problem on previous slide.

Proof: exercise after class.

Special case of general phenomenon of Greedy working for maximum weight independent set in a matroid. Beyond scope of course.
THE END

... (for now)