**Goal:** choose as many disjoint intervals as possible, (i.e., without any overlap)

**Algorithm:**
1. Sort the intervals in increasing order of finishing times. At any stage, choose the earliest finishing interval that is disjoint from all previously chosen intervals (i.e., the local evaluation criterion is $f_i$).
2. "Greedy stays ahead" argument
   - Intuition: out of the a given set of intervals, greedy picks as many as optimal

**VISUAL EXAMPLE**

Input: $A[1..n]$  
$G$: $A_1$, $A_2$, $A_3$, $A_4$, $A_5$  
$O$: $A_1$, $A_2$, $A_3$, $A_4$, $A_5$

How to compare $G$ and $O$? Imagine reordering $O$ to match $G$!

**CRUCIAL:** We are NOT assuming the optimal algorithm uses the same sort order! We are merely imagining reordering the intervals chosen by the optimal algorithm so we can easily compare their finish times to intervals in $G$.
REORDERING O BY INCREASING FINISH TIME

Now O' and G are both ordered by increasing finish time.
This ordering helps us leverage what we know about G in our comparison with O'.
Argue for a prefix of the intervals sorted this way, G chooses as many as O'.

COMPARING O' WITH G

Looks like \( f(G_1) \leq f(O'_1) \) and \( f(G_2) \leq f(O'_2) \)...
If this trend holds in general, then G chooses as many intervals as O'.

PROVING LEMMA: \( f(G_i) \leq f(O'_i) \) FOR ALL \( i \)

Base case: \( f(G_1) \leq f(O'_1) \) since G chooses the interval with the earliest finish time first.

PROVING LEMMA: \( f(G_i) \leq f(O'_i) \) FOR ALL \( i \)

Inductive step: assume \( f(G_{i-1}) \leq f(O'_{i-1}) \). Show \( f(G_i) \leq f(O'_i) \).
• Since O' is feasible, \( f(O'_{i+1}) \leq s(O'_i) \)
• So \( f(G_{i+1}) \leq s(O'_i) \)
• So G can choose \( O'_{i+1} \) if it has the smallest finish time
• So \( f(G_i) \leq f(O'_i) \)

USING THIS LEMMA

• Suppose \( |O'| > |G| \) to obtain a contradiction
  • So if G chooses \( k \) intervals, O' chooses at least \( k+1 \)
  • By the lemma, \( f(G_k) \leq f(O'_k) \)
  • Since O' is feasible, \( f(O'_{k+1}) \leq s(O'_k) \)
  • But then G can, and would, pick \( O'_{k+1} \).
  • Contradiction!

A DIFFERENT PROOF

"Slick" ad-hoc approaches are sometimes possible...
Would be chosen by greedy! (contradiction)

No interval finishes strictly to the left
No interval starts strictly to the right

So, in addition to the intervals in $X$, only the following types of intervals are possible:
- Contains $f_i$
- Contains $f_j$
- Contains $f_i$ and $f_j$

Thus, every interval contains some finishing time in $F$.
And, two intervals in $O$ cannot contain the same element of $F$.

So, there must be as many finishing times in $F$ as there are intervals in $O$. QED

Problem 4.4

Knapsack
Instance: Profits $P = [p_1, \ldots, p_n]$, weights $W = [w_1, \ldots, w_n]$ and a capacity $M$. These are all positive integers.
Feasible solution: An integer $X = [x_1, \ldots, x_n]$ where $\sum_{i=1}^{n} w_i x_i \leq M$.

Gotta respect the weight limit $M$.

POSSIBLE GREEDY STRATEGIES FOR KNAPSACK PROBLEMS

• Strategy 1: consider items in decreasing order of profit (i.e., we maximize the local evaluation criterion $p_i$)
• Let’s try an example input:
  • Profits $P = [20, 50, 100]$
  • Weights $W = [10, 20, 10]$
  • Weight limit $M = 10$
  • Algorithm selects last item for 100 profit
  • Looks optimal in this example.

POSSIBLE GREEDY STRATEGIES FOR KNAPSACK PROBLEMS

• Strategy 1: consider items in decreasing order of profit (i.e., we maximize the local evaluation criterion $p_i$)
• How about a second example input:
  • Profits $P = [20, 50, 100]$
  • Weights $W = [10, 20, 10]$
  • Weight limit $M = 10$
  • Algorithm selects last item for 10 profit
  • Not optimal!
POSSIBLE GREEDY STRATEGIES FOR KNAPSACK PROBLEMS

• Strategy 2: consider items in increasing order of weight (i.e., we minimize the local evaluation criterion \( w_i \)).
  
  Counterexample:
  - Profits \( P = [20, 50, 100] \)
  - Weights \( W = [10, 20, 10] \)
  - Weight limit \( M = 10 \)
  - Algorithm selects first item for 20 profit
  - It could select half of second item, for 25 profit!

• Strategy 3: consider items in decreasing order of profit divided by weight (i.e., we maximize local evaluation criterion \( p_i/w_i \)).
  
  Let’s try our second example input:
  - Profits \( P = [20, 50, 100] \)
  - Weights \( W = [10, 20, 100] \)
  - Weight limit \( M = 10 \)
  - Profit divided by weight
    - \( P/W = [2, 2.5, 1] \)
  - Algorithm selects last item for 100 profit (optimal)

INFORMAL FEASIBILITY ARGUMENT
(SHOULD BE GOOD ENOUGH TO SHOW FEASIBILITY ON ASSESSMENTS)

- Feasibility: all \( x_i \) are in \([0, 1]\) and total weight is \( \leq M \)
- Either everything fits in the knapsack, or:
  - When we exit the loop, weight is exactly \( M \)
  - Every time we write to \( x_i \) it’s either 0, 1 or \((M - \text{weight})/w_i\) where \( \text{weight} + w_i > M \)
  - Rearranging the latter we get \((M - \text{weight})/w_i < 1\)
  - And weight \( \leq M \), so \((M - \text{weight})/w_i \geq 0\)
  - So, we have \( x_i \in [0, 1] \)

Running time complexity?

Can do preprocessing in \( O(n \log n) \) time.

Sort \( A \) by decreasing profit divided by weight

GreedyKnapsack(p, w, M)

Create array \( p[w] \)

for \( i = 0 \) to \( n - 1 \)
  if weight + \( w_i \) \( \leq M \)
    \( x_i = 1 \)
    weight = weight + \( w_i \)
  else
    \( x_i = 0 \)

Return \( x \)
MINOR MODIFICATION TO FACILITATE FORMAL PROOF

```python
GreedyRationalKnapsack(p, w, M)
X = [0, 0, ...]
weight = 0
for i = 1..n
    if weight + w[i] <= M
        weight = weight + w[i]
        X[i] = w[i]
    else
        break
return X
```

**MINOR MODIFICATION TO FACILITATE FORMAL PROOF**

**FORMAL FEASIBILITY ARG**

- Loop Invariant: \( V_j : x_j \in [0, 1] \) and \( \text{weight} = \sum_{i=1}^{n} w_i x_i \leq M \)

- Base case. Initially weight = 0 and \( V_j : x_j = 0 \)
  - So 0 = weight = \( \sum_{i=1}^{n} w_i \cdot 0 = \sum_{i=1}^{n} w_i x_i \leq M \)

- Inductive step.
  - Suppose invariant holds at start of iteration \( i \)
  - Let \( w_i, x'_i \) denote values of weight, \( x_i \) at end of iteration \( i \)
  - Prove invariant holds at end of iteration \( i \)
    - i.e., \( V_j : x'_j \in [0, 1] \) and \( \text{weight}' = \sum_{i=1}^{n} w_i x'_i \leq M \)

**FORMAL FEASIBILITY ARG**

- **Case 1:** weight + \( w_j \leq M \)
  - \( x'_j = 1 \) which is in \( [0, 1] \) (by line 11)
  - weight' = weight + \( w_j \) (by line 12)
  - and this is \( \leq M \) by the case
    - weight = \( \sum_{i=1}^{n} x_i w_i + w_j \) (by invariant)
    - And \( x'_j = x_j \) for all \( k \neq j \) and \( x_j = 0 \) if \( \sum_{i=1}^{n} x_i w_i = x_j w_j + \sum_{i=1}^{n} x_i w_i' \)
    - Rearrange to get \( \sum_{i=1}^{n} x_i w_i = \sum_{i=1}^{n} x_i w_i - x_j w_j \)
    - So weight = \( \sum_{i=1}^{n} x_i w_i - x_j w_j \) + \( x_j w_j = \sum_{i=1}^{n} x_i w_i \)

- **Case 2:** weight = \( w_j > M \)
  - We have \( w_j > M \) - weight and \( M = \text{weight} \geq 0 \)
  - So \( 0 \leq \text{weight} < 1 \) which means \( x'_j \in (0, 1) \)
    - weight' = \( \text{weight} + (M - \text{weight}) \) (by line 8)
    - weight' = \( \sum_{i=1}^{n} x_i w_i + (M - \text{weight}) \) (by invariant)
    - But \( x'_j = x_j \) for all \( k \neq j \) and \( x_j = 0 \) so \( \sum_{i=1}^{n} x_i w_i = x_j w_j + \sum_{i=1}^{n} x_i w_i' \)
    - Rearrange to get \( \sum_{i=1}^{n} x_i w_i = \sum_{i=1}^{n} x_i w_i' - x_j w_j \)
    - So weight' = \( \sum_{i=1}^{n} x_i w_i - x_j w_j + (M - \text{weight}) \)
    - And \( M = \text{weight} = x_j w_j \) so weight = \( \sum_{i=1}^{n} x_i w_i \)

**FORMAL FEASIBILITY ARG**

**FORMAL FEASIBILITY ARG**

**OPTIMALITY = AN EXCHANGE ARGUMENT**

For simplicity, assume that the profit/weight ratios are all distinct, so \( \frac{p_1}{w_1} > \frac{p_2}{w_2} > \cdots > \frac{p_n}{w_n} \)

Suppose the greedy solution is \( X = (x_1, \ldots, x_n) \) and the optimal solution is \( Y = (y_1, \ldots, y_n) \).

We will prove that \( X = Y \), i.e., \( x_j = y_j \) for \( j = 1, 2, \ldots, n \). Therefore there is a unique optimal solution and it is equal to the greedy solution. Suppose \( X \neq Y \), to obtain a contradiction.

Pick the smallest integer \( j \) such that \( x_j \neq y_j \), i.e., \( x_j > y_j \) or \( y_j > x_j \), respectively.

**EXCHANGE ARGUMENT** for proving optimality
What’s the relationship between $y_j$ and $x_j$?

Can we have $y_j > x_j$?

Not greedy would take more of item $j$ if it could.

Not it would be worth less than $x_j$.

Can $y_j$ be all zeros after $y_j$?

Since item $j$ is worth more per unit weight, replacing even a tiny amount of item $k$ with item $j$ will improve the solution.

So, we remove an infinitesimal $d > 0$ of weight of item $k$ and add a weight of item $j$. 

Remove some of item $k$ and replace it with some of item $j$.

How much of item $k$ should we remove?
FEASIBILITY OF $Y'$

- To show $Y'$ is feasible, we show $y'_j \geq 0, y'_j \leq 1$ and $\text{weight}(Y') \leq M$
- Let's show $y'_j \geq 0$
  - By definition, $y'_j = y_j - \frac{\delta}{w_j}$
  - So, $y'_j \geq 0$ if $y_j = \frac{\delta}{w_j} \geq 0$ if $\delta \leq y_j w_j$
  - And we know $y_j$ and $w_j$ are both positive
  - So, this constrains $\delta$ to be smaller than some positive number
  - Therefore, it is possible to choose positive $\delta$ s.t. $y'_j \geq 0$

Existence proof but a non-constructive one

FEASIBILITY OF $Y'$

- To show $Y'$ is feasible, we show $y'_j \geq 0, y'_j \leq 1$ and $\text{weight}(Y') \leq M$
- Now let's show $y'_j \leq 1$
  - By definition, $y'_j = y_j + \frac{\delta}{w_j}$
  - So, $y'_j \leq 1$ if $y_j + \frac{\delta}{w_j} \leq 1$ if $\delta \leq (1 - y_j)w_j$
  - Recall $y_j < y_k$ so $y_j < 1$, which means $(1 - y_j) > 0$
  - So, this constrains $\delta$ to be smaller than some positive number

SUPERIORITY OF $Y'$

- Finally we compute $\text{profit}(Y')$
  - $\text{profit}(Y') = \text{profit}(Y) + \delta \frac{y_j - y'_j}{w_j}$
  - $= \text{profit}(Y) + \delta \frac{y_j - y'_j}{w_j} - \frac{\delta}{w_j} y_j$
  - Since $j$ is before $k$, and we consider items with more profit per unit weight first, we have $\frac{y_j - y'_j}{w_j} > 0$
  - So, if $\delta > 0$ then $\delta \left( \frac{y_j - y'_j}{w_j} - \frac{\delta}{w_j} \right) > 0$
  - Since we can choose $\delta > 0$, we have $\text{profit}(Y') > \text{profit}(Y)$.
WHAT IF ELEMENTS DON'T HAVE DISTINCT PROFIT/WEIGHT RATIOS?

Covering the next 9 slides is homework!

OPTIMALITY PROOF WITHOUT DISTINCTNESS

• There may be many optimal solutions

  • Key idea: Let \( Y \) be an optimal solution that matches \( X \) on a maximal number of indices

  • Observe: If \( X \) is really optimal, then \( Y = X \)

  • Suppose not for contra

  • We will modify \( Y \), preserving its optimality, but making it match \( X \) on one more index (a contradiction!)

Greedy solution \( X \)

Optimal solution \( Y \)

Fraction of item in knapsack

\( x_1 \)

\( x_2 \)

\( y_1 \)

\( y_2 \)

\( y_j \)

\( y_{j-1} \)

\( y_{j-2} \)

\( \vdots \)

\( y_0 \)

0

1

Item 1

Item 2

Item \( n \)

Item \( j \)

Suppose \( \delta = w_j(x_j - y_j) \)

In this case, since \( \delta = w_j(x_j - y_j) \), we end up with \( y_j = x_j \)

If \( \delta \) were \( w_k(x_k - y_k) \), we would have \( y_k = x_k \)

Let \( \delta = \min(w_j(x_j - y_j), w_k(x_k - y_k)) \)

Observe \( \delta > 0 \)

Weight to add: \( w_j(x_j - y_j) \)

Weight to remove: \( w_k(x_k - y_k) \)

Fraction we should add to \( j \) to make solutions equal on index \( j \): \( x_j - y_j \)

Fraction we should remove from \( k \) to make solutions equal on index \( k \): \( y_k - x_k \)

Modified optimal solution \( Y' \)

\( y_j' = y_j + \delta \)

\( y_k' = y_k - \delta \)
To show $Y'$ is feasible, we show $\text{weight}(Y') \leq M$ and $y'_k \geq 0, y'_j \leq 1$.

**Modified optimal solution $Y'$**

We move $\delta$ weight from item $k$ to item $j$.

This does not change the total weight! So $\text{weight}(Y') = \text{weight}(Y) = M$.

**FEASIBILITY OF $Y'$**

- **Showing $y'_k \geq 0$**
  - By definition, $y'_k = y_k - \frac{\delta}{w_k} \geq 0$ iff $\delta \leq w_k y'_k$.
  - But $\delta$ is the minimum of $w_j(y_j - y'_j)$ and $w_k(y_k - x_k) \leq w_k y'_k$.
  - And $w_k(y_k - x_k) \leq w_k y'_k$ so $\delta \leq w_k y'_k$.

- **Showing $y'_j \leq 1$**
  - $y'_j = y_j + \frac{\delta}{w_j} \leq 1$ iff $\frac{\delta}{w_j} \leq 1 - y_j$ iff $\delta \leq w_j(1 - y_j)$ (rearranging).
  - $\delta \leq w_j(y_j - y'_j)$ (definition of $\delta$).
  - And $w_j(y_j - y'_j) \leq w_j(1 - y_j)$ (by feasibility of $X$, i.e., $x_j \leq 1$).

**Profit of $Y'$**

- $\text{profit}(Y') = \text{profit}(Y) + \frac{\delta}{w_j} p_j - \frac{\delta}{w_k} p_k = \text{profit}(Y) + \delta \left( \frac{p_j}{w_j} - \frac{p_k}{w_k} \right)$
  - Since $j$ is before $k$, and we consider items with more profit per unit weight first, we have $\frac{p_j}{w_j} > \frac{p_k}{w_k}$.
  - Since $\delta > 0$ and $\frac{p_j}{w_j} > \frac{p_k}{w_k}$, we have $\delta \left( \frac{p_j}{w_j} - \frac{p_k}{w_k} \right) \geq 0$.
  - Since $Y$ is optimal, this cannot be positive.
  - So $Y'$ is a new optimal solution that matches $X$ on one more index than $Y$.
  - Contradiction: $Y$ matched $X$ on a maximal number of indices.

**SUMMARIZING EXCHANGE ARGUMENTS**

- **If inputs are distinct**
  - So there is a unique optimal solution.
  - Let $O \neq G$ be an optimal solution that beats greedy.
  - Show how to change $O$ to obtain a better solution.

- **If not**
  - There may be many optimal solutions.
  - Let $O \neq G$ be an optimal solution that matches greedy on as many choices as possible.
  - Show how to change $O$ to obtain an optimal solution $O'$ that matches greedy for even more choices.