# Lecture 16 -GREEDY ALGORITHMS

CLRS-Chapter 16

We have already seen two general problem-solving techniques: divide-and-conquer and dynamic-programming. In this section we introduce a third basic technique: the greedy paradigm.

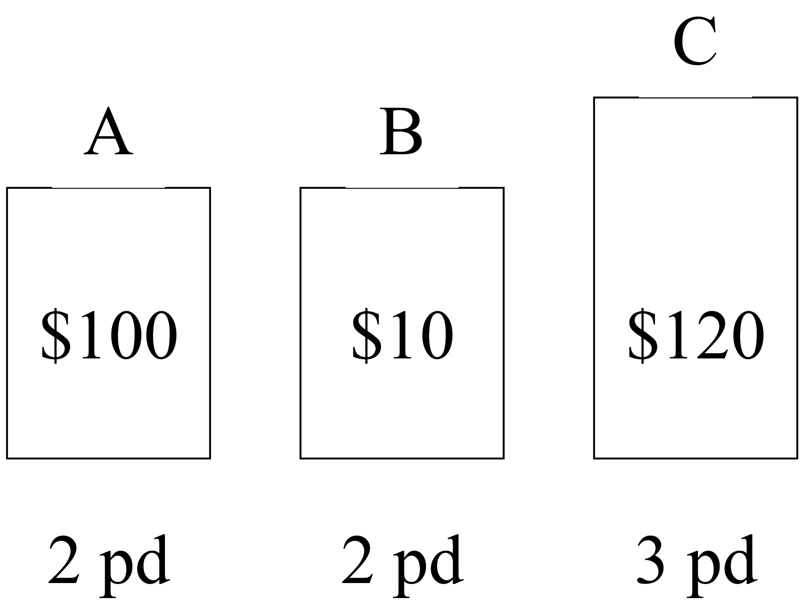
A greedy algorithm for an optimization problem **always makes the choice that looks best at the moment** and adds it to the current subsolution. What's output at the end is an optimal solution. Examples already seen are Dijkstra's shortest path algorithm and Prim/Kruskal's MST algorithms.

Greedy algorithms don't always yield optimal solutions but, when they do, they're usually the simplest and most efficient algorithms available.

# The Knapsack Problem

We review the knapsack problem and see a greedy algorithm for the fractional knapsack. We also see that greedy doesn't work for the 0-1 knapsack (which must be solved using DP).

A thief enters a store and sees the following items:




His Knapsack holds 4 pounds.
What should he steal to maximize profit?

# 1. Fractional Knapsack Problem

Thief can take a fraction of an item.

| Solution = | 2 pounds of item A<br>+ 2 pounds of item C |
|---|---|

| | |
|---|---|
| 2 pds<br>A<br>$100 | 2 pds<br>C<br>$80 |

# 2. 0-1 Knapsack Problem

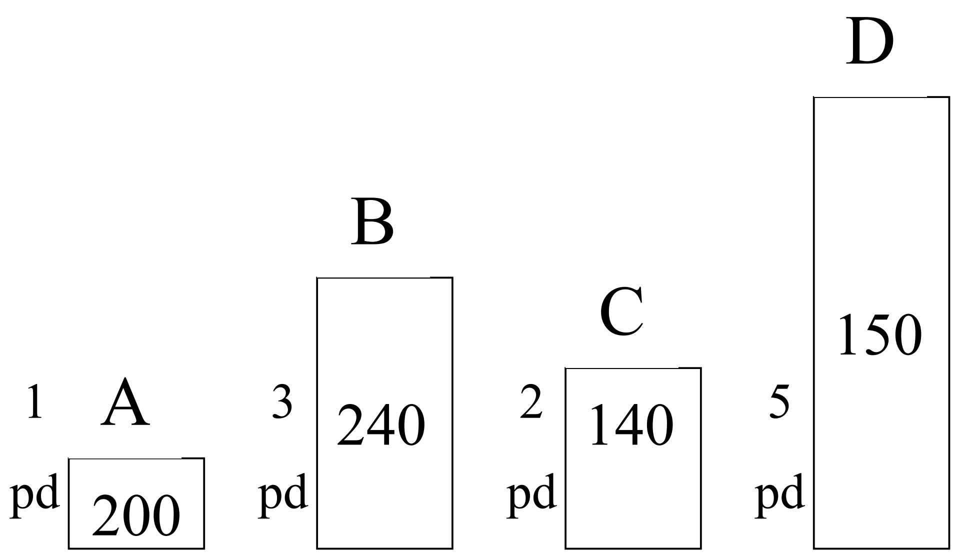
Thief can only take or leave item. He can't take a fraction.

| Solution = | 3 pounds of item C |
|---|---|

| | |
|---|---|
| 3 pds<br>C<br>$120 | |

## Fractional Knapsack has a greedy solution

Sort items by decreasing cost per pound




| | | | | |
|---|---|---|---|---|
| $\frac{\text{cost}}{\text{weight}}$ | 200 | 80 | 70 | 30 |

If knapsack holds k=5 pds, solution is:

1 pds A
3 pds B
1 pds C

# General Algorithm-O(n):

Given:

| | | | | |
|---|---|---|---|---|
| weight | $w_1$ | $w_2$ | … | $w_n$ |
| cost | $c_1$ | $c_2$ | … | $c_n$ |

Knapsack weight limit K

1. Calculate $v_i = c_i / w_i$ for i = 1, 2, …, n

2. Sort the item by decreasing $v_i$

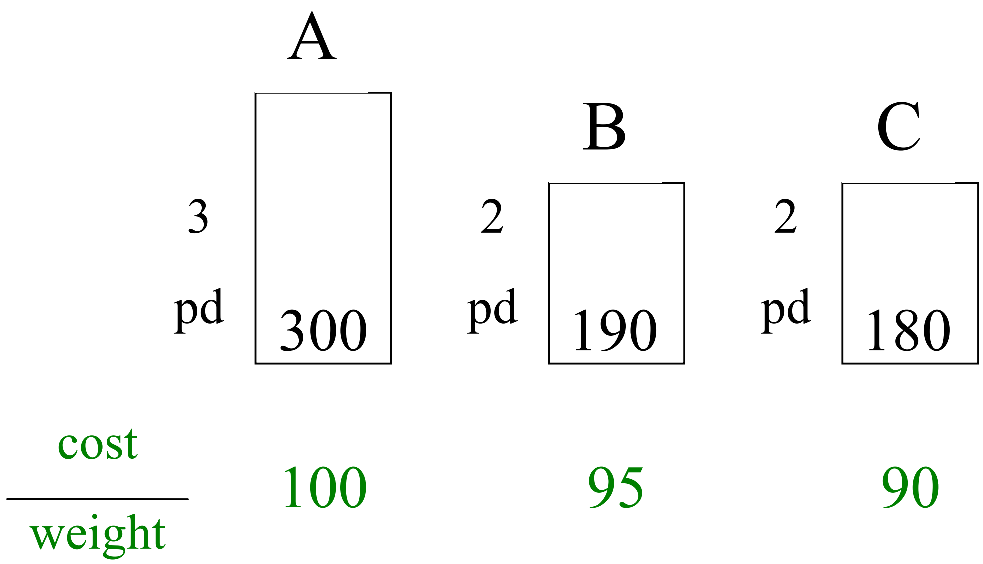
3. Find $j$, s.t.

$$w_1 + w_2 + \ldots + w_j \leq k < w_1 + w_2 + \ldots + w_{j+1}$$

Answer is $\begin{cases} w_i \text{ pds item } i, \text{ for } i \leq j \\ K - \Sigma_{i \leq j} w_i \text{ pds item } j+1 \end{cases}$

# The 0-1 Knapsack Problem does not have a greedy solution!

<u>Example:</u>




K=4

Solution is item B + item C

Best algorithm known is the O(nK) DP one developed earlier.