Fractional Knapsack

Greedy Algorithms

Outline

- Introduction
- The Knapsack problem.
- A greedy algorithm for the fractional knapsack problem
- Correctness

Introduction to Greedy Algorithm

- A greedy algorithm for an optimization problem always makes the choice that looks best at the moment and adds it to the current sub solution.

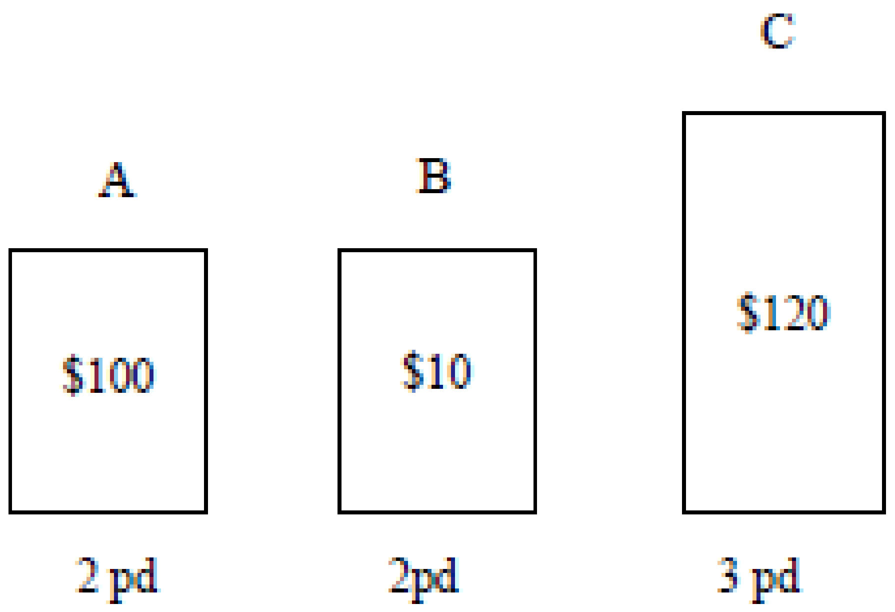
- Final output is an optimal solution.

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

Outline

- Introduction
- The Knapsack problem.
- A greedy algorithm for the fractional knapsack problem
- Correctness

The Knapsack Problem



Capacity of knapsack: K = 4

* Sort by value
* Sort by weight

value per weight

$\frac{60}{10} = 6$ $\frac{100}{20} = 5$ $\frac{120}{30} = 4$

A	B	C
10pd	20pd	30pd
\$60	\$100	\$120

knapsack

B	20d	\$100
C	30pd	\$120

K = 50pd

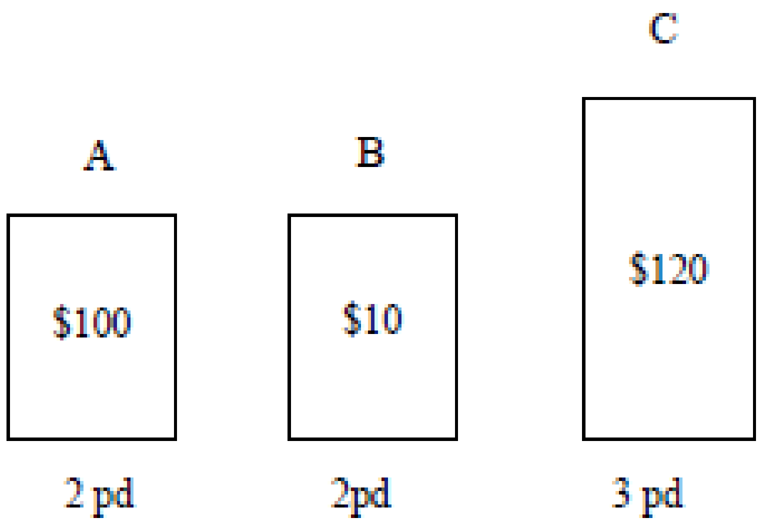
Total Value = \$220

\$60	A	10p
\$100	B	20pd
\$80 $= \frac{2 \times 120}{3}$	C	20pd

$\frac{2}{3} \times 30 = 20$

Total Value = \$240

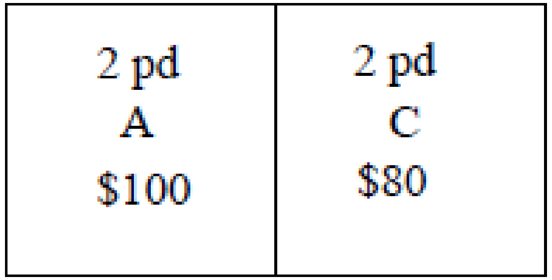
The Knapsack Problem



Capacity of knapsack: $K = 4$

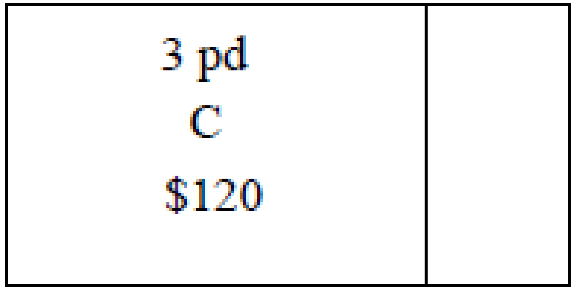
Fractional Knapsack Problem:
Can take a fraction of an item.

Solution:



0-1 Knapsack Problem:
Can only take or leave item. You can't take a fraction.

Solution:



The Fractional Knapsack Problem: Formal Definition

- Given K and a set of n items:

weight	w_1	w_2	...	w_n
value	v_1	v_2	...	v_n

Find: $0 \leq x_i \leq 1$, $i = 1, 2, \ldots, n$ such that

$$\sum_{i=1}^{n} x_i w_i \leq \boxed{K}$$

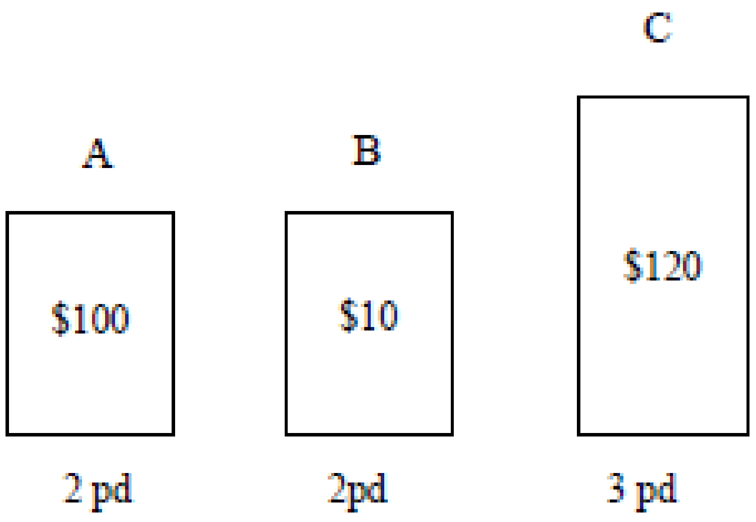
and the following is maximized:

$$\sum_{i=1}^{n} x_i v_i$$

Outline

- Introduction
- The Knapsack problem.
- A greedy algorithm for the fractional knapsack problem
- Correctness

Greedy Solution for Fractional Knapsack



Capacity of knapsack: $K = 4$

Solution 1: Sort by value,
select the item with maximum value first

Solution 1 is not correct
Counter example

C $120 3 pd	A $50 1 pd

Total value = $170

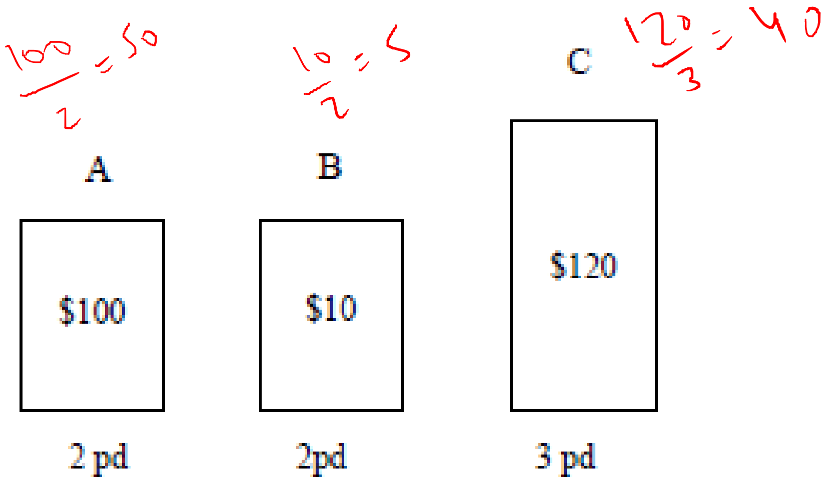
Solution 2: Sort by weight,
select the item with minimum weight first

Solution 2 is not correct
Counter example

A $100 2 pd	B $10 2pd

Total value = $110

Greedy Solution for Fractional Knapsack



Capacity of knapsack: $K = 4$

Solution 3: Sort by value per pound, select the item with maximum value per pound first

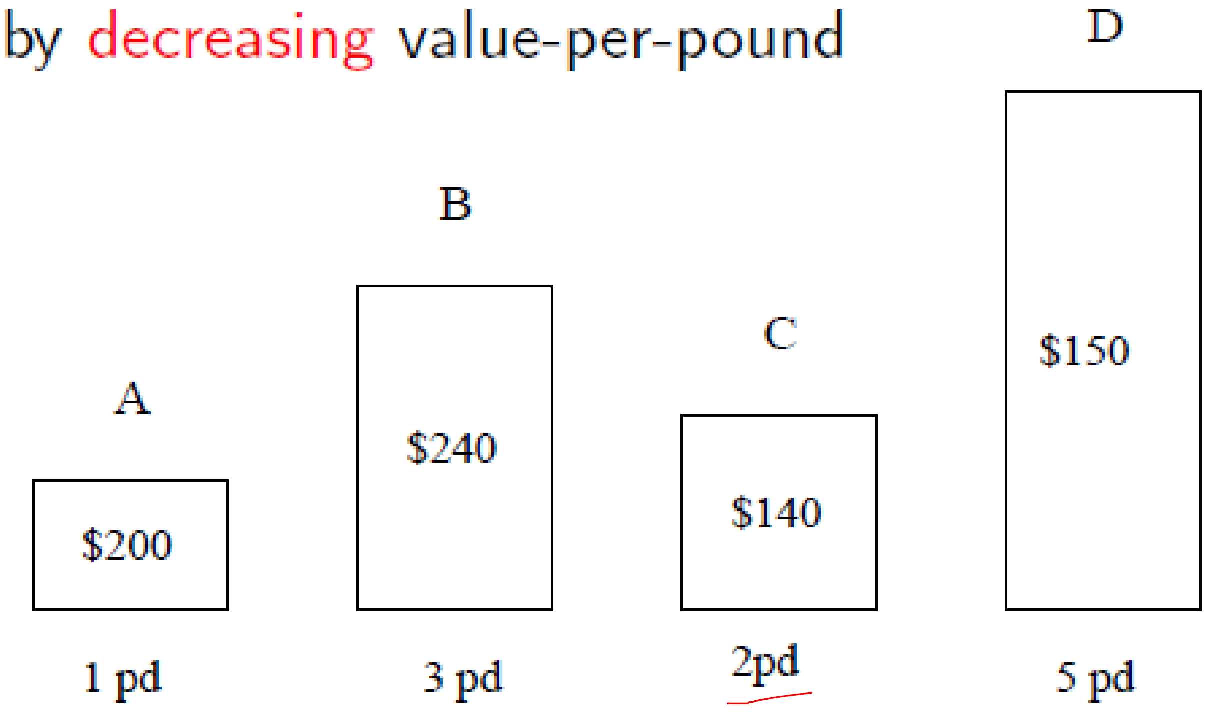
Solution is correct example

A	C
$100	$80
2 pd	2 pd

Total value = $180

Greedy Solution for Fractional Knapsack

Sort items by decreasing value-per-pound



	A	B	C	D
value–per–pound:	200	80	70	30

If knapsack holds $K = 5$ pd, solution is:

1	pd	A
3	pd	B
1	pd	C

1

$\frac{K \times (w_i)}{w_i} = K$

$\frac{K}{w_i}$

$\left(\frac{1}{2}\right)$

0.5

Greedy Solution for Fractional Knapsack

- Calculate the value-per-pound $\rho_i = \frac{v_i}{w_i}$ for $i = 1, 2, \ldots, n$. $\Theta(n)$
- Sort the items by decreasing ρ_i.
  Let the sorted item sequence be $1, 2, \ldots, i, \ldots n$, and the corresponding value-per-pound and weight be ρ_i and w_i respectively. $\Theta(n \lg n)$
- Let k be the current weight limit (Initially, $k = K$).
  In each iteration, we choose item i from the head of the unselected list. $\Theta(n)$
  - If $k \geq w_i$, set $x_i = 1$ (we take item i), and reduce $k = k - w_i$, then consider the next unselected item.
  - If $k < w_i$, set $x_i = k/w_i$ (we take a fraction k/w_i of item i), Then the algorithm terminates.

Running time: $O(n \log n)$.

$n + n\lg n + n = \Theta(n \lg n)$

Greedy Solution for Fractional Knapsack

- Observe that the algorithm may take a fraction of an item. This can only be the last selected item.

- We claim that the total value for this set of items is the optimal value.

0.25, 1, 0.75

1, 1, 0.25, 0, 0

Outline

- Introduction
- The Knapsack problem.
- A greedy algorithm for the fractional knapsack problem
- Correctness

Correctness

Given a set of n items $\{1, 2, \ldots, n\}$.

- Assume items sorted by per-pound values: $\rho_1 \geq \rho_2 \geq \ldots \geq \rho_n$.

Let the greedy solution be $G = \langle x_1, x_2, \ldots, x_k \rangle$

- x_i indicates fraction of item i taken (all $x_i = 1$, except possibly for $i = k$).

Consider any optimal solution $O = \langle y_1, y_2, \ldots, y_n \rangle$

- y_i indicates fraction of item i taken in O (for all i, $0 \leq y_i \leq 1$).
- Knapsack must be full in both G and O:
  $\sum_{i=1}^{n} x_i w_i = \sum_{i=1}^{n} y_i w_i = K$.

Consider the first item i where the two selections differ.

- By definition, solution G takes a greater amount of item i than solution O (because the greedy solution always takes as much as it can). Let $x = x_i - y_i$.

Ordered by value per weight

1 2 3 ... i ... n

$G = \langle x_1 \; x_2 \; x_3 \dots x_i \dots x_n \rangle$ $\quad 0 \le x_i \le 1$

$O = \langle y_1 \; y_2 \; y_3 \dots y_i \dots y_n \rangle$ $\quad 0 \le y_i \le 1$

k

$x_i \neq y_i$

gold silver

Total weight of O' = Total weight of O

$\boxed{x_i > y_i}$

1, 1 (0.85), 0 0 0

subtract: $(x_i - y_i) w_i$ from $y_{i+1} \dots y_n$

25

$O' = \langle y_1 \; y_2 \; y_3 \dots x_i \; y_i \dots \rangle$

k

y_i 75

x_i 100

Total Value Total Value

Total value of O' = of O

Correctness

Consider the following new solution O' constructed from O:

$x = (x_i - y_i) w_i$

- For $j < i$, keep $y'_j = y_j$.
- Set $y'_i = x_i$.
- In O, remove items of total weight xw_i from items $i + 1$ to n, resetting the y'_j appropriately.

This is always doable because $\sum_{j=i}^{n} x_j = \sum_{j=i}^{n} y_j$

- The total value of solution O' is greater than or equal to the total value of solution O (why?)
- Since O is largest possible solution and value of O' cannot be smaller than that of O, O and O' must be equal.
- Thus solution O' is also optimal.

By repeating this process, we will eventually convert O into G, without changing the total value of the selection. **Therefore G is also optimal!**