Greedy Algorithms: The Fractional Knapsack

Version of September 17, 2016

THE DEPARTMENT OF COMPUTER SCIENCE
計算機科學系 HKUST

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 subsolution.

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 subsolution.
- Final output is an optimal solution.

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 subsolution.
- Final output is an optimal solution.
- Greedy algorithms don't always yield optimal solutions

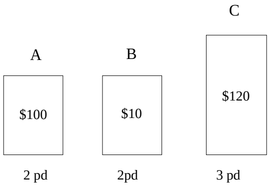
Introduction to Greedy Algorithm

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

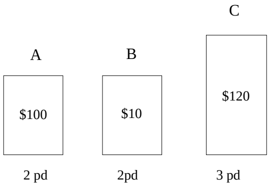
- Introduction
- The Knapsack problem.
- A greedy algorithm for the fractional knapsack problem
- Correctness

The Knapsack Problem...



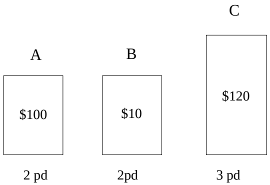
Capacity of knapsack: $K = 4$

The Knapsack Problem...



Capacity of knapsack: $K = 4$

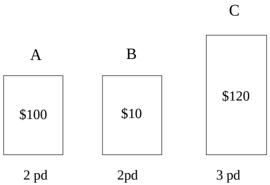
Fractional Knapsack Problem:



Capacity of knapsack: $K = 4$

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

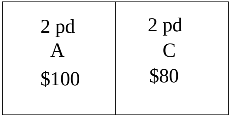
The Knapsack Problem...

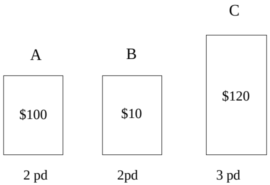


Capacity of knapsack: $K = 4$

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

Solution:

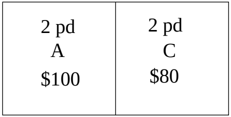




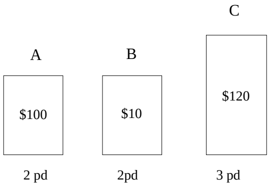
Capacity of knapsack: $K = 4$

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

Solution:



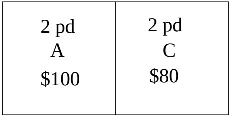
0-1 Knapsack Problem:
Can only take or leave item. You can't take a fraction.



Capacity of knapsack: $K = 4$

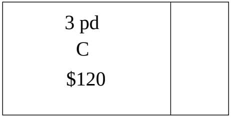
Fractional Knapsack Problem:
Can take a fraction of an item.

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	$\ldots$	w_n
value	v_1	v_2	$\ldots$	v_n

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

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

and the following is maximized:

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

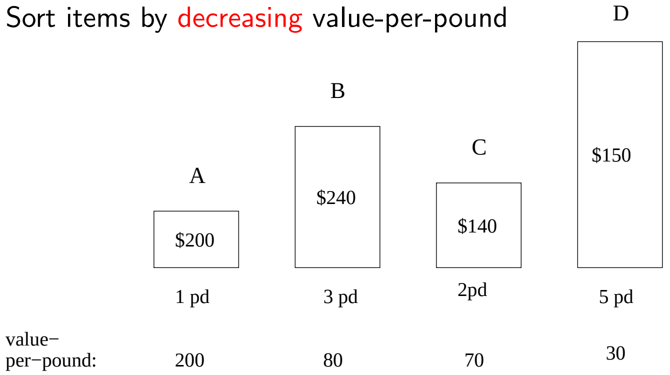
Outline

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

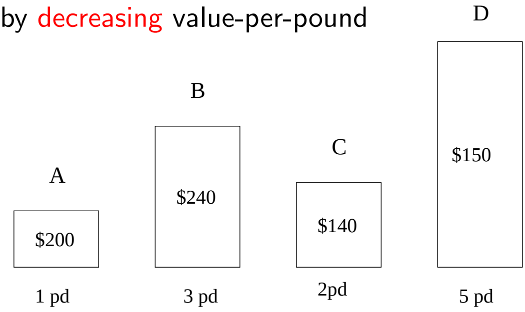
Greedy Solution for Fractional Knapsack

Sort items by decreasing value-per-pound

Sort items by decreasing value-per-pound



Sort items by decreasing value-per-pound



value−per−pound:	200	80	70	30

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

1	pd	A
3	pd	B
1	pd	C

Greedy Solution for Fractional Knapsack

- Calculate the value-per-pound $\rho_i = \frac{v_i}{w_i}$ for $i = 1, 2, \ldots, n$.

Greedy Solution for Fractional Knapsack

- Calculate the value-per-pound $\rho_i = \frac{v_i}{w_i}$ for $i = 1, 2, \ldots, n$.
- Sort the items by decreasing ρ_i.

Greedy Solution for Fractional Knapsack

- Calculate the value-per-pound $\rho_i = \frac{v_i}{w_i}$ for $i = 1, 2, \ldots, n$.
- Sort the items by decreasing ρ_i.
  Let the sorted item sequence be $1, 2, \ldots, i, \ldots n$,

- Calculate the value-per-pound $\rho_i = \frac{v_i}{w_i}$ for $i = 1, 2, \ldots, n$.
- 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.

Greedy Solution for Fractional Knapsack

- Calculate the value-per-pound $\rho_i = \frac{v_i}{w_i}$ for $i = 1, 2, \ldots, n$.
- 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.
- Let k be the current weight limit (Initially, $k = K$).

Greedy Solution for Fractional Knapsack

- Calculate the value-per-pound $\rho_i = \frac{v_i}{w_i}$ for $i = 1, 2, \ldots, n$.
- 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.
- Let k be the current weight limit (Initially, $k = K$).
  In each iteration, we choose item i from the head of the unselected list.

Greedy Solution for Fractional Knapsack

- Calculate the value-per-pound $\rho_i = \frac{v_i}{w_i}$ for $i = 1, 2, \ldots, n$.
- 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.
- Let k be the current weight limit (Initially, $k = K$).
  In each iteration, we choose item i from the head of the unselected list.
  - If $k \geq w_i$,

Greedy Solution for Fractional Knapsack

- Calculate the value-per-pound $\rho_i = \frac{v_i}{w_i}$ for $i = 1, 2, \ldots, n$.
- 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.
- Let k be the current weight limit (Initially, $k = K$).
  In each iteration, we choose item i from the head of the unselected list.
  - 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.

Greedy Solution for Fractional Knapsack

- Calculate the value-per-pound $\rho_i = \frac{v_i}{w_i}$ for $i = 1, 2, \ldots, n$.
- 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.
- Let k be the current weight limit (Initially, $k = K$).
  In each iteration, we choose item i from the head of the unselected list.
  - 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 $k < w_i$,

Greedy Solution for Fractional Knapsack

- Calculate the value-per-pound $\rho_i = \frac{v_i}{w_i}$ for $i = 1, 2, \ldots, n$.
- 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.
- Let k be the current weight limit (Initially, $k = K$).
  In each iteration, we choose item i from the head of the unselected list.
    - 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),

Greedy Solution for Fractional Knapsack

- Calculate the value-per-pound $\rho_i = \frac{v_i}{w_i}$ for $i = 1, 2, \ldots, n$.
- 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.
- Let k be the current weight limit (Initially, $k = K$).
  In each iteration, we choose item i from the head of the unselected list.
  - 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.

Greedy Solution for Fractional Knapsack

- Calculate the value-per-pound $\rho_i = \frac{v_i}{w_i}$ for $i = 1, 2, \ldots, n$.
- 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.
- Let k be the current weight limit (Initially, $k = K$).
  In each iteration, we choose item i from the head of the unselected list.
  - 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: $O(n \log n)$.

Greedy Solution for Fractional Knapsack

- Observe that the algorithm may take a fraction of an item.

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

- 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.

Outline

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

Correctness

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

Correctness

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

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

Correctness

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

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

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

Correctness

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

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

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

- x_i indicates fraction of item i taken

Correctness

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

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

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

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

Correctness

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

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

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

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

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

Correctness

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

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

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

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

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

- y_i indicates fraction of item i taken in O

Correctness

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

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

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

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

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

- y_i indicates fraction of item i taken in O $\big($for all i, $0 \leq y_i \leq 1\big)$.

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

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

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

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

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

- 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:
  $\sum_{i=1}^{n} x_i w_i = \sum_{i=1}^{n} y_i w_i = K$.

Correctness

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

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

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

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

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

- y_i indicates fraction of item i taken in O $\big($for all i, $0 \leq y_i \leq 1\big)$.
- 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.

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

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

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

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

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

- y_i indicates fraction of item i taken in O $\big($for all i, $0 \leq y_i \leq 1\big)$.
- 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

Correctness

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

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

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

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

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

- y_i indicates fraction of item i taken in O $\big($for all i, $0 \leq y_i \leq 1\big)$.
- 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).

Correctness

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

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

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

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

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

- y_i indicates fraction of item i taken in O $\left(\text{for all } i,\ 0 \leq y_i \leq 1\right)$.
- 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 $x = x_i - y_i$.

Consider the following new solution O' constructed from O:

Consider the following new solution O' constructed from O:

- 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.

Correctness...

Consider the following new solution O' constructed from O:

- 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?)

Consider the following new solution O' constructed from O:

- 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.

Consider the following new solution O' constructed from O:

- 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.

Correctness...

Consider the following new solution O' constructed from O:

- 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.

Consider the following new solution O' constructed from O:

- 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.

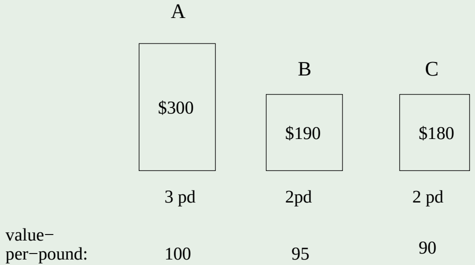
  This is always doable because $\sum_{j=i}^{n} x_j = \sum_{j=i}^{n} y_j$
- The total value of solution O' is greater than or equal to the total value of solution O (why?)
- Since O is largest possible solution and value of O' cannot be smaller than that of O, O and O' must be equal.
- Thus solution O' 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!

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

Example

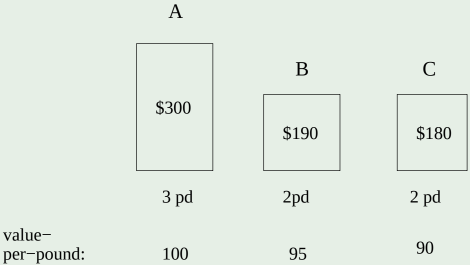


$K = 4$. Solution is

Greedy solution for 0-1 Knapsack Problem?

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

Example

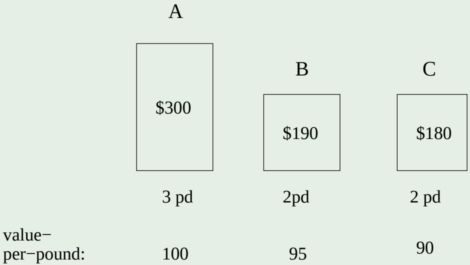


$K = 4$. Solution is item B + item C

Greedy solution for 0-1 Knapsack Problem?

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

Example



$K = 4$. Solution is item B + item C

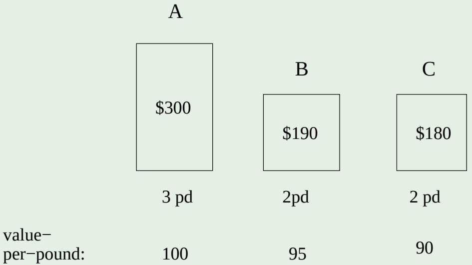
Question

Suppose we tried to prove the greedy algorithm for 0-1 knapsack problem **does** construct an optimal solution.

Greedy solution for 0-1 Knapsack Problem?

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

Example



$K = 4$. Solution is item B + item C

Question

Suppose we tried to prove the greedy algorithm for 0-1 knapsack problem **does** construct an optimal solution. If we follow exactly the same argument as in the fractional knapsack problem where does the proof fail?