Greedy Method

CLRS 16.1 – 16.3

(+ some supplemental material)

Greedy Method Technique

- The greedy method is a general algorithm design paradigm, built on the following elements:
  - configurations: different choices, collections, or values to find
  - objective function: a score assigned to configurations, which we want to either maximize or minimize

- Idea: make a greedy choice (locally optimal) in **hopes** it will eventually lead to a globally optimal solution.

- It works best when applied to problems with the greedy-choice property
  - a globally-optimal solution can always be found by a series of local improvements from a starting configuration.

- Example: climbing a hill

Example: Making Change

- **Problem**: A dollar amount to reach and a collection of coin amounts to use to get there.
  - configuration: A dollar amount yet to return to a customer plus the coins already returned
  - objective function: Minimize number of coins returned.
- **Greedy solution**: Always return the largest coin you can.

- Ex. 1: Coins are valued $.32, $.08, $.01
  - Has the greedy-choice property, since no amount over $.32 can be made with a minimum number of coins by omitting a $.32 coin (similarly for amounts over $.08, but under $.32).

- Ex. 2: Coins are valued $.30, $.20, $.05, $.01
  - Does not have greedy-choice property, since $.40 is best made with two $.20's, but the greedy solution will pick three coins (which ones?)

Example: Knapsack Problem

- Given: A set S of n items, with each item i having:
  - b_i a positive benefit
  - w_i a positive weight
- Goal: Choose items with maximum total benefit but with weight at most W.

There are two common variations:

- 0-1 Knapsack Problem: we can either leave an item (0) or take it in its entirety (1)
- **Fractional Knapsack Problem**: we can take a fractional amount of an item

Only one of these can be solved with a greedy approach!

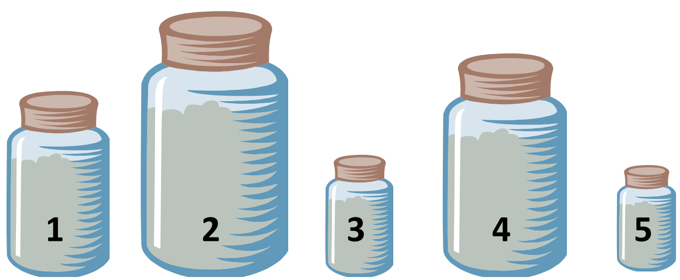
Example: **Fractional** Knapsack Problem

- Given: A set S of n items, with each item i having:
  - b_i a positive benefit
  - w_i a positive weight
- Note: we can take a fractional amount $x_i \leq w_i$ of an item i
- Goal: Choose items with maximum total benefit but with weight at most W.

Objective: maximize $\sum_{i \in S} b_i (x_i / w_i)$

Constraint: $\sum_{i \in S} x_i \leq W$

Items:



	1	2	3	4	5
Weight:	4 ml	8 ml	2 ml	6 ml	1 ml
Benefit:	\$12	\$32	\$40	\$30	\$50
Value: (\$ per ml)	3	4	20	5	50

“knapsack”

10 ml

Solution:

- 1 ml of item 5
- 2 ml of item 3
- 6 ml of item 4
- 1 ml of item 2

Example: Fractional Knapsack Problem

- Greedy choice: Keep taking item with highest **value** (benefit to weight ratio)

Algorithm ***fractionalKnapsack***(***S***, ***W***)

Input: set ***S*** of items w/ benefit b_i and weight w_i; max. weight W

Output: amount x_i of each item i to maximize benefit with weight at most W

```
for each item i in S
    x_i ← 0
    v_i ← b_i / w_i        {value}
w ← 0                      {total weight}
while w < W
    remove item i with highest v_i
    x_i ← min{w_i , W − w}
    w ← w + x_i
```

Run time: $O(n \log n)$ - why?

Proof of correctness: We must establish that this problem has the greedy choice property. Use a proof by contradiction.

Suppose there is a optimal solution S* better than our greedy solution S.

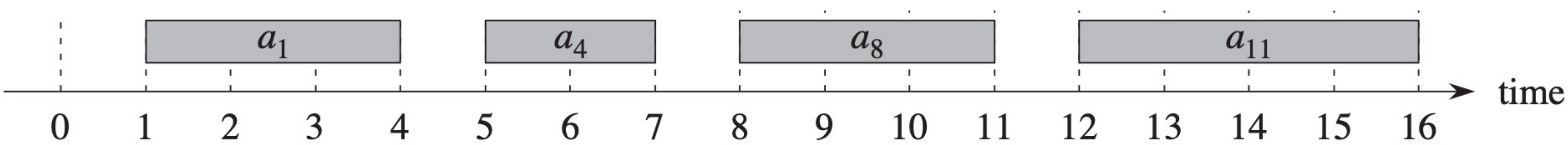
- There is an item *i* in S with higher value than a chosen item *j* from S*, i.e., $v_i > v_j$ but $x_i < w_i$ and $x_j > 0$.
- If we substitute some *i* with *j*, we get a better solution in S*, a contradiction
  - How much of *i*: $\min\{w_i - x_i, x_j\}$
- Thus, there is no better solution than the greedy one

Example: Activity Selection

- Given: A set S of n activities that wish to use a resource, with each activity a_i having a start time s_i and finish time f_i; the activity takes place during $[s_i, f_i)$
- Goal: Select a maximum-size subset of mutually non-conflicting activities

- Example with 11 activities
- How many are non-overlapping?

i	1	2	3	4	5	6	7	8	9	10	11
s_i	1	3	0	5	3	5	6	8	8	2	12
f_i	4	5	6	7	9	9	10	11	12	14	16



- What greedy choice would you make?
- Does it satisfy the greedy choice property? Prove it.