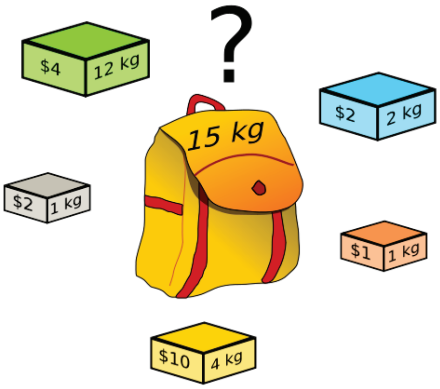
Dynamic Programming Algorithms

Knapsack

CS 336: Design and Analysis of Algorithms



Knapsack

Picture credit: Wikipedia.com

Knapsack Problem

We are given n items 1,…,n with weights $w_1, \dots, w_n$ and values $v_1, \dots, v_n$. Our goal is to find a collection of items $S \subset \{1, \dots, n\}$ of maximum possible value that fits in a knapsack of size W.

Goal: $\text{maximize } \sum_{i \in S} v_i$

Subject to the *packing constraint*: $\sum_{i \in S} w_i \le W$.

Knapsack Problem

We are given n items 1,…,n with weights $w_1, \dots, w_n$ and values $v_1, \dots, v_n$. Our goal is to find a collection of items $S \subset \{1, \dots, n\}$ of maximum possible value that fits in a knapsack of size W.

Goal: $\text{maximize } \sum_{i \in S} v_i$

Subject to the *packing constraint*: $\sum_{i \in S} w_i \leq W$.

Today, we will assume that all weights w_i and W are “small” integers.

Warm-up: All weights w_i are equal to 1

- What shall we do?

Warm-up: $W \gg \max_i w_i$

- What shall we do now?

Warm-up: $W \gg \max_i w_i$

- What shall we do now?
- Pick items with maximum value per unit of weight v_i/w_i.

DP for Knapsack

What is the right subproblem for Knapsack?

- Approach 1: Find the optimum solution for items $\{1, \ldots, i\}$.
  - What is the optimum? Maximum value or minimum weight?

DP for Knapsack

What is the right subproblem for Knapsack?

- Approach 1: Find the optimum solution for items $\{1, \ldots, i\}$.
- Approach 2: MaxKnapsack (i, W') – the maximum value of items from the set $\{1, \ldots, i\}$ we can pack in Knapsack of size W'.

To solve the original Knapsack problem we need to find MaxKnapsack(n, W).

Recursive Relation

- Two options:
  a. We put item i in the knapsack; or
  b. We don’t put item i in the knapsack.

$\text{MaxKnapsack}(i, W') = \max\{$
a. $v_i + \text{MaxKnapsack}(i-1, W'-w_i),$
b. $\text{MaxKnapsack}(i-1, W')\}$

DP Algorithm

```
for $i = 1, \ldots, n$
  for $W' = 1, \ldots, W$
     $\text{OptionA} = v_i + \text{MaxKnapsack}(i-1, W'-w_i);$
     $\text{OptionB} = \text{MaxKnapsack}(i-1, W');$
    $\text{MaxKnapsack}(i, W') = \max(\text{OptionA}, \text{OptionB});$
```

Are we missing anything?

DP Algorithm

```
for $i = 1, \ldots, n$
  for $W' = 1, \ldots, W$
     $\text{OptionA} = v_i + \text{MaxKnapsack}(i-1, W'-w_i);$
     $\text{OptionB} = \text{MaxKnapsack}(i-1, W');$
    $\text{MaxKnapsack}(i, W') = \max(\text{OptionA}, \text{OptionB});$
```

Are we missing anything? Need to handle the case of $i = 1$ and $w_i > W'$ separately.