CSE 421
Intro to Algorithms
Winter 2004

The Fraction Knapsack Problem:
A Greedy Example

Fractional Knapsack

Given:

A knapsack of n items with: Capacity: W
Weights: $w_1, w_2, \ldots, w_n$
Values: $v_1, v_1, \ldots, v_n$

Find:

$\alpha_1, \alpha_2, \ldots, \alpha_n$, maximizing $\sum_{i=1}^{n} \alpha_i v_i$

Subject to: $0 \le \alpha_i \le 1$, and $\sum_{i=1}^{n} \alpha_i w_i = W$

[Note: "0-1 Knapsack" same, except α_i = 0 or 1.]

Examples

Object	Weight	Value
Liqueur-Filled Bon Bons	1	$12
Dark Chocolate Truffles	2	$18
Milk Choc. Spring Surprise	3	$24

W = 5

i	w_i	v_i	α_i	$\alpha_i v_i$	α_i	$\alpha_i v_i$	α_i	$\alpha_i v_i$
BB	1	$12	0	$0	5/6	$10	1	$12
T	2	$18	1	$18	5/6	$15	1	$18
SS	3	$24	1	$24	5/6	$20	2/3	$16
Total				$42		$45		$46

Greedy Solution

- Order by decreasing value per unit weight (renumbering as needed)

$$\frac{v_1}{w_1} \ge \frac{v_2}{w_2} \ge \cdots \ge \frac{v_n}{w_n}$$

- Take as much 1 as possible, then as much 2 as possible, …

The Greedy Choice Pays

Claim 1: ∃ an optimal solution with as much as possible of item 1 in the knapsack, namely $\min(w_1, W)$. Equivalently $\alpha_1 = \min(w_1, W)/w_1$.

Proof: Among all optimal solutions, let $\beta_1, \beta_2, \ldots, \beta_n$ be one with maximum β_1, but suppose (for the sake of contradiction) $\beta_1 < \alpha_1$. Since β has less of 1 than α, it must have more of something else, say j, i.e. $\beta_j > \alpha_j$. Form β' from β by carrying a little more 1 and less j, say $\varepsilon = \min((\beta_j - \alpha_j)\, w_j, (\alpha_1 - \beta_1)\, w_1) > 0$. Then β' will not have a lower value than β, since $\varepsilon(v_1/w_1 - v_j/w_j) \ge 0$, but $\beta_1' > \beta_1$, contradicting our choice of β. QED

Optimal Sub-solutions

Claim 2: The best solution for any given α_1 has $\alpha_2, \ldots, \alpha_n$ equal to an optimal solution for the smaller knapsack problem having items 2, 3, …, n and capacity $W - \alpha_1 w_1$.

Proof: If not, we could get a better solution.

Keys to Greedy Algorithms

"Greedy Choice Property":

Making a locally optimal ("greedy") 1st step cannot prevent reaching a global optimum.

[E.g., see Claim 1.]

"Optimal Substructure":

The optimal solution to the problem contains optimal solutions to subproblems.

[E.g., see Claim 2. True of Dynamic Programming, too.]

NOTE:

- Greedy algorithms are very natural for optimization problems, but
- *they don't always work*
- E.g., if you try greedy approach for 0-1 knapsack on the candy example, it will choose to take all of BB & T, for a total value of $30, well below the optimal $42
- So: *Correctness proofs are important!*