The Fractional Knapsack Problem

We are given n items $I_1, I_2, \ldots, I_i, \ldots, I_n$
having weights $w_1, w_2, \ldots, w_i, \ldots, w_n$
and profits $p_1, p_2, \ldots, p_i, \ldots, p_n$
A knapsack capacity W is given. Our objective is to choose the items (we can also choose fraction of an item) in such a way that the selected items fit into the knapsack (total weight of the selected items is less than or equal to W), and the profit is maximized. Suppose we choose fraction x_i of I_i $(0 \le x_i \le 1)$. It will contribute $w_i x_i$ weight and $p_i x_i$ profit. Total weight of selected items will be $\sum_{i=1}^{n} w_i x_i$ and total profit of selected items will be $\sum_{i=1}^{n} p_i x_i$.
We can write a Linear Programming (LP) formulation of the fractional knapsack problem as follows:

Maximize $\sum_{i=1}^{n} p_i x_i$ subject to

$$\sum_{i=1}^{n} w_i x_i \le W \quad \text{and}$$

$$0 \le x_i \le 1 \quad \forall\, i \in [1 \ldots n]$$

Example:

	I_1	I_2	I_3	I_4	I_5
w_i :	1	2	3	4	5
p_i :	5	4	3	2	1

$W = 8$, optimal solution = $(1, 1, 1, \frac{1}{2}, 0)$ having profit 13.

Greedy-knapsack algorithm can be implemented in time $O(n \log n)$ to solve the Fractional knapsack problem

A Greedy Algorithm for Solving the Fractional Knapsack Problem

```
Greedy-knapsack((w_i)_{i=1}^n, (p_i)_{i=1}^n, W)
1   Sort the items in non-increasing order of p_i/w_i values.
2   Assuming p_1/w_1 ≥ p_2/w_2 ≥ ... ≥ p_i/w_i ≥ ... p_n/w_n
3   if n = 1
4       if w_1 ≤ W
5           return (1)                    → No need to solve any subproblem
6       else
7           return (W/w_1)
8   if w_1 ≥ W                            ↑ Optimal solution to the original problem contains within it optimal solutions to subproblems. This is called Optimal Substructure
9       return (W/w_1, 0, ..., 0)
10  else
11      (y_2, y_3, ..., y_n) ← Greedy-knapsack((w_2, w_3, ... w_n),
                                (p_2, p_3, ..., p_n), W − w_1)
12      return (1, y_2, y_3, ..., y_n)
                ↳ Partial information about the optimal solution is known even before solving the subproblem. Choosing w_1 (most profitable item) is a locally optimal choice (we are not solving any subproblems to choose w_1) which leads to a globally optimal solution. This is called the Greedy-choice Property
```

Proof of Greedy-Choice Property for the Fractional Knapsack Problem: Take the most profitable item having $\max_{i=1}^{n}\left(\frac{p_i}{w_i}\right)$ (we call it item 1 in Greedy-knapsack). Optimality of lines 5 and 7 is obvious. If there is only one item, we take it (or its fraction) as much as we can to maximize our profit. Consider line 9. We have $n>1$ and $w_1 \geq W$.

our profit is $p_1\left(\frac{W}{w_1}\right) = \left(\frac{p_1}{w_1}\right)W = P(\text{Greedy-knapsack}) \longrightarrow (1)$

Consider any other solution $(y_1, y_2, \ldots, y_n)$.

We have $\sum_{i=1}^{n} w_i y_i \leq W \longrightarrow (2)$

and profit is $\sum_{i=1}^{n} p_i y_i = \sum_{i=1}^{n}\left(\frac{p_i}{w_i}\right)(w_i y_i) = P(\text{other-Algo}) \longrightarrow (3)$

From (1), (2), and (3), we get (assuming $\frac{p_1}{w_1} \geq \frac{p_2}{w_2} \geq \cdots \geq \frac{p_i}{w_i} \geq \cdots \frac{p_n}{w_n}$)

$$P(\text{other-Algo}) = \sum_{i=1}^{n}\left(\frac{p_i}{w_i}\right)(w_i y_i) \leq \sum_{i=1}^{n}\left(\frac{p_1}{w_1}\right)(w_i y_i)$$

$$= \left(\frac{p_1}{w_1}\right)\sum_{i=1}^{n} w_i y_i \leq \left(\frac{p_1}{w_1}\right)W = P(\text{Greedy-knapsack})$$

Therefore, our solution is optimal.

Now consider line 12: We have $n>1$ and $w_1 < W$.

Our claim is that there exists an optimal solution with $y_1 = 1$.

We will prove our claim by using Exchange Argument.

Consider any solution $(y_1, y_2, \ldots, y_n)$ with $y_1 < 1$.

Consider two cases:

case 1: all $y_2, y_3, \ldots, y_n$ are 0

$P(\text{other-Algo}) = y_1 p_1 < p_1 = P(\text{Greedy-knapsack})$

case 2: Some y_j $(2 \leq j \leq n)$ is greater than 0.

Exchange some fraction z_j $(0 < z_j \leq y_j)$ of w_j with w_1 such that

$y_1 w_1 + z_j w_j \leq w_1$

Weight of w_j is reduced to $w_j - z_j w_j$

Weight of w_1 is increased to $w_1 y_1 + z_j w_j$

Total weight of items remains same $= \sum_{i=1}^{n} w_i y_i \leq W$

$$P(\text{Before Exchange}) = \sum_{i=1}^{n} p_i y_i = p_1 y_1 + p_j y_j + \sum_{\substack{i=2 \\ i \neq j}}^{n} p_i y_i$$

$$P(\text{After Exchange}) = p_1\left(y_1 + \frac{z_j w_j}{w_1}\right) + p_j(y_j - z_j) + \sum_{\substack{i=2 \\ i \neq j}}^{n} p_i y_i$$

$$= \left(\sum_{i=1}^{n} p_i y_i\right) + \left(\frac{p_1}{w_1}\right)(z_j w_j) - \left(\frac{p_j}{w_j}\right)(z_j w_j)$$

$$= P(\text{Before Exchange}) + z_j w_j\left(\frac{p_1}{w_1} - \frac{p_j}{w_j}\right)$$

$$\Rightarrow P(\text{After Exchange}) - P(\text{Before Exchange})$$

$$= z_j w_j\left(\frac{p_1}{w_1} - \frac{p_j}{w_j}\right) \geq 0 \quad \left(\text{assuming } \frac{p_1}{w_1} \geq \frac{p_j}{w_j}\right)$$

We can keep doing exchanges until we get $y_1 = 1$ ($w_1 < w$)

$\Rightarrow$ P(Greedy-knapsack) is optimal.

Greedy Algorithm Design Steps

(1) Cast the optimization problem as one in which we make a choice and are left with one subproblem to solve.

(2) Prove that there is always an optimal solution to the original problem that makes the greedy choice, so that the greedy choice is always safe.

(3) Demonstrate optimal substructure by showing that, having made the greedy choice, what remains is a subproblem with the property that if we combine an optimal solution to the subproblem with the greedy choice we have made, we arrive at an optimal solution to the original problem.

Optimal Substructure of Greedy knapsack: We use induction on n. Base case ($n=1$) is trivial. We assume that line 11 gives optimal solution. Line 12 will give the optimal solution because of the greedy choice property (there exists an optimal solution with $y_1 = 1$).