Fractional Knapsack Problem

The fractional knapsack problem is also one of the techniques which are used to solve the knapsack problem. In fractional knapsack, the items are broken in order to maximize the profit. The problem in which we break the item is known as a Fractional knapsack problem.

This problem can be solved with the help of using two techniques:

- **Brute-force approach**: The brute-force approach tries all the possible solutions with all the different fractions but it is a time-consuming approach.
- **Greedy approach**: In Greedy approach, we calculate the **ratio of profit/weight**, and accordingly, we will select the item. The item with the highest ratio would be selected first.

There are basically <u>three approaches</u> to solve the problem:

- The first approach is to select the item based on the maximum profit.
- The second approach is to select the item based on the minimum weight.
- The third approach is to calculate the ratio of profit/weight.

Knapsack Problem

Given a set of items, each with a weight and a value, determine a subset of items to include in a collection so that the total weight is less than or equal to a given limit and the total value is as large as possible.

The knapsack problem is in combinatorial optimization problem. It appears as a sub-problem in many, more complex mathematical models of real-world problems. One general approach to difficult problems is to identify the most restrictive constraint, ignore the others, solve a knapsack problem, and somehow adjust the solution to satisfy the ignored constraints.

Problem Scenario

A thief is robbing a store and can carry a maximal weight of W into his knapsack. There are n items available in the store and weight of i^{th} item is w_i and its profit is p_i. What items should the thief take?

In this context, the items should be selected in such a way that the thief will carry those items for which he will gain maximum profit. Hence, the objective of the thief is <u>to maximize the profit.</u>

Fractional Knapsack

In this case, items can be broken into smaller pieces, hence the thief can select fractions of items.

According to the problem statement,

- There are **n** items in the store
- Weight of $\mathbf{i^{th}}$ item $\mathbf{w_i > 0}$
- Profit for $\mathbf{i^{th}}$ item $\mathbf{p_i > 0}$ and
- Capacity of the Knapsack is **W**

In this version of Knapsack problem, items can be broken into smaller pieces. So, the thief may take only a fraction x_i of $\mathbf{i^{th}}$ item.

$$0 \leqslant xi \leqslant 1$$

The $\mathbf{i^{th}}$ item contributes the weight $\mathbf{x_i.w_i}$ to the total weight in the knapsack and profit $\mathbf{x_i.p_i}$ to the total profit.

Hence, the objective of this algorithm is to

$$maximize \sum_{n=1}^{n} (x_i.pi)$$

subject to constraint,

$$\sum_{n=1}^{n} (x_i.wi) \leqslant W$$

It is clear that an optimal solution must fill the knapsack exactly, otherwise we could add a fraction of one of the remaining items and increase the overall profit.

Thus, an optimal solution can be obtained by

$$\sum_{n=1}^{n} (x_i.wi) = W$$

In this context, first we need to sort those items according to the value of $\frac{p_i}{w_i}$, so that $\frac{p_i+1}{w_i+1} \le \frac{p_i}{w_i}$. Here, ***x*** is an array to store the fraction of items.

```
Algorithm: Greedy-Fractional-Knapsack (w[1..n], p[1..n], W)
for i = 1 to n
   do x[i] = 0
weight = 0
for i = 1 to n
   if weight + w[i] ≤ W then
      x[i] = 1
      weight = weight + w[i]
   else
      x[i] = (W - weight) / w[i]
      weight = W
```

Analysis

If the provided items are already sorted into a decreasing order of piwi, then the while loop takes a time in ***O(n)***; Therefore, the total time including the sort is in ***O(n logn)***.

Example

Let us consider that the capacity of the knapsack ***W* = 60** and the list of provided items are shown in the following table −

Item	A	B	C	D
Profit	280	100	120	120
Weight	40	10	20	24
Ratio ($p_i w_i$)	7	10	6	5

As the provided items are not sorted based on $\mathbf{p_i w_i}$. After sorting, the items are as shown in the following table.

Item	B	A	C	D

Profit	100	280	120	120
Weight	10	40	20	24
Ratio ($p_i w_i$)	10	7	6	5

Solution

After sorting all the items according to $\mathbf{p_i w_i}$. First all of ***B*** is chosen as weight of ***B*** is less than the capacity of the knapsack. Next, item ***A*** is chosen, as the available capacity of the knapsack is greater than the weight of ***A***. Now, ***C*** is chosen as the next item. However, the whole item cannot be chosen as the remaining capacity of the knapsack is less than the weight of ***C***.

Hence, fraction of ***C*** (i.e. $(60 - 50)/20$) is chosen.

Now, the capacity of the Knapsack is equal to the selected items. Hence, no more items can be selected.

The total weight of the selected items is **10 + 40 + 20 * (10/20) = 60**

And the total profit is **100 + 280 + 120 * (10/20) = 380 + 60 = 440**

This is the optimal solution. We cannot gain more profit selecting any different combination of items.