Fractional Knapsack Problem

Using Greedy Method

Exam-Ready Notes Based on Jenny's Lectures

Problem Statement & Given Data

We are given a knapsack (bag) with a maximum capacity $W = 15$ kg, and 7 items, each with a specific profit and weight. The objective is to select items to maximize the total profit without exceeding the capacity.
Because this is a **fractional** knapsack problem, if the remaining capacity is less than an item's weight, we can break the item and take a fraction of it.

Items Data Matrix:

Item	Item 1	Item 2	Item 3	Item 4	Item 5	Item 6	Item 7
Profit (P)	5	10	15	7	8	9	8
Weight (W)	1	3	5	4	1	3	4

The Optimal Strategy: Profit-by-Weight Ratio

The Intuition (Why use a ratio?)
Imagine you are filling a small bag, and you have to choose between a heavy rock that sells for $10, or a tiny diamond that also sells for $10. Which one do you pick? The diamond, of course! Even though the total profit is the same, the diamond takes up much less space. It gives you a much higher **value per kilogram**.

In the Fractional Knapsack Problem, we cannot just look at the highest profit (because the item might be too heavy and fill the whole bag) or the lowest weight (because the item might be worthless). We need the best balance. We find this by calculating the **Profit-by-Weight Ratio** for every item.

Step 1: Calculate the Ratio

The formula is very simple:

$$\text{Ratio} = \frac{\text{Profit of the Item}}{\text{Weight of the Item}}$$

This ratio tells us exactly how much profit we earn for **every 1 kg** of that specific item. Let's calculate this for all our items:

Item	Profit (P)	Weight (W)	Calculation (P / W)	Ratio (Value per kg)
Item 1	5	1	$5/1$	**5.00**
Item 2	10	3	$10/3$	**3.33**
Item 3	15	5	$15/5$	**3.00**
Item 4	7	4	$7/4$	**1.75**
Item 5	8	1	$8/1$	**8.00** (Highest!)
Item 6	9	3	$9/3$	**3.00**
Item 7	8	4	$8/4$	**2.00**

Step 2: Sort in Descending Order

Why do we sort in descending order? We want to pack our bag with the most valuable items first before we run out of space. By picking the items with the highest "Value per kg" first, we guarantee the maximum possible profit.

Our Selection Order:
Item 5 (8.00) $\rightarrow$ Item 1 (5.00) $\rightarrow$ Item 2 (3.33) $\rightarrow$ Item 3 (3.00) $\rightarrow$ Item 6 (3.00) $\rightarrow$ Item 7 (2.00) $\rightarrow$ Item 4 (1.75)

Step 3: Step-by-Step Selection

Now, we will pick the items one by one according to our sorted list, keeping an eye on our remaining bag capacity (Starts at 15 kg).

Selection 1: Item 5
Item 5 gives the best value per kg. It weighs 1 kg. Our bag can hold 15 kg, so it easily fits. We take the whole item.

Item	Fraction Taken	Profit Added	Weight Used	Remaining Capacity
Item 5	1 (All)	8	1	$15 - 1 =$ **14**

Selection 2: Item 1
The next best is Item 1. It weighs 1 kg. We have 14 kg of space left, so it easily fits. We take the whole item.

Item	Fraction Taken	Profit Added	Weight Used	Remaining Capacity
Item 5	1 (All)	8	1	14
Item 1	1 (All)	5	1	$14 - 1 =$ **13**

Selection 3: Item 2
Next is Item 2, weighing 3 kg. We have 13 kg of space left, so it fits perfectly.

Item	Fraction Taken	Profit Added	Weight Used	Remaining Capacity
Item 5	1 (All)	8	1	14
Item 1	1 (All)	5	1	13
Item 2	1 (All)	10	3	$13 - 3 =$ **10**

Selection 4: Item 3
Next is Item 3, weighing 5 kg. We have 10 kg of space left, so we drop the whole item in.

Item	Fraction Taken	Profit Added	Weight Used	Remaining Capacity
Item 5	1 (All)	8	1	14
Item 1	1 (All)	5	1	13
Item 2	1 (All)	10	3	10
Item 3	1 (All)	15	5	$10 - 5 =$ **5**

Selection 5: Item 6
Next is Item 6, weighing 3 kg. We have 5 kg space remaining. It fits!

Item	Fraction Taken	Profit Added	Weight Used	Remaining Capacity
Item 5	1 (All)	8	1	14
Item 1	1 (All)	5	1	13
Item 2	1 (All)	10	3	10
Item 3	1 (All)	15	5	5
Item 6	1 (All)	9	3	$5 - 3 = \mathbf{2}$

Selection 6: Item 7
Note: Our remaining capacity is exactly 2 kg, but our next best item, Item 7, weighs 4 kg. Because it is too heavy to fit entirely, we must break it and take a fraction. We take exactly what we need: $\frac{2}{4}$ (or half) of the item.

Item	Fraction Taken	Profit Added	Weight Used	Remaining Capacity
Item 5	1 (All)	8	1	14
Item 1	1 (All)	5	1	13
Item 2	1 (All)	10	3	10
Item 3	1 (All)	15	5	5
Item 6	1 (All)	9	3	2
Item 7	$2/4$	$8 \times \frac{2}{4} = 4$	2	$2 - 2 = \mathbf{0}$

Since our remaining capacity has reached 0, our knapsack is completely full. We stop here and ignore the remaining Item 4.

Final Result

By summing up the Profit Added column, we get our maximum possible profit:
Total Optimal Profit = $8 + 5 + 10 + 15 + 9 + 4 = \mathbf{51}$

Summary: Why is this method Optimal?

The Greedy Approach using the **Profit/Weight Ratio** gives us the absolute highest profit (51) for the Fractional Knapsack Problem.
It works because by picking the items with the highest "value per unit of weight", we guarantee that every single inch of space inside our knapsack is filled with the most profitable material available. There is no wasted space and no wasted opportunity.

The Golden Rule of Knapsack Problems:

- **Fractional Knapsack** $\rightarrow$ Items *can* be broken into pieces. The **Greedy Method (Ratio)** works perfectly and gives the optimal answer.
- **0/1 Knapsack** $\rightarrow$ Items *cannot* be broken (you either take it completely or leave it). Here, the Greedy Method **fails**, and we must use a completely different technique called **Dynamic Programming**.

0/1 Knapsack Problem

Dynamic Programming Approach

Exam-Ready Notes Based on Jenny's Lectures

1. Introduction

The **Knapsack Problem** involves a container (knapsack) with a maximum weight capacity W, and a set of available items, each with a specific weight and profit. The objective is to select items to maximize the total profit such that the total weight does not exceed W.

There are two main variations:

- **0/1 Knapsack Problem:** An item must be picked completely or left behind (0 or 1). It cannot be divided. Solved using **Dynamic Programming**.
- **Fractional Knapsack Problem:** Items can be divided into smaller fractions. Solved using the **Greedy Method**.

2. Problem Statement & Given Example

Suppose we are given a knapsack capacity $W = 8$ and the following 4 items:

Item	1	2	3	4
Weight	3	4	6	5
Profit	2	3	1	4

Important Pre-Processing Step

Always **sort the items in ascending order of their weights** before building the matrix. Let's arrange them properly:

Item (Sorted)	**1**	**2**	**3**	**4**
Weight (W_i)	3	4	5	6
Profit (P_i)	2	3	4	1

Note on Brute Force: Testing all combinations yields $2^4 = 16$ combinations. To avoid exponential time complexity, we apply Dynamic Programming.

3. Dynamic Programming Formula

We solve subproblems step-by-step using a matrix M. The rows represent available items ($i = 0 \ldots 4$), and the columns represent incremental knapsack capacities ($w = 0 \ldots 8$).

DP Recurrence Relation

$$M[i,w] = \begin{cases} 0 & \text{if } i = 0 \text{ or } w = 0 \\ M[i-1,w] & \text{if } W_i > w \quad \text{(Item too heavy, copy from above)} \\ \max\Big(M[i-1,w], P_i + M[i-1,w-W_i]\Big) & \text{if } W_i \leq w \quad \text{(Pick max of excluding vs. including)} \end{cases}$$

4. Step-by-Step Matrix Construction

As explicitly requested, here is the visual progression of the matrix. A separate table is provided for **every new DP formula calculation** to show exactly how the values fill out left-to-right.

Initialization & Row 1 ($i = 1, W_1 = 3, P_1 = 2$)

Row 0 and Col 0 are initialized to 0. For Row 1, if $w < 3$, we cannot pick the item $\rightarrow$ copy 0. For $w \geq 3$, we pick the item and get profit 2.

i	W_i	P_i	$w=0$	$w=1$	$w=2$	$w=3$	$w=4$	$w=5$	$w=6$	$w=7$	$w=8$
0	0	0	0	0	0	0	0	0	0	0	0
1	3	2	0	0	0	2	2	2	2	2	2
2	4	3									
3	5	4									
4	6	1									

Filling Row 2 ($i = 2, W_2 = 4, P_2 = 3$)

For $w = 1, 2, 3$, the capacity is strictly less than 4, so we simply copy values from the row above $(0, 0, 2)$. We start calculating from $w = 4$.

Calculate for $i = 2, w = 4$:

$$M[2,4] = \max\{M[1,4], P_2 + M[1,4-4]\} = \max\{2, 3+0\} = 3$$

i	W_i	P_i	$w=0$	$w=1$	$w=2$	$w=3$	$w=4$	$w=5$	$w=6$	$w=7$	$w=8$
0	0	0	0	0	0	0	0	0	0	0	0
1	3	2	0	0	0	2	2	2	2	2	2
2	4	3	0	0	0	2	3				
3	5	4									
4	6	1									

Calculate for $i = 2, w = 5$:

$$M[2,5] = \max\{M[1,5], P_2 + M[1,5-4]\} = \max\{2, 3+0\} = 3$$

i	W_i	P_i	$w=0$	$w=1$	$w=2$	$w=3$	$w=4$	$w=5$	$w=6$	$w=7$	$w=8$
0	0	0	0	0	0	0	0	0	0	0	0
1	3	2	0	0	0	2	2	2	2	2	2
2	4	3	0	0	0	2	3	3			
3	5	4									
4	6	1									

Calculate for $i = 2, w = 6$:

$$M[2,6] = \max\{M[1,6], P_2 + M[1,6-4]\} = \max\{2, 3+0\} = 3$$

i	W_i	P_i	$w=0$	$w=1$	$w=2$	$w=3$	$w=4$	$w=5$	$w=6$	$w=7$	$w=8$
0	0	0	0	0	0	0	0	0	0	0	0
1	3	2	0	0	0	2	2	2	2	2	2
2	4	3	0	0	0	2	3	3	3		
3	5	4									
4	6	1									

Calculate for $i=2, w=7$:

$$M[2,7] = \max\{M[1,7], P_2 + M[1,7-4]\} = \max\{2, 3+2\} = 5$$

i	W_i	P_i	$w=0$	$w=1$	$w=2$	$w=3$	$w=4$	$w=5$	$w=6$	$w=7$	$w=8$
0	0	0	0	0	0	0	0	0	0	0	0
1	3	2	0	0	0	2	2	2	2	2	2
2	4	3	0	0	0	2	3	3	3	5	
3	5	4									
4	6	1									

Calculate for $i=2, w=8$:

$$M[2,8] = \max\{M[1,8], P_2 + M[1,8-4]\} = \max\{2, 3+2\} = 5$$

i	W_i	P_i	$w=0$	$w=1$	$w=2$	$w=3$	$w=4$	$w=5$	$w=6$	$w=7$	$w=8$
0	0	0	0	0	0	0	0	0	0	0	0
1	3	2	0	0	0	2	2	2	2	2	2
2	4	3	0	0	0	2	3	3	3	5	5
3	5	4									
4	6	1									

Filling Row 3 ($i = 3, W_3 = 5, P_3 = 4$)

For $w = 1 \dots 4$, copy values directly from above $(0, 0, 2, 3)$. Calculations begin at $w = 5$.

Calculate for $i = 3, w = 5$:

$$M[3,5] = \max\{M[2,5], P_3 + M[2,5-5]\} = \max\{3, 4+0\} = 4$$

i	W_i	P_i	$w=0$	$w=1$	$w=2$	$w=3$	$w=4$	$w=5$	$w=6$	$w=7$	$w=8$
0	0	0	0	0	0	0	0	0	0	0	0
1	3	2	0	0	0	2	2	2	2	2	2
2	4	3	0	0	0	2	3	3	3	5	5
3	5	4	0	0	0	2	3	4			
4	6	1									

Calculate for $i = 3, w = 6$:

$$M[3,6] = \max\{M[2,6], P_3 + M[2,6-5]\} = \max\{3, 4+0\} = 4$$

i	W_i	P_i	$w=0$	$w=1$	$w=2$	$w=3$	$w=4$	$w=5$	$w=6$	$w=7$	$w=8$
0	0	0	0	0	0	0	0	0	0	0	0
1	3	2	0	0	0	2	2	2	2	2	2
2	4	3	0	0	0	2	3	3	3	5	5
3	5	4	0	0	0	2	3	4	4		
4	6	1									

Calculate for $i = 3, w = 7$:

$$M[3,7] = \max\{M[2,7], P_3 + M[2,7-5]\} = \max\{5, 4+0\} = 5$$

i	W_i	P_i	$w=0$	$w=1$	$w=2$	$w=3$	$w=4$	$w=5$	$w=6$	$w=7$	$w=8$
0	0	0	0	0	0	0	0	0	0	0	0
1	3	2	0	0	0	2	2	2	2	2	2
2	4	3	0	0	0	2	3	3	3	5	5
3	5	4	0	0	0	2	3	4	4	5	
4	6	1									

Calculate for $i = 3, w = 8$:

$$M[3,8] = \max\{M[2,8], P_3 + M[2,8-5]\} = \max\{5, 4+2\} = 6$$

i	W_i	P_i	$w=0$	$w=1$	$w=2$	$w=3$	$w=4$	$w=5$	$w=6$	$w=7$	$w=8$
0	0	0	0	0	0	0	0	0	0	0	0
1	3	2	0	0	0	2	2	2	2	2	2
2	4	3	0	0	0	2	3	3	3	5	5
3	5	4	0	0	0	2	3	4	4	5	6
4	6	1									

Filling Row 4 ($i = 4, W_4 = 6, P_4 = 1$)

For $w = 1 \dots 5$, copy values directly from above $(0, 0, 2, 3, 4)$. Calculations begin at $w = 6$.

Calculate for $i = 4, w = 6$:

$$M[4,6] = \max\{M[3,6], P_4 + M[3,6-6]\} = \max\{4, 1+0\} = 4$$

i	W_i	P_i	$w=0$	$w=1$	$w=2$	$w=3$	$w=4$	$w=5$	$w=6$	$w=7$	$w=8$
0	0	0	0	0	0	0	0	0	0	0	0
1	3	2	0	0	0	2	2	2	2	2	2
2	4	3	0	0	0	2	3	3	3	5	5
3	5	4	0	0	0	2	3	4	4	5	6
4	6	1	0	0	0	2	3	4	4		

Calculate for $i = 4, w = 7$:

$$M[4,7] = \max\{M[3,7], P_4 + M[3,7-6]\} = \max\{5, 1+0\} = 5$$

i	W_i	P_i	$w=0$	$w=1$	$w=2$	$w=3$	$w=4$	$w=5$	$w=6$	$w=7$	$w=8$
0	0	0	0	0	0	0	0	0	0	0	0
1	3	2	0	0	0	2	2	2	2	2	2
2	4	3	0	0	0	2	3	3	3	5	5
3	5	4	0	0	0	2	3	4	4	5	6
4	6	1	0	0	0	2	3	4	4	5	

Calculate for $i = 4, w = 8$:

$$M[4,8] = \max\{M[3,8], P_4 + M[3,8-6]\} = \max\{6, 1+0\} = 6$$

i	W_i	P_i	$w=0$	$w=1$	$w=2$	$w=3$	$w=4$	$w=5$	$w=6$	$w=7$	$w=8$
0	0	0	0	0	0	0	0	0	0	0	0
1	3	2	0	0	0	2	2	2	2	2	2
2	4	3	0	0	0	2	3	3	3	5	5
3	5	4	0	0	0	2	3	4	4	5	6
4	6	1	0	0	0	2	3	4	4	5	6

5. Finding the Selected Items (Backtracking)

To trace back which items were actually selected, we start at the bottom-right corner of the matrix (**$M[4,8] = 6$**) and move upwards. **Rule:** If the value comes exactly from the row above, the item was **not** selected. If the value differs, the item **was** selected.

- **Start at** $M[4,8] = 6$. Check upper cell $M[3,8] = 6$. The value is the same.
  $\Longrightarrow$ **Item 4 is NOT selected**. Move pointer up to $M[3,8]$.

- **At** $M[3,8] = 6$. Check upper cell $M[2,8] = 5$. The value changed ($6 \neq 5$).
  $\Longrightarrow$ **Item 3 IS selected**. Remaining weight = $8 - W_3 = 8 - 5 = 3$.
  Move pointer up to row 2 and jump to the column for the remaining weight: $M[2,3]$.

- **At** $M[2,3] = 2$. Check upper cell $M[1,3] = 2$. The value is the same.
  $\implies$ **Item 2 is NOT selected**. Move pointer up to $M[1,3]$.

- **At** $M[1,3] = 2$. Check upper cell $M[0,3] = 0$. The value changed ($2 \neq 0$).
  $\implies$ **Item 1 IS selected**. Remaining weight = $3 - W_1 = 3 - 3 = 0$.
  Move pointer up to row 0, column 0 ($M[0,0]$). We are done!

Final Answer

Selected Items: Item 1 and Item 3.
Total Weight in Bag: $W_1 + W_3 = 3 + 5 =$ **8 kg** (Perfectly fills capacity)
Maximum Profit Generated: $P_1 + P_3 = 2 + 4 =$ **6**