Proyecto 2: El Problema de la Mochila

Emily Sanchez
Viviana Vargas

Curso: Investigación de Operaciones
II Semestre 2025

September 19, 2025

1 Problema de la Mochila (Knapsack Problem)

El **problema de la mochila** es un clasico de la *optimizacion combinatoria*. Se dispone de una **mochila** con una **capacidad maxima** W y un conjunto de n objetos. Cada objeto i tiene un **peso** w_i y un **valor** v_i. El objetivo es seleccionar los objetos de manera que:

- La suma total de los pesos no exceda la capacidad W.
- Se maximice el valor total de los objetos elegidos.

1.1 Variantes principales

0/1 Knapsack Cada objeto puede elegirse una sola vez o no elegirse: decision binaria.

Bounded Knapsack Cada objeto puede seleccionarse un numero limitado de veces.

Unbounded Knapsack Se permite una cantidad ilimitada de cada objeto.

1.2 Solucion

0/1 Knapsack Se resuelve comunmente con **programacion dinamica**. Sea $dp[i][w]$ el valor maximo al considerar los primeros i objetos y capacidad w.

$$dp[i][w] = \begin{cases} dp[i-1][w] & \text{si } w_i > w, \\ \max(dp[i-1][w], v_i + dp[i-1][w-w_i]) & \text{si } w_i \leq w. \end{cases}$$

Bounded Knapsack Similar al 0/1 pero puede tener uno o más cantidades por objeto. Es limitado, por lo que no puede ser infinito.

$$dp[i][w] = \max_{0 \leq k \leq c_i,\ k\, w_i \leq w} \left(dp[i-1][w - kw_i] + kv_i \right).$$

Unbounded Knapsack Similar al bounded pero permitiendo repeticiones sin limite de cantidades (infinito).

$$dp[w] = \max(dp[w], v_i + dp[w - w_i]).$$

Tipo de problema: Unbounded Knapsack
Capacidad máxima: 17
Número de objetos: 7

Datos del Problema

Objeto	Costo	Valor	Cantidad
A	12,00	16,00	∞
B	7,00	6,00	∞
C	8,00	10,00	∞
D	9,00	11,00	∞
E	15,00	9,00	∞
F	2,00	17,00	∞
G	3,00	2,00	∞

Tabla de Programación Dinámica Detallada

Capacidad	A	B	C	D	E	F	G
0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
2	0	0	0	0	0	17(1)	17
3	0	0	0	0	0	17(1)	17
4	0	0	0	0	0	34(2)	34
5	0	0	0	0	0	34(2)	34
6	0	0	0	0	0	51(3)	51
7	0	6(1)	6	6	6	51(3)	51
8	0	6(1)	10(1)	10	10	68(4)	68
9	0	6(1)	10(1)	11(1)	11	68(4)	68
10	0	6(1)	10(1)	11(1)	11	85(5)	85
11	0	6(1)	10(1)	11(1)	11	85(5)	85
12	16(1)	16	16	16	16	102(6)	102
13	16(1)	16	16	16	16	102(6)	102
14	16(1)	16	16	16	16	119(7)	119
15	16(1)	16	16	16	16	119(7)	119
16	16(1)	16	20(2)	20	20	136(8)	136
17	16(1)	16	20(2)	21(1)	21	136(8)	136

Solución Óptima

Valor máximo obtenido: 136
Objetos seleccionados: F:8
Capacidad utilizada: 16