

## 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.

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

**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:** Bounded Knapsack  
**Capacidad máxima:** 7  
**Número de objetos:** 2

## Datos del Problema

Objeto	Costo	Valor	Cantidad
A	2,00	5,00	1
B	3,00	8,00	1

## Tabla de Programación Dinámica Detallada

Capacidad	A	B
0	0	0
1	0	0
2	5(1)	5
3	5(1)	8(1)
4	5(1)	8(1)
5	5(1)	13(1)
6	5(1)	13(1)
7	5(1)	13(1)

## Solución Óptima

**Valor máximo obtenido:** 13  
**Objetos seleccionados:** B:1, A:1  
**Capacidad utilizada:** 5