Knapsack

Problema de la mochila

También conocido por *Knapsack* por su nombre en inglés consiste en que, dados unos recursos limitados R (entero no negativo) y un conjunto de N elementos cada uno con una ganancia g_i (no negativo) y un costo c_i (entero no negativo), ¿Cuál es el subconjunto de elementos que maximiza las ganancias sin superar el limitante de recursos?

Fuente: https://openclipart.org/detail/313728/thief

Ejemplo: para R = 6

g	12	18	30	44
c	1	2	3	4

Solución: {2,4}, con una ganancia de 62

Solución mediante programación dinámica

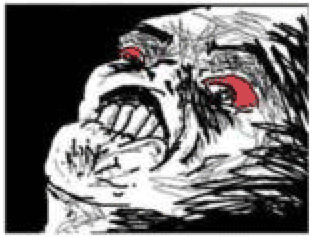
¿Qué forma debería tener la solución óptima de un subproblema?

Dados los primeros *i* elementos y una cantidad *j* de recursos disponibles la solución óptima $A_{i,j}$ para ese subproblema consistiría en elegir una de las dos siguientes opciones:

1. Incluir el elemento *i*, sumar g_i a la máxima ganancia obtenida con los *i-1* elementos restantes y disminuir *j* en c_i

2. No incluir el elemento *i* y la máxima ganancia sería la obtenida con los i-1 elementos restantes

Si las conociéramos …



Solución mediante programación dinámica

```
for j = 0 to R:
    A0,j = 0
for i = 1 to N:
    for j = 0 to R:
        if ci <= j:
            k = j - ci
            Ai,j = MAX(gi + Ai-1,k, Ai-1,j)
        else:
            Ai,j = Ai-1,j
print(AN,R)
```

¿Cuál es la eficiencia de este algoritmo?

O($\boldsymbol{NR}$), mucho mejor que O($\boldsymbol{N2^N}$)

Ejemplo: para $R = 6$

i	1	2	3	4
g	12	18	30	44
c	1	2	3	4

j Recursos que tengo disponibles

Elementos que puedo escoger i

	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	12	12	12	12	12	12
2	0	12	18	30	30	30	30
3	0	12	18	30	42	48	60
4	0	12	18	30	44	56	**62**