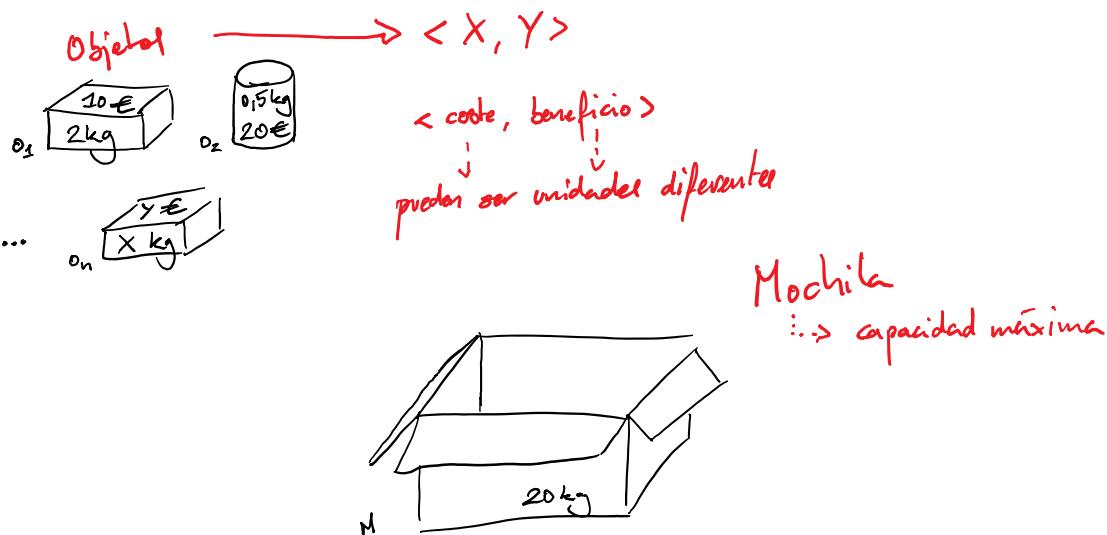


El problema de la mochila

martes, 3 de diciembre de 2019 23:26



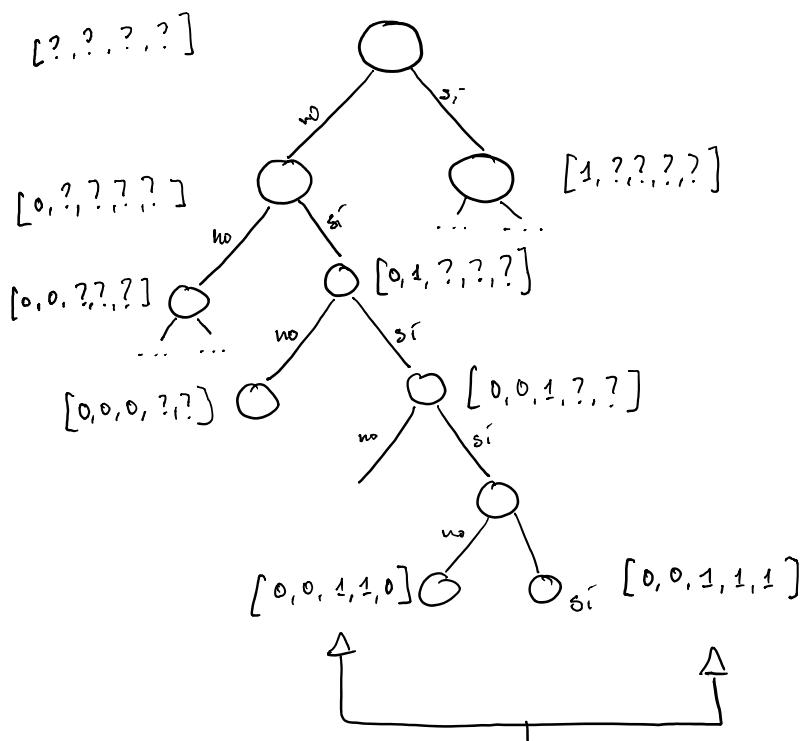
¿Qué subconjunto de objetos maximiza $\sum_i y_i$?

① Decidir la representación de la solución S

$$S = \{0, 1\}^n \equiv [0, 1, 0, 1 \dots \{0, 1\}^n] \equiv [\text{true}, \text{false}, \text{true} \dots]$$

② Generar posibles combinaciones

Vamos a representarlo con un árbol, donde cada nivel hará referencia a un objeto de los disponibles y cada hijo resultará de considerar (o no) dicho objeto.



nodos hoja
≡ candidatos a
solucionar

③ Resolviendo el árbol

si S_c es variable: → función que comprueba si $\sum_i x_i \cdot S_{ci} \leq M$
return $\max\{S_c^{HI}, S_c^{HO}\}$

si no
return $\boxed{\quad}$ → lo que pongáis aquí será una
solución inválida para parar
la ejecución de ramas inválidas
recursividad

Este paso ya no devuelve una solución óptima (si el que
la hay) al problema de la mochila

BACKTRACKING

El principal problema de esta solución es que su coste
computacional es, al menos, $O(2^n)$. En espacio se puede
llegar a expandir a $O(n^2)$.

— Una vez más, el tamaño sí importa —

Alternativamente... —

Se puede recurrir a la programación dinámica para resolver
el problema de la mochila (discreta).

Esto se entiende mejor directamente con un ejemplo:

Objetos: $<2, 4>$, $<3, 1>$, $<2, 5>$, $<1, 3>$, $<6, 12>$

Mochila: 9

- En cada etapa, se considera el objeto i y todos los anteriores
- Cada vez que elegimos un objeto, el espacio restante en la mochila
es un problema diferente:

Ej.: elegimos meter el objeto 2 \Rightarrow el resto de la solución equivale a resolver el problema de la mochila donde:
 Objetos: ①, ③, ④, ⑤
 Mochila: 17

Reg.: Mochila DISCRETA

Lo he agarré el troqui

② Objetos indivisibles \equiv no puedo coger medio objeto, p. ej.

③ El coste/peso de los objetos se move en un dominio DISCRETO

		0	1	2	3	4	5	6	7	8	9
Coste	Beneficio										
① <Ningún objeto>											
② 2 4											
③ 3 1											
④ 2 5											
⑤ 1 3											
⑥ 6 12											

Vamos al tío:

y esto son todos los posibles subproblemas

aquí la capacidad máxima de la mochila



en cada celda vamos a meter el beneficio de la mejor opción entre:

- La mejor solución hasta ahora
- La solución que se genera considerando el objeto de la etapa actual

aquí nos pondrá la lista de objetos.

siempre se considera el actual y todos los anteriores. La primera fila es el caso "no hay objetos".

Ahora, a rellenar:

		Capacidad									
Coste	Beneficio	0	1	2	3	4	5	6	7	8	9
① <Ningún objeto>		0	0	0	0	0	0	0	0	0	0
② 2 4		0	0	4	4	4	4	4	4	4	4
③ 3 1											
④ 2 5											
⑤ 1 3											
⑥ 6 12											

sólo hay un objeto disponible y pesa 2 unidades. No podemos meterlo en mochila de capacidad 0 ni 1

Una mochila de capacidad 2 sí puede albergar el objeto ② y nos da un beneficio de:

$$\max \{ 0, 4 \} = 4$$

Si no hay objetos, no hay beneficio

$$\max \{ 0, 4 \} = 4$$

el máximo hasta ahora
(es decir, si no consideramos el objeto actual)

el máximo si sí lo consideramos

el valor de la etapa anterior

y así con todo :)

		Capacidad									
		0	1	2	3	4	5	6	7	8	9
L	B	0	0	0	0	0	0	0	0	0	0
①	<Ningún objeto>	0	0	0	0	0	0	0	0	0	0
②	2 4	0	0	4	4	4	4	4	4	4	4
③	3 1	0	0	4	4	4	5	5	5	5	5
④	2 5	0	0	5	5	9	9	9	10	10	10
⑤	1 3	0	3	5	8	9	12	12	12	13	13
⑥	6 12	0	3	5	8	9	12	12	15	17	20

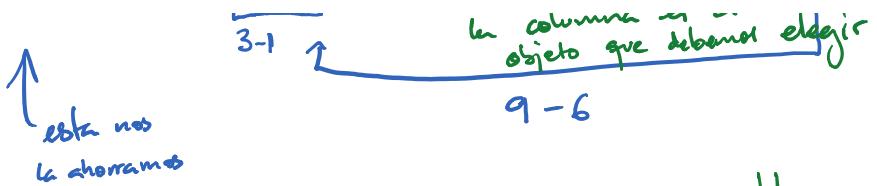
Una vez la tenemos, reconstruimos la solución:

		Capacidad									
		0	1	2	3	4	5	6	7	8	9
L	B	0	0	0	0	0	0	0	0	0	0
①	<Ningún objeto>	0	0	0	0	0	0	0	0	0	0
②	2 4	0	0	4	4	4	4	4	4	4	4
③	3 1	0	0	4	4	4	5	5	5	5	5
④	2 5	0	0	5	5	9	9	9	10	10	10
⑤	1 3	0	3	5	8	9	12	12	12	13	13
⑥	6 12	0	3	5	8	9	12	12	15	17	20

(espero que no haya errores :P)

2-2 3-1 a-r

el primer máximo de la columna es el objeto que debemos elegir



Solución: $< 0, 0, 1, 1, 1 >$

$\frac{11}{\square}$

$\leq O(n^2)$ para simplificar

Coste en espacio y tiempo $O(nm)$ siendo
n el numero de objetos y m el tamaño de la mochila

¡Dudas? francisco@decsai.ugr.es

Muaks