0-1 ZAINO — KNAPSACK

Titolo nota 17/03/2016

Ladro: peso max W

n oggetti da rubare

valore $v_1, v_2, \dots, v_n$

peso $p_1, p_2, \dots, p_n$

vuole trovare una soluzione S = (sottoinsieme degli n oggetti)

$$\max_{i \in S} \sum v_i \quad \text{subject to} \quad \sum_{i \in S} p_i \leq W$$

?

$W = 8$

	PESO	VAL
1	5	10
2	4	6
3	4	5

Greedy (goloso) = selezione l'ottimo locale

- ordinare in ordine non crescente del valore
- selezionare in questo ordine finché entrano nello zaino

10, 6, 5

$S = \{x_1\}$ $V = 10$ $P = 5$

ottimo ? NO

$S = \{x_2, x_3\}$ $V = 11$ $P = 8 \leq W$

Titolo nota 17/03/2016

Titolo nota 17/03/2016

Greedy 2 : • Ordinare gli elementi secondo valore/peso
• selezionare in questo ordine

	p	v	v/p
1	5	10	2
2	4	6	6/4 = 1.5
3	4	5	1.25

$S = \{x_1\}$ $V = 10$ $P = 5$ 3

$S' = \{x_1, x_3\}$ $V = 11$ $P = 8$

non è ottima

Dà la soluzione ottima per i problemi di ZAINO frazionabile

0 1 - ZAINO

"difficile"

Titolo nota 17/03/2016

"difficile" = non si sa come risolverlo se non "forza bruta"

brute force

facendo tutte le prove.

$\{x_1, \ldots, x_n\}$

1) Calcoliamo tutti i sottoinsiemi degli n oggetti
2) Per ogni sottoinsieme calcoliamo Valore e Peso associati agli elementi selezionati e ci si ricorda la sol. ottima calcolata fino a quel momento.

n oggetti

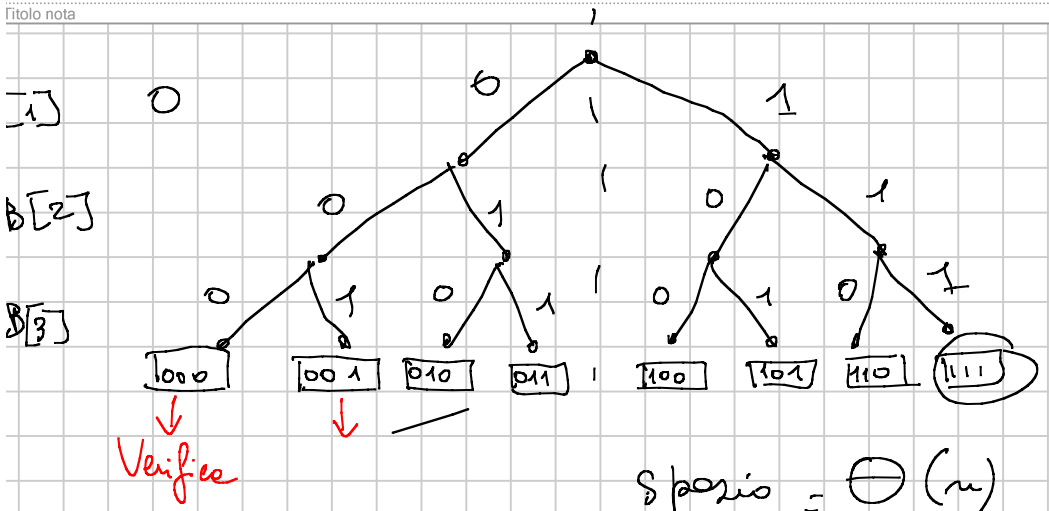
$\{x_1 \dots x_n\}$ $\quad n = 8$ vettore di appartenenza n bit

$\{x_2, x_5, x_6, x_8\} = 01001101$

$\{x_1, x_3\} = 10100000$

- Generare tutti i sottoinsiemi = stringhe binarie di n bit

$n \Rightarrow 2^n$ complessità esponenziale



si alloca un array B di n elementi

2^3 sottoinsiemi K il numero di bit già calcolati

```
Genera Binarie (B, k)
if (k == n) Elabora(B);
else {
    B[k+1] = 0;
    GeneraBinarie(B, k+1);
    B[k+1] = 1;
    Genera Binarie( B, k+1);
}
```

spazio = $\Theta(n)$

tempo = $\Theta(2^n)$

1° chiamata

GeneraBinarie(B, 0)

$$T(n) = 2\,T(n+1) + c$$

```
Elabora(B)
peso = 0 ; valore = 0;
for i = 1 to n {
    peso = peso + B[i] . p(i);
    valore = valore + B[i] . vi;
} if (peso ≤ W && valore > Vmax){
    Vmax = valore;
    for j = 1 to n Sol[j] = B[j];
}
```

B

x_1	x_2	x_3
1	0	1
p_1	p_2	p_3

V_{max} : valore della soluzione ottima corrente

Sol : elementi selezionati della sol. ottima corrente

$\Theta(n)$

01-ZAINO (v, p, W);

$V_{max} = 0$;

Sol = nuovo array di dimi n;

B = " " " " ";

Genera Binarie (B, 0);

print V_{max};

print SOL;

v array di n valori

p array di n pesi

W peso max dello zaino

$\Theta(2^n * n) = \Theta(2^n)$

Zaino = { non si conosce un algoritmo efficiente
non si sa se questo alg. possa esistere }

$n! \cong O(n^n)$

soluzioni approssimate
soluzioni probabilistiche
} problemi "difficili"

↓

passi casuali

Programmazione Dinamica per lo Zaino ??

ZAINO 01 per Progr. Dinamica

Titolo nota 17/03/2016

- La soluzione generale si trova dalla soluzione dei sottoproblemi.
- Caratterizzazione del problema generale e dei sottoproblemi

$Z(i,j)$ = il sottoproblema dello zaino che considera $0 \le i \le n$ elementi dello zaino con soluzione a valore massimo e peso $\le j$ $\quad 0 \le j \le W$

Regola Ricorsiva

n oggetti W

v array di n valori

p array di n pesi

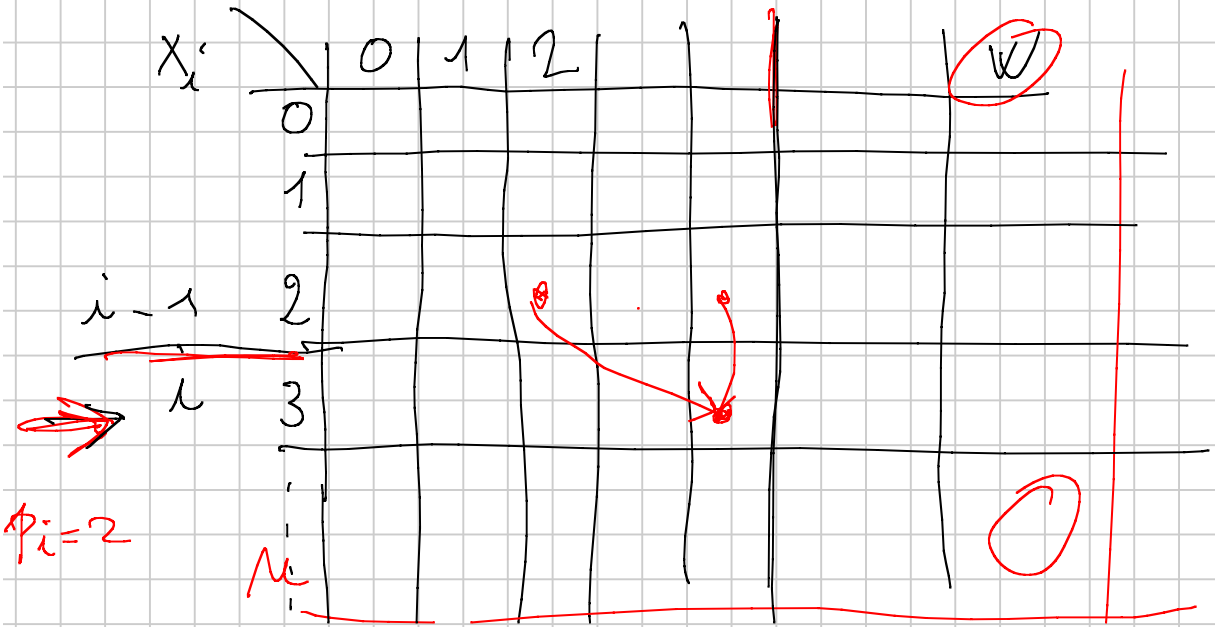
$$Z(0,j) = \emptyset$$

$$Z(i,0) = \emptyset$$

$Z(i,j)$

$Z(i,j) = \max$ { metto x_i nella soluz. $Z(i-1, j-p_i) + v_i$ (se $j \geq p_i$) ; non metto x_i nella sol. $Z(i-1, j)$ }

Soluzione finale in $Z(n, W)$



3 oggetti

	ort 1	ort 2	ort 3
v	60 €	100 €	120 €
p	1 Kg	2 Kg	3 Kg

$W = 5$

Conflenta ?

Spazio $\Theta(nW)$

tempo $\Theta(nW)$??

$Z(n, W) = 220$

$S = \{ort\,2, ort\,3\}$

Ricostruire la soluzione

	0	1	2	3	4	5
0	0	0	0	0	0	0
1	0	60	60	60	60	60
2	0	60	100	160	160	160
3	0	60	100	160	180	220

$Z(2,2) = \max \begin{cases} 60 \\ Z(i-1, j-p_2) + v_2 \end{cases}$

$Z(2,3) = \max \begin{cases} 60 \\ Z(i-1, 1) + v_2 = 160 \end{cases}$

ZAINO

Titolo nota 17/03/2016

1) Algoritmo Brute force

$\Theta(2^n \cdot n)$

3 oggetti e $W = 10'000$

1) $2^3 \cdot 3 = 24$
2) $3 \cdot 10'000 = 30'000$

2) Alg. Progr. Dinamica

$\Theta(n \cdot W)$

pseudo-polinomiale

10 oggetti e $W = 10$

1) $2^{10} \cdot 10 =$ ENORME
2) $10 \cdot 0$ OK

$\Theta(n \cdot W)$ complessità

dimensione dei dati: n, $\log W$

n, il numero di cifre (bit) per rappresentare W

$$\log_{10} W$$

2^n W

$\log W \to W$

$i \to 10^i$

W cresce esponenzialmente rispetto a $\log W$