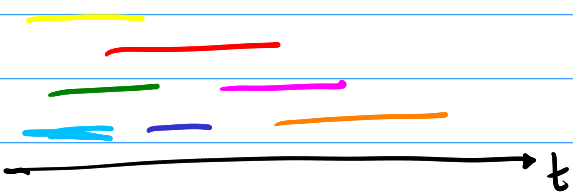


ESERCIZI SU PROGRAMMAZIONE DINAMICA (presi anche da vecchi esami)

SELEZIONE DI ATTIVITÀ PESATE

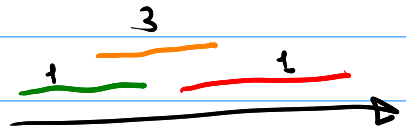


- Abbiamo una lista di attività $\{(a_i, b_i)\}_i$
 - a_i = inizio attività i
 - b_i = fine attività i
- E per ogni attività si ha un punteggio v_i

Lo scopo è quello di selezionare un insieme di attività

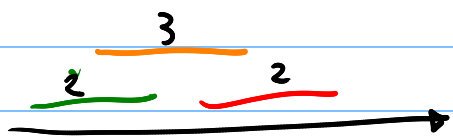
- senza sovrapposizioni
- di punteggio totale massimo

Es



- soluzione greedy vista a lezione = 2
- soluzione ottima = 3

Es



- soluzione greedy "prendi il valore massimo" = 3
- soluzione ottima = 4

ESERCIZIO

Trovare un algoritmo di complessità $O(n)$

dove n = # attività, e **MASSIMIZZARE** il valore complessivo

PROBLEMA DELLA BISACCIA (Knapsack)

②

- Abbiamo una lista di n oggetti, ognuno con volume C_i e valore V_i
- Abbiamo una bisaccia con capienza C

ESERCIZIO

Trovate un algoritmo che in tempo $\Theta(nC)$ trovi una lista di oggetti il cui volume totale sia

- $\leq C$
- la somma dei valori degli oggetti sia massima
- ognuno degli n oggetti può essere incluso al massimo una volta

Es $C=20$ pesi = $[10, 5, 8]$ valori = $[3, 2, 3]$

l'ottimo è 6

Es $C=100$ pesi = $[82, 15, 40, 95, 31, 50, 40, 28]$ valori = pesi

l'ottimo è 99 = $40 + 31 + 28$ (elementi 2, 4, 7)

CICLI NEGATIVI

ESERCIZIO

Dato un grafo orientato e pesato $G = (V, E)$ con $|V|=n$ $|E|=m$

descrivete un algoritmo che in tempo $\Theta(n^2 + mn)$ scopra

se in G esiste un ciclo negativo

Indizio Ricordate che i dati in input possono essere modificati dall'algoritmo.

Conteggi

3

Quando cerchiamo di identificare la sottostruttura ottima in un problema di ottimizzazione, tipicamente otteniamo una soluzione

S = soluzioni ottime S_1, S_2, \dots, S_k = soluzioni ottime di sottoproblemi

dove $S = \cup S_i$

e in particolare una soluzione ottima può essere contenuta in più di un S_i

Se invece vogliamo **CONTARE** oggetti

possiamo scomporre il conteggio C nei sottocounteggi

C_1, C_2, C_3, \dots

ma se vogliamo $C = C_1 + C_2 + C_3 + \dots$

è importante che gli oggetti contati nei vari C_i siano disgiunti.

STRINGHE BINARIE

Contare in $O(n)$ tutte le stringhe in $\{0,1\}^n, \dots$

— Esercizio: che non contengano 2 zeri adiacenti

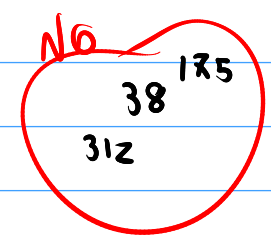
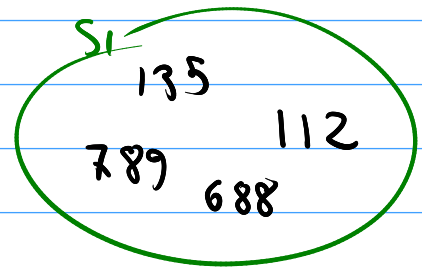
$n=4$ $sol=8$ 0101, 0110, 0111, 1010, 1011, 1101, 1110, 1111

— Esercizio: che non contengano 3 uni consecutivi

$n=4$ $sol=13$ 0000, 0001, 0010, 0011, 0100, 0101, 0110, 1000, 1001, 1010, 1011, 1100, 1101.

Esercizio: Contare in $O(n)$ i numeri decimali di esattamente n cifre, dove le cifre sono non decrescenti.

• Per esempio $n=3$



Esercizio

Contate in $O(n)$ i modi di mettere n persone in stanze da 1 o da 2

(le stanze non sono identificate)

- Es $n=4$ (1, 2, 3, 4) (12, 3, 4) (13, 2, 4) (1, 23, 4)
- Sol = 10 (14, 2, 3) (14, 23) (24, 1, 3) (24, 13) (34, 1, 2) (34, 12)

Esercizio

Contate in $O(n)$ i modi di mettere n persone in stanze da 1, 2 o 3 persone.

Esercizio

Contate in $O(k \cdot n)$ i modi di dividere n elementi in esattamente k insiemi

- $n=4$ $k=2$ {1, 2, 3} {4} | {1} {2, 3, 4} | {1, 4} {2, 3} |
- Sol = 7 {2} {1, 3, 4} | {2, 4} {1, 3} | {1, 2} {3, 4} |
- {1, 2, 4} {3}

5

Esercizio

Contate il numero di sequenze di n cifre decimali che non contengono due cifre pari adiacenti.

L'algoritmo deve avere tempo $O(n)$

Sottovettore di SOMMA MASSIMA

Esercizio

Trovate un algoritmo di complessità $O(n)$ che dato un array A di n interi,

trovi l'intervallo i, j con $0 \leq i < j \leq n$ tale che

$$\sum_{t=i}^{j-1} A[t] \text{ sia massima}$$

Osservate che noi abbiamo visto una soluzione di tipo divide et impera. L'esercizio chiede un approccio in programmazione dinamica