

ASD Laboratorio 02

The A(SD)-Team

UniTN

2026-10-22

ESERCITATORI

- **Quintino Francesco Lotito** (`quintino.lotito@unitn.it`)
- **Cristian Consonni** (`cristian.consonni@unitn.it`)

TUTOR

- **Daniele Dalla Vecchia**
(`daniele.dallavecchia@studenti.unitn.it`)
- **Alessandro Duranti**
(`alessandro.duranti@studenti.unitn.it`)

CALENDARIO

24/09	Introduzione
22/10	Ad-hoc
29/10	Grafi 1
19/11	Grafi 2
26/11	Grafi 3
03/12	Lab Progetto 1

CALCOLO COMBINATORIO (I)

Quante sono le **COPPIE** senza ripetizioni e senza tenere conto dell'ordine da un insieme di n elementi?

NUMERO DI COMBINAZIONI DI **2** ELEMENTI DATI n SENZA RIPETIZIONI

$$C(n, 2) = \binom{n}{2} = \frac{n \cdot (n - 1)}{2}$$

Caso particolare del **COEFFICIENTE BINOMIALE**:

$$C(n, k) = \binom{n}{k} = \frac{n!}{k!(n - k)!}$$

Intuizione (numero di disposizioni di k elementi da n , diviso per il numero di permutazioni di k elementi):

$$C(n, k) = \frac{D(n, k)}{P(k)} = \frac{n!}{(n - k)!} \cdot \frac{1}{k!}$$

CALCOLO COMBINATORIO (II)

Considerando le ripetizioni:

NUMERO DI COMBINAZIONI DI 2 ELEMENTI DATI n CON RIPETIZIONE

$$C'(n, 2) = \binom{n+1}{2} = \frac{n \cdot (n+1)}{2}$$

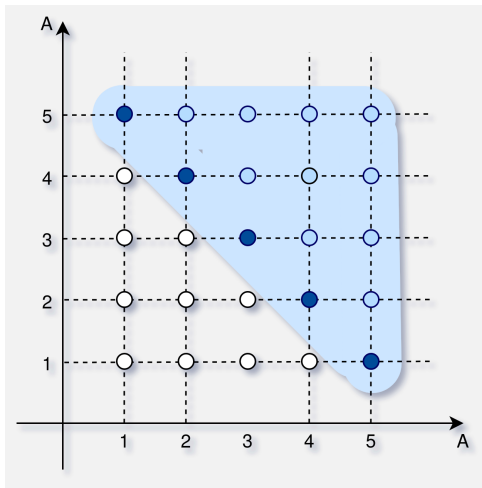
In questo caso:

$$C'(n, k) = \binom{n+k-1}{k} = \frac{(n+k-1)!}{(n-1)!k!}$$

Intuizione:

- corrispondenza biunivoca tra combinazione con ripetizioni e combinazioni senza ripetizione k elementi da $n + k - 1$

CALCOLO COMBINATORIO (III)



$$C = \{(i, j) \mid i, j \in A \subseteq N, j \geq i\}$$

SOTTOSEQ: SOLUZIONE $\mathcal{O}(N^2)$ (I)

SOLUZIONE $\mathcal{O}(N^2)$

Costruiamo array delle somme:

$$S_i = \sum_{j=1}^i A_j$$

Per ogni sottosequenza, calcoliamo la somma in $\mathcal{O}(1)$:

$$\text{Somma da } i \text{ a } j = S_j - S_{i-1}$$

$$S_i = \sum_{j=1}^i A_j$$

Esempio:

5
2
-3
4
1
5

Array delle somme:

index	1	2	3	4	5
array	2	-3	4	1	5
S	2	-1	3	4	9

Combinazioni (i, j) :

i/j	1	2	3	4	5
1	2	-1	3	4	9
2	—	-3	1	2	7
3	—	—	4	5	10
4	—	—	—	1	6
5	—	—	—	—	5

La sottosequenza di somma massima conterrà un elemento con indice massimo, sia esso i :

- B_i la sottosequenza di somma massima che ha come ultimo elemento il numero in posizione i ;
- assumendo di conoscere B_{i-1} , procedendo per induzione allora:
 $B_i = \max(A_i, B_{i-1} + A_i)$;
- terminiamo restituendo il valore massimo individuato durante l'induzione: $\max(B_0, B_1, \dots, B_{N-1})$.

ALGORITMO DI KADANE, $\mathcal{O}(N)$

```
int last = 0, mx = 0;
for(int i=0; i<N; i++) {
    in >> cur;
    last = max(cur, cur+last);
    mx = max(mx, last);
}
out << mx << endl;
```

SOTTOSEQ: SOLUZIONE $\mathcal{O}(N)$ (II)

Esempio:

5
2
-3
4
1
5

index	1	2	3	4	5
array	2	-3	4	1	5
last	2	-1	4	5	10
mx	2	2	4	5	10

Soluzione “a forza bruta” $\mathcal{O}((RC)^3) \sim \mathcal{O}(N^6)$:

- ❶ Ci sono $(RC)^2 \sim N^4$ sottomatrici¹
- ❷ È possibile calcolare la somma di una sottomatrice in meno di $\mathcal{O}(RC) \sim \mathcal{O}(N^2)$?
- ❸ Dobbiamo veramente guardare tutte le sottomatrici?

¹([link di approfondimento](#))

SOTTOMAT: MATRICE DELLE SOMME (I)

CALCOLARE LA SOMMA IN $O(1)$

Stessa idea di prima. Riempiamo un array somma ($O(RC)$)

$$S[i, j] = \sum_{a=1}^i \sum_{b=1}^j A[a, b]$$

Per calcolare la somma da $[r_1, c_1]$ a $[r_2, c_2]$:

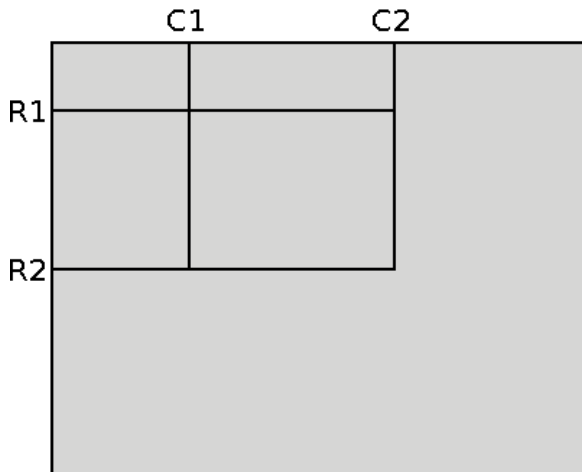
$$S[r_2, c_2] + S[r_1, c_1] - S[r_2, c_1] - S[r_1, c_2]$$

Sfruttando questa idea otteniamo un algoritmo $O((RC)^2)$.

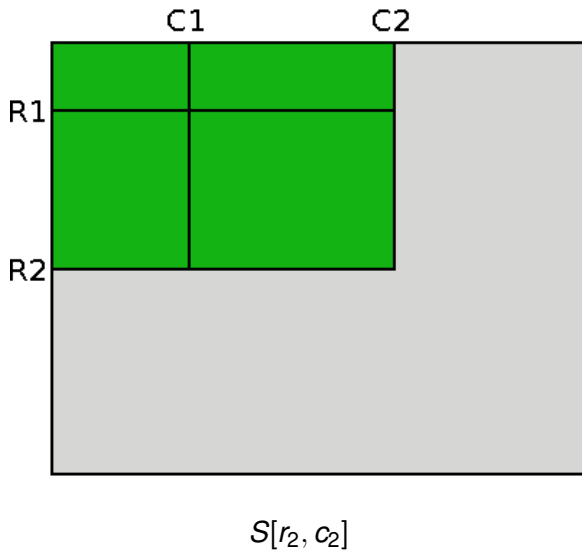
NOTA IMPLEMENTATIVA

Creando $S[i, j]$ con un "orlo" di zeri si semplifica la gestione degli indici.

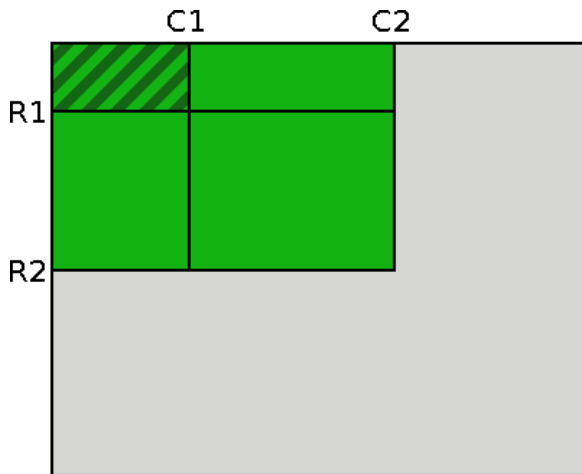
SOTTOMAT: MATRICE DELLE SOMME (II)



SOTTOMAT: MATRICE DELLE SOMME (III)

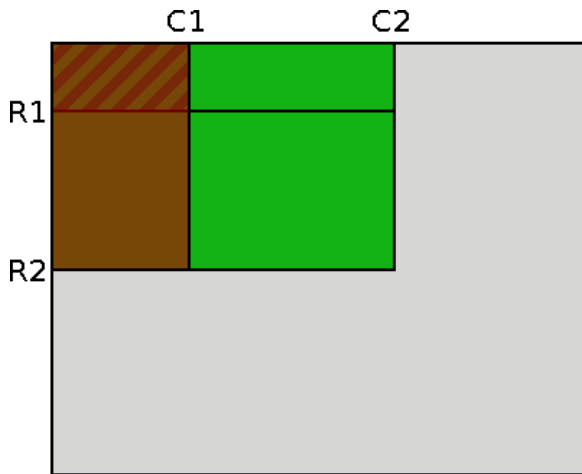


SOTTOMAT: MATRICE DELLE SOMME (IV)



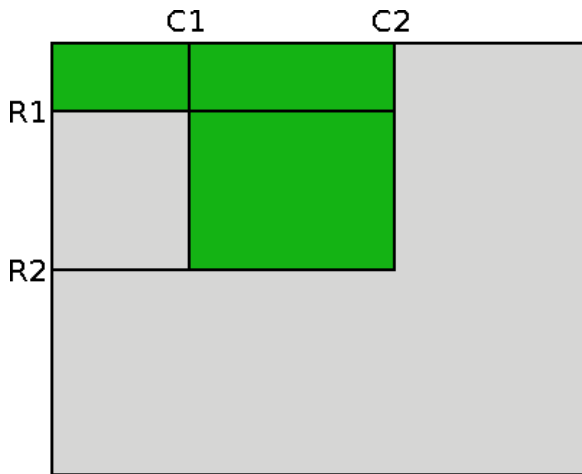
$$S[r_2, c_2] + S[r_1, c_1]$$

SOTTOMAT: MATRICE DELLE SOMME (V)



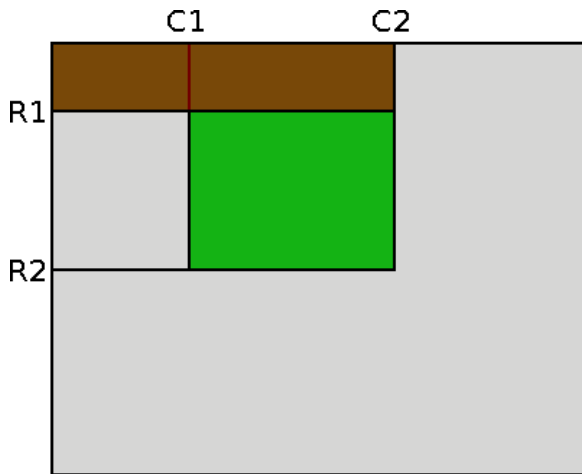
$$S[r_2, c_2] + S[r_1, c_1] - S[r_2, c_1]$$

SOTTOMAT: MATRICE DELLE SOMME (VI)



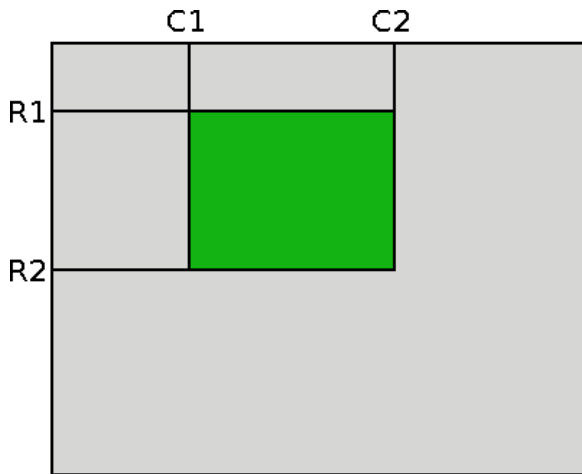
$$S[r_2, c_2] + S[r_1, c_1] - S[r_2, c_1]$$

SOTTOMAT: MATRICE DELLE SOMME (VII)



$$S[r_2, c_2] + S[r_1, c_1] - S[r_2, c_1] - S[r_1, c_2]$$

SOTTOMAT: MATRICE DELLE SOMME (VIII)



$$S[r_2, c_2] + S[r_1, c_1] - S[r_2, c_1] - S[r_1, c_2]$$

Potete definire la matrice somma $S[i, j]$ nel modo seguente:

```
for(int i=0; i<R; i++){  
    for(int j=0; j<C; j++){  
        in>>A[i][j];  
        if(i==0){  
            if(j==0) {  
                S[i][j]=A[i][j];  
            }  
            ...  
            S[i][j]=S[i][j-1] + \  
                    S[i-1][j] - \  
                    S[i-1][j-1] + \  
                    A[i][j];  
            ...  
        }  
    }  
}
```

ma esiste un modo più furbo che vi semplifica la vita.

SOLUZIONE $\mathcal{O}((RC)^2) \sim \mathcal{O}(N^4)$

- per ogni coppia di righe $r_s, r_e \rightarrow \mathcal{O}(R^2)$
- per ogni coppia di colonne $c_s, c_e \rightarrow \mathcal{O}(C^2)$

$\Rightarrow$ calcoliamo la somma $\rightarrow \mathcal{O}(1)$:

$$S[r_s, c_s] + S[r_e, c_e] - S[r_e, c_s] - S[r_s, c_e]$$

- possiamo sfruttare la soluzione ottima $\mathcal{O}(N)$ del problema della sottosequenza di somma massima per trovare la sottomatrice di somma massima?
- consideriamo tutte le sottomatrici che partono dalla colonna^(*) C_1 e arrivano alla colonna C_2 , possiamo applicare la sottosequenza di somma massima?
 - ▶ se $C_1 = C_2$, stiamo considerando una singola colonna, possiamo applicare facilmente la sottosequenza di somma massima
 - ▶ negli altri casi?

(*) il discorso è speculare per righe e colonne

SOTTOMAT: SOLUZIONE $\mathcal{O}(N^3)$, ESEMPIO (I)

Per ogni coppia C_1, C_2 creiamo un'istanza del problema della sottosequenza di somma massima.

$\Rightarrow$ se $C_1 = C_2$:

$$C_1 = C_2 = 2$$

/	1	2	3	4		$(r_1, c_1), (r_2, c_2)$		
1	2	-9	2	3	$\rightsquigarrow$	-9	$(1, 2), (1, 2)$	
2	1	4	5	1		4	$(1, 2), (2, 2)$	$(2, 2), (2, 2)$
3	-2	3	4	1		3	$(1, 2), (3, 2)$	$(2, 2), (3, 2)$ $(3, 2), (3, 2)$

Con Kadane riusciamo a considerare tutte e 6 le possibili sottomatrici.

SOTTOMAT: SOLUZIONE $\mathcal{O}(N^3)$, ESEMPIO (II)

Per ogni coppia C_1, C_2 creiamo un'istanza del problema della sottosequenza di somma massima.

$\Rightarrow$ se $C_1 \neq C_2$:

$$C_1 = 2; C_2 = 4$$

/	1	2	3	4		$(r_1, c_1), (r_2, c_2)$		
1	2	-9	2	3	$\rightsquigarrow$	-4	(1, 2), (1, 4)	
2	1	4	5	1		10	(1, 2), (2, 4)	(2, 2), (2, 4)
3	-2	3	4	1		8	(1, 2), (3, 4)	(2, 2), (3, 4) (3, 2), (3, 4)

Con Kadane troviamo che la sottosequenza massima è 18 ($10 + 8$) e corrisponde alla sottomatrice (2, 2), (3, 4)

SOLUZIONE $\mathcal{O}(N^3)$

Per ogni coppia di colonne C_1, C_2 :

- 1 Costruiamo l'array $S[1..R]$, di dimensione pari al numero di righe R ;
- 2 Inseriamo in $S[i]$ *“la somma degli elementi appartenenti alla riga i e compresi fra le colonne C_1, C_2 ”*. In formula:
$$S[i] = \sum_{j=C_1}^{C_2} A[i][j];$$
- 3 Usiamo l'algoritmo lineare per la sottosequenza di somma massima su S .

$\Rightarrow \mathcal{O}(RC^2)$, oppure $\mathcal{O}(R^2C)$

I sorgenti sono disponibili su <https://asdlab.disi.unitn.it/>

PROBLEMI

Testi completi su <https://asdlab.disi.unitn.it/>.

`SORTING SORT`

Implementate un algoritmo di ordinamento $\mathcal{O}(N \log N)$.

`INTERVALLI INTERVALLI`

Dato un insieme di intervalli temporali, scoprire il periodo più lungo non coperto da alcun intervallo.

`FUNZIONI CEIL (CEIL)`

Dato un vettore V di interi distinti e un vettore A di interi, **ordinato in modo crescente**, nei quali possono comparire valori ripetuti. Scrivere un algoritmo che, per ogni $v \in V$ restituisca il più piccolo elemento di A strettamente maggiore di v , oppure -1 se tale elemento non esiste.

ORDINAMENTO PESATO SORT PESATO

Avete un array di N interi, con i numeri da 1 a N (in ordine sparso). Ad ogni turno potete scambiare le posizioni di due interi, pagando la loro somma. Qual è il numero minimo di turni per ordinare l'array? Quant'è il prezzo minimo?

PROGETTO PROGRAMMAZIONE DINAMICA A.A. 2014/2015

Natale a Flatlandia