ASD Laboratorio 02

Cristian Consonni/Alessio Guerrieri

UniTN

07/10/2016

Calendario

30/09	Introduzione
07/10	Ad-Hoc
14/10	No laboratorio
21/10	Grafi 1
28/10	Grafi 2
04/11	No laboratorio
11/11	Progetto 1
18/11	Progetto 1
25/11	Dinamica 1
02/12	Dinamica 2
09/12	No laboratorio
16/12	Progetto 2
21/12	Progetto 2

Progetti:

- 11-18 novembre;
- 16-21 dicembre;

Iscrizione ai progetti entro il **07 novembre**:
`http://bit.ly/ASDprog`

Soluzioni: Sottosequenza di somma massima

Soluzioni presenti sulle prime slides del prof. Montresor

Idea di soluzione alternativa $O(N^2)$

Costruiamo array delle somme:

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

Per ogni sottosequenza, calcoliamo la somma in $O(1)$:

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

	1	2	3	4	5
Array	2	-3	4	1	5
Somma	2	-1	3	4	9

Soluzioni: Sottomatrice di somma massima

Soluzione ovvia $(RC)^3$

- Ci sono $(RC)^2$ sottomatrici
- É possibile calcolare la somma di una sottomatrice in meno di $O(RC)$?
- Dobbiamo veramente guardare tutte le sottomatrici?

Soluzioni: Sottomatrice di somma massima

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 $[r_2, c_2]$:

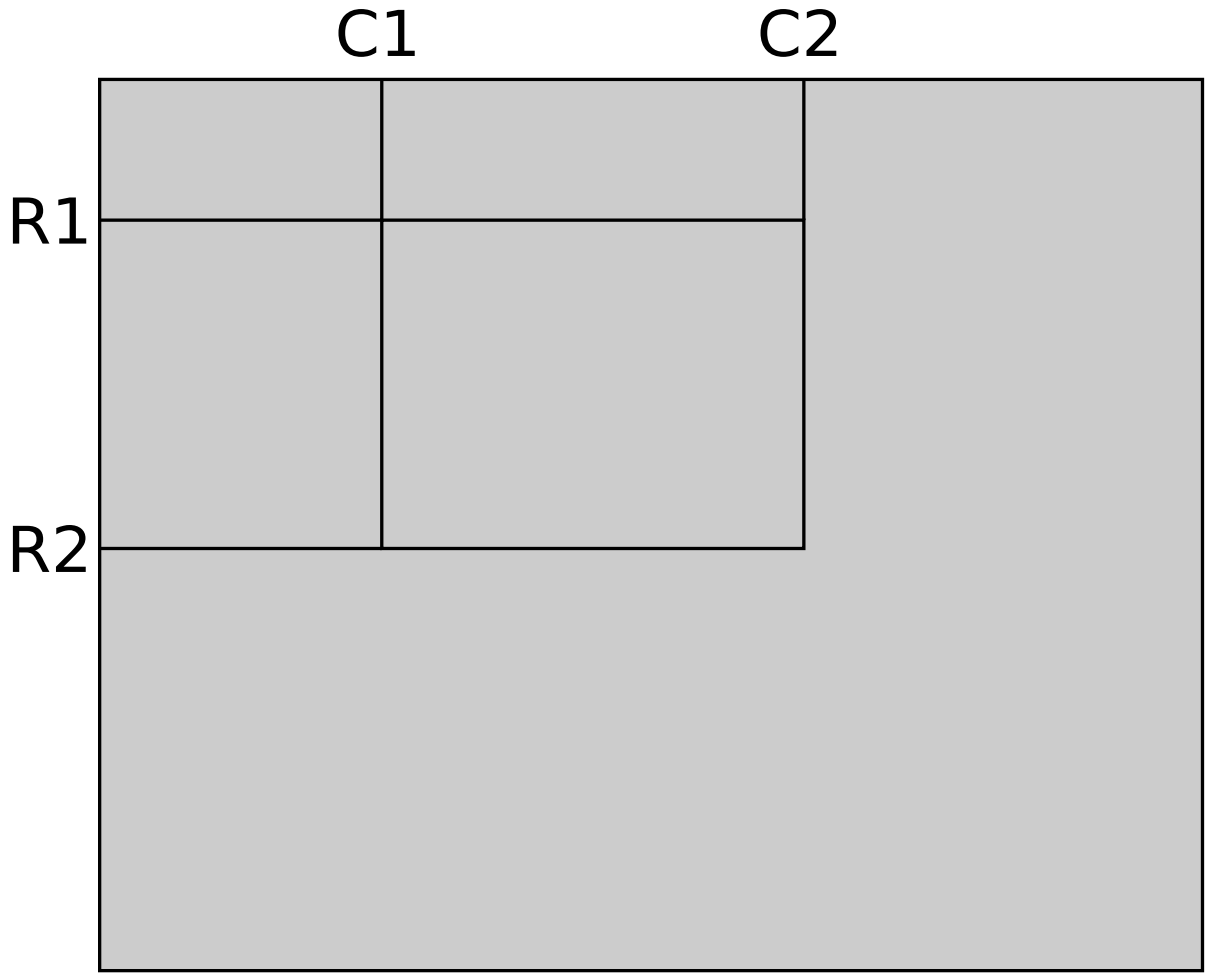
$$S[r_2, c_2] - S[r_2, c_1] - S[r_1, c_2] + S[r_1, c_1]$$

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

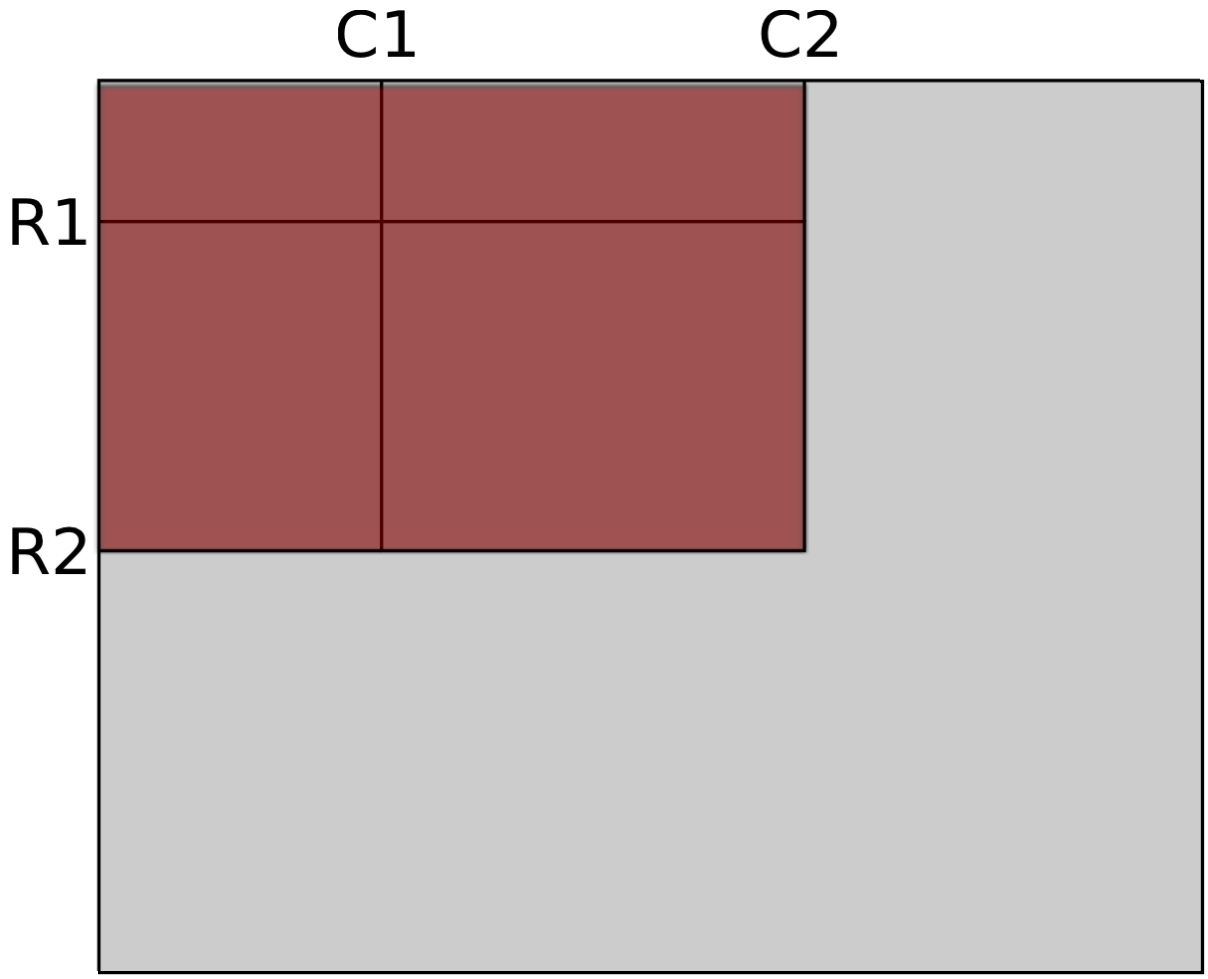
Nota implementativa

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

Soluzioni: Sottomatrice di somma massima

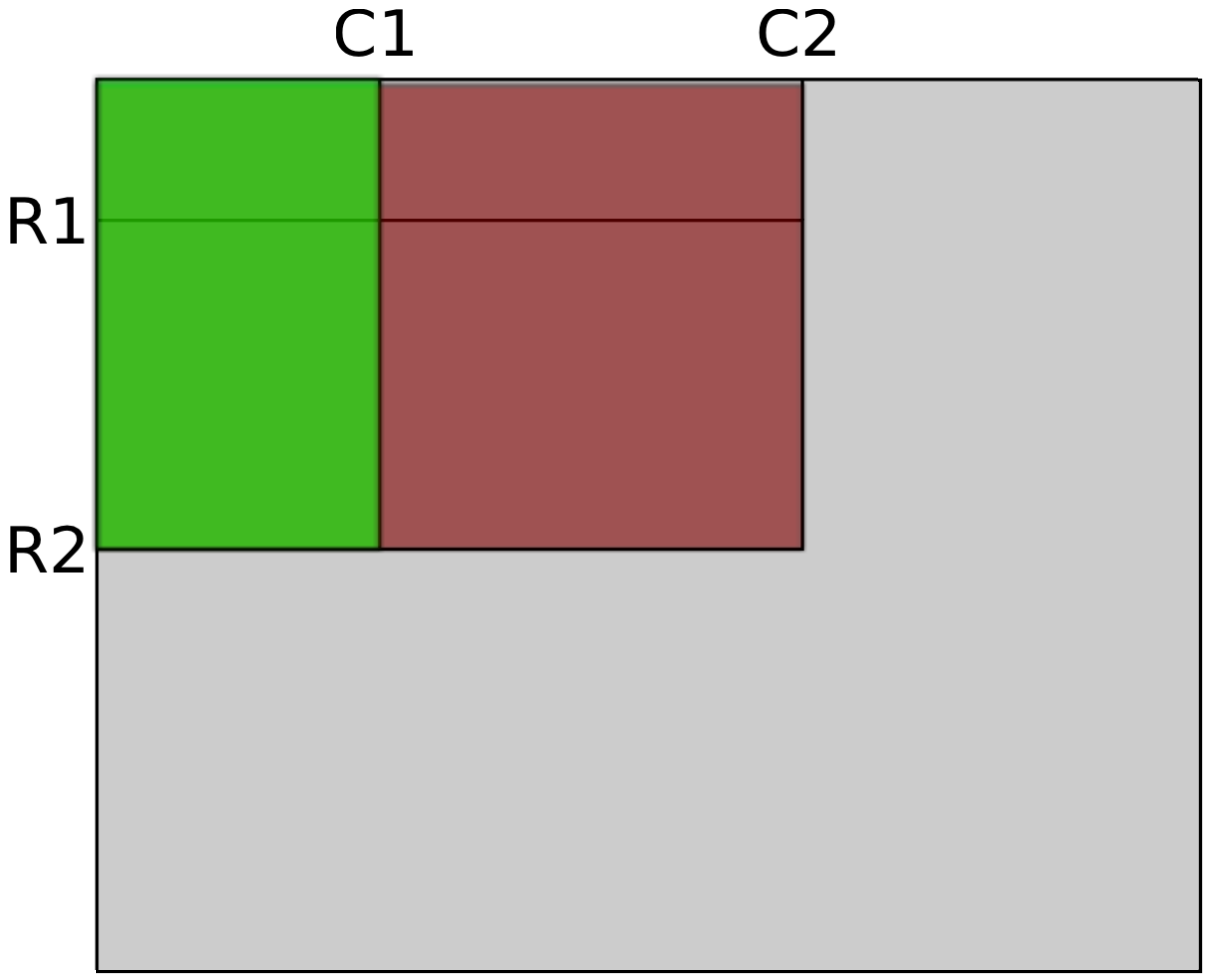


Soluzioni: Sottomatrice di somma massima



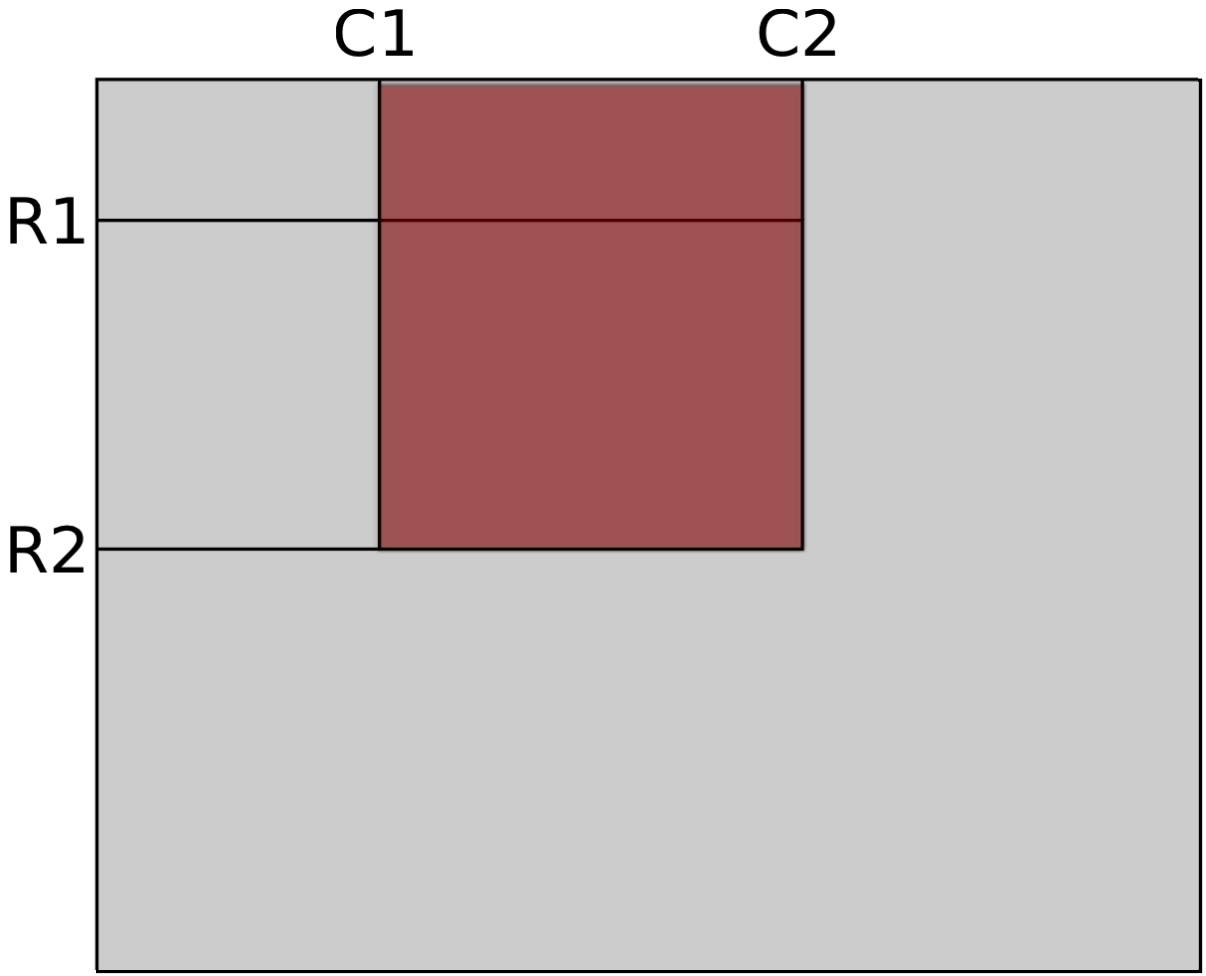
$S[r_2, c_2]$

Soluzioni: Sottomatrice di somma massima



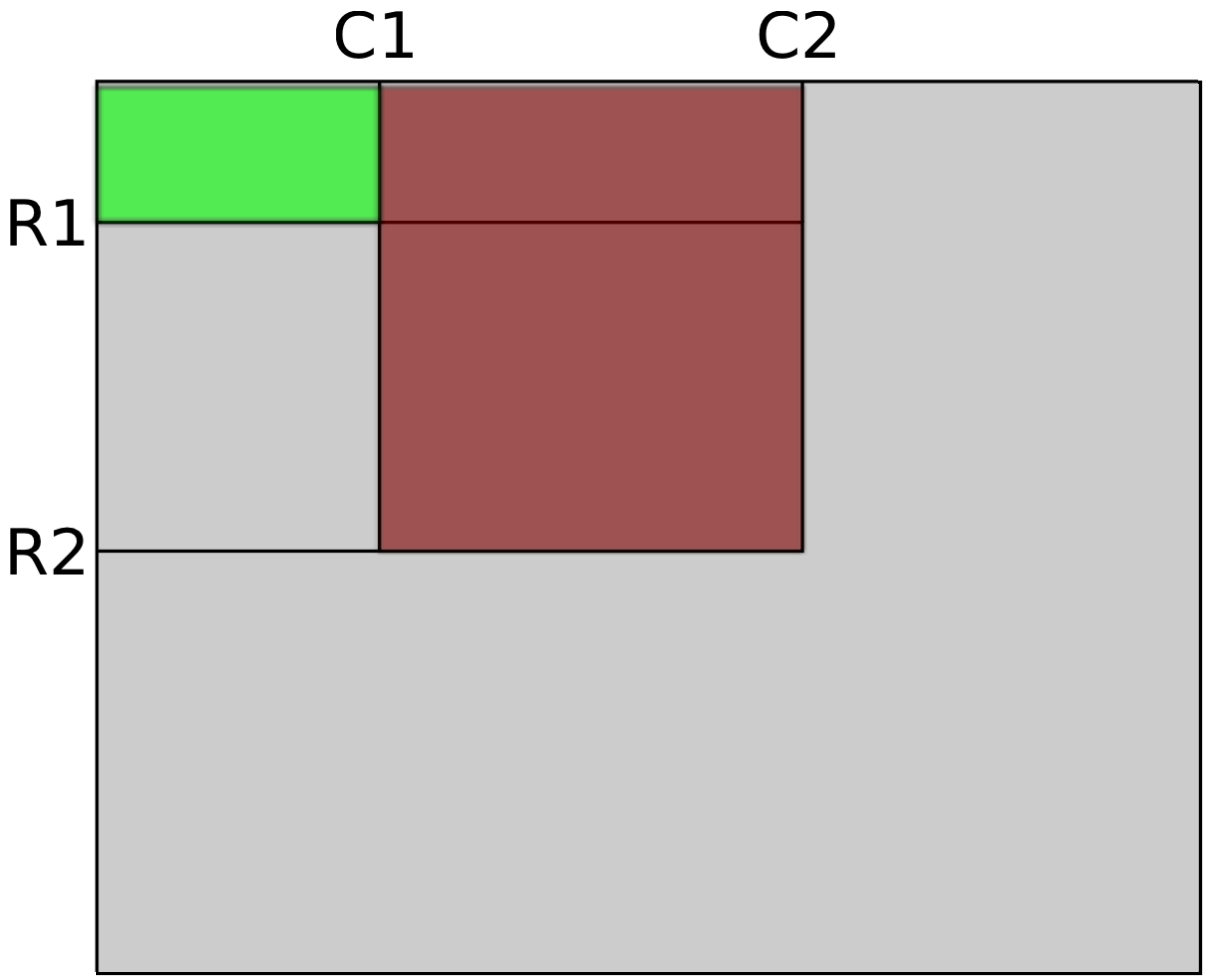
$$S[r_2, c_2] - S[r_2, c_1]$$

Soluzioni: Sottomatrice di somma massima



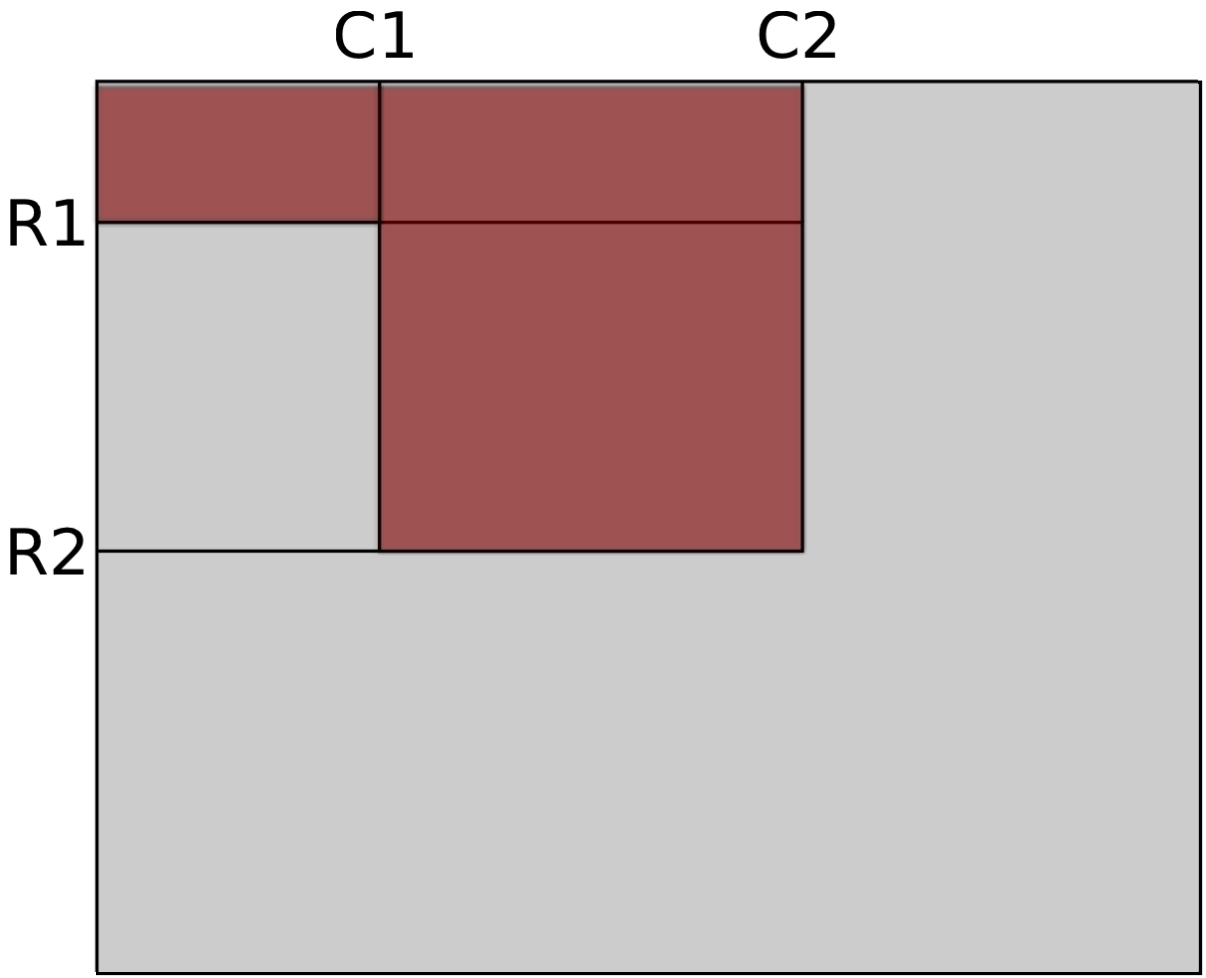
$$S[r_2, c_2] - S[r_2, c_1]$$

Soluzioni: Sottomatrice di somma massima



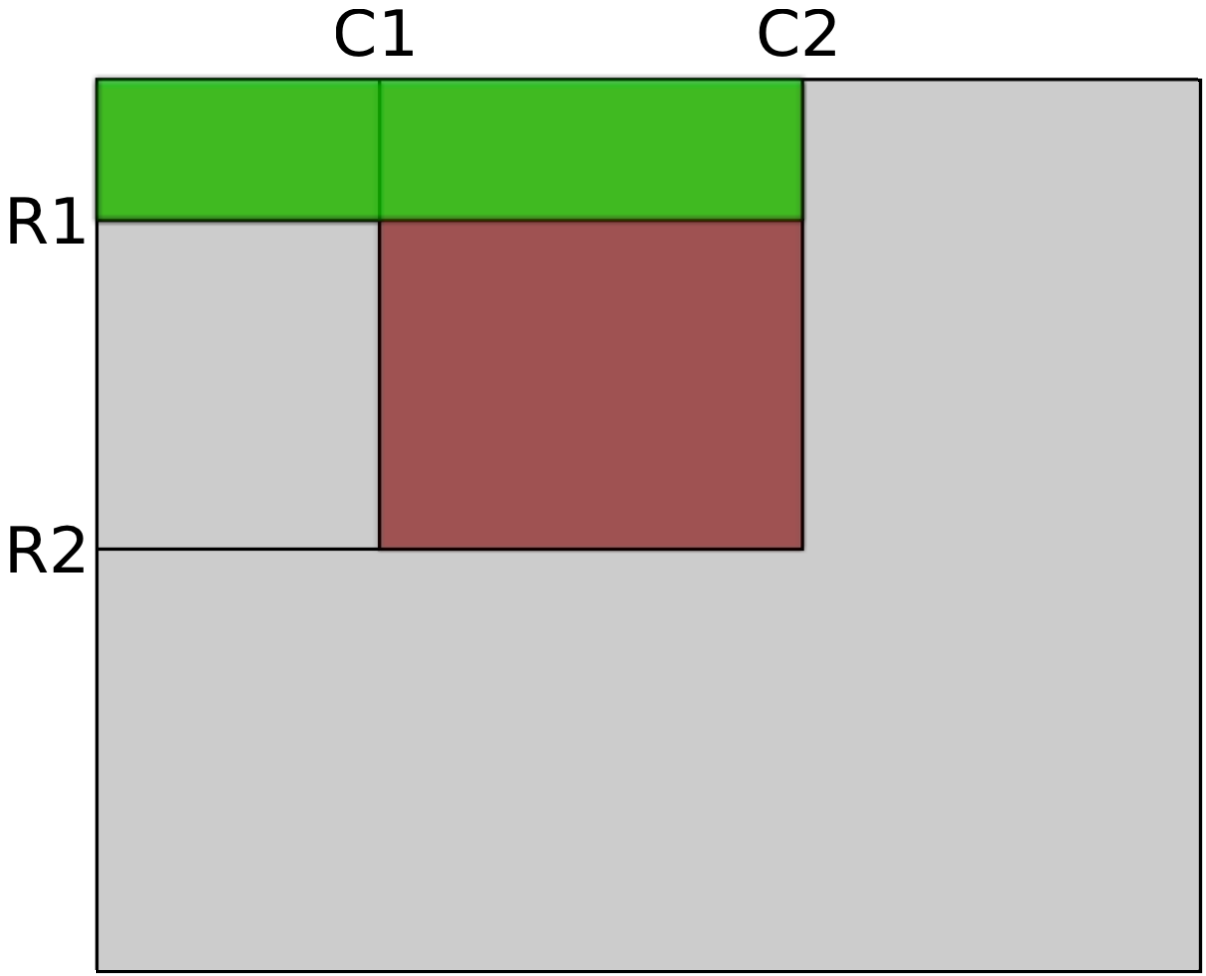
$$S[r_2, c_2] - S[r_2, c_1] + S[r_1, c_1]$$

Soluzioni: Sottomatrice di somma massima



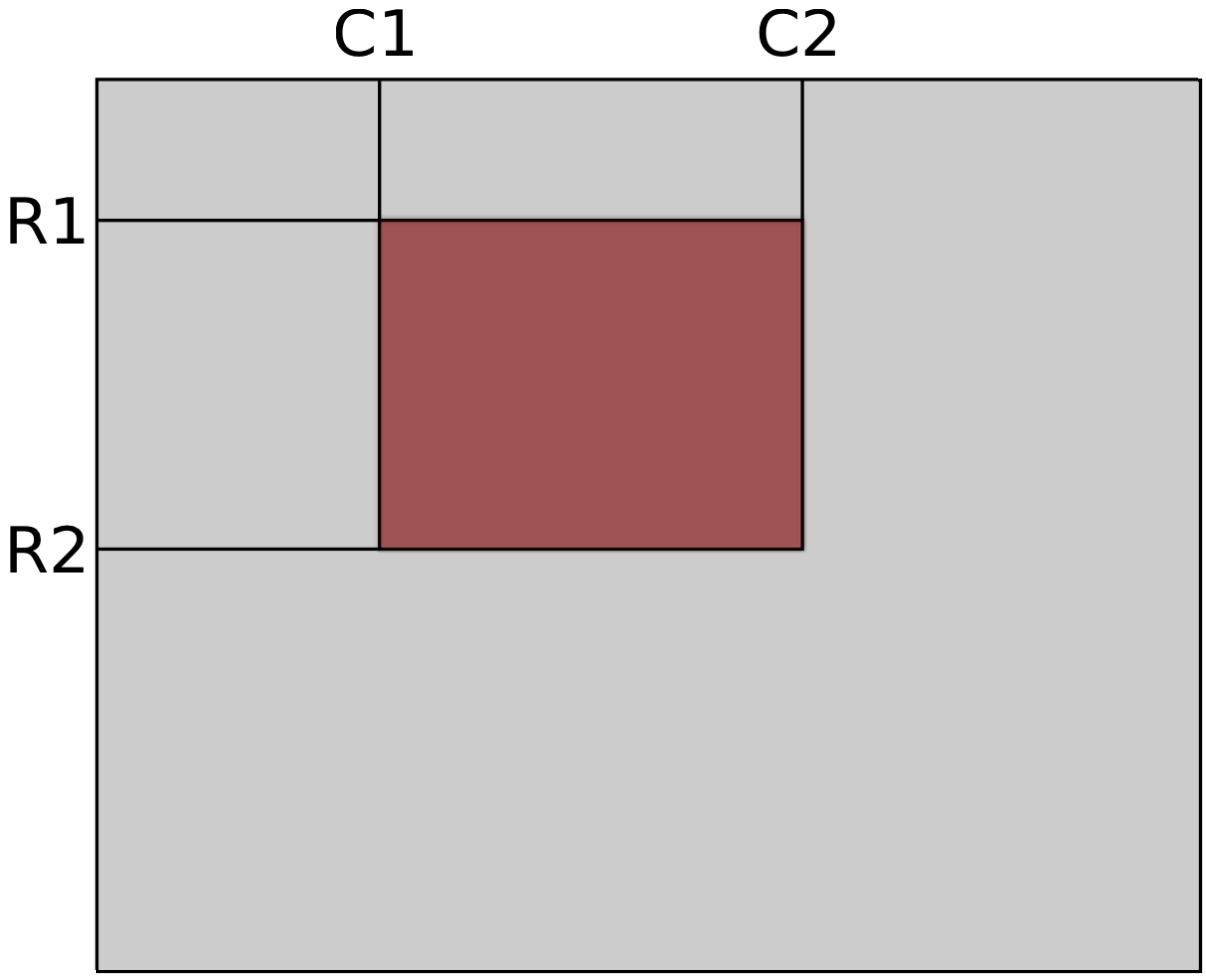
$$S[r_2, c_2] - S[r_2, c_1] + S[r_1, c_1]$$

Soluzioni: Sottomatrice di somma massima



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

Soluzioni: Sottomatrice di somma massima



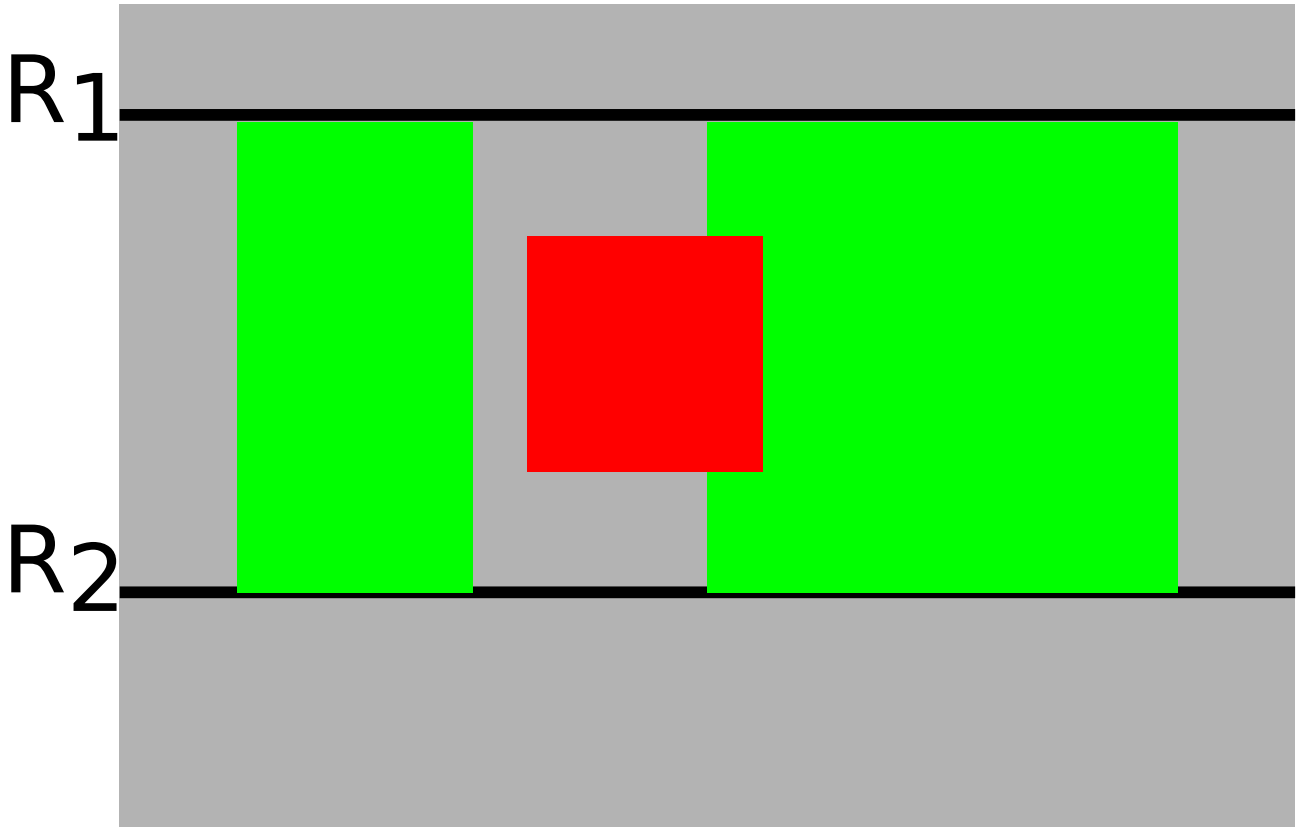
$$S[r_2, c_2] - S[r_2, c_1] + S[r_1, c_1] - S[r_1, c_2]$$

SOLUZIONE OTTIMA

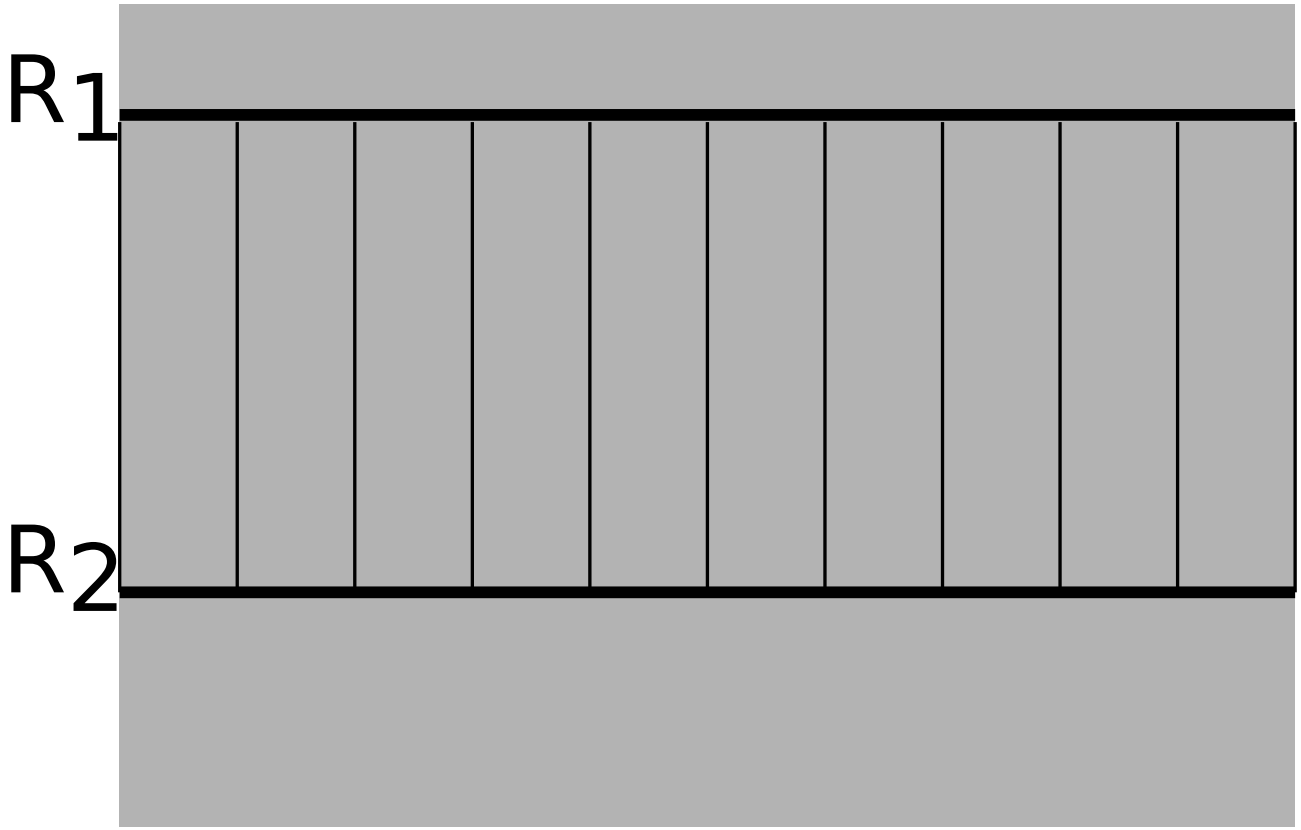
Vogliamo sfruttare la soluzione ottima $O(N)$ di sottosequenza per sottomatrice.
Proviamo ad analizzare tutte le sottomatrici che partono dalla riga R_1 e finiscono alla riga R_2 (comprese)
Se $R_2 = R_1$ siamo nel caso della sottosequenza.
Negli altri casi?

Soluzione ottima

Soluzione ottima



Soluzione ottima



Se una sottomatrice contiene l'elemento $A[R_3][C']$ allora deve contenere tutti gli elementi su quella colonna che siano inclusi fra R_1 e R_2

Per ogni coppia R_1, R_2 creiamo un istanza del problema della sottosequenza di somma massima.

-1	2	4	3	2
1	3	-4	1	1
3	1	2	-5	2
2	-1	0	1	1
-2	4	2	-1	3

6	3	-2	-3	4

Esempio: $R_1 = 2, R_2 = 4$. La sottosequenza massima 6+3 corrisponde al rettangolo $(2, 1)(4, 2)$

SOLUZIONE OTTIMA

Per ogni R_1, R_2:

1. crea array $S[1..C]$
2. $S[i] = \sum_{j=R_1}^{R_2} A[j][i]$ (calcolato con somme parziali della colonna)
3. Usa l'algoritmo lineare per la sottosequenza di somma massima su S

Sorgenti disponibili sul sito.

Problemi

Testi completi su Judge.

Sorting

Implementate un algoritmo di ordinamento $N \log N$

Intervalli

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

Sorting pesato

Avete un array di interi. Ad ogni turno potete scambiare le posizioni di due interi, pagando la loro somma. Quale é il numero minimo di turni per ordinare l'array? Quanto é il prezzo minimo?

Flatlandia

- Vecchio progetto di algoritmi
- Trovate le slides sul sito (secondo progetto, 2014/15)
- Esiste soluzione con Programmazione Dinamica
- Esiste anche soluzione ad-hoc.

SOTTOMAT: NOTA IMPLEMENTATIVA

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.