$dp[i][j]$ = min costo
per sistemare gli el. in $[1, i]$
con $x[i] = j$ alla fine

$(i, j) \longleftarrow (i-1, k)$

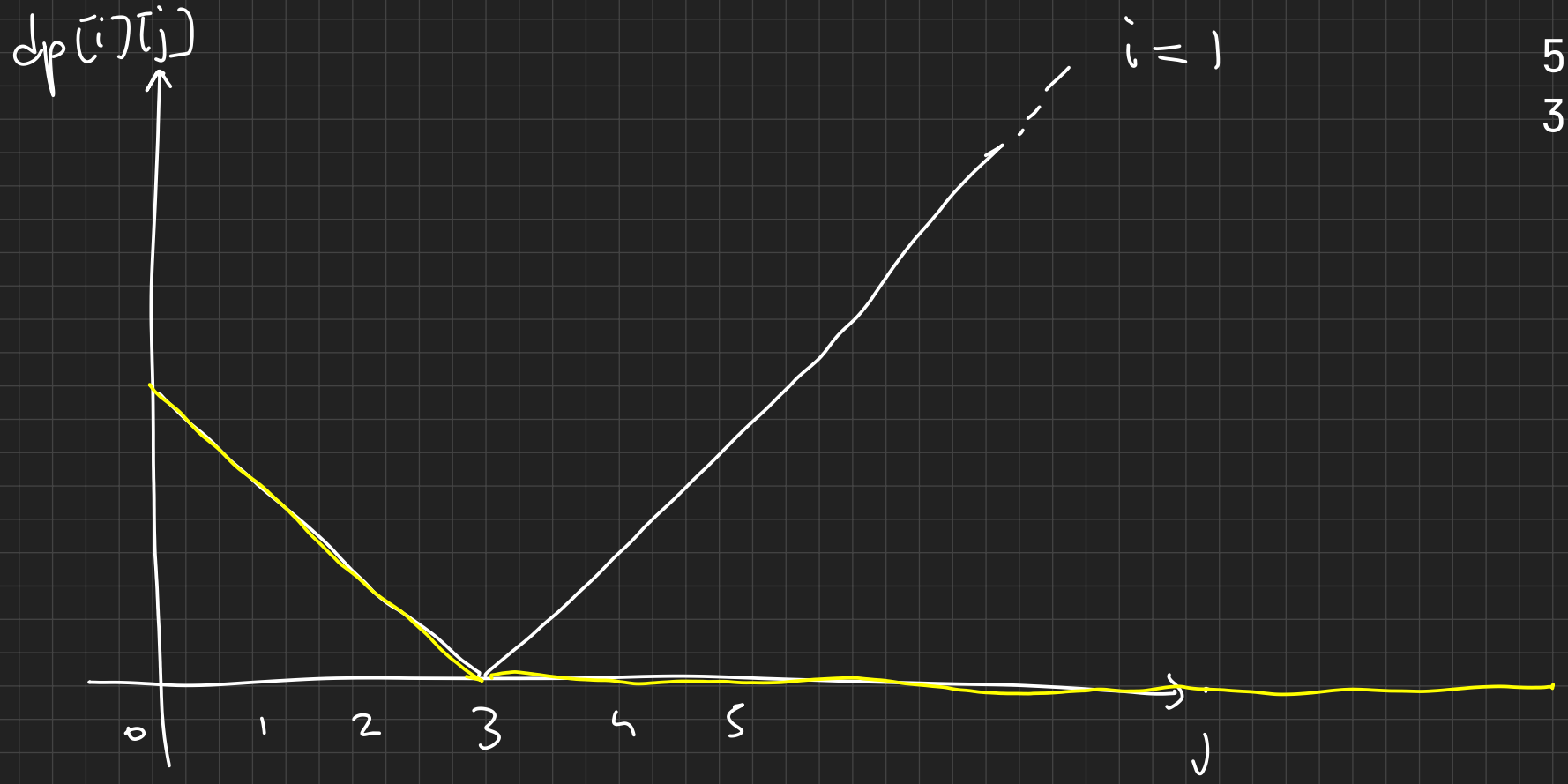
$k \leq j$

$dp[i][j] = \min(dp[i][j], dp[i-1][k] + |x[i] - j|)$

$dp[i][j] = \min_{k \leq j} dp[i-1][k] + |x[i] - j|$

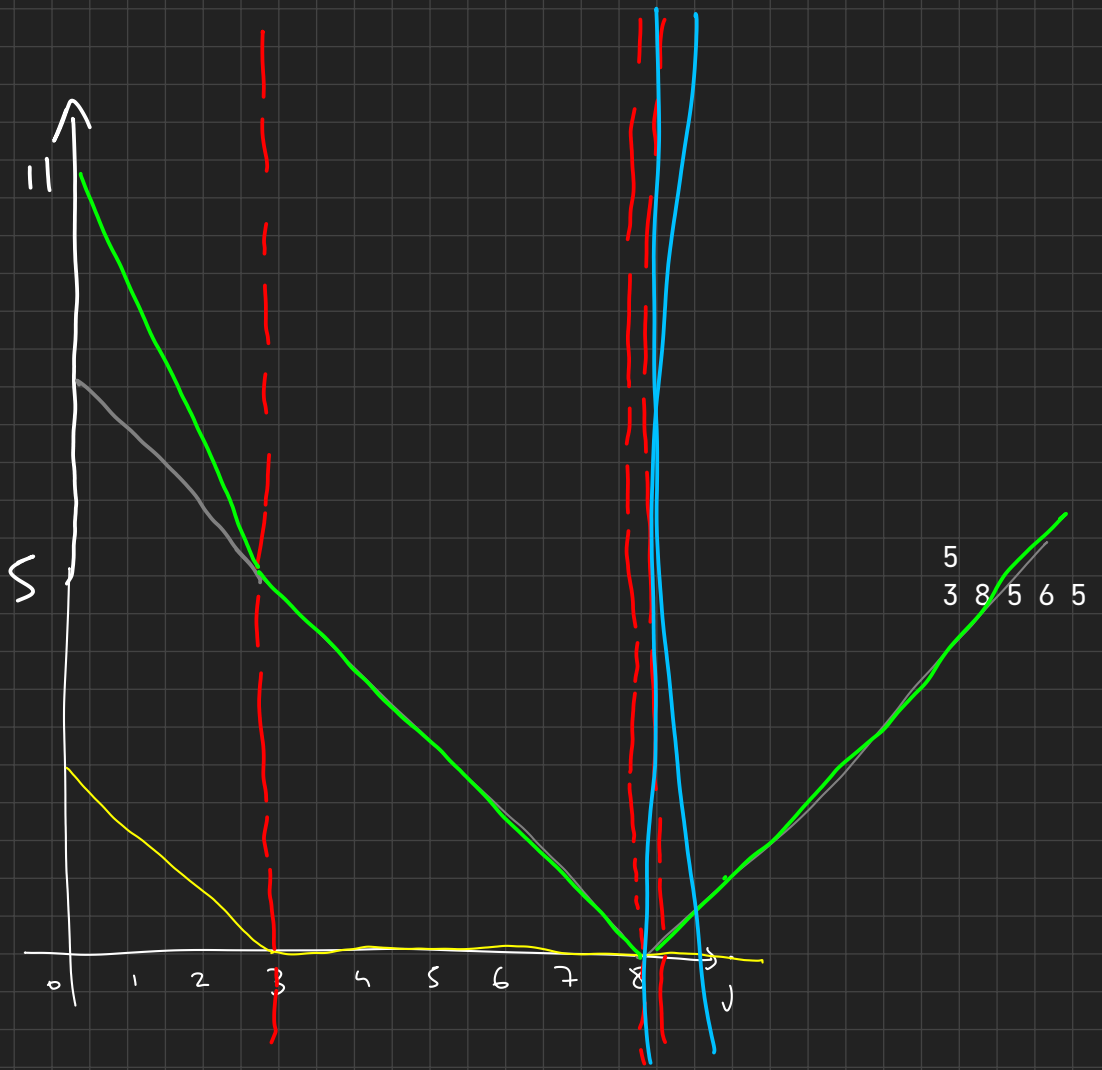
$pm[i][j] = \min_{k \leq j} dp[i][k]$

$dp[i][j] = pm[i-1][j] + |x[i] - j|$



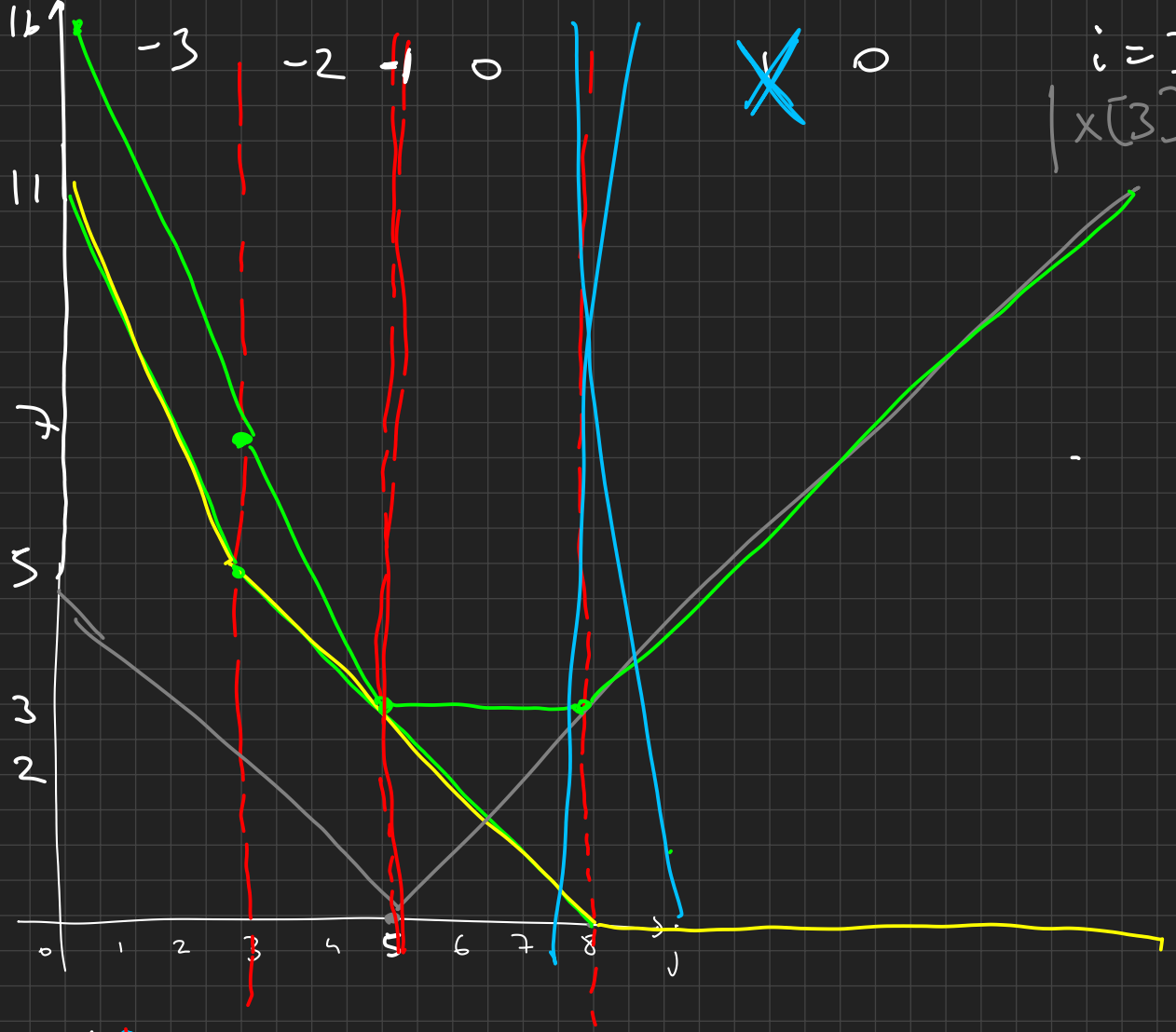
5
3 8 5 6 5


dp[i][j]
i = 2
5
3 8 5 6 5
3
0 1 2 3 4 5 6 7 8
j



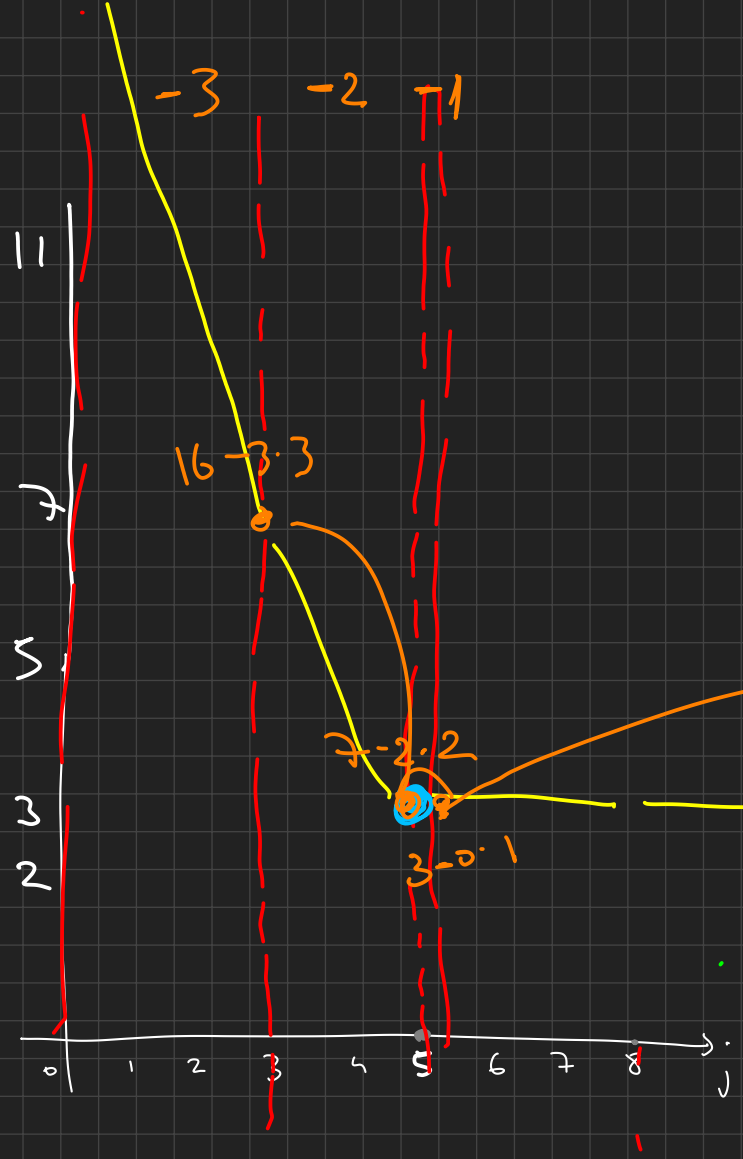
5
3 8 5 6 5

5
3 8 5 6 5



5
3 8 5 6 5

$[-i, 1]$

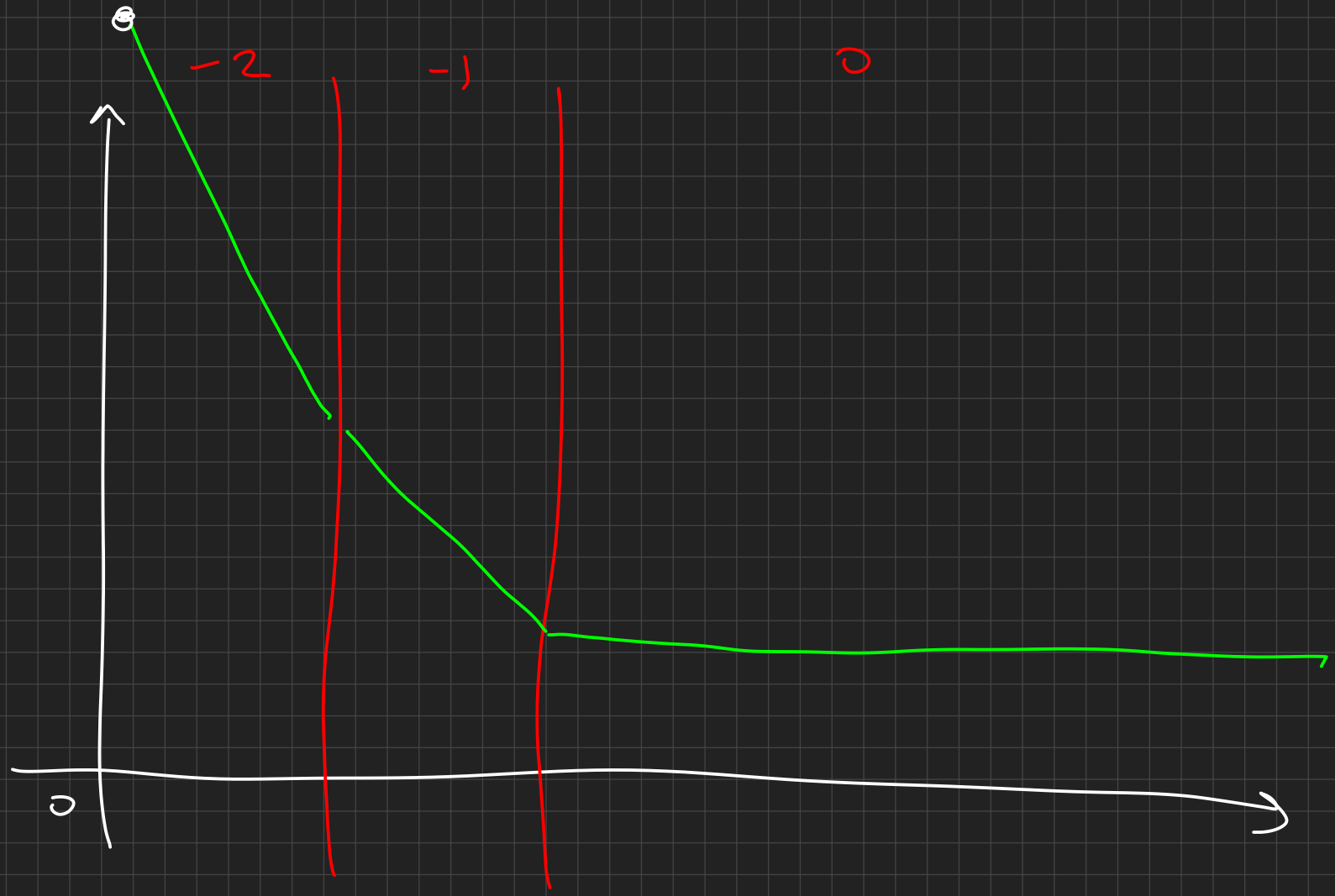


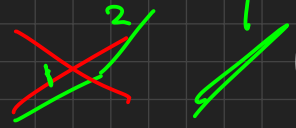
$i = 3 \quad |x[3] - 5|$

$dp[n][0]$ = min costo per sistemare $[1, i]$ con $x[n] = 0$

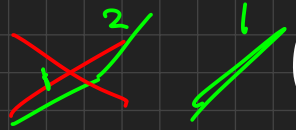
• sol. ottimale: portare tutto a 0

costo: $\sum x[i]$


-2
-1
0
0

- sommo
  - diminuisco coeff. di 1 a sinistra, aumento di 1 a destra
- tolgo  (prefix min)
  - allo step precedente (i-1) avevo coefficienti ≤ 0 quindi adesso ho coeff. ≤ 1 devo togliere solo 1 tratto

non si verifica mai che ho ≥ 2 tratti crescenti (ne ho sempre esattamente 1)

- sommo
  - diminuisco coeff. di 1 a sinistra, aumento di 1 a destra = aggiungo 2 linee rosse in x(i)
  - tolgo 

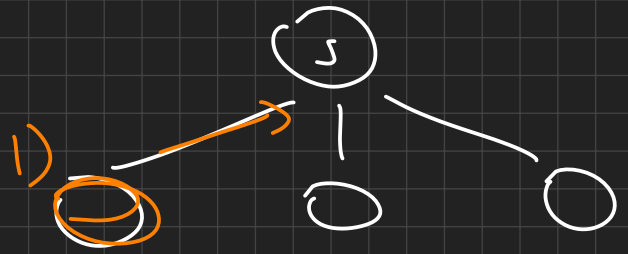
(prefix min)
    - allo step precedente (i-1) avevo coefficienti ≤ 0 quindi adesso ho coeff. ≤ 1 devo togliere solo 1 tratto = tolgo la max linea rossa

non si verifica mai che ho ≥ 2 tratti (ne ho sempre esattamente 1)

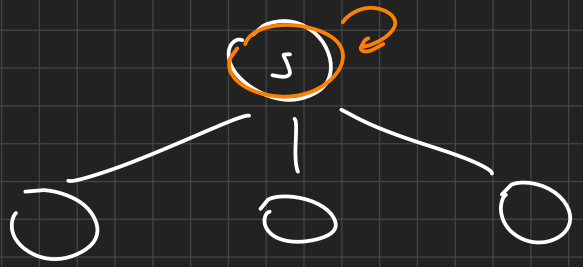
- alla fine, come trovo il risultato?
  - lo trovo in corrispondenza della max linea rossa

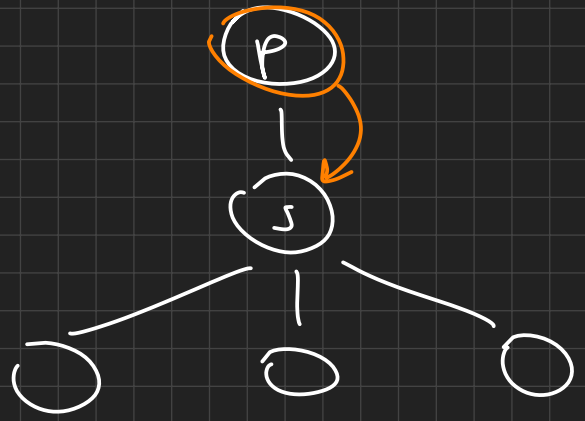
- controllo tutti i nodi

dp[s] = # nodi per controllare il subtree di s



1) controllo s con un figlio
2) controllo s con se stesso
3) controllo s con il padre





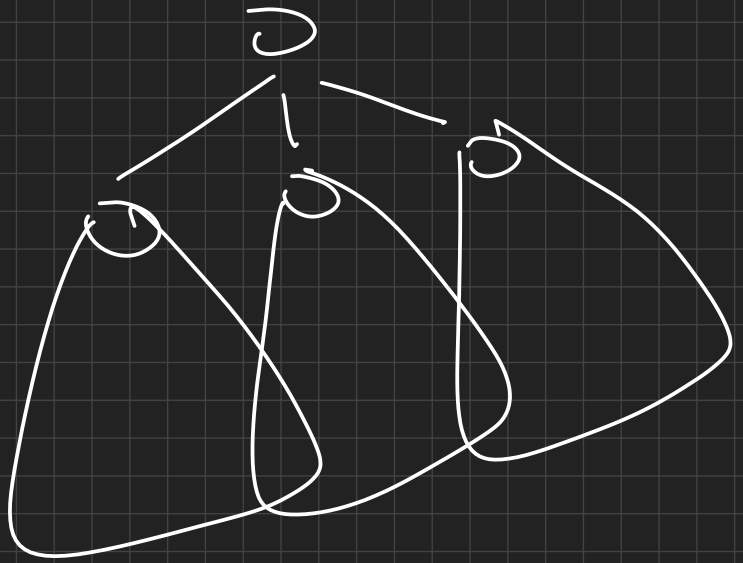
$dp[s][b][c]$ = # modi

per controllare il subtree di s

b = [ho scelto il nodo s nel sottoinsieme?]

c = [s è controllato?]

b = 1 → c = 1



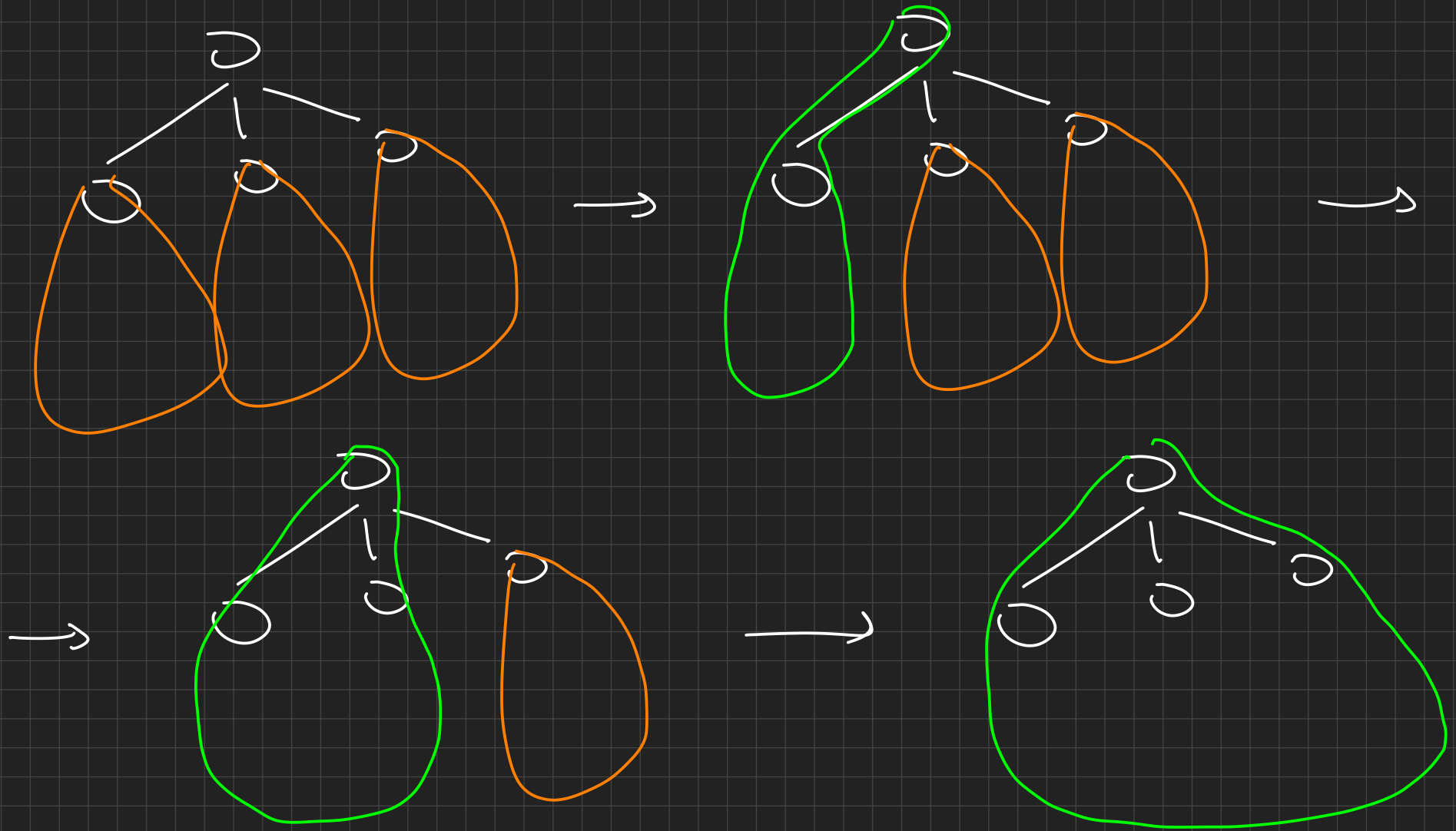
$dp[s][j]$ = # modi per controllare il subtree di s

- s è scelto — $j = 0$
- s è controllato — $j = 1$
- s non è controllato (dovrà essere controllato dal padre) — $j = 2$

$dp[s][c][j]$ = # modi

per controllare c nodi nel subtree di s

- s è scelto — $j = 0$
- s è controllato — $j = 1$
- s non è controllato (dovrà essere controllato dal padre) — $j = 2$



$dp[s][j]$ = # modi per controllare j nodi nel subtree di s

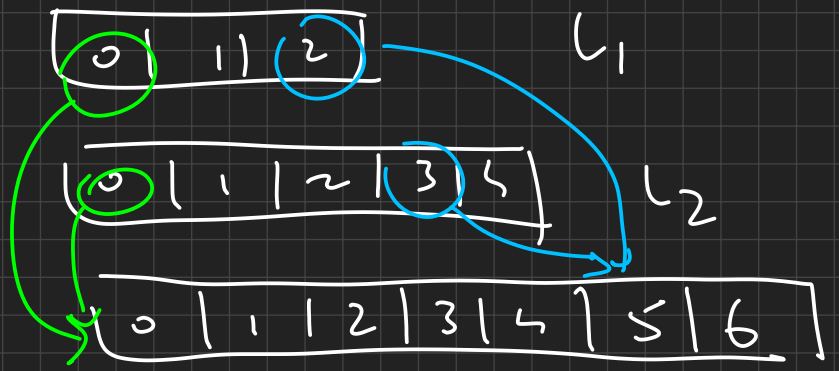
$(a, b) \longrightarrow s$

$$dp[s][j] = \sum_{k=0}^{j} dp[a][k] \cdot dp[b][j-k]$$

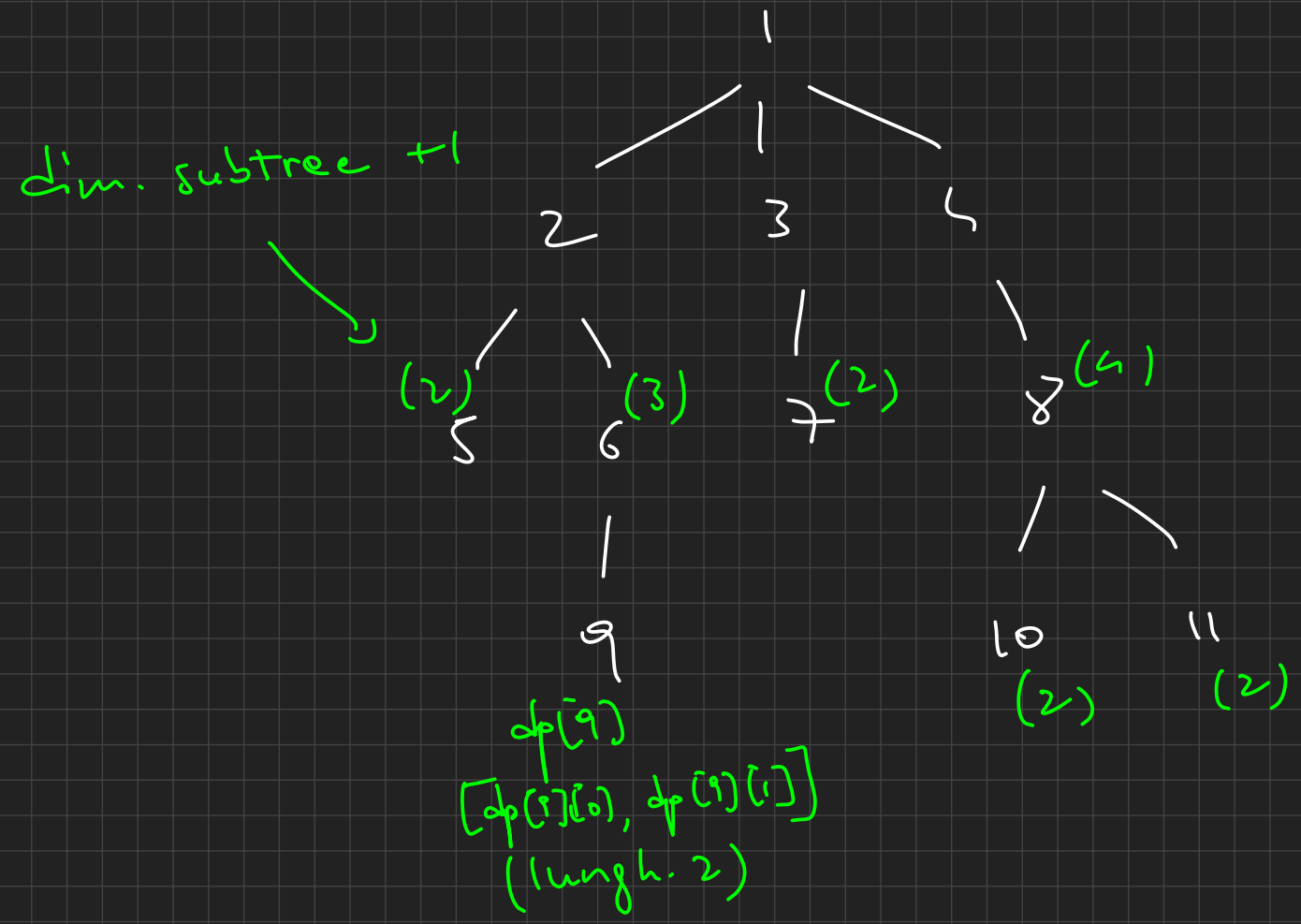
transizione in $O(n)$

$n \cdot n$

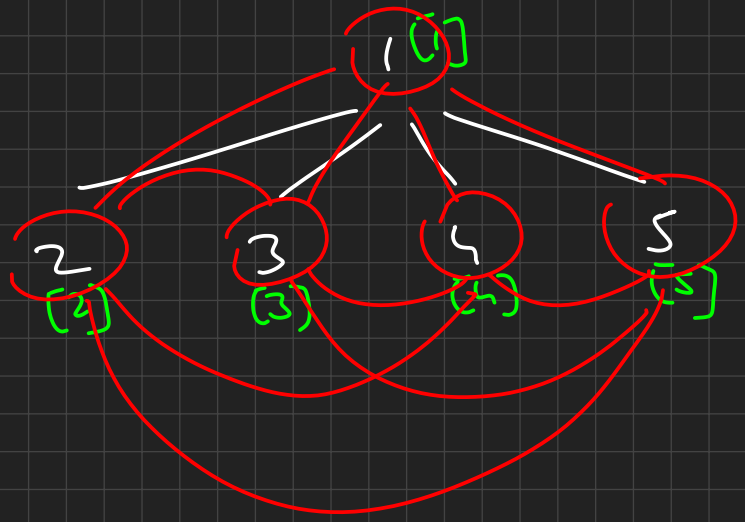
$O(n^3)$

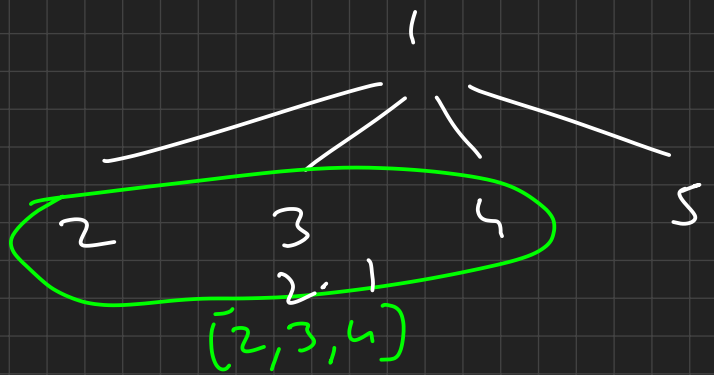


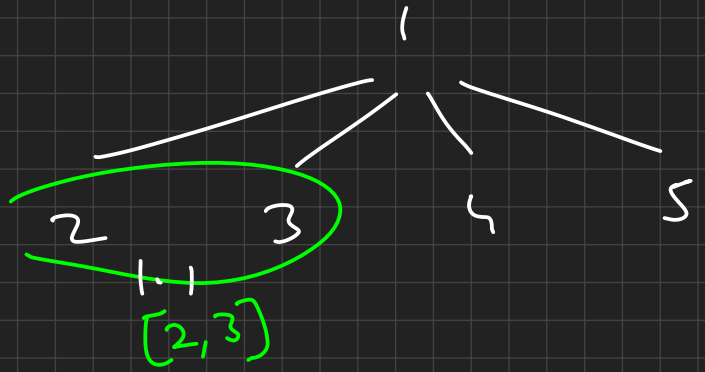
$L_1 \cdot L_2$ op.

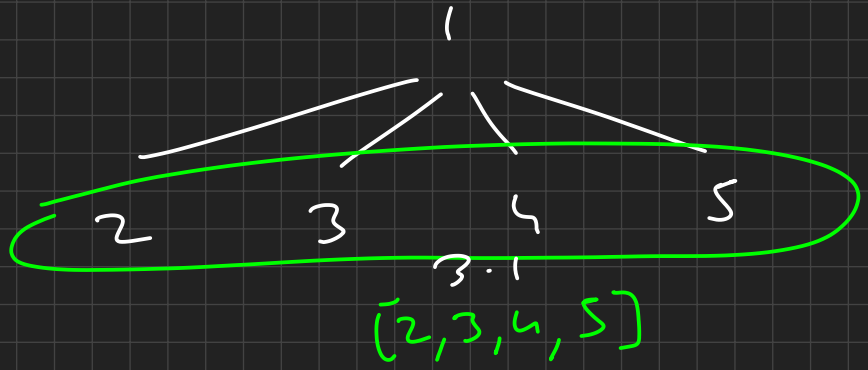


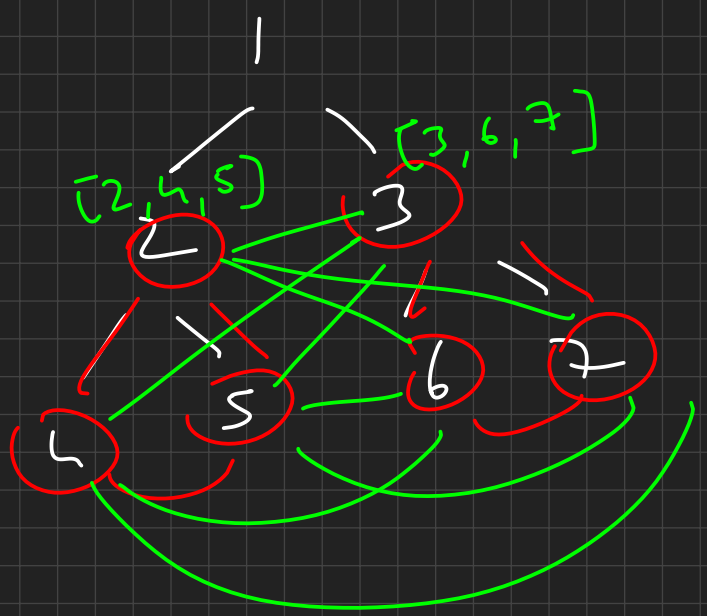
$O(n^2)$


1
[]
2
[4]
3
[3]
4
[4]
5
[5]


1
2
3
4
5
2, 1
[2,3,4]


1
2
3
4
5
1, 1
[2,3]

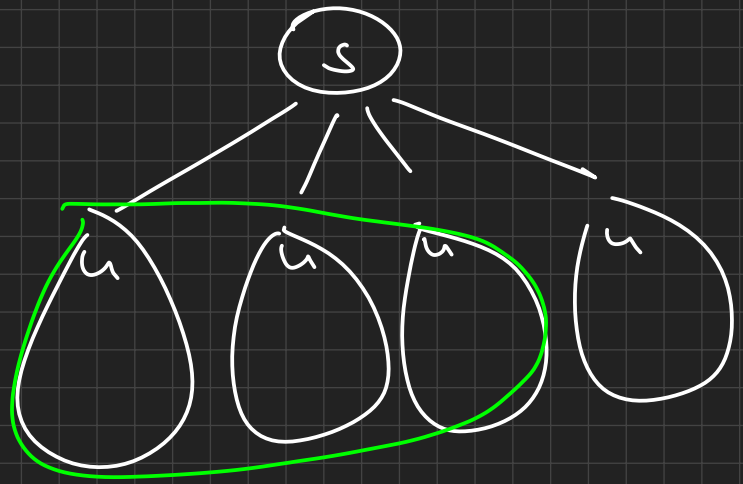

1
2
3
4
5
3, 1
[2,3,4,5]



alt.

- claim: per ogni subtree S, impiego $\binom{sz(S)}{2}$ op.

(si dimostra per induzione)



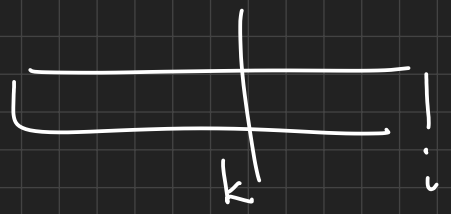
- decido se prendere s
  - se lo prendo, $j = 2 \Rightarrow +1$ ai nodi controllati
  - se non lo prendo, ma $j = 0 \rightarrow$ $\Rightarrow +1$ ai nodi controllati

$0 \rightarrow 1$

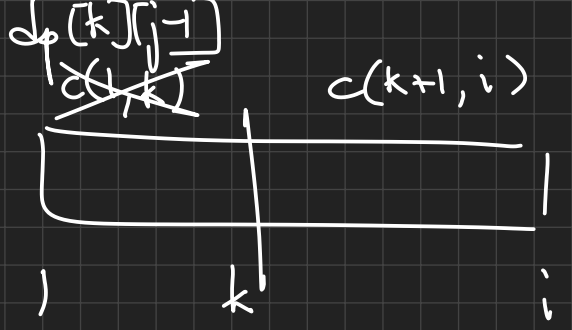
$1 | 2 \rightarrow 2$

- u è scelto — $j = 0$
- u è controllato — $j = 1$
- u non è controllato (dovrà essere controllato dal padre) — $j = 2$

$dp[i][j]$ = min costo per dividere $(1, i)$ in j pezzi



$$dp[i][j] = \min_{k<i} \left(dp[k][j-1] + c(k+1, i) \right)$$

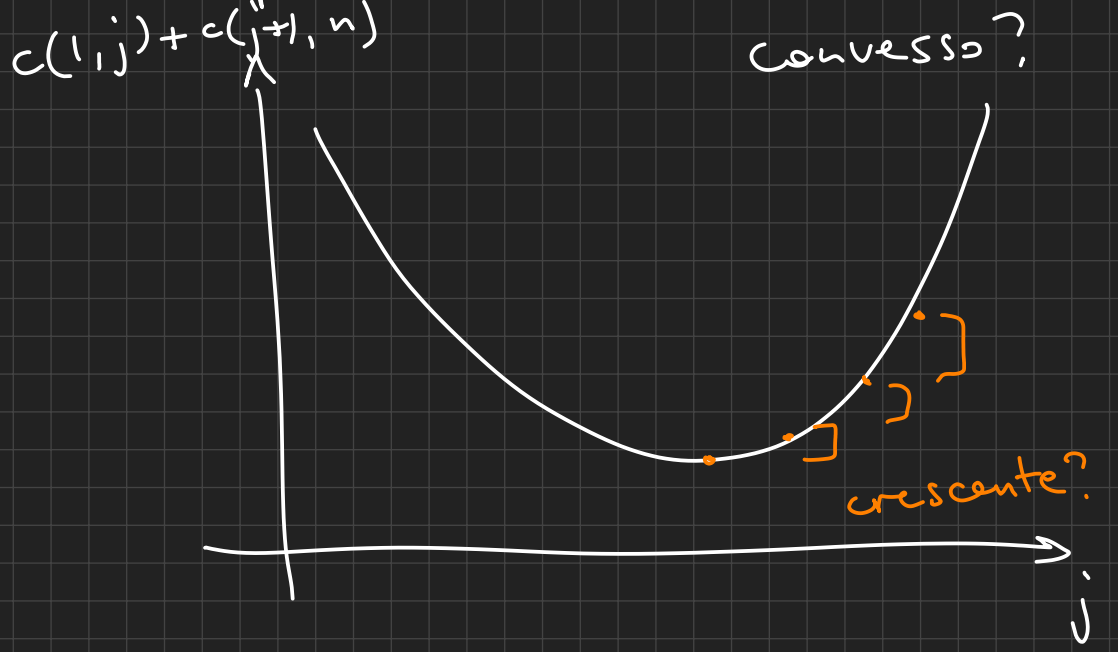


2 subarray

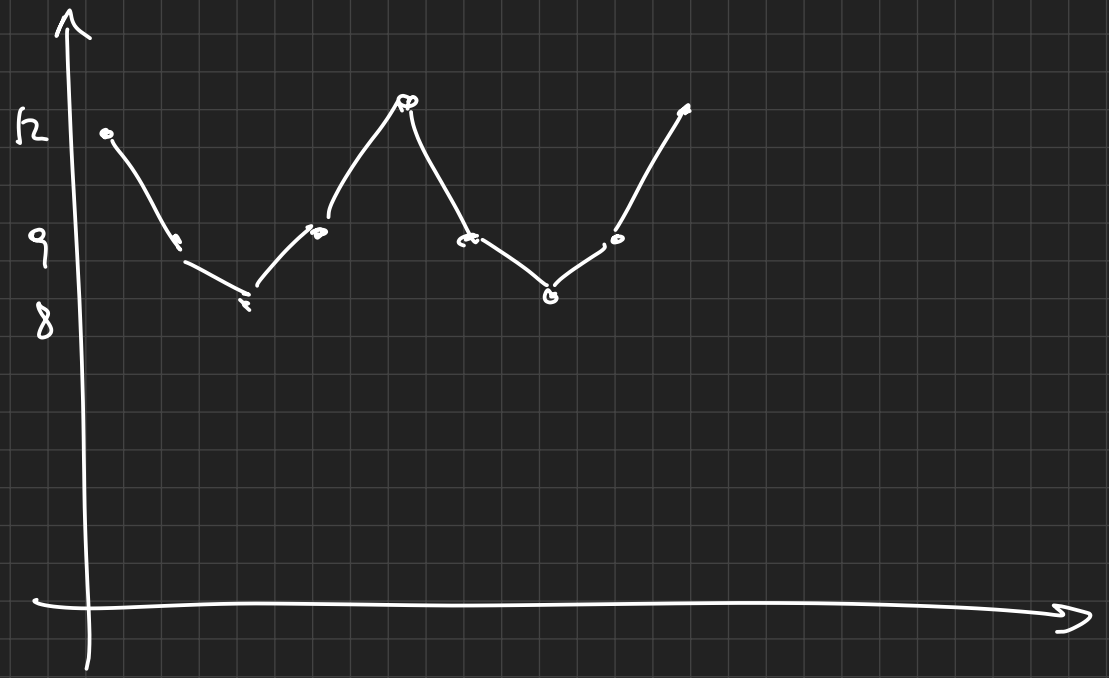
$$dp[i][j] = \min_{k<i} \left(c(1, k) + c(k+1, i) \right)$$

- $k = 1$ ✓
- $k = 2$

cosa succede se voglio calcolare per ogni $[1, i]$ il costo minimo?

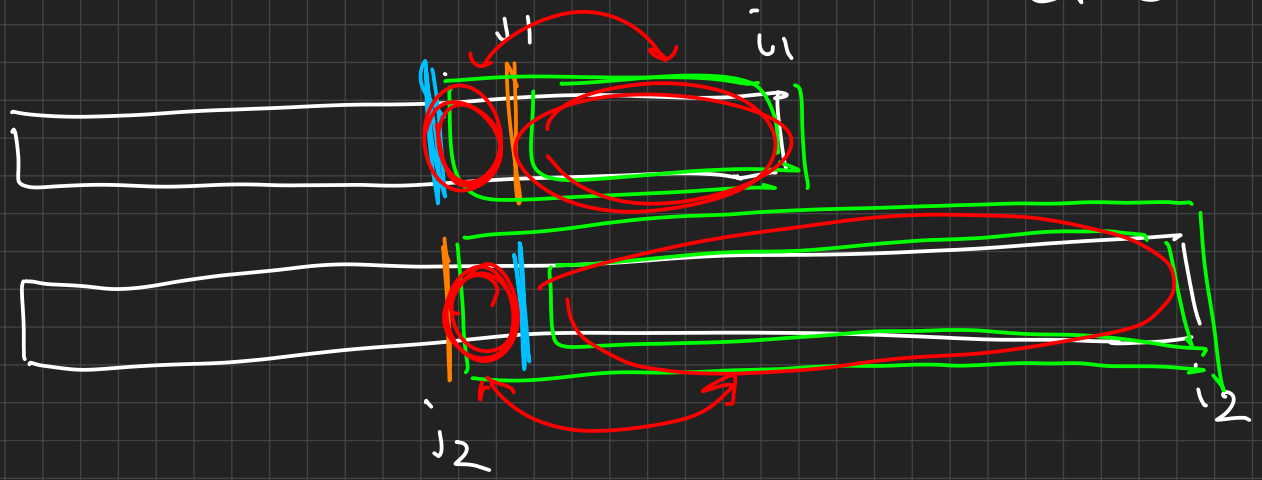


1 1 1 1 2 2 2 2



- claim: la posizione ottimale di 1 è crescente
  - pos. ottimale := quella con costo minimo
    in caso di parità, quella a sinistra

per assurdo,



$c(j_2+1, x) - c(j_1+1, x)$
crescente in x

$$c(1, j_1) + c(j_1+1, i_1) < c(1, j_2) + c(j_2+1, i_1)$$

$$c(1, j_2) + c(j_2+1, i_2) \leq c(1, j_1) + c(j_1+1, i_2)$$

$$c(j_2+1, i_1) - c(j_1+1, i_1) \leq c(j_2+1, i_2) - c(j_1+1, i_2)$$

dovrebbe dare un assurdo

$$c(1, j_1) + c(j_1+1, i_1) < c(1, j_2) + c(j_2+1, i_1)$$

$$c(1, j_2) + c(j_2+1, i_2) \leq c(1, j_1) + c(j_1+1, i_2)$$

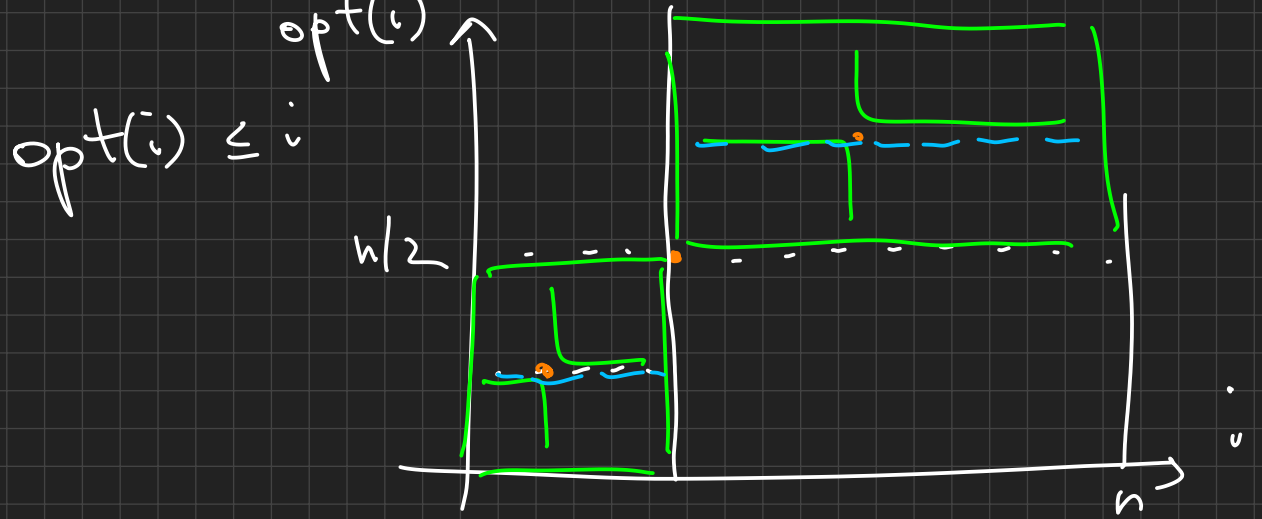
$$c(j_2+1, i_1) - c(j_1+1, i_1) \leq c(j_2+1, i_2) - c(j_1+1, i_2)$$

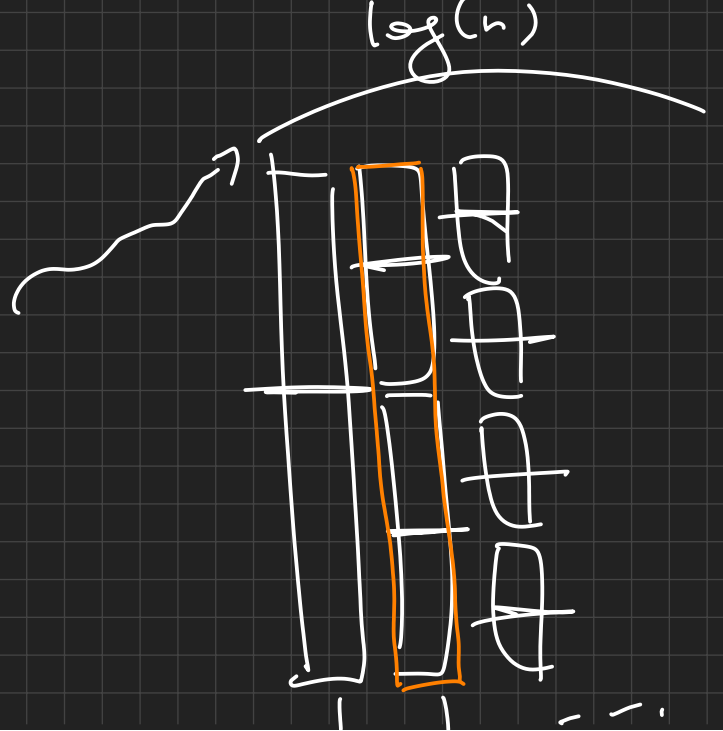
$$c(j_1+1, i_1) + c(j_2+1, i_2) < c(j_1+1, i_2) + c(j_2+1, i_1)$$

opt(i) = pos. di | per il prefisso (1, i)

opt(i) crescente

- calcolo opt(n/2) a mano (in O(n))
- continuo ricorsivamente



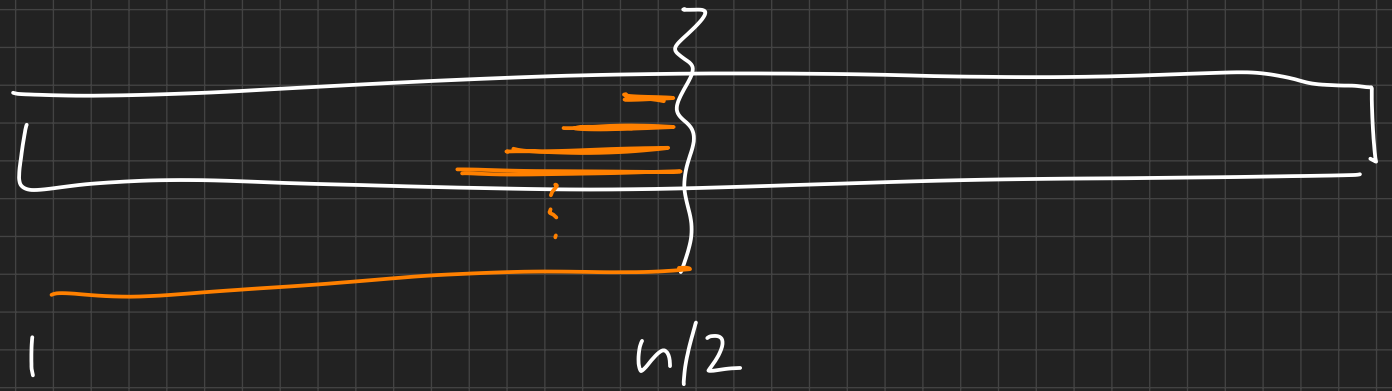


- assumo di saper calcolare $c(l,r)$ in $O(1)$
  so anche trovare un punto separazione in $O(y)$
  $O(n \log(n))$ in tutto

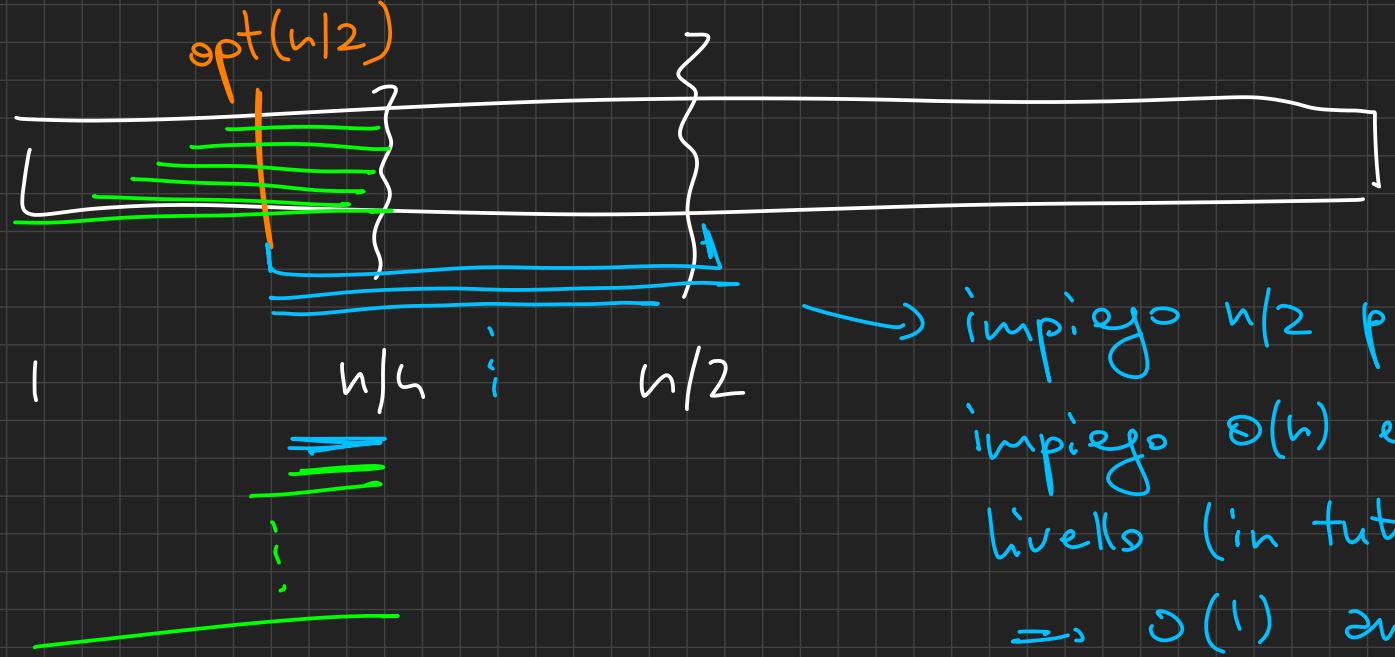
- k generico
  - trovo gli split con $k=2$ $\Rightarrow$ trovo il costo per dividere (l,i) in 2
  - trovo gli split con $k=3$ (usando ___) $\Rightarrow$
    $\Rightarrow$ trovo il costo per dividere in 3

$O(kn\log(n))$

- calcolo $c(l, r)$ in $O(1)$ ammortizzato



- scorro da destra a sinistra



→ impiego n/2 per togliere el.
impiego $\Theta(n)$ extra per ogni livello (in tutto) ⇒
⇒ $O(1)$ ammortizzato