ESERCIZIO 1

$S_1 = \{1, 8, 10\}$

$S_2 = \{1, 2, 5, 7, 10, 15, 20\}$

Mutual partition algorithm

Prendiamo il set S1 che ha meno elementi, prendiamo 8 che è l'elemento di mezzo e lo cerchiamo in S2.
Dato che non c'è copianza che dobbiamo fare due ricerche di 1 in
$\{1, 2, 5, 7\}$ e di 10 in $\{10, 15, 20\}$
la ricerca dice che 1 e 10 compaiono in S2.
Quindi l'intersezione è $\{1, 10\}$

$S_1 = \{1, 8, 10\}$

$S_2 = \{1, 2, 5, 7, 10, 15, 20\}$

Binary search with exponential jump:

Cerchiamo gli elementi di S1 in S2

→ funziona da 1 che viene cercato in
$[i, 2^k+i]$ con $i = \emptyset$ $k=0$
$[0, 1]$ → 1 lo trovo subito e
lo restituisco

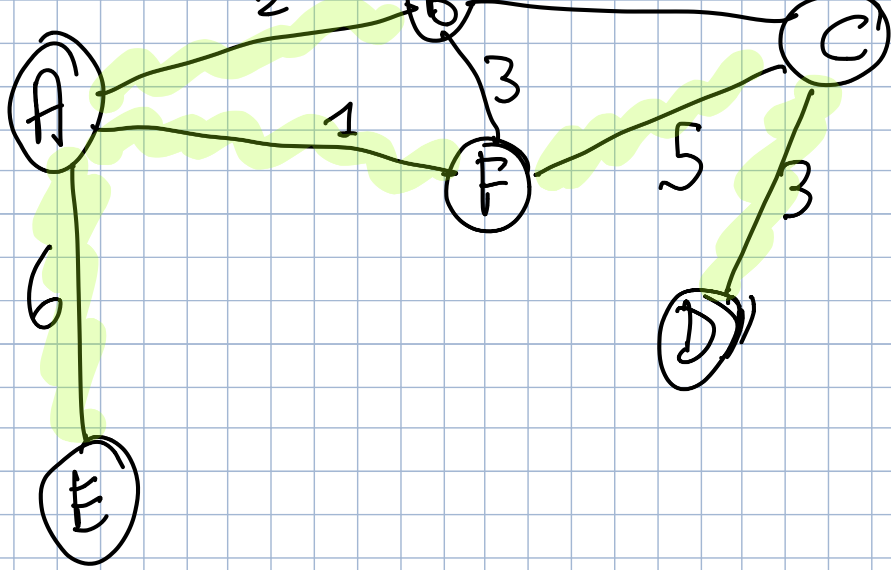
→ Cerchiamo 8 in $i = 1$ $k=0$
$[1, 2^k+1] = [1, 1+1] = [1,2)$ non c'è
$[1, 2^1+1] = [1, 3)$ no
$[1, 2^2+1] = [1, 5]$ trovo $10 > 8$
quindi se c'è 8 è nel range
$[3, 5)$ cerco qui e non lo trovo

→ Cerchiamo 10 $i = 2$

$$S_2 = \{1, 2, 5, 7, 10, 15, 20\}$$

10 qui trovo 10 e
lo restituisco

ESERCIZIO 2



Kruskal:

(A, F, 1)

(A, B, 2)

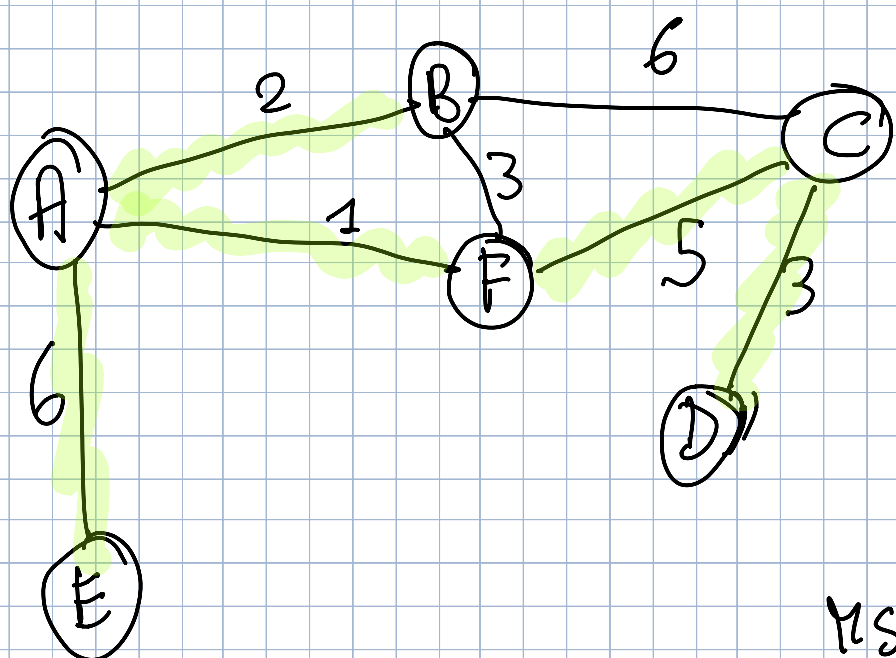
no si crea un ciclo ← (B, F, 3)

(C, D, 3)

(F, C, 5)

(A, E, 6)

(B, C, 6)



MST: A, F

Prim Iniziamo da A

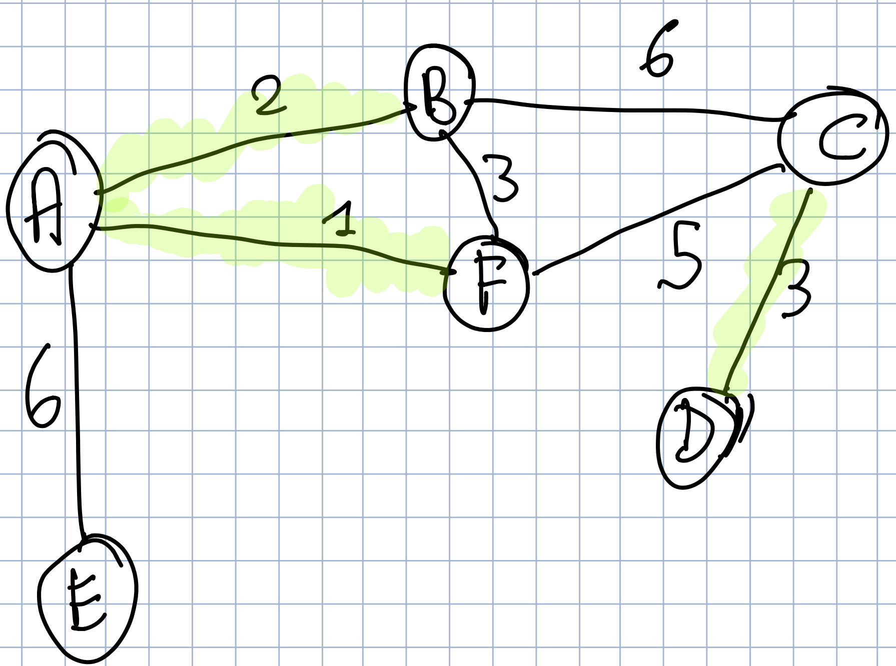
$Q = (\langle A, F, 1\rangle, \langle A, B, 2\rangle, \langle A, E, 6\rangle)$

$Q = (\langle A,B,2\rangle, \langle F,C,5\rangle, \langle A,E,6\rangle)$

$Q = (\langle F,C,5\rangle, \langle A,E,6\rangle)$

$Q = (\langle C,D,3\rangle, \langle A,E,6\rangle)$

$Q = \langle A,E,6\rangle)$



$\langle A,F,1,A,F\rangle$
$\langle A,B,2,A,B\rangle$
$\langle A,E,6,A,E\rangle$

$\langle B,F,3,B,F\rangle$
$\langle B,C,6,B,C\rangle$
$\langle C,D,3,C,D\rangle$
$\langle C,F,5,C,F\rangle$

Sybern

$\langle A,F,1,A,F\rangle$
ralinkto = F

$\langle B,F,2,A,B\rangle$

$\langle B,F,3,B,F\rangle$

$\langle B,E,6,A,E\rangle$

$\langle B,C,6,B,C\rangle$

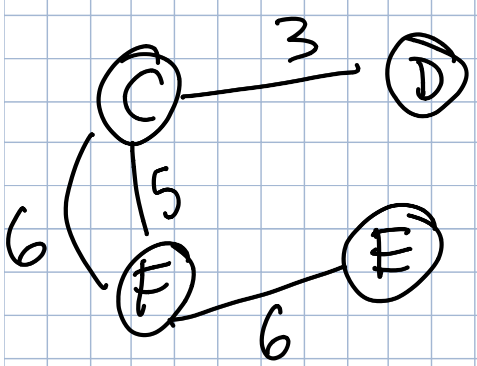
$\langle B,F,2,A,B\rangle$

A, B $\in$ in MST

< B, C, 6, B, C >

< C, D, 3, C, D >

< C, F, 5, C, F >



< D, F, 5, C, F >

< D, F, 6, B, C >

< D, F, 6, A, E >

< E, F, 6, A, E >

< C, D, 3, C, D >

< C, F, 5, C, F >

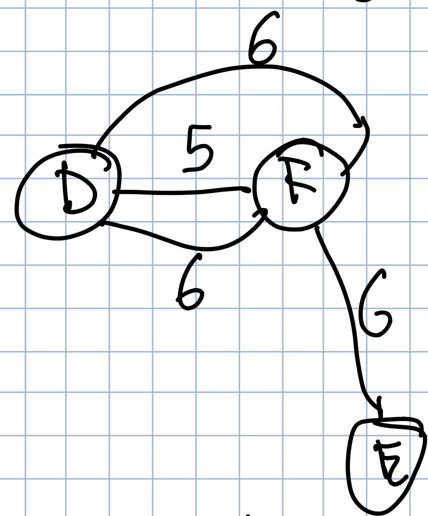
< C, F, 6, B, C >

< E, F, 6, A, E >

< C, D, 3, C, D >

C, D in MST

RelinkTo=D



Mi fermo quando il numero dei nodi entra in memoria.

ESERCIZIO 3

$S = (1, 3, 4, 6, 8, 9, 10)$

1	3	4	6	8	9	10

$l = 1$ $\quad$ $m = 4$ $\quad$ $r = 7$

$low = 1$ $\quad$ $S_m = 6$ $\quad$ $hi = 10$

$$[low + m - l \,,\, hi + m - r]$$

Codificare $S_m - low - m + l$

in $\lg_2(hi - low + l - r + 1)$ bit

6 viene codificato come:

$$6 - 1 - 4 + 1 = 2 \text{ in}$$

$$\lg_2(10 - 1 + 1 - 7 + 1) \text{ bit}$$

$$\lg_2 \lceil (4) \rceil \text{ bit} = 2 \text{ bit}$$

$$6 = 10$$

1	3	4

$l = 1$ $\quad$ $m = 2$ $\quad$ $r = 3$

$low = 1$ $\quad$ $S_m = 3$ $\quad$ $hi = 5$

8	9	10

↓

no bit emitted

Codificare

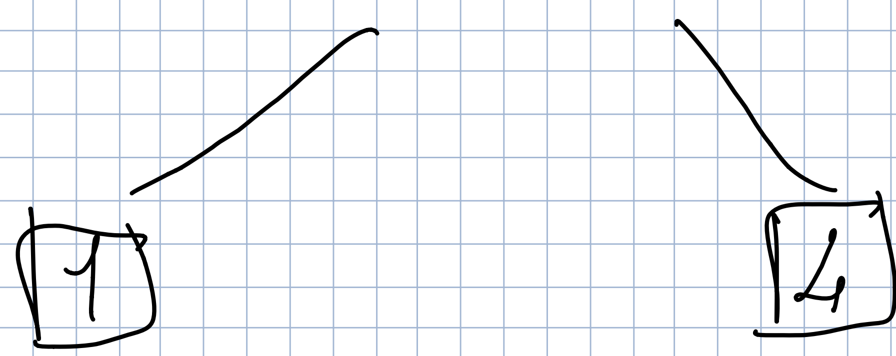
$3 - 1 = 2 + 1 = 1$

$sn - low - m + l$

in $\log_2 \lceil hi - low + l - r + 1 \rceil$ bit

$\log_2 \lceil 5 - 1 + 1 - 3 + 1 \rceil = \lceil \log_2 3 \rceil = 2$ bit

$3 = 01$



Vengono codificati 1 e 4 nello stesso modo

ESERCIZIO 4

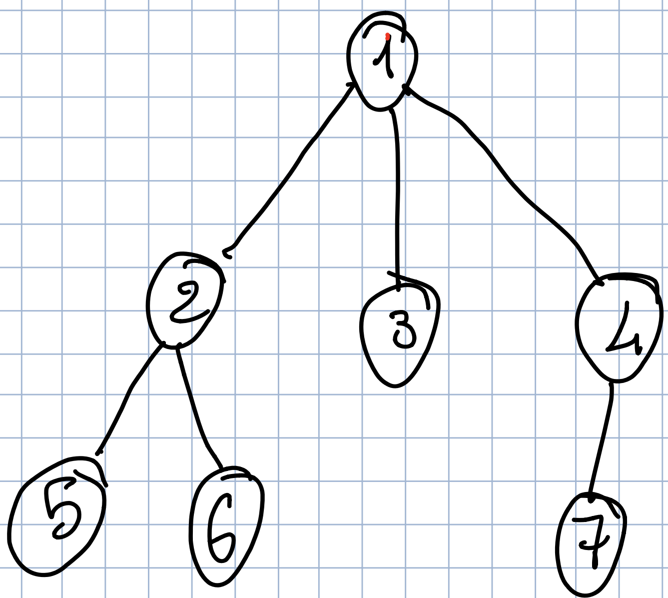
LCA → RMQ

Dato l'albero per trasformare il problema dell'LCA in RMQ si effettua un Euler tour dell'albero producendo un array Δ di lunghezza $2e$ dove e è il numero dei nodi.

Poi da Δ si crea D che è l'array in cui salviamo le profondità per ogni nodo visitato, la particolarità di D è che ogni valore è $+1$ o -1

rispetto al precedente.
D viene diviso in blocchi e per ogni blocco si calcola il minimo → Si crea l'array M
La struttura dati che viene usata, invece di prendere i possibili (i, j) che troviamo in M andiamo a considerare i range
$[i, i+2^L-1]$, $(j-2^L+1, j)$ e in questi range calcoliamo il minimo. In questo modo si riduce lo spazio necessario a
$O(n \lg n)$.
Per risolvere le query interne ad un blocco dobbiamo fare un altro passaggio,
Per ogni blocco D_k salviamo $\langle D_k[1], \Delta_k \rangle$
poi calcoliamo i possibili Δ_k che sono 2^{d-1}
dove d = len del blocco e prendiamo i possibili i, j all'interno di ogni blocco
si crea una tabella
$\langle i_0, j_0, pos\,min, \Delta_k \rangle$
che occupa $O(n)$ e mi fa calcolare in $O(1)$ l'lca.
$$\downarrow$$
$O(\sqrt{n}) \cdot O(\lg^2 n)$

possibili Δ   possibili i_0, j_0



ET: 1 2 5 | 2 6 2 | 1 3 1 | 4 7 4 | 1

D: 1 2 3 | 2 3 2 | 1 2 1 | 2 3 2 | 1

Devo dividere in blocchi e per ogni blocco calcolare il minimo
poi presi le posizioni di 5 e 7 andiamo a calcolare il minimo degli elementi compresi tra 5 e 7.

Se dividiamo in blocchi da 3

M =	1	4	7	10	13	posizioni
	1	2	1	2	1	numeri

Ora possiamo calcolare

	0	1	2
1	<1,1>	<1,1>	<1,1>
2	<2,4>	<1,7>	<1,7>
3	<1,7>	<1,7>	<1,7
4	<2,10>	<1,13	/
5	<1,13>	/	/

Pa: 1 1 1 ¦ 4 4 4 ¦ 7 7 7 ¦ 10 10 10 ¦ 13

Sa: 1 2 3 ¦ 4 6 6 ¦ 7 9 9 ¦ 10 12 12 ¦ 13

Forse per risolvere LCA(5,7) ora devo guardare Sa(1) e prendere l'elemento in posizione 3, per 7 devo prendere 12 nel 2° blocco

3 in entrambi i casi e poi considero l'array M con i minimi sull'array 2 e 3 (2 e 1)

Quindi lca = 1