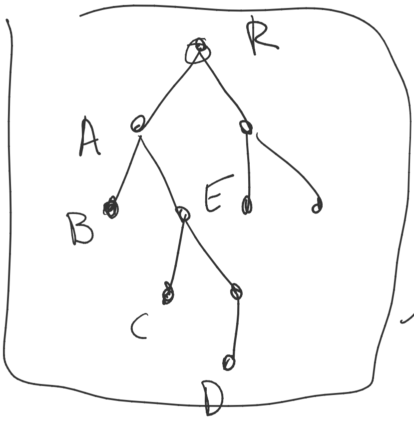
Es 3

Friday, March 27, 2020 10:57 AM

altezza di un nodo è la massima distanza dal nodo a una foglia. (contata in numero di archi)

Altezza di $T = h(R) = 4$



$h(A) = 3$

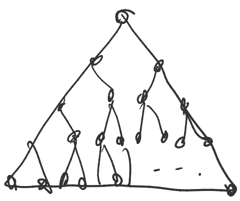
$h(E) = 2$

$h(R) = 4$

$h(B) = h(C) = h(D) = 0$

altezza di un albero = altezza della radice.

Albero binario completo: ogni nodo ha esattamente 2 figli, meno le foglie che hanno 0 figli



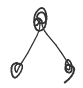
n h

$\boxed{Th}$: In un albero binario completo di altezza h il numero di nodi

- $n = N_h = 2^{h+1} - 1$
- di cui $F_h = 2^h$ foglie
- $I_h = 2^h - 1$ nodi interni

Prova: per induzione su h

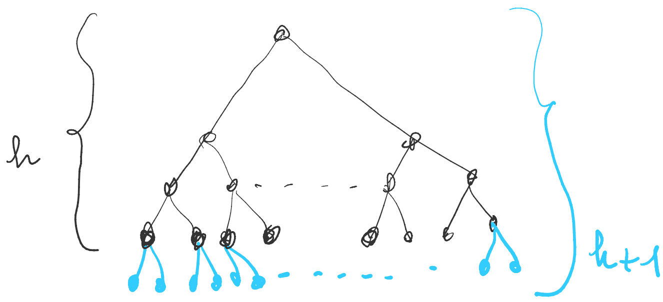
caso base; $h = 0$



$$\left.\begin{aligned} N_0 &= 2^{0+1} - 1 = 1 \\ F_0 &= 2^0 = 1 \\ I_0 &= 2^0 - 1 = 0 \end{aligned}\right\} \text{vero}$$

passo induttivo:

$$h \longrightarrow h+1$$



$$N_{h+1} = N_h + 2 \cdot F_h$$

$$= 2^{h+1} - 1 + 2 \cdot 2^h =$$

$$= 2^{h+1} - 1 + 2^{h+1} =$$

$$= 2^{h+2} - 1 \qquad \text{vero}$$

$$F_{h+1} = 2 F_h = 2^{h+1} \qquad \text{vero}$$

$$I_{h+1} = N_h = 2^{h+1} - 1 \qquad \text{vero}$$

Un albero di n nodi è binario completo ha altezza $h = \log(n+1) - 1$

$$n = 2^{h+1} - 1 \qquad n + 1 = 2^{h+1}$$

$$\log(n+1) = h + 1 \qquad \boxed{h = \log(n+1) - 1}$$

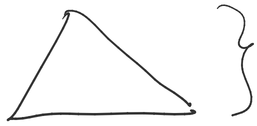
$$h = \Theta(\log n)$$

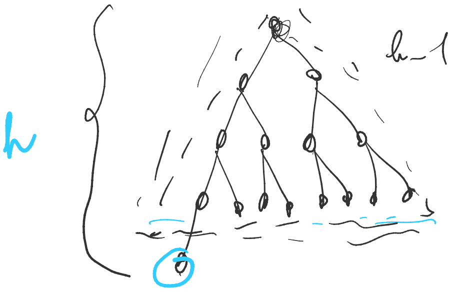
Studiamo la relazione tra n e h in

uno heap

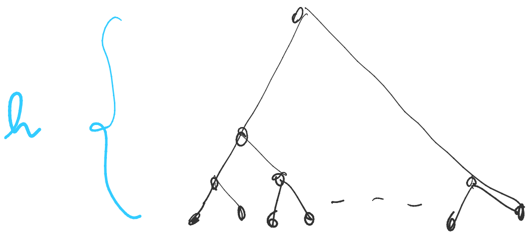
heap di altezza h è un albero binario completo fino a altezza $h-1$ e addossato a sinistra a altezza h.

se $n = 2^{h+1} - 1$





heap di altezza h con minimo numero di nodi



$$\left(2^{h-\cancel{1}+\cancel{1}} - \cancel{1}\right) + \cancel{1} \le n \le 2^{h+1} - \cancel{1}$$

↓ alb. binario completo

$$2^h \le n < 2^{h+1}$$

$$h \le \log n < h+1$$

$$\begin{cases} h \le \log n \\ h > \log n - 1 \end{cases} \qquad h = \lfloor \log n \rfloor \Rightarrow \Theta(\log n)$$

Dimostrare per induzione che in un heap vale la proprietà:

nodo i figlio sinistro $2i$

figlio destro $2i+1$

Simulazione di Heapsort:

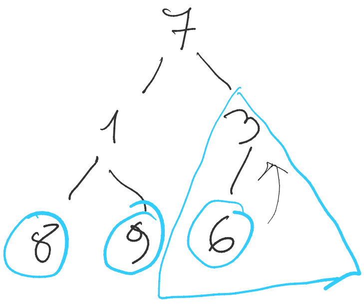
in loco: non uso spazio aggiuntivo

1) Costruisce un heap dell'array di input
2) Trovo il massimo e lo sposto in fondo n volte

Esempio

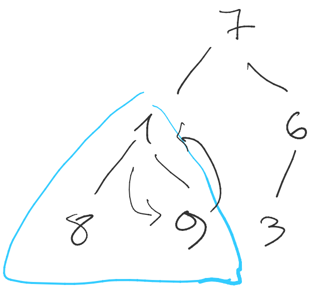
a

1	2	3	4	5	6
7	1	3	8	9	6



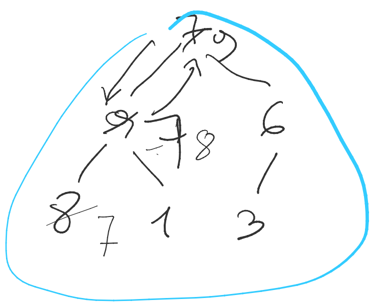
Build heap → bottom-up

Heapify (3)



7	1	6	8	9	3
	2		4	5	

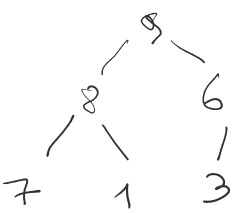
Heapify (2)



7	9	6	8	1	3

Heapify (1)

$\Theta(n)$



9	8	6	7	1	3

Heap

Sorting Heapsort ciclo

$i = 6$ estrai max → a[6]

3	8	6	7	1	9

heapify(1);

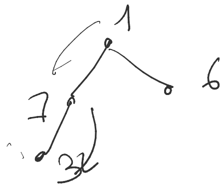
8	7	6	3	1	9

$i = 5$

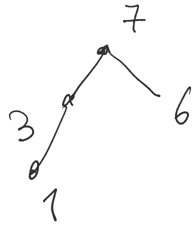
max → a[5]

1	7	6	3	8	9

heapify(0)



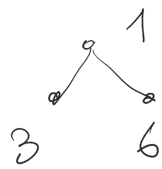
7	3	6	1	8	9



$i = 4$

max → a(4)

1	3	6	7	8	9



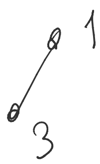
heapify(0)

$i = 3$

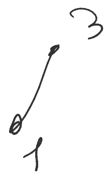
6	3	1	2	8	9

max → a(3)

1	3	6	7	8	9



heapify(0)



3	1	6	7	8	9

$i = 2$ max → a(2)

1	3	6	7	8	9	• 1

1	3	6	7	8	9

$$\Theta(n \log n)$$

Equazioni di ricorrenza

Algoritmo A $\Rightarrow T(n) = 7\,T\left(\frac{n}{2}\right) + n^2$

Algoritmo A' $\Rightarrow T(n) = \boxed{a}\,T\left(\frac{n}{4}\right) + n^2$

Determinare il max valore di a per cui A' è più veloce di A

Alg. A: Teorema dell'esperto

$a = 7 \quad b = 2 \quad n^{\log_2 7} \cong n^{2.8} \quad f(n) = n^2$

$f(n) = n^2 = O\left(n^{2.8-\varepsilon}\right)$ vero per $\varepsilon \leq 0.8$

$\boxed{T(n) = \Theta\left(n^{2.8}\right)}$ caso 1

Alg. A': $a \quad b = 4 \quad n^{\log_4 a} \; : \; f(n) = n^2$

$a < 16 \quad n^{\log_4 a} < n^2$

$f(n) = \Omega\left(n^{\log_4 a + \varepsilon}\right)$ vero $\log_4 a + \varepsilon \leq 2 \Rightarrow$

$\varepsilon \leq 2 - \log_4 a$

caso 3

$\boxed{T(n) = \Theta\left(n^2\right)}$ se vale la proprietà di regolarità

$$a f\left(\frac{n}{b}\right) \leq c f(n)$$

$$a\left(\frac{n}{4}\right)^2 \leq c n^2 \qquad \frac{a n^2}{16} \leq c n^2 \text{ vero}$$

$$c \geq \frac{a}{16} \qquad \text{vero}$$

$\boxed{a = 16}$

$$n^{\log_4 16} = n^2$$

$\boxed{\text{Caso } 2}$ $$T(n) = \Theta\left(n^{\log_4 16}\right) = \Theta(n^2 \log n)$$

conviene A'

$a > 16$

$$f(n) = n^2 = O\left(n^{\log_4 a - \varepsilon}\right) \qquad \log_4 a - \varepsilon \geq 2$$

$$-\varepsilon \geq 2 - \log_4 a$$

$$\text{vero per } \varepsilon \leq \log_4 a - 2$$

Caso 1 $$T(n) = \Theta\left(n^{\log_4 a}\right)$$

$$A' \rightarrow \quad n^{\log_4 a} \leq n^{\log_2 7} \leftarrow A$$

$$\log_4 a \leq \log_2 7$$

$$\boxed{\log_b a = \frac{\log_c a}{\log_c b}}$$

$$\frac{\log_2 a}{\log_2 4} \leq \log_2 7$$

$$\log_2 a \leq 2 \log_2 7 \leq \log_2 7^2$$

$$2^{\log_2 a} \le 2^{\log_2 7^2}$$

$$\boxed{a \le 49}$$

$a = 49$

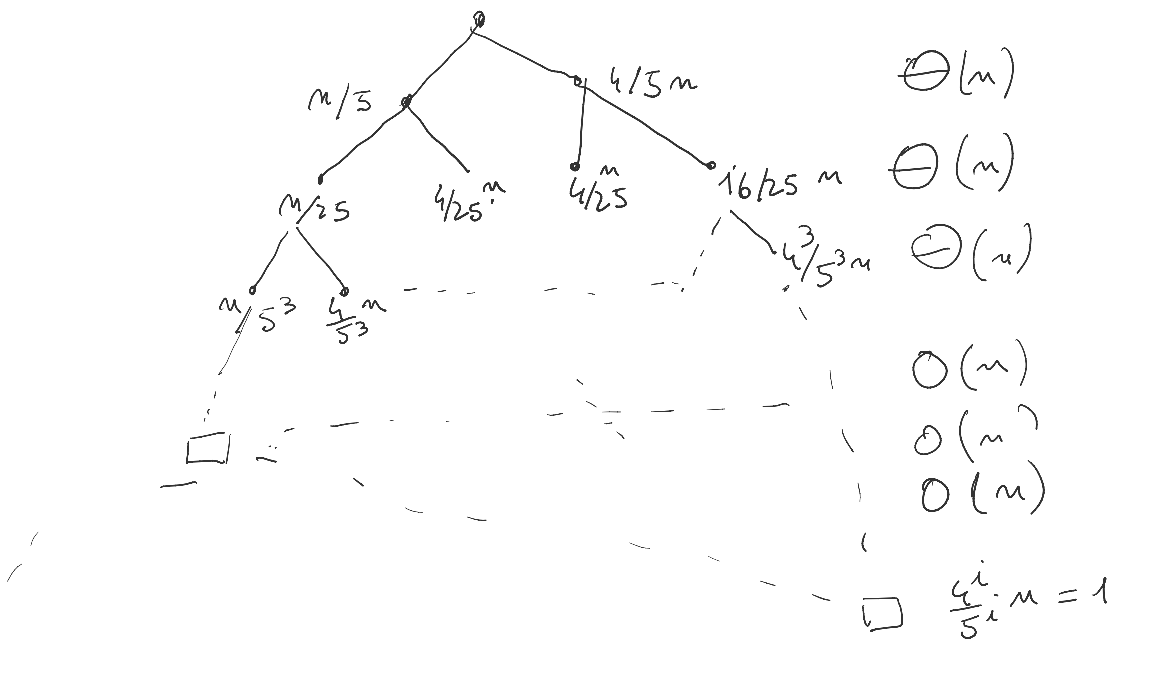
$A = A'$

max valore di a per cui conviene A' è $a = 48$

$$\begin{cases} T(n) = \Theta(1) & \text{se } n = 1 \\ T(n) = T\left(\frac{n}{5}\right) + T\left(\frac{4}{5}n\right) + \Theta(n) & \text{se } n > 1 \end{cases}$$

$$T(n) = a\,T\left(\frac{n}{b}\right) + \Theta(n)$$

$$T(n) \le 2\,T\left(\frac{4}{5}n\right) + \Theta(n)$$ teorema esperto



$\left(\frac{4}{5}\right)^i n = 1$ $\quad n = \left(\frac{5}{4}\right)^i$ $\quad i = \log_{5/4} n$

n° livelli dell'albero di ricorsione

è $= i$

in ogni livello $O(n)$ operazioni

per la ricombinazione

$$T(n) = O(n \log n)$$