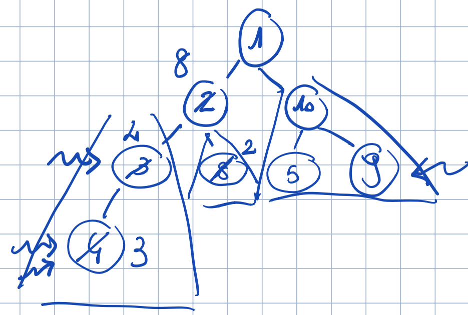
Esempio

1	2	10	3	8	5	9	4

foglie



BuildHeap

Heapify (A, i)

% Precondizioni A[2i] A[2i+1] sono radici di Max-Heap

% Postcondizioni A[i] radice di Max-Heap

Uso Heapify (A, 4)

posizione

4 3

1	2	10	~~3~~	8	5	9	~~4~~

radici di Max Heap

Uso Heapify (A, 3)

4 3

1	2	10	~~3~~	8	5	9	~~4~~

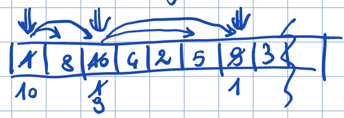
Uso Heapify (A, 2)

4 3

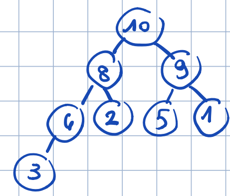
1	~~2~~	10	~~3~~	~~8~~	5	9	~~4~~

8 2

Uso Heapify $(A, 1)$



Heap



$\Theta(n)$

BuildHeap (A) { % vettore $\boxed{\cancel{O(n \log n)}}$

$A.heapsize \leftarrow A.length$ ⇝ $\Theta(1)$

for $(i \leftarrow \left\lfloor \frac{A.length}{2} \right\rfloor$ downto $1)$ {

⇝ Heapify (A, i) ⇝ $O(\log n) \cdot \Theta(n)$

}

}

Correttezza

BuildHeap(A) termina con A Max-heap

Invariante ciclo for

All'inizio dell'iesima iterazione del ciclo for $A[i+1], A[i+2] \dots A[A.length]$ sono radici di Max-heap

Dim

Per induzione su i

BASE $i = \left\lfloor \frac{A.length}{2} \right\rfloor$

$A[i+1]$ ... $A[A.length]$ Sono foglie quindi radici di Max-Heap

PASSO

Hp Ind) Tutto funziona fino all'inizio della i-esima iterazione

All'inizio dell'i-esima $A[i+1]$ ....

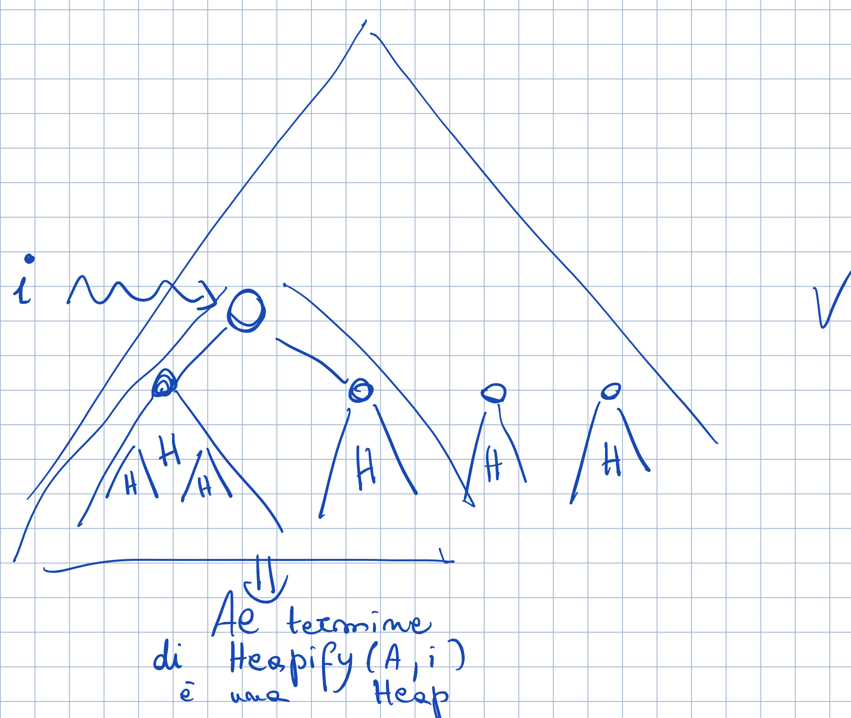
radici di max-heap

Eseguo Heapify(A,i)

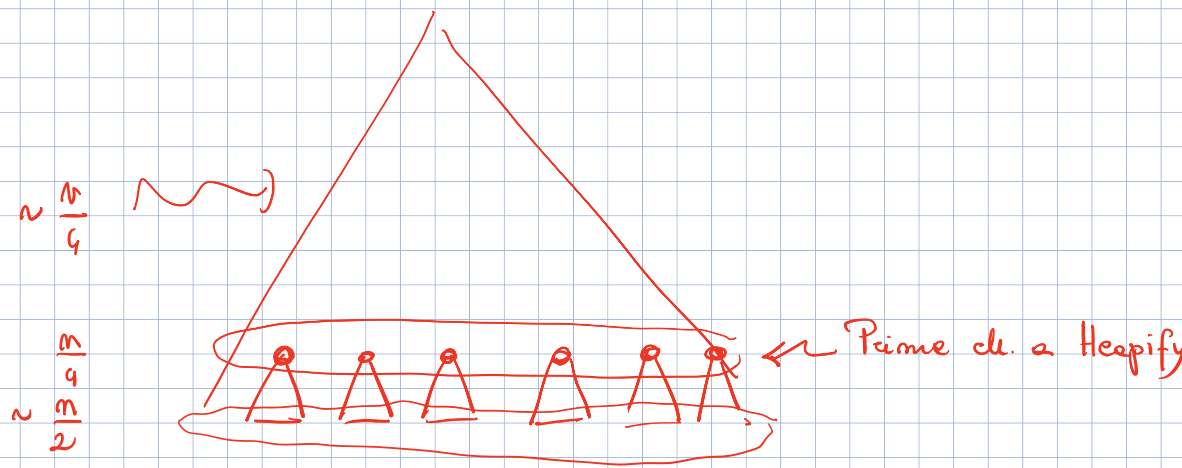
Ts) All'inizio dell'iterazione $i-1$

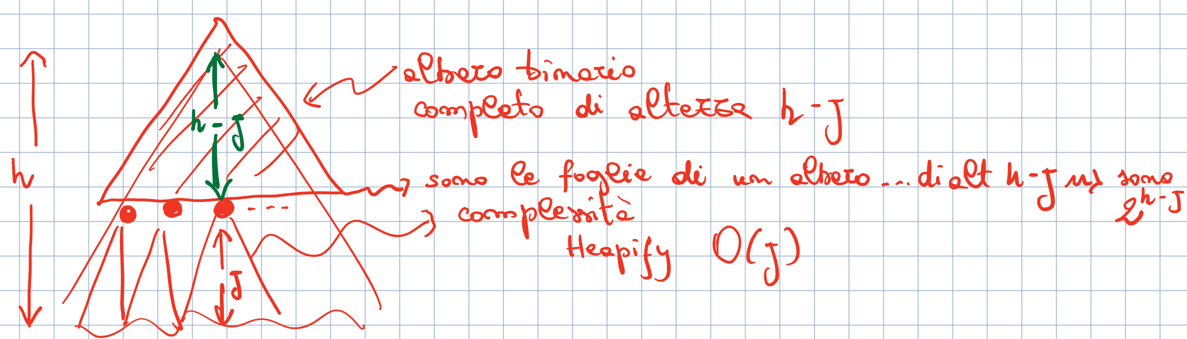
$A[i]$, $A[i+1]$ ...

radici di max-heap



Complessità





2^{h-J} modi che hanno un costo di Heapify $\cancel{O(J)}$ $C \cdot J$

$\sum$ Costo di BuildHeap

h altezza della Heap

$2^h = n$ modi

$$T(h) = \sum_{J=0}^{h} 2^{h-J} \cdot C \cdot J =$$

$$C \sum_{J=0}^{h} J \cdot \frac{2^h}{2^J} = C \cdot 2^h \underbrace{\sum_{J=0}^{h} \frac{J}{2^J}}_{= C'} \leq C \cdot C' \cdot 2^h$$

$x = \frac{1}{2}$

$$2 = \frac{1/2}{(1-\frac{1}{2})^2}$$

Somiglia alla serie geometrica

$$T(h) = O(2^h)$$

$$T(n) = O(n)$$

$$T(n) = \underset{\text{banale}}{\Omega(n)}$$

$$\boxed{T(n) = \Theta(n)}$$

$$\sum_{J=0}^{k} J\left(\frac{1}{2}\right)^J \leq c'$$

$$\sum_{i=0}^{n} x^{i} = \frac{x^{n+1} - 1}{x - 1} \qquad\qquad \sum_{i=0}^{n} i \cdot x^{i}$$

$\Downarrow$ derivata di entrambi rispetto ad x

$$\sum_{i=0}^{n} i\, x^{i-1} = \frac{(n+1)x^{n}(x-1) - (x^{n+1}-1)}{(x-1)^2} \overset{\S}{=} \frac{(n+1)x^{n+1} - (n+1)x^{n} - x^{n+1} + 1}{(x-1)^2}$$

$$\sum_{i=0}^{n} i \cdot x^{i-1} \overset{\S}{=} \frac{n x^{n+1} - (n+1)x^{n} + 1}{(x-1)^2}$$

$\Downarrow$ moltiplico entrambi per x

$$\sum_{i=0}^{n} i \cdot x^{i} = \frac{n x^{n+2} - (n+1)x^{n+1} + x}{(x-1)^2}$$

$$\sum_{i=0}^{\infty} i \cdot x^{i} = \lim_{n\to\infty} \sum_{i=0}^{n} i \cdot x^{i} = \lim_{n\to\infty} \frac{n\, x^{n+2} - (n+1)x^{n+1} + x}{(x-1)^2}$$

$x > 1$: ∞

$x < 1$: $\dfrac{x}{(x-1)^2}$

$$\sum_{i=0}^{n} i \cdot x^{i} \leq \sum_{i=0}^{\infty} i \cdot x^{i} \underset{x<1}{=} \frac{x}{(x-1)^2}$$

$\boxed{O(n \cdot \log n)}$

HeapSort (A) {

⇝ BuildHeap (A) $\qquad O(n)$

⇒ for (i ← A.length downto 2) { $\qquad O(n)$

$\left.\begin{array}{l}\text{swap } (A, 1, i) \\ A.\text{heapsize} \leftarrow A.\text{heapsize} - 1 \\ \text{Heapify } (A, 1)\end{array}\right\}$ Extract ← $O(\log n)$

}

}

Exempio

| 1 | 2 | 10 | 3 | 8 | 5 | 9 | 4 | $\quad$ A

⇓ BuildHeap

| ~~1~~ | 8 | ~~10~~ | 4 | 2 | 5 | ~~8~~ | 3 |

10 $\quad$ ~~1~~ 9 $\quad$ 1

| 10 | 8 | 9 | 4 | 2 | 5 | 1 | 3 | $\quad$ A $\quad$ Max Heap

↑ i

9 $\quad$ 5 $\quad$ 3

| ~~3~~ | 8 | ~~8~~ | 4 | 2 | ~~5~~ | 1 | 10 |

⇑

| 9 | 8 | 5 | 4 | 2 | 3 | 1 | 10 |

H $\quad$ ≤

8 $\quad$ 4

| ~~1~~ | ~~8~~ | 5 | ~~4~~ | 2 | 3 | 9 | 10 |

1

⇑

H $\quad$ ≤

Correttezza

HeapSort (A) termina con A ordinato

Invariante

All'inizio dell'i-esima ...

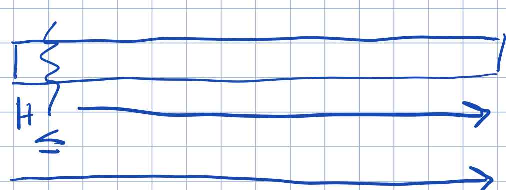
- $A[1..i]$ max-heap
- $A[i+1..A.length]$ è ordinato
- $A[1..i] \leq A[i+1..A.length]$

Dim

Per induzione su i

- - -

Invariante con $i = 1$



Caso migliore, caso peggiore HeapSort?

$O(n \log n)$ è sovrastimato?

È in-place?   È stabile?