Titolo nota 17/03/2016

Proprietà: In un heap i nodi interni sono in $A[1] \dots A[\lfloor \frac{n}{2} \rfloor]$

foglie $A[\lfloor n/2 \rfloor + 1] \dots A[n]$

$A[i]$

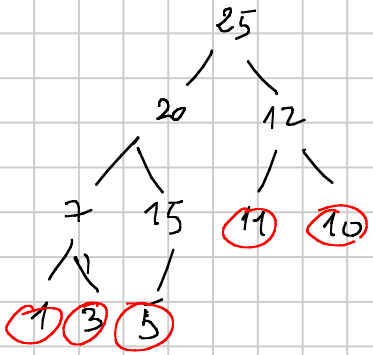
i

$parent(i) = \lfloor i/2 \rfloor \qquad i \geq 2$

$left(i) = 2i$

$right(i) = 2i + 1$

MAX-HEAP



1	2	3	4	5	6	7	8	9	10
25	20	12	7	15	11	10	1	2	5

A { i = pari; i = dispari

$parent(left(i)) = parent(2i) = \lfloor \frac{2i}{2} \rfloor = i$

$parent(right(i)) = parent(2i+1) = \lfloor \frac{2i+1}{2} \rfloor =$

$\lfloor i + \frac{1}{2} \rfloor = i$

Titolo nota 17/03/2016

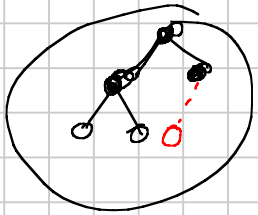
Prova induzione su n

base: $n = 1$ • $\left\lfloor \frac{1}{2} \right\rfloor = 0$ nodi interni vera

$\left\lfloor \frac{1}{2} \right\rfloor + 1 = 1$ foglie

passo induttivo: $n-1 \to n$

1) caso $n-1$ pari $\to n$ dispari



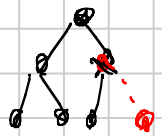
il numero di foglie rimane invariato

il numero di nodi interni aumenta di 1.

$$\left\lfloor \frac{n}{2} \right\rfloor = \left\lfloor \frac{n-1}{2} \right\rfloor + 1 \quad \text{vero}$$

Titolo nota 17/03/2016

2) caso 2: $n-1$ dispari n pari



nodo interno più a destra rimane incriato

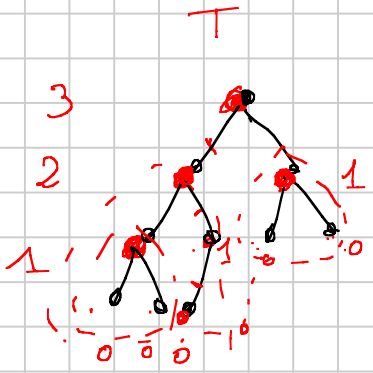
$$\left\lfloor \frac{n-1}{2} \right\rfloor = \left\lfloor \frac{n}{2} \right\rfloor$$ vero

Titolo nota 17/03/2016

Proprietà: In un heap di n nodi e altezza $\lfloor \log n \rfloor$ i nodi di altezza h sono: $0 \leq h \leq \lfloor \log n \rfloor$

$$n_h = \left\lceil \frac{n}{2^{h+1}} \right\rceil$$

$$n_2 = \frac{10}{2^3} = \left\lceil \frac{10}{8} \right\rceil = 2$$



altezza di un nodo = altezza del sottoalbero di cui è radice

$h(T) = h(\text{radice}) = 3$

$$n_1 = \left\lceil \frac{10}{2^2} \right\rceil = 3 \qquad n_3 = \left\lceil \frac{10}{2^4} \right\rceil = 1$$

Titolo nota 17/03/2016

Costruzione Heap

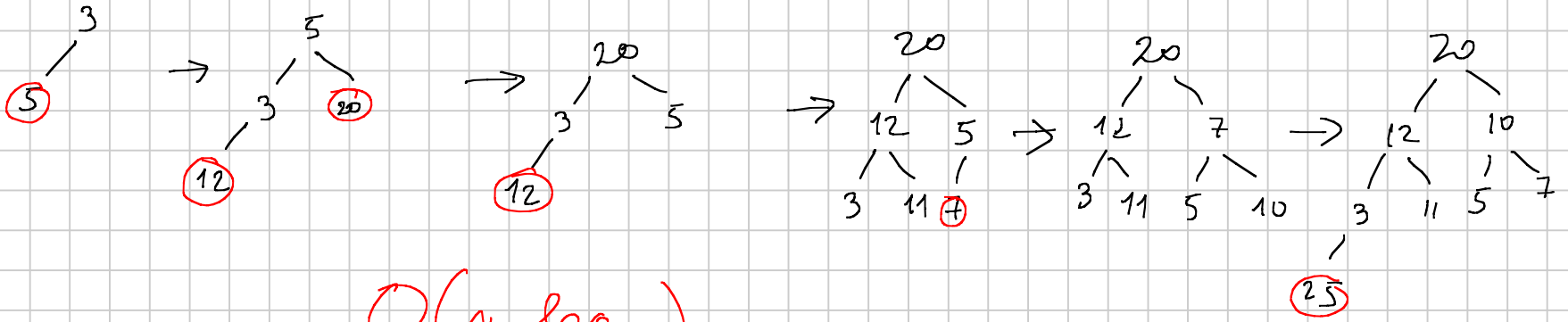
Input : array A

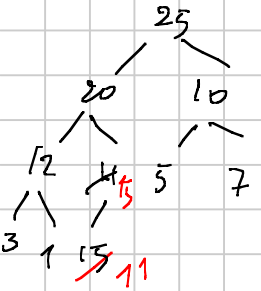
Output : max-heap

A

3	5	20	12	11	7	10	25	1	15

algoritmo banale : n operazioni HEAP-INSERT $O(\log n)$

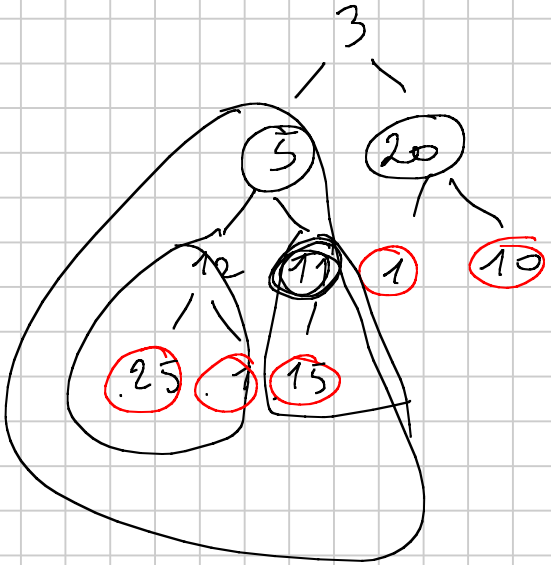




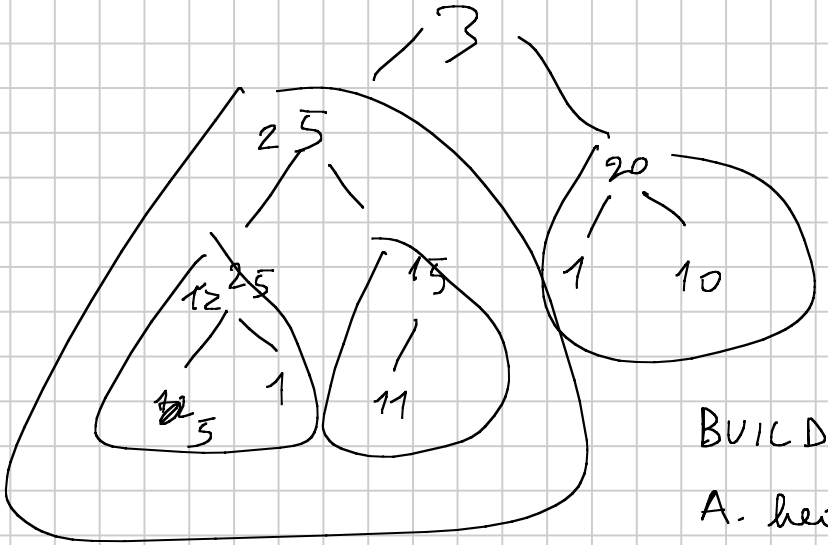
$O(n \log n)$

Titolo nota 17/03/2016

Migliore



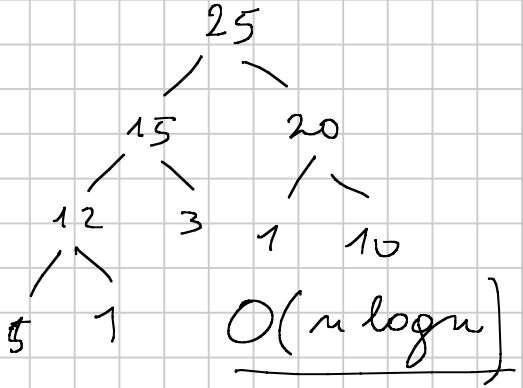
BUILD-MAX-HEAP (A)



"Per tutti i nodi interni"

$O(\log n)$

HEAPIFY (A, i)



$O(n \log n)$

BUILD-MAX-HEAP(A):

A.heapsize = A.length;

for i = $\lfloor \frac{A.length}{2} \rfloor$ downto 1

MAX-HEAPIFY (A, i)

Titolo nota 17/03/2016

il numero di nodi di altezza h è $n_h \leq \left\lceil \frac{n}{2^{h+1}} \right\rceil$ costo heapify $O(h)$

$$\sum_{h=0}^{\lfloor \log n \rfloor} \left\lceil \frac{n}{2^{h+1}} \right\rceil O(h) = O\left(n \sum_{h=0}^{\log n} \frac{h}{2^h}\right) = O(n)$$

↑ n° nodi altezza h

↖ MAX-HEAPIFY

$$\sum_{h=0}^{\infty} \frac{h}{2^h} = \frac{1/2}{(1-1/2)^2} = 2$$

$\sum h \cdot x^h$ A 8 appendice A

BUILD-HEAP costa $O(n)$

$O(n)$

Titolo nota 17/03/2016

HEAP-SORT (A)

costruisci Heap n volte

BUILD-MAX-HEAP(A); $O(n)$

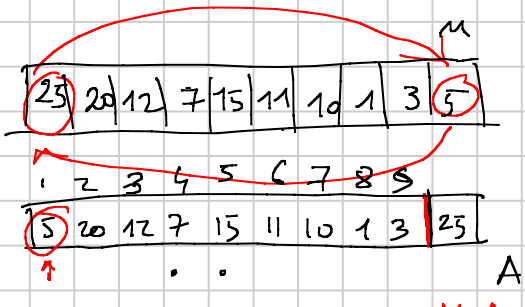
for i = A.length downto 2

$\Theta(1)$ "scambio A[i] con A[1]"

$\Theta(1)$ A.heapsize = A.heapsize − 1;

$O(\log n)$ MAX-HEAPIFY (A, 1);

$O(n \log n)$ ottimo

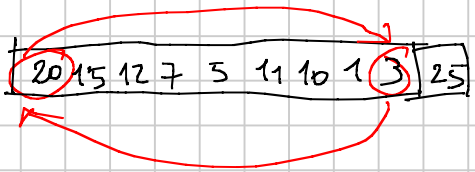


A.heapsize = n − 1

MAX HEAPIFY (A, 1)

20 5 12

20 15 12 7 5



3 15 12 7 5 11 10 1 20 25

MAX-HEAPIFY (A, 1)

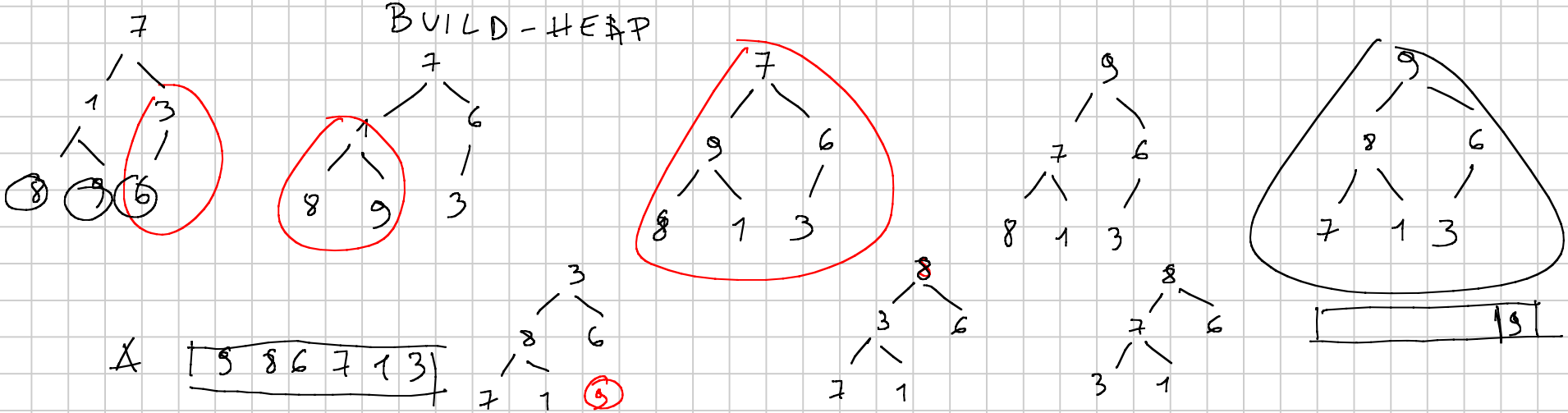
Titolo nota 17/03/2016

	caso medio tempo		spazio
MERGESORT	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n) + \Theta(\log n) = \Theta(n)$
QUICKSORT	$O(n \log n)$	$O(n^2)$	$O(n)$ caso peggiore $O(\log n)$ caso medio
HEAPSORT	$O(n \log n)$	$O(n \log n)$	$O(\log n)$

pila delle ricorsive

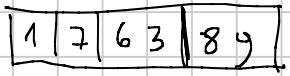
A | 7 1 3 | 8 9 6 |

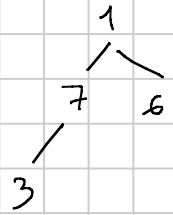
BUILD-HEAP

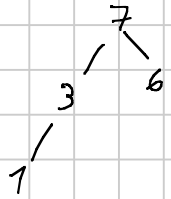


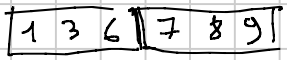
A | 9 8 6 7 1 3 |

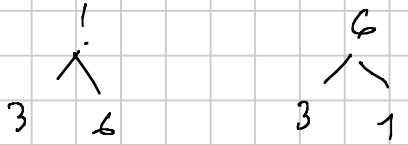
Titolo nota 17/03/2016

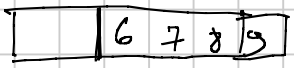


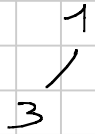


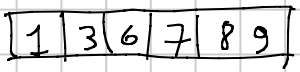












- Coda priorità
  - Inserisce
  - Estrazione del Max
  - ~~Riceve~~

- Base per nuovo algoritmo ottimo di ordinamento