Lezione 31 marzo

Tuesday, March 31, 2020 4:00 PM

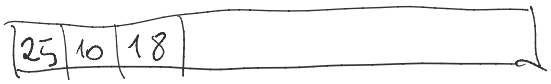
Heap come coda di priorità

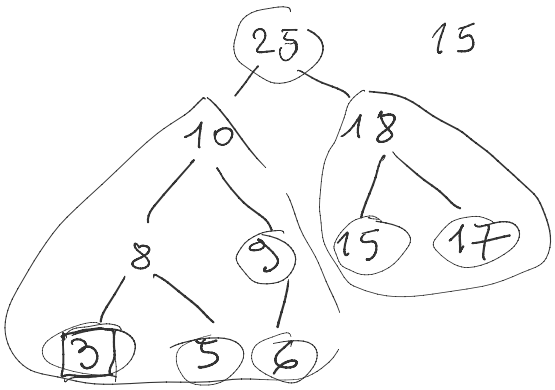
Heapsort ottimo $O(n \log n)$

ottimo come spazio

in loco

Max-Heap





Ricerca (a, k)

caso ottimo: $k >$ del max dell'heap

ricerca senza successo $\Theta(1)$

caso pessimo: $k <$ nodo $\forall$ nodo visitato

$\Theta(n)$

Ricerca $O(n) \equiv$ ricerca sequenziale

- estrazione max $\Theta(\log n)$ ricerca
- inserzione $\Theta(\log n)$

Input: heap di massimo

Output: il minimo

dove si trova il minimo?

il minimo si trova su una foglia.

Prova per assurdo

Prova per assurdo

Dimostriamo che in un heap il numero di foglie $= \lceil \frac{n}{2} \rceil$

Per induzione su n

Base $n = 1$

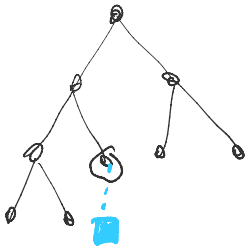
$f = 1 \quad \lceil \frac{1}{2} \rceil = 1$ vera

Ipotesi induttiva $n \to n+1$

$f = \lceil \frac{n}{2} \rceil$

2 casi

1°) caso la nuova foglia è un figlio sinistro



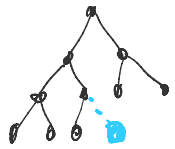
n è dispari
$n+1$ è pari

la nuova foglia annulla una foglia dell'albero prec.

n foglie $= \lceil \frac{n}{2} \rceil = \lceil \frac{n+1}{2} \rceil$ vera

2° caso: n pari $n+1$ dispari

la nuova foglia è un figlio destro

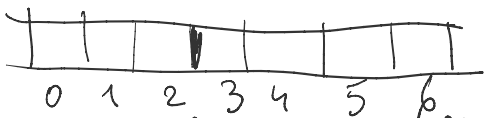


la nuova foglia è il figlio destro

il n° di foglie = n° foglie precedenti + 1 = $\left\lceil \frac{n}{2} \right\rceil + 1$

$\left\lceil \frac{n+1}{2} \right\rceil$ $\quad n = 10 \quad \left\lceil \frac{10+1}{2} \right\rceil = 6$

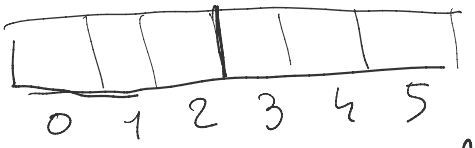
Ricerca del minimo in Max-heap



$\left\lceil \frac{n}{2} \right\rceil$ foglie

$\left\lfloor \frac{n}{2} \right\rfloor$ nodi interni

$n = 7$ → 4 foglie dalla pos. 3



$n = 6$ 3 foglie dalla pos. 3

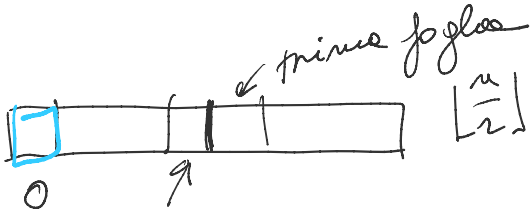
$\left\lfloor \frac{n}{2} \right\rfloor$ è la posizione della foglia più a sinistra nell'array $a[0..n-1]$

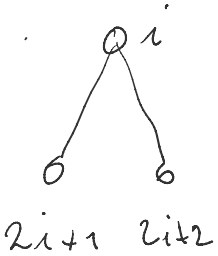
```
Min_Max_Heap (a);
min = a[⌊n/2⌋];
for ( i = ⌊n/2⌋ + 1; i < n; i++) {
    if (a[i] < min) scambia a[i] : min;
} return min;
```

$n/2$ $\qquad \Theta(n)$

Verificare se un array di n el. è Max heap.

```
IsMHeap?(a):
for(i = 0; i ≤ ⌊n/2⌋ - 1; i++){
  if(a[i] < a[2i+1]) return false;
  if((2i+2 < n) && a[i] < a[2i+2])
      return false;
} return true;
```



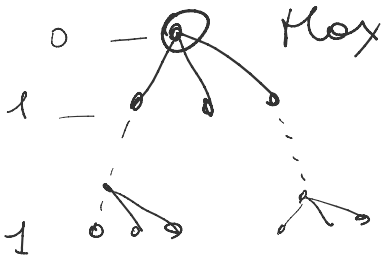


figlio sinistro esiste

esiste figlio destro?

$O(n)$

Max-heap ternario



$0 \leq i < h$

3^i nodi per livello i

- livello h; il n° di nodi

$$1 \leq n_h \leq 3^h$$

- tutte le foglie sono addossate e sin.

- tutte le foglie sono ordinate e sin.
- il valore di un nodo è maggiore di quello dei figli

Descrivere un'implementazione efficiente mediante $\forall$ nodo i:

i - primo figlio terzo figlio
- secondo figlio padre

oper: - Enqueue
- Dequeue

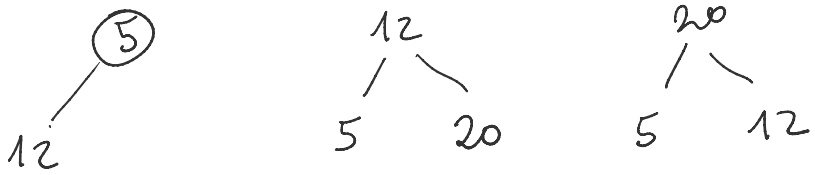
Strutture complesse.

Selezione del k-esimo elemento utilizzando la struttura dati heap.

- Progettare una soluzione $O(n \log n)$
- " " $O(n + k \log n)$
- " " $O(n \log k)$

Selection (con modifica di QuickSort)
$O(n)$ in media

Costruzione dell'heap n. oper. di Enqueue



$O(n \log n)$

2) Buildheap  $O(n)$

1) Ordinare l'array con HeapSort e restituire $a[k]$ $\boxed{O(n \log n)}$

2) - costruisce un heap di minimo $O(n)$
- k estrazioni dall'heap - Dequeue k volte $k \log n$

$O(n + k \log n)$

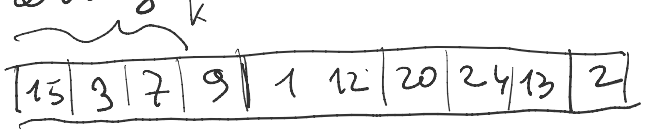
Spazio $\Theta(1)$

$k = O(n)$

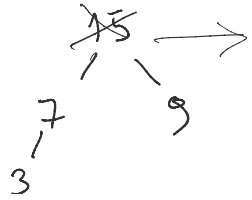
$k = n/2$

3) $n \log k$

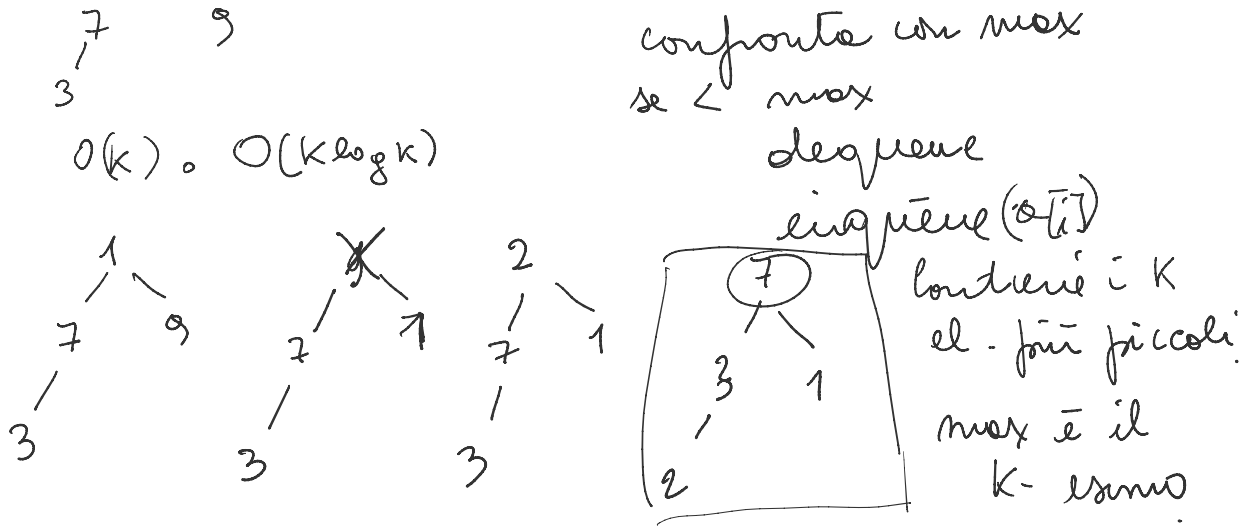
si costruisce un heap di massimo di k elementi prendendo i primi k elementi dell'array



$k = 4$



$\forall$ $a[i]$
confronto con max
se > max



$O(k \log k + n \log k)$ $\quad \Theta(k)$ spazio

```
Selection(a, k):
H = nuovo array di k el.
for (i = 0; i<k; i++)
    enqueue(H, a[i]);
for (i = k; i < n; i++) {
    if (a[i] < H[0]) {
        Dequeue(H);
        Enqueue(H, a[i]);
    }
}
return(H[0]);
```

$O(k \log k)$

$O(n \log k)$

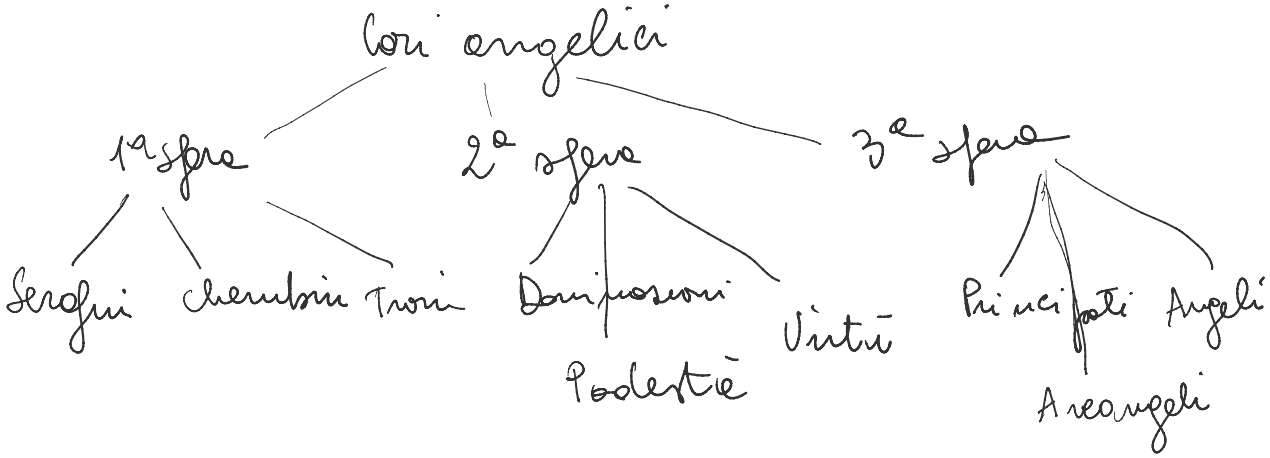
10-17 Vacanze di Pasqua

?

Alberi

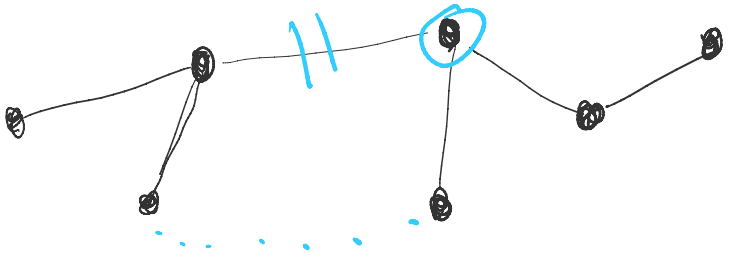
1) Struttura algoritmo (Divide et Impera)
2) Alberi di decisione (studiare limiti inferiori)
3) Alberi di ricorsione (risolvere equazioni ricorrenze)

Alberi per rappresentare l'informazione



Un grafo connesso privo di cicli

albero libero



n nodi
n-1 archi

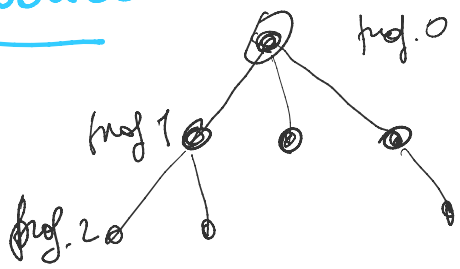
foresta

alberi con radice

fig. 0

alberi con radice

profondità distanza da radice



la radice induce un ordinamento per livelli

altezza di un nodo = distanza della foglia più lontana

altezza di un albero = altezza della radice

botanica alberi genealogici

parentela è definita su un albero genealogico = distanza in numero di archi