Introduzione agli algoritmi
Prova di esame del 19/9/2016
Prof.sse E. Fachini - R. Petreschi

Parte prima

1) Si dimostri il teorema sulla limitazione inferiore per il tempo asintotico di esecuzione nel caso peggiore di un algoritmo di ordinamento nel modello basato sui confronti.

2) a) Dimostrare la verità o la falsità delle seguenti asserzioni:
$2^{n+1} = O(2^n)$
$2^{2n} = O(2^n)$
$f(n) = O(g(n))$ implica $2^{f(n)} = O(2^{g(n)})$

b) Se si dimostra che un algoritmo ha tempo di esecuzione $\Theta(n^2)$ nel caso migliore, posso dedurne che nel caso peggiore terminerà in $O(n^2)$ passi?

3) Un albero d-ario è definito come un albero binario, ma il massimo numero di figli che può avere è $d \geq 2$ e non 2. Un albero d-ario quasi completo è un albero d-ario in cui tutti i livelli sono pieni, tranne l'ultimo nel quale le foglie sono a sinistra. Supponiamo che le chiavi dei nodi siano intere. Un albero d-ario quasi completo può essere rappresentato in memoria in un array A come segue:

La chiave della radice è in A[1], quelle dei figli sono in A[2] ,…, A[d+1] e così via come nel caso binario.

Il padre di un nodo si individua usando la procedura:

D-ARY-PARENT(i) return $\lfloor (i - 2)/d + 1 \rfloor$.

Il j-simo figlio di un nodo i si individua usando la procedura:

D-ARY-CHILD(i, j) return $d(i - 1) + j + 1$

L'altezza dell'albero d-ario quasi completo è $\Theta(\log_d n)$.

Supponiamo ora che le chiavi soddisfino la proprietà del max-heap, cioè che ogni nodo abbia chiave maggiore o uguale a quella dei suoi figli.Si implementi le si analizzi dal punto di vista de tempo di esecuzione asintotico l'operazione di estrazione, con eliminazione, del massimo d-ARY-HEAP-EXTRACT-MAX.

Introduzione agli algoritmi
Prova di esame del 19/9/2016
Prof.sse E. Fachini - R. Petreschi

Parte seconda

1. Disegnare l'albero binario, con chiavi alfabetiche, le cui visite inordine e preordine sono rispettivamente **GDHBAECJIKF** e **ABDGHCEFIJK**.

2. Si determini il tempo di esecuzione asintotico T(n) della seguente funzione:

```
analizzami(A,i,j)
n = j - i+1
c = 1
m = (i + j +1)/2
k = (i+m+1)/2
h = (m+j+1)/2
d = m
while d > 1 do
      for j = 1 to n do c++
      d = d - 2
if n > 1 then
         return analizzami(A,i,k) + analizzami(A,k+1,m) + analizzami(A,m+1,h) + analizzami(A,h+1,j)
```

3. Sia T un albero binario radicato in cui ogni nodo ha un valore intero come chiave. Definiamo costo di un cammino radice-foglia la somma delle chiavi dei nodi che compongono il cammino. Il diametro di T è il massimo fra i costi dei cammini radice foglia in T. Si progetti un algoritmo per trovare il diametro di un albero T.