Introduzione agli algoritmi
Prova di esame del 7/6/2016
Prof.sse E. Fachini - R. Petreschi

B

1. (Max punti 10) a. Dimostrare che l'altezza h di un dato albero binario B di n vertici è un valore compreso fra log n e n (log n ≤ h ≤ n). L'altezza è definita come il massimo numero di archi (o rami) di un cammino radice-foglia.
   b. Quali sono i limiti inferiore e superiore per la lunghezza del cam mino minimo radice-foglia, mh, in un albero binario?
   c. Quali sono il minimo e il massimo numero di nodi in un albero quasi completo di altezza h?

   Sol. si veda libro di testo e lucidi

2. (Max punti 8) Si considerino le seguente affermazioni:
   A. se g(n) = O(f(n)) e h(n) = O(f(n)) allora g(n) = O(h(n)).
      E' vera o falsa?
      Se ne dimostri la verità con una dimostrazione generale oppure se ne dimostri la falsità producendo tre funzioni che la contraddicono.

   Sol. E' falsa. Infatti g ed h sono limitate in crescita entrambe da f, ma non sappiamo come crescono l'una verso l'altra. Se si prende $g(n) = n^2$, $h(n) = n \lg n$ e $f(n) = n^2$, è vero che $n^2 = O(n^2)$ e $n \lg n = O(n^2)$ ma $n^2 \neq O(n \lg n)$

   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 $\Omega(n^2)$ passi?

   Sol. L'affermazione è vera, il caso migliore fornisce un limite inferiore al tempo di esecuzione in ogni caso.

3. (Max punti 12) Si definisca una funzione Heap-Sost-Key(H,i,k) che sostituisce la chiave presente nella posizione i del Max-Heap H con k, ristabilendo la proprietà di Max-Heap eventualmente violata dal-

Introduzione agli algoritmi
Prova di esame del 7/6/2016
Prof.sse E. Fachini - R. Petreschi

B

la modifica. Si valuti il tempo di esecuzione asintotico dell'algoritmo presentato.

Sol. La nuova chiave k potrebbe essere maggiore del padre di A[i] o minore di uno o entrambi i figli di A[i], per ripristinare la proprietà di essere maxheap per A bisogna quindi considerare i due casi, come nel caso della cancellazione di un elemento in un max-heap.
Il tempo di esecuzione nel caso peggiore O(lg n), perché al più i=1 e la proprietà viene ripristinata scambiando padri e figli fino a una foglia oppure anche se i > n/2, e quindi è una foglia e la proprietà viene ripristinata scambiando padri e figli risalendo fino alla radice.

```
Heap-Sost-Key(H,i,k)
input: A è un max-heap, 1≤i≤A.heap-size
output: A è il max-heap in input in cui l'i-simo elemento è rim-
        piazzato con k
  A[i] = k
  if (2i+1 ≤ A.heap-size and  A[i] ≤ A[2i+1])
    or (2i ≤ A.heap-size and A[i] ≤ A[2i])
          then Max-Heapify(A,i)
  proprietà violata rispetto ai figli
  while (i > 1 and A[i] > A[i/2])  proprietà violata rispetto al padre
     do scambia A[i] con A[i/2]
              i = i/2
```