Corso di Progettazione di algoritmi

Esercizi 5

Esercizio 1

Supponi di dover scegliere tra i seguenti tre algoritmi che risolvono un certo problema:

1. L'algoritmo A risolve ricorsivamente il problema su istanze di dimensione n dividendolo in 5 sottoproblemi ciascuno di dimensione $n/2$ e poi ricombinando le soluzioni in tempo lineare.

2. L'algoritmo B risolve ricorsivamente il problema su istanze di dimensione n dividendolo in 2 sottoproblemi, ciascuno di dimensione $n - 1$ e poi ricombinando le soluzioni tempo costante

3. L'algoritmo C risolve il problema su istanze di dimensione n dividendolo in 9 sottoproblemi, ciascuno di dimensione $n/3$, e poi ricombinando le soluzioni in tempo $O(n^3)$.

Assumendo che il tempo richiesto da ciascun algoritmo per suddividere il problema in sottoproblemi è $O(1)$, determinare la complessità asintotica dei tre algoritmi e individuare quello asintoticamete più veloce.

Esercizio 2

Supponi di dover scegliere tra i seguenti tre algoritmi che risolvono un certo problema:

1. L'algoritmo A risolve ricorsivamente il problema su istanze di dimensione n dividendolo in 4 sottoproblemi ciascuno di dimensione $n/16$ e poi ricombinando le soluzioni in tempo quadratico.

2. L'algoritmo B risolve ricorsivamente il problema su istanze di dimensione n dividendolo in 3 sottoproblemi, ciascuno di dimensione $n - 4$ e poi ricombinando le soluzioni tempo costante

3. L'algoritmo C risolve il problema su istanze di dimensione n dividendolo in 3 sottoproblemi, ciascuno di dimensione $n/5$, e poi ricombinando le soluzioni in tempo $O(n^2 \log n)$.

Assumendo che il tempo richiesto da ciascun algoritmo per suddividere il problema in sottoproblemi è $O(n)$, determinare la complessità asintotica dei tre algoritmi e individuare quello asintoticamete più veloce.

Esercizio 3

Progettare un algoritmo che, preso un vettore ordinato V di n interi distinti, determini se esiste un indice i tale che $V[i] = i$ in $O(\log n)$ tempo.

Esercizio 4

In un vettore V *di* n *interi si dice* gap *un indice* i, $1 \leq i < n$, *tale che* $V[i] < V[i+1]$.

- Dato un vettore V di $n \geq 2$ interi tale che $V[1] < V[n]$, provare che V ha almeno un *gap*.
- Progettare un algoritmo che, dato un vettore V di $n \geq 2$ interi tale che $V[1] < V[n]$, trovi un *gap* in $O(\log n)$ tempo.

Esercizio 5

In un vettore V *di* n *interi si dice* double-gap *un indice* i, $1 \leq i < n$, *tale che* $V[i+1] - V[i] \geq 2$.

- Dato un vettore V di $n \geq 2$ interi tale che $V[n] - V[1] \geq n$, provare che V ha almeno un *double-gap*.

- Progettare un algoritmo che, dato un vettore V di $n \geq 2$ interi tale che $V[n] - V[1] \geq n$, trovi un *double-gap* in $O(\log n)$ tempo.

Esercizio 6

In un vettore V di interi si dice *spessore* del vettore la differenza tra il massimo e il minimo del vettore. Progettare un algoritmo che, preso un vettore V di n interi ed un intero C, trovi un sottovettore (una sequenza di elementi consecutivi del vettore) di, lunghezza massima tra quelli di spessore al più C. La complessità dell'algoritmo deve essere $O(n \log n)$.

Esercizio 7

Sia V un vettore di n interi. Si dice che V è *continuo* se per ogni $i = 1, 2, \ldots, n-1$, $|V[i+1] - V[i]| \leq 1$. Si dice *zero* del vettore un indice k tale che $V[k] = 0$.

- Dato un vettore V di $n \geq 2$ interi continuo tale che $V[1] < 0$ e $V[n] > 0$, provare che V ha almeno uno *zero*.

- Progettare un algoritmo che, dato un vettore V di $n \geq 2$ interi continuo e tale che $V[1] < 0$ e $V[n] > 0$, trovi uno *zero* in $O(\log n)$ tempo.

Esercizio 8

Progettare un algoritmo che, dato un intero n, calcoli il valore $\lfloor \sqrt{n} \rfloor$ in $O(\log n)$ tempo, usando solo operazioni aritmetiche.

Esercizio 9

Si supponga di avere in input un vettore ordinato di n interi il cui contenuto è stato ruotato di k posizioni.

- Supponendo di conoscere solo n, progettare un algoritmo che restituisca l'elemento massimo del vettore in $O(\log n)$ tempo.
- Supponendo di conoscere n e k, progettare un algoritmo che restituisca l'elemento massimo del vettore in $O(1)$ tempo.

Esercizio 10

Dato un vettore ordinato di interi V ed un intero x vogliamo scoprire se x è presente tra gli elementi del vettore V.
Non è noto il numero n di elementi del vettore ma si dispone di una procedura SUB che, presi due interi i e y, restituisce -1 se il vettore V ha meno di i elementi, 0 se $V[i] \neq y$ e 1 se $V[i] = y$.
Assumendo che l'esecuzione della procedura SUB richiede tempo $O(1)$, progettare un algoritmo che risolve il problema in tempo $O(\log n)$.