Problem Set 2

docente: Luciano Gualà

Esercizio 1 (*equazioni di ricorrenza*)
Si risolvano le seguenti equazioni di ricorrenza. Si assuma sempre $T(1) = 1$.

(*a*) $T(n) = T(n - 10) + 10$.

(*b*) $T(n) = T(n/2) + 2^n$.

(*c*) $T(n) = T(n/3) + T(n/6) + n^{\sqrt{\log n}}$.

(*d*) $T(n) = T(\sqrt{n}) + \Theta(\log \log n)$.

(*e*) $T(n) = T(n/2 + \sqrt{n}) + \Theta(1)$.

(*f*) $T(n) = \sqrt{n}T(\sqrt{n}) + n$.

Esercizio 2 (*ricerca di un picco in una dimensione*)
Sia $V[1 : n]$ un vettore di n numeri non negativi. Un *picco* in V è una posizione $p \in \{1, \ldots, n\}$ il cui elemento ha a sinistra e a destra elementi non più grandi di lui, ovvero $V[p] \geq \max\{V[p - 1], V[p + 1]\}$. Quando p è sul bordo, la condizione è da considerarsi relativa solo all'unico elemento esistente adiacente a p. Pertanto, per semplicità e in modo equivalente considereremo $V[i] = -\infty$, quando $i < 1$ o $i > n$. Progettare un algoritmo che, dato V, trovi un picco in tempo $o(n)$.

Esercizio 3 (*ricerca di un picco in due dimensioni*)
Sia $M[1 : n, 1 : m]$ una matrice con n righe, m colonne, tale che $M[i, j]$ è un numero non negativo. Un *picco* in M è una posizione (p, q), $p \in \{1, \ldots, n\}$ e $q \in \{1, \ldots, m\}$ il cui elemento ha a sinistra, a destra, sopra e sotto elementi non più grandi di lui, ovvero $M[p, q] \geq \max\{M[p-1, q], M[p+1, q], M[p, q-1], M[p, q+1]\}$. Quando (p, q) è sul bordo, la condizione è da considerarsi relativa solo agli elementi esistenti adiacenti a (p, q). Pertanto, per semplicità e in modo equivalente considereremo $M[i, j] = -\infty$, quando uno dei due indici i o j è minore di 1 o maggiore di n.

- Si consideri il seguente algoritmo che essenzialmente esegue una ricerca di un picco spostandosi sempre su elementi più grandi. Più in dettaglio, l'algoritmo parte da una data posizione (i, j). Se tale elemento non è un picco si controlla gli (al più 4) elementi adiacenti a (i, j), si sposta sull'elemento di valore massimo, e procede ricorsivamente. Si dimostri se tale algoritmo è corretto e se ne studi la complessità asintotica nel caso peggiore.

- Si progetti un algoritmo che trovi un picco in M con complessità temporale $o(nm)$ e che usi la tecnica *divide et impera*.

Esercizio 4 (*embedding di un albero binario completo su una griglia bidimensionale*)
Sia dato un albero binario completo T con n foglie e una griglia bidimensionale (foglio a quadretti). Si vuole disegnare T sulla griglia in modo da non far intrecciare gli archi e minimizzando lo spazio utilizzato. Più precisamente, un *embedding* di T è un disegno di T sul foglio tale che: (i) i nodi di T sono disegnati sulle intersezioni della griglia (ovvero

come cerchi centrati su un qualche spigolo di un qualche quadratino), (ii) gli archi sono delle linee "seghettate" che seguono le linee del foglio, (iii) le linee che rappresentano gli archi non possono incrociarsi reciprocamente. Dato un embedding di T il suo *bounding box* è il rettangolo più piccolo che contiene l'intero embedding. Trovare un embedding di T che minimizza (asintoticamente) l'area del bounding box.