Algoritmi per Bioinformatica **Anno Acc. 2014-15**

Esercizi 1

1. Per ognuna delle seguenti affermazioni rispondere con vero o falso. Se vero, si fornisca una prova. Se falso, si fornisca un controesempio.

   - Per ogni istanza del problema dello stable matching esiste una soluzione (cioè uno stable matching) che include una coppia (m, w) tale che m è il primo nella lista di w e w è la prima nella lista di m.
   - Sia I un'istanza del problema dello stable matching per la quale esistono m e w tali che m è il primo della lista di w e w è la prima nella lista di m. Allora, ogni soluzione per tale istanza contiene la coppia (m, w)

2. Si consideri una variante del problema dello stable matching in cui alcuni uomini ed alcune donne esprimono ripugnanza per alcuni possibili accoppiamenti. In altre parole, per ogni uomo m le donne sono innanzitutto classificate in due categorie: possibili partner e partner inaccettabili. Le partner possibili sono a loro volta classificate in ordine dalla prima all'ultima. Ugualmente, ogni donna ha la propria classifica dei partner in cui alcuni sono considerati inaccettabili.

   Si provi a definire uno stable matching per questa variante e a modificare l'algoritmo di Gale-Stapley per risolvere questo problema.

3. Si supponga di avere per un dato problema diversi algoritmi con i seguenti running time. Si ha a disposizione un computer che esegue 10^9 operazioni al secondo ma lo si può usare per al più un ora di elaborazione. Si dica per ognuno degli algoritmi quale è la taglia massima n dell'input per la quale si riuscirebbe ad ottenere il risultato in un'ora.

   $$(a)\ n^2 \quad (b)\ n^3 \quad (c)\ 100n^2 \quad (d)\ n \log n \quad (e)\ 2^n \quad (f)\ 2^{\sqrt{n}}$$

4. Si ordinino le funzioni dell'esempio precedente, in modo tale che se f precede g nell'ordine, deve valere $f = O(g)$. Per ogni coppia di funzioni consecutive f e g si dica poi se vale anche $g = O(f)$, motivando la risposta.

   Quanto più lento diventa ognuno degli algoritmi di cui al punto precedente se: (a) la taglia dell'input raddoppia; (b) la taglia dell'input aumenta di 1.

5. Si consideri il seguente problema. Si cerca di capire il massimo grado di tolleranza di una determinata specie di cavie ad una data sostanza. Supponiamo che la sostanza possa essere dosata secondo una scala di interi. Quindi possiamo dare ad una cavia una dose di 1 unità, oppure una dose di 2 unità, oppure 3 unità e così via fino ad una dosa massima di n unità, per un qualche valore di n fissato.

   L'esperimento che possiamo fare è scegliere la dose x e somministrarla alla cavia. Se essa muore sappiamo che la dose è maggiore della massima tolleranza. Quindi dobbiamo provare con una dose più piccola usando un'altra cavia. Al contrario se la cavia sopravvive sappiamo che la massima tolleranza non è più piccola della dose somministrata.

   Vorremmo una soluzione che allo stesso tempo ci permetta di tenere basso sia il numero di esperimenti sia le cavie che rischiamo di far morire prima di trovare il valore della dose massima.

   Si analizzino inizialmente i seguenti due schemi:

   - Scegliamo le dosi seguendo l'algoritmo di ricerca binaria. Quale risulta essere il numero massimo di test? Quale risulta essere il numero massimo di cavie morte?
   - Scegliamo le dosi nell'ordine: 1, 2, 3, ... Quale risulta essere il massimo numero di test? Quale risulta essere il massimo numero di cavie morte?

   Cosa possiamo fare se abbiamo a disposizione al massimo 2 cavie e non possiamo usare più di $\Theta(\sqrt{n})$ test?