

HY380 – Αλγόριθμοι και πολυπλοκότητα

3^η Σειρά ασκήσεων

Ημερομηνία Παράδοσης: 24/03/2025 στο elearn
Υπεύθυνος βοηθός για τη 3^η Σειρά: Αποστόλης Μπανιώτης,
baniotis@csd.uoc.gr ή csdp1398@csd.uoc.gr

Άσκηση 1:

Δείξτε ότι η καλύτερη περίπτωση εκτέλεσης του αλγορίθμου QuickSort , γίνεται σε χρόνο $\Omega(n \log n)$.

Άσκηση 2:

Γράψτε έναν αλγόριθμο ο οποίος δέχεται δύο αλφαριθμητικά (Strings) S1 και S2, καθορίζοντας κατά πόσον το ένα είναι μία υπο ακολουθία του άλλου.

Άσκηση 3:

Επιλέξτε εάν οι παρακάτω προτάσεις είναι Αληθείς ή Ψευδές.

Δώστε μια σύντομη εξήγηση για την επιλογή σας.

- a) Πολυωνυμικός: καλός . Εκθετικός: κακός.
- (b) Radix sort δουλεύει σωστά όταν χρησιμοποιείς οποιονδήποτε σωστό αλγόριθμο ταξινόμησης για να ταξινομήσεις κάθε ψηφίο.
- (c) Δοθέντος ενός πίνακα $A[1::n]$ από ακεραίους, ο χρόνος που παίρνει ο Counting Sort είναι πολυωνυμικός σε σχέση με το μέγεθος εισόδου n .
- (d) Δοθέντος ενός πίνακα $A[1::n]$ από ακεραίους, ο χρόνος που παίρνει ο HeapSort είναι πολυωνυμικός σε σχέση με το μέγεθος εισόδου n .
- (ε) Για έναν αλγόριθμο Δυναμικού Προγραμματισμού, ο υπολογισμός όλων των τιμών με bottom-up είναι ασυμπτωτικά ταχύτερος από την χρησιμοποίηση αναδρομής και memoization.
- (ζ) Ο χρόνος ενός δυναμικού αλγορίθμου είναι πάντα $O(P)$ όπου P είναι ο αριθμός των υποπροβλημάτων.
- (η) Κάθε πρόβλημα που ανήκει στο NP επιλύεται σε εκθετικό χρόνο.

ΔΙΑΜΕΡΙΣΗ-ΤΑΧΥΤΑΞΙΝΟΜΗΣΗ (PARTITION-QUICK SORT)

Η διαδικασία ΔΙΑΜΕΡΙΣΗ (PARTITION) αναδιατάσσει την υποσυστοιχία $A[p..r]$ επί τόπου, επιλέγοντας πάντοτε το στοιχείο $x = A[r]$ ως οδηγό γύρω από τον οποίο θα διαμεριστεί η υποσυστοιχία.

PARTITION(A, p, r)

1. $x \leftarrow A[r]$
2. $i \leftarrow p - 1$
3. for $j \leftarrow p$ to $r - 1$
4. if $A[j] \leq x$ then
5. $i \leftarrow i + 1$
6. swap $A[i] \leftrightarrow A[j]$
7. swap $A[i + 1] \leftrightarrow A[r]$
8. return $i + 1$

Η TAXYTAΞΙΝΟΜΗΣΗ (QUICK SORT) υλοποιείται μέσω της ακόλουθης διαδικασίας.

QUICK SORT(A, p, r)

1. if $p < r$ then
2. $q \leftarrow \text{PARTITION}(A, p, r)$
3. QUICK SORT($A, p, q-1$)
4. QUICK SORT($A, q+1, r$)

Άσκηση 4:

- a'**. Περιγράψτε τη λειτουργία της διαδικασίας ΔΙΑΜΕΡΙΣΗ (PARTITION) στη συστοιχία $A = \langle 13, 19, 9, 5, 12, 8, 7, 4, 21, 2, 6, 11 \rangle$
- b'**. Ποια τιμή q επιστρέφει η ΔΙΑΜΕΡΙΣΗ όταν όλα τα στοιχεία της συστοιχίας $A[p..r]$ έχουν την ίδια τιμή; Τροποποιήστε τη ΔΙΑΜΕΡΙΣΗ έτσι ώστε όταν όλα τα στοιχεία της $A[p..r]$ έχουν την ίδια τιμή να επιστρέφει $q = \lfloor (p + r) / 2 \rfloor$.
- c'**. Εξηγήστε γιατί ο χρόνος εκτέλεσης της ΔΙΑΜΕΡΙΣΗΣ σε μια υποσυστοιχία μεγέθους n είναι $\Theta(n)$.

Άσκηση 5:

- a'**. Πως θα μπορούσε να τροποποιηθεί η TAXYTAΞΙΝΟΜΗΣΗ (QUICK SORT) ώστε να ταξινομεί τα στοιχεία σε μη αύξουσα σειρά;
- b'**. Ποιος είναι ο χρόνος εκτέλεσης της TAXYTAΞΙΝΟΜΗΣΗΣ όταν όλα τα στοιχεία της συστοιχίας A έχουν την ίδια τιμή;
- c'**. Δείξτε ότι, όταν η συστοιχία A περιέχει διαφορετικά στοιχεία και είναι ταξινομημένη κατά φθίνουσα σειρά, ο χρόνος εκτέλεσης της TAXYTAΞΙΝΟΜΗΣΗΣ είναι $\Theta(n^2)$.
- d'**. Δείξτε ότι ο χρόνος καλύτερης περίπτωσης της TAXYTAΞΙΝΟΜΗΣΗΣ σε μια συστοιχία με διαφορετικά ανά δύο στοιχεία, είναι: $O(n \lg n)$.

Άσκηση 6:

Περιγράψτε την λειτουργία της διαδικασίας ΑΠΑΡΙΘΜΗΤΙΚΗ ΤΑΞΙΝΟΜΗΣΗ (COUNTING SORT) στη συστοιχία $A = \langle 6, 0, 2, 0, 1, 3, 4, 6, 1, 3, 2 \rangle$

Άσκηση 7:

Περιγράψτε την λειτουργία της διαδικασίας ΑΡΙΘΜΟΤΑΚΤΙΚΗ ΤΑΞΙΝΟΜΗΣΗ (RADIX SORT) στην ακόλουθη λίστα λέξεων της Αγγλικής: COW, DOG, SEA, RUG, ROW, MOB, BOX, TAB, BAR, EAR, TAR, DIG, BIG, TEA, NOW, FOX.

Άσκηση 8:

Περιγράψτε την λειτουργία της διαδικασίας ΤΑΞΙΝΟΜΗΣΗ ΜΕ ΔΟΧΕΙΑ (BUCKET SORT) στη συστοιχία

$A = \langle 0.75, 0.13, 0.16, 0.64, 0.39, 0.20, 0.89, 0.53, 0.71, 0.42, 0.19 \rangle$, χωρίζοντας το διάστημα $[0, 1)$ σε 10 ίσου μεγέθους υποδιαστήματα.

Άσκηση 9:

Ποιοι από τους παρακάτω αλγορίθμους ταξινόμησης είναι (η μπορούν να υλοποιηθούν ως) ευσταθείς (stable): ενθετική ταξινόμηση (insertion sort), συγχωνευτική ταξινόμηση (merge sort), ταξινόμηση σωρού (heap sort), και ταχυταξινόμηση (quick sort);

Άσκηση 10:

Σκεφτείτε το πρόβλημα της συναλλαγής για n σεντς, χρησιμοποιώντας το μικρότερο αριθμό κερμάτων.

Ας υποθέσουμε ότι η αξία του κάθε νομίσματος είναι ένας ακέραιος .

α . Περιγράψτε ένα άπληστο αλγόριθμο ο οποίος θα κάνει συναλλαγές που αποτελούνται από 25 σεντς ,

10 σεντς , 5 σεντς , και 1 σεντ . Αποδείξτε ότι ο αλγόριθμος σας δίνει μια βέλτιστη λύση.

β . Ας υποθέσουμε ότι τα διαθέσιμα νομίσματα είναι σε ονομαστικές αξίες που είναι δυνάμεις του c , δηλαδή , οι αξίες είναι c^0 , c^1 , ... , c^k για ορισμένους ακέραιους.

$c > 1$ και $k \geq 1$. Δείξτε ότι ο άπληστος αλγόριθμος αποδίδει πάντα βέλτιστη λύση.

γ . Δώστε ένα σύνολο κερμάτων , για τα οποία ο άπληστος αλγόριθμος δεν θα αποδώσει μια βέλτιστη λύση. Η συσκευή σας θα πρέπει να περιλαμβάνει κέρματα του 1 σεντ, έτσι ώστε να υπάρχει μια λύση για κάθε τιμή του n .

δ . Δώστε ένα $O(nk)$ -Time αλγόριθμο που κάνει την συναλλαγή για οποιοδήποτε σύνολο k διαφορετικής αξίας νομισμάτων , υποθέτοντας ότι ένα από τα κέρματα είναι 1 σεντ.