Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή ΗΜ&ΜΥ
Αλγόριθμοι και Πολυπλοκότητα
7ο εξάμηνο, Ροή Λ
Ακαδημαϊκή περίοδος: 2011-2012

2η Σειρά Γραπτών Ασκήσεων

Γερακάρης Βασίλης
<vgerak@gmail.com>
Α.Μ.: 03108092

4 Ιανουαρίου 2012

1 Επιτροπή Αντιπροσώπων (KT 4.15)

Ταξινομούμε τον πίνακα σε αύξουσα σειρά με βάση τα f_i. Βρίσκουμε το διάστημα (έστω m) με το μέγιστο f_m που η αρχή του s_m βρίσκεται πρίν το τέλος του 1ου, f_1 και το επιλέγουμε ως αντιπρόσωπο. Θα αποδείξουμε ότι η επιλογή που κάνουμε (σε κάθε βήμα) είναι και η βέλτιστη δυνατή.

Έστω k, ένας διαφορετικός αντιπρόσωπος, που είναι η βέλτιστη επιλογή. Δεν είναι δυνατόν να έχει δείκτη $k > m$, αφού έτσι δε θα επικάλυπτε το 1ο διάστημα (εξ'ορισμού του m). Αφού λοιπόν $k \leq m$ θα ισχύει και $f_k \leq f_m$. Άρα ο k επικαλύπτει **το πολύ** όσα διαστήματα επικαλύπτει η επιλογή μας, επομένως η επιμέρους λύση μας είναι η βέλτιστη. Επαγωγικά προκύπτει η ορθότητα του αλγορίθμου μας.

Algorithm 1 Άσκηση 1

```
Sort A on ascending order using f_i as key
procedure RepresentantivesSelect(A, N)
    if N = 0 then
        return RepresentativesList
    else
        i ← 1
        while i ≤ N and s_i < f_1 do
            i ← i + 1
        i ← i − 1
        RepresentativesList.append(i)
        j ← i + 1
        while f_i > s_j do
            A.remove(j)
            N ← N − 1
            j ← j + 1
        for j ← i to 1 step -1 do
            A.remove(j)
            N ← N − 1
        A.remove(i)
        return RepresentativesSelect (A, N)
```

2 Βιαστικός Μοτοσυκλετιστής

Ταξινομούμε τα διαστήματα σε αύξουσα σειρά με βάση τα όρια ταχυτήτων v_i και δημιουργούμε τον πίνακα χρόνων t, όπου $t_i = l_i/v_i$. Έστω T ο χρόνος που θα ξεπεράσουμε το όριο ταχύτητας και u τα χιλιόμετρα κατά τα οποία θα το ξεπεράσουμε. Έαν $T \leq t_1$, τότε κάνουμε T λεπτά με $v_1 + u$ km/h, και τα υπόλοιπα με το όριο ταχύτητας, αλλιώς κάνουμε t_1 λεπτά υπερβαίνοντας το όριο και επαναλαμβάνουμε τη διαδικασία για $T - t_1$ λεπτά για το επόμενο διάστημα.

Θα αποδείξουμε ότι η επιλογή που κάνουμε σε κάθε βήμα, είναι η βέλτιστη δυνατή. Ο συνολικός χρόνος που χρειάζεται για να φτάσει στον προορισμό του είναι:

$T_{total} = \sum_{i=1}^{n} t_i = \sum_{i=1}^{n} \frac{l_i}{u_i}$.

Η μέγιστη (ποσοστιαία) ελάττωση των χρονικών διαστημάτων προκύπτει αν προσθέσουμε την ταχύτητα u στο μικρότερο παρονομαστή v_i. Καθώς λοιπόν όλες οι τιμές είναι δεδομένες και σταθερές (T, v_i, l_i), οποιοδήποτε άλλο διάστημα αν επιλέγαμε, θα έδινε ποσοστιαίο κέρδος χρόνου **το πολύ** όσο αυτό που υπολογίσαμε, επομένως η επιμέρους λύση μας είναι η βέλτιστη.

Algorithm 2 Άσκηση 2

Sort A on ascending order using v_i as key
for $i \leftarrow 1$ **to** N **do**
    $t_i \leftarrow l_i / v_i$
procedure TimeSelect(A, N)
    $i \leftarrow 1$
    while $T \geq t_i$ **do**
        $Selection.append(i, t_i)$
        $T \leftarrow T - t_i$
        $i \leftarrow i + 1$
    $Selection.append(i, T)$
    return $Selection$

Η πολυπλοκότητα του αλγορίθμου είναι $\Theta(n \log n)$, όσο χρειάζεται η ταξινόμηση. Στον ταξινομημένο πίνακα το αποτέλεσμα προκύπτει σε γραμμικό χρόνο. Στην περίπτωση που η υπέρβαση στο όριο γινόταν κατά παράγοντα α > 1, η επιλογή του διαστήματος δε θα επηρρέαζε το αποτέλεσμα, αφού το ποσοστιαίο κέρδος θα ήταν το ίδιο για όλα τα διαστήματα.

3 Βότσαλα στη Σκακιέρα (DVP 6.5)

Test3

4 Χωρισμός Κειμένου σε Γραμμές

Test4

5 Αντίγραφα Αρχείου (KT 6.12)

Test5

6 Bonus: Έλεγχος Ταξινόμησης

Bonus!