1^η Εργασία Στο Μάθημα Αλγόριθμοι

Ονοματεπώνυμο: Βίκτωρ Κυρτσούδης
ΑΕΜ: 4143

Άσκηση 1

$\sqrt{\log_9 n}, \log\sqrt{n}, 2^{\frac{\log n}{2}}, 8^{\log n}, \log(4^n), \log(n^n), (\log(2^n))^2, n^{\log n}, 2^n, 2^{n\log n}, n!$

Άσκηση 2

Αλγόριθμος εύρεσης του πλήθους ελάχιστων μονοπατιών από τον κόμβο s στον κόμβο t σε έναν γράφο $G = (V, E)$

```
getAmountOfShortestPaths(G, s, t):

Θέτω PathsTo[s] ← 1
for κάθε κόμβο v ∈ V − {s} do
   Θέτω PathsTo[v] ← 0
end for
Θέτω LevelOf[s] ← 0
for κάθε κόμβο v ∈ V − {s} do
   Θέτω LevelOf[v] ← −1
end for
Εισάγω στην Layer[0] τον κόμβο s
Θέτω τον layer counter i ← 0
Θέτω τον paths counter cnt ← 0
while η Layer[i] δεν είναι κενή ΚΑΙ cnt == 0 do
   Αρχικοποιώ την Layer[i+1] ως κενή
   for κάθε κόμβο u ∈ Layer[i] do
      for κάθε κόμβο v για τον οποίο υπάρχει ακμή (u, v) do
         if LevelOf[v] == −1 then
            Θέτω LevelOf[v] ← i + 1
            Εισάγω τον κόμβο v στην Layer[i+1]
         end if
         if LevelOf[v] > LevelOf[u] then
            Θέτω PathsTo[v] ← PathsTo[u] + PathsTo[v]
         end if
         if v == t then
            Θέτω cnt ← cnt + PathsTo[u]
         end if
      end for
   end for
   Θέτω i ← i + 1
end while
return cnt
```

Ο αλγόριθμος βασίζεται στον BFS ο οποίος έχει χρονική πολυπλοκότητα $O(|V| + |E|)$, αλλά όλα τα ελάχιστα μονοπάτια θα εμφανιστούν στο ίδιο Layer οπότε όταν βρω πρώτη φορά το t θα είναι το τελευταίο Layer που εξετάζω και το πλήθος των ελάχιστων μονοπατιών στο t θα είναι ίσο με το άθροισμα των ελάχιστων μονοπατιών των κόμβων του προηγούμενου Layer που συνδέονται με το t.

Άσκηση 3

(a) Το συγκεκριμένο πρόβλημα μπορεί να μοντελοποιηθεί με ένα κατευθυνόμενο γράφημα όπου οι κόμβοι είναι όλες οι πιθανές καταστάσεις των τριών δοχείων, δηλαδή η ποσότητα νερού που περιέχουν. Κάθε ακμή ξεκινάει από έναν κόμβο u και καταλήγει σε έναν κόμβο v αν και μόνο αν η κατάσταση του κόμβου v προκύπτει από την κατάσταση του κόμβου u με την εφαρμογή μιας επιτρεπτής εκροής.
Ξεκινώντας από την κατάσταση (a=0,b=7,c=4) πρέπει να καταλήξουμε σε οποιαδήποτε κατάσταση με b=2 ή/και c=2

(b) Μπορούμε να χρησιμοποιήσούμε είτε τον DFS είτε τον BFS αλγόριθμο αλλά με τον BFS θα βρούμε την επίλυση με τις λιγότερες απαιτούμενες ενέργειες.

Άσκηση 4

(a) Σε κάθε κλήση της συνάρτησης εκτυπώνονται n αριθμοί και ξανακαλείται η συνάρτηση 2 φορές για $\lfloor n/2 \rfloor$ αριθμούς. Άρα η αναδρομική σχέση είναι:

$$T(n) = \begin{cases} 0 & \text{if } n = 0 \\ 2T(\lfloor n/2 \rfloor) + n & \text{if } n \geq 1 \end{cases}$$

και απο Master Theorem με $a = 2, b = 2, c = 1$ έχουμε ότι
$T(n) = \Theta(n \log_2 n)$ αριθμοί εκτυπώνονται.

(b) Ο αριθμός 1 εμφανίζεται μία φορά σε κάθε κλήση της συνάρτησης άρα η αναδρομική σχέση γίνεται:

$$T(n) = \begin{cases} 0 & \text{if } n = 0 \\ 2T(\lfloor n/2 \rfloor) + 1 & \text{if } n \geq 1 \end{cases}$$

και απο Master Theorem με $a = 2, b = 2, c = 0$ έχουμε ότι
$T(n) = \Theta(n^{\log_2 2}) = \Theta(n)$ φορές εκτυπώνεται το 1.

Άσκηση 5

(a) Εφόσων το x είναι πλειοψηφικό στοιχείο του A ισχύει ότι

$$\text{πλήθος } x > n/2 \tag{1}$$

Άρα υπάρχει τουλάχιστον ένα ζεύγος με δύο φορές το x και άρα θα περάσει μία φορά στο B **(2)**. Έστω ότι έχουμε πίνακα M που έχει σε όλα τα ζεύγη τουλάχιστον μία φορά το x (λόγω του **(1)**). Αυτό σημαίνει ότι ο B_M θα πάρει σίγουρα ένα x (λόγω **(2)**) και απ' όλα τα άλλα ζευγάρια θα πάρει ή ένα x, αν και τα δύο στοιχεία είναι x, ή τίποτα, αν το δεύτερο στοιχείο είναι διαφορετικό από x. Άρα ο B_M θα πάρει μόνο x και άρα το x θα είναι πλειοψηφικό στοιχείο του.

Ξεκινώντας απ'τον M μπορεί να προκύψει ένας M′ αν πάρουμε δύο ζεύγη του M: [x,y] και [x,z] με y,z οποιαδήποτε στοιχεία και αντιμεταθέσουμε το y με το δεύτερο x, δηλαδή πάρουμε τα ζεύγη [x,x] και [y,z]. Τότε στον $B_{M'}$ αντί να περάσουν το πολύ δύο x, θα περάσει σίγουρα ένα x και το πολύ ένα y(αν $y = z$). Οπότε τα y που θα περάσουν στον $B_{M'}$ θα είναι σίγουρα λιγότερα από τα x, λόγω **(2)** και άρα ο $B_{M'}$ θα έχει πλειοψηφικό στοιχείο το x.

Οπότε οι αντιμεταθέσεις τέτοιας μορφής σε έναν πίνακα A δεν επηρεάζουν την ύπαρξη πλειοψηφικού στοιχείου στο B_A.

Αφού ο κάθε πίνακας A μπορεί να προκύψει απ' τον M με τις κατάλληλες αντιμεταθέσεις, αν ο A έχει πλειοψηφικό στοιχείο το x, τότε και ο B_A έχει πλειοψηφικό στοιχείο το x.

(b) Στην περίπτωση που εξετάζουμε, το περισσευούμενο στοιχείο δεν είναι το πλειοψηφικό άρα ο πίνακας A′ που δεν το περιέχει έχει άρτιο μέγεθος και έχει κι αυτός πλειοψηφικό το x.

Αφού ο B_A είναι ίσος με τον $B_{A'}$ ο οποίος έχει πλειοψηφικό το x (από **(a)**), τότε και ο B_A έχει πλειοψηφικό το x.

Αλγόριθμος εύρεσης πλειοψηφικού στοιχείου σε έναν πίνακα A

(c)

```
getMajorityItem(A):

n ← |A|
B ← τα ζεύγη του A χωρίς το περισσευούμενο αν υπάρχει
for κάθε ζεύγος [x,y] του B do
  if x=y then
    Αντικαθιστώ το [x,y] με x
  else
    Σβήνω το [x,y]
  end if
end for

if n > 0 then
  if n άρτιος then
    b ← getMajorityItem(B)
    if b είναι πλειοψηφικό στοιχείο του B then
      return b
    end if
  else if τελευταίο στοιχείο του A είναι πλειοψηφικό στοιχείο του A then
    return τελευταίο στοιχείο του A
  else
    b ← getMajorityItem(B)
    if b είναι πλειοψηφικό στοιχείο του B then
      return b
    end if
  end if
end if

return null
```

Στην αρχή βρίσκει το B_A σε γραμμικό χρόνο και αγνοεί το περισσευούμενο στοιχείο αν υπάρχει. Αν το n είναι άρτιος τότε καλεί τον εαυτό της με τον πίνακα B_A, ελέγχει αν το στοιχείο που επέστρεψε είναι πλειοψηφικό σε γραμμικό χρόνο και αν είναι το επιστρέφει. Αν είναι περιττός ελέγχει αν το τελευταίο στοιχείο του A είναι πλειοψηφικό και αν είναι το επιστρέφει αλλιώς κάνει το ίδιο με πριν. Σε κάθε άλλη περίπτωση επιστρέφει null.

(d) Η πολυπλοκότητα του αλγορίθμου μπορεί να βρεθεί απ' τον αναδρομικό τύπο:

$$T(n) = \begin{cases} 0 & \text{if } n = 0 \\ T(\lfloor n/2 \rfloor) + n & \text{if } n \geq 1 \end{cases}$$

Άρα από το Master Theorem με $a = 1, b = 2$ και $c = 1$ έχουμε ότι $T(n) = \Theta(n^1) = \Theta(n)$.