

ΣΧΕΔΙΑΣΗ ΚΑΙ ΑΝΑΛΥΣΗ ΑΛΓΟΡΙΘΜΩΝ
2023
ΕΦ' ΟΛΗΣ ΤΗΣ ΥΛΗΣ

ΘΕΩΡΙΑ

Εισαγωγή στην Μαγογλώσσα – Ιδιότητες και αποτύπωση αλγορίθμων

notes\01 Εισαγωγή.pdf σελ. 1-9

ΣYB

notes\01_Εισαγωγή.pdf σελ. 11-14

- το περιπλό αυτό είναι εξ. ορθογώνιος αριθμός το ο ή διάφορα το ι. Ο συνομιλώς αριθμός των ΣΥΒ είναι επίσης ένας περιπλός αριθμός, γιατί $2 + 3 = 5$.

Εφόσον η μέγιστη κυρτή που μπορεί να φτάσει ο περιπλός είναι \sqrt{n} , τότε η μέγιστη κυρτή που μπορεί να φτάσει ο περιπλός των ΣΥΒ του Γραμμή 5.6. Στο άλλο δέρο μπορεί να κάθε επιπλόγμα του [δύο] που να έχουμε την περιπλόση όπως $i \mid n$ ($\pi_i = n$, $i = 12$). Επιδή ο αριθμός των επιπλόγματων είναι ίσος με $\lfloor \sqrt{n} \rfloor - 1 + 1 = \lfloor \sqrt{n} \rfloor$, ο συνομιλώς αριθμός των ΣΥΒ που εκτελέστηκε στην γραμμή $4 + 6 = 8$ κυρτίζεται από $2 \lfloor \sqrt{n} \rfloor$ έως $5 \lfloor \sqrt{n} \rfloor$. Ο αριθμός των ΣΥΒ που δηλώνονται στη Γραμμή 3 και αφορούν τον δέκτη i είναι $3 \lfloor \sqrt{n} \rfloor + 2$. Επίσης εκπλήνουμε της ΣΥΒ στη Γραμμή 2 το αλγόριθμο. Κατα συνακός αριθμός των ΣΥΒ προστιθέξουμε από άποτο (ένα φρέσκο) από την ποσοτήρια

$$5\lfloor \sqrt{n} \rfloor + 3\lfloor \sqrt{n} \rfloor + 2 + 3 = 8\lfloor \sqrt{n} \rfloor + 5, \quad (1.2)$$

και από το κάτιο (άποτο φρέσκο) από⁵

$$2\lfloor \sqrt{n} \rfloor + 3\lfloor \sqrt{n} \rfloor + 4 + 3 = 5\lfloor \sqrt{n} \rfloor + 5. \quad (1.3)$$

⁵Το κάτιο φρέσκο μπορεί να [βαθιστεί] παρατηρήσας, ότι οι εντολές των Γραμμών 5, 6 θα εκτελεστούν πάνω π_1 - π_{12} μέσα φορά στο n είναι πράξης. Μέσω της ανάλογης αποτίση στο κάτιο φρέσκο προστίθεται η σταθερά 3.

16

ΚΕΦΑΛΑΙΟ 1. ΒΑΣΙΚΕΣ ΕΝΝΟΙΕΣ

Όπως θα δούμε σε επόμενα κεφάλαια η απύλαση εστιάζει στα φρέσκά της του αριθμού των ΣΥΒ πάρι στον επακριβή προσδιορισμό του αριθμού των σε κάλιο στηγμάτων.

1.7 Ασκήσεις

Πους είναι ο αριθμός των ΣΥΒ που εκτάλει ο ωντοδοκίνος που απεικονίζεται στην πρώτη στήλη των Ημίνικα 1.1. Είναι ίσος με τον αριθμό των ΣΥΒ που εκτάλει ο ωντοδοκίνος που απεικονίζεται στη δεύτερη στήλη;

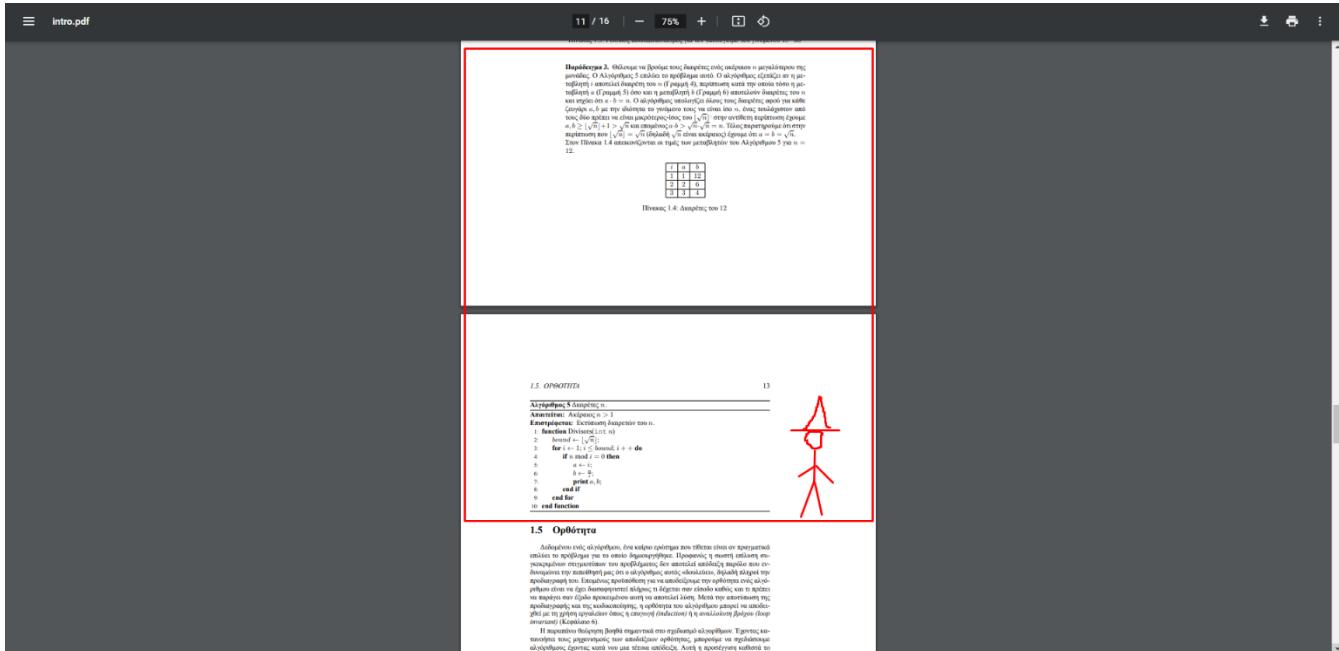
Θεωρούμε τις παρούσα εξιδικεύσεις της εντολής `for` που παρουσιάστηκε στην Ενίστημα 1.4:

- (a') `for i ← m; i ≤ n; i + do`
- (b') `for i ← m; i ≤ n; i ← i + a do`



Divisors

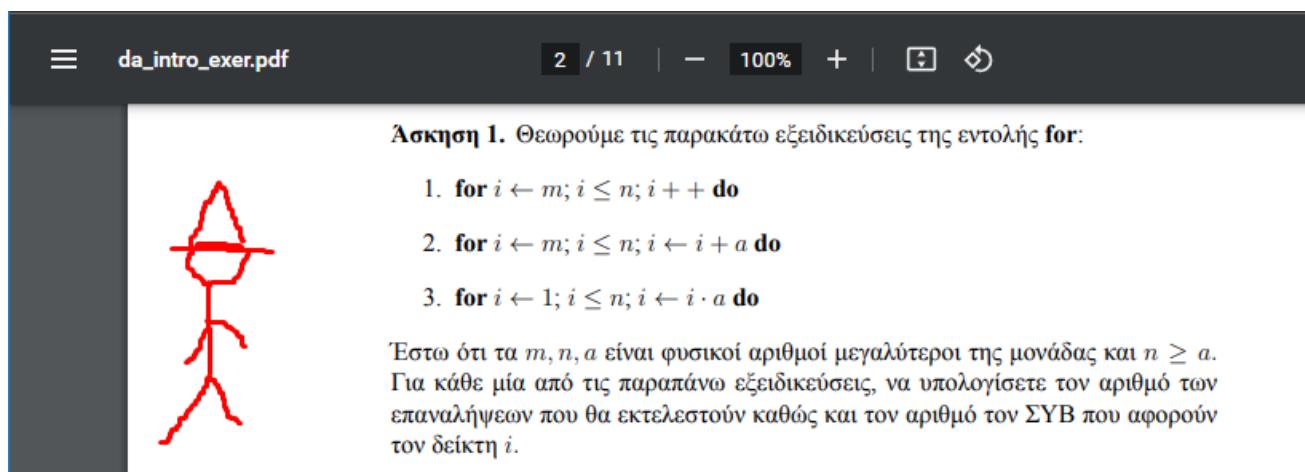
notes\01 Εισαγωγή.pdf σελ. 11-12



ΑΣΚΗΣΕΙΣ

ΣYB $\sigma\varepsilon$ *for-loops*

[exercises\Εισαγωγικές Ασκήσεις.pdf](#) / σελ. 2-3



$$(A) \# \text{ loops} = \begin{cases} 0, & \text{an } m > n \\ n-m+1, & \text{an } m \leq n \end{cases}$$

$$\# \text{EYB} = \begin{cases} l + l = 2, & \text{av } m > n \quad (i \leq m + i \leq n = 2) \\ l + 3(n-m+l) + l, & \text{av } m \leq n \quad (i \leq m + (i \leq n + (i+1)) * \# \text{loops} + \\ & \quad i \leq n) \end{cases}$$

$$\Leftrightarrow \# \Sigma B = \begin{cases} 2, & n > m \\ 3^{*(n-m+1)} + 2, & n \leq m \end{cases}$$

$$(\beta) \quad \# \text{ loops} = \begin{cases} 0, & \text{on } m > n \\ k, & \text{on } m \leq n \end{cases}$$

End 1st loop: $i \leftarrow m + a$ für alle m Tiquationen \oplus Gleichungen $i = m + k + a > n (=)$

$$\text{End 2nd loop: } i = m + 2a \quad t > \frac{m-m}{a} \quad , \quad t = \left\lfloor \frac{n-m}{a} \right\rfloor + 1$$

Enw \neq Corp : is-mt ker

$$\# \text{EB} = \begin{cases} L + l = 2, & \text{av } m > n \\ 3 * \left(\lfloor \frac{n-m}{\alpha} \rfloor + l \right) + 2, & \text{av } m \leq n \end{cases}$$

$$(F) \quad \# \text{Loopr} = \begin{cases} 0, & \text{an even number } n \leq l \\ k, & \text{an } n > l \end{cases}$$

End 1st loop: ~~finalized~~ Final preparation. Boxes ready

\leftarrow no 2nd step: it $a^2 = a^k > n \Leftrightarrow k > \log_a n$

$$t = \lfloor \log_2 n \rfloor + 1$$

End of Coup: 1st d - 5-

$$\# \Sigma B = \begin{cases} L + l = 2, & \text{when } n < l \\ 3^*(\lfloor \log_3 n \rfloor + 1) + 2, & \text{when } n \geq l \end{cases}$$

≡ da_intro_exer.pdf 3 / 11 | - 100% + | ☰



Ασκηση 2. Να εκπονήσετε έναν αλγόριθμο που να δέχεται σαν είσοδο έναν μη-αρνητικό ακέραιο n και υπολογίζει τον μικρότερο ακέραιο k για τον οποίο ισχύει ότι $n \leq 2^k$. Να υπολογίσετε τον αριθμό των επαναλήψεων που εκτελεί ο αλγόριθμος.

Algo problem: Είναι δύο αριθμοί $n > 0$ και r
που λαμβάνουν το μικρότερο ακέραιο k , στον οποίο
 $n \leq 2^k$

Analisis: με αρνητικός αριθμός $n > 0$

Επιτρέπεται: Σλάχιστος αριθμός k , στον οποίο $n \leq 2^k$

function Min_k (int n) # loops = 0 , αν $r > n$

```

    k ← 0;
    r ← 2;
    while r < n do
        r ← 2 * r;
        k++;
    end_while
    Return k;
end_function

```

~~End If loop:~~

~~End End loop:~~

~~End k loop:~~

$r = 2 \cdot k \geq n \quad (=) \quad k \geq$

$r = 2^k \geq n \Rightarrow \left\{ \begin{array}{l} k \geq \lg n \Rightarrow k = \lceil \lg n \rceil , n > 1 \\ 0 , n \leq 1 \end{array} \right.$

Γνώστα μαγικά
 $\lg n = \log_2 n$

ΣYB σε for-loop με βήμα πάτωμα

[exercises/Eισαγωγικές Ασκήσεις.pdf](#) σελ. 9-10



9 / 11 | - 100% + ⊞ ◊

Ασκηση 7. Έστω n, a φυσικοί αριθμοί μεγαλύτεροι της μονάδας. Να υπολογίσετε το πλήθος των επαναλήψεων του βρόχου

```
for t ← n; t ≥ 1; t ← ⌊t/a⌋ do
...
end for
```

End 1st loop: $\left\lfloor \frac{t}{a} \right\rfloor$

End 2nd loop: $\left\lfloor \frac{\left\lfloor \frac{n}{a} \right\rfloor}{a} \right\rfloor = \left\lfloor \frac{n}{a^2} \right\rfloor$ * | Σωρτρα πατημάτων

End k loop: $\left\lfloor \frac{n}{a^k} \right\rfloor$

Για να τερματιστεί το loop γρίφη $t = \left\lfloor \frac{n}{a^k} \right\rfloor < 1 \Leftrightarrow$
 $\left\lfloor \frac{n}{a^k} \right\rfloor < 1 \Leftrightarrow \frac{n}{a^k} < 1 \Leftrightarrow a^k > n \Leftrightarrow k > \log_a n \Leftrightarrow$

$k = \lfloor \log_a n \rfloor + 1$

[exercises/Εισαγωγικές Ασκήσεις.pdf](#) σελ. 4-5

≡ da_intro_exer.pdf 4 / 11 | - 100% + | ☰



Άσκηση 3. Να εκπονήσετε έναν αλγόριθμο που να τυπώνει τους διψήφιους ακέραιους με κανένα ίδιο ψηφίο. Να υπολογίσετε τον αριθμό των ΣΥΒ που εκτελεί ο αλγόριθμος.

17/3/2023] Εγχροί - σελ. 24 [Chapt 4a]

Exer [Αλγεράριος NeyDigits]

Anaτύπωση:

Επιτρέπεται: ακέραιοι από 10 έως 99 με διαφορετικά ψηφία!

```

Function NeyDigits()
    for i<=9; i<=9; i++ do
        for j<=0; j<=9; j++ do
            if i ≠ j then
                num ← 10 * i + j;
                Print num;
    end_if
end_for
end_for
end_function

```

-F-

$9 \times 4 + 1 = 3F$ ΣΥΒ (loops 2nd loop * (if i=j) + (if i ≠ j))

$3 + 10 + 2 = 32$ ΣΥΒ

$32 + 3F = 69$ loops total

$(9 \times 69 + 3 \times 9 + 2 = 650)$

$(i \leq 9) + (\# loops 1^{st} loop) + (i \leq 9)$

$(i \leq 9) + (\# loops 2^{nd} loop) + (\# loops total) + (i \leq 9) + (\# loops 2^{nd} loop) + (i \leq 9)$

in i=j και δεν θα εκτελωται 1 if

Chapt 4a

exercises\Εισαγωγικές Ασκήσεις.pdf σελ. 5

Guess_01

[exercises|Εισαγωγικές Ασκήσεις.pdf σελ. 6-9](#)

da_intro_exer.pdf 7 / 11 | - 100% + | ☰ ⌂



7

Ασκηση 5. Να υπολογίσετε την τιμή που επιστρέφει η συνάρτηση που περιγράφεται στον Αλγόριθμο 4 και στον Αλγόριθμο 5.

da_intro_exer.pdf 8 / 11 | - 100% + | ☰ ⌂



8

Ασκηση 6. Να υπολογίσετε τον αριθμό των ΣΥΒ που εκτελούνται από τον Αλγόριθμο 4.

da_intro_exer.pdf 6 / 11 | - 100% + | ☰ ⌂



6

Αλγόριθμος 4

Απαιτείται: ακέραιος n

Επιστρέφεται: ακέραιος r

```
1: function Guess01(int n)
2:     r ← 0;
3:     for i ← 1; i ≤ n - 1; i ++ do
4:         for j ← i + 1; j ≤ n; j ++ do
5:             for k ← 1; k ≤ j; k ++ do
6:                 r++;
7:             end for
8:         end for
9:     end for
10:    return r;
11: end function
```

Chapter 4.C

Algebraic Guess OI

through i - Array cells (4.15)

constant: requires n

inception: requires r

function (guess OI) (int n)

r ← 0;

for i ← l; i ≤ n - l; i++ do

 for j ← i + l; j ≤ n; j++ do

 for k ← l; k ≤ j; k++ do

 r += j // SIB

end-for

end-for

end-for

return r;

end-function

$$\sum_{i=1}^n i = \frac{n(n+1)}{2}$$

$$\sum_{i=1}^n i^2 = \frac{n(n+1)(2n+1)}{6}$$

$$\sum_{i=1}^n L = n - L + L$$

$$r = \sum_{i=1}^{n-1} \sum_{j=i+l}^n L \Rightarrow r = \sum_{i=1}^{n-L} \sum_{j=i+l}^n j \stackrel{\textcircled{R}}{=} r = \sum_{i=1}^{n-L} \left(\frac{n(n+1)}{2} - \frac{(i+l)(i+l+1)}{2} \right)$$

$$r = \sum_{i=1}^{n-L} \frac{n(n+1)}{2} - \sum_{i=1}^{n-L} \frac{i(i+1)}{2} = \left(n-L \right) \cdot \frac{n(n+1)}{2} - \frac{1}{2} \left(\sum_{i=1}^{n-L} i^2 + \sum_{i=1}^{n-L} i \right)$$

$$r = \left(n-L \right) \cdot \frac{n(n+1)}{2} - \frac{1}{2} \left[\frac{(n-L) \cdot n \cdot (2(n-L)+L)}{6} + \frac{n(n-L)}{2} \right] =$$

$$r = (n-L) \cdot \frac{n(n+1)}{2} - \frac{9}{2}$$

$$\begin{aligned}
 r &= \frac{1}{3}n(n-L)(n+L) \\
 \text{1st loop } n-L &\quad \# \text{ loops} \quad \text{for } (i \in L \quad i \leq n-L) \\
 \sum_{i=L}^{n-L} B(i) + 3(n-L) + 2 &= \# \sum YB \Rightarrow \\
 \# \sum YB &= \sum_{i=L}^{n-L} \left(\sum_{j=L}^{n-i} A(j) + 3(n-i) + 2 \right) + 3(n-L) + 2 \Rightarrow \\
 \# \sum YP &= \sum_{i=L}^{n-L} \left(\sum_{j=L}^{n-i} (S_j + 2) + 3(n-i) + 2 \right) + 3(n-L) + 2 \\
 \Rightarrow H \sum YB &= \boxed{\frac{5n(n-L)(n+L)}{6} + 5n - 2}
 \end{aligned}$$

Return \star ,
end-function

$$\sum_{i=L}^n i = \frac{n(n+L)}{2}, \quad \sum_{i=L}^n i^2 = \frac{n(n+L)(2n+L)}{6}, \quad \sum_{i=L}^n L = n-L+L$$

≡ da_intro_exer.pdf 10 / 11 | - 100% + | ☰ ⌂



Άσκηση 8. Εστω φυσικοί n, m με $n \geq m > 1$. Να υλοποιήσετε σε ψευδοκώδικα τον αλγόριθμο του Ευκλείδη ο οποίος υπολογίζει τον μέγιστο κοινό διαιρέτη των αριθμών αυτών. Να βρείτε ένα άνω φράγμα στον αριθμό των επαναλήψεων που εκτελεί ο αλγόριθμος.

Επίλογος

$n \geq m > 1$

MKD των n και m των

Αλγόριθμος MKD

Απαιτείται: αριθμοί $n \geq m > 1$

Επιτρέπεται: ο MKD των n και m

function MKD (int m, int n)

```

    a <- n;
    b <- m;
    while b > 0 do
        r <- a mod b;
        a <- b;
        b <- r;
    end-while
    return a;
end-function

```

- 10 -

Chapter 5a

$\text{MKD}(35, 15) = 5$

$$35 = 15 \times 2 + 5$$

$$15 = 5 \times 3 + 0$$

Έχει λύθει σπανακητικά
μήπερ να γρίζεται
το υπόλοιπο

$$\textcircled{*} \quad r < \frac{a}{2} \quad \textcircled{1}$$

$$a) \quad b \leq \frac{a}{2} \quad \left| \begin{array}{l} \Rightarrow 0 \leq r \leq b-1 \\ b \leq \frac{a}{2} \end{array} \right| \Rightarrow r \leq \frac{a}{2}-1 < \frac{a}{2}, \text{ lösbar in } \textcircled{1}$$

$$b) \quad b > \frac{a}{2} \quad \left| \begin{array}{l} \Rightarrow a-b < \frac{a}{2} \\ r = a-b \end{array} \right| \Rightarrow r < \frac{a}{2}, \text{ lösbar in } \textcircled{1}$$

④ $\Sigma_{i=0}^{\infty}$ von b aufger. nicht lösbar, wenn $a \leq b$

$$b \leq a \quad \textcircled{2}$$

$$\textcircled{1}, \textcircled{2} \quad \left| \Rightarrow b - \cancel{a} < \frac{1}{2}(a-b) \right.$$

$$\begin{array}{l} a \leq b; \\ b = 0 \quad \text{rite } a \cdot b = 0 \\ b \neq 0; \end{array}$$

Bleiben noch additive zu
a · b, wodurch die proprieit
der Σ nicht anwendbar zu b

$\Sigma_{i=0}^{\infty}$ von b auf a aufger. \Leftrightarrow # loops (k)

$$\left\lfloor \frac{n+m}{2^k} \right\rfloor = 0 \Rightarrow \frac{n+m}{2^k} < 1 \Rightarrow n+m < 2^k \Rightarrow$$

$$k > \lg n + \lg m \Rightarrow k = \left\lfloor \lg n + \lg m \right\rfloor + 1$$

Hinweis: $\lg n + \lg m = \lg(nm) = 0$ angesichts Σ loops
für Fixierung von k

02_ΚΛΑΣΕΙΣ ΠΟΛΥΠΛΟΚΟΤΗΤΑΣ

ΘΕΩΡΙΑ

Κλάσεις και Ιεραρχία συναρτήσεων

[Slides \(Παλαιό Υλικό\) Ασυμπτωτικός Συμβολισμός και Ιεραρχία Συναρτήσεων.pdf σελ. 1 – 17](#)

CompClass.pdf

Επίλυση (συνέχεια)

Βασικές αρχές

Ιεραρχία

- Τάξεις Συναρτήσεων
- Συμβολισμοί
- Παράδειγμα
- Επίλυση
- Επίλυση (συνέχεια)
- Επίλυση (συνέχεια)
- Επίλυση (συνέχεια)
- Τάξεις Συναρτήσεων (συνεχ.)
- Τάξεις Συναρτήσεων (συνεχ.)
- Λουμπιωτική ταξινόμηση συναρτήσεων
- Χρήση οριών
- Λουμπιωτική ταξινόμηση

Συνολικά, δείξαμε ότι

$$\beta \cdot n^2 \leq \frac{1}{2}n^2 - 3n \leq \alpha \cdot n^2,$$

ισχύει για $\alpha = \frac{1}{2}, \beta = \frac{1}{14}, n \geq 7$ και άρα

$$T(n) = \frac{1}{2}n^2 - 3n = \Theta(n^2).$$

A hand-drawn red stick figure is on the right side of the slide.

Ιδιότητες κλάσεων και ασυμπτωτική κατάταξη συναρτήσεων

[notes/02_Κλάσεις πολυπλοκότητας.pdf σελ. 1-10](#)

[extraNotes\Ιεραρχία Συναρτήσεων και Πολυπλοκότητες.pdf](#) έξτρα σημειώσεις

comp_class.pdf

28 ΚΕΦΑΛΑΙΟ 2. ΙΕΡΑΡΧΙΑ ΣΥΝΑΡΤΗΣΕΩΝ

Θέτοντας $v = \lg^2 n$ έρχουμε $f_3(n) = f(v) = \lg v$. Στο Παρόρθετα 6 δείχνεται ότι $\lim_{v \rightarrow \infty} \frac{\lg v}{v} = 0$ (θεωρείστε v στη θέση των n στην (2.6)). Εκομένως, ασυμπτωτικά ισχύει η ανασύρση

$$\lg v < v \Rightarrow f_3(n) = \lg \lg^2 n < \lg^2 n = f_2(n). \quad (2.19)$$

Συνοδούνται τις (2.16), ..., (2.19) έρχουμε την κατάταξη

$$f_3(n) < f_2(n) < f_4(n) < f_3(n) < f_1(n).$$

Αξίζει να παρατηρήσουμε ότι παρόλο που $\Theta(f_1(n)) = \Theta(f_3(n))$ αφού οι συναρτήσεις $f_1(n), f_3(n)$ είναι αφορετικές πολύσημων συμπτωτικής με ίδιο μεγοτεύσιμο όρο, μπορούμε να τις διατάξουμε με βάση την (2.15).

2.5 Ασυμπτωτική συμπεριφορά αλγόριθμων

Απειροτελείς ακίνητης περιοχής περιήρχεται $T(n)$, οπότε λογισμόμας απειλεται η απονομή της συνάρτησης $\eta(n)$ για την ακίνητη περιοχή $T(n) = \theta(\eta(n))$, όπου $\eta \in (\Omega, \Theta, \omega)$. Σημηνεύεται ότι δεν έχει ανάγκης είναι τότες (ποσόν, πολλούς λογαρίθμους) $\theta(\eta(n))$. Οριστούμε ότι η απομετρική συμπεριφορά του αλγόριθμου είναι $\theta(g(n))$. Επανάστηνε στις κλάσεις, αν ένας αλγόριθμος είναι τάξης $\Omega(g(n))$ αν οι αποδείξεις που έχει στην περιοχή $T(n)$ για απομετρική συμπεριφοράς $g(n)$ δεν μπορεί να εξασθενήσει την περιοχή $T(n)$ συμπεριφέρεται με μία θετική σταθερή c . Για το λόγο αυτό η $\Omega(g(n))$ αναφέρεται απομετρική εκτίμησης «κα» των α . Αντίτοιχα στην $T(n) = \Omega(g(n))$ παίρνεται $\Omega(g(n))$ αναφέρεται απομετρική εκτίμησης για την περιοχή $T(n)$. Είναι αλγόριθμος αναφέρεται ομογενής σε απομετρική $g(n)$ εάν κάθε άνθρωπος στην περιοχή $T(n)$ παρατηρεί την περιοχή $T(n)$ στην περιοχή $T(n)$ στην περιοχή $T(n)$.

Τα διαφορετικά αποτελέσματα των Εννέτευρων 2.2, 2.4 διανοούλονται την εκτίμηση της απομετρικής συμπεριφοράς ενός αλγόριθμου. Για παράδειγμα, είδουμε ότι η συνάρτηση δίνει τον αλγόριθμο για το Αλγόριθμο 3 $T(n) = 7n + 3$ (Επόμενη 1.6 Σελίδα). Η περιοχή $T(n) = 7n + 3$ διανοούλεται ότι η περιοχή $T(n)$ δεν έχει αναποτύχηση a , $a > 0$, που να ακανονίζουν τις (2.2), (2.3), (2.4) δραστηριότητες να δείξουμε ότι $7n + 3 = \Theta(n)$. Παρατηρούμε όμως ότι κάτι τέτοιο προκύπτει σαν άμεση εφαρμογή του Πορίσματος 3 αφού από το Αλγόριθμο 2 έχουμε $\frac{T(n)}{n} = \frac{7n+3}{n} = 7$.

Ο Αλγόριθμος 9 παρατηρείται ότι η περιοχή $T(n)$ διανοούλεται ότι η περιοχή $T(n)$ δεν έχει αναποτύχηση a που να αποδεικνύεται ότι $T(n) = \Theta(n)$. Οι αριθμητικοί βρόγοι που αποδεικνύουν το γνωμόνιο j (μεταβλητή $muffles$, t) όπου ο λεκτικός j αναφέρεται στον $(j-1)$ —οπότε όφει της προδόσης. Ο δεκτικός j μεταβάλλεται στον περιοχή $T(n)$ που προσδιορίζεται από την προδόση $muffles$ καθώς και το διέρθεται των πρόσων $j+1$ όφει της προδόσης (μεταβλητή $muffles$). Οι αριθμοί των ΣΥΒ των Αλγορίθμων 9 υπολογίζεται ως εξής. Στη Γραμμή 1 εκτελέστηκαν μία προδόση και μία εκχύρωση ενός ο αριθμού των επαναλόγων που εκτελεύτησαν τα δύο από ΣΥΒ είναι ίσος με την ημι του δεκτική j . Επομέν για την έλειψη

Ασυμπτωτική συμπεριφορά Sum_of_Terms02

notes\02_Κλάσεις πολυπλοκότητας.pdf σελ. 10-13

2.6 ΑΣΚΗΣΕΙΣ 31

Μέρος παρεργούμενος στις κάθε επανάληψη αποτελείται από μία τιμή του δείκτη j και για αυτήν η ανάληψη αποδίδει αποτέλεσμα συμβόλια ΣΥΒ. Οι αντίστοιχες τιμές του ΣΥΒ που αποδίδει το επαναληπτικό βήμα της αποτάσσουν ειδικά απόλογα με την τιμή του δείκτη j . Εποι θα προσφέρονται η τιμή μεγέθους του ΣΥΒ του αλγόριθμου: ο δείκτης j παρέχει η διαφορετικές τιμές και για κάθε τέτοιη τιμή εκτελούνται $\Theta(j)$ ΣΥΒ από τον επαναληπτικό βήμα. Άρα ο αριθμός των ΣΥΒ που εκτελούνται από τον επαναληπτικό βήμα συνολικά (κατά την εκτέλεση του αλγόριθμου) δίνεται από το άθροισμα των ΣΥΒ αυτών για κάθε τιμή $j \in \{0, \dots, n-1\}$ με μεθυματικούς όρους, έχουμε:

$$\sum_{j=0}^{n-1} \Theta(j) = \sum_{j=0}^{n-1} c_1 \cdot j = c_1 \sum_{j=0}^{n-1} j = c_1 \frac{n(n-1)}{2} = \Theta(n^2). \quad (2.23)$$

Με αντίτυπο τόσο πρόσφατο να αναλύονται του Αλγόριθμο 3. Στον αλγόριθμο αυτό υπάρχει μόνο ένας βήμα. Ο αριθμός των ΣΥΒ σε κάθε επανάληψη είναι σταθερός, έτσι ότι $c > 0$, ενώ ο αριθμός των επαναλήψεων είναι οριζόμενος n . Άρα ο συνολικός αριθμός των ΣΥΒ είναι $c \cdot n$.

Η παρακάτω ανάλυση οδηγεί στο συμπέρασμα ότι ο μεγαλύτερος αριθμός των βήματων που είναι διαδεχόμενοι καθούς, είναι την ομοιαρική συμπεριφορά ενός αλγόριθμου. Θα πρέπει να είμαστε ιδιαίτερα προστατικοί με το συμπέρασμα αυτό αριθμού διεργασίας για γενικότερες κανόνες. Για να ισχύει θα πρέπει ο αριθμός των επαναλήψεων του κάθε βήματος να είναι συναρτήση - έτσι και έμμεσα - της πορείατρου ή ποτο αντανακλά το μέγεθος της συγκεκριμένης και επαλόνον αριθμός των ΣΥΒ σε κάθε επανάληψη - έτσι ότι ο ΣΥΒ που εκτελούνται από τους διαφορετικούς βήματος - να είναι σταθερός, δηλαδή αναμένεται της πορείατρου n . Επειδόν θα πρέπει να διασυντηθούμε στον παρακάτω παραγράφης δεν αφορούν απολογισμούς αλγόριθμων.

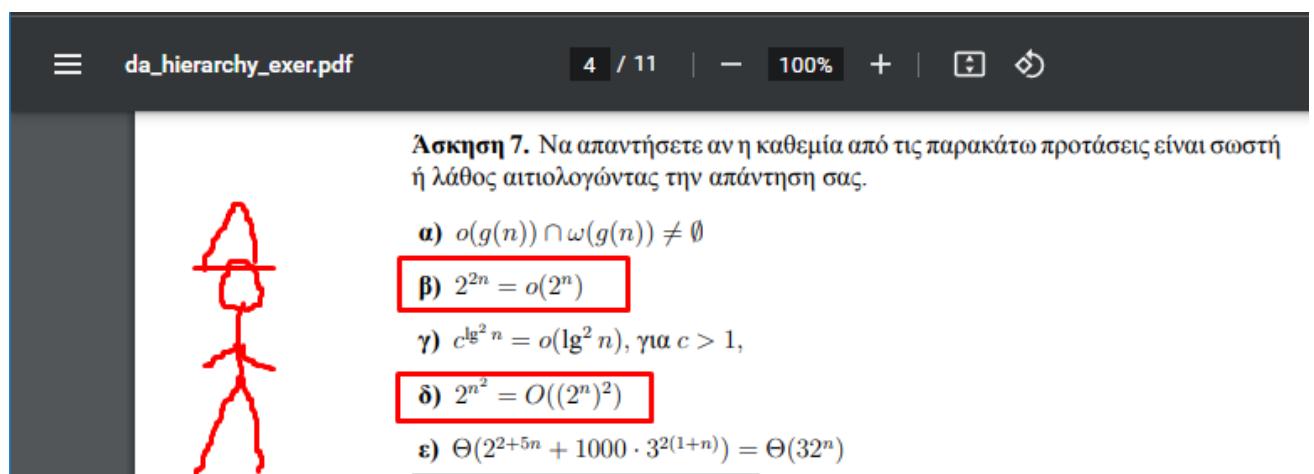
2.6 Ασκήσεις

1. Έστο $f(n) = 3n^2 - 5n + 4$. Χαρίζεται τη χρήση όρων να δείξετε ότι $f(n) = \Theta(n^2)$.
2. Έστο $a > 1$. Να δείξετε ότι $f(n) = \Theta(\lg_a n)$ τότε $f(n) = \Theta(\lg n)$.
3. Να δείξετε ότι $f(n) = a_k n^k + a_{k-1} n^{k-1} + \dots + a_1 n + a_0 = \Theta(n^k)$, όπου $n, k \in \mathbb{N}$, και $a_i \in \mathbb{R}_+$, για κάθε $i \in \{0, \dots, k\}$.
4. Να δείξετε ότι $\Theta(\lg(n-1)) = \Theta(\lg n) = \Theta(\lg(n+1))$.

ΑΣΚΗΣΕΙΣ

Ασκήση με πολυπλοκότητες

exercises\Ασκήσεις σε ιεραρχίες συναρτήσεων.pdf σελ. 4-5



Ασκηση 7. Να απαντήσετε αν η καθεμία από τις παρακάτω προτάσεις είναι σωστή ή λάθος αιτιολογώντας την απάντηση σας.

α) $o(g(n)) \cap \omega(g(n)) \neq \emptyset$

β) $2^{2n} = o(2^n)$

γ) $c^{\lg^2 n} = o(\lg^2 n)$, για $c > 1$,

δ) $2^{n^2} = O((2^n)^2)$

ε) $\Theta(2^{2+5n} + 1000 \cdot 3^{2(1+n)}) = \Theta(32^n)$

$$1 \text{ unit} \quad L/2(S_n^2 + H_{n-3})$$

Chapter Fb

- $f(n) = O(g(n)) \Rightarrow f(n) = o(g(n))$ ΙΑΘΟΣ
- $f(n) = o(g(n)) \Rightarrow f(n) = O(g(n))$ ΙΑΘΟΣ

- $2^{2^n} = o(2^n)$ ΙΑΘΟΣ

$$\lim \frac{2^{2^n}}{2^n} = \lim \frac{2^{n+1}}{2^n} = \lim \frac{2^n \cdot 2^n}{2^n} = \lim 2^n = \infty$$

- $2^{n^2} = O((2^n)^2)$ ΙΑΘΟΣ

$$\begin{aligned} n^2 &> 2n \quad \forall n \geq 3 \\ \Rightarrow 2^{n^2} &> 2^{2n} = (2^n)^2 \end{aligned}$$

κανονικός

Ασκήση με ιεραρχία συναρτήσεων

[exercises/Ασκήσεις σε ιεραρχίες συναρτήσεων.pdf](#) σελ. 5-6

≡ da_hierarchy_exer.pdf 6 / 11 - 100% + ⌂ ⌂

Άσκηση 8. Καθώς $n \rightarrow \infty$, να διατάξετε σε αύξουσα σειρά τις συναρτήσεις

1. $f_1 = 2^{\lg \sqrt{n}}$
2. $f_2 = 3^{n^2}$
3. $f_3 = n^2 \cdot 2^{\lg \lg n^3}$



6

4. $f_4 = n^{\sqrt{n}}$
5. $f_5 = 2^{\sqrt{2 \lg n}}$

$\Rightarrow 2^n > 2 \quad (= 1)$

Given:
 $f_1 = 2^{\lg n^n}$
 $f_2 = 3^n$
 $f_3 = n^2 \cdot 2^{\lg n^3}$
 $f_4 = n^{\lg n}$
 $f_5 = 2^{\lg \lg n}$

$f_5 = 2^{\lg \lg n} \Rightarrow 2^{\lg \lg n^2} = 2^{\lg (n^2)} = 2^{(n^2)^{1/2}} \quad \text{Kwisi Kwisi}$
 $f_3 = n^2 \cdot 2^{\lg n^3} = n^2 \cdot 2^{\lg n^3} > n^2 = 2^{\lg n^2} \quad \boxed{f_3 > f_5} \quad \text{Chapter Fe} \quad \text{①}$

$f_2 = 2^{\lg n^n} = \sqrt{n^n} = n^{n/2}$
 $f_4 = n^{\lg n} = n^{\lg n^3} \quad \text{Kwisi Kwisi} \quad \boxed{f_2 > f_4} \quad \text{②}$

$\forall n > 4 \quad \frac{n}{2} > \sqrt{n} \Rightarrow n^{n/2} > n^{\sqrt{n}} \Rightarrow f_2 > f_4 \quad \boxed{f_2 > f_4} \quad \text{③}$

$n^2 > n^2 \lg n^3 \Rightarrow n^3 > f_3 \quad \boxed{f_4 > f_3} \quad \text{④}$

$n > \frac{n}{2} \quad \boxed{n^2 > \frac{n}{2} \lg n^3} \Rightarrow f_2 = 3^{n^2} > 3^{n/2 \lg n^3} = f_1 \quad \text{⑤}$
 ~~$n^3/n > \lg n^3$~~
 ~~$\lg n$~~

$\text{①, ②, ③} \Rightarrow f_1 > f_4 > f_3 > f_5 \quad \text{⑥}$
 $\text{④, ⑤} \Rightarrow f_2 > f_1 > f_4 > f_3 > f_5 \quad \text{⑦}$

Απόδειξη ασυμπτωτικής εκτίμησης συνάρτησης

[exercises\Ασκήσεις σε ιεραρχίες συναρτήσεων.pdf](#) σελ. 7-8

≡ da_hierarchy_exer.pdf 7 / 11 | - 100% + | ☰ ⌂

Ασκηση 10. Να δείξετε ότι $\lg n! = \Theta(n \lg n)$.



Exer.

$$\text{W.S.o. } \lg n! = \Theta(n \cdot \lg n)$$

$$1. \lg n! = O(n \cdot \lg n)$$

$$n! < n^n \Rightarrow \lg n! < \lg n^n = n \cdot \lg n \quad \forall n \geq 2$$

$$2. \lg n! = \Omega(n \cdot \lg n) \quad \left\{ \begin{array}{l} \lg a+b = \lg a + \lg b \\ \lg n! = \sum_{i=1}^n \lg i \geq \sum_{i=\frac{n}{2}+1}^n \lg i \geq \sum_{i=\frac{n}{2}+1}^n \lg \frac{n}{2} = \frac{n}{2} \lg \frac{n}{2} = \frac{n}{2} (\lg n - 1) = \end{array} \right.$$

$$\frac{n}{2} \lg n - \frac{n}{2} \geq c \cdot n \lg n \Rightarrow \cancel{\frac{1}{2}(\cancel{c} + \cancel{\frac{1}{2}})} \cancel{\lg n}$$

$$\frac{n}{2} (\lg n - 1) \geq c \cdot n \lg n \Rightarrow \frac{1}{2} (1 - \frac{1}{\lg n}) \geq c$$

$$1 - \frac{1}{\lg n} > 0 \quad \forall n \geq 2$$

Απόδειξη ασυμπτωτικής εκτιμήσης συνάρτησης

[exercises|Ασκήσεις σε ιεραρχίες συναρτήσεων.pdf](#) σελ. 8

da_hierarchy_exer.pdf

8 / 11 | - 100% + | ☰

Ασκηση 11. Για $a > 1$ να δείξετε ότι $a^{n^{-1}} = \Theta(1)$.

03_ΠΙΝΑΚΕΣ

ΘΕΩΡΙΑ

Πίνακες και άλλες δομές

[notes\03_Πίνακες και άλλες δομές.pdf σελ. 1-17](#)

The screenshot shows a PDF document titled "tables.pdf". At the top, there are navigation controls: a menu icon, the file name "tables.pdf", page number "17 / 17", zoom level "100%", and icons for download, print, and more. Below the controls, there is pseudocode:

```
7:     size_Q --;
8:     return a;
9: end if
10: end function
```

Below the pseudocode, a note states: "Παρατηρούμε ότι ο αριθμός των ΣΥΒ σε κάθε μία από τις λειτουργίες push και pop είναι τάξης Θ(1)."

3.3.2 Ουρά

Η ουρά είναι ένα αντικείμενο αντίστοιχο της στοίβας με τη διαφορά ότι η διαγείριση σε σχέση με την εισαγωγή και εξαγωγή στοιχείων γίνεται σύμφωνα με τον κανόνα FIFO (first-in, first-out). Δηλαδή ένα στοιχείο μπορεί να εισαχθεί όπουδήποτε στηγμή ώλλες το στοιχείο που εξέρχεται πρώτο είναι αυτό που έχει παραμείνει στην ουρά περισσότερο. Άρα κάθε νέο στοιχείο προστίθεται στο τέλος της ουράς ενώ το παλαιότερο στοιχείο βρίσκεται στην κορυφή της. Η λειτουργία εισαγωγής ονομάζεται enqueue ενώ δεκμενε η διαδικασία εξαγωγής.

Όπος και στην περίπτωση της στοίβας, η ουρά Q μπορεί να υλοποιηθεί με τη χρήση μονοδιάστατου πίνακας το πλήθος των στοιχείων στην ουρά δίνεται (πάλι) από την τιμή της μεταβλητής $size_Q$. Η τιμή της μεταβλητής αυτής μαίνεται (κατά ένα) από κάθε κλήση της διάδικτιας enqueue ενώ αυξάνεται (κατά ένα) από κάθε κλήση της διάδικτιας enqueue. Προφανώς αν η τιμή της μεταβλητής αυτής είναι ιση με μηδέν τότε

ΑΣΚΗΣΕΙΣ

LSR

[exercises/Ασκήσεις σε πίνακες.pdf σελ. 4-6](#)

The screenshot shows a PDF document titled "da_data_exer.pdf". At the top, there are navigation controls: a menu icon, the file name "da_data_exer.pdf", page number "4 / 8", zoom level "100%", and icons for download, print, and more. Below the controls, there is a section titled "20. end function".

Ασκηση 4. (Τεχνική δύο δεικτών) Δίνεται μια ακολουθία θετικών ακεραίων αριθμών οι οποίοι είναι αποθηκευμένοι σε έναν πίνακα A στις θέσεις από lo ως hi . Ζητείται αλγόριθμος γραμμικού χρόνου που να υπολογίζει το μεγαλύτερο δυνατό αριθμό a , για τον οποίο να υπάρχουν ακέραιοι p, q με την ιδιότητα

$$\sum_{i=lo}^{lo+p-1} A[i] = a = \sum_{j=hi-q+1}^{hi} A[j]$$

και επιπλέον $p + q \leq hi - lo + 1$.
Προσέξτε ότι το πρόβλημα έχει πάντα λύση (για $a = 0$ και $p = q = 0$).

Chapter 9a

~~Ex. 8/1/2023~~

loshi left-sum right-sum
 ↑ ↑
 $a = \sum_{i=l_0}^{l_0+P-1} A[i] = a = \sum_{j=h_i-q+1}^{h_i} A[j]$ ①
 ↓ ↓
 $P + q \leq h_i - l_0 + 1$ ②

2 σετες Να βρεθη αλγόριθμος που να υπολογίζεται το a
 ής γενικός χρώς που να αποσύνει τις συντετροφές ①, ②

To p Εξισων στην την αρχή την ημερα A[]

To q Εξισων στην την αρχή την ημερα A[]

function L-R-S (int A[], int l₀, int h_i)

i ← l₀; a ← 0;

j ← h_i;

left-sum ← 0; right-sum ← 0;

while (i <= j) do 0 ηρος που ο αριθμος εκτελείται

left-sum ← left-sum + A[i]; ής γενικός χρώς

right-sum ← right-sum + A[j];

A[i] ← 0; A[j] ← 0;

if left-sum > right-sum then

j --;

else if left-sum < right-sum then

i ++;

else

a ← left-sum;

i ++;

end_if

end_while

return a;

end_function

Triads

[exercises\Ασκήσεις σε πίνακες.pdf](#) σελ. 3-4

da_data_exer.pdf

4 / 8

100%

+

□



Άσκηση 3. Δίνεται μια ακολουθία ακεραίων αριθμών οι οποίοι είναι αποθηκευμένοι σε έναν πίνακα A στις θέσεις από 1 ως n . Να εκπονηθεί αλγόριθμος ο οποίος να υπολογίζει το πλήθος των τριάδων των αριθμών που περιέχονται στον πίνακα και το άθροισμα τους είναι ίσο με μηδέν.

```
function Triads( int A[], int n )
    Tafwirhetai A; → O(n2)
    count ← 0;
    for k ← 1; k ≤ n; k++ do
        i ← k+1;
        j ← n;
        while i < j do
            sum ← A[i] + A[j] + A[k];
            if sum = 0 then
                count++;
                i++; j--;
            else if sum < 0 then
                i++;
            else
                j--;
        end_if
    end_while
end_for
return count;
```

Add - function

3 διάκτο
 i, k, j

$$T(n) \leq \sum_{k=1}^n [n - (k+1)] \cdot c = O(n^2)$$

O ΗΣΥΒ ρέβα στην

while. Είναι σπάνιας

αρνητικός, σε χριστιανούς
νε γνωρίσεις ανατυπώνει

τη ΣΥΒ

ΘΕΩΡΙΑ

Εισαγωγή στην Αναδρομή

[notes\04_Αναδρομή.pdf σελ. 1-4](#)

recursion.pdf 5 / 8 | - 100% + | ☰ ⌂ ⌄ ⌅

αυτές υποθετικών τις σεσεις των στοιχειων του πίνακα A με τα όποια συγκρίνεται το στοιχείο a. Πιο συγκεκριμένα, οι τιμές που παίρνει ο δείκτης j όσο η συνθήκη του βρόχου της Γραμμής 16 είναι αληθής αποτελούν το σύνολο

$$\left\{ hi, hi - r, hi - r - \lfloor \frac{hi - lo}{2} \rfloor, hi - r - 2\lfloor \frac{hi - lo}{2} \rfloor \right\}$$

Κάνοντας πράξεις το παραπάνω σύνολο γράφεται ως

$$\left\{ hi, hi - r, lo + \lfloor \frac{hi - lo}{2} \rfloor, lo \right\}$$

Ο βρόχος δεν εκτελείται αν το στοιχείο a είναι μεγαλύτερο από το A[hi] ή μικρότερο από το A[lo] ενώ τερματίζει αν είναι μικρότερο-ίσο από κάποιο από τα στοιχεία που υποθετικών οι τιμές του παραπάνω συνόλου - τα στοιχεία αυτά εξετάζονται με τη σειρά που εμφανίζονται οι τιμές στην απεικόνιση του παραπάνω συνόλου. Μετά τον τερματισμό, εξετάζεται αν το j δεικτοδοτεί στοιχείο ίσο με το a (Γραμμή 20). Αν δεν συμβαίνει αυτό τότε αν υπάρχει το στοιχείο a στον πίνακα A τότε θα πρέπει να αναζητηθεί είτε στην ομάδα που δεικτοδοτείται από το σύνολο

$$\{hi - r - 1, \dots, hi - r - \lfloor \frac{hi - lo}{2} \rfloor + 1\} = \{hi - r - 1, \dots, lo + \lfloor \frac{hi - lo}{2} \rfloor + 1\}$$

ή στην ομάδα

$$\{hi - r - \lfloor \frac{hi - lo}{2} \rfloor - 1, \dots, hi - r - 2\lfloor \frac{hi - lo}{2} \rfloor + 1\} = \{lo + \lfloor \frac{hi - lo}{2} \rfloor - 1, \dots, lo + 1\}$$

ανάλογα με το αν $a > lo + \lfloor \frac{hi - lo}{2} \rfloor$ ή $a < lo + \lfloor \frac{hi - lo}{2} \rfloor$. Οπότε ο αλγόριθμος καλείται αναδρομικά με παραμέτρους που δεικτοδοτούν το πρώτο (μικρότερο) και το τελευταίο



$$\lfloor \frac{n - 1}{2} \rfloor - 1.$$



7 / 8 | - 100% + | ☰ ⌂

4.2.4 Διατεταγμένα στατιστικά

Ένα από τα πρώτα προβλήματα που περιγράφαμε (Κεφάλαιο 1) ήταν αυτό της εύρεσης του μικρότερου σε μία αταξινόμητη σειρά ακεραίων. Θα γενικεύσουμε το πρόβλημα αυτό στο πρόβλημα εύρεσης του k -ιοστού μικρότερου στοιχείου σε μία αταξινόμητη σειρά n στοιχείων, όπου $k \leq n$. Θα θεωρήσουμε ότι η σειρά των ακεραίων βρίσκεται αποθηκευμένη σε έναν πίνακα A στις θέσεις από lo έως hi .

Ο Αλγόριθμος 27 επιλέγει το παραπάνω πρόβλημα καλώντας τη συνάρτηση Pivot_Partition (Αλγόριθμος 10) η οποία, έχοντας αναδιατάξει τα στοιχεία του πίνακα A , επιστρέφει τη σωστή θέση, έστω j , (Ιδιότητα 1) που έχει τοποθετήσει το στοιχείο που βρισκότων πριν σε μία τυχαία θέση pvt του πίνακα A . Η επιλογή της pvt γίνεται από τη συνάρτηση



recursion.pdf

8 / 8 | - 97% + | ☰ ⌂

Rand η οποία επιστρέφει έναν «τυχαίο» ακέραιο από το σύνολο $\{lo, \dots, hi\}$. Έστω ότι το στοιχείο $A[j]$ - που πλέον έχει τοποθετηθεί στη σωστή θέση (Γραμμή 5) - είναι το i -οστό μικρότερο στοιχείο του πίνακα: παρατηρούμε ότι η τιμή i υπολογίζεται από τη σχέση $i = j - lo + 1$. Αν $i = k$ τότε το στοιχείο αυτό είναι το ζητούμενο και ο υπολογισμός ολοκληρώνεται. Διαφορετικά, αν $i > k$ τότε η διαδικασία επαναλαμβάνεται θέτοντας $hi = j - 1$, ενώ αν $i < k$ θέτοντας $lo = j + 1$ και αναζητώντας το $k - i$ μικρότερο στοιχείο. Η διαδικασία επαναλαμβάνεται αναδρομικά μέχρι να βρεθεί μία θέση j για την οποία $i = k$. Αυτό, αν δεν προκύψει νεράτερα, θα συμβεί αναγκαστικά όταν η σειρά αποτελείται από ένα στοιχείο ($lo = hi$). Επομένως η διαδικασία τερματίζεται.

Αλγόριθμος 27 k -ιοστό μικρότερο στοιχείο

Απαιτείται: πίνακας A με στοιχεία στις θέσεις από lo έως hi , ακέραιος $k \in \{1, \dots, hi - lo + 1\}$.

Επιστρέφεται: k -ιοστό μικρότερο στοιχείο του πίνακα A .

```
1: function k-Element(int A[], int lo,int hi, int k)
2:   if lo = hi then
3:     return lo
4:   end if
5:   n ← hi - lo + 1;
6:   pvt ← Rand(n) + lo;
7:   j ← Pivot_Partition(A,lo,hi,pvt);
8:   i ← j - lo + 1;
9:   if i = k then
10:    return A[j];
11:   else if i > k then
12:    return k-Element(A,lo,j - 1,k);
13:   else
14:    return k-Element(A,j + 1,hi,k - i);
15:   end if
16: end function
```



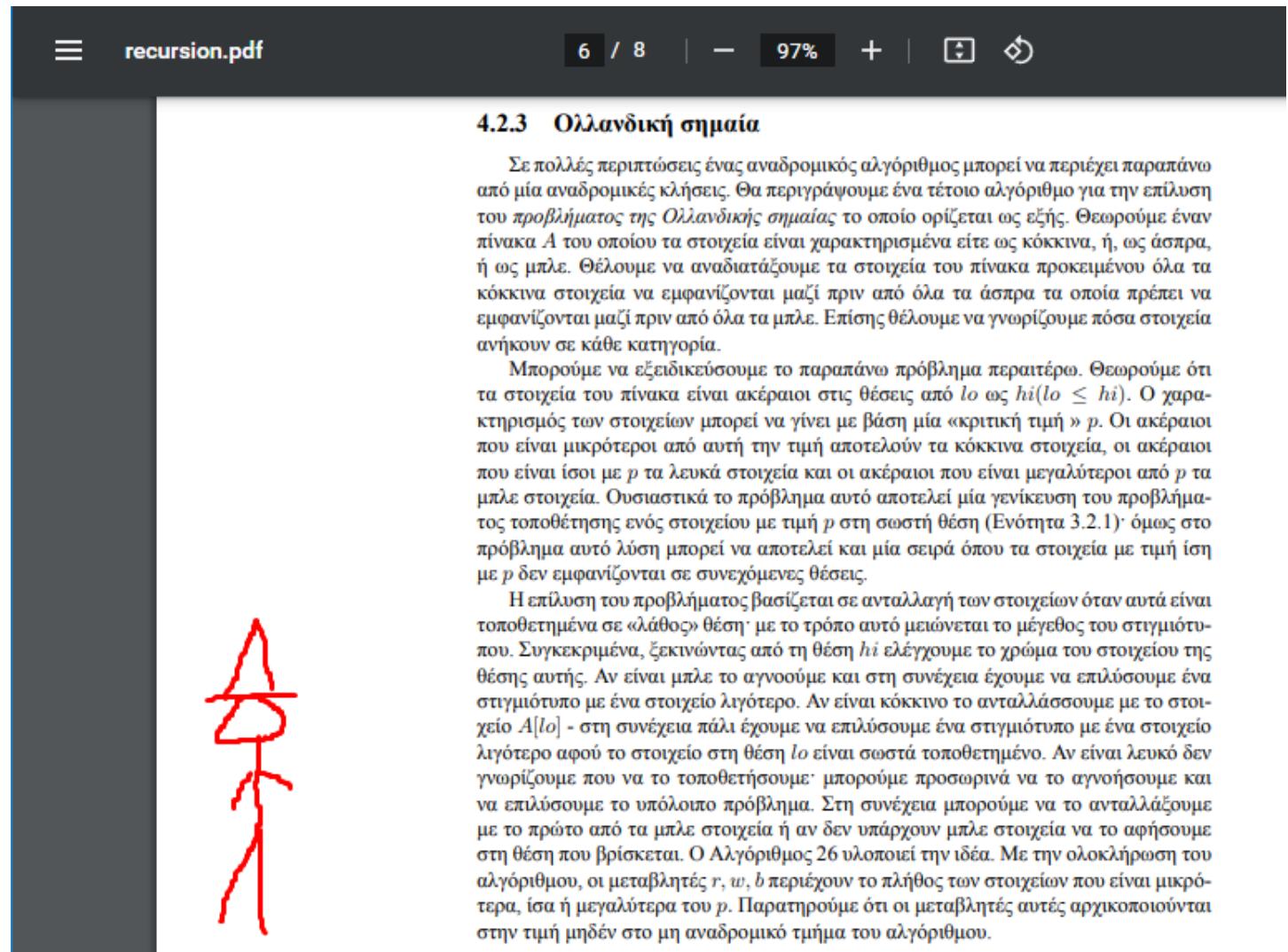
Function k-Element (int A[], int lo, int hi)
 if $lo = hi$ then } Πάντα στας ανδρούμενις αλγόριθμος
 return A[lo]; } Σε κάποια να μην πάρει συστήκη τερματισμού.
 end-if
 $n \leftarrow hi - lo + 1;$ → ολίγους των στοιχείων
 $pvt \leftarrow Rand(n) + lo;$ →
 $j \leftarrow Partition(A, lo, hi, pvt);$
 $i \leftarrow j - lo + 1;$
 if $i = k$ then
 return A[j];
 else if $i > k$ then
 return k-Element(A, lo, j-1, k);
 else
 return k-Element(A, i+1, hi, k-i);
 end-if
 end-function

Chapter 9d

Tip Ο κώδικας της Partition εμπεριέχεται στην διαφάνεια [notes\03_Πίνακες και άλλες δομές.pdf](#) (σελ. 3) και ουσιαστικά ταξινομεί ένα στοιχείο στον πίνακα (το βαζει στην σωστή θέση ταξινόμησης). Είναι τυχαιοκρατικός ο αλγόριθμος δεν θα εμβαθύνει σε τέτοιους απλά του αρέσει με καμμένους αλγορίθμους να διώχνει κόσμο από τις διαλέξεις για να έχει ησυχία

The Netherlands flag

[notes\04 Αναδρομή.pdf σελ. 6-7](#)



A screenshot of a PDF viewer window titled "recursion.pdf". The page number "6 / 8" is at the top right, along with zoom controls (97%, +, -) and a search icon. The main content is section 4.2.3 titled "Ολλανδική σημαία". It contains text about the recursive definition of the Dutch flag and a red hand-drawn stick figure with a triangle hat and a circle body.

4.2.3 Ολλανδική σημαία

Σε πολλές περιπτώσεις ένας αναδρομικός αλγόριθμος μπορεί να περιέχει παραπάνω από μία αναδρομικές κλήσεις. Θα περιγράψουμε ένα τέτοιο αλγόριθμο για την επίλυση του προβλήματος της Ολλανδικής σημαίας το οποίο ορίζεται ως εξής. Θεωρούμε έναν πίνακα A του οποίου τα στοιχεία είναι χαρακτηρισμένα είτε ως κόκκινα, ή, ως άσπρα, ή ως μπλε. Θέλουμε να αναδιατάξουμε τα στοιχεία του πίνακα προκειμένου όλα τα κόκκινα στοιχεία να εμφανίζονται μαζί πριν από όλα τα άσπρα τα οποία πρέπει να εμφανίζονται μαζί πριν από όλα τα μπλε. Επίσης θέλουμε να γνωρίζουμε πόσα στοιχεία ανήκουν σε κάθε κατηγορία.

Μπορούμε να εξειδικεύσουμε το παραπάνω πρόβλημα περαιτέρω. Θεωρούμε ότι τα στοιχεία του πίνακα είναι ακέραιοι στις θέσεις από lo ως hi ($lo \leq hi$). Ο χαρακτηρισμός των στοιχείων μπορεί να γίνει με βάση μία «κριτική τιμή» p . Οι ακέραιοι που είναι μικρότεροι από αυτή την τιμή αποτελούν τα κόκκινα στοιχεία, οι ακέραιοι που είναι ίσοι με p τα λευκά στοιχεία και οι ακέραιοι που είναι μεγαλύτεροι από p τα μπλε στοιχεία. Ουσιαστικά το πρόβλημα αυτό αποτελεί μία γενίκευση του προβλήματος τοποθέτησης ενός στοιχείου με τιμή p στη σωστή θέση (Ενότητα 3.2.1): όμως στο πρόβλημα αυτό λόστη μπορεί να αποτελεί και μία σειρά όπου τα στοιχεία με τιμή ίση με p δεν εμφανίζονται σε συνεχόμενες θέσεις.

Η επίλυση του προβλήματος βασίζεται σε ανταλλαγή των στοιχείων όταν αυτά είναι τοποθετημένα σε «λάθος» θέση: με το τρόπο αυτό μειώνεται το μέγεθος του στιγμιότυπου. Συγκεκριμένα, ξεκινώντας από τη θέση hi ελέγχουμε το χρώμα του στοιχείου της θέσης αυτής. Αν είναι μπλε το αγνοούμε και στη συνέχεια έχουμε να επιλύσουμε ένα στιγμιότυπο με ένα στοιχείο λιγότερο. Αν είναι κόκκινο το ανταλλάσσουμε με το στοιχείο $A[lo]$ - στη συνέχεια πάλι έχουμε να επιλύσουμε ένα στιγμιότυπο με ένα στοιχείο λιγότερο αφού το στοιχείο στη θέση lo είναι σωστά τοποθετημένο. Αν είναι λευκό δεν γνωρίζουμε που να το τοποθετήσουμε: μπορούμε προσωρινά να το αγνοήσουμε και να επιλύσουμε το υπόλοιπο πρόβλημα. Στη συνέχεια μπορούμε να το ανταλλάξουμε με το πρώτο από τα μπλε στοιχεία ή αν δεν υπάρχουν μπλε στοιχεία να το αφήσουμε στη θέση που βρίσκεται. Ο Αλγόριθμος 26 υλοποιεί την ιδέα. Με την ολοκλήρωση του αλγόριθμου, οι μεταβλητές r , w , b περιέχουν το πλήθος των στοιχείων που είναι μικρότερα, ίσα ή μεγαλύτερα του p . Παρατηρούμε ότι οι μεταβλητές αυτές αρχικοποιούνται στην τιμή μηδέν στο μη αναδρομικό τμήμα του αλγόριθμου.



Αλγόριθμος 26 Αλγόριθμος για το πρόβλημα της Ολλανδικής Σημαίας

Απαιτείται: Πίνακας A με στοιχεία στις θέσεις από lo ως hi ($lo \leq hi$), κριτική τιμής p .

Επιστρέφεται: Αναδιάταξη των στοιχείων του A με τα στοιχεία που είναι μικρότερα του p (κόκκινα) να εμφανίζονται στις πρώτες θέσεις, τα στοιχεία που είναι μεγαλύτερα του p (μπλε) στις τελευταίες θέσεις και τα στοιχεία που είναι ίσα με p (λευκά) μετά από τα στοιχεία που είναι μικρότερα του p και πριν από τα στοιχεία που είναι μεγαλύτερα του p . Το πλήθος των κόκκινων, λευκών και μπλε στοιχείων επιστρέφονται στις μεταβλητές r , w , b , αντίστοιχα.

```
1: function Dutch_Flag(int p, int A[], int lo, int hi, int r,int w, int b)
2:   if lo > hi then
3:     r ← 0; w ← 0; b ← 0;
4:   else
5:     if A[hi] < p then
6:       Swap(A[lo], A[hi]);
7:       Dutch_Flag(p, A, lo + 1, hi, r, w, b);
8:       r++;
9:     else if A[hi] = p then
10:      Dutch_Flag(p, A, lo, hi - 1, r, w, b);
11:      Swap(A[lo + r + w], A[hi]);
12:      w++;
13:    else
14:      Dutch_Flag(p, A, lo, hi - 1, r, w, b);
15:      b++;
16:    end if
17:   end if
18: end function
```

Palindrome

[exercises\Ασκήσεις σε αναδρομή.pdf](#) σελ. 2-3

≡ da_rec_exer.pdf 2 / 7 | - 100% + | ☰ ⌂



Άσκηση 2. Μία συμβολοσειρά αποτελούμενη από ψηφία του συνόλου {0, 1} ονομάζεται παλίνδρομη αν ταυτίζεται με την αντίστροφή της. Να εκπονήσετε αλγόριθμο ο οποίος να δέχεται σαν είσοδο μία σειρά από αριθμούς και να αποφασίζει αν η συμβολοσειρά αυτή είναι παλίνδρομη. Να διατυπώσετε την αναδρομική σχέση που περιγράφει τον αριθμό των ΣΥΒ.

function Palindrome (int A[], int lo, int hi)

if lo < hi then (+) Ταύτωση

 if A[lo] = A[hi] Η ανισορία καλείται για n-2 στοιχίων

 return Palindrome(A, lo+1, hi-1); Έξοι

 else hi - l - (lo + 1) + 1 στοιχίων πάνω
 return False; n = hi - lo + 1 στοιχίων των A[i]

end-if

else

 return True;

end-if

end-function

! Το n είναι το μήκος των αντιμετών, δηλαδή, των αριθμών των χαρακτήρων της συρβυτούσερίας

$\bar{T}(n) = \begin{cases} \bar{T}(n-2) + \Theta(1), & n \geq 2 \\ \Theta(1), & n = 1 \end{cases}$ \Leftrightarrow

$T(n) \leq \bar{T}(n)$, $n \geq 1$ Δεν γνωρίζω αρκετά το $\bar{T}(n)$



Άσκηση 3. Μία σειρά από ακεραίους a_1, \dots, a_n ονομάζεται μονότροπη (*unimodal*) αν υπάρχει ένα στοιχείο a_k , $1 \leq k \leq n$, για το οποίο ισχύει

$$a_1 < a_2 < \dots < a_{k-1} < a_k > a_{k+1} > \dots > a_{n-1} > a_n.$$

Από τον παραπάνω ορισμό προκύπτει ότι το σημείο καμπής μπορεί να είναι το πρώτο ή το τελευταίο στοιχείο της σειράς. Το στοιχείο a_k ονομάζεται *σημείο καμπής*.

Δίνεται μία μονότροπη σειρά ακεραίων αποθηκευμένη στις θέσεις $1, \dots, n$ του πίνακα A . Να εκπονήσετε αναδρομικό αλγόριθμο που να επιστρέφει τη θέση του σημείου καμπής και να διατυπώσετε την αναδρομική σχέση που περιγράφει τον αριθμό των ΣΥΒ που εκτελούνται.

```

function Unimodal (int ACT, int lo, int hi)
    if lo = hi then
        return A[lo];
    end-if
    j ← ⌊ ⌈lo+hi⌉ / 2 ⌋; → Binary Search
    if A[j] < A[j+1] then
        return Unimodal(A, j+1, hi); • hi - (j+1) + 1 =
        else
            return Unimodal(A, lo, j); • hi - ⌊ ⌈lo+hi⌉ / 2 ⌋ + 1
        end-if
    end-function
    • n = hi - lo + 1 συνειδητά
    • hi - (j+1) + 1 = hi - ⌊ ⌈lo+hi⌉ / 2 ⌋ + 1
    • hi - lo - ⌊ ⌈lo+hi⌉ / 2 ⌋ + 1 = hi + lo - ⌊ ⌈lo+hi⌉ / 2 ⌋ + 1 = n - 1 + 2lo = n + 2lo - 1
    • hi + lo = ⌊ ⌈lo+hi⌉ / 2 ⌋ + lo
    • ⌊ ⌈lo+hi⌉ / 2 ⌋ ≥ ⌊ n / 2 ⌋
    T(n) ≤ { T(⌈ n / 2 ⌉) + Θ(1), n ≥ 2
              Θ(d)           n = d
  
```

ΘΕΩΡΙΑ

Εισαγωγή στις αναδρομικές σχέσεις

[notes\05_Αναδρομή \(Συμπληρωματικό Υλικό\).pdf](#) σελ. 1-4

recursion02.pdf 4 / 4 - 100% + ⊞ ◊

Η επιλυση των αναδρομικων συναρτησεων συνισταται στην ευρεση μιας εξισωσης η οποια θα εκφραζει τον αριθμο των ΣΥΒ σαν συνάρτηση μόνο της παραμέτρου n . Μια τέτοια μαθηματική σχέση ονομάζεται κλειστή μορφή της αναδρομικής συνάρτησης. Για παράδειγμα, η κλειστή μορφή της (4.4) είναι $T(n) = 4n + 1$.

Παρόλο που η κλειστή μορφή αποτελεί ζητούμενο μερικές φορές είναι αρκετά δύσκολο να υπολογιστεί. Όμως - όπως είδαμε και σε προηγούμενο κεφάλαιο - στην ανάλυση των αλγόριθμων δεν ενδιαφέρει τόσο ο απόλυτος αριθμός των ΣΥΒ αλλά η τάξη μεγέθους του (ασυμπτωτική εκτίμηση). Αυτό διευκολύνει αρκετά: όταν γράφουμε την αναδρομική συνάρτηση η συνιστώσα που αφορά τα ΣΥΒ που εκτελούνται σε κάθε

4.5. ΑΣΚΗΣΕΙΣ

71

κλήση του αλγόριθμου αρκεί να περιγράφεται ως τάξη μεγέθους και όχι απαραίτητα ως μία ακριβής έκφραση. Για παράδειγμα, αντί της (4.4), μπορούμε να γράψουμε την αναδρομική σχέση που περιγράφει τον αριθμό των ΣΥΒ που εκτελεί ο Αλγόριθμος 23 ως

$$T(n) = \begin{cases} T(n-1) + \Theta(1), & n \geq 1, \\ 1, & n = 0. \end{cases} \quad (4.6)$$

Ο όρος $\Theta(1)$ υποδηλώνει ότι σε κάθε κλήση του αλγόριθμου με $n \geq 1$ εκτελείται σταθερός αριθμός ΣΥΒ. Ο αριθμός αυτός δεν χρειάζεται να προσδιοριστεί επακριβώς προκειμένου να εκτιμηθεί η ασυμπτωτική συμπεριφορά του αλγόριθμου: σε κλειστή μορφή, ο αριθμός των ΣΥΒ που εκτελούνται δίνεται από τη συνάρτηση $T(n) = cn + 1$, όπου c μία θετική σταθερά μεγαλύτερη του μηδενός. Οπότε έχουμε ότι $T(n) = \Theta(n)$. Παρατηρείστε το ίδιο αποτέλεσμα πάινουμε και στην περίπτωση που έχουμε προσδιορίσει ότι $c = 4$.

Σε επόμενο κεφάλαιο θα παρουσιάσουμε μεθόδους επίλυσης αναδρομικών εξισώσεων και προσδιορισμού της ασυμπτωτικής συμπεριφοράς τους.



4.5 Ασκήσεις

1. Να εκπονήσετε αλγόριθμο ο οποίος να εξάγει το k -ιοτό στοιχείο από μία στοιβα με n στοιχεία, όπου $k \leq n$.
2. Το πλήθος των διαφορετικών ομάδων καθεμία αποτελούμενη από k αντικείμενα που μπορούν να σχηματιστούν από ένα σύνολο n διακριτών αντικειμένων συμ-

[Slides \(Παλαιό Υλικό\)](#)|[Αναδρομικές Συέσεις.pdf](#) σελ. 1-22

Email:dmagos@uniwa.gr

22 / 40 | - 75% + | ☰ 🔍 19 / 34

Παρατηρήσεις

Χρησιμότητα
Η Δύναμη της Αναδρομής

Πόλυπλοκότητα

Εικασία

Αντικατάσταση

- Μέθοδος Επαναληπτικής Αντικατάστασης
- Παράδειγμα
- Παρατηρήσεις

Δένδρο Αναδρομής

Κεντρικό Θεώρημα

Από την ανάλυση που κάνομε είναι φανερό ότι ο υπολογισμός του ακριβούς αριθμού των στοιχειωδών πράξεων σε κάθε κλήση του αναδρομικού αλγόριθμου δεν είναι απαραίτητος για την κατάταξη του αλγορίθμου στην ιεραρχία πολυπλοκότητας. Ακόμα και αν θεωρούσαμε ότι σε κάθε κλήση εκτελούνται ένας **σταθερός** αριθμός στοιχειωδών πράξεων, έστω c (χωρίς να προσδιορίζαμε την ακριβή τιμή του c), θα προέκυπτε ότι $T(n) = c \cdot n + 1$ και άρα πάλι $T(n) = \Theta(n)$.

Με αντίστοιχο τρόπο παρατηρούμε ότι δεν είναι απαραίτητο να υπολογίσουμε επακριβώς τον αριθμό των στοιχειωδών πράξεων που εκτελεί ο αλγόριθμος όταν παύει να είναι αναδρομικός.

Email:dmagos@uniwa.gr

20 / 34

Ορισμοί

Χρησιμότητα
Η Δύναμη της Αναδρομής

Πόλυπλοκότητα

Εικασία

Αντικατάσταση

Ορισμός 5 Ένα δένδρο είναι ένα άκυκλο συνδεδεμένο γράφημα.

Παράδειγμα 6 Ένα δένδρο με 8 κόμβους απεικονίζεται στο

[notes\06 Αναδρομικές Συέσεις.pdf σελ. 1-7](#)

rec_relations.pdf 7 / 16 | - 100% + | ☰ ⌂

οποιοδήποτε μονοπάτια από τη ρίζα σε φύλλο αποτελεί το ύψος του δένδρου. Στο μονοπάτι αυτό η κάθε ακμή αντιστοιχεί σε μία διαιρεση του μεγέθους του στιγμάτου που με το 3. Άρα το ύψος του δένδρου, έστω h , μας δίνει μικρότερο αριθμό διαιρέσεων του n με το 3 προκειμένου να φτάσουμε σε τιμή μικρότερη-ίση με 1. Δηλαδή,

$$\frac{n}{3^h} \leq 1 \Rightarrow n \leq 3^h \Rightarrow h \geq \lg_3 n \Rightarrow h = \lceil \lg_3 n \rceil. \quad (5.12)$$

Παρατηρούμε ότι η (5.12) δίνει το ύψος του δένδρου όπως αυτό προκύπτει από την (5.9), η οποία ισχύει σαν ισότητα (αφού το δένδρο είναι πλήρες δυαδικό), με αριθμό κορυφών $d = 2^{\lceil \lg_3 n \rceil} + 1$. Στο Σχήμα 5.5 σημειώνεται το ύψος του δένδρου καθώς και ο αριθμός των ΣΥΒ που εκτελούνται σε κάθε ύψος (ποσότητες στο αριστερό περιθώριο του σχήματος).

Μετρώντας από επάνω προς τα κάτω, θεωρούμε τον δείκτη βάθους $j \in \{0, \dots, \lceil \lg_3 n \rceil\}$ - σε βάθος $j = 0$ βρίσκεται η ρίζα του δένδρου που συνεισφέρει n στο συνολικό αριθμό των, σε βάθος $j = 1$ οι δύο κορυφές που συνεισφέρουν η καθεμία $\frac{n}{3}$, κλπ. Δηλαδή, ο αριθμός των ΣΥΒ σε κάθε βάθος είναι

$$\begin{aligned} & \left(\frac{2}{3}\right)^j \cdot n, \quad j = 0, \dots, \lceil \lg_3 n \rceil - 1, \\ & 2^j \cdot 1, \quad j = \lceil \lg_3 n \rceil. \end{aligned}$$

Αθροίζοντας, για όλες τις τιμές του δείκτη j , έχουμε

$$\begin{aligned} T(n) &= n \sum_{j=0}^{\lceil \lg_3 n \rceil - 1} \left(\frac{2}{3}\right)^j + 2^{\lceil \lg_3 n \rceil} \\ &\leq n \sum_{j=0}^{\lceil \lg_3 n + 1 \rceil - 1} \left(\frac{2}{3}\right)^j + 2^{\lceil \lg_3 n + 1 \rceil} \\ &= n \sum_{j=0}^{\lg_3 n} \left(\frac{2}{3}\right)^j + 2 \cdot 2^{\lg_3 n} \end{aligned}$$

Επομένως,

$$\begin{aligned} T(n) &\leq n \sum_{j=0}^{\infty} \left(\frac{2}{3}\right)^j + 2 \cdot n^{\lg_3 2} \\ &= \frac{1}{1 - \frac{2}{3}} n + 2n^{\lg_3 2} = 3n + 2n^{\lg_3 2} \\ &\leq 3n + 2n \\ &T(n) \leq 5n. \end{aligned}$$


Από την τελευταία ανισότητα έχουμε ότι $T(n) = O(n)$.

Μπορούμε να γενικεύσουμε την ανάλυση που πραγματοποιήθηκε ως εξής. Έστω

$$T(n) = \begin{cases} aT\left(\frac{n}{b}\right) + cn, & \text{για } n > 1, \\ 1, & \text{για } n \leq 1, \end{cases} \quad (5.13)$$

[notes/06_Αναδρομικές Σγέσεις.pdf σελ. 7-16](#)

rec_relation...

16 / 16

—

100%

+

↶

↷

⤵

⤶

⋮

με την $T(n-1)$. Στη συνεχεία θετούμε $t(n) = T(n)/(n+1)$ και οι παραπάνω σχέση γίνεται

$$t(n) = t(n-1) + \frac{2}{n+1} - \frac{1}{n(n+1)} = t(n-1) + \frac{3}{n+1} - \frac{1}{n}.$$

Με διαδοχική αντικατάσταση και θεωρώντας ότι $t(0) = 1$ η παραπάνω σχέση συνεπάγεται

$$\begin{aligned} t(n) &= 1 + \sum_{j=2}^{n+1} \frac{3}{j} - \sum_{j=1}^n \frac{1}{j} \\ &= \frac{3}{n+1} - 2 + \sum_{j=1}^n \frac{3}{j} - \sum_{j=1}^n \frac{1}{j} \\ &= \frac{3}{n+1} - 2 + 2 \sum_{j=1}^n \frac{1}{j}. \end{aligned}$$

Επειδή $\frac{3}{n+1} - 2 = \Theta(1)$ και $\sum_{j=1}^n \frac{1}{j}$ είναι ο αρμονικός μέσος ο οποίος είναι τάξης $\Theta(\lg n)$, έχουμε ότι $t(n) = \Theta(\lg n)$. Αντικαθιστώντας, από τη σχέση $t(n) = T(n)/(n+1)$ προκύπτει ότι

$$T(n) = \Theta(n \lg n).$$



ΑΣΚΗΣΕΙΣ

Ασκηση στο Θεώρημα κυριαρχικού όρουν

[exercises\Ασκήσεις σε αναδρομικές σγέσεις \(eClass\).pdf](#) σελ. 10-11

10 / 22 | - 125% + | ☰ ⚡

Άσκηση 4 Μπορείτε με τη χρήση του κεντρικού θεωρήματος να δώσετε μια ασυμπτωτική εκτίμηση για τη σχέση

$$T(n) = 2T\left(\frac{n}{2}\right) + n \lg n;$$

Αν ναι να την διατυπώσετε. Αν όχι να χρησιμοποιείστε άλλη μέθοδο.



6xer

$$T(n) = 4 T\left(\frac{n}{2}\right) + n \cdot \text{ερη}$$

Chapter 15a

Με Οινόρημα κυριαρχικών όρων

$$g_b a = 2 \Leftrightarrow a = 4, b = 2, f(n) = n \cdot \lg n$$

?

$$\lim_{n \rightarrow \infty} \frac{f(n)}{n^{2-\varepsilon}} = \lim_{n \rightarrow \infty} \frac{n \cdot \lg n}{n^{2-\varepsilon}} = \lim_{n \rightarrow \infty} \frac{\lg n}{n^{1-\varepsilon}} = \lim_{n \rightarrow \infty} \frac{\frac{1}{n \cdot \lg 2}}{\frac{1-\varepsilon}{n^\varepsilon}} = \\ = \lim_{n \rightarrow \infty} \frac{1}{(1-\varepsilon) n^{1-\varepsilon} \lg 2} = 0 \quad \text{με } 1 > \varepsilon < 0 \Rightarrow$$

$$f(n) = o(n^{2-\varepsilon}) \Rightarrow f(n) = O(n^{2-\varepsilon})$$

$$T(n) = O(n^2)$$

$$\text{Αν } \lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = 0 \Rightarrow f(n) = o(g(n))$$

Άσκηση στη Μέθοδο της Εικασίας με επαγωγή

exercises\Ασκήσεις σε αναδρομικές σχέσεις (Mega).pdf σελ. 6-9

Συνολικά θεωρείται ότι $T(n) = \Theta(n^2)$.

Άσκηση 4 Να δώσετε μία ασυμπτωτική εκτίμηση για τη σχέση

$$T(n) = \begin{cases} 4T(\lfloor \frac{n}{2} \rfloor) + \Theta(n), & n \geq 2, \\ 1, & n \leq 1. \end{cases}$$


8/5/2021

Exer.

$$T(n) = \begin{cases} 4T(\lfloor \frac{n}{2} \rfloor) + \Theta(n), & n \geq 2 \\ 1, & n \leq 1 \end{cases}$$

Chapter 13b

Με μέρους της Euclidian και Enaywif's $\bar{T}(n) \leq \bar{f}(n)$

Οu λειώ την αναρτωτική σύριγξεωρία των

$$\bar{T}(n) = 4\bar{T}\left(\frac{n}{2}\right) + \Theta(n) = 4\bar{T}\left(\frac{n}{2}\right) + c \cdot n$$

$a = 4$, $b = 2$, $\log_b a = 2$, $f(n) = c \cdot n$

$c \cdot n = O(n^{2-\varepsilon})$, $1 > \varepsilon > 0$ | $\Rightarrow \bar{T}(n) = \Theta(n^2)$

Οσιόργανη
κυριαρχία
δύνα

Աղքա $T(n) = O(n^2)$ $\Rightarrow T(n) \leq a \cdot n^2$ $a > 0$, $n \geq n_0 > 0$

Երացրեն Կամ $T(\lfloor \frac{n}{2} \rfloor) \leq a \lfloor \frac{n}{2} \rfloor^2 \Rightarrow$

$$4T\left(\lfloor \frac{n}{2} \rfloor\right) + cn \leq 4a \left(\frac{n}{2}\right)^2 + cn \Rightarrow$$

$$T(n) \leq 4 \cdot a \left(\frac{n}{2}\right)^2 + cn \leq 4a \left(\frac{n}{2}\right)^2 + cn = \\ = \underline{an^2 + cn}$$

Երացրեն Բիռ: $T(n) \leq a \cdot n^2$ ՃEN ԼԵԿԵՐԸ

$$T(n) \leq an^2 + cn$$

O.S.O. $T(n) \leq an^2 - 2b \cdot n$ ✓, $a > 0$, $b > 0$, $n \geq n_0 > 0$

Նեա Երացրեն Կամ: $T(\lfloor \frac{n}{2} \rfloor) \leq a \left(\frac{n}{2}\right)^2 - 2b \left(\frac{n}{2}\right)$

$$\Rightarrow 4T\left(\lfloor \frac{n}{2} \rfloor\right) + cn \leq 4a \left(\frac{n}{2}\right)^2 - 2b \left(\frac{n}{2}\right) + cn \Rightarrow$$

$$T(n) \leq 4 \left(a \left(\frac{n}{2}\right)^2 - 2b \left(\frac{n}{2} - 1\right)\right) + cn = \\ = an^2 - (3b - c)n + b(8 - n) \Rightarrow$$

$T(n) \leq an^2 - (3b - c) \cdot n$ յստ $n \geq 8$ $\Rightarrow \begin{cases} \text{այսուհետ} \\ \text{անընդունելիք} \end{cases}$

$T(n) \leq an^2 - 2b \cdot n$ ✓ յստ $b \geq c$ $\begin{cases} \text{բարեփակ} \\ \text{բարեփակ} \end{cases}$

$$an^2 - (3b - c) \cdot n \leq an^2 - 2b \cdot n \Rightarrow b \geq c$$

Խան Երացրեն $T(n) = O(n^2)$

$$T(n) = \Omega(n^2) \Rightarrow T(n) \geq an^2, \quad a > 0, n_0 > 0$$

E.Y. $T(\lfloor \frac{n}{2} \rfloor) \geq \alpha \left[\frac{n^2}{4} \right] \Rightarrow$

$$T(n) \geq 4\alpha \left[\frac{n}{2} \right]^2 + cn \Rightarrow$$

$$T(n) \geq 4\alpha \left(\frac{n}{2} - 1 \right)^2 + cn = an^2 - 4an + cn + cn$$

$$T(n) \geq an^2 - 4an + cn \geq an^2 \quad \text{f.t. } a \leq \frac{c}{4}$$

Άριστα $T(n) = \Omega(n^2)$ $a > 0!$

① + ② $\Rightarrow \boxed{T(n) = \Theta(n^2)}$

Άσκηση στο Δένδρο Αναδρομής

[exercises\Άσκησεις σε αναδρομικές σχέσεις \(eClass\).pdf](#) σελ. 3-7 (Παρόμοια εκφώνηση)

[exercises\Άσκησεις σε αναδρομικές σχέσεις \(Mega\).pdf](#) σελ. 4-6 (Παρόμοια εκφώνηση)

3 / 22 | - 125% + ⊞

Άσκηση 2 Χρησιμοποιώντας το δένδρο αναδρομής να προσδιορίσετε ένα ασυμπτωτικό φράγμα για τις σχέσεις

1. $T(n) = 4T(\lfloor \frac{n}{2} \rfloor) + \Theta(n)$

2. $T(n) = 8T(\frac{n}{2}) + n^2$, με n δύναμη του 2.

Και στις δύο περιπτώσεις θεωρείστε ότι $T(1) = 1$.

RecRelExer_vr04.pdf x +

← ⌂ File | C:/Users/billa/Downloads/RecRelExer_vr04.pdf

≡ | ⌂ | ⌂ Draw | ⌂ | ⌂ | Read aloud - + ⊞ | 4 of 22 | ⌂ | ⌂

Άσκηση 3 Θεωρώντας ότι $n = 2^t$, $t \geq 1$ και με τη χρήση του δένδρου αναδρομής να δώσετε μία ασυμπτωτική εκτίμηση για τις σχέση

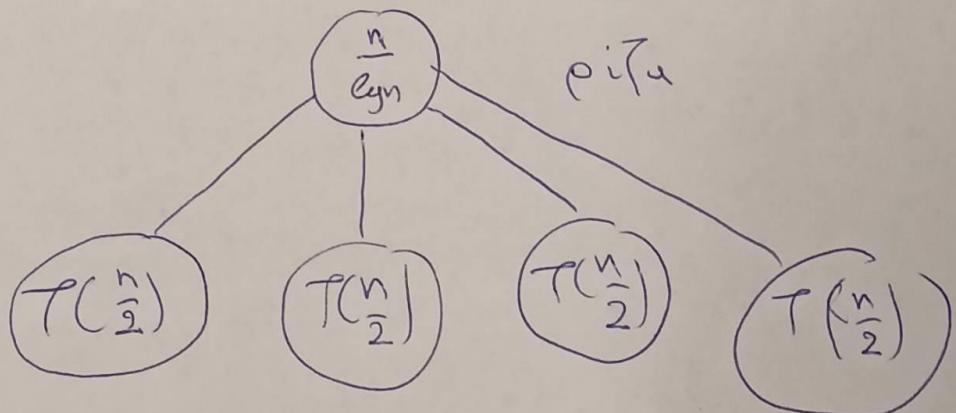
$$T(n) = \begin{cases} 8T(\frac{n}{2}) + n^2, & n \geq 2, \\ 1, & n = 1. \end{cases}$$

6ερ

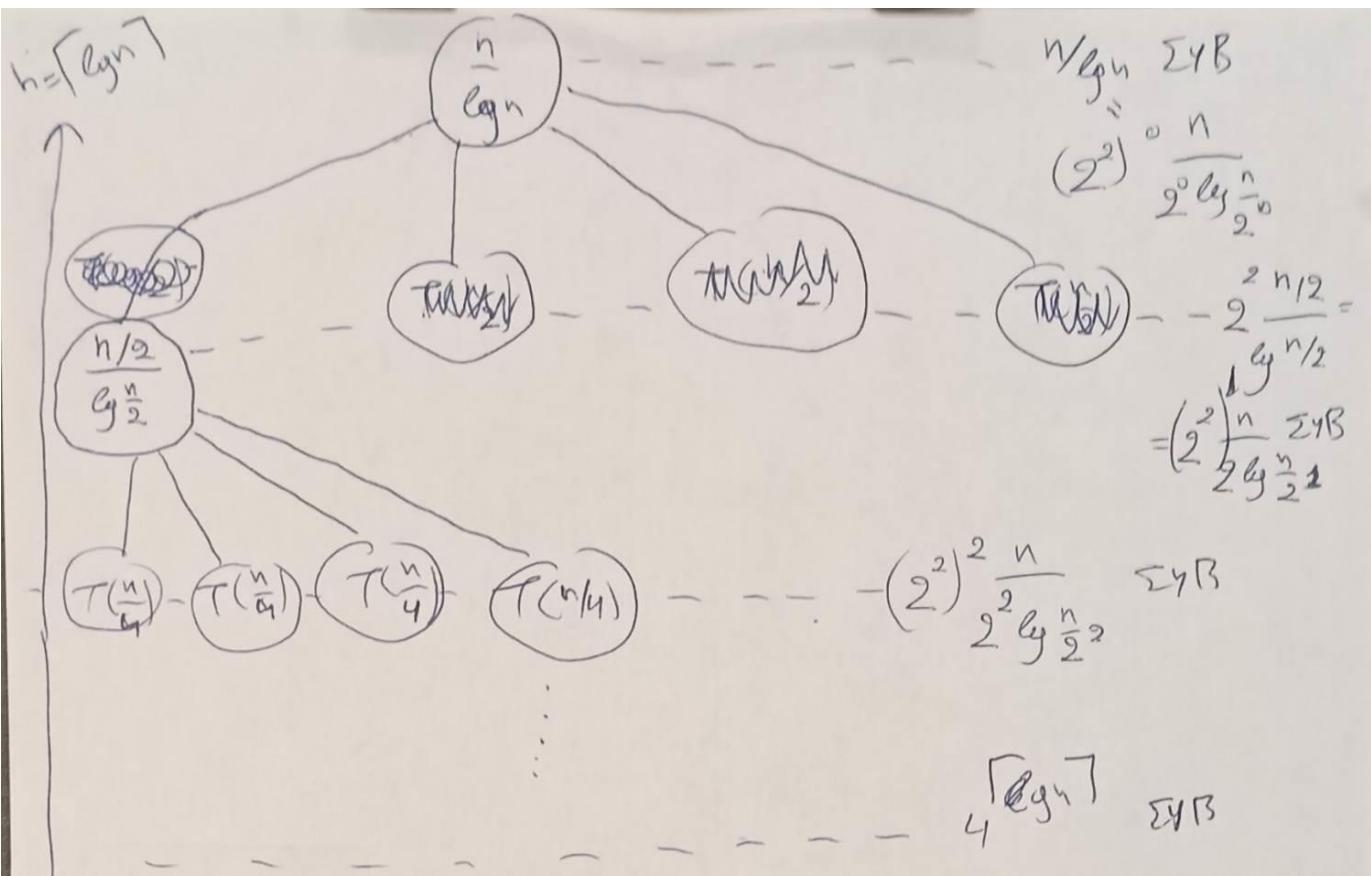
$$T(n) = \begin{cases} 4T\left(\frac{n}{2}\right) + n/\lg n, & n \geq 2 \\ 1, & n \leq 1 \end{cases}$$

Chapter 13c

Με τη ρίζα του δένδρου άντεχομες



④ εύλλαγμα



$$f(k) = \begin{cases} \left(2^2\right)^k \frac{n}{2^k \log \frac{n}{2^k}}, & 0 \leq k \leq \lceil \log n \rceil \\ n^{\lceil \log n \rceil}, & k = \lceil \log n \rceil \end{cases}$$

$$T(n) = 4^{lgn} + 4 \sum_{k=0}^{\lceil lgn \rceil - 1} (2^2)^k \frac{n}{2^k} = 4^{lgn} + 4 \sum_{k=0}^{\lceil lgn \rceil - 1} 4^k n = 4^{lgn} + n \sum_{k=0}^{\lceil lgn \rceil - 1} 4^k = 4^{lgn} + n(4^{\lceil lgn \rceil} - 1) / (4 - 1)$$

$$T(n) = 4^{\lceil \lg n \rceil} + n \left(\frac{2^0}{\lg n - 0} + \frac{2^1}{\lg n - 1} + \frac{2^2}{\lg n - 2} + \dots + \frac{2^{\lceil \lg n \rceil - 1}}{\lg n - (\lceil \lg n \rceil - 1)} \right)$$

$$\begin{aligned}
 ① \Rightarrow T(n) &\leq 4^{log_2 n + 1} + n \left(2^0 + 2^1 + \dots + 2^{log_2 n - 1} \right) \\
 &\leq 4n^2 + n \left(2^{log_2 n - 1} \right) \leq 4n^2 + n \left(2^{log_2 n + 1 - 1} \right) \\
 &= 4n^2 + n \left(2 \cdot 2^{log_2 n} - 1 \right) = 4n^2 + 2n^2 - n \\
 &= 6n^2 - n \leq c \cdot n^2
 \end{aligned}$$

Άρετα $T(n) = O(n^2)$

$$\begin{aligned} T(n) &\geq 4^{\lg n} + n \left(\frac{2^0}{\lg n} + \frac{2^1}{\lg n} + \dots + \frac{2^{\lg n-1}}{\lg n} \right) \\ &= n^2 + \frac{n}{\lg n} \left(2^0 + 2^1 + \dots + 2^{\lg n-1} \right) \\ &= n^2 + \frac{n}{\lg n} (2^{\lg n} - 1) = n^2 + \frac{n}{\lg n} (n-1) \geq n^2 \end{aligned}$$

$T(n) = \Omega(n^2)$

, $\Rightarrow T(n) = \Theta(n^2)$

Άσκηση στη Μέθοδο της Αντικατάστασης (Μετατροπές)

[exercises\Άσκήσεις σε αναδρομικές σχέσεις \(eClass\).pdf](#) σελ. 11-12

11 / 22 | - 125% + | ☰ ⌂

Άσκηση 5 Να δώσετε μία ασυμπτωτική εκτίμηση για τη συνάρτηση

$$T(n) = T\left(\frac{n}{3}\right) + T\left(\frac{n}{5}\right) + n.$$



$$T(n) = 2T\left(\frac{n}{4}\right) + 3T\left(\frac{n}{6}\right) + n \lg n \quad ①$$

Chapter 13d

MEHREREN RECURSIEVERGELIJKINGEN (MERGE ONES)

DITW

$$U(n) = \frac{1}{n} T\left(\frac{n}{2}\right) \Rightarrow U(2n) = \frac{1}{2n} T(n)$$

$$①: T(n) = 2T\left(\frac{n/2}{2}\right) + 3T\left(\frac{n/2}{3}\right) + n \lg n \quad \text{#}$$

$$\frac{1}{2n} T(n) = \frac{1}{2} \cdot \frac{1}{n/2} T\left(\frac{n/2}{2}\right) + \frac{1}{2} \cdot \frac{1}{n/3} T\left(\frac{n/2}{3}\right) + \frac{1}{2} \lg n$$

$$U(n) = \frac{1}{2} (U(n/2) + U(n/3) + \lg n) \quad ②$$

$$g(n) = 2^n \quad ③$$

$$S(n) = U(g(n)) \quad ④$$

$$②, ③, ④: S(n) = \frac{1}{2} S\left(\frac{n}{4}\right) + \frac{1}{2} S\left(\frac{n}{6}\right) + \frac{1}{2} \lg n \quad ⑤$$

$$⑤: S(n) \leq S\left(\frac{n}{4}\right) + \frac{1}{2} \lg n \quad \begin{matrix} 2^{\text{#}} \text{ RECURSIES} \\ \text{Onderstaande complexiteit} \end{matrix} \Rightarrow S(n) = O(\lg^2 n)$$

$$⑤: S(n) \geq S\left(\frac{n}{6}\right) + \frac{1}{2} \lg n \quad \begin{matrix} 2^{\text{#}} \text{ RECURSIES OK} \\ = \end{matrix} S(n) = \Omega(\lg^2 n)$$

$$\text{Aan} \quad S(n) = \Theta(\lg^2 n) \quad =$$

$$U(2n) = \Theta(\lg^2 2n) = \Theta(\lg^2 n) = U(n)$$

...?

Στην λύση γράφω "2η περίπτωση ΘK" είναι η μεσαία περίπτωση του Θεωρήματος Κυριαρχικού όρου

Οινόργανη κυριαρχίας

$$T(n) = aT\left(\frac{n}{b}\right) + f(n)$$

$$a > 0, \quad b > 1, \quad f(n)$$

$$T(n) = \begin{cases} \Theta(n^{\lg_b a}), & f(n) = O(n^{\lg_b a - \varepsilon}), \varepsilon > 0 \\ \Theta(n^{\lg_b a} \cdot \lg^{k+1} n), & \text{av } f(n) = \Theta(n^{\lg_b a} \cdot \lg^k n) \quad k \geq 0 \\ \Theta(f(n)), & \text{av } f(n) = \Omega(n^{\lg_b a + \varepsilon}), \varepsilon > 0 \\ & af(n/b) \leq cf(n), c < 1 \end{cases}$$

ΘΕΩΡΙΑ

Εισαγωγή στο ΔκΒ, Βασικές αρχές, Quicksort

[notes\07_Διαιρει και Βασιλευε.pdf](#) σελ. 1-6

[extraNotes\QuickSort ΔκΒ.pdf](#) έξτρα σημειώσεις

5 / 20 | - 125% + | ☰ ⌂

$$\lg(n-3) - 1 \geq \lg n - 3. \quad (7.5)$$

Από (7.4), (7.5) έχουμε

$$\begin{aligned} \bar{T}(n) &\geq a(n-3)(\lg n - 3) + cn = an \lg n - 3a \lg n - 3an + 9a + cn \\ &\geq an \lg n - 3a \lg n - 3an + cn \\ &\geq an \lg n - 3an - 3an + cn = an \lg n - (6a - c)n \Rightarrow \\ \bar{T}(n) &\geq an \lg n, \end{aligned}$$

$$\text{για } 6a - c \leq 0 \Rightarrow a \leq \frac{c}{6}.$$

Άρα $\bar{T}(n) = \Omega(n \lg n)$ και συνεπώς $T(n) = \Omega(n \lg n)$

2

$$\lg(n-3) - 1 \geq \lg n - 3 \Rightarrow \lg(n-3) \geq \lg n - 2 \Rightarrow \lg(n-3) \geq \lg \frac{n}{4} \Rightarrow n-3 \geq n/4 \Rightarrow n \geq 4.$$

Πόρισμα 6. Ο αριθμός των ΣΥΒ που εκτελεί ο αλγόριθμος της ταχυταξινόμησης, για οποιοδήποτε στιγμιότυπο μεγέθους n , είναι τάξης $O(n^2)$ και $\Omega(n \lg n)$.



7.2.2 Συγχωνευτική ταξινόμηση

Μία από τις πλέον χαρακτηριστικές περιπτώσεις αλγορίθμου που εκφράζει το πνεύμα της μεθοδολογίας «Διαιρει και Κυρίευε» είναι η συγχωνευτική ταξινόμηση (*merge sort*).

[notes\07_Διαιρεί και Βασίλευε.pdf σελ.6-9](#)

[extraNotes\MergeSort ΔκΒ.pdf έξτρα σημειώσεις](#)

9 / 20 | - 125% + | ☰ ⌂

$$Q = \begin{bmatrix} CE + DG & CF + DH \end{bmatrix}. \quad (7.11)$$

Η παραπάνω παράσταση παραπέμπει σε αλγορίθμική υλοποίηση τύπου «Διαιρεί και Κυρίευε» αφού ο υπολογισμός του γινομένου δύο πινάκων διάστασης $n \times n$ μπορεί να γίνει με βάση τα γινόμενα οκτώ πινάκων διάστασης $\frac{n}{2} \times \frac{n}{2}$ τα οποία συνδυάζονται με τέσσερις πράξεις πρόσθεσης πινάκων αντίστοιχης διάστασης. Η αναδρομική σχέση που ικανοποιεί ο αριθμός των ΣΥΒ είναι

$$T(n) = \begin{cases} \Theta(1), & n = 1, \\ 8T(\frac{n}{2}) + cn^2, & n > 1 \end{cases}$$

Όμως και πάλι ο αριθμός των ΣΥΒ για τον υπολογισμό του πίνακα Q , όπως προκύπτει από το Θεώρημα 1 (πρώτη περίπτωση) είναι τάξης $\Theta(n^3)$. Στην εργασία [18] χρησιμοποιούνται οι υποπίνακες A, \dots, H που ορίστηκαν προηγουμένως προκειμένου να επιτευχθεί ο υπολογισμός σε ασυμπτωτικά καλύτερο χρόνο. Συγκεκριμένα, αποδεικνύεται με στοιχειώδη άλγεβρα πινάκων ότι ο πίνακας Q προκύπτει ως εξής:

$$Q = \begin{bmatrix} Q_1 + Q_2 - Q_4 + Q_6 & Q_4 + Q_5 \\ Q_6 + Q_7 & Q_2 - Q_3 + Q_5 - Q_7 \end{bmatrix} \quad (7.12)$$

όπου

$$\begin{aligned} Q_1 &= (B - D)(G + H), \\ Q_2 &= (A + D)(E + H), \\ Q_3 &= (A - C)(E + F), \\ Q_4 &= (A + B)H, \\ Q_5 &= A(F - H), \\ Q_6 &= D(G - E), \\ Q_7 &= (C + D)E. \end{aligned}$$



Παρατηρήσεις

Περιγραφή

Μονοτροπία

Γρήγορη Ταξινόμηση

Πολλαπλασιασμός

- Εισαγωγή
- Παράδειγμα
- Το πρόβλημα
- Θεμελίωση
- Θεμελίωση (συνέχ.)
- Παράδειγμα (συνέχ.)
- Ο αλγόριθμος
- Ανάλυση
- Ανάλυση(συνεχ.)
- Παρατηρήσεις

Η υλοποίηση του αλγόριθμου με βάση την παραπάνω περιγραφή χρήζει προσοχής στον υπολογισμό του p_3 . Τα αθροίσματα $(x^L + x^R)$, $(y^L + y^R)$ μπορεί να μην έχουν τον ίδιο αριθμό ψηφίων. Προκειμένου να ξεπεραστεί το πρόβλημα θα πρέπει να γίνει “έυθυγράμμιση” προσθέτοντας στο άθροισμα με το μικρότερο αριθμό ψηφίων ένα “προπορευόμενο” μηδενικό (γιατί ;).



• Υπόθεση για $n=2^r$ σε N
Karachuba's multiplication technique

$$5235 = 52 \cdot 10^2 + 35$$

$$x \cdot y = 10^n (x^L y^L) + (x^R y^R) + 10^{n/2} (x^L y^R + x^R y^L)$$

$$x \cdot y = 10^n (x^L y^L) + (x^R y^R) + 10^{n/2} [(x^L + x^R)(y^L + y^R) - (x^L y^R + x^R y^L)]$$

3 πολλαπλασιάσκανται
 $\frac{n}{2}$ αριθμούς (μεταξύ)

• Δε λεγόμενει ως
πολλαπλασιάσκανται
επίσημη πολλαπλασιάσκανται
ως διδύνουνται
κανενίκα μη τέλος

• $O(n)$ ου και ΣΥΒ πρωτότιτη από Σιαχωριόδο
(μηδέτεν)

P_1, P_2, P_3 αναδρομική γήρανση $\rightarrow T\left(\frac{n}{2}\right)$

• Με ενδιαφέρει να φράξει από τα πάντα τον
πολλαπλασιαστή

Συνολικά, ο αριθμός των ΣΥΒ που εκτελεί ο αλγόριθμος ικανοποιεί την αναδρομική σχέση⁵

$$T(n) = \begin{cases} \Theta(1), & n \leq 5, \\ T\left(\frac{n}{5}\right) + T\left(\frac{7n}{10}\right) + cn, & n \geq 6. \end{cases}$$

Το δεξί μέλος της παραπάνω σχέσης για $n \geq 6$ γράφεται ως

$$T\left(\frac{n}{5}\right) + T\left(\frac{7n}{10}\right) + cn = T\left(\frac{n}{10}\right) + T\left(\frac{n}{7}\right) + cn. \quad (7.13)$$

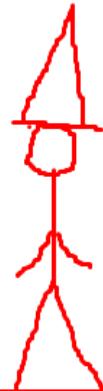
Για να υπολογίσουμε την τάξη της παράστασης αυτής με τη χρήση του Θεωρήματος 3 πρέπει να βρούμε το p για το οποίο

$$\left(\frac{2}{10}\right)^p + \left(\frac{7}{10}\right)^p = 1.$$

Δεν χρειάζεται να υπολογίσουμε το p ακριβώς: επειδή το άθροισμα των δύο λόγων είναι μικρότερο της μονάδας, έχουμε ότι $0 < p < 1$ και άρα

$$\int_1^n \frac{g(u)}{u^{p+1}} du = \int_1^n u^{-p} du = \frac{u^{1-p}}{1-p} \Big|_{u=1}^n = \frac{n^{1-p} - 1}{1-p} = \Theta(n^{1-p}).$$

Επομένως σύμφωνα με το Θεώρημα 3, η (7.13) είναι τάξης $\Theta(n^p \cdot (1 + \Theta(n^{1-p}))) = \Theta(n)$ και άρα $T(n) = O(n)$.



7.5 Εγγύτερο ζεύγος σημείων

Το πρόβλημα αυτό είναι πολύ απλό στη διατύπωση του: δεδομένου ενός συνόλου σημείων $P \subset \mathbb{R}^2$, να βρεθεί το ζεύγος των σημείων που ελαχιστοποιεί την μεταξύ τους απόσταση.

Χωρίς βλάβη της γενικότητας μπορούμε να θεωρήσουμε να θεωρήσουμε ως μέτρο την Ευκλειδεία απόσταση- δεδομένων δύο σημείων $x^1 = (x_1, y_1)$ και $x^2 = (x_2, y_2)$,

$$d(x^1, x^2) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}. \quad (7.14)$$

Έτσι το παραπάνω πρόβλημα έγκειται στην εύρεση σημείων $x^p, x^q \in P$ τέτοιων ώστε

```
47:     return δ;
48: end function
```

φράγμα, δηλαδή $T(n) = O(n \lg n)$. Παρατηρούμε ότι ο αριθμός των ΣΥΒ που εκτελούνται αρχικώς (πριν την κλήση του Αλγόριθμου 48) για την ταξινόμηση των σημείων σε πίνακες P, Q , είναι ίδιας τάξης, επομένως συνολικός αριθμός ΣΥΒ για τον υπολογισμό εγγύτερου ζεύγους σημείων είναι τάξης $O(n \lg n)$.



7.6 Ασκήσεις

1. Ένας πίνακας A περιέχει n ταξινομημένους ακεραίους (θέσεις lb, \dots, ub) διαφορετικούς μεταξύ τους. Να περιγράψετε έναν αλγόριθμο πολυπλοκότητας $\Theta(\lg n)$ που να βρίσκει αν υπάρχει στον πίνακα θέση k τέτοια ώστε $A[k] = k$.
2. Δίνεται πίνακας A ο οποίος στις θέσεις από 1 ως n περιέχει ακεραίους από το σύνολο $\{0, 1\}$. Θεωρούμε ότι ο πίνακας είναι ταξινομημένος σε αύξουσα σειρά.
 - (α') Να περιγράψετε σε ψευδοκώδικα αλγόριθμο τύπου «Διαίρει και Κυρίευε» τάξης $\Theta(\lg n)$ ο οποίος να υπολογίζει τον πλήθος των στοιχείων του πίνακα που είναι ίσα με 1.
 - (β') Να αποδείξετε την πολυπλοκότητα του αλγόριθμου.
3. Δοθέντος ενός συνόλου S n σημείων της γραμμής \mathbb{R} , θέλουμε να βρούμε το ζεύγος των σημείων $p, q \in S$ που ελαχιστοποιεί την απόσταση $|s - r|$, για $s, r \in S$. Να προτείνετε έναν αλγόριθμο για την επίλυση του προβλήματος αυτού με χρήση της τεχνικής «Διαίρει και Κυρίευε». Προσέξτε η πολυπλοκότητα του αλγόριθμου.

Άσκηση με πλειοψηφούν στοιχείο πίνακα, revisiting τον αλγόριθμο για k -ιοστο μικρότερο στοιχείο πίνακα αλλά με χρήση της τεχνικής ΔκΒ (διάμεσος των διαμέσων)

[notes\04_Aναδρούμ.pdf](#) σελ. 8 (Ο αλγόριθμος k-Element)

Αλγόριθμος 27 k -ιοστο μικρότερο στοιχείο

Απαιτείται: πίνακας A με στοιχεία στις θέσεις από lo έως hi , ακέραιος $k \in \{1, \dots, hi - lo + 1\}$.

Επιστρέφεται: k -ιοστό μικρότερο στοιχείο του πίνακα A .

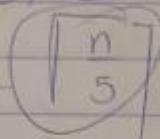
```
1: function k-Element(int A[], int lo, int hi, int k)
2:   if lo = hi then
3:     return lo
4:   end if
5:   n  $\leftarrow$  hi - lo + 1;
6:   pvt  $\leftarrow$  Rand(n) + lo;
7:   j  $\leftarrow$  Pivot_Partition(A, lo, hi, pvt);
8:   i  $\leftarrow$  j - lo + 1;
9:   if i = k then
10:    return A[j];
11:   else if i > k then
12:     return k-Element(A, lo, j - 1, k);
13:   else
14:     return k-Element(A, j + 1, hi, k - i);
15:   end if
16: end function
```

- Αλγορίθμος 27, οε αριθμητικό χρωματισμός! (recursion.pdf)
- Η επιλογή του στατιστικού δεν είναι σημαντική

DivCon.pdf

- Αν έχει συγκεκριμένο μέγεθος τιμών, μπορεί να βρει
τη στάδιοσσο σε σταδιού χρωματισμό (5)
- Αλγόριθμος: Να γίνει με 7 στάδια

$$O(5 \lg 5) = C$$



• Αλγορίθμος median of medians

$$\{8, 2, 9, 10, 12 | 3, 4, 5, 4, 15 | 3, 10, 2, 1, 7\}$$

9 4 3

8, 2	9, 3	10, 4, 15
2, 1	3	7, 10

$$\frac{n}{5} = \frac{n}{10} + \frac{2n}{10} = \frac{3n}{10}$$

$$\frac{5}{2}$$

Ηένω ηε προβλήμα
τινεκει ταυτοχρόνως

$$T(n) = \begin{cases} T\left(\frac{n}{5}\right) + T\left(\frac{2n}{10}\right) + cn, & n \geq 6 \\ O(1) & n \leq 5 \end{cases}$$

Γραφει: αριθμος σταδιων

Άσκηση με πελιοψηφούν στοιχείο

exercises|Άσκήσεις σε Διαίρει και Βασίλευε.pdf σελ. 18-19

18 / 20 | - 100% + | ☰



Άσκηση 12 Ένας πίνακας $A[1, \dots, n]$ έχει ένα πλειοψηφούν στοιχείο αν τα περισσότερα από τα μισά στοιχεία του είναι ίδια. Να εκπονηθεί αλγόριθμος πολυπλοκότητας $O(n \lg n)$ ο οποίος να προσδιορίζει αν ο πίνακας έχει πλειοψηφούν στοιχείο και αν ναι να βρίσκει το στοιχείο αυτό. Θεωρήστε ότι τα στοιχεία που βρίσκονται αποδημεύμενα στον πίνακα δεν είναι απαραίτητα αριθμοί και επομένως μπορεί να χρησιμοποιηθούν λογικές εκφράσεις τις μορφής $A[i] = A[j]$ ή $A[i] \neq A[j]$ και όχι $A[i] >= A[j]$ ή $A[i] > A[j]$.

Divide Exer vr02 (Alg. 12)

Ενδιαφέροντας για την άσκηση η θέση της πλειοψηφούν στοιχείου στον πίνακα.

Να εντοπιστεί πρώτα ο πίνακας της πλειοψηφούν στοιχείου.

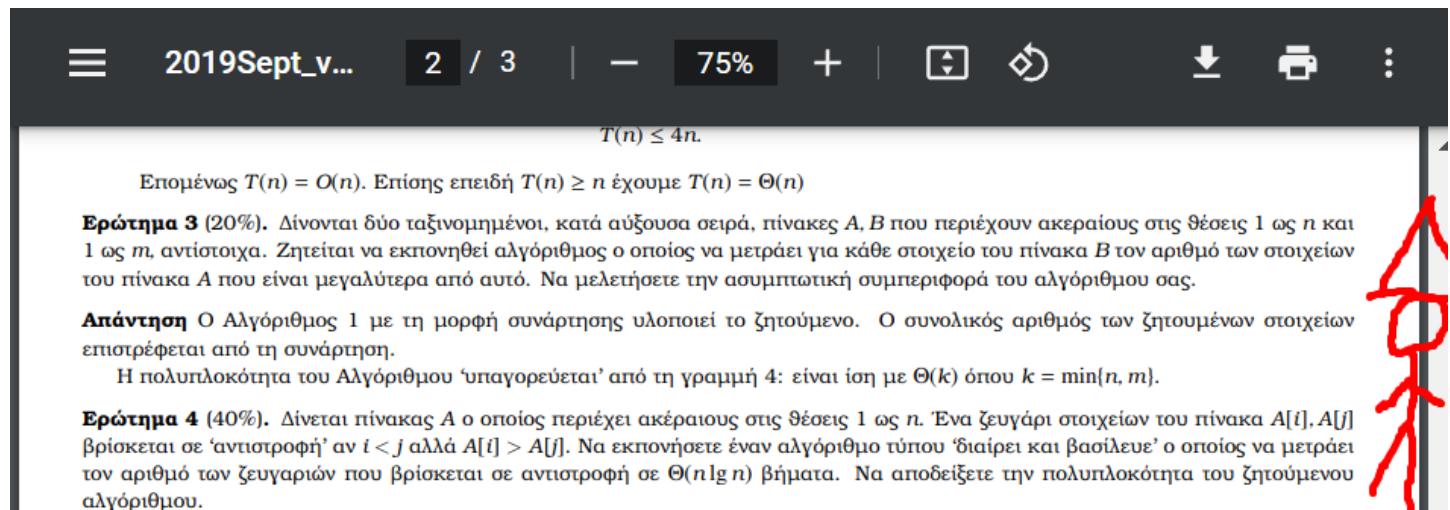
Ο πίνακας προσδιορίζεται από την πλειοψηφούν στοιχεία και από την πλειοψηφούν στοιχεία.

• Τα στοιχεία του πίνακα προσδιορίζεται από την πλειοψηφούν στοιχεία, από επιτρέποντα $A[i] = A[j]$ των πολλών σημαντικών σημείων, από επιτρέποντα $A[i] \neq A[j]$ των πολλών σημαντικών σημείων.

• Αν έχω επιλεγεί το λεπτό που θέλω να εξέρω από αυτήν την άσκηση $A[i] = A[j]$

• Αν επιλέγει η πλειοψηφούν στοιχεία της πλειοψηφούν στοιχείου του πίνακα, θα πρέπει να είναι την πλειοψηφούν στοιχείο του πίνακα σε ολόκληρη την πλειοψηφούν στοιχεία του πίνακα.

[exams\2019Sept_vr02.pdf σελ. 2-3](#)
[extraNotes\Τελευταίο Μάθημα.pdf](#)



2 / 3 | - 75% + | T(n) ≤ 4n. | Download Print | ...

Επομένως $T(n) = O(n)$. Επίσης επειδή $T(n) \geq n$ έχουμε $T(n) = \Theta(n)$

Ερώτημα 3 (20%). Δίνονται δύο ταξινομημένοι, κατά αύξουσα σειρά, πίνακες A, B που περιέχουν ακέραιους στις θέσεις 1 ως n και 1 ως m , αντίστοιχα. Ζητείται να εκπονηθεί αλγόριθμος ο οποίος να μετράει για κάθε στοιχείο του πίνακα B τον αριθμό των στοιχείων του πίνακα A που είναι μεγαλύτερα από αυτό. Να μελετήσετε την ασυμπτωτική συμπεριφορά του αλγόριθμου σας.

Απάντηση Ο Αλγόριθμος 1 με τη μορφή συνάρτησης υλοποιεί το ζητούμενο. Ο συνολικός αριθμός των ζητουμένων στοιχείων επιστρέφεται από τη συνάρτηση.

Η πολυπλοκότητα του Αλγόριθμου ‘υπαγορεύεται’ από τη γραμμή 4: είναι ίση με $\Theta(k)$ όπου $k = \min\{n, m\}$.

Ερώτημα 4 (40%). Δίνεται πίνακας A ο οποίος περιέχει ακέραιους στις θέσεις 1 ως n . Ένα ζευγάρι στοιχείων του πίνακα $A[i], A[j]$ βρίσκεται σε ‘αντιστροφή’ αν $i < j$ αλλά $A[i] > A[j]$. Να εκπονήσετε έναν αλγόριθμο τύπου ‘διαίρει και βασίλευε’ ο οποίος να μετράει τον αριθμό των ζευγαριών που βρίσκεται σε αντιστροφή σε $\Theta(n \lg n)$ βήματα. Να αποδείξετε την πολυπλοκότητα του ζητούμενου αλγόριθμου.



ΚΑΛΗ ΜΑΣ ΕΞΕΤΑΣΤΙΚΗ ΜΑΓΚΕΣ

