

ΑΛΓΟΡΙΘΜΟΙ - Εαρινό εξάμηνο

2022

Αρ. Μητρ.: 3200098

Όνομα/νικό ΑΛΒΙΩΝΑ
ΜΑΝΤΣΟ

Ομάδα Ασκίσεων # 1

Άσκ. 1 Στις δοσμένες συναρτήσεις μπορούν να γίνουν οι εξής μετασχηματισμοί:

- $n!$ $\Theta(n \log n)$ $\Theta((\log n)^{\log n})$

- $(\log n)^9$ $\Theta(2^{(\log n)^2}) = \Theta(2^{\log \log n}) = \Theta(n^{\log 9})$ $\Theta(n \log(\log n)^9) = \Theta(n \log(\log n))$

- $\sqrt[n]{n} = n^{\frac{1}{n}}$ $\Theta(n^n)$ $\Theta(n \log n)$

- $\sqrt{n}^n = (n^{\frac{1}{2}})^n = n^{\frac{n}{2}}$

- $2^{n \log n} = 2^{\log n^n} = n^n$

- $n^{\sqrt{n}} = n^{n^{\frac{1}{2}}}$

Λαμβάνοντας τα παραπάνω υπόψιν, η σειρά f_1, f_2, f_3, \dots ($f_i = O(f_{i+1})$) είναι:

$$(\log n)^9, \Theta(n^{\frac{1}{9}}), (\log n)^{\log n}, n \log(\log n)^9, n \log n, n^{\log n}, 2^{(\log n)^2}, n^{\sqrt{n}}, \sqrt{n}^n, n!,$$

$$\rightarrow 2^{n \log n}, n^n$$

Άσκ. 2

- $T(n) = T(n/2) + O(n)$: $a=1$ (>0), $b=2$ (>1), $d=1$ και $b^d = 2^1 = 2 > a$ \rightarrow Αριθ. $O(n)$
- $T(n) = T(n/2) + O(1)$: $a=1$, $b=2$, $d=0$ και $b^d = 2^0 = 1 = a$ \rightarrow Αριθ. $O(\log_2 n)$
- $T(n) = 2T(n/2) + O(1)$: $a=2$, $b=2$, $d=0$ και $b^d = 2^0 = 1 < a$ \rightarrow Αριθ. $O(n)$
- $T(n) = 4T(n/2) + O(n^2)$: $a=4$, $b=2$, $d=2$ και $b^d = 2^2 = 4 = a$ \rightarrow Αριθ. $O(n^2 \log n)$
- $T(n) = 4T(n/2) + O(n^3)$: $a=4$, $b=2$, $d=3$ και $b^d = 2^3 = 8 > a$ \rightarrow Αριθ. $O(n^3)$
- $T(n) = 4T(n/2) + O(n)$: $a=4$, $b=2$, $d=1$ και $b^d = 2 < a$ \rightarrow Αριθ. $O(n^2)$

► Αναγνωριση συνθήκης βρόχου:

Θα γίνουν $\lfloor \log_2 a \rfloor + 1 = n$ επαναλήψεις, δηλαδόταν τα ψηφία της δυαδικής αναπαράστασης του a . Γράφουμε το δυαδικό μορφή του a ως $a_n a_{n-1} \dots a_2 a_1$ (σε δυαδική μορφή)

→ Στην επανάληψη k (όπου $k \leq n$) θα λογίζει:

$$\left[\begin{array}{l} p = b_{\text{init}} (a_{k-1} \dots a_1) \\ a = a_{\text{init}} / 2^{k-1} \\ b = b_{\text{init}} \cdot 2^{k-1} \end{array} \right] \quad (\text{όπου } a_{\text{init}}, b_{\text{init}} \text{ οι είσοδοι του αλγορίθμου})$$

Απόδειξη

- Βάση: Για $k=1$ $a = a_{\text{init}} = a_{\text{init}} / 2^0 \checkmark$
 $b = b_{\text{init}} = b_{\text{init}} \cdot 2^0 \checkmark$
 $p = b_{\text{init}} \cdot (a_0 \cancel{\dots} a_1) = 0 \checkmark$

• Υπόθεση:

Έστω ότι για $k=m$ λογίζουν: $p = b_{\text{init}} \cdot (a_{m-1} \dots a_1)$
 $a = a_{\text{init}} / 2^{m-1}$
 $b = b_{\text{init}} \cdot 2^{m-1}$

• Βήμα: ($k=m+1$)

Στην πρώτη της επανάληψης $k=m$ λογίζουν εξ' υποθέσεως τα παραπάνω.

Από χρ. 3, γίνεται ο έλεγχος $a \bmod 2 = 1$ και αφού $a = a_{\text{init}} / 2^{m-1}$

ο έλεγχος αυτός συνιστάται στον $a_m = 1$ (το m -οσέρ ψηφίο της δυαδικής αναπαράστασης)

'Ετοιμοι είχουμε:

$$p = \begin{cases} b_{\text{init}} (a_{m-1} \dots a_1) + b, & \text{αν } a_m = 1 \\ b_{\text{init}} (a_{m-1} \dots a_1), & \text{αν } a_m = 0 \end{cases} \quad \begin{matrix} ① \\ ② \end{matrix}$$

όμως (1) $\Rightarrow p = b_{\text{init}} (a_{m-1} \dots a_1) + b_{\text{init}} \cdot 2^{m-1} = b_{\text{init}} (a_{m-1} \dots a_1 + 2^{m-1}) =$ *(βλ. σελ 4)
 $= b_{\text{init}} (1 a_{m-1} \dots a_1) = b_{\text{init}} (a_m a_{m-1} \dots a_1)$

(2) $\Rightarrow p = b_{\text{init}} (a_{m-1} \dots a_1) = b_{\text{init}} (0 a_{m-1} \dots a_1) = b_{\text{init}} (a_m a_{m-1} \dots a_1)$

Στις χρ. 4,5: $a = \lfloor a / 2 \rfloor = a_{\text{init}} / 2^m \checkmark$, $b = b \cdot 2 = b_{\text{init}} \cdot 2^m \checkmark$

και αυτές οι τιμές λογίζουν ουσιαστικά στην επανάληψη $k=m+1$.

- Τερματισμός: Όταν το $k = \lfloor \log_2 a \rfloor + 1 + 1$ (τελευταίος έπειγχος συνθήκης while)

τότε:

$$\rightarrow p = b_{\text{init}} \cdot (a_{\lfloor \log_2 a \rfloor + 1} \dots a_1) = b_{\text{init}} \cdot a_{\text{init}}, \quad (*)$$

που είναι και αυτό που θέλουμε.

Παρατίρηση

(*) Σε αυτήν την δύκη η δυαδική αναπαράσταση έχει γραφεί στη μορφή $a_s \dots a_1$ και όχι $a_s \dots a_0$ απότελε με αυτήν την αντιμετώπιση ισχύει η λογική στην προηγουμένων σελίδα και εδώ.

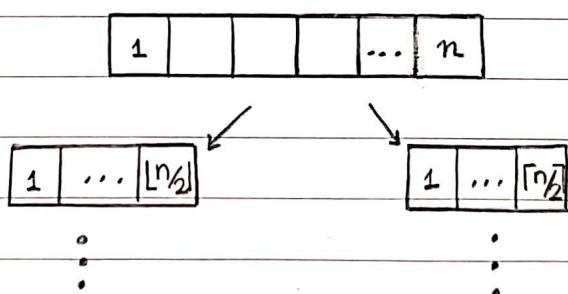
Άσκ. 4

Υπάρχουν δύο περιπτώσεις για το $a > 0$:

- 1) $a > 1$
- 2) $a \in [0, 1]$

Η περιπτώση (1) είναι τετριμένη αφού για $a > 1$ δεν θα υπάρχει κανένα στοιχείο που εμφανίζεται $a \cdot n > n$ φορές (αφού η είναι όλα τα στοιχεία)

Για την περιπτώση (2) υπορούμε να σκεφτούμε με Διαιρετική & Βασικές ως εξής:



Αν γνωρίζουμε ότι τα α-δημοφιλή στοιχεία των πινάκων του προηγουμένου επιπέδου είναι τα συνολα

$A_{\text{left}} = \{l_1, l_2, \dots\}$, $A_{\text{right}} = \{r_1, r_2, \dots\}$
τότε τα α-δημοφιλή στοιχεία του ψηλότερου επιπέδου θα είναι

$$A_{\text{total}} \subseteq A_{\text{left}} \cup A_{\text{right}},$$

δηλαδή θα βρίσκονται σίγουρα μεταξύ των α-δημοφιλών του προηγουμένου επιπέδου.

Μόλιστα επειδή αν σε ένα επιπέδο έχουμε k στοιχεία, στο προηγουμένο θα έχουμε $\sim \frac{k}{2}$, θα λογίζει ότι $\text{item} \in A_{\text{left}} \cup A_{\text{right}} \Rightarrow P(\text{item}) \geq a \cdot \frac{k}{2}$ και



$\text{item} \in A_{\text{left}} \cap A_{\text{right}} \Rightarrow f(\text{item}) \geq a \left(\frac{k}{2} + \frac{k}{2} \right) = a \cdot k$ (όπου $f(\text{item})$ οι φορές εμφάνισης του item στο άνω επίπεδο)

Αυτό σημαίνει ότι:

Για τα στοιχεία που εμφανίζονται και στους 2 υπονύματες δεν χρειάζεται να ελέγξουμε το πλήθος εμφάνισής τους στον από-πάνω πίνακα: είναι εγγυημένα α-δημοφιλή. Για τα υπόλοιπα στοιχεία (που εμφανίζονται μόνο στον έναν ή των δύο υπονύματων) πρέπει να αναζητήσουμε το πλήθος εμφάνισής τους στον από-πάνω πίνακα: Θα είναι α-δημοφιλή μόνον αν εμφανίζονται $>$ αν φορές.

Πότε σταματούν οι αναδρομικές κλήσεις? (θεωρητής/τετριμμένη περιπτώση)

Όταν μπορούμε να αποφανθούμε ότι τα α-δημοφιλή στοιχεία άμεσα.

Ουσιαστικά όταν οι υπονύματες έχουν 1 στοιχείο, τότε στους υπονύματες αυτούς το στοιχείο είναι α-δημοφιλής $\forall e[0,1]$ αφού:

$a \cdot 1 \leq 1 \rightsquigarrow$ αρκεί τα στοιχεία να εμφανίζονται τουλάχιστον 1 φορά

DRIVER(A, a)

1. if $a > 1$ then return \emptyset
2. else $n \leftarrow A.\text{length}$
3. return DIV-N-CONQ(A, 1, n, a)

DIV-N-CONQ(A, l, r, a)

1. $n \leftarrow r - l + 1$
2. $\text{total} \leftarrow \emptyset$
3. if $n \leq 1$ // base
 4. $\text{total} \leftarrow \text{total} \cup \{A[l]\}$
 5. return total
6. $m = \text{floor}((p+r)/2)$
7. $\text{left} = \text{DIV-N-CONQ}(A, l, m, a)$
8. $\text{right} = \text{DIV-N-CONQ}(A, m+1, r, a)$



```

9. candidates ← left ∪ right (2)
10. for each item in candidates (1)
11.   count ← 0
12.   for i ← l to r
13.     if A[i] = item
14.       count ← count + 1
15.   if count ≥ a * n then
16.     total ← total ∪ {item}
return total
END
  
```

Παρατήρηση !

Τα συνολα εδω χρησιμοποιούνται με τη μαθηματική τους έννοια και όχι σόντας συνήθως υλοποιούνται στη δλ. προγραμμ. (hashing)

Δρα αρκεί να πάρει/έλεγχος της λογικής

Πρέπει να χίνουν οι εξής παρατηρήσεις

(1) Τα υποψήφια (candidates) α-δημοφιλή στοιχεία των υπονιγάκων ^{δυο} θα είναι ≤ min {n, $\lfloor \frac{n}{a} \rfloor \}$ το πανθός καθίς.

↳ Για τον κάθε υπονιγάκα (του πίνακα μεχεθούς k) :

Μέγεθος: $\frac{k}{2}$ Φορές εμφάνισης: $a \cdot \frac{k}{2}$, ίφατο πολὺ $\frac{1}{2}$ στοιχεία θα είναι $\frac{a \cdot k}{2}$ α-δημοφιλή.

↳ Έχουμε 2 υπονιγάκες δρα το πολὺ $2 \cdot \frac{1}{a} = \frac{2}{a}$ στοιχεία υποψήφια

(2) Στην ανάλυση του συλλογικού παρατηρούμε ότι τα στοιχεία που ανήκουν στην τομή των α-δημοφιλών των 2 υπονιγάκων δεν χρειάζεται να έλεγχούν. Σε ψευδοκώδικα αυτήν παρατήρηση δίνεται:

```

total ← left ∩ right
union ← left ∪ right
candidates ← union \ total
  
```

... και κάθε μια ανό αυτές

τις πράξεις γίνεται σε $\leq (\min \{n, \lfloor \frac{1}{a} \rfloor \})^2$

9. [candidates ← left ∪ right]

εναντί

7

... γίνεται σε $\leq (\min \{n, \lfloor \frac{1}{a} \rfloor \})^2$
αλλά μικρότερα συντελεστής

Kai στις 2 περιπτώσεις, n for tns xp. 10 εκτελείται

$\leq \min \{n, \lfloor \frac{2}{a} \rfloor \}$ φορές (worst case: left ∩ right = $\emptyset \Rightarrow$ candidates = union)



Με βάση τα παραπάνω έχουμε:

Πολυπλοκότητα DIV-N-CONQ:

$$T(N) = 2T(N/2) + O(N^2) + O(N^2) \Rightarrow$$

2 αναδρομικές

υποκλήσεις (left, right)

μεγέθους $N/2$ έκαστη

worst case

δια πράξης

μεταξύ συνόλων

(όταν $\min\{n, \lfloor \frac{N}{2} \rfloor\} = n$)

worst case

στα το double for loop
(χρονιμές $10+12$)

(όταν $\min\{n, \lfloor \frac{N}{2} \rfloor\} = n$)

$$\Rightarrow T(N) = 2T(N/2) + O(N^2)$$

$$a=2$$

$$b=2$$

$$b^d = 4 > a$$

$$d=a$$

Master Theorem $\longrightarrow O(N^2)$

$$d=2$$



Aσκ. 5

Ο αλγόριθμος θα παίρνει ως εισοδό τους όλην αριθμούς $(x_1, x_2, \dots, x_{2n})$ και τα ίεύχη που θα παράγει θα έχουν ως εξής:

{ο μικρότερος μετανάστερο}, {ο 2^{ος} μικρότερος μετανάστερο}, ...
{ο k-οςτός μικρότερος μετανάστερο}...
...

Για να διευκολυνθεί αυτή η επιλογή και να εφαρμοστεί αλγορίθμικά, η εισοδος πρέπει να ταξινομηθεί ώστε οι αριθμοί να είναι διατεταγμένοι. Ειστούν έχουμε μια γένια μετάθεση $\sigma_{2n} > \sigma_{2n-1} > \dots > \sigma_2 > \sigma_1$ που έχει προκύψει από την αρχική εισοδο. Τα ίεύχη που θα παράγει ο αλγόριθμος μπορούν να κωδικοποιηθούν τώρα ως εξής:

$\{\sigma_{2n}, \sigma_1\}$, $\{\sigma_{2n-1}, \sigma_2\}$, ..., $\{\sigma_{2n-k+1}, \sigma_k\}$, ..., $\{\sigma_{n/2+1}, \sigma_{n/2}\}$

Μπορούμε να δούμε αυτήν την ιδέα με τον ακόλουθο τρόπο: Με σύχοο να ελαχιστοποιήσουμε το μέγιστο μερικό (ανά ίεύχο) αθροισμα, διατρέχοντας τους αριθμούς από τον μεγαλύτερο τοποθετούμε τον εκάστοτε αριθμό με τον πρώτο διαθέσιμο μικρότερο αριθμό. Κάνουμε δηλαδή μια επιλογή που την τρέχουσα στιγμή φαίνεται η βέλτιστη, επιζηγούμε ότι θα είναι η καθολικά βέλτιστη. Πρόκειται λοιπόν για έναν δηλητικό αλγόριθμο.

Ισχυρίζομαστε ότι αυτός ο αλγόριθμος θα οδηγήσει στο min max.

Απόδειξη

Έστω πως ο greedy δεν είναι ο βέλτιστος. Θα υπάρχει λοιπόν κάποια άλλη βέλτιστη διαμέτρηση σε ίεύχη. Έστω επιστρέψτε ο μικρότερος δείκτης για τον οποίο έχουμε $\text{Ιεύχος}_r < \text{Ιεύχος}_{r_{opt}}$. Δηλαδή έχουμε:

Greedy:	$\{\sigma_{2n}, \sigma_1\}$,	$\{\sigma_{2n-1}, \sigma_2\}$...	$\{\sigma_{2n-r+1}, \sigma_r\}$...	
Opt:	$\{\sigma_{2n}, \sigma_1\}$,	$\{\sigma_{2n-1}, \sigma_2\}$...	$\{\sigma_{2n-r+1}, \sigma_i\}$...	$\{\sigma_r, \sigma_j\}$...
	Ιεύχος_1	Ιεύχος_2 ...	Ιεύχος_r ...	

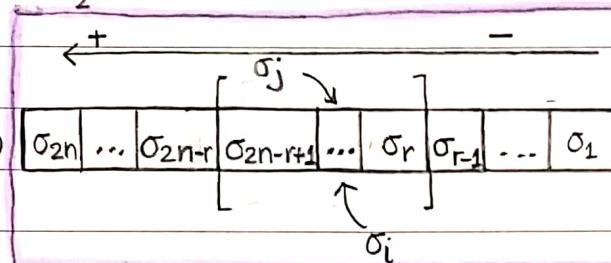
- Εξ' υποθέσεως λοχύει:

$$\max_{\text{opt}} < \max_{\text{Greedy}}$$

- Επιπλέον λοχύει $\sigma_{j+1} \leq \max_{\text{opt}}$ $\forall i \in [1, \frac{n}{2}]$ και συνεπώς:

$$(\sigma_{2n-r+1} + \sigma_i) \leq \max_{\text{opt}} \quad (1)$$

και $(\sigma_r + \sigma_j) \leq \max_{\text{opt}} \quad (2)$



- Έχουμε τις εξής παρατηρήσεις:

$\sigma_r < \sigma_i$ ⁽³⁾ αναγκαστικά γιατί 1) $\sigma_i \neq \sigma_r$ και 2) $\forall i \in [1, r-1]$ σ_i έχει ήδη επιλεχεί
και $\sigma_j < \sigma_{2n-r+1}$ ⁽⁴⁾ αναγκαστικά (παρομοίως)

- Εστιν λοιπόν πως κάνουμε την ανταλλαγή $\{\sigma_{2n-r+1}, \sigma_r\}$, $\{\sigma_i, \sigma_j\}$

Ανό τις παραπάνω σχέσεις λαμβάνουμε:
(3) $\cancel{\sigma_r + \sigma_{2n-r+1}} \leq \sigma_i + \sigma_{2n-r+1} \stackrel{(1)}{\leq} \max_{\text{opt}}$

(4) $\cancel{\sigma_i + \sigma_j} \leq \sigma_{2n-r+1} + \sigma_i \stackrel{(1)}{\leq} \max_{\text{opt}}$

↪ Αυτή λοιπόν η ανταλλαγή:

- αν το \max_{opt} ήταν το $(\sigma_{2n-r+1} + \sigma_i)$ πετυχαίνει μικρότερο \max_{opt} → ΑΤΟΠΟ
- αν το \max_{opt} δρισκόταν αλλού, τότε είναι και αυτή βέλτιστη

Αν συνεχίσουμε αυτές τις ανταλλαγές: Greedy \equiv Opt \wedge Opt δεν είναι optimal (άντονο!)

Άρα ο Greedy αναδριθμός είναι και ο βέλτιστος.

Αλγόριθμος
(σε ψευδοκώδικα) →



GREEDY($x[1, 2, \dots, 2n]$)

$\sigma[1, 2, \dots, 2n] \leftarrow \text{sort_descending_order}(x[1, 2, \dots, 2n])$ //e.g. mergesort

$\max \leftarrow \infty$

for $i \leftarrow 1$ to $2n/2$

 write($\sigma[i], \sigma[2n-i+1]$)

$\text{sum} \leftarrow \sigma[i] + \sigma[2n-i+1]$

 if $\text{sum} > \max$

$\max \leftarrow \text{sum}$

 end-if

end-for

return \max //should be the minimum possible

(Πολυπλοκότητα: $O(N \log N) \rightsquigarrow \text{sorting}$)