Αλγόριθμοι & Πολυπλοκότητα

1η Σειρά Γραπτών Ασκήσεων

7ο Εξάμηνο - Φθινόπωρο 2020

Δημήτρης Δήμος [031 17 165]

dimitris.dimos647@gmail.com

Άσκηση 1: Ασυμπτωτικός Συμβολισμός, Αναδρομικές Σχέσεις

(α) $a(n) = (\sqrt{n})^{\log_2\log_2(n!)}$ $\quad b(n) = 2^{(\log_2\log_2 n)^4}$ $\quad \gamma(n) = \dfrac{\log(n!)}{(\log\log n)^5}$ $\quad \delta(n) = n\cdot 2^{2^{2^{100}}}$

$\varepsilon(n) = n\sum_{k=0}^{n}\binom{n}{k}$ $\quad \zeta(n) = \log\left(\binom{2n}{n}\right)$ $\quad \eta(n) = \sum_{k=1}^{n} k2^{k}$ $\quad \vartheta(n) = \sum_{k=1}^{n} k2^{-k}$

$$\xrightarrow{\text{αύξουσα}}$$

Η ζητούμενη διάταξη είναι: $\vartheta(n), b(n), \delta(n), \zeta(n), \gamma(n), a(n), \varepsilon(n), \eta(n)$

με $\delta(n), \zeta(n) = \Theta(n)$ και $\varepsilon(n), \eta(n) = \Theta(n\cdot 2^n)$.

- $\vartheta(n) = \sum_{k=1}^{n} k2^{-k} = \dfrac{2^{n+1}-n-2}{2^n} \xrightarrow{n\to\infty} 2 = O(1)$

Φανερά, λοιπόν: $\vartheta(n) = O(b(n))$

- Είναι $\delta(n) = n\cdot 2^{2^{2^{100}}} = c\cdot n = c\cdot 2^{\log_2 n}$ και $b(n) = 2^{(\log_2\log_2 n)^4}$ $\Longrightarrow b(n) = O(\delta(n))$

Απόδειξη: $\lim_{n\to+\infty} \dfrac{\delta(n)}{b(n)} = \lim_{n\to+\infty}\left[c\cdot 2^{\log_2 n-(\log_2\log_2 n)^4}\right] = c\cdot\lim_{n\to+\infty} 2^{\log_2 n\left(1-\frac{(\log_2\log_2 n)^4}{\log_2 n}\right)}$

όπου $\lim_{n\to+\infty}\dfrac{(\log_2\log_2 n)^4}{\log_2 n} = \lim_{n\to+\infty}\dfrac{4(\log_2\log_2 n)^3\cdot\frac{1}{\log_2 n}\cdot\frac{1}{n\cdot\log_{10}(2)}}{\frac{1}{n\cdot\log_{10}(2)}}$

$= \lim_{n\to+\infty}\dfrac{4(\log_2\log_2 n)^3}{\log_2 n} = \dots = \lim_{n\to+\infty}\left[(c')\cdot\dfrac{\log_2\log_2 n}{\log_2 n}\right]$

$= \lim_{n\to+\infty}(c')\cdot\dfrac{1/\log_2 n\cdot 1/n}{1/n} = 0$

Άρα $\lim_{n\to+\infty}\dfrac{\delta(n)}{b(n)} = c\cdot 2^{\infty\cdot(1-0)} = \infty \Rightarrow b(n) = O(\delta(n))$

• Γνωρίζουμε ότι: $\log(n!) = \Theta(n\log n)$, άρα εξ' ορισμού $\exists\, c_1, c_2, n_0 > 0$:

$$\forall n \geq n_0, \quad c_1 \cdot n\log n \leq \log(n!) \leq c_2 \cdot n\log n$$

$$\Rightarrow c_1 \cdot \frac{n\log n}{(\log\log n)^5 \cdot n} \leq \frac{\log(n!)}{(\log\log n)^5 \cdot n} \leq c_2 \cdot \frac{n\log n}{(\log\log n)^5 \cdot n}$$

$$\Rightarrow c_1 \cdot \underbrace{\frac{\log n}{(\log\log n)^5}}_{\xrightarrow{n\to+\infty}\ \infty} \leq \frac{\log(n!)}{(\log\log n)^5 \cdot n} \leq c_2 \cdot \underbrace{\frac{\log n}{(\log\log n)^5}}_{\xrightarrow{n\to+\infty}\ \infty}$$

[βάσει της αμέσως προηγούμενης απόδειξης]

Άρα $\lim\limits_{n\to+\infty} \frac{\log(n!)}{(\log\log n)^5 \cdot n} = \lim\limits_{n\to+\infty} \gamma(n)/\delta(n) = \infty \Longrightarrow \delta(n) = O(\gamma(n))$

• Γνωρίζουμε ότι ασυμπτωτικά: $\binom{2n}{n} \xrightarrow{n\to+\infty} \frac{4^n}{\sqrt{n\pi}}$

[απόδειξη στη σελίδα proofwiki.org/wiki/Approximation_to_2n_Choose_n]

Άρα: $\log\left(\binom{2n}{n}\right) \xrightarrow{\text{ασυμπτωτικά}} n\log 4 - \frac{1}{2}\cdot\log(\pi\cdot n) = \Theta(n)$

Επομένως, $\delta(n) = \Theta(n)$ και $J(n) = \Theta(n)$

• Η $\alpha(n)$ είναι εκθετική συνάρτηση, ενώ η $\gamma(n)$ είναι ασυμπτωτικά "καλύτερη" από την $\log(n!) = \Theta(n\log n)$. Εμφανώς, λοιπόν: $\gamma(n) = O(\alpha(n))$

• $\varepsilon(n) = n \underbrace{\sum_{k=0}^{n}\binom{n}{k}}_{2^n} = n \cdot 2^n$ ②

$\alpha(n) = 2^{\frac{1}{2}\cdot\log_2(n)\cdot\log_2\log_2(n!)}$

$\Longrightarrow \alpha(n) = O(\varepsilon(n))$

• $\eta(n) = \sum_{k=1}^{n} k \cdot 2^k \overset{③}{=} (n-1) \cdot 2^{n+1} + 2 = 2 + 2 \cdot n \cdot 2^n - 2 \cdot 2^n = \Theta(n \cdot 2^n)$

Επομένως, $\varepsilon(n) = \Theta(n2^n)$ και $\eta(n) = \Theta(n2^n)$

(β)

1. $T(n) = 6 \cdot T(n/3) + n^2 \log n$, $T(1) = \Theta(1)$

- $n^{\log_b a} = n^{\log_3 6} \approx n^{1,63}$

- $f(n) = n^2 \log n \geq n^2 \Rightarrow f(n) = \Omega(n^{1,63+0,37})$

  ασυμπτωτικά

Συνθήκη ομαλότητας: $a f(n/b) < f(n) \Leftrightarrow 6 \cdot f(n/3) < f(n)$

$\Leftrightarrow 6 \cdot (n/3)^2 \cdot \log(n/3) < n^2 \log n$

$\Leftrightarrow \frac{2}{3} n^2 (\log n - \log 3) < n^2 \log n$

$\Leftrightarrow 2/3 \cdot n^2 \log n - (\text{κάτι θετικό}) < n^2 \log n$, που εμφανώς ισχύει.

Άρα, από Master Theorem: $T(n) = \Theta(f(n)) = \Theta(n^2 \log n)$

2. $T(n) = 9 \cdot T(n/3) + n^2 \log n$, $T(1) = \Theta(1)$

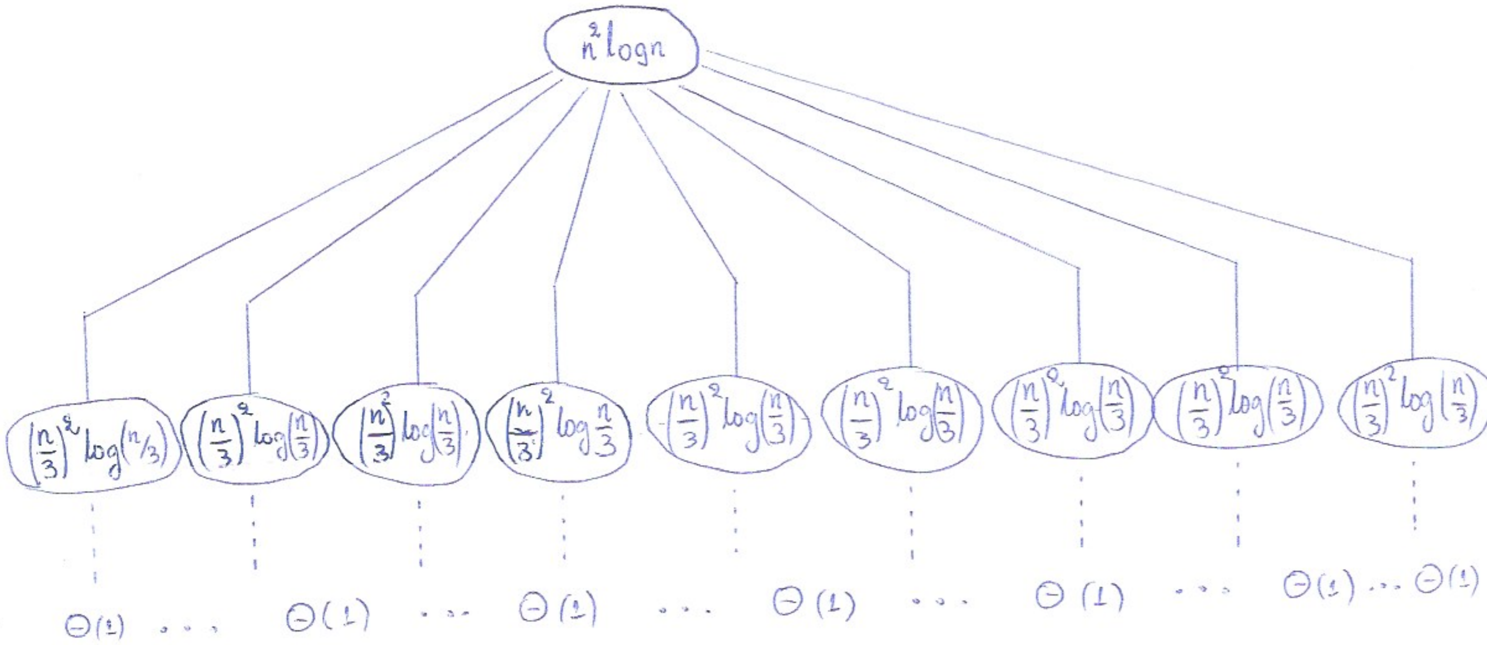
Προσπαθούμε να εφαρμόσουμε το Master Theorem, αλλά:

- $n^{\log_3 9} = n^2$
- $f(n) = n^2 \log n$

$\Rightarrow$ Η f(n) είναι μεγαλύτερη από την ποσότητα n^2, αλλά όχι πολυωνυμικά. Άρα, θα βρούμε άλλο τρόπο.

Κατασκευάζουμε δένδρο αναδρομής: (επόμενη σελίδα)

Θα κάνουμε υπόθεση του Θ(), όπως θα φανεί παρακάτω, και θα το απο-δείξουμε, στη συνέχεια.



Για την υπόθεση, δεχόμαστε την εξής ανακρίβεια:

- το n είναι δύναμη του 3, ώστε κάθε υποπρόβλημα να έχει ακέραια είσοδο

Έτσι:

- ύψος δέντρου $= \log_3 n$
- στο i-οστό επίπεδο έχουμε 9^i κόμβους $= 3^{2i}$
- κάθε κόμβος του i-οστού επιπέδου κοστίζει: $\frac{n^2}{3^{2i}}\log\left(\frac{n}{3^i}\right)$

Συνολικό κόστος: $\overbrace{\left(\log_3 n+1\right)}^{\text{επίπεδα}}\cdot 3^{2i}\cdot\frac{n^2}{3^{2i}}\cdot\log\left(\frac{n}{3^i}\right)$

$$= n^2\cdot\left(\log n - i\cdot\log 3\right)\left(\log_3 n+1\right)$$

$$= \left(n^2\log n - (i\cdot\log 3)\cdot n^2\right)\left(\log_3 n+1\right)$$

$$= n^2\log n\cdot\log_3 n + n^2\log n - (i\log 3)\cdot n^2\log_3 n - (i\log 3)\,n^2$$

Βάσει των παραπάνω, υποθέτουμε ότι $T(n)=\Theta(n^2\log^2 n)$.

Θα αποδείξουμε την υπόθεση με επαγωγή:

- Βάση: $T(3) = 9 \cdot T(1) + n^2 \log n \Big|_{n=3} = 9 \cdot d + 9 \cdot \log 3$

$$\Longrightarrow T(3) = 9\,(d + \log 3), \quad d > 0 \quad (\text{αφού } T(1) = \Theta(1))$$

Θέλουμε να $\exists\, c_1, c_2$ (έστω θετικές): $\forall\, n \geq 3$

$$c_1 \cdot n^2 \log^2 n \leq T(n) \leq c_2 \cdot n^2 \log^2 n \quad ①, \text{ οπότε:}$$

$$c_1 \cdot (3)^2 \cdot \log^2 3 \leq T(3) \leq c_2 \cdot 3^2 \cdot \log^2 3 \iff$$

$$c_1 \cdot 9 \cdot \log^2 3 \leq 9\,(d + \log 3) \leq c_2 \cdot 9 \cdot \log^2 3$$

$$\Longrightarrow \begin{cases} c_1 \leq \dfrac{d + \log 3}{\log^2 3} & \text{και} \\[2ex] c_2 \geq \dfrac{d + \log 3}{\log^2 3} \end{cases}$$

- Βήμα: έστω ότι ισχύει η ① $\forall\, k < n$, Ν.δ.ο. ισχύει και για n.

$$T(k) = \Theta(n^2 \log^2 n) \overset{k = n/3}{\Longrightarrow} T(n/3) = \Theta\left(\tfrac{n^2}{9} \log^2(n/3)\right) \quad [c_1, c_2 \text{ ίδιες με την ①}]$$

Οπότε: $T(n) = 9 \cdot T(n/3) + n^2 \log n$

$$\geq 9 \cdot c_1 \cdot \left(\frac{n}{3}\right)^2 \cdot \log^2\left(\frac{n}{3}\right) + n^2 \log n$$

$$= c_1 \cdot n^2 \cdot (\log n - \log 3)^2 + n^2 \log n$$

$$= c_1 \cdot n^2 (\log^2 n - 2 \log 3 \log n + \log^2 3) + n^2 \log n$$

$$= c_1 \cdot n^2 \log^2 n + n^2 \underbrace{\left(\log n - 2c_1 \log 3 \log n + c_1 \log^2 3\right)}_{A}, \text{ οπότε αν } A \geq 0 \Longrightarrow$$

$$T(n) \geq c_1 \cdot n^2 \log^2 n.$$

$A \geq 0 \Longrightarrow \log n \left(1 - 2c_1 \log 3 \right) + c_1 \cdot \log^2 3 \geq 0 \Longrightarrow$

$$\log n \cdot \frac{1 - 2c_1 \cdot \log 3}{c_1 \cdot \log^2 3} \geq -1 \Longrightarrow$$

$$\log n \cdot \frac{2c_1 \cdot \log 3 - 1}{c_1 \cdot \log^2 3} \leq 1,$$

που για να ισχύει $\forall n \geq 3$ (όταν και η $\log n$ είναι θετική και γν. αύξουσα) πρέπει:

$$\frac{2c_1 \cdot \log 3 - 1}{c_1 \cdot \log^2 3} \leq 0 \Longrightarrow c_1 \leq \frac{1}{2\log 3}$$

Εντελώς ανάλογα για το άνω όριο, έχουμε $c_2 \geq \frac{1}{2 \cdot \log 3}$.

Τελικά: $T(n) = \Theta(n^2 \log^2 n)$, $\quad 0 < c_1 \leq \min\left\{ \frac{1}{2\log 3}, \frac{d + \log 3}{\log^2 3} \right\}$

και $\quad c_2 \geq \max\left\{ \frac{1}{2\log 3}, \frac{d + \log 3}{\log^2 3} \right\}$

3. $T(n) = 11 \cdot T(n/3) + n^2 \log n$, $T(1) = \Theta(1)$

- $n^{\log_3 11} \approx n^{2,183}$

- $f(n) = n^2 \log n \underset{\substack{\uparrow \\ \text{ασυμπτωτικά}}}{\leq} n^{2,183} \Longrightarrow f(n) = O\left(n^{2,183}\right)$

Από Master Theorem: $T(n) = \Theta\left(n^{\log_3 11}\right)$

4. $T(n) = T(n/4) + T(n/2) + n$

$= T(\gamma_1 \cdot n) + T(\gamma_2 \cdot n) + \Theta(n)$, $T(1) = \Theta(1)$

Είναι $\gamma_1 + \gamma_2 = 1 - 1/4$, άρα από Master Theorem $T(n) = \Theta(n)$.

5. $T(n) = 2T(n/4) + T(n/2) + n$

Παρατηρούμε ότι στο δέντρο αναδρομής, μέχρι το $\log_4 n$ επίπεδο, η είσοδος n διατηρείται στα παιδιά του κάθε επιπέδου μοιρασμένη, αφού τα παιδιά κάθε γονέα έχουν αθροιστικά τόσα n όσα ο γονέας μόνος του.
Άρα σε κάθε επίπεδο μέχρι το $\log_4 n$, εκτελούνται τουλάχιστον $c \cdot n$ λειτουργίες. Άρα, $\forall n$ χρεωνόμαστε $c \cdot n \cdot (\log_4 n + 1)$ τουλάχιστον

$\Rightarrow T(n) = \Omega(n \log n)$

Ακόμη, εμφανώς, οι λειτουργίες που εκτελούνται στα χαμηλότερα επίπεδα του $\log_4 n$ θα είναι σε κάθε επίπεδο λιγότερες από $c \cdot n$, αφού κάποια παιδιά δεν γεννούν άλλα παιδιά.
Άρα οι συνολικές λειτουργίες είναι αυστηρά λιγότερες από $c \cdot n \cdot (\log_2 n + 1)$ (αφού $\log_2 n$ το ύψος του δέντρου) $\Rightarrow T(n) = O(n \log n)$

Άρα, $T(n) = \Theta(n \log n)$.

6. $T(n) = T(n^{2/3}) + \Theta(\log n)$, $T(1) = \Theta(1)$

Για να γίνει το n (περίπου έστω) 1 θα γίνουν m αναδρομές:

$$n^{(2/3)^m} = 2^{2/3} \approx 1{,}59 \Longrightarrow n = 2^{2/3 \cdot (2/3)^{-m}} \Longrightarrow$$

$$\Longrightarrow \log_2 n = \left(\frac{2}{3}\right)^{1-m} \Longrightarrow \log_2 n = \left(\frac{3}{2}\right)^{m-1}$$

$$\Longrightarrow \log_{3/2} \log_2 n = m-1 \Longrightarrow m = 1 + \log_{3/2} \log_2 n$$

Άρα κόστος $= \sum_{i=0}^{1+\log_{3/2}\log_2 n} c_i \cdot \log n^{(2/3)^i}$, όπου οι c_i αυστηρά φραγμένες.

Άρα τάξη κόστους $= \log n \cdot \sum_{i=0}^{1+\log_{3/2}\log_2 n} (2/3)^i$

$$= \log n \cdot \frac{1 - (2/3)^{1+\log_{3/2}\log_2 n}}{1 - 2/3}$$

$$= \log n \cdot 3 \cdot \left[1 - 2/3 \cdot (\log_2 n)^{-1}\right]$$

$$= 3 \cdot \log n - 2 = \Theta(\log n)$$

7. $T(n) = T(n/3) + \sqrt{n}$, $T(1) = \Theta(1)$

- $n^{\log_3 1} = 1$

- $f(n) = \sqrt{n}$ και $a f(n/b) < f(n) \Leftrightarrow f(n/3) < f(n)$

$$\Leftrightarrow \frac{\sqrt{n}}{\sqrt{3}} < \sqrt{n}$$, που ισχύει!

Από Master Theorem, αφού $f(n) = \Omega\left(n^{\log_3 1 + 1/2}\right) \Rightarrow T(n) = \Theta(\sqrt{n})$

Άσκηση 2: Προθεματική Ταξινόμηση

(α) Αλγόριθμος: 1) Βρες το μέγιστο στοιχείο από το $A[1]$ έως το $A[n]$

2) Το στοιχείο-max βρέθηκε στη θέση i

Αν $i=n \longrightarrow$ προχώρα στο βήμα 3

αλλιώς: • προθ/κή περ/φή για $k=i$

• " " " $k=n$

3) Αν $n-1 \geq 1 \longrightarrow$ επανάλαβε τον αλγόριθμο για τις $n-1$ πρώτες θέσεις, αλλιώς τέλος.

Έτσι, ο πίνακας ταξινομείται από το τέλος προς την αρχή με το πολύ 2 περιστροφές $\forall$ στοιχείο.

- Πίνακας με $n>3$ στοιχεία:
  - ⇝ έστω ότι έχουμε ταξινομήσει τα $n-3$ μεγαλύτερα στοιχεία και τα μη ταξινομημένα βρίσκονται στις 3 αριστερότερες θέσεις, κόστος το πολύ $2\cdot(n-3)=2n-6$
  - ⇝ αποδεικνύεται τετριμμένα πως τα 3 αριστερότερα στοιχεία ταξινομούνται με το πολύ 3 περ/φές.
  - ⇝ Max κόστος συνολικά: $2n-6+3=2n-3$
- Πίνακας με $n=3$ στοιχεία: (τετριμμένα) $\overset{max}{\text{κόστος}} = 3 = 2\cdot3-3 \; (=2n-3)$
- " " $n=2$ " : (") $\overset{max}{\text{κόστος}} = 1 \; (=2n-3)$

(β) Διατηρώντας τον αλγόριθμο του (α) ερωτήματος, με τη διαφοροποίηση ότι κάθε φορά αναζητείται ο αριθμός με τη μεγαλύτερη απόλυτη τιμή, ο χρόνος παραμένει $2n-3$, αλλά η ταξινόμηση θα γίνει βάσει απόλυτων τιμών και ο τελικός πίνακας ενδέχεται να έχει και αρνητικά πρόσημα.

Προσθέτουμε, λοιπόν, το εξής βήμα:

Όταν είναι η σειρά του στοιχείου i να μπει στη σωστή θέση και κάνουμε περιστροφή ώστε να βρεθεί πρώτα στη θέση 1, αν έχει θετικό πρόσημο εκεί, κάνουμε μια ακόμα περιστροφή με $k=1$, για να αλλάξει πρόσημο και να μπει στη σωστή θέση με "+".

Έστω ότι για πίνακα n στοιχείων, βάσει των παραπάνω, έχουμε κόστος $3(n-3)$ για την ταξινόμηση των $n-3$ μεγαλύτερων στοιχείων. Μένουν τα 3 αριστερότερα στοιχεία.

Η πιο δυσμενής περίπτωση είναι αυτή που απαιτεί κάθε ένα εκ των 3 τελευταίων αριθμών να απαιτεί διόρθωση προσήμου μετά την περιστροφή για να βρεθούν στη θέση 1 (με εξαίρεση τον αριθμό 1 που θα "γλιτώσει" την 1η περιστροφή). Παράδειγμα με τις παραπάνω ιδιότητες:

$$\left.\begin{array}{ccc} -2 & -3 & 1 \\ 3 & 2 & 1 \\ -3 & 2 & 1 \\ -1 & -2 & 3 \\ 2 & 1 & 3 \\ -2 & 1 & 3 \\ -1 & 2 & 3 \\ 1 & 2 & 3 \end{array}\right\} 7 \text{ περιστροφές}$$

Ακόμη, επειδή νομοτελειακά το μέγιστο στοιχείο θα ταξινομηθεί με 2 ακριβώς περιστροφές, δεν θα χρειαστεί αλλαγή προσήμου.

Σύνολο κόστους: $-1+3(n-3)+7 = 3n-2$ περιστροφές το πολύ γ) Αρχικά, αποδεικνύουμε την παρακάτω χρήσιμη θέση:

Θέση: Αν $\forall i \in \{1, \dots, n-1\}$, που στον εκάστοτε πίνακα At εμφανίζεται ως $+i$ ή $-i$, ο αμέσως μεγαλύτερός του κατά απόλυτη τιμή, που στον εκάστοτε πίνακα At εμφανίζεται ως $+(i+1)$ ή $-(i+1)$, βρίσκεται δεξιότερα τότε ο At είναι ταξινομημένος κατά αύξουσα απόλυτη τιμή.

Απόδειξη: Έστω ότι $\forall i \in \{1, \dots, n-1\}$, που στον At είναι $+i$ ή $-i$, ο αριθμός με απόλυτη τιμή $|i+1|$ βρίσκεται δεξιότερα και ο At ΔΕΝ είναι ταξινομημένος κατά απόλυτη τιμή (αύξουσα).

Τότε, υπάρχει τουλάχιστον ένα ζεύγος αριθμών που δεν είναι ταξινομημένοι κατά αύξουσα απόλυτη τιμή.

Αν, δηλαδή, "βάλουμε" $|\cdot|$ σε όλα τα στοιχεία του At, αυτός θα έχει μορφή $At = [\dots, i+c, \dots, i, \dots]$, $c > 0$, i και $i+c$ το ζεύγος που προαναφέραμε.

Από υπόθεση, ο $\pm(i+1)$ είναι δεξιότερα από τον $\pm(i)$,

ο $\pm(i+2)$ " " " " $\pm(i+1)$,

ο $\pm(i+3)$ " " " " $\pm(i+2)$,

⋮

ο $\pm(i+c-1)$ " " " " $\pm(i+c-2)$,

ο $\pm(i+c)$ " " " " $\pm(i+c-1)$

Άρα, λόγω μεταβατικής ιδιότητας, ο $\pm(i+c)$ είναι δεξιότερα του $\pm i$ $\Rightarrow$ ΑΤΟΠΟ.

Άρα, ισχύει η θέση που περιγράψαμε στην αρχή του ερωτήματος.

1. Απόδειξη ζητούμενου:

Κατάσταση 1: Όλοι οι αριθμοί στον At είναι αρνητικοί.

- Αν $At=[-1,-2,-3,\dots,-n]$, τότε τετριμμένα φαίνεται πως ΔΕΝ γίνεται να κατασκευαστεί συμβατό ζεύγος με 2 περιστροφές

- Αν $At\neq[-1,-2,\dots,-n]$ τότε $\exists -a\neq -n$ που να βρίσκεται δεξιότερα από τον $-(a+1)$, διαφορετικά, λόγω της απόδειξης στην αρχή του ερωτήματος και του ότι όλοι οι αριθμοί του At είναι αρνητικοί, θα ήταν $At=[-1,-2,\dots,-n]$.

Άρα: $[\underbrace{\dots,-(a+1)}, \dots, -a,\dots] \longrightarrow [\underbrace{a+1,\dots,\dots,\dots}, -a,\dots]$

$\longrightarrow [\dots,-(a+1),-a,\dots]$, όπου $(-(a+1),-a)$ συμβατό ζεύγος

Στην κατάσταση 2 αγνοούμε το max τμήμα $A_t[i] \longrightarrow A_t[n]$ που είναι ταξινομημένο όπως θα ήταν αν ολόκληρος ο πίνακας ήταν ταξινομημένος σωστά

Κατάσταση 2: $\exists A_t[i] \in \{1,\dots,n\}$ (δηλ. έστω 1 θετικό στοιχείο)

- Αν από τα θετικά $At[i]$ το μέγιστο είναι το n, τότε:
  Λόγω της δήλωσης θα είναι $A_t[n]\neq n$

  με δεδομένο πως η έννοια του συμβατού ζεύγους αποσκοπεί στην μείωση της διάστασης του πίνακα (βλ. επόμενο ερώτημα), η κίνηση: $[\dots,n,\dots] \rightsquigarrow [-n,\dots] \rightsquigarrow [\dots,n]$ αποτελεί κατασκευή συμβατού ζεύγους, αφού το n από εκεί και πέρα αγνοείται.

- Αν από τα θετικά $At[i]$ το μέγιστο είναι το $a\neq n$, τότε το $(a+1)$ εμφανίζεται με αρνητικό πρόσημο:

$\rightsquigarrow [\underbrace{\ldots, -(a+1), \ldots, a}, \ldots] \rightarrow [\underbrace{-a, \ldots}, a+1, \ldots] \rightarrow [\ldots, a, a+1, \ldots]$

$\rightsquigarrow [\underbrace{\ldots, a, \ldots, -(a+1)}, \ldots] \rightarrow [\underbrace{(a+1), \ldots, -a}, \ldots] \rightarrow [\ldots, -(a+1), -a, \ldots]$

Άρα, με το πολύ 2 περιστροφές έχουμε $\forall At \neq [-1, \ldots, -n]$ ένα νέο συμβατό ζεύγος.

2. • Αν $At = [-1, -2, \ldots, -n]$ τότε: (1) περίστρεψε όλο τον πίνακα
(2) περίστρεψε τα πρώτα $n-1$ στοιχεία
(3) επανάλαβε τα (1) κ (2) n φορές. (A)

Με τον αλγόριθμο (A) ταξινομείται κατ' αύξουσα σειρά ο At με την παραπάνω μορφή, με το πολύ $2n$ περιστροφές.

Απόδειξη:

Βάση επαγωγής: Για $n=0$ προφανώς ισχύει το ζητούμενο

Βήμα επαγωγής: Έστω ότι ισχύει και για n στοιχεία.

Παρατηρούμε ότι τρέχοντας τον αλγόριθμο (A), κάθε ~~δύο περιστροφές~~ φορά που τελειώνει το βήμα (2) ο πίνακας μεταβάλλεται ως εξής:

$[-1, -2, -3, \ldots, -(n-1), -n]$ 2 περ/φές

$[-2, -3, -4, \ldots, -n, 1]$ 4 περ/φές

$[-3, -4, -5, \ldots, -n, 1, 2]$ 6 περ/φές

$[-4, -5, -3, \ldots, -n, 1, 2, 3]$

$\vdots$

$[-n, 1, 2, \ldots, n-2, n-1]$ $2(n-1)$ περ/φές

$[1, 2, 3, \ldots, n-1, n]$ $2n$ περ/φές (όπως λέει και η υπόθεση)

Αν είχαμε n+1 στοιχεία, τότε θα ήταν ίδια όλα μέχρι το εξής σημείο:

$[-(n), -(n+1), 1, 2, 3, \dots, n-1]$ 2(n-1) περ/φές

$[-(n+1), 1, 2, \dots, n-1, n]$ 2n περ/φές

όπου θα χρειαστούμε 2 ακόμα περ/φές για να μπει το $-(n+1)$ στο τέλος με A1.

$[1, 2, 3, \dots, n-1, n, n+1]$ $2n+2 = 2(n+1)$

Άρα, ο $At=[-1,\dots,-n]$ ταξινομείται με 2n περ/φές, σύμφωνα με τον αλγόριθμο (A).

- Αν $At \neq [-1,\dots,-n]$ τότε σύμφωνα με το προηγούμενο ερώτημα, μπορούμε με το πολύ 2 περιστροφές να φτάξουμε ένα συμβατό ζεύγος, μειώνοντας τη διάσταση του πίνακα κατά 1 κάθε φορά, αφού από εκεί και πέρα το ζεύγος αντιμετωπίζεται ως ένας αριθμός (δεν το σπάμε ποτέ ή αν αυτό είναι απλώς το n στο τέλος του πίνακα δεν το πειράζουμε).
  ↳ το συμβατό ζεύγος

Κάθε φορά που δημιουργούμε συμβατό ζεύγος, αυτό δεν ξανασπάει και αντιμετωπίζεται σαν αριθμός, πχ το $(1,2) \to 2$

Αντιμετωπίζουμε τα συμβατά ζεύγη ως τον αριθμό που υπερτερεί κατά απόλυτη τιμή στο ζεύγος (με κατάλληλο πρόσημο). $(1,2) \to 2$

$(-2,-1) \to -2$

Έτσι, ∀ζεύγος μειώνεται κατά 1 η διάσταση του At και ο "αριθμός" που το "αντικατέστησε" μπορεί να φτάξει συμβατό ζεύγος με τον αμέσως ~~μεγαλύτερό~~ μικρότερο ή μεγαλύτερό του (κατά απόλυτη τιμή).

- πχ.1) $1,2 \to (1,2)=2$

2) $2,3 \to (2,3) = ((1,2),3) = (1,2,3) = 3$

πχ.1) $7,8 \to (7,8)=8$

2) $6,8 \to (6,8) = (6,(7,8)) = 6,7,8) = 8$ αυτό το "8" μπορεί να κάνει ζεύγος με το ±5 ή το ±9.

Βάσει των προηγουμένων:

→ Ή θα φτιάχνουμε ανά δύο περιστροφές συμβατά ζεύγη (συγχωνεύοντας τους αριθμούς τους και μειώνοντα τη διάσταση του At κατά 1) μέχρι ο πίνακας να ταξινομηθεί όλος με κόστος:

μέγεθος n → 2 περ

" $n-1$ → 4 περ

" $n-2$ → 6 περ

" $n-3$

⋮

" 3 → $2(n-3)$ περ

" 2 → $2(n-3)+2$ περ

" 1 → $2(n-3)+4$ περ $= 2n-2$ περ/φές

Στη χειρότερη, θα χρειαστεί και αλλαγή προσήμου στο τέλος → $2n-1$ περ. τελικά

→ Ή θα φτιάχνουμε νέα συμβατά ζεύγη ανά δύο περιστροφές μέχρι ο At να έχει μορφή

$[-1,-2,\ldots,-i,$ (ενιαίο ζεύγος σωστά ταξινομημένο)$]$ (αφού σε κάθε άλλη περίπτωση μπορούμε να συνεχίσουμε να φτιάχνουμε νέα συμβατά ζεύγη και να τα συγχωνεύουμε). Τότε ταξινομούμε το κομμάτι $[-1,\ldots,-i]$ με τον αλγόριθμο Α και έχουμε συνολικό κόστος (χειρότερης περίπτωσης):

- φτάνουμε στα x στοιχεία → $2(n-x)$
- $+2$ περ. για να πάει το μέγιστο στο τέλος → $2(n-x)+2$
- $+2(x-1)$ για να ταξινομηθούν τα $x-1$ μορφής $[-1,-2,\ldots,-(x-1)]$ → $2(n-x)+2+2(x-1) = 2n$ περιστροφές

Άσκηση 3: Παίζοντας χαρτιά

1. Αλγόριθμος: Τοποθετούμε κάθε κάρτα που τραβάμε στη στοίβα με τη μικρότερη κεφαλή που δεν υπολείπεται της αξίας της κάρτας.
Αν $\forall$στοίβα: κεφαλή > κάρτας $\longrightarrow$ δημιούργησε νέα στοίβα.

Ορθότητα: Ονομάζουμε:
- c_i την i-οστή κάρτα που τραβάμε
- W_i το πλήθος στοιβών μετά την τοποθέτηση της c_i
- a_i^k την κορυφή της k-οστής στοίβας -//-
$(k \le W_i)$

Θεωρούμε:
- αλγόριθμο A: εκτελεί τον αλγόριθμο του ερωτήματος
- αλγόριθμο B: τυχαίος αλγόριθμος για το παιχνίδι καρτών

Συμβολίζουμε:
- $[a_i^1, a_i^2, \ldots, a_i^{W_i}]$ τις κορυφές των στοιβών που προκύπτουν με χρήση του αλγορίθμου A, ταξινομημένες σε αύξουσα σειρά, όταν μόλις τοποθετήσαμε την κάρτα c_i.
- $[b_i^1, b_i^2, \ldots, b_i^{Z_i}]$ ομοίως για τον αλγόριθμο B

Ισχυρισμός: $\forall$αλγόριθμο B ισχύουν: (1) $W_i \le Z_i$

(2) $a_i^j \geq b_i^j$, $\forall j \in \{1, \ldots, W_i\}$

Θα χρησιμοποιήσουμε μαθηματική επαγωγή για να αποδείξουμε τον ισχυρισμό μας και, κατά συνέπεια, την ορθότητα του αλγορίθμου A, αφού θα έχουμε δείξει ότι $W_i \le Z_i$, $\forall$αλγ. B.

<u>Βάση Επαγωγής:</u> Για $i=1$, $W_1 = Z_1 = 1 \Longrightarrow$ ισχύει το (1)

$a_1^1 = b_1^1 = c_1 \Longrightarrow$ ισχύει το (2)

<u>Επαγωγική Υπόθεση:</u> Έστω ότι ισχύει το (1) και το (2) για τυχαίο $i>1$.

<u>Επαγωγικό Βήμα:</u> Τραβάμε την κάρτα c_{i+1} και υπάρχουν 2 εκδοχές για τη συνέχεια.

- <u>Εκδοχή 1:</u> Ο Α τοποθετεί την c_{i+1} σε νέα στοίβα

Αναγκαστικά θα πρέπει: $c_{i+1} > a_i^{W_i} \geq a_i^j \quad \forall j \in \{1,\dots,W_i\}$, αφού τα a_i^j είναι κατ'αύξουσα σειρά ταξινομημένα.

Ακόμη, από επαγωγική υπόθεση: $a_i^j \geq b_i^j$, $\forall j \in \{1,\dots,W_i\}$

$$\Longrightarrow a_i^{W_i} \geq b_i^{W_i} \geq b_i^j, \ \forall j \in \{1,\dots,W_i\}$$

αφού τα b_i^j είναι κατ'αύξουσα σειρά ταξινομημένα.

Άρα: $c_{i+1} > a_i^{W_i} \geq b_i^{W_i}$ (*)

- Αν $Z_i = W_i$, τότε λόγω της (*) και ο Β αναγκάζεται να φτιάξει νέα στοίβα, οπότε $Z_{i+1} = Z_i + 1 = W_i + 1 = W_{i+1} \Longrightarrow W_{i+1} = Z_{i+1}$

- Αν $Z_i > W_i$, τότε θα είναι: $\left.\begin{cases} \text{ή } Z_{i+1} = Z_i + 1 > W_i + 1 = W_{i+1} \\ \\ \text{ή } Z_{i+1} = Z_i \geq W_i + 1 = W_{i+1} \end{cases}\right\} \Longrightarrow Z_{i+1} \geq W_{i+1}$

Σε κάθε περίπτωση της εκδοχής 1: $Z_{i+1} \geq W_{i+1} \Longrightarrow$ ισχύει το (1)

- Αν ο Β δημιούργησε νέα στοίβα: τότε $\forall j \in \{1,\dots,W_i\}$ $a_{i+1}^j = a_i^j$ και $b_{i+1}^j = b_i^j$

$$\Longrightarrow a_{i+1}^j = a_i^j \geq b_i^j = b_{i+1}^j \Longrightarrow a_{i+1}^j \geq b_{i+1}^j$$

Ακόμη, $c_{i+1} = b_{i+1}^{Z_{i+1}} \geq b_{i+1}^{W_{i+1}}$ και $c_{i+1} = a_{i+1}^{W_{i+1}}$.

Άρα, $a_{i+1}^{W_{i+1}} \geq b_{i+1}^{W_{i+1}}$. Συνεπώς, $a_{i+1}^j \geq b_{i+1}^j$, $\forall j \in \{1,\dots,W_{i+1}\}$

• Αν ο B δεν δημιούργησε νέα στοίβα: τότε αφού

$c_{i+1} = a_{i+1}^{w_{i+1}} > a_{i+1}^{w_i} = a_i^{w_i} > b_i^j \;,\; \forall j \in \{1,\dots,w_i\} \Longrightarrow$

$c_{i+1} > b_i^{w_i} \Longrightarrow$ η c_{i+1} μπαίνει στην κορυφή δεξιότερης από τη θέση w_i στοίβας και γι'αυτό $b_{i+1}^j = b_i^j$ $\forall j \in \{1,\dots,w_i\}$

Άρα, $a_{i+1}^j = \underset{\substack{\text{από επ.}\\\text{υπόθεση}}}{a_i^j \geq b_i^j} = b_{i+1}^j \;,\; \forall j \in \{1,\dots,w_i\}$.

Τη χρονική στιγμή $i+1$, η στοίβα b με κορυφή το c_{i+1} θα είναι στη θέση w_{i+1} ή δεξιότερα, άρα $b_{i+1}^{w_{i+1}} \leq c_{i+1} = a_{i+1}^{w_{i+1}}$

Άρα, $a_{i+1}^j \geq b_{i+1}^j \;,\; \forall j \in \{1,\dots,w_{i+1}\}$

Σε κάθε περίπτωση της εκδοχής 1: $a_{i+1}^j \geq b_{i+1}^j \;,\; \forall j \in \{1,\dots,w_{i+1}\}$

$\Longrightarrow$ ισχύει το (2)

• Εκδοχή 2: Ο A τοποθετεί τη c_{i+1} σε ήδη υπάρχουσα στοίβα

Έστω m η στοίβα αυτή: $1 \le m \le w_i = w_{i+1}$

Προφανώς, $z_{i+1} = z_i$ ή $z_{i+1} = z_i + 1$, οπότε από επαγωγική υπόθεση έχουμε $z_i \geq w_i = w_{i+1} \Longrightarrow$ $w_{i+1} = w_i \le z_i \le z_{i+1} \Rightarrow w_{i+1} \le z_{i+1}$

$\boxed{\Longrightarrow \text{ισχύει το (1)}}$

Σύμφωνα με τη θέση Θ, την οποία αποδεικνύουμε στο τέλος του ερωτήματος, η κάρτα c_{i+1} που τοποθετείται από τον A αλγόριθμο γίνεται κορυφή της στοίβας m και οι στοίβες παραμένουν ταξινομημένες σε αύξουσα σειρά.

Άρα $c_{i+1} > a_i^{m-1} \geq b_i^{m-1} \geq b_i^j$, $\forall j \in \{1, \dots, m-1\} \Longrightarrow$

$c_{i+1} > b_i^j$, $\forall j \in \{1, \dots, m-1\}$, άρα ο B ΔΕΝ μπορεί να τοποθέτησε την c_{i+1} πάνω σε καμία από τις πρώτες $m-1$ στοίβες. Άρα, όπου και να μπει η c_{i+1}, οι $m-1$ πρώτες στοίβες b_{i+1} θα είναι αυτές που ήταν πριν μπει η c_{i+1} κάπου. Δηλ. $b_{i+1}^j = b_i^j$ $\forall j \in \{1, \dots, m-1\}$

Άρα, $a_{i+1}^j = a_i^j \underset{\text{επ.υποθ.}}{\geq} b_i^j = b_{i+1}^j \Longrightarrow$ $a_{i+1}^j \geq b_{i+1}^j$, $\forall j \in \{1, \dots, m-1\}$ (* *)

• Περίπτωση 1: η c_{i+1} μπαίνει πάνω στο b_i^h, $h \in \{m, z_{i+1}\}$ και η διάταξη παραμένει ταξινομημένη. Τότε:

$$b_{i+1}^j = b_i^j \quad \forall j \in \{m, h-1\} \cup \{h+1, z_{i+1}\} \text{ και}$$

• $j \in \{h+1, w_{i+1}\}$ αν ορίζεται: $b_{i+1}^j = b_i^j \le a_i^j = a_{i+1}^j \Rightarrow a_{i+1}^j \ge b_{i+1}^j$

• $j \in \{m+1, \min\{h-1, w_{i+1}\}\}$: $b_{i+1}^j = b_i^j \le a_i^j = a_{i+1}^j \Rightarrow a_{i+1}^j \ge b_{i+1}^j$

• $j = m$: $\underbrace{\overset{b_{i+1}^h =}{c_{i+1}} \ge b_{i+1}^m}_{\text{αφού είναι ταξινομημένη}} \Rightarrow a_{i+1}^m \ge b_{i+1}^m$

• $j = h$: αν $h = m$, ό,τι από πάνω

αν $h > m$, $b_{i+1}^h = c_{i+1} < b_i^h \le a_i^h = a_{i+1}^h$
$\Rightarrow a_{i+1}^h \ge b_{i+1}^h$

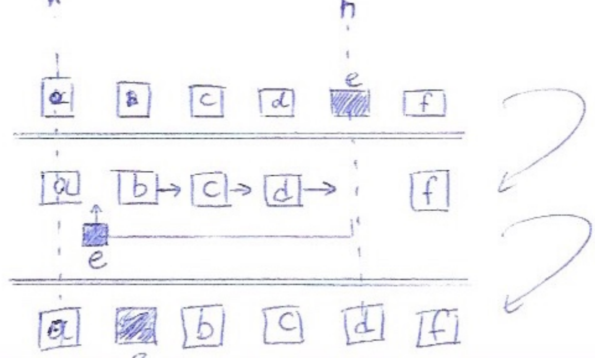
Άρα, $\forall j \in \{m, \dots, w_{i+1}\}$ $a_{i+1}^j \ge b_{i+1}^j$, και σε συνδυασμό με την $(**)$

$$\Rightarrow \boxed{a_{i+1}^j \ge b_{i+1}^j \quad \forall j \in \{1, \dots, w_{i+1}\}}$$

• Περίπτωση 2: η c_{i+1} μπαίνει πάνω στο b_i^h, $h \in \{m, z_{i+1}\}$ και η διάταξη δεν είναι πλέον ταξινομημένη.

Για να ταξινομηθεί η διάταξη μετατοπίζουμε την b_i^h με κεφαλή την c_{i+1} προς τα αριστερά και την βάζουμε δεξιά από τη στοίβα με τη μεγαλύτερη κεφαλή που δεν υπερβαίνει το c_{i+1}, έστω b_i^k (προφανώς $k \ge m-1$), και ολισθαίνουμε τις στοίβες που προσπέρασε μια θέση δεξιά.

Σχηματικά:



• $j \in \{h+1, N_{i+1}\}$, αν ορίζεται: $a^j_{i+1} \geq b^j_{i+1}$ εμφανώς

$$\left(a^j_{i+1} = a^j_i \geq b^j_i = b^j_{i+1}\right)$$

• $j \in \{k+2, \min\{N_{i+1}, h\}\}$: $a^j_{i+1} \geq b^j_{i+1}$, διότι

έστω

Δ1: (a) (b) (c), αύξουσα

Δ2: (g) (d) (e) (f), αύξουσα

και $a>d$, $b>e$, $c>f$

τότε στις Δ1: (a) (b) (c)

Δ_2': (g) (d) (e)

είναι $a>d>g \Rightarrow a>g$

$b>e>d \Rightarrow b>d$

$c>f>e \Rightarrow c>e$

• $j = k+1$: $b^{k+1}_{i+1} = C_{i+1} = a^m_{i+1} \leq a^{k+1}_{i+1}$

$\Longrightarrow b^{k+1}_{i+1} \leq a^{k+1}_{i+1}$, με το ίσον να ισχύει για $k = m-1$

• $j \in \{m, k\}$ (αν ορίζεται) τότε όλα αμετάβλητα

οπότε $a^j_{i+1} \geq b^j_{i+1}$, εκτός του $j=m$

για το οποίο έχουμε $a^m_{i+1} = C_{i+1} = b^{k+1}_{i+1} \geq b^m_{i+1} \Rightarrow a^m_{i+1} \geq b^m_{i+1}$ (το = για $k=m-1$)

Άρα $\forall j \in \{m, \dots, N_{i+1}\}$ $a^j_{i+1} \geq b^j_{i+1}$ και σε συνδυασμό με την (**)

$$\Longrightarrow \boxed{a^j_{i+1} \geq b^j_{i+1}, \ \forall j \in \{1, \dots, N_{i+1}\}}$$

Άρα σε κάθε περίπτωση της εκδοχής 2, ισχύει το (2)

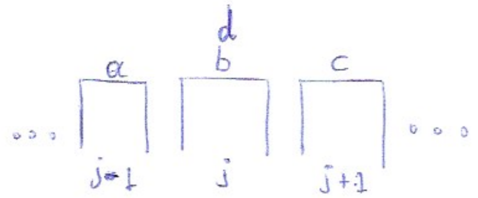
Τελικά, $\forall$ εκδοχή ισχύει το (1) και το (2) $\Longrightarrow$ η επαγωγή ολοκληρώθηκε.

Θέση Θ: Ο αλγόριθμος (Α) οδηγεί πάντα σε στοίβες που οι κεφαλές τους είναι ταξινομημένες από αριστερά προς τα δεξιά κατά αύξουσα τιμή.

Βάση: Για $i=1$ κάρτα ισχύει

Υπόθεση: Έστω ότι ισχύει για n κάρτες

Βήμα: Η επόμενη κάρτα (έστω d (η $n+1$-οστή)) τοποθετείται στη στοίβα j



$a<b<c$ από υπόθεση

- πρέπει $b>d$ και από υπόθεση $c>b \Longrightarrow c>d$
- πρέπει το b να είναι η μικρότερη κάρτα-κεφαλή που δεν υπολείπεται του d. Είναι $a<b$, αλλά η d μπήκε πάνω από το b. Άρα το a υπολείπεται του $d \Longrightarrow a<d$.

Άρα $a<d<c \longrightarrow$ για $n+1$ κάρτες, οι κεφαλές είναι κατ' αύξουσα σειρά ταξινομημένες.

Η χειρότερη περίπτωση απαιτεί και τις περισσότερες συγκρίσεις.
Ο μη βελτιστοποιημένος αλγόριθμος κάνει ∀κάρτα σύγκριση
με όλες τις υπάρχουσες στοίβες. Άρα για n κάρτες γίνονται

$0+1+2+3+\dots+n-1$ συγκρίσεις $=$

$$(1+2+3+\dots+n-1+n)-n = \frac{n(n-1)}{2} - n = \frac{n^2}{2} - \frac{n}{2} - n = \frac{n^2}{2} - \frac{3n}{2} \text{ συγκρίσεις}$$

Πολυπλοκότητα: $\Theta(n^2)$

Βελτίωση: Γνωρίζουμε ότι, σύμφωνα με τη θέση Θ, ο αλγόριθμος A δημιουργεί στοίβες με κεφαλές που είναι ταξινομημένες σε αύξουσα σειρά.
Άρα για να βρούμε τη στοίβα που θα μπεί κάθε κάρτα, κάνουμε δυαδική αναζήτηση ~~για να βρούμε~~ αναζητώντας την κάρτα ανάμεσα σε ποιές κεφαλές στοιβών βρίσκεται η αξία της κάρτας. Από αυτές διαλέγουμε τη μεγαλύτερη για να τοποθετήσουμε την κάρτα.
Αν η αναζήτηση μας οδηγήσει στην εντελώς δεξιά στοίβα και

- κάρτα > κεφαλής $\longrightarrow$ φτιάχνουμε νέα στοίβα.
- κάρτα < κεφαλής $\longrightarrow$ βάλε την κάρτα ως νέα κεφαλή

Αν μας οδηγήσει στην εντελώς αριστερή στοίβα, και κάρτα<κεφαλής $\longrightarrow$ βάλε την κάρτα ως νέα κεφαλή

Έτσι, στη χειρότερη περίπτωση έχουμε n κάρτες με $\log(\#\text{στοίβες})$ συγκρίσεις για κάθε μία, αλλά πάντα $n \geq \#\text{στοίβες} \Rightarrow \log n \geq \log(\#\text{στοιβών})$

$\Longrightarrow$ πολυπλοκότητα: $n\log(\#\text{στοιβών}) \leq n\log n \Longrightarrow \boxed{O(n\log n)}$

2. Εκτελούμε τον αλγόριθμο A : κόστος $O(n\log n)$

Τώρα έχουμε x στοίβες, η καθεμιά ταξινομημένη.

Για τη συγχώνευση των στοιβών ανά δύο, σύμφωνα με τον αλγόριθμο merge(), έχουμε κόστος #(κάρτες ζεύγους) −1 $\forall$ ζεύγος στοιβών,

δηλ. $$\sum_{i=1}^{x/2}\left[\#(\text{κάρτες i-οστού ζεύγους}) - 1\right]$$ συνολικά

$$= \sum_{i=1}^{x/2}\left[\#(\text{κάρτες i-οστού ζεύγους})\right] - \frac{x}{2}$$

$$= n - \frac{x}{2}, \text{ με } x \leq n.$$

Κόστος για τη διαίρεση του (#στοιβών) δια δύο (μένουν x/2 στοίβες) = K.

$$n/2 \leq K \leq n \implies K = \Theta(n).$$

Οι στοίβες θα μένουν κάθε φορά μισές σε πλήθος.

$\log_2$(#στοιβών) φορές μέχρι να μείνει 1 στοίβα.

Άρα θα κάνουμε log(#στοιβών) φορές $\Theta(n)$ λειτουργίες.

log(#στοιβών) $\leq \log_2 n$. Άρα τελικό κόστος $O(n\log n)$ για την ταξινόμηση.

Συνολικά ο αλγόριθμος κοστίζει: $O(n\log n) + O(n\log n) = O(n\log n)$

3. Είσοδος: 3, 2, 4, 7, 8, 1, 5, 6

στοίβες

Αλγόριθμος ερωτήματος 1

1. 2, 4, 7, 8, 1, 5, 6

   3

2. 4, 7, 8, 1, 5, 6

   2
   3

3. 7, 8, 1, 5, 6

   2
   3, 4

4. 8, 1, 5, 6

   2
   3, 4, 7

5. 1, 5, 6

   2
   3, 4, 7, 8

6. 5, 6

   1
   2
   3, 4, 7, 8

7. 6

   1
   2   5
   3, 4, 7, 8

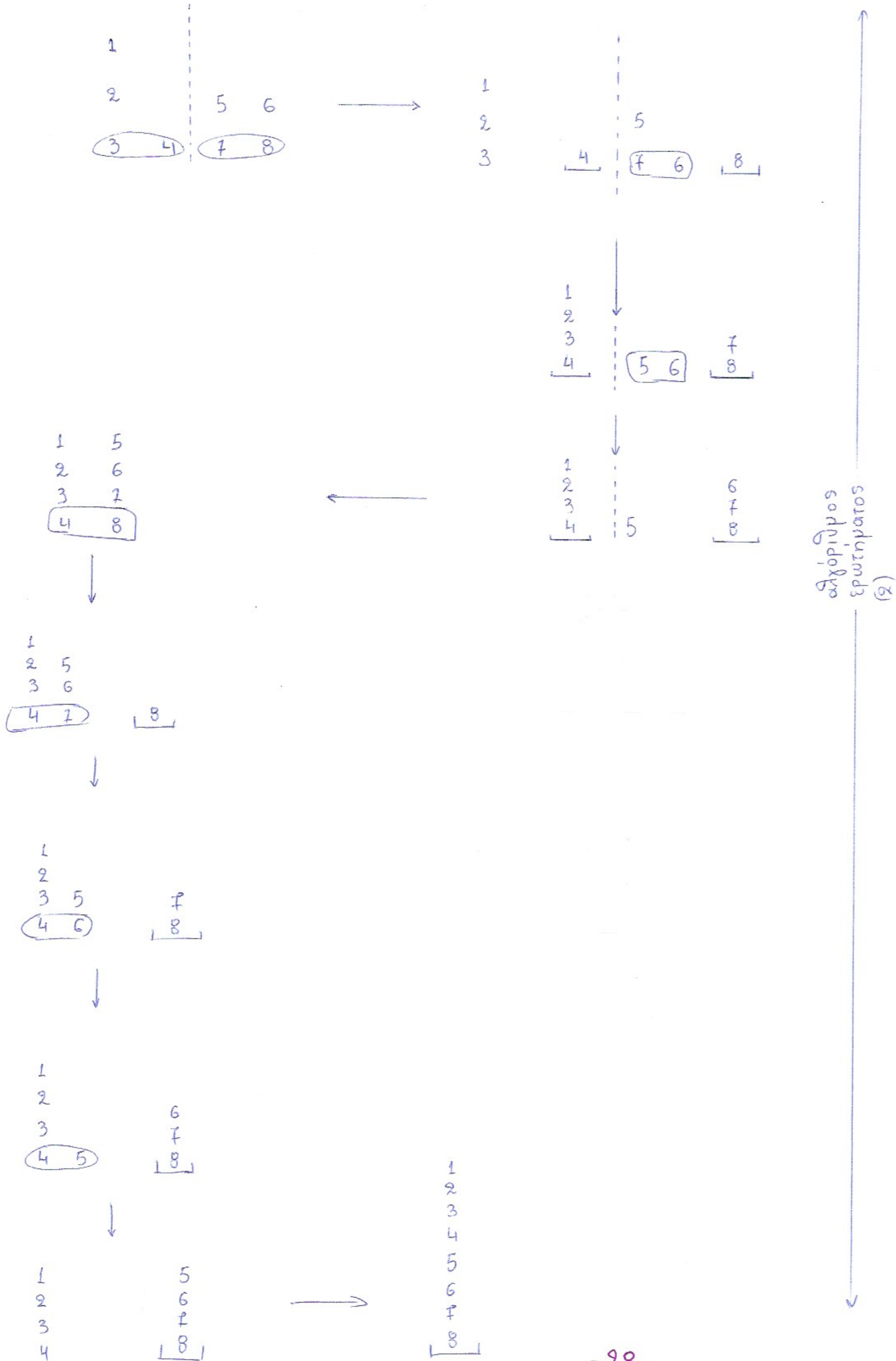
8. ∅

   1
   2   5   6
   3, 4, 7, 8

προκύπτουν 4 στοίβες.

το μήκος της μέγιστης αύξουσας υποακολουθίας εισόδου είναι 4 (μια από αυτές είναι η 3 4 7 8)

(οι νέες στοίβες έχουν το σύμβολο ⊔ στον πυθμένα τους)



4. Στο ερώτημα (1) δείξαμε πως ο αλγόριθμος A δημιουργεί τον ελάχιστο αριθμό στοιβών. Οπότε, αν δείξουμε πως ο αριθμός στοιβών που δημιουργείται από τον A είναι μεγαλύτερος ή ίσος με το μήκος της μέγιστης αύξουσας υποακολουθίας εισόδου, τότε θα ισχύει το τελικό ζητούμενο.

Δηλ. #στοιβών αλγ. T ≥ #στοιβών αλγ. A ≥ μήκος max_inc_sub

⟹ <u>μήκος max_inc_sub ≤ #στοιβών αλγ. T</u>, όπου T: οποιοσδήποτε αλγόριθμος κατανομής των καρτών.

Θα δείξουμε το ζητούμενο με μαθηματική επαγωγή, με χρήση της θέσης Θ που αποδείξαμε στο ερώτημα 1.

(Β) { Ισχυρισμός: Ο #στοιβών που δημιουργεί ο A είναι μεγαλύτερος ή ίσος με τη μέγιστη αύξουσα υποακολουθία εισόδου.

Ο ισχυρισμός μας είναι αληθής αν ισχύει ότι:

(Γ)
- σε κάθε στοίβα υπάρχει ή 1 ή 0 στοιχεία από κάθε αύξουσα υπρ/ία

και
- τα μεγαλύτερα στοιχεία κάθε υποακολουθίας (αύξουσας) βρίσκονται σε δεξιότερες στοίβες.

και
- αν υπάρχουν περισσότερες από μία ακολουθίες αύξουσες, τότε η καθεμιά πληροί τις δύο πρώτες κουκίδες.

<u>Βάση Επαγωγής:</u> Για i=1 αριθμό ως είσοδο, τετριμμένα ισχύουν οι ισχυρισμοί (Γ)

<u>Υπόθεση Επαγωγής:</u> Έστω ότι οι ισχυρισμοί (Γ) ισχύουν για i αριθμούς (κάρτες) εισόδου.

<u>Βήμα Επαγωγής:</u> Έχουμε αύξουσες υποακολουθίες που πληρούν τους (Γ) και έρχεται στην είσοδο η (i+1)-οστή κάρτα.

Έστω ότι η $(i+1)$-οστή κάρτα μπορεί να μπει στο τέλος της τυχαίας - εκ των όποιων υπάρχουν - αύξουσας υποακολουθίας, ονόματι:

$[a_1, a_2, \ldots, a_{k-1}, a_k]$.

Θα πρέπει: αξία $(i+1)$-οστής κάρτας $> a_k >$ κεφαλή στοίβας $>$ κεφαλές όλων των προηγούμενων στοιβών,

όπου η τελευταία ανισότητα ισχύει σύμφωνα με τη Θέση Θ.

Άρα αξία $(i+1)$-οστής κάρτας $>$ κεφαλές στοιβών που είναι από το a_k και πίσω.

Άρα η $(i+1)$-οστή κάρτα πρέπει να μπει δεξιότερα από το μέγιστο στοιχείο της τυχαίας (άρα και από όλες όσες επεκτείνει) ακολουθίας, σε στοίβα που μέχρι τον i-κάρτα δεν είχε κανένα στοιχείο της αυτής ακολουθίας.

Άρα, $\forall$ αύξουσα ακολουθία που προκύπτει με την $(i+1)$-οστή κάρτα, ισχύουν οι ισχυρισμοί (Γ), άρα και ο (Β), άρα και το ζητούμενο.

5.

Σε όλα τα παρακάτω, η κατασκευή των στοιβών, γίνεται βάσει του A.

Ισχυρισμός 1: Το κορυφαίο στοιχείο της k-οστής στοίβας είναι το τελευταίο στοιχείο τουλάχιστον μιας αύξουσας υποακολουθίας με k ακριβώς στοιχεία (μαζί με το ίδιο).

Ισχυρισμός 2: Ίδιος με τον 1, αλλά αντί για το κορυφαίο στοιχείο αναφέρεται σε κάθε στοιχείο της k-οστής στοίβας.

Βάση Επαγωγής: Για $k=1$ στοίβα, τετριμμένα ισχύει ο ισχυρισμός 2.

Υπόθεση: Έστω ότι ισχύει ο ισχυρισμός 2 για τις πρώτες k στοίβες.

Βήμα: Έστω C κάποιο στοιχείο της $(k+1)$-οστής στοίβας.

Τουλάχιστον 1 στοιχείο από αυτά που έχει η στοίβα k υπήρχε πριν την τοποθέτηση του C. Όταν τοποθετούνταν το C στην $(k+1)$-οστή στοίβα, η στοίβα k είχε κεφαλή, έστω y. Για να μπήκε το C στην $(k+1)$-οστή στοίβα θα πρέπει $y<C$ (σύμφωνα με τον αλγόριθμο A).

Άρα, $\forall C\ \exists y$: $y<C$ & y προγενέστερο του C.

Άρα $\forall C$ επιμηκύνει κατά 1 μια υποακολουθία που τελείωνε στο αντίστοιχο y με μήκος k.

Ακόμη, στο ερ. 4 δείξαμε ότι κάθε στοίβα περιέχει 1 ή κανένα στοιχείο από κάθε αύξουσα υποακολουθία.

Τελικά, συνοπτικά προκύπτει ότι κάθε στοιχείο της (k+1)-οστής στοίβας είναι τέλος μιας αύξουσας υποακολουθίας με ακριβώς k+1 στοιχεία (με αυτό μαζί).

Άρα ισχύει ο ισχυρισμός 2 $\Longrightarrow$ ισχυρισμός 1.

Στο ερ. 4 δείξαμε ότι το μήκος της max αύξουσας υποακολουθίας είναι μικρότερο ή ίσο από τις στοίβες του αλγορίθμου A.

Στο ερ. 5 δείξαμε ότι κάθε στοιχείο της τελευταίας στοίβας αποτελεί τέλος αύξουσας υποακολουθίας με πλήθος στοιχείων ακριβώς τόσο όσο ο αύξων αριθμός της στοίβας = # στοιβών.

Τα δύο παραπάνω συναληθεύουν $\Longrightarrow$ οι στοίβες του A είναι τόσες όσο το μήκος της μέγιστης αύξουσας υποακολουθίας.

Έστω n κάρτες και x, y 2 εξ' αυτών με το x πρώτο από το y χρονικά και $x > y$.

Είναι $-x < -y \Rightarrow n - x < n - y \Longrightarrow \boxed{(n+1) - x < (n+1) - y}$

Άρα αν αντικαταστήσουμε κάθε κάρτα z με την $(n+1)-z$, οι ανισοτικές σχέσεις που είχε με τις υπόλοιπες κάρτες εισόδου αναστρέφονται. Άρα οι αύξουσες ακολουθίες της μιας εισόδου (που έχει τα x, y) είναι οι φθίνουσες της άλλης (που έχει τα $(n+1)-x$, $(n+1)-y$) και αντίστροφα.

Αν θέλουμε, λοιπόν, να βρούμε το μήκος της μέγιστης φθίνουσας υποακολουθία, ανακαθιστούμε κάθε $x \in \{1,\dots,n\}$ με το $(n+1)-x$ και εκτελούμε τον αλγόριθμο A. Ο αριθμός στοιβών που προκύπτουν θα είναι και το μήκος της μέγιστης φθίνουσας υποακολουθίας της αρχικής τράπουλα

6. Έστω τυχαίο ανακάτεμα $nm+1$ φύλλων.

Έστω ότι ο αλγόριθμος (1) δημιούργησε x στοίβες με την i-οστή να έχει y_i φύλλα.

$$\left.\begin{array}{l}\text{Επίσης } y_i < m+1, \ \forall i. \\ \text{και } \quad x < n+1\end{array}\right\} \Longrightarrow \left\{\begin{array}{l} y_i \leq m \\ x \leq n \end{array}\right., \forall i. \Longrightarrow x \cdot y_{max} \leq n \cdot m$$

$$\text{Άρα: } n \cdot m + 1 = \sum_{i=1}^{x} y_i \leq \sum_{i=1}^{x} y_{max} = x \cdot y_{max} \leq n \cdot m$$

$$\Longrightarrow nm + 1 \leq nm \text{ , άτοπο.}$$

Άρα είτε έχουμε τουλάχιστον $n+1$ στοίβες
είτε κάποια στοίβα έχει τουλ. $m+1$ κάρτες.

Τα παραπάνω συνδέονται και συμφωνούν με την Αρχή του Περιστερών των Διακριτών Μαθηματικών.

Μια ακολουθία εισόδου όπως αυτή που ζητείται είναι η:

21 - 22 - 23 - 24 - 25 - 16 - 17 - 18 - 19 - 20 - 11 - 12 - 13 - 14 - 15 - 6 - 7 - 8 - 9 - 10 - 1 - 2 - 3 - 4 - 5.

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

Άσκηση 4: Γρήγορη Επιλογή στο Πεδίο της Μάχης

1. Έστω ότι τα πρώτα σωματίδια που συγκρούονται κινούνται:

→ με μέγιστη ταχύτητα: τότε συγκρούνται το συντομότερο δυνατό στη θέση $L/2$, έχοντας διανύσει $L/2$ το καθένα. $L/2 = V_{max} \cdot t_{min} \Longrightarrow t_{min} = \frac{L}{2V_{max}}$

→ με ελάχιστη ταχύτητα: τότε συγκρούονται το αργότερο δυνατό στη θέση $L/2$, έχοντας διανύσει $L/2$ το καθένα. $L/2 = V_{min} \cdot t_{max} \Longrightarrow t_{max} = \frac{L}{2V_{min}}$

Επομένως, η πρώτη σύγκρουση δεν μπορεί παρά να συμβεί στο χρονικό διάστημα $[t_{min}, t_{max}] = \left[\frac{L}{2V_{max}}, \frac{L}{2V_{min}}\right]$.

Διακριτοποιούμε το διάστημα αυτό, θεωρώντας πως η στιγμή που μας ενδιαφέρει να βρούμε είναι πολλαπλάσιο του x sec (πχ. msec, nsec...). Τότε κάθε "κόβεται" σε $\frac{1}{x}$ κομμάτια, οπότε θα περιλάβει $\frac{1}{x}$ στιγμές + 1.

Επομένως, οι στιγμές που έχουμε να "δουλέψουμε" είναι $\frac{1}{x}\left(\frac{L}{2V_{min}} - \frac{L}{2V_{max}}\right) + 1$
$= \Theta\left(L\left(\frac{1}{V_{min}} - \frac{1}{V_{max}}\right)\right)$

Σε αυτό το σύνολο των χρονικών στιγμών, γνωρίζουμε $\forall$ χρ. στιγμή ποιά ήταν η θέση κάθε σωματιδίου. Άρα, αν κάποια στιγμή όλα τα σωματίδια α είναι πιο αριστερά από όλα τα σωματίδια β, τότε η πρώτη σύγκρουση έγινε αργότερα, ενώ αν κάποια χρονική στιγμή κάποιο σωματίδιο α βρίσκεται δεξιότερα από κάποιο β, τότε η πρώτη σύγκρουση έχει προηγηθεί.

Αλγόριθμος Binary Search - λίγο τροποποιημένος:

- ξεκινά από τη μέση του (προφανώς ταξινομημένου) πίνακα στιγμών και για τη στιγμή t_0 που εξετάζεις, δώσε είσοδο στον υπερυπολογιστή (i, t_0), $\forall i$ (τα i είναι όλα: $n+n=2n$) κόστος σε κλήσεις: $O(n)$.

$c \cdot 2n = n \cdot O(1)$

- αν στην t_0, όλα τα a είναι αριστερά των b, επανάλαβε το πρώτο βήμα για το δεξί μισό του πίνακα
- αν στην t_0, κάποιο a είναι δεξιότερα ~~του~~ ενός b, επανάλαβε το πρώτο βήμα για το αριστερό μισό του πίνακα
- αλλιώς η t_0 είναι η ζητούμενη στιγμή

Σε κάθε επανάληψη (αναδρομή) του αλγορίθμου ο πίνακας στιγμών "κόβεται" στη μέση.

Άρα συνολικό κόστος αλγορίθμου (στη χειρότερη περίπτωση):

$$\left.\begin{array}{l} O(n) \text{ κλήσεις} \\ O\left(\log\left(L\left(\frac{1}{V_{min}} - \frac{1}{V_{max}}\right)\right)\right) \text{ φορές} \end{array}\right\} \sim O\left(n \log\left(L \cdot \left(\frac{1}{V_{min}} - \frac{1}{V_{max}}\right)\right)\right)$$

2. Η πρώτη σκέψη μας είναι να βρούμε για κάθε ζευγάρι σωματιδίων (a, b) τη στιγμή σύγκρουσης και να κρατήσουμε ως λύση την συντομότερη.

$$\left.\begin{array}{l} O(1) \text{ χρόνος για να βρούμε τη στιγμή σύγκρουσης} \\ O(n^2) \text{ χρόνος για όλα τα ζεύγη σωματιδίων} \end{array}\right\} \Rightarrow O(n^2) \text{ πολυπλοκότητα}$$

Προσπαθούμε να βελτιώσουμε την πολυπλοκότητα μειώνοντας τον αριθμό των υποψήφιων για την πρώτη σύγκρουση σωματιδίων. Για να το κάνουμε αυτό, εκτελούμε τον παρακάτω αλγόριθμο, ο οποίος είναι κατά βάση όμοιος με τον Quickselect.

Διαλέγουμε ένα σωματίδιο από τα a και βρίσκουμε τους θεωρητικούς χρόνους σύγκρουσής του με όλα τα $b \Rightarrow O(n)$.

Έστω t_0 ο συντομότερος χρόνος. Την χρονική στιγμή t_0 όλα τα σωματίδια b βρίσκονται στο σημείο της σύγκρουσης του a που διαλέξαμε και του b με το οποίο συγκρούεται τη στιγμή t_0 ή πιο δεξιά.
Αν έστω 1 σωματίδιο b ήταν πιο αριστερά, τότε θα είχε προηγηθεί η σύγκρουσή του με το επιλεγμένο a και θα είχε χρόνο σύγκρουσης με το επιλεγμένο $a < t_0 \Rightarrow$ άτοπο.
Άρα, όσα σωματίδια a βρίσκονται πίσω από το a που συγκρούεται δεν έχουν έρθει σε επαφή με κανένα b και άρα οι συγκρούσεις τους με σωματίδια b θα γίνουν μετά την t_0. Άρα μπορούμε να τα απορρίψουμε ως υποψήφια σωματίδια πρώτης σύγκρουσης.

Γνωρίζουμε, από την ανάλυση πολυπλοκότητας της Quickselect, πως τα σωματίδια που απορρίπτονται είναι στη μέση περίπτωση και με μεγάλη πιθανότητα τουλάχιστον $n/4$.

Επαναλαμβάνοντας, έτσι, την παραπάνω διαδικασία, μειώνοντας όλο και περισσότερο τα σωματίδια α, καταλήγουμε στο 1 σωματίδιο α που συγκρούστηκε πρώτο.

Για να απορρίψουμε κάποια α σωματίδια, βρίσκουμε τους χρόνους σύγκρουσης καθενός με το σωματίδιο β που συγκρούεται με το επιλεγμένο α την t_0. Όσα έχουν χρόνο $\geq t_0$, απορρίπτονται (είναι αριστερά της σύγκρουσης). Αυτό σε κάθε επανάληψη κοστίζει (κάτι $\leq n$).

Άρα έχουμε στη μέση περίπτωση:

- n + (κάτι $\leq n$) υπολογισμούς $= O(n)$
- $O(1)$ ο κάθε υπολογισμός
- $\log_{4/3} n$ επαναλήψεις

$\Longrightarrow O(n\log n)$ πολυπλοκότητα μέσης περίπτωσης.

Στη συνέχεια, μπορούμε να κάνουμε την εξής παρατήρηση.

Η quickselect γενικά έχει πολυπλοκότητα μέσης περίπτωσης $O(n)$ (όπως αναλύεται στις διαφάνειες του μαθήματος).

Ο δικός μας αλγόριθμος, όμως αναγκάζεται να κάνει $\sqrt{n}$ τουλ. φορές $\Theta(1)$ υπολογισμούς σε κάθε επανάληψη, αφού τα β σωματίδια είναι πάντα n σε πλήθος.

Αν, λοιπόν, τον παραλλάξουμε ώστε να απορρίπτει σωματίδια και από τα τύπου β, οι υποχρεωτικοί υπολογισμοί θα μειώνονται σε κάθε επανάληψη.

Εφαρμόζουμε, λοιπόν, την παραπάνω στρατηγική εναλλάξ στα τύπου α και τύπου β σωματίδια, ώστε να μειώνονται και τα μεν και τα δε και οι υποχρεωτικοί υπολογισμοί να μειώνονται, επίσης (όπως συμβαίνει και με την κλασσική quicksort).

Τότε, θα έχουμε κόστος:

- $\Theta(1)$ ∀ υπολογισμό

- $2 \cdot \log_{4/3} n$ κλήσεις

- η γραμμική αναζήτηση επιφέρει × πράξεις που σε κάθε 2 αναδρομές πολ/ζονται με 3/4:

1) n
2) $3n/4$
3) $3n/4$
4) $9n/16$
5) $9n/16$

⋮

$$\Rightarrow 2 \cdot \sum_{i=1}^{\log_{4/3} n} \left(\frac{3}{4}\right)^{i-1} \cdot n =$$

$$= 2 \cdot n \cdot \frac{1 - (3/4)^{\log_{4/3} n - 1}}{1/4}$$

$$= 8n \cdot \left(1 - \frac{4}{3} \cdot n^{-1}\right)$$

$$= 8n - \frac{32}{3} = O(n)$$

Άσκηση 5: Ερωτήματα Ανήκειν

Θεωρούμε k hash συναρτήσεις, έστω $h_1, h_2, h_3, \dots, h_k$, και τον πίνακα $[m]$ της εκφώνησης

$$h_i : \mathbb{N}_+ \longrightarrow [m], \quad \forall i \in \{1, \dots, k\} \quad \text{τ.ω.} \quad \mathrm{Prob}[h_i(x) = j] = \frac{1}{m} \quad \forall x \in \mathbb{N}_+ \text{ και } \forall j \in [m]$$

Δηλαδή:
- οι h_i είναι ανεξάρτητες μεταξύ τους
- οι θέσεις του $[m]$ επιλέγονται από κάποια h_i με ίση πιθανότητα

Άρα η όταν κάποια h_i κάνει επιλογή θέσης, η πιθανότητα να επιλεγεί κάποια συγκεκριμένη θέση είναι $1/m$.

Άρα η πιθανότητα να μην επιλεγεί κάποια συγκεκριμένη θέση είναι:

$$1 - \frac{1}{m}.$$

Άρα η πιθανότητα να μην επιλεγεί κάποια συγκεκριμένη θέση από καμία εκ των h_i, όταν αυτές hash-άρουν το ίδιο στοιχείο $\in S$, είναι

$$\left(1 - \frac{1}{m}\right)^k$$

Άρα όταν θα έχουμε hash-άρει όλα τα στοιχεία του S (και με τις k συναρτήσεις) η πιθανότητα κάποιο συγκεκριμένη θέση του $[m]$ να μην έχει επιλεγεί ακόμα είναι: $\left[\left(1 - \frac{1}{m}\right)^k\right]^n = \left(1 - 1/m\right)^{k \cdot n}$ και η πιθανότητα να έχει επιλεγεί είναι $1 - \left(1 - 1/m\right)^{k \cdot n}$

Έχοντας, λοιπόν, ένα στοιχείο $x \notin S$ και ελέγχοντας αν αυτό υπάρχει (ανήκει) στο S, η πιθανότητα να είναι 1 όλες οι θέσεις του $[m]$ στις οποίες αντιστοιχίζουν οι k hash functions το x είναι:

$$\left[1-\left(1-\frac{1}{m}\right)^{k \cdot n}\right]^{k}$$

, όπου έχουμε υποθέσει ανεξαρτησία των πιθανοτήτων να είναι καθένα bit ίσο με 1.

Αποδεικνύεται πως η προσέγγιση αυτή (με την από πάνω υπόθεση) είναι πολύ κοντά στην πραγματικότητα.

Η πιθανότητα για false positive απάντηση είναι:

$$\left[1-\left(1-\frac{1}{m}\right)^{k \cdot n}\right]^{k} = \left[1-\left(1-\frac{1}{m}\right)^{m \cdot \frac{kn}{m}}\right]^{k} \xrightarrow{m:\text{μεγάλο}} \cancel{1-1}\left(1-e^{-kn/m}\right)^{k}$$

(α) $m = 8n$ και $k=1$: $P[\text{false positive}] \approx 1 - e^{-1 \cdot \frac{n}{8n}} = 1 - e^{-1/8} \approx 0{,}118$

(β) $m = 8n$ και $k \geq 1$: $P[\text{false positive}] \approx 1 - e^{-1 \cdot kn/8n} = 1 - e^{-k/8}$

$n \gg 1$ και $k \gg 1$: $P[\text{false positive}] \approx \left(1-e^{-kn/m}\right)^{k}$

Βέλτιστο k για ελαχιστοποίηση πιθανότητας false positive:

- $k = \frac{m}{n} \ln 2 \xrightarrow{m=8n} k \approx 5{,}55$, αλλά k = ακέραιος. Άρα $k=6$.
- $P[\text{false positive}] = \left(1-e^{-6 \cdot 1/8}\right)^{6} \approx 0{,}0216$

Άσκηση 6: Προθέματα Συμβολοσειρών

Για να αποθηκεύσουμε τις συμβολοσειρές του συνόλου S, θα χρησιμοποιήσουμε τη δομή δεδομένων Tries, η οποία εν γένει είναι μια δενδρική δομή.

Κατασκευή Trie:

Χρονική Πολυπλοκότητα: Κάθε φορά που διασχίζουμε μια συμβολοσειρά, εκτελούνται ενέργειες που για το κάθε γράμμα κοστίζουν σταθερό χρόνο. Άρα για να διασχίσουμε όλες τις συμβολοσειρές του S, θα μας κοστίσει χρονικά: (c λειτουργίες)×(N γράμματα) = $O(N)$

Χωρική Πολυπλοκότητα: Στην χειρότερη περίπτωση θα αποθηκευτούν όλα τα γράμματα των συμβολοσειρών του S σε διαφορετικά nodes, οπότε και θα χρειαστούν #nodes = #γραμμάτων : $O(N)$

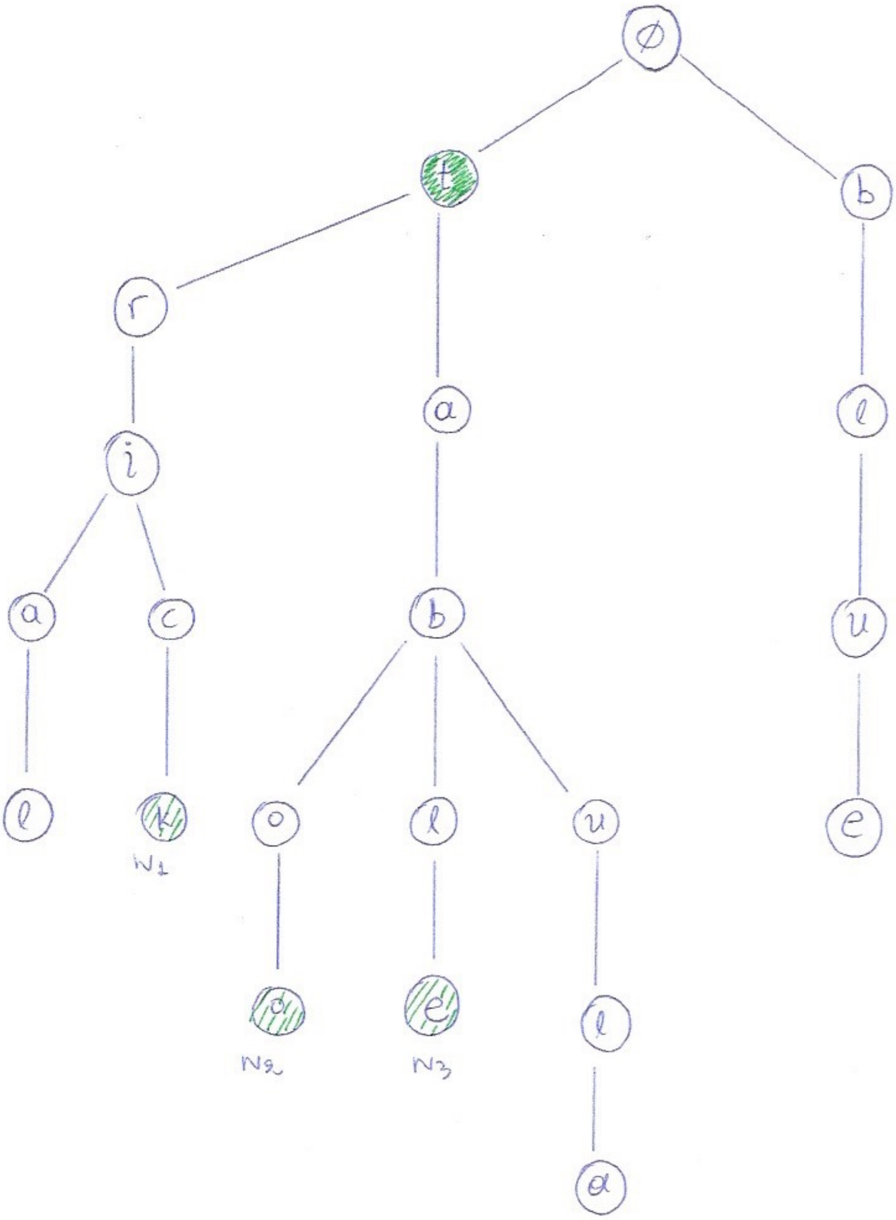
Έστω ένα υποσύνολο του S που περιέχει k συμβολοσειρές.

Query: Ποιό είναι το μέγιστο κοινό πρόθεμά τους;

Απάντηση: Το μέγιστο κοινό πρόθεμά τους είναι μια συμβολοσειρά που στο κατασκευασμένο Trie απεικονίζεται ως μονοπάτι με αρχή τη ρίζα του Trie και τέλος τον κόμβο w.
Έστω $w_1, w_2, \dots, w_k$ οι κόμβοι που περιέχουν το τελευταίο γράμμα από τις συμβολοσειρές $l_1, l_2, \dots, l_k$, αντίστοιχα.
Τότε ο w είναι ο Lowest Common Ancestor των $w_1, w_2, \dots, w_k$.

Παράδειγμα: $S = \{\text{table, blue, tabula, taboo, trial, trick}\}$

οι k συμβολοσειρές $(k=3)$



Το μονοπάτι $\emptyset \rightarrow t$ απεικονίζει τη συμβολοσειρά "t", που φανερά είναι ο lowest common ancestor των N_1, N_2, N_3 και το μέγιστο κοινό πρόθεμα των l_1, l_2, l_3.

- Το query που μας τίθεται μπορεί λοιπόν να απαντηθεί αν έχουμε γνωστό τον $LCA(w_1, w_2, \dots, w_k)$, σε χρόνο χειρότερης περίπτωσης: (χρόνος BFS για να εντοπίσουμε τον $LCA(w_1, w_2, \dots, w_k)$ αποθηκεύοντας παράλληλα το μονοπάτι = longest common prefix) $= O(2N-1) = O(N)$

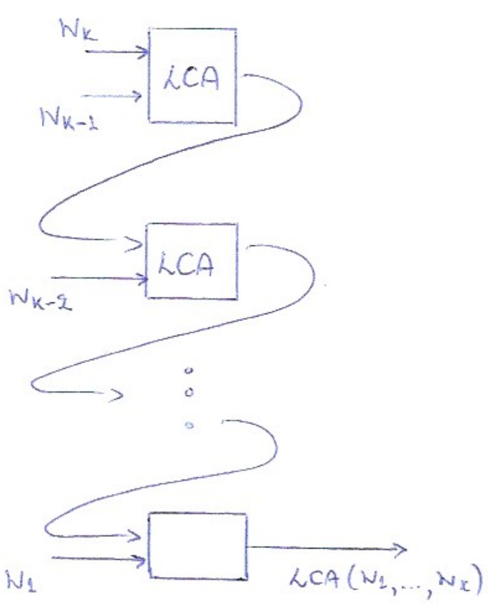
- Για να βρούμε τον $LCA(w_1, w_2, \dots, w_k)$ θα κάνουμε αναγωγή του LCA προβλήματος σε RMQ (Range Minimum Query) πρόβλημα. Η αναγωγή γίνεται σε γραμμικό χρόνο: $O(N)$. (με Euler Walk)

- Η προεπεξεργασία για τη δημιουργία Sparse Table, που θα χρησιμοποιεί το RMQ κοστίζει $O(N\log N)$.

- Κάθε query του RMQ απαντάται πλέον σε σταθερό χρόνο $O(1)$.

- Άρα για να βρούμε τον $LCA(w_1, w_2, \dots, w_k)$ κάνουμε:

$LCA(w_1, w_2, \dots, w_k) = LCA(w_1, LCA(w_2, LCA(w_3, \dots)))$, μέχρι το $LCA(w_{k-1}, w_k)$.



- Άρα για την απάντηση του $LCA(w_1, \dots, w_k)$ query, κόστος: $O(1)$ $(k-1)$ φορές
Δηλ. $O(k)$

Συγκεντρωτικά:

Αποθήκευση S: Δενδρική Δομή Trie

Query μέγιστου κοινού προθέματος:
- Αναγωγή LCA σε RMQ (μία φορά και ισχύει ∀επόμενο query)
- Βρίσκω LCA των k συμβολοσειρών με k-1 RMQ queries.
- Βρίσκω μονομιάς - ζητούμενη συμβολοσειρά με έναν BFS μέχρι τον LCA που βρήκα πριν.

Κατασκευή Trie: Χρονική πολ/τα: $O(N)$

Χωρική πολ/τα: $O(N)$

Χρονική πολυπλοκότητα για αναγωγή LCA → RMQ: $O(N)$

" " για κατασκευή sparse table του RMQ: $O(N\log N)$

Απάντηση query RMQ: $O(1)$

Απάντηση query μέγιστου κοινού προθέματος:

$$\underbrace{(k-1)\cdot O(1)}_{O(k)} + \underbrace{O(|V|+|E|)}_{BFS} = O(k) + O(2N-1) = O(k) + O(N)$$

$$= O(N)$$