ΑΛΓΟΡΙΘΜΟΙ & ΠΟΛΥΠΛΟΚΟΤΗΤΑ - 1η ΣΕΙΡΑ ΓΡΑΠΤΩΝ ΑΣΚΗΣΕΩΝ

Καραβαγγέλης Αθανάσιος
Α.Μ.: 03117022

ΑΣΚΗΣΗ 1α

- $\sqrt{n}^{\log\log(n!)} = \Theta\left(2^{\log(\sqrt{n})\cdot(\log n + \log\log n)}\right)$

Όμως $\log\sqrt{n}(\log n + \log\log n) \geq \log n$ και αφού

$\sqrt{n}^{\log\log(n!)} = 2^{\log(\sqrt{n})\cdot\log(\log(n!))}$ και

$\log\log(n!) = \Theta(\log n)$

Άρα $\sqrt{n}^{\log\log(n!)} = \underline{O}(2^{\log n}) = \underline{O}(n)$.

- $2^{(\log\log n)^4} = O(n)$ αφού

$m = \log n$

$\log m \leq \sqrt[4]{m} \Rightarrow$

$\log\log n \leq \sqrt[4]{\log n} \Rightarrow$

$(\log\log n)^4 \leq \log n \overset{2^n \nearrow}{\Longrightarrow}$

$2^{(\log\log n)^4} \leq 2^{\log n} \Longrightarrow 2^{(\log\log n)^4} = O(n)$

- $\log(n!)/(\log\log n)^5 = 2^{\log\left(\frac{\log(n!)}{(\log\log n)^5}\right)} = 2^{\log\log(n!) - \log(\log\log n)^5} =$

$= 2^{\log\log(n!) - 5\log\log\log n}$

Όμως $n\log n \leq \log(n!) \leq n\log n \Rightarrow$

$\log(n\log n) \leq \log(\log(n!)) \leq \log(n\log n) \Rightarrow$

$\log n + \log\log n \leq \log(\log(n!)) \leq \log n + \log\log n$

Άρα $\log(n!)/(\log\log n)^5 = \Theta\left(2^{\log n + \log\log n - 5\log\log\log n}\right) = \Theta(2^{\log n}) =$

$= \Theta(n)$

- $n \cdot 2^{2^{2^{100}}} = n \cdot 16^{100} = \Theta(n)$

- $n \cdot \sum_{k=0}^{n} \binom{n}{k} = \Theta(n \cdot 2^n)$, από Binomial Theorem: $\sum_{k=0}^{n} \binom{n}{k} = 2^n$

- $\log\left(\binom{2n}{n}\right) = \log\left(\frac{2n!}{n!n!}\right) = \log(2n!) - \log(n!) - \log(n!) =$

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

$$= \Theta\left(2n \log\left(\frac{2\cancel{n}}{\cancel{n}}\right)\right) = \Theta(2n) = \underline{\Theta(n)}$$

, αφού $\log(n!) = \Theta(n\log n)$.

- $\sum_{k=1}^{n} k \cdot 2^k = 2(2^n \cdot n - 2^n + 1) = \underline{\Theta(2^n \cdot n)}$

- $\sum_{k=1}^{n} k \cdot 2^{-k} = 2^{-n}(-n + 2^{n+1} - 2) = \underline{\Theta(1)}$

Τελικά, καταλήγουμε στη διάταξη:

$$\sum_{k=1}^{n} k \cdot 2^{-k} < 2^{(\log\log n)^4} < \frac{\log(n!)}{(\log\log n)^5} < \log\left(\binom{2n}{n}\right) = n \cdot 2^{2^{100}} <$$

$$\sqrt{n}^{\log\log(n!)} < n \cdot \sum_{k=0}^{n} \binom{n}{k} = \sum_{k=1}^{n} k \cdot 2^k$$

ΑΣΚΗΣΗ 1Β

1) $T(n) = 6T(\frac{n}{3}) + n^2\log n$

Λύση

Έχουμε $a=6$, $b=3$, $f(n) = n^2\log n$

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

Ασυμπτωτικά: Προφανώς $n^{1,63} < n^2 < n^2\log n$

Πολυωνυμικά: Για $\varepsilon = 0,3 > 0$ θα ισχύει:

$$f(n) \geq n^{1,63} \cdot n^{\varepsilon} = n^{1,63} \cdot n^{0,3} \Rightarrow f(n) = \Omega(n^{\log_3 6+\varepsilon})$$

Επομένως, η παραπάνω παράσταση εμπίπτει στην 3η περίπτωση του Master Theorem άρα $\boxed{T(n) = \Theta(f(n)) = \Theta(n^2\log n)}$

2) $T(n) = 9T(\frac{n}{3}) + n^2\log n$

Λύση

Έχουμε $a=9$, $b=3$, $f(n) = n^2\log n$

Άρα $n^{\log_b a} = n^{\log_3 9} = n^2$

Ασυμπτωτικά: Προφανώς για $n>2$: $n^2 < n^2\log n = f(n)$

Πολυωνυμικά: Παρατηρούμε πως δεν υπάρχει ε ώστε $f(n) = \Omega(n^{\log_3 9+\varepsilon})$, άρα η $f(n)$ δεν είναι πολυωνυμικά μεγαλύτερη της n^2 κι επομένως δε μπορούμε να κάνουμε χρήση του Master Theorem.

Για διευκόλυνσή μας θα θεωρήσουμε: $n = 3^m$ και $T(3^m) = S(m)$

Δηλαδή: $T(3^m) = 9T(3^{m-1}) + 3^{2m}\cdot\log 3^m \Rightarrow$

$$\Rightarrow S(m) = 9S(m-1) + 3^{2m}\cdot m\cdot\log 3 \Rightarrow$$

$$\rightarrow S(m) = 9^2S(m-2) + \frac{9}{9}\cdot 3^{2m}\cdot(m-1)\cdot\log 3 + 3^{2m}\cdot m\cdot\log 3 =$$

$$= 9^iS(m-i) + \cancel{\frac{9^i}{3^{2i}}}\cdot 3^{2m}(m-i)\cdot\log 3 + \ldots + 3^{2m}\cdot m\cdot\log 3 =$$

$$= 9^iS(m-i) + 3^{2m}\cdot\log 3\cdot\sum_{k=0}^{i-1}(m-k) =$$

$$\overset{i=m}{=} 9^mS(0) + 3^{2m}\cdot\log 3\cdot\sum_{k=1}^{m}k = 9^m\cdot S(0) + 3^{2m}\cdot\log 3\cdot\left(\frac{m^2}{2}+\frac{m}{2}\right) =$$

Αντίστροφα: $m = \log_3 n$

$$= n^2\cdot\Theta(1) + \frac{\log 3}{2}\cdot n^2\cdot(\log_3 n)^2 + \frac{\log 3}{2}\cdot n^2\cdot\log_3 n$$

Άρα $\boxed{T(n) = \Theta(n^2 \log n^2)}$ αφού αυτός είναι ο κυρίαρχος όρος.

3) $T(n) = 11T\left(\frac{n}{3}\right) + n^2 \log n$

Προς διευκόλυνσή μας θέτουμε $n = 3^m$ (1) και $T(3^m) = S(m)$ και έχουμε:

$$S(m) = 11 \cdot S(m-1) + 3^{2m} \cdot \log(3^m) \stackrel{S(m-1)=11S(m-2)+\frac{3^{2m}}{3^2}\log 3^{m-1}}{=\!=\!=}$$

$$= 11^2 \cdot S(m-2) + \frac{11}{9} \cdot 3^{2m} \cdot \log 3 \cdot (m-1) + 3^{2m} \log 3 \cdot m =$$

$$= 11^i \cdot S(m-i) + \frac{11^{i-1}}{9^i} \cdot 3^{2m} \cdot \log 3 (m-i+1) + \ldots + 3^{2m} m \cdot \log 3 =$$

$$= 11^i \cdot S(m-i) + 3^{2m} \cdot \log 3 \cdot \frac{1}{11} \sum_{k=0}^{i-1} (m-k) \left(\frac{11}{9}\right)^{k+1} =$$

$$\stackrel{i=m}{=} 11^m S(0) + 3^{2m} \cdot \log 3 \cdot \frac{1}{11} \cdot \sum_{k=0}^{m-1} (m-k) \left(\frac{11}{9}\right)^{k+1} =$$

$$= 11^m \cdot \Theta(1) + 3^{2m} \cdot \log 3 \cdot \frac{1}{\cancel{11}} \cdot \frac{\cancel{11}}{4} \left[11\left(\left(\frac{11}{9}\right)^m - 1\right) - 2m\right] =$$

$$= 11^m \cdot \Theta(1) + 3^{2m} \cdot \frac{\log 3 \cdot 11}{4} \cdot \left(\frac{11}{9}\right)^m - 3^{2m} \cdot \frac{\log 3 \cdot 11}{4} - 3^{2m} \cdot 2m \cdot \frac{\log 3}{4} =$$

$$\stackrel{(1) \Rightarrow m = \log_3 n}{=\!=\!=} 11^{\log_3 n} \Theta(1) + n^2 \cdot \frac{11}{4} \log 3 \cdot \left(\frac{11}{9}\right)^{\log_3 n} - n^2 \cdot \frac{11 \log 3}{4} - n^2 \cdot \frac{\log 3 \cdot \log_3 n}{2} =$$

$$= 11^{\log_3 n} \Theta(1) + \frac{11}{4} \log 3 \cdot n^{\frac{\log_2 11}{\log_2 3}} - \ldots =$$

$$\boxed{= \Theta\left(n^{\log_2 11 / \log_2 3}\right)}$$

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

Η παραπάνω αναδρομική σχέση είναι της μορφής:

$T(n) = T(\gamma_1 \cdot n) + T(\gamma_2 \cdot n) + \Theta(n)$ με $\gamma_1 = \frac{1}{4}$, $\gamma_2 = \frac{1}{2}$.

Παρατηρώ πως $\underline{\gamma_1 + \gamma_2 \leq 1 - \epsilon}$ $\forall \epsilon \in (0, 0{,}25)$ άρα η παραπάνω εμπίπτει στις ειδικές μορφές του Master Theorem και συγκεκριμένα στη 3η περίπτωση επομένως:

$\boxed{T(n) = \Theta(n)}$

5) $T(n) = 2T\left(\frac{n}{4}\right) + T\left(\frac{n}{2}\right) + n$

·Όπως και η προηγούμενη, η παραπάνω είναι της μορφής:
$T(n) = T(\gamma_1 \cdot n) + T(\gamma_2 \cdot n) + T(\gamma_3 \cdot n) + \Theta(n)$ με $\gamma_1 = \gamma_2 = \frac{1}{4}$, $\gamma_3 = \frac{1}{2}$

αφού $T(n) = T\left(\frac{n}{4}\right) + T\left(\frac{n}{4}\right) + T\left(\frac{n}{2}\right) + n$.

Ισχύει $\gamma_1 + \gamma_2 + \gamma_3 = 1$, άρα η παραπάνω εμπίπτει στη 2η περίπτωση των ειδικών μορφών του Master Theorem, όπου $\boxed{T(n) = \Theta(n \log n)}$

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

Θα προσπαθήσουμε να φέρουμε την παραπάνω αναδρομική σχέση σε μία μορφή που θα εμπίπτει σε μία εκ των περιπτώσεων του Master Theorem.

Θέτω $n = \left(\frac{2}{3}\right)^m$ και έχω:

$$T\left(\frac{2}{3}^m\right) = T\left(\left(\frac{2}{3}\right)^{\frac{2m}{3}}\right) + \Theta\left(\log\left(\frac{2}{3}\right)^m\right)$$

Θέτω ακόμα $T\left(\left(\frac{2}{3}\right)^m\right) = S(m)$ και έχω:

$$S(m) = S\left(\frac{2m}{3}\right) + \Theta\left(\log\left(\frac{2}{3}\right)^m\right)$$

Παρατηρώ πως η παραπάνω έχει έρθει πλέον σε μορφή που μπορώ να εφαρμόσω Master Theorem.

Έχουμε $a = 1$, $b = \frac{3}{2}$, $f(m) = \Theta\left(\log\left(\frac{2}{3}\right)^m\right)$

• $n^{\log_b a} = n^{\log_{\frac{3}{2}} 1} = n^{\log_{1,5} 1} = n^{\frac{\log_2 1}{\log_2(1,5)}} = n^0 = 1$

• $f(m) = \Theta\left(\log\left(\frac{2}{3}\right)^m\right) = \Theta(m \cdot (\log 2 - \log 3)) = \Theta(m)$

Άρα $f(m) = \Theta(m) \geq 1$ και ασυμπτωτικά και πολυωνυμικά.

Άρα $S(m) = \Theta(m) \Rightarrow T\left(\frac{2}{3}^m\right) = \Theta(m) \Rightarrow$

$\Rightarrow \boxed{T(n) = \Theta(\log_{\frac{2}{3}} n)}$ ∎

7) $T(n) = T(n/3) + \sqrt{n}$

Λύση

$a = 1$, $b = 3$, $f(n) = \sqrt{n} = n^{1/2} = n^{0,5}$

$$\left.\begin{array}{l} \bullet\ n^{\log_b a} = n^{\log_3 1} = n^0 = 1 \\ \bullet\ f(n) = n^{1/2} \end{array}\right\} \frac{f(n)}{n^{\log_b a}} = n^{1/2}$$

Επομένως η $f(n)$ είναι τόσο ασυμπτωτικά όσο και πολυωνυμικά μεγαλύτερη της $g(n) = 1$.
Άρα η παραπάνω αναδρομική σχέση εμπίπτει στη $3^{η}$ περίπτωση του Master Theorem, όπου δηλαδή $f(n) = \underline{\Omega}(n^{\log_b a + \epsilon})$, $\epsilon > 0$ και άρα:

$$\boxed{T(n) = \Theta(\sqrt{n})}$$

Scanned with CamScanner

ASKHSH 2

Προθεματική ταξινόμηση

(α) Αρχικά, για έναν πίνακα με 1 μόνο στοιχείο δε χρειάζεται καμία προθεματική περιστροφή άρα $T(1) = 0$. Εάν έχουμε έναν πίνακα 2 στοιχείων ($n=2$) τότε τα στοιχεία απαιτούν το πολύ μία προθεματική περιστροφή δηλαδή μπορούμε να πούμε ότι $T(2) \leq 1$.

Αλλιώς, για περιπτώσεις τριών και άνω στοιχείων ($n \geq 3$) επαναλαμβάνουμε την παρακάτω διαδικασία:

- Εντοπίζουμε το μεγαλύτερο στοιχείο, κάνουμε μία προθεματική περιστροφή και αυτό έρχεται στην πρώτη θέση του πίνακα. Τότε έστω $A[k]$ το μεγαλύτερο στοιχείο:

$$[\underline{A[1], A[2], \dots, A[k]}, A[k+1], \dots] \rightarrow [\underline{A[k], \dots, A[2], A[1]}, A[k+1], \dots]$$

- Έπειτα με άλλη μία προθεματική περιστροφή το τοποθετούμε στην τελευταία θέση του πίνακα ως εξής:

$$[\underline{A[k], \dots, A[2], A[1], A[k+1], \dots A[k+i]}] \rightarrow [\underline{A[k+i], \dots, A[k+1], A[1], A[2], \dots, A[k]}]$$

- Επαναλαμβάνουμε την ίδια διαδικασία για τα υπόλοιπα $n-1$ στοιχεία

Επομένως για $n \geq 3$ θα ισχύει: $T(n) \leq T(n+1) + 2$.

Για τα n στοιχεία θα σκεφτόμουν εξής. Χρειάζομαι 2 περιστροφές για την εκτέλεση του αλγορίθμου μέχρι να φτάσω στα 2 στοιχεία όπου χρειάζομαι το πολύ μία δηλαδή:

$$T(n) \leq 2(n-2) + T(2) \xrightarrow{T(2) \leq 1} T(n) \leq 2(n-2) + 1 \Rightarrow$$

$$\Rightarrow T(n) \leq 2n - 4 + 1 \Rightarrow \boxed{T(n) \leq 2n-3 \quad , n \geq 2}$$

Καταλήγουμε ότι ο αλγόριθμός μας χρησιμοποιεί το πολύ

2n-3 προθεματικές περιστροφές.

β) Θεωρούμε ότι αρχικά όλα τα στοιχεία του μη-ταξινομημένου πίνακα είναι θετικά. Για $n \leq 2$, πραγματοποιώντας προθεματικές περιστροφές έχουμε τις εξής περιπτώσεις: $[1, 2]$: όπου απαιτούνται 0 κινήσεις

και $[2, 1] \rightarrow [-2, 1] \rightarrow [-1, 2] \rightarrow [1, 2]$: όπου απαιτούνται 3 κινήσεις.

Γενικά, για $n > 2$ επαναλαμβάνουμε διαδοχικά την παρακάτω διαδικασία:

- Στη χειρότερη περίπτωση κάθε φορά, θα πρέπει να εντοπίζουμε το μεγαλύτερο κατ' απόλυτη τιμή στοιχείο και με μία περιστροφή να το φέρνουμε στην κορυφή του πίνακα, δηλαδή στη πρώτη θέση:

$$[\underline{A[1], A[2], \ldots, A[k]}, \ldots A[k+i]] \rightarrow [\underline{-A[k], \ldots -A[2], -A[1]}, \ldots A[k+i]]$$

- Έπειτα αν αυτό το στοιχείο έχει θετική τιμή, θα πρέπει να εκτελέσουμε μία περιστροφή μόνο για το στοιχείο της πρώτης θέσης:

$$[\underline{A[k]}, \ldots, -A[2], -A[1], \ldots, A[k+i]] \rightarrow [\underline{-A[k]}, \ldots, -A[2], -A[1], \ldots, A[k+i]]$$

- Τέλος, εκτελούμε μια περιστροφή σε όλα τα στοιχεία του πίνακα:

$$[\underline{-A[k] \ldots, -A[2], -A[1], \ldots, A[k+i]}] \rightarrow [\underline{-A[k+i], \ldots, A[1], A[2], \ldots, A[k]}]$$

2γ1) Θα δείξουμε ότι για κάθε ενδιάμεσο πίνακα Α± που είναι διαφορετικός από τον [-1, -2, , n] μπορούμε να κατασκευάσουμε ένα νέο συμβατό ζεύγος έπειτα από 2 το πολύ προσημασμένες περιστροφές παίρνωντας όλες τις δυνατές περιπτώσεις.

■ 1η περίπτωση-

Υπάρχει τουλάχιστον ένας θετικός αριθμός στον πίνακα και ο μέγιστος θετικός έστω m είναι $m<n$.

Τότε:

> Είτε ο $-(m+1)$ είναι μετά τον m και οπότε έχω τις εξής διαδοχικές κινήσεις:

Αρχικά → $[\ldots m \ldots -(m+1) \ldots]$

Κίνηση 1 → $[(m+1) \ldots -m \ldots\ldots]$ (Κάνω μία προθεματική περιστροφή επιλέγοντας k στοιχεία όπου το <u>k = θέση(-(m+1))</u>)

Κίνηση 2 → $[\ldots \boxed{-(m+1)\ -m} \ldots\ldots]$ (Κάνω μια προσημασμένη προθ. περιστροφή επιλέγοντας k στοιχεία όπου <u>k = θέση(-m) - 1</u>)

> Είτε ο $-(m+1)$ είναι πρίν τον m και οπότε έχω τις εξής διαδοχικές κινήσεις:

Αρχικά → $[\ldots -(m+1) \ldots m \ldots]$

Κίνηση 1 → $[-m \ldots m+1 \ldots\ldots]$ (Προσημασμένη προθ. περιστροφή για k στοιχεία όπου το <u>k = θέση(m)</u>)

Κίνηση 2 → $[\ldots \boxed{m\ \ m+1} \ldots\ldots]$ (Προσημασμένη προθεματική περιστροφή για <u>k = θέση(m+1) - 1</u> στοιχεία)

■ 2η περίπτωση-

Υπάρχει τουλάχιστον ένας θετικός αριθμός κι ο μέγιστος θετικός έστω m είναι $m=n$. Από εκφώνηση, αρκεί να μεταφέρουμε το μέγιστο αριθμό με θετικό πρόσημο στην τελευταία θέση για να δημιουργηθεί συμβατό ζεύγος:

Αρχικά → $[\ldots\ m \ldots]$

Κίνηση 1 → $[-m \ldots\ldots]$ (Εκτελώ προσημασμένη προθεματική περιστροφή για k = θέση(m) στοιχεία)

Κίνηση 2 → $[\ldots\ \ldots\ \boxed{m}]$ (Εκτελώ προσημασμένη προθεματική περιστροφή και για τα n στοιχεία του πίνακα)

Επομένως, συνολικά εκτελούμε 3 περιστροφές κάθε φορά που επαναλαμβάνουμε τη διαδικασία (στη χειρότερη περίπτωση). Ωστόσο, πρέπει να παρατηρήσουμε ότι στην πρώτη επανάληψη κάθε φορά απαιτούνται μόνο 2 περιστροφές εφόσον το στοιχείο με τη μεγαλύτερη απόλυτη τιμή είναι θετικό. Άρα συνολικά για n στοιχεία μπορούμε να διατυπώσουμε το εξής:

$$T(n) \leq 2 + 3(n-3) + T(2)$$

- Όταν απομένουν 2 στοιχεία έχουμε τις εξής περιπτώσεις:

$[1, 2, \dots]$: 0 κινήσεις

$[-1, 2, \dots] \to [1, 2, \dots]$: 1 κίνηση

$[2, 1, \dots] \to [-2, 1, \dots] \to [-1, 2, \dots] \to [1, 2, \dots]$: 3 κινήσεις

$[-2, -1, \dots] \to [1, 2, \dots]$: 1 κίνηση

$[-2, 1, \dots] \to [-1, 2, \dots] \to [1, 2, \dots]$: 2 κινήσεις

$[-1, -2] \to [2, 1, \dots] \to [-2, 1, \dots] \to [-1, 2, \dots] \to [1, 2]$: 4 κινήσεις

$[1, -2, \dots] \to [2, -1] \to [-2, -1, \dots] \to [1, 2, \dots]$: 3 κινήσεις

$[2, -1, \dots] \to [-2, -1, \dots] \to [1, 2, \dots]$: 2 κινήσεις

άρα 4 ή λιγότερες κινήσεις $\Rightarrow T(2) \leq 4$.

Επομένως: $\boxed{T(n) \leq 2 + 3(n-3) + 4 = 3n-3}$, $n > 2$

Scanned with CamScanner

■ $3^{η}$ περίπτωση-

Έχουμε μόνον αρνητικούς αριθμούς τώρα και για κάποιον αριθμό $-m$ έχουμε το $-(m+1)$ πιο πριν στον πίνακα.

Τότε:

Αρχικά → $[\ \dots\ -(m+1)\ \dots\ -m\ \dots\ ]$

Κίνηση 1 → $[(m+1)\ \dots\ \dots\ -m\ \dots\ ]$ (Προσημασμένη προθ. περιστροφή για $k =$ θέση$(-m+1)$ στοιχεία)

Κίνηση 2 → $[\ \dots\ \dots\ -(m+1)\ -m\ \dots\ ]$ (Προσημασμένη προθ. περιστροφή για $k =$ θέση$(-m) - 1$ στοιχεία)

Άρα έδειξα ότι σε κάθε περίπτωση εκτός της $A_t = [-1, -2, -3, \dots, -n]$ μπορώ με το πολύ 2 περιστροφές να δημιουργήσω ένα σύμβατο ζεύγος.

2γ2) Για την ταξινόμηση του πίνακα, αρκεί η δημιουργία n συμβατών ζευγών, αφού με τη δημιουργία ενός συμβατού ζεύγους δημιουργώ μία ταξινομημένη δυάδα που δεν απαιτεί άλλη επεξεργασία.

- Θα δείξω ότι ο $A_t = [-1, -2, -3, \dots, -n]$ ταξινομείται σε $2n$ περιστροφές:

<u>Επαγωγικά</u>:

> Θα δείξω ότι μετά από $2j$ περιστροφές όπου $j \leq n$ ο πίνακας θα είναι:

$$[-(j+1)\ -(j+2)\ \dots\ -n\ \ 1\ \ 2\ \dots\ j-1\ \ j]$$

> Βάση της επαγωγής για $i=0$ έχουμε

$$[\ -1\ \ -2\ \dots\ \dots\ \ -n]$$

> Έστω ότι ισχύει για i δηλαδή ότι μετά από $2i$ περιστροφές έχω τον:

$$[-(i+1)\ -(i+2)\ \dots\ -n\ \ 1\ \ 2\ \dots\ i-1\ \ i]$$

> Κάνουμε διαδοχικά 2 περιστροφές, πρώτα και για τα n στοιχεία κι έπειτα για τα $(n-1)$ στοιχεία και λαμβάνω:

$$[-i\ \ -(i-1)\ \dots\ -2\ \ -1\ \ n\ \dots\ i+2\ \ i+1]$$

$$[-(i+2)\ \dots\ -n\ \ 1\ \ 2\ \dots\ (i-1)\ \ i\ \ i+1]$$

> Παρατηρώ, λοιπόν, ότι ο τελευταίος πίνακας για $j=i+1$ είναι ο ίδιος με αυτόν στον οποίο ήθελα να καταλήξω. Σε αυτόν κατέληξα μετά από $2i+2=2j$ περιστροφές άρα αποδείξαμε το ζητούμενο.

> Άρα ξεκινάω από πίνακα n στοιχείων. Κάνω κάθε φορά 2 περιστροφές και δημιουργώντας συμβατά ζεύγη μειώνω ουσιαστικά κατά 1 το πλήθος των στοιχείων εισόδου συνεχώς. Συνολικά λοιπόν θα απαιτούνται $2i+2(n-i)$ περιστροφές αν θεωρήσουμε ότι μετά από $2i$ περιστροφές δημιουργώ τον πίνακα $[-1,-2,-3,\dots,-n]$. Επομένως $2i+2(n-i)=2i$ περιστροφές το πολύ.

ASKHSH 3

Παίζοντας χαρτιά

1. Έστω το σύνολο $S = \{c_1, c_2, \dots, c_n\}$ που απεικονίζει τις κάρτες και $P = \{p_1, p_2, \dots, p_k\}$ που απεικονίζει το σύνολο των k στοιβών που θα δημιουργηθούν ως εξής:

- Αρχικά, δημιουργούμε μία καινούρια στοίβα p_1 με τη κάρτα στη κορυφή που είναι η c_1.
- Έπειτα, για τις υπόλοιπες κάρτες $j = 2, \dots, n$ και έστω $c_1, \dots, c_{j-1}$ οι κάρτες που έχουν δημιουργήσει τις στοίβες $p_1, \dots, p_k$, λειτουργούμε ως εξής:

$$p_1 = \begin{cases} c_{1p_1} \\ \vdots \\ c_{x_1 p_1} \end{cases}, \quad p_2 = \begin{cases} c_{1p_2} \\ \vdots \\ c_{x_2 p_2} \end{cases}, \quad \dots, \quad p_k = \begin{cases} c_{1p_k} \\ \vdots \\ c_{x_k p_k} \end{cases}$$

→ Αν η $c_j > c_{1p_i}$ για κάθε $i = 1, \dots, k$ τότε δημιουργούμε μία νέα στοίβα p_{k+1} με την c_j στην κορυφή.

→ Αλλιώς τοποθετούμε τη c_j στη στοίβα p_m όπου:

$$p_m = \begin{cases} c_j \\ c_{1p_m} \\ \vdots \\ c_{x_m p_m} \end{cases} \quad \text{όπου } c_{1p_m} = \min\{c_{1p_1}, c_{1p_2}, \dots, c_{1p_k}\}.$$

(1) Με άλλα λόγια, τοποθετώ κάθε φορά ένα φύλλο στην κορυφή μίας στοίβας όπου το κορυφαίο της φύλλο ήταν το ελάχιστο από τα φύλλα που είναι μεγαλύτερα από αυτό που θέλω να τοποθετήσω.

(2) Ο παραπάνω αλγόριθμος είναι ένα είδος "άπληστου αλγορίθμου" για το παραπάνω πρόβλημα, αφού δημιουργεί τον ελάχιστο αριθμό στοιβών.

Απόδειξη

Ας υποθέσουμε ότι σ' ένα σημείο του "παιχνιδιού", παίζοντας με βάση τον "άπληστο αλγόριθμο" που διατυπώσαμε παραπάνω, η κάρτα c θα τοποθετηθεί στη στοίβα i. Ας υποθέσουμε ακόμα ότι υπάρχει ένας άλλος αλγόριθμος βάσει του οποίου θα τοποθετούσαμε την c στη στοίβα j. Έστω B ο αλγόριθμος (στρατηγική) αυτός. Θα δείξουμε ότι η c πρέπει να τοποθετηθεί στη στοίβα i "κλέβοντας" τη στρατηγική B.

Έστω c_i, c_j οι κάρτες στην κορυφή των i, j αντίστοιχα. Αφού ο άπληστος αλγόριθμος τοποθετεί την c στην κορυφή της στοίβας με τη μικρότερη κάρτα, θα είναι: $c < c_i \leq c_j$. Άρα εάν τοποθετούσαμε την c στη στοίβα j τότε από εκείνη τη στιγμή κι έπειτα, κάθε κάρτα που θα μπορούσε να παιχτεί στην κορυφή της νέας στοίβας j που θα προκύψει, θα μπορούσε να παιχτεί στην κορυφή της i ($c < c_i$). Έτσι, μπορούμε να δημιουργήσουμε έναν νέο αλγόριθμο - στρατηγική Γ ο οποίος μιμείται ακριβώς τον B απλά με τους ρόλους των στοιβών i και j ανεστραμμένους από τη στιγμή που θα βάλουμε την c στην j. Έτσι, βέλτιστο είναι να τοποθετήσω την c στη στοίβα i.

Το παραπάνω, αν εφαρμοστεί σε κάθε κάρτα μας δίνει ως αποτέλεσμα ότι κανένας άλλος αλγόριθμος δεν επιτυγχάνει ελάχιστο αριθμό στοιβών.

- **Πολυπλοκότητα**

Εάν κάνουμε γραμμική αναζήτηση στις στοίβες για να βρούμε εκείνη με το μικρότερο σε τιμή κορυφαίο στοιχείο έχουμε στη χειρότερη περίπτωση να αναζητήσουμε σε n στοίβες. Άρα για τα n φύλλα η πολυπλοκότητα θα είναι της τάξης $O(n)$.

Ωστόσο, θα μπορούσαμε αντί για το παραπάνω να εκτελούμε δυαδική αναζήτηση για την εύρεση του ελάχιστου "κεφαλιού" στοίβας που μπορεί να τοποθετηθεί η κάρτα μας, η οποία στη χειρότερη περίπτωση απαιτεί $\log n$ συγκρίσεις για 1 αναζήτηση. Άρα για n φύλλα, το worst-case scenario θα είναι: $\boxed{O(n \log n)}$

2. Έχοντας χρησιμοποιήσει τον αλγόριθμο που διατυπώσαμε στο (1) έχουμε δημιουργήσει για n φύλλα k φθίνουσες στοίβες από την "κορυφή" προς τον "πάτο" τους.

Θα ταξινομήσουμε την τράπουλα χρησιμοποιώντας έναν σωρό ελαχίστων και τις k στοίβες που έχουν προκύψει από μία τυχαία μετάθεση των n φύλλων.

Θα κατασκευάσουμε το σωρό ελαχίστων ως εξής:

- Αρχικά, θα τοποθετήσουμε σε αυτόν τα k "heads" των στοιβών. Σε κάθε κόμβο που θα απεικονίζει ένα "head" θα αποθηκεύουμε επίσης μία μεταβλητή p η οποία θα δείχνει από ποια στοίβα προήλθε το στοιχείο.
- Κάθε φορά που θα "βγάζουμε" ένα στοιχείο από το σωρό θα τοποθετούμε με τη βοήθεια της μεταβλητής p το στοιχείο από τη στοίβα που βρίσκεται εκείνη τη στιγμή στην κορυφή έως ότου τελειώσουν όλα τα στοιχεία - φύλλα.

Έτσι, λοιπόν ο αλγόριθμος μας έχει ως εξής:

- Αρχικά, τοποθετούμε - εισάγουμε όλα τα heads στο σωρό.
- Εξάγουμε από το σωρό το ελάχιστο στοιχείο και ακολούθως συμπληρώνουμε το σωρό με το επόμενο στοιχείο της στοίβας που μας έδειξε η μεταβλητή p.
- Εκτελούμε το παραπάνω μέχρι να τελειώσουν τα n φύλλα και να αδειάσει ο σωρός.

Για τον παραπάνω θα χρειαστούμε ένα σωρό k-θέσεων. Για κάθε ένα από τα n φύλλα θα απαιτηθεί 1 εισαγωγή και 1 διαγραφή από το σωρό. Κάθε εισαγωγή και διαγραφή κοστίζει logk χρονικά. Επομένως για n εισαγωγές & διαγραφές στη χειρότερη περίπτωση θα απαιτηθεί πολυπλοκότητα: $O(2 \cdot n \cdot \log k)$ δηλαδή στη χειρότερη περίπτωση όπου η είσοδος του αλγορίθμου του (1) είναι αύξουσα ακολουθία θα απαιτηθεί πολυπλοκότητα: $\boxed{O(2n\log n) = O(n\log n)}$

3.

- Είσοδος : 3, 2, 4, 7, 8, 1, 5, 6

Εκτελώντας τον αλγόριθμο του (1) θα προκύψουν οι εξής στοίβες:

- Εκτέλεση 0:

0
3

Το πρώτο φύλλο δημιουργεί μία νέα στοίβα.

- -//- 1:

0
2
3

Το 2 < 3 μπορεί να τοποθετηθεί στην ήδη υπάρχουσα στοίβα

- -//- 2:

0	1
2	
3	4

Το 4 είναι μεγαλύτερο του 2 άρα δημιουργεί νέα στοίβα.

- -//- 3:

0	1	2
2		
3	4	7

Το 7 > 4 και 7 > 2 άρα δε μπορεί να τοποθετηθεί σε υπάρχουσα στοίβα και δημιουργεί νέα.

- -//- 4:

0	1	2	3
2			
3	4	7	8

Ισχύει 8 > 7 > 4 > 2 άρα το 8 δημιουργεί επίσης μία νέα στοίβα.

- -//- 5:

0	1	2	3
1			
2			
3	4	7	8

Ισχύει 1 < 2 < 4 < 7 < 8, επομένως το 1 βάσει του αλγορίθμου τοποθετείται επάνω από το ελάχιστο φύλλο που μπορεί και αυτό είναι το 2.

- -//- 6:

0	1	2	3
1			
2		5	
3	4	7	8

Το 5 μπορεί να τοποθετηθεί είτε επάνω από το 7 είτε από το 8. Ωστόσο 7 = min{7,8}.

- -//- 7:

0	1	2	3
1			
2		5	6
3	4	7	8

Η μόνη επιλογή για το 6 είναι η στοίβα του 8 αφού 6 > 5 > 4 > 1 κι έτσι τοποθετείται εκεί.

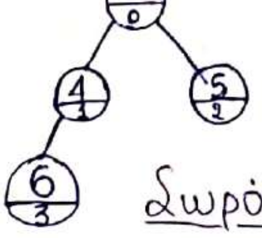
Επομένως, παρατηρώ πως προκύπτουν 4 στοίβες.

Για την ταξινόμηση λειτουργούμε ως εξής:

- Εκτέλεση 0: Τοποθετώ τα "heads" των στοιβών στο σωρό δημιουργώντας κόμβους με την τιμή του φύλλου και τη στοίβα από την οποία προέρχεται:

0	1	2	3
2			
3		7	8

Στοίβες



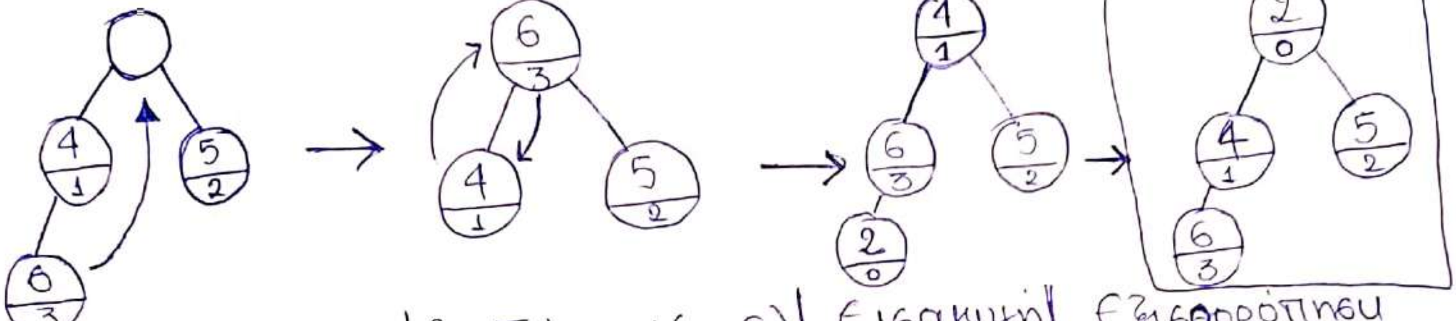
Σωρός

[ ]

Output

Εκτέλεση 1:

• Εξάγω το ελάχιστο στοιχείο (1) από το σωρό και προσθέτω-εισάγω το επόμενο της στοίβας 0.



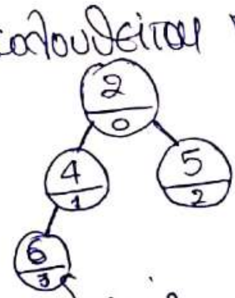
Το δεξιότερο στοιχείο του χαμηλότερου επιπέδου αντικαθιστά το 1.	Επειδή το 4(παιδί) μικρότερο του 6 (γονιός), αυτά αλλάζουν θέσεις.	Εισαγωγή του 2.	Εξισορρόπηση του σωρού μετά τη νέα εισαγωγή.

(Η παραπάνω διαδικασία ακολουθείται παρομοίως σε κάθε νέα εκτέλεση)

Άρα:

5	_	7	8
0	1	2	3

Στοίβες



Σωρός

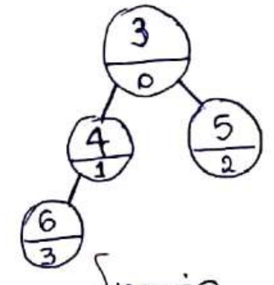
[1]

Output

• Εκτέλεση 2:

7	8
2	3

Στοίβες



Σωρός

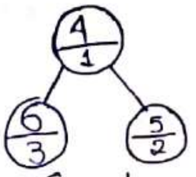
[1, 2]

Output

• Εκτέλεση 3:

Στοίβες:

7	8
2	3



Σωρός

[1, 2, 3]

Output

•
•
•

• Εκτέλεση 7: Κενές στοίβες & σωρός.
Output: [1, 2, 3, 4, 5, 6, 7, 8]

• Η μέγιστη αύξουσα υπακολουθία της εισόδου είναι η: 2,4,7,8 και έχει μήκος 4.

4. Έστω μία μέγιστη αύξουσα υπακολουθία μήκους N. Με βάση τους κανόνες του παιχνιδιού κάθε ένα από τα N φύλλα δε μπορεί να τοποθετηθεί επάνω σε κάποιο από τα άλλα N-1. Οπότε, είναι δεδομένο ότι είτε θα τοποθετηθούν σε διαφορετικές υπάρχουσες στοίβες, είτε θα δημιουργήσουν νέες στοίβες, είτε ένα συνδυασμό των δύο πάντοτε όμως χωρίς 2 φύλλα της υπακολουθίας μήκους N να καταλήγουν στην ίδια στοίβα.

Έστω M υπάρχουσες στοίβες όταν "ανοίγω" το 1ο φύλλο της μέγιστης αύξουσας υπακολουθίας.

• Αν $M<N$:

Τότε αφού για τα N φύλλα απαιτούνται N στοίβες, θα δημιουργηθούν ακόμη N−M στοίβες μέχρι να τελειώσει η τράπουλα, άρα τελικά θα έχω $k \geq N$ στοίβες.

• Αν $M=N$:

Τότε έχουμε ήδη N στοίβες.

• Αν $M>N$:

Πρακτικά, αυτό δε γίνεται να συμβεί διότι θα σήμαινε ότι υπάρχει μία αύξουσα υπακολουθία μήκους M που θα απαιτούσε την ύπαρξη M στοιβών. Όμως αυτό δεν ισχύει αφού $M>N$ και η μήκους N υπακολουθία είναι αυτή που είναι η μέγιστη αύξουσα.

Σε κάθε περίπτωση οι στοίβες θα είναι $k \geq N$.

Scanned with CamScanner

5. >Το κορυφαίο στοιχείο της k-οστής στοίβας μας δείχνει ότι υπάρχει μία αύξουσα υπακολουθία μήκους k που καταλήγει στο στοιχείο αυτό.

Αυτό μπορεί να γίνει κατανοητό με την εξής υλοποίηση. Κάθε φορά που τοποθετώ ένα στοιχείο σε μία στοίβα (εκτός της 1ης) χρησιμοποιώντας τον αλγόριθμο του (1) θα τοποθετώ ένα δείκτη από αυτό προς το κορυφαία στοιχείο της αριστεράς στοίβας. Το στοιχείο αυτό θα είναι σίγουρα μικρότερο αφού σε διαφορετική περίπτωση, με βάση τον αλγόριθμο του (1) θα τοποθετούσαμε το νέο στοιχείο σε εκείνη τη στοίβα κι όχι στη δεξιά της.

Για παράδειγμα, η είσοδος του (3) αναπαρίσταται όπως παρακάτω: ←3,2,4,7,8,1,5,6

```
1
2 ↖ ↙5←6
3  4←7←8
```

• Παρατηρώ πως το κορυφαίο στοιχείο κάθε k στοίβας συμμετέχει σε μία αύξουσα υπακολουθία μήκους k με τα στοιχεία που επιδεικνύουν οι δείκτες ακολουθώντας τους από δεξιά προς τ' αριστερά:

k=1 : [1]   k=3: [2,4,5]
k=2 : [2,4]   k=4: [2,4,5,6]

• Βλέπουμε, πως αν έπειτα από την εκτέλεση του (1) έχουμε k στοίβες και καλέσουμε το κορυφαίο στοιχείο της k-οστής στοίβας a_k τότε αν ακολουθήσουμε τους δείκτες προς την k-1, μετά την k-2 κ.τ.λ θα λάβουμε τη μέγιστη αύξουσα υπακολουθία S. Άρα συμπεραίνουμε ότι θα έχουμε το πολύ length(S) = k στοίβες. Έτσι, ο αλγόριθμος του (1) με τον τρόπο αυτό λύνει το πρόβλημα της μέγιστης αύξουσας υπακολουθίας. ■

> Σχετικά με το μήκος της μέγιστης φθίνουσας υπακολουθίας παρατηρώ ότι ταυτίζεται με το μήκος της μεγαλύτερης στοίβας σε στοιχεία.

6. Ας υποθέσουμε τα εξής:

Έστω $l(a_i)$: το μήκος της μεγαλύτερης αύξουσας υπακολουθίας που ξεκινά με το a_i

και $d(a_i)$: το μήκος της μεγαλύτερης φθίνουσας υπακολουθίας που ξεκινά με το a_i

Αντιστοιχίζω, κάθε $a_1, a_2, \dots, a_{n \cdot m+1}$ με το αντίστοιχο ζεύγος $(l(a_i), d(a_i))$, $i = 1, 2, \dots, n \cdot m+1$ και παρατηρώ πως όλα τα ζεύγη είναι διαφορετικά. Για παράδειγμα, για $m = 2, n = 3, m \cdot n + 1 = 7$

ai	3	2	4	7	1	5	6
l	4	4	3	1	3	2	1
d	3	2	2	2	1	1	1

• Στο σημείο αυτό, μπορούμε να συνδέσουμε την απόδειξη αυτή με το γνωστό θεώρημα της Αρχής του Περιστεριώνα από τα Διακριτά Μαθηματικά. Στην περίπτωσή μας μπορούμε να χρησιμοποιήσουμε ως "περιστέρια" τα a_i φύλλα και ως "τρύπες" τα ζεύγη $(l(a_i), d(a_i))$.

<u>Απόδειξη</u>

Όλα τα "περιστέρια" μας θα πρέπει να αντιστοιχίζονται σε διαφορετικές "τρύπες".

Έστω 2 κάρτες a_i, a_j με $i < j$. Τότε:

> Αν $a_i < a_j$ τότε κάθε αύξουσα υπακολουθία που ξεκινά από το a_j μπορεί να επεκταθεί από αριστερά με το a_i. Έτσι $l(a_i) > l(a_j)$.

> Αν $a_i > a_j$ τότε αντίστοιχα κάθε φθίνουσα υπακολουθία που ξεκινά από το a_j μπορεί να επεκταθεί από το a_i. Έτσι $d(a_i) > d(a_j)$.

Άρα πράγματι κάθε ζεύγος είναι διαφορετικό κι όλα, τα "περιστέρια" αντιστοιχίζονται σε διαφορετικά "ζεύγη"-"τρύπες".

Αλλά αν δεν υπάρχουν στοίβες (αύξουσες υπακολουθίες) $n+1$ ή στοίβες μεγέθους $m+1$ (φθίνουσες υπακολουθίες) τότε θα υπάρχαν μόνο $n \cdot m$ "τρύπες". Αυτό είναι αδύνατο με βάση την Αρχή του Περιστεριώνα.

> Μια ακολουθία εισόδου για 25 κάρτες που δημιουργούν 5 στοίβες μεγέθους 5 είναι η:

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

Στοίβες:

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

ΑΣΚΗΣΗ 4

α) Γνωρίζουμε από εκφώνηση πως για κάθε ζεύγος (i,t) δηλαδή (σωματίδιο, χρονική στιγμή) ο υπερυπολογιστής της πρώτης περίπτωσης επιστρέφει τη θέση του σωματιδίου i σε σταθερό χρόνο $\Theta(1)$.

Θα βρούμε ότι όλες οι συγκρούσεις συμβαίνουν σε ένα συγκεκριμένο χρονικό διάστημα $[t_s, t_f]$. Η χρονική στιγμή t_f θα είναι για την περίπτωση όπου δύο σωματίδια κινούνται σταθερά με την ελάχιστη ταχύτητα, άρα τη χρονική στιγμή $\frac{L}{2V_{min}}$. Αντίστοιχα, η χρονική στιγμή t_s θα είναι για την περίπτωση όπου δύο σωματίδια κινούνται με τη μέγιστη ταχύτητα, σταθερά, άρα τη χρονική στιγμή $\frac{L}{2V_{max}}$.

Επομένως, όλες οι συγκρούσεις γίνονται στο $\left[\frac{L}{2V_{max}}, \frac{L}{2V_{min}}\right]$.

Για τον αλγόριθμο μας θα χρησιμοποιήσουμε δυαδική αναζήτηση ως εξής. Παρατηρούμε πως αν σε κάποια χρονική στιγμή σωματίδια τύπου β βρίσκονται πιο αριστερά από τύπου α, τότε έχει προηγηθεί μία τουλάχιστον σύγκρουση, οπότε δε μας ενδιαφέρουν όλοι οι μεταγενέστεροι χρόνοι. Αντίστοιχα, αν βρούμε ότι το σωματίδιο τύπου α που έχει διανύσει βρίσκεται πιο αριστερά από το αντίστοιχο σωματίδιο τύπου β, τότε η πρώτη σύγκρουση δεν έχει συμβεί ακόμη και επομένως αναζητούμε σε μεταγενέστερες χρονικές στιγμές. Εάν βρούμε, ωστόσο, ότι το σωματίδιο τύπου α που έχει διανύσει τη μεγαλύτερη απόσταση βρίσκεται στην ίδια θέση με το αντίστοιχο τύπου β τότε βρήκαμε τη στιγμή της πρώτης σύγκρουσης. Τότε, επιστρέφουμε τα δύο σωματίδια και τη χρονική στιγμή t και σταματάμε τη διαδικασία.

Για τα παραπάνω, ασχολούμαστε με τα σωματίδια τύπου α και τύπου β που έχουν διανύσει τη μεγαλύτερη απόσταση, δηλαδή τα queries μας έχουν χρονική πολυπλοκότητα $O(\max\Theta(i,t) + \max\Theta(j,t))$, i,j σωματίδια τύπου a, b αντίστοιχα δηλαδή $O(2n)$, σε κάθε χρονική στιγμή που μελετάμε.

Έπειτα, η δυαδική μας αναζήτηση γίνεται για N χρονικές στιγμές. Αυτές οι χρονικές στιγμές θεωρούμε ότι είναι κβαντισμένες κατά μία σταθερά q. Δηλαδή, με βάση και το διάστημα $[t_s, t_f]$, μελετάμε $N = \frac{1}{q}(t_f - t_s) \Rightarrow N = \frac{1}{q} \cdot L\left(\frac{1}{2V_{min}} - \frac{1}{2V_{max}}\right)$ χρονικά στιγμιότυπα.

Αφού κάνουμε δυαδική αναζήτηση σε αυτά θα πάρουμε:

$$\boxed{O(2n \cdot \log N) = O\left(2n \cdot \log\left(\frac{1}{q} \cdot L\left(\frac{1}{2V_{min}} - \frac{1}{2V_{max}}\right)\right)\right)}$$

κλήσεις

β) Για κάθε ζεύγος i,j από σωματίδια ο υπερυπολογιστής υπολογίζει σε $\Theta(1)$ την χρονική στιγμή όπου τα 2 σωματίδια συγκρούονται.

Για αρχή, επιλέγουμε ένα σωματίδιο τύπου α και υπολογίζουμε τον ελάχιστο χρόνο σύγκρουσης με σωματίδιο β τη χρονική στιγμή t_{ij}, κάνοντας μία γραμμική αναζήτηση σε όλους τους χρόνους που προκύπτουν από τα ζεύγη (α, β_k), $k = 1, 2, \ldots, n$. Δηλαδή, κόστος $O(n)$. Έστω α_r το σωματίδιο αυτό και β_p το σωματίδιο με το οποίο συγκρούστηκε την t_{ij}.

Έπειτα, εφόσον το β_p είναι το πρώτο σωματίδιο που συγκρούστηκε με το α_r δε θα υπάρχουν άλλα σωματίδια τύπου β αριστερά της σύγκρουσης. Αγνοώ λοιπόν όλα τα υπόλοιπα σωματίδια τύπου α αριστερά της σύγκρουσης. Για την εύρεση αυτών επιλέγω το β_p και απορρίπτω όσα σωματίδια τύπου α μας επιστρέφουν χρόνο $t > t_{ij}$.

Επαναλαμβάνουμε την παραπάνω διαδικασία μέχρι να απομείνει μόνο ένα σωματίδιο τύπου α και βρίσκουμε με ποιό από τα σωματίδια τύπου β συγκρούεται πρώτα.

Η παραπάνω ανάλυση ταυτίζεται με τον αλγόριθμο ταξινόμησης Quickselect. Μετά από κάθε εκτέλεση του αλγορίθμου μένουν το πολύ 3/4 σωματίδια τύπου α, έτσι θα γίνουν στη χειρότερη περίπτωση $O(\log n)$ με κόστος $O(n)$ η καθεμιά. Άρα έχουμε πολυπλοκότητα $O(n^2)$ στη χειρότερη περίπτωση και $O(n \log n)$ στη μέση περίπτωση.

ΑΣΚΗΣΗ 5

(α) Για την υποστήριξη της λύσης του παραπάνω προβλήματος η δομή δεδομένων που θα χρησιμοποιήσουμε είναι μία δομή Bloom filter. Στη δομή αυτή είναι δυνατό να υπάρξουν false positives αλλά όχι false negatives όπως μας ζητείται.

Η δομή μας αποτελείται από έναν πίνακα m bits, 1 bit για κάθε κελί του και από 1 hash function η οποία "χαρτογραφεί" ένα σύνολο στοιχείων με μία από τις m θέσεις για κάθε στοιχείο.

• Για να υλοποιήσουμε την λειτουργία "πρόσθεσε το x στο S", τροφοδοτούμε το x στην hash function μας ή αν έχουμε περισσότερες σε κάθε μία τους, για να λάβουμε 1 ή περισσότερες θέσεις στον πίνακα αντίστοιχα. Θέτουμε τις θέσεις αυτές του πίνακα σε 1.

• Για να υλοποιήσουμε τη λειτουργία "έλεγξε αν $x \in S$" το τροφοδοτούμε στις 1 ή περισσότερες hash functions για να λάβουμε 1 ή περισσότερες θέσεις πίνακα. Αν κάποιο από τα bits στις θέσεις αυτές είναι 0 το στοιχείο σίγουρα δεν ανήκει στο S, αν ήταν όλα τα bits στις θέσεις αυτές θα ήταν 1 από την είσοδο του στο S. Αν όλα τα bits είναι 1 τότε είτε το $x \in S$ είτε τα bits έχουν τυχαία τεθεί σε 1 κατά τη προσθήκη άλλων στοιχείων στο S, οδηγώντας σε ένα false positive.

• Η πολυπλοκότητα κάθε λειτουργίας είναι $O(k)$ όπου k ο αριθμός των hash functions, δηλαδή στο ερώτημα αυτό $O(1)$.

• Η hash function μας θα είναι η $h(x) = x \bmod m$.

Για τη πιθανότητα false positive απάντησης:

Υποθέτουμε ότι η hash function επιλέγει κάθε θέση από τις m του πίνακα με ίση πιθανότητα. Αν m οι θέσεις του πίνακα, η πιθανότητα ένα συγκεκριμένο bit-κελί να μην τεθεί σε 1 είναι : $1 - \frac{1}{m}$

Αν έχουμε εισάγει n στοιχεία, η πιθανότητα για να είναι ένα συγκεκριμένο bit ακόμα 0 είναι:

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

Η πιθανότητα να είναι 1, είναι επομένως:

$$P_{conflict}=1-\left(1-\frac{1}{m}\right)^n \approx 1-e^{-\frac{n}{m}}$$

Άρα αυτή είναι κι η πιθανότητα για ένα false positive.

> Αν $m=8n$, η πιθανότητα false positive απάντησης θα είναι: $P_{conflict}=1-\left(1-\frac{1}{m}\right)^n=1-\left(\frac{8n-1}{8n}\right)^n \approx 1-e^{-\frac{1}{8}}$

$$\approx 0{,}118$$

β) Έστω, τώρα ότι έχουμε $k \geq 1$ hash functions. Η δομή μας θα παραμείνει σχεδόν ίδια με εκείνη του ερωτήματος (α) με μόνη διαφορά τον αριθμό των hash functions και ότι οι λειτουργίες μας θα επηρεάζουν / επιστρέφουν k bits κι όχι 1.

• Η πιθανότητα για κάθε θέση του πίνακα να είναι 1 παραμένει $\frac{1}{m}$ κι η πιθανότητα να μη γίνει conflict είναι $1-\frac{1}{m}$. Αυτή τη φορά, όμως, έχοντας k hash functions για κάθε ένα από τα n στοιχεία, η πιθανότητα να μην υπάρχει conflict είναι:

$$P_{no\_confl}=\left(1-\frac{1}{m}\right)^{n\cdot k}$$

• Η πιθανότητα να επιστρέφει false positive απάντηση μία από τις k hash functions είναι:

$$P_{conflict}=1-\left[1-\frac{1}{m}\right]^{n\cdot k}$$

Άρα για k hash functions:

$$\boxed{P_{false\_positive}=\left[1-\left(1-\frac{1}{m}\right)^{n\cdot k}\right]^k}$$

Για $m=8n$:

$$P_{false\_positive} = \left[1-\left(\frac{8n-1}{8n}\right)^{n\cdot k}\right]^k$$

• Για την εύρεση της βέλτιστης τιμής του k ψάχνουμε το min της συνάρτησης:

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

Έπειτα από πράξεις καταλήγουμε ότι:

$$k_{best} = \frac{m}{n}\ln 2$$

Αν $m=8n$: $k_{best} = \frac{8n}{n}\ln 2 = 8\ln 2 = 5{,}546 \Rightarrow$

$$\boxed{k_{best} = 5{,}546 \text{ hash functions}}$$

ΑΣΚΗΣΗ 6

Για την αποδοτική αποθήκευση των συμβολοσειρών του S θα χρησιμοποιήσουμε tries.
Για την αποδοτική απάντηση ερωτημάτων κοινού προθέματος (μέγιστου) θα λειτουργήσουμε ως κάτωθι:

> Αφού $k \in S$, οι k λέξεις θα υπάρχουν στο trie του συνόλου S άρα θα χρησιμοποιήσω την ίδια δομή.
> Θα διατρέξουμε τις πρώτες k-1 λέξεις με τη σειρά. Σε κάθε κόμβο-γράμμα θα αποθηκεύουμε μία μεταβλητή c που θα λειτουργεί ως μετρητής, την τιμή της οποίας θα αυξάνω κατά 1 κάθε φορά που περνάω από τον κόμβο.
> Για την τελευταία λέξη, τη διασχίζουμε μέχρι να βρούμε κόμβο-γράμμα με $c<k-1$ και επιστρέφουμε τα γράμματα με $c=k$. Όταν βρούμε κόμβο με $c<k-1$ η αναζήτηση έχει τελειώσει.

Οι απαιτήσεις των παραπάνω θα είναι:

- Για τη δημιουργία: $\underline{O(N)}$, όπου N το συνολικό μήκος των λέξεων στο S.
- Απαίτηση σε χώρο: $\underline{O(N)}$
- Χρόνος για απάντηση του ερωτήματος: $O\left(\sum_{i=1}^{k} len(s_i)\right)$