Δομές Δεδομένων

Μέρος 1ο: Εισαγωγή

Εξάμηνο Σπουδών: 6ο

Κωδικός Μαθήματος: 681

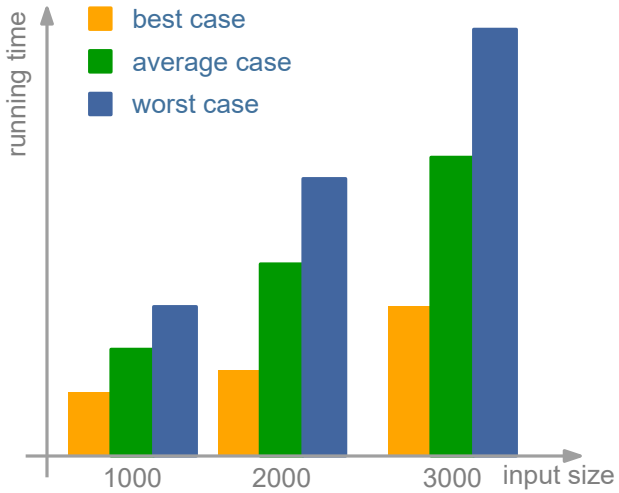
Τμήμα Μαθηματικών
Πανεπιστήμιο Ιωαννίνων

Μιχάλης Α. Μπέκος
bekos@uoi.gr

Εισαγωγή

- Αλγόριθμος: Μια ακολουθία υπολογιστικών βημάτων, η οποία απεικονίζει την είσοδο ενός προβλημάτος (τα δεδομένα) στην έξοδο (τη λύση)

- Νόμιμη είσοδος: Ικανοποιεί τις προδιαγραφές του προβλήματος
- Στιγμιότυπο: Αποτίμηση των παραμέτρων του προβλήματος
- Χρόνος εκτέλεσης: Τυπικά αυξάνει με το μέγεθος της εισόδου
  - Ο μέσος χρόνος είναι συχνά δύσκολο να καθοριστεί
  - Συνήθως προσδιορίζουμε τον χρόνο χειρότερης περίπτωσης



Ορθότητα Αλγορίθμων

- Επαγωγή: Induction

  Έστω μια πρόταση π, η οποία εξαρτάται από ένα φυσικό αριθμό $n \geq n_0$.

  Εάν αποδειχθεί ότι η πρόταση π ισχύει:

  - για $n = n_0$, και
  - για $n = k + 1$ εφόσον ισχύει για κάθε $n \leq k$,

  τότε η πρόταση π ισχύει για κάθε φυσικό αριθμό $n \geq n_0$.

Ορθότητα Αλγορίθμων

- **Αμετάβλητη συνθήκη βρόχου:** Πρόταση που παραμένει αληθής μετά την ολοκλήρωση κάθε επανάληψης ενός βρόχου
  Loop invariant

Ψευδοκώδικας

Περιγραφή υψηλού επιπέδου ενός αλγορίθμου

Λιγότερο αναλυτική από ένα πρόγραμμα

Αποκρύπτει θέματα σχεδιασμού υλοποίησης

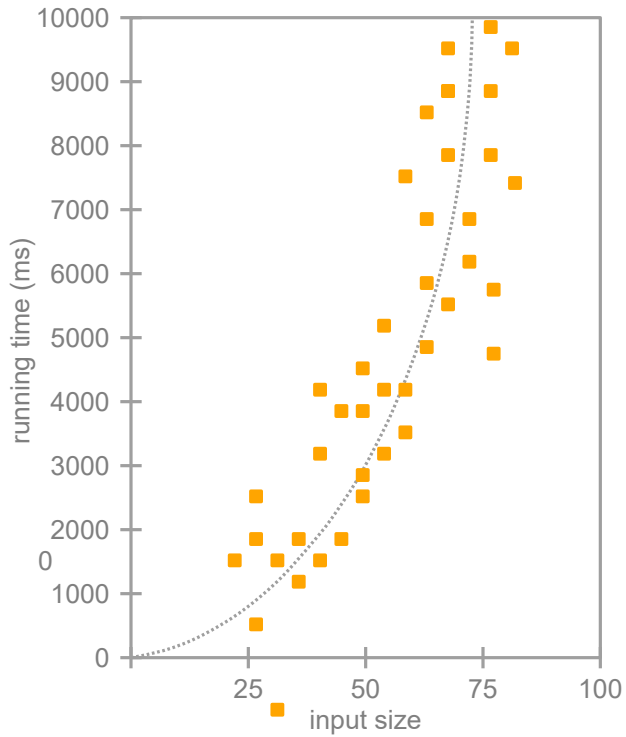
Είσοδος: Ένα διάνυσμα A με n στοιχεία.
Εξοδος : Το ελάχιστο στοιχείο του A.

```
1 min = A[1];
2 for i = 1 to n do
3 |   if A[i] < min then
4 |   |   min = A[i];
5 return min;
```

"*Κατά το ξεκίνημα της i-οστής επανάληψης, η μεταβλητή min αποθηκεύει το ελάχιστο των πρώτων i στοιχείων*"

Αποδοτικότητα Αλγορίθμων

- **Πειραματική Αξιολόγηση Αλγορίθμων**
  - Υλοποίηση ενός προγράμματος που εφαρμόζει τον αλγόριθμο
  - Εκτέλεση του προγράμματος με εισόδους διαφορετικού μεγέθους και σύνθεσης
  - Χρησιμοποίηση μιας μεθόδου τύπου clock() για τη λήψη ακριβής μέτρησης του χρόνου εκτέλεσης
  - Απεικόνιση των αποτελεσμάτων



- Περιορισμοί:
  - Απαιτείται υλοποίηση, η οποία μπορεί να ναι δύσκολη
  - Τα αποτελέσματα ενδέχεται να μην είναι ενδεικτικά για άλλες εισόδους
  - Για τη σύγκριση δύο αλγορίθμων απαιτείται η χρήση κοινών περιβάλλοντων υλικού/λογισμικού

Αποδοτικότητα Αλγορίθμων

- **Θεωρητική Ανάλυση Αλγορίθμων**

Είσοδος: Ένα διάνυσμα A με n στοιχεία.
Εξοδος : Το ελάχιστο στοιχείο του A.

```
1 min = A[1];
2 for i = 1 to n do
3 |   if A[i] < min then
4 |   |   min = A[i];
5 return min;
```

γραμμή	κόστος	αρ. εκτελέσεων
1	c_1	1
2	c_2	n
3	c_3	$n-1$
4	c_4	$\leq n-1$
5	c_5	1

- Συνολικός χρόνος: $\leq c_1 + c_2 n + c_3(n-1) + c_4(n-1) + c_5 \quad \Rightarrow$

$$T[n] \leq (c_2 + c_3 + c_4)n + (c_1 - c_3 - c_4 + c_5)$$

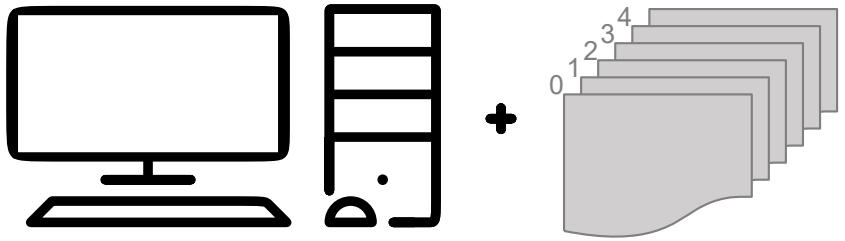
εξαρτάται από το μέγεθος της εισόδου

Το μοντέλο τυχαίας προσπέλασης

Random Access Machine (RAM) Model

Αποτελείται από:

- Ένα επεξεργαστή
  CPU
- Μια συστοιχία κελιών μνήμης (δυνητικά απεριόριστη)
  Memory cells



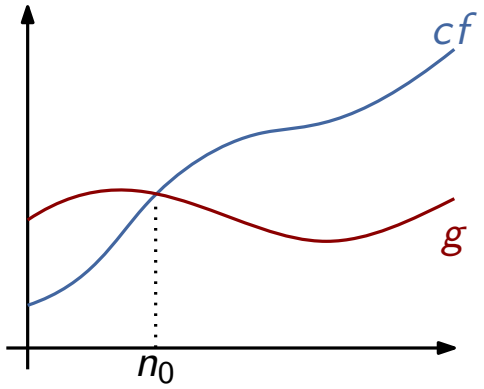
  - κάθε κελί στη συστοιχία είναι αριθμημένο
  - και μπορεί να περιέχει έναν αριθμό ή χαρακτήρα
  - η πρόσβαση στο περιεχόμενο ενός κελιού γίνεται σε μοναδιαίο χρόνο

Θεωρητική Ανάλυση Αλγορίθμων

- Χρησιμοποιείται μια περιγραφή υψηλού επιπέδου του αλγορίθμου αντί μιας υλοποίησης
- Επιτρέπει την αξιολόγηση της απόδοσης ενός αλγορίθμου ανεξάρτητα από το περιβάλλον υλικού/λογισμικού
- Χαρακτηρίζει τον χρόνο εκτέλεσης ως συνάρτηση του μεγέθους εισόδου
- Λαμβάνει υπόψη όλες τις πιθανές εισόδους του προβλήματος
- Επιτυγχάνεται εξετάζοντας τους βασικούς υπολογισμούς που εκτελούνται από τον αλγόριθμο
  - Ανάθεση τιμής σε μια μεταβλητή
  - Κλήση μιας μεθόδου
  - Επιστροφή τιμής από μια μέθοδο
  - Αποτίμηση μιας έκφρασης
- Υπόθεση: Κάθε τέτοιος υπολογισμός έχει μοναδιαίο χρόνο εκτέλεσης στο RAM μοντέλο

Ασυμπτωτική Ανάλυση

- Ρυθμός ανάπτυξης: Έστω μια συνάρτηση $f : \mathbb{N} \to \mathbb{R}$
  Order of growth



$$\mathcal{O}(f(n)) = \{g : \mathbb{N} \to \mathbb{R} : \ \exists c > 0, \exists n_0 > 0, \forall n > n_0 : g(n) \leq cf(n)\}$$

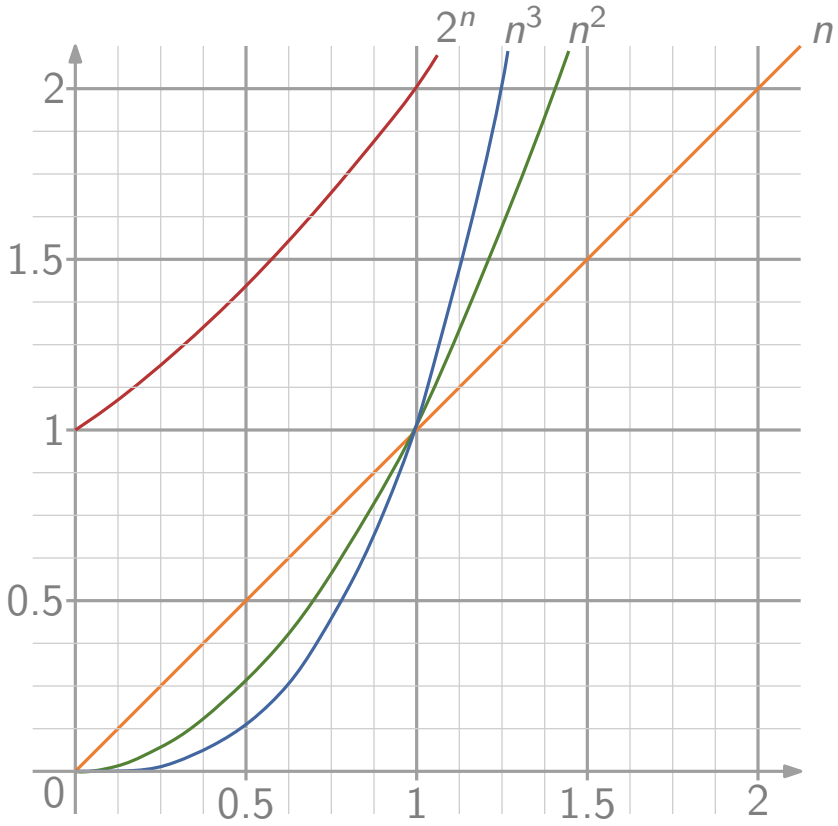
$=$ Όλες οι συναρτήσεις που έχουν ασυμπτωτικό άνω φράγμα την συνάρτηση f
asymptotic upper bound

- Παραδείγματα:
  - $5n = \mathcal{O}(n)$
  - $5n + 100 = \mathcal{O}(n)$
  - $n = \mathcal{O}(n \log n)$
  - $\log n = \mathcal{O}(n)$

- Ιδιότητες:
  - $g_1(n) = \mathcal{O}(f(n))$ και $g_2(n) = \mathcal{O}(f(n)) \Rightarrow c_1 g_1(n) + c_2 g_2(n) = \mathcal{O}(f(n))$
  - $g_1(n) = \mathcal{O}(f_1(n))$ και $g_2(n) = \mathcal{O}(f_2(n)) \Rightarrow g_1(n) \cdot g_2(n) = \mathcal{O}(f_1(n) f_2(n))$
  - μεταβατικότητα

Επτά Σημαντικές Συναρτήσεις

- Επτά συναρτήσεις που εμφανίζονται συχνά στην ανάλυση αλγορίθμων:
  - Σταθερή $\rightarrow \mathcal{O}(1)$
  - Λογαριθμική $\rightarrow \mathcal{O}(\log n)$
  - Γραμμική $\rightarrow \mathcal{O}(n)$
  - N-Log-N $\rightarrow \mathcal{O}(n \log n)$
  - Τετραγωνική $\rightarrow \mathcal{O}(n^2)$
  - Κυβική $\rightarrow \mathcal{O}(n^3)$
  - Εκθετική $\rightarrow \mathcal{O}(2^n)$
  - Παραγοντικό $\rightarrow \mathcal{O}(n!)$



- Παράδειγμα: Ταξινόμηση n στοιχείων
  - Insertion sort: $O(n^2)$
  - Merge sort: $O(n \log n)$

$n = 10^6$

$\approx$ 50 ώρες

$\approx$ 40 δευτ.

Βασικοί Κανόνες

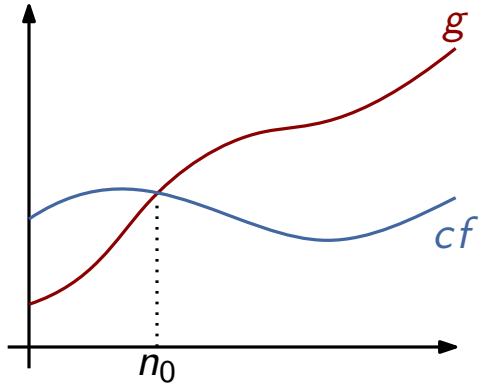
- Αν η συνάρτηση $f(n)$ είναι ένα πολυώνυμο βαθμού d, τότε $f(n) = \mathcal{O}(n^d)$, δηλ.
  - αγνοούμε όρους χαμηλότερης τάξης
  - αγνοούμε σταθερούς παράγοντες

- Χρησιμοποιούμε τη μικρότερη δυνατή κλάση συναρτήσεων
  - προτιμούμε $2n = \mathcal{O}(n)$ αντί $2n = \mathcal{O}(n^2)$

- Χρησιμοποιούμε την απλούστερη έκφραση της κλάσης
  - προτιμούμε $3n + 7 = \mathcal{O}(n)$ αντί $3n + 7 = \mathcal{O}(3n)$

Ασυμπτωτική Ανάλυση: Συνέχεια

- Κάτω φράγμα:
  Lower bound

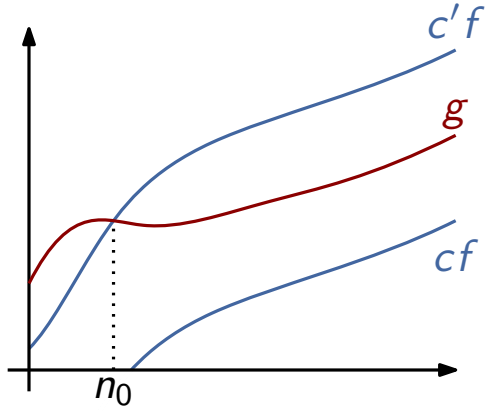


Έστω μια συνάρτηση $f : \mathbb{N} \to \mathbb{R}$

$$\Omega(f(n)) = \{g : \mathbb{N} \to \mathbb{R} : \ \exists c > 0, \exists n_0 > 0, \forall n > n_0 : cf(n) \leq g(n)\}$$

$=$ Όλες οι συναρτήσεις που έχουν ασυμπτωτικό κάτω φράγμα την συνάρτηση f
asymptotic lower bound

- Σφιχτό Φράγμα:
  Tight bound



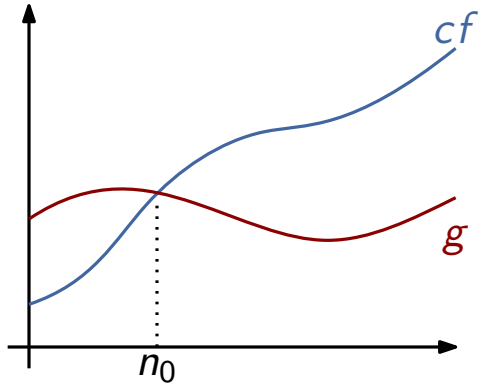
$$\Theta(f(n)) = \{g : \mathbb{N} \to \mathbb{R} : \ \exists c, c' > 0, \exists n_0 > 0, \forall n > n_0 : cf(n) \leq g(n) \leq c'f(n)\}$$

$=$ Όλες οι συναρτήσεις που έχουν ασυμπτωτικό σφιχτό φράγμα την συνάρτηση f
asymptotic tight bound

Παραδείγματα: ○ $n^2 = \Omega(n)$ ○ $10n^2 + 20 = \Theta(n^2)$

Ασυμπτωτική Ανάλυση: Συνέχεια

- Αυστηρό άνω φράγμα: Έστω μια συνάρτηση $f : \mathbb{N} \to \mathbb{R}$
  Strict upper bound

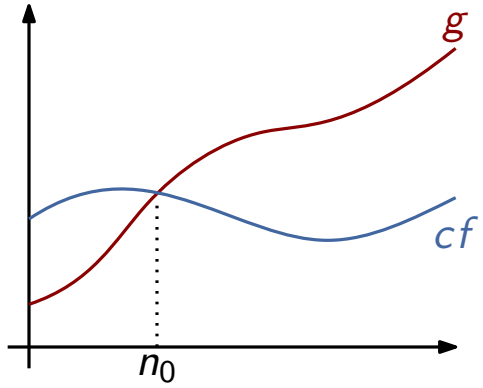


$$o(f(n)) = \{g : \mathbb{N} \to \mathbb{R} : \ \forall c > 0, \exists n_0 > 0, \forall n > n_0 : g(n) < cf(n)\}$$

$=$ Όλες οι συναρτήσεις που έχουν αυστηρό ασυμπτωτικό άνω φράγμα την συνάρτηση f

- Αυστηρό κάτω φράγμα
  Strict lower bound

$$\omega(f(n)) = \{g : \mathbb{N} \to \mathbb{R} : \ \forall c > 0, \exists n_0 > 0, \forall n > n_0 : cf(n) < g(n)\}$$



$=$ Όλες οι συναρτήσεις που έχουν αυστηρό ασυμπτωτικό κάτω φράγμα την συνάρτηση f

Παραδείγματα: ○ $n \log n = o(n^2)$ ○ $10n^2 + 20 = \omega(n \log n)$

Πολυπλοκότητα Προβλήματος

- Άνω όριο: Καθορίζεται από τον “καλύτερο” αλγόριθμο επίλυσης του
  Upper bound

- Κάτω όριο: Κάθε αλγόριθμος που το επιλύει έχει τουλάχιστον τέτοια πολυπλοκότητα
  Lower bound

- Παράδειγμα: Ταξινόμηση n στοιχείων
  - Άνω όριο: $\mathcal{O}(n \log n)$ $\rightarrow$ Merge Sort
  - Κάτω όριο: $\Omega(n \log n)$

Πολυπλοκότητα Αλγορίθμου

Ένας αλγόριθμος έχει πολυπλοκότητα $\Theta(f(n))$ εάν:

- τερματίζει μετά από χρόνο $\mathcal{O}(f(n))$
- υπάρχει ένα στιγμιότυπο του προβλήματος, που αναγκάζει τον αλγόριθμο να εκτελείται για $\Omega(f(n))$ χρόνο

Ανάλυση Χειρότερης και Μέσης Περίπτωσης

Worst and average case analysis

Είσοδος: Ένα διάνυσμα A με n στοιχεία και ένας ακέραιος x.
Εξοδος : Η θέση του x στο A; -1, εάν $x \notin A$.

```
1 for i = 1 to n do
2 |   if A[i] == x then
3 |   |   return i;
4 return -1;
```

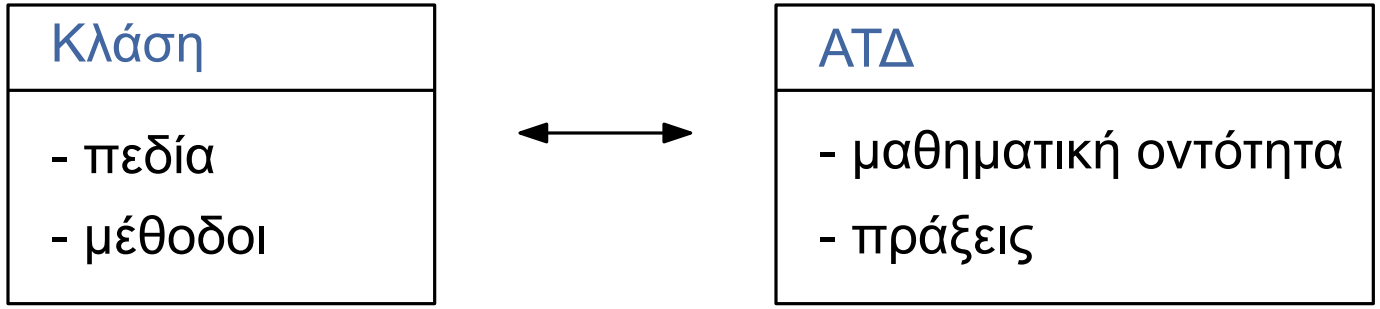
- Χειρότερη περίπτωση: n επαναλήψεις $\rightarrow \mathcal{O}(n)$
- Μέση περίπτωση:
  - $x \in A$: $\frac{n}{2}$ επαναλήψεις
  - $x \notin A$: n επαναλήψεις

  $\rightarrow \mathcal{O}(n)$

Αφηρημένος τύπος δεδομένων

Abstract data type

- Μια μαθηματική οντότητα μαζί με μια συλλογή πράξεων/λειτουργιών επί των στοιχείων της
- Αντιστοιχία κλάσης - ΑΤΔ:



- Παράδειγμα: ΑΤΔ μοντελοποίησης ενός απλού συστήματος διαπραγμάτευσης μετοχών
  - Στοιχεία: Παραγγελίες αγοράς / πώλησης
  - Λειτουργίες:

```
order buy(stock, shares, price)
order sell(stock, shares, price)
void cancel(order)
```

Χρήσιμες Μαθηματικές Ιδιότητες

- Ιδιότητες λογαρίθμων:
  - $\log_x a^b = b \log_x a$
  - $\log_x ab = \log_x a + \log_x b$
  - $x = y^{\log_y x}$
  - $\log_x a = \frac{\log_y a}{\log_y x}$
  - $\log_x \frac{a}{b} = \log_x a - \log_x b$
- Ιδιότητες αθροισμάτων:
  - $\sum_{i=1}^{n} q^i = \frac{q^{n+1}-1}{q-1}$
  - $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$
  - $\sum_{i=1}^{\infty} q^i = \frac{1}{1-q}$, για $q < 1$
  - $\sum_{i=1}^{n} \frac{1}{i} = H_n$, όπου $\ln n \leq H_n \leq \ln n + 1$

Ανάλυση Επιμερισμένης Περίπτωσης

Amortised analysis

- T[n] $\leftarrow$ ο μέγιστος χρόνος εκτέλεσης μιας οποιασδήποτε ακολουθίας n πράξεων

  $\Rightarrow \frac{T[n]}{n}$: ο επιμερισμένος χρόνος εκτέλεσης μιας πράξης

- Παράδειγμα: ΑΤΔ δυαδικός μετρητής

Binary Counter
- `setZero()` - `increment()`

πράξη: αλλαγή ψηφίου

$1111 \xrightarrow{\texttt{increment()}} 10000 \xrightarrow{\texttt{increment()}} 10001$

- Παρατήρηση: Το i-οστό ψηφίο αλλάζει $\lfloor n/2^i \rfloor$ φορές

  Ο μετρητής έχει $\lceil \log n \rceil$ ψηφία

$\Rightarrow$ $\boxed{T[n] = \sum_{i=0}^{\lceil \log n \rceil} \lfloor \frac{n}{2^i} \rfloor \leq n \cdot \frac{1}{1-\frac{1}{2}} = 2n}$

σταθερός επιμερισμένος χρόνος

Η Μέθοδος Λογαριασμού Τραπεζίτη

Bank account method

- Κάθε πράξη χρεώνεται ένα επιμερισμένο κόστος (π.χ., σε νομίσματα)
- "Φθηνές" πράξεις χρεώνονται παραπάνω απ'το πραγματικό τους κόστος δημιουργώντας απόθεμα
- Το πραγματικό κόστος των "ακριβών" πράξεων πληρώνεται από το απόθεμα
- Το συνολικό κόστος n πράξεων ισούται με το πλήθος των νομισμάτων που ξοδέψαμε

- Πίσω στο παράδειγμα:
  - "Χρεώνουμε" 2 μονάδες για να θέσουμε ένα ψηφίο στην τιμή 1.
    [η `increment()` αλλάζει το πολύ ένα ψηφίο από "0" $\rightarrow$ "1"]
  - $\Rightarrow$ Κόστος ακολουθίας: $\mathcal{O}(n)$

0	0	0	0
0	0	0	1^1
0	0	1^1	0
0	0	1^1	1^1
0	1^1	0	0
0	1^1	0	1^1

Επιπλέον υλικό

- Κεφάλαιο 1:
  Michael T. Goodrich, Roberto Tamassia, Αλγόριθμοι σχεδίαση και εφαρμογές
  ISBN: 9789605126971, εκδόσεις Γκιούρδα, 2016.