Δομές Δεδομένων

Εισαγωγή

Α. Συμβώνης
Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Εφαρμοσμένων Μαθηματικών και Φυσικών Επιστημών
Τομέας Μαθηματικών

Γενικές Πληροφορίες

- Διδάσκων: Αντώνιος Συμβώνης
  ΣΕΜΦΕ, κτίριο Ε, 3.18
  symvonis@math.ntua.gr
  www.math.ntua.gr/~symvonis
  Helios …ΕΜΦΕ

- Διαλέξεις: Τρίτη 8:45-10:30
  Παρασκευή 12:45-14:30
  online
  BigBlueButton

- Αξιολόγηση: Προγραμματιστικές/Γραπτές Ασκήσεις: 15%
  Γραπτό Διαγώνισμα: 85%

Γενικές Πληροφορίες

- Σύγγραμμα: Δομές Δεδομένων και Αλγόριθμοι σε Java, Πέμπτη έκδοση, 2013
  Michael T. Goodrich, Roberto Tamassia
  Επιμέλεια - Μετάφραση: Μιχάλης Χατζόπουλος
  Δίαυλος Α.Ε. Εκδόσεις Βιβλίων

  Δομές Δεδομένων – Ταξινόμηση και Αναζήτηση με Java
  Παναγιώτης Μποζάνης
  Εκδόσεις Τζιόλα

  Αλγόριθμοι και Δομές Δεδομένων: τα Βασικά Εργαλεία
  Kurt Mehlhorn, Peter Sanders
  Εκδόσεις Κλειδάριθμος

Αλγόριθμος

Αλγόριθμος:

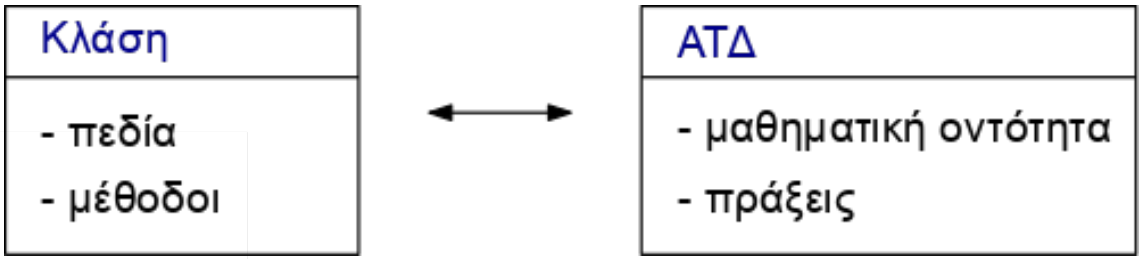
Μία ακολουθία υπολογιστικών βημάτων η οποία απεικονίζει την είσοδο του προβλήματος (τα δεδομένα) στην έξοδο (την λύση, το αποτέλεσμα).

- Νόμιμη είσοδος: Ικανοποιεί τις προδιαγραφές του προβλήματος
- Στιγμιότυπο

Αφηρημένος Τύπος Δεδομένων [Abstract Data Type]

Αφηρημένος Τύπος Δεδομένων (ΑΤΔ):
Μία μαθηματική οντότητα μαζί με μία συλλογή πράξεων επί των στοιχείων της.

- Αντιστοιχία κλάσης - ΑΤΔ



Ορθότητα Αλγορίθμων

Επαγωγή [*induction*]:

[*Ισχυρή επαγωγή – strong induction*]

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

Εάν αποδειχθεί ότι η π:

- ισχύει για $n = n_0$, και
- ισχύει για $n = k + 1$ εφόσον ισχύει για **κάθε** $n \leq k$, τότε η π ισχύει για κάθε $n \geq n_0$

Ορθότητα Αλγορίθμων

Αμετάβλητη συνθήκη βρόχου [*loop invariant*]:

Πρόταση που παραμένει αληθής μετά την ολοκλήρωση κάθε επανάληψης ενός βρόχου.

```
Algorithm getMinFromArray(A)
(1)  min = A[0];
(2)  for i=1 to A.length-1 do
(3)      if A[i]<min then
(4)          min = A[i];
(5)      end
(6)  end
(7)  return min;
```

Loop Invariant:

Κατά την ολοκλήρωση της i-οστής επανάληψης, $0 \leq i \leq A.length - 1$, η μεταβλητή min αποθηκεύει το ελάχιστο των πρώτων $i + 1$ στοιχείων.

Ανάλυση Αλγορίθμων

Γραμμή	Κόστος	# Εκτελέσεων
1	c_1	1
2	c_2	n
3	c_3	$n-1$
4	c_4	$\leq n-1$
7	c_5	1

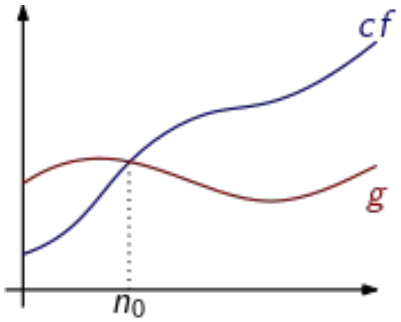
```
Algorithm getMinFromArray(A)
(1)  min = A[0];
(2)  for i=1 to A.length-1 do
(3)      if A[i]<min then
(4)          min = A[i];
(5)      end
(6)  end
(7)  return min;
```

- Συνολικός χρόνος:

$$\leq c_1 + c_2 \cdot n + c_3 \cdot (n-1) + c_4 \cdot (n-1) + c_5 \Rightarrow$$

$$\boldsymbol{T(n) \leq (c_2 + c_3 + c_4) \cdot n + (c_1 - c_3 - c_4 - c_5)}$$

Ασυμπτωτική Ανάλυση



Άνω φράγμα [*upper bound*]:

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

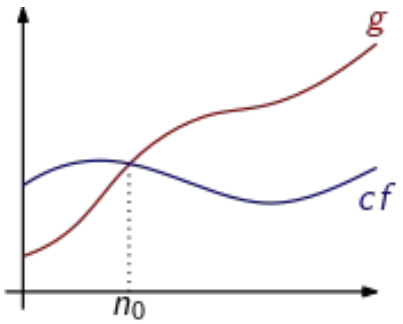
$O(f) = \{g: \mathbb{N} \to \mathbb{N} | \exists c, n_0 > 0: g(n) \leq cf(n), \forall n \geq n_0\}$

= Όλες οι συναρτήσεις που έχουν ασυμπτωτικό άνω όριο (*asymptotic upper bound*) τη συνάρτηση f

- Ιδιότητες
  - $f_1(n) = O(g(n))$ και $f_2(n) = O(g(n)) \Rightarrow c_1 f_1(n) + c_2 f_2(n) = O(g(n))$
  - $f_1(n) = O(g_1(n))$ και $f_2(n) = O(g_2(n)) \Rightarrow f_1(n) \cdot f_2(n) = O(g_1(n) \cdot g_2(n))$
  - μεταβατικότητα

- Παραδείγματα
  - $5n = O(n)$
  - $5n + 1000 = O(n)$
  - $n = O(n \log n)$
  - $\log n = O(n)$

Ασυμπτωτική Ανάλυση

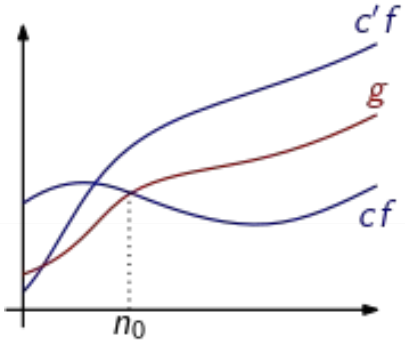


Κάτω φράγμα [*lower bound*]:

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

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

= Όλες οι συναρτήσεις για τις οποίες η f συνιστά ασυμπτωτικό κάτω όριο (*asymptotic lower bound*)



Αυστηρό Όριο [*tight bound*]:

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

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

= Όλες οι συναρτήσεις που αποτελούν αυστηρό όριο (*asymptotic tight bound*) της f

- Παραδείγματα (Ω)
  - $5n = \Omega(n)$
  - $n^2 = \Omega(n)$
  - $5n - 1000 = \Omega(n)$
- Παραδείγματα (Θ)
  - $10n^2 + 20 = \Theta(n^2)$

Πολυπλοκότητα Προβλήματος

Άνω όριο [*upper bound*]:
Καθορίζεται από τον "καλύτερο" αλγόριθμο επίλυσης του προβλήματος.

Κάτω όριο [*lower bound*]:
Κάθε αλγόριθμος που το επιλύει έχει τουλάχιστον τέτοια πολυπλοκότητα.

- Παράδειγμα
  - Ταξινόμηση n στοιχείων
    - Άνω όριο: $O(n \log n)$ → Merge Sort
    - Κάτω όριο: $\Omega(n \log n)$

$\Rightarrow \boldsymbol{\Theta(n \log n)}$

Πολυπλοκότητα Αλγορίθμου

- Ένας **αλγόριθμος** έχει πολυπλοκότητα $\Theta(f)$ εάν:
  - τερματίζει μετά από $O(f)$ χρόνο
  - υπάρχει ένα στιγμιότυπο του προβλήματος που αναγκάζει τον αλγόριθμο να εκτελείται για $\Omega(f)$ χρόνο

Ανάλυση Χειρότερης και Μέσης Περίπτωσης

```
Algorithm findPosition(A,x)
(1)  for i=0 to A.length-1 do
(2)      if A[i]==x then
(3)          return i;
(4)      end
(5)  end
(6)  return -1;
```

- Χειρότερη περίπτωση [*worst case*]: n επαναλήψεις → $O(n)$
- Μέση περίπτωση [*average case*]: $x \in A$: $\frac{n}{2}$ επαναλήψεις
  $x \notin A$: n επαναλήψεις
  → $O(n)$

Ανάλυση Επιμερισμένης/Κατανεμημένης Περίπτωσης

- $T[n]$: Ο μέγιστος χρόνος εκτέλεσης μιας **οποιασδήποτε** ακολουθίας n πράξεων επί μιας δομής

  $\Rightarrow \frac{\mathrm{T}[n]}{n}$: ο επιμερισμένος [*amortized case*] χρόνος για μία πράξη

- Παράδειγμα:
  - ΑΤΔ Binary Counter [δυαδικός μετρητής] εύρους $0 \ldots n = 2^k - 1$.

Binary Counter
$n : (\alpha_{k-1}, \ldots, \alpha_2, \alpha_1, \alpha_0)$
- setZero() - increment()

Κόστος: αλλαγή ψηφίου

1111 $\xrightarrow{\text{increment}}$ 10000 $\xrightarrow{\text{increment}}$ 10001

Ανάλυση Επιμερισμένης/Κατανεμημένης Περίπτωσης

- Ανάλυση - #1:
  - **Παρατήρηση**: Κατά την αύξηση ενός αριθμού, μπορεί να αλλάξει κάθε ένα από τα k ψηφία του μετρητή.

$$\Rightarrow T[n] \leq \sum_{i=0}^{n-1} k = nk = n\log(n+1)$$
$$\Rightarrow \boldsymbol{T[n] \leq n\,log(n+1)}$$

- Ανάλυση - #2:
  - **Παρατήρηση**: Το i-οστό ψηφίο (από τα αριστερά, $1 \leq i \leq k$) αλλάζει 2^i φορές.

$$\Rightarrow T[n] = \sum_{i=1}^{k} 2^i = \sum_{j=0}^{k-1} 2^{j+1} = 2\sum_{j=0}^{k-1} 2^j = 2\left(2^k - 1\right) = 2n$$
$$\Rightarrow \boldsymbol{T[n] = 2n}$$

Η Μέθοδος Λογαριασμού Τραπεζίτη [Banker Account Method]

- Κάθε πράξη χρεώνεται ένα επιμερισμένο κόστος το οποίο ίσως είναι μικρότερο ή μεγαλύτερο από το πραγματικό
- Η διαφορά ανάμεσα στο πραγματικό και το επιμερισμένο κόστος χαρακτηρίζεται ως: πίστωση (credit) ή δάνειο.

- Για το παράδειγμα του δυαδικού μετρητή:
  - "Χρεώνουμε" 2 μονάδες για να θέσουμε ένα ψηφίο στην τιμή 1.
    [*Το πολύ ένα ψηφίο γίνεται από "0" → "1"*]
  - Γίνονται ακριβώς n αυξήσεις του μετρητή.
    $\Rightarrow \boldsymbol{T[n] = 2n}$

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