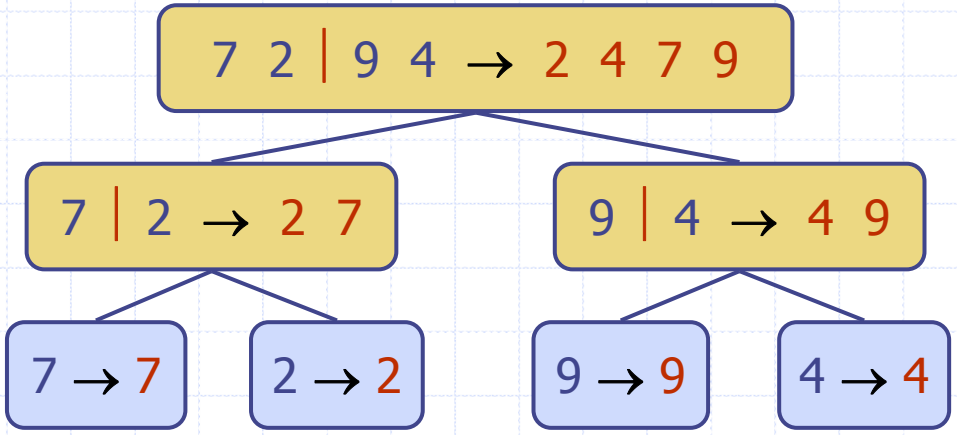
ΕΛΛΗΝΙΚΗ ΔΗΜΟΚΡΑΤΙΑ
ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΡΗΤΗΣ

Αλγόριθμοι και πολυπλοκότητα

Διαίρει και Κυρίευε

Ιωάννης Τόλλης
Τμήμα Επιστήμης Υπολογιστών

Διαίρει και Κυρίευε

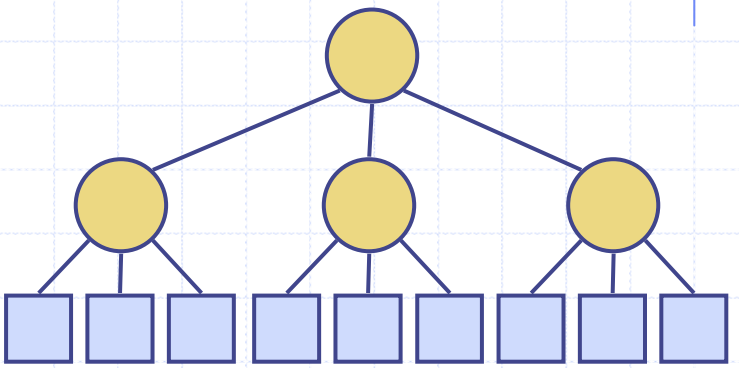


Περιγραφή

- Διαίρει και Κυρίευε παράδειγμα (§5.2)
- Ανασκόπηση Merge-sort (§4.1.1)
- Αναδρομικές Σχέσεις (§5.2.1)
  - Επαναληπτικές αντικαταστάσεις
  - Αναδρομικά δέντρα
  - Guess-and-test
  - Η κύρια μέθοδος
- Πολλαπλασιασμός Ακεραίων (§5.2.2)

Διαίρει και Κυρίευε

- Διαίρει και Κυρίευε είναι ένας γενικός αλγόριθμος σχεδίασης παράδειγμα:
  - Διαίρει : Το σύνολο εισόδου S, χωρίζεται σε δύο ή περισσότερα υποσύνολα $S_1, S_2, \ldots$
  - Κυρίευε: Επίλυση των επιμέρους προβλημάτων αναδρομικά
  - Συνδύασε: Συνδυασμός των λύσεων $S_1, S_2, \ldots$, σε μία λύση για το S
- Το βασικό βήμα για την αναδρομή είναι επιμέρους προβλήματα σταθερούς μεγέθους
- Η ανάλυση μπορεί να γίνει χρησιμοποιώντας **αναδρομικές σχέσεις**



Ανασκόπηση Συγχωνευτικής Ταξινόμησης

- Η συγχωνευτική ταξινόμηση (merge sort) με μία ακολουθιακή είσοδο S με n στοιχεία αποτελείται από τρία βήματα:
  - Διαίρει: Το σύνολο εισόδου S, χωρίζεται σε δύο υποσύνολα S_1 και S_2 με $n/2$ στοιχεία το καθένα
  - Κυρίευε: Τα S_1 και S_2 ταξινομούνται αναδρομικά
  - Συνδύασε: Ταξινόμηση των S_1 και S_2 σε μία ταξινομημένη ακολουθία

Algorithm *mergeSort*(S, C)

Input sequence S with n elements, comparator C

Output sequence S sorted according to C

if $S.size() > 1$

$(S_1, S_2) \leftarrow partition(S, n/2)$

$mergeSort(S_1, C)$

$mergeSort(S_2, C)$

$S \leftarrow merge(S_1, S_2)$

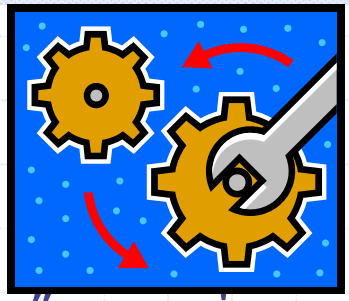
Ανάλυση Αναδρομικών Σχέσεων

- Το βήμα κυρίευε της συγχωνευτικής ταξινόμησης αποτελείται από την συγχώνευση δύο ταξινομημένων ακολουθιών,κάθε μία με ***n***/2 στοιχεία και η υλοποίηση τους με διπλά συνδεδεμένη λίστα, παίρνει περισσότερο από ***bn*** βήματα, για κάποιο σταθερό ***b***.
- Επιπλέον, η περίπτωση (***n*** < 2) θα απαιτούσε ***b*** περισσότερα βήματα.
- Επομένως, εάν θεωρήσουμε ***T(n)*** να υποδηλώνει τον τρέχων χρόνο της συγχωνευτικής ταξινόμησης τότε:

$$T(n) = \begin{cases} b & \text{αν } n < 2 \\ 2T(n/2) + bn & \text{αν } n \geq 2 \end{cases}$$

- Μπορούμε επομένως να αναλύσουμε τον τρέχων χρόνο της συγχωνευτικής ταξινόμησης από την εύρεση μιας **λύσης κλειστής μορφής** της παραπάνω εξίσωσης.
  - Δηλαδή, μία λύση που έχει ***T(n)*** μόνο στην αριστερή πλευρά

Επαναληπτική Αντικατάσταση

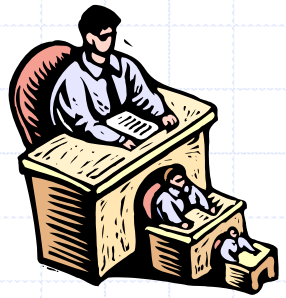


- Στην επαναληπτική αντικατάσταση, ή "plug-and-chug," τεχνική, εφαρμόζουμε επαναληπτικά την επαναληπτική αντικατάσταση σε αυτό και βλέπουμε αν μπορούμε να βρούμε ένα πρότυπο:

$$
\begin{aligned}
T(n) &= 2T(n/2) + bn \\
&= 2(2T(n/2^2)) + b(n/2)) + bn \\
&= 2^2 T(n/2^2) + 2bn \\
&= 2^3 T(n/2^3) + 3bn \\
&= 2^4 T(n/2^4) + 4bn \\
&= \ldots \\
&= 2^i T(n/2^i) + ibn
\end{aligned}
$$

- Σημειώστε ότι η περίπτωση, T(n)=b, εμφανίζεται όταν $2^i=n$. Δηλαδή, i = log n.
- Έτσι,
- Έτσι, T(n) είναι O(n log n).

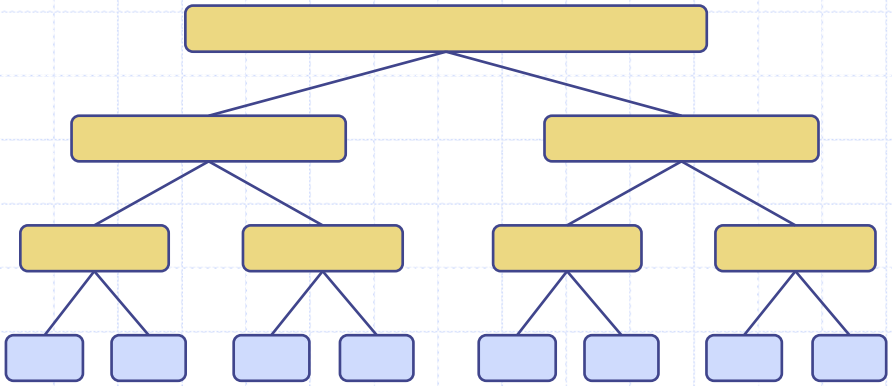
Αναδρομικά δένδρα



- Σχεδιάστε το αναδρομικό δέντρο για την αναδρομική σχέση και αναζητήστε ένα πρότυπο:

$$T(n) = \begin{cases} b & \text{if } n < 2 \\ 2T(n/2) + bn & \text{if } n \geq 2 \end{cases}$$

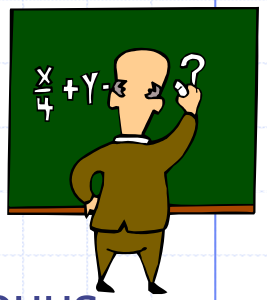
depth	T's	size	time
0	1	n	bn
1	2	$n/2$	bn
i	2^i	$n/2^i$	bn
…	…	…	…



Συνολικός χρόνος = $bn + bn \log n$

(Τελευταίο επίπεδο συν όλα τα προηγούμενα)

Η Μέθοδος της αντικατάστασης



- Στην μέθοδο της αντικατάστασης (guess-and-test), υποθέτουμε μια λύση κλειστής μορφής και έπειτα προσπαθούμε να αποδείξουμε ότι είναι αληθείς με απαγωγή:

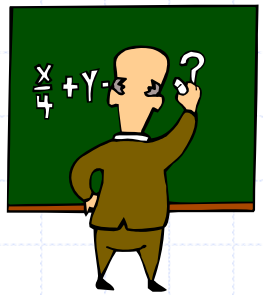
$$T(n) = \begin{cases} b & \text{if } n < 2 \\ 2T(n/2) + bn\log n & \text{if } n \geq 2 \end{cases}$$

- Υπόθεση: T(n) < cn log n.

$$\begin{aligned} T(n) &= 2T(n/2) + bn\log n \\ &= 2(c(n/2)\log(n/2)) + bn\log n \\ &= cn(\log n - \log 2) + bn\log n \\ &= cn\log n - cn + bn\log n \end{aligned}$$

- Λάθος: δεν μπορούμε να κάνουμε την τελευταία αυτή γραμμή να είναι μικρότερη από cn log n

Η μέθοδος της αντικατάστασης, Τμήμα 2



- Ανάκληση την αναδρομικής σχέσης:

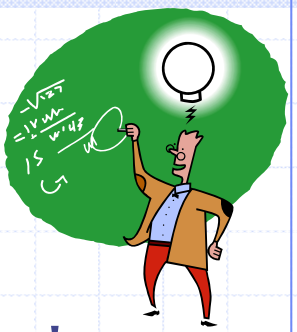
$$T(n) = \begin{cases} b & \text{if } n < 2 \\ 2T(n/2) + bn\log n & \text{if } n \geq 2 \end{cases}$$

- Υπόθεση #2: T(n) < cn log^2 n.

$$\begin{aligned} T(n) &= 2T(n/2) + bn\log n \\ &= 2(c(n/2)\log^2(n/2)) + bn\log n \\ &= cn(\log n - \log 2)^2 + bn\log n \\ &= cn\log^2 n - 2cn\log n + cn + bn\log n \\ &\leq cn\log^2 n \end{aligned}$$

  - if c > b.
- Έτσι, T(n) είναι O(n log^2 n).
- Γενικά, η χρήση αυτής την μεθόδου, απαιτεί να έχετε μια καλή υπόθεση και να είστε καλοί στις αποδείξεις επαγωγής.

Κύρια Μέθοδος



- Αρκετές διαίρει και κυρίευε αναδρομικές σχέσεις έχουν την μορφή:

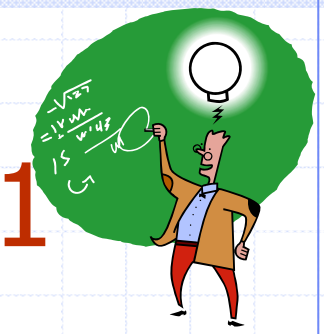
$$T(n) = \begin{cases} c & \text{if } n < d \\ aT(n/b) + f(n) & \text{if } n \geq d \end{cases}$$

- Το κύριο θεώρημα:

1. if $f(n)$ is $O(n^{\log_b a - \varepsilon})$, then $T(n)$ is $\Theta(n^{\log_b a})$
2. if $f(n)$ is $\Theta(n^{\log_b a} \log^k n)$, then $T(n)$ is $\Theta(n^{\log_b a} \log^{k+1} n)$
3. if $f(n)$ is $\Omega(n^{\log_b a + \varepsilon})$, then $T(n)$ is $\Theta(f(n))$,

   provided $af(n/b) \leq \delta f(n)$ for some $\delta < 1$.

Κύρια Μέθοδος, Παράδειγμα 1

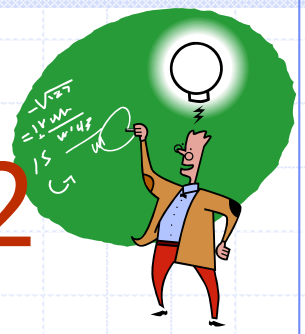


- Η μορφή: $$T(n) = \begin{cases} c & \text{if } n < d \\ aT(n/b) + f(n) & \text{if } n \geq d \end{cases}$$
- Το κύριο θεώρημα:
  1. if $f(n)$ is $O(n^{\log_b a - \varepsilon})$, then $T(n)$ is $\Theta(n^{\log_b a})$
  2. if $f(n)$ is $\Theta(n^{\log_b a} \log^k n)$, then $T(n)$ is $\Theta(n^{\log_b a} \log^{k+1} n)$
  3. if $f(n)$ is $\Omega(n^{\log_b a + \varepsilon})$, then $T(n)$ is $\Theta(f(n))$, provided $af(n/b) \leq \delta f(n)$ for some $\delta < 1$.
- Παράδειγμα:

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

Λύση: $\log_b a = 2$, έτσι η περίπτωση 1 λέει $T(n)$ είναι $O(n^2)$.

Κύρια Μέθοδος, Παράδειγμα 2

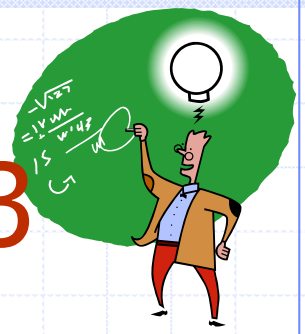


- Η μορφή: $$T(n) = \begin{cases} c & \text{if } n < d \\ aT(n/b) + f(n) & \text{if } n \geq d \end{cases}$$
- Το κύριο θεώρημα :
  1. if $f(n)$ is $O(n^{\log_b a - \varepsilon})$, then $T(n)$ is $\Theta(n^{\log_b a})$
  2. if $f(n)$ is $\Theta(n^{\log_b a} \log^k n)$, then $T(n)$ is $\Theta(n^{\log_b a} \log^{k+1} n)$
  3. if $f(n)$ is $\Omega(n^{\log_b a + \varepsilon})$, then $T(n)$ is $\Theta(f(n))$, provided $af(n/b) \leq \delta f(n)$ for some $\delta < 1$.
- Παράδειγμα:

$$T(n) = 2T(n/2) + n \log n$$

Λύση: $\log_b a = 1$, έτσι η περίπτωση 2 λέει T(n) είναι $O(n \log^2 n)$.

Κύρια Μέθοδος, Παράδειγμα 3

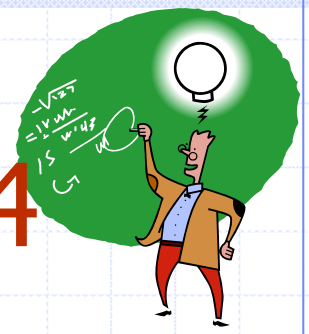


- Η μορφή: $$T(n) = \begin{cases} c & \text{if } n < d \\ aT(n/b) + f(n) & \text{if } n \geq d \end{cases}$$
- Το κύριο θεώρημα :
  1. if $f(n)$ is $O(n^{\log_b a - \varepsilon})$, then $T(n)$ is $\Theta(n^{\log_b a})$
  2. if $f(n)$ is $\Theta(n^{\log_b a} \log^k n)$, then $T(n)$ is $\Theta(n^{\log_b a} \log^{k+1} n)$
  3. if $f(n)$ is $\Omega(n^{\log_b a + \varepsilon})$, then $T(n)$ is $\Theta(f(n))$, provided $af(n/b) \leq \delta f(n)$ for some $\delta < 1$.
- Παράδειγμα:

$$T(n) = T(n/3) + n \log n$$

Λύση: $\log_b a = 0$, έτσι η περίπτωση 3 λέει T(n) είναι O(n log n).

Κύρια Μέθοδος, Παράδειγμα 4

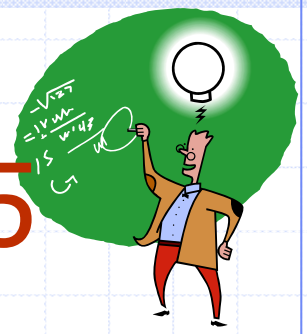


- Η μορφή: $$T(n) = \begin{cases} c & \text{if } n < d \\ aT(n/b) + f(n) & \text{if } n \geq d \end{cases}$$
- Το κύριο θεώρημα:
  1. if $f(n)$ is $O(n^{\log_b a - \varepsilon})$, then $T(n)$ is $\Theta(n^{\log_b a})$
  2. if $f(n)$ is $\Theta(n^{\log_b a} \log^k n)$, then $T(n)$ is $\Theta(n^{\log_b a} \log^{k+1} n)$
  3. if $f(n)$ is $\Omega(n^{\log_b a + \varepsilon})$, then $T(n)$ is $\Theta(f(n))$, provided $af(n/b) \leq \delta f(n)$ for some $\delta < 1$.
- Παράδειγμα:

$$T(n) = 8T(n/2) + n^2$$

Λύση: $\log_b a = 3$, έτσι η περίπτωση 1 λέει T(n) είναι $O(n^3)$.

Κύρια Μέθοδος, Παράδειγμα 5

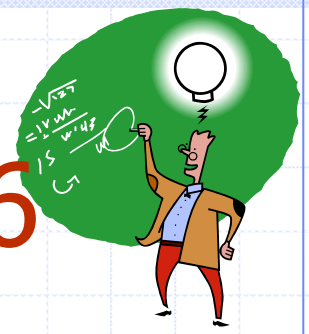


- Η μορφή: $$T(n) = \begin{cases} c & \text{if } n < d \\ aT(n/b) + f(n) & \text{if } n \geq d \end{cases}$$
- Το κύριο θεώρημα:
  1. if $f(n)$ is $O(n^{\log_b a - \varepsilon})$, then $T(n)$ is $\Theta(n^{\log_b a})$
  2. if $f(n)$ is $\Theta(n^{\log_b a} \log^k n)$, then $T(n)$ is $\Theta(n^{\log_b a} \log^{k+1} n)$
  3. if $f(n)$ is $\Omega(n^{\log_b a + \varepsilon})$, then $T(n)$ is $\Theta(f(n))$, provided $af(n/b) \leq \delta f(n)$ for some $\delta < 1$.
- Παράδειγμα:

$$T(n) = 9T(n/3) + n^3$$

Λύση: $\log_b a = 2$, έτσι η περίπτωση 3 λέει T(n) είναι $O(n^3)$.

Κύρια Μέθοδος, Παράδειγμα 6

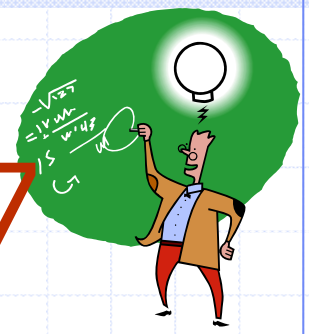


- Η μορφή: $$T(n) = \begin{cases} c & \text{if } n < d \\ aT(n/b) + f(n) & \text{if } n \geq d \end{cases}$$
- Το κύριο θεώρημα:
  1. if $f(n)$ is $O(n^{\log_b a - \varepsilon})$, then $T(n)$ is $\Theta(n^{\log_b a})$
  2. if $f(n)$ is $\Theta(n^{\log_b a} \log^k n)$, then $T(n)$ is $\Theta(n^{\log_b a} \log^{k+1} n)$
  3. if $f(n)$ is $\Omega(n^{\log_b a + \varepsilon})$, then $T(n)$ is $\Theta(f(n))$, provided $af(n/b) \leq \delta f(n)$ for some $\delta < 1$.
- Παράδειγμα:

$$T(n) = T(n/2) + 1$$ (διαδική αναζήτηση)

Λύση: $\log_b a = 0$, έτσι η περίπτωση 2 λέει T(n) είναι O(log n).

Κύρια Μέθοδος, Παράδειγμα 7

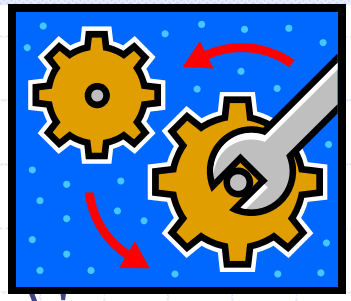


- Η μορφή: $$T(n) = \begin{cases} c & \text{if } n < d \\ aT(n/b) + f(n) & \text{if } n \geq d \end{cases}$$
- Το κύριο θεώρημα:
  1. if $f(n)$ is $O(n^{\log_b a - \varepsilon})$, then $T(n)$ is $\Theta(n^{\log_b a})$
  2. if $f(n)$ is $\Theta(n^{\log_b a} \log^k n)$, then $T(n)$ is $\Theta(n^{\log_b a} \log^{k+1} n)$
  3. if $f(n)$ is $\Omega(n^{\log_b a + \varepsilon})$, then $T(n)$ is $\Theta(f(n))$, provided $af(n/b) \leq \delta f(n)$ for some $\delta < 1$.
- Παράδειγμα:

$$T(n) = 2T(n/2) + \log n$$ (κατασκευή heap)

Λύση: $\log_b a = 1$, έτσι η περίπτωση 1 λέει T(n) είναι O(n).

Επαναληπτική "Απόδειξη" του κύριου θεωρήματος

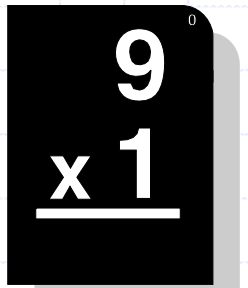


- Χρησιμοποιώντας την επαναληπτική αντικατάσταση, ελέγχουμε αν μπορούμε να βρούμε ένα πρότυπο:

$$
\begin{aligned}
T(n) &= aT(n/b) + f(n) \\
&= a(aT(n/b^2)) + f(n/b)) + bn \\
&= a^2T(n/b^2) + af(n/b) + f(n) \\
&= a^3T(n/b^3) + a^2 f(n/b^2) + af(n/b) + f(n) \\
&= \ldots \\
&= a^{\log_b n}T(1) + \sum_{i=0}^{(\log_b n)-1} a^i f(n/b^i) \\
&= n^{\log_b a}T(1) + \sum_{i=0}^{(\log_b n)-1} a^i f(n/b^i)
\end{aligned}
$$

- Διακρίνουμε τρεις περιπτώσεις, όπως:
  - Ο πρώτος όρος είναι κυρίαρχος
  - Κάθε μέρος του αθροίσματος είναι εξίσου κυρίαρχο
  - Το άθροισμα είναι μια γεωμετρική σειρά.

Πολλαπλασιασμός Ακεραίων



- Αλγόριθμος: Πολλαπλασιασμός δύο n-bit ακεραίων I και J.
  - Βήμα διαίρεσης: Χωρίζει το I και J σε υψηλή και χαμηλή τάξη bits.

$$I = I_h 2^{n/2} + I_l$$

$$J = J_h 2^{n/2} + J_l$$

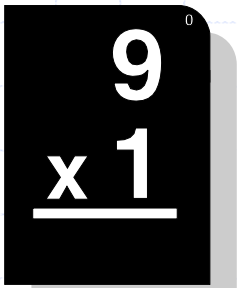
  - Εμείς έπειτα μπορούμε να ορίσουμε το I*J από τον πολλαπλασιασμό των τμημάτων και την πρόσθεση τους:

$$I * J = (I_h 2^{n/2} + I_l) * (J_h 2^{n/2} + J_l)$$

$$= I_h J_h 2^n + I_h J_l 2^{n/2} + I_l J_h 2^{n/2} + I_l J_l$$

  - Έτσι, T(n) = 4T(n/2) + n, που συνεπάγεται T(n) είναι $O(n^2)$.
  - Αλλά αυτός δεν είναι καλύτερος από τον αλγόριθμο που θα μάθετε στο μάθημα.

Ένας ταχύτερος αλγόριθμος πολλαπλασιασμού ακεραίων



- Αλγόριθμος: Πολλαπλασιασμός δύο n-bit ακεραίων I και J.
  - Βήμα διαίρεσης: Χωρίζει το I και J σε υψηλή και χαμηλή τάξη bits.

$$I = I_h 2^{n/2} + I_l$$

$$J = J_h 2^{n/2} + J_l$$

  - Παρατηρούμε ότι υπάρχει ένας διαφορετικός τρόπος να πολλαπλασιαστούν τα μέρη:

$$I * J = I_h J_h 2^n + [(I_h - I_l)(J_l - J_h) + I_h J_h + I_l J_l]2^{n/2} + I_l J_l$$

$$= I_h J_h 2^n + [(I_h J_l - I_l J_l - I_h J_h + I_l J_h) + I_h J_h + I_l J_l]2^{n/2} + I_l J_l$$

$$= I_h J_h 2^n + (I_h J_l + I_l J_h)2^{n/2} + I_l J_l$$

  - Έτσι, T(n) = 3T(n/2) + n, που υπονοεί T(n) είναι $O(n^{\log_2 3})$, από το κύριο θεώρημα
  - Έτσι, T(n) είναι $O(n^{1.585})$.

Τέλος Ενότητας

Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «Ανοικτά Ακαδημαϊκά Μαθήματα στο Πανεπιστήμιο Κρήτης» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο

ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ ΚΑΙ ΘΡΗΣΚΕΥΜΑΤΩΝ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ

ΕΣΠΑ 2007-2013
πρόγραμμα για την ανάπτυξη
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ

Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Σημειώματα

Σημείωμα αδειοδότησης

- Το παρόν υλικό διατίθεται με τους όρους της άδειας χρήσης Creative Commons Αναφορά, Μη Εμπορική Χρήση, Όχι Παράγωγο Έργο 4.0 [1] ή μεταγενέστερη, Διεθνής Έκδοση. Εξαιρούνται τα αυτοτελή έργα τρίτων π.χ. φωτογραφίες, διαγράμματα κ.λ.π., τα οποία εμπεριέχονται σε αυτό και τα οποία αναφέρονται μαζί με τους όρους χρήσης τους στο «Σημείωμα Χρήσης Έργων Τρίτων».

[1] http://creativecommons.org/licenses/by-nc-nd/4.0/

- Ως Μη Εμπορική ορίζεται η χρήση:
  - που δεν περιλαμβάνει άμεσο ή έμμεσο οικονομικό όφελος από την χρήση του έργου, για το διανομέα του έργου και αδειοδόχο
  - που δεν περιλαμβάνει οικονομική συναλλαγή ως προϋπόθεση για τη χρήση ή πρόσβαση στο έργο
  - που δεν προσπορίζει στο διανομέα του έργου και αδειοδόχο έμμεσο οικονομικό όφελος (π.χ. διαφημίσεις) από την προβολή του έργου σε διαδικτυακό τόπο

- Ο δικαιούχος μπορεί να παρέχει στον αδειοδόχο ξεχωριστή άδεια να χρησιμοποιεί το έργο για εμπορική χρήση, εφόσον αυτό του ζητηθεί.

Σημείωμα Αναφοράς

Διατήρηση Σημειωμάτων

Οποιαδήποτε αναπαραγωγή ή διασκευή του υλικού θα πρέπει να συμπεριλαμβάνει:

- το Σημείωμα Αναφοράς
- το Σημείωμα Αδειοδότησης
- τη δήλωση Διατήρησης Σημειωμάτων
- το Σημείωμα Χρήσης Έργων Τρίτων (εφόσον υπάρχει)

μαζί με τους συνοδευόμενους υπερσυνδέσμους.