

# Δομές δεδομένων

- Sorting & Searching, Springer Verlag, 1984
- Intro to Algorithms, Cormen, Leirerson, Rivest, Stern, MIT PRESS, 200.
- ΠΔ Μποζάνης, Δομές Δεδομένων

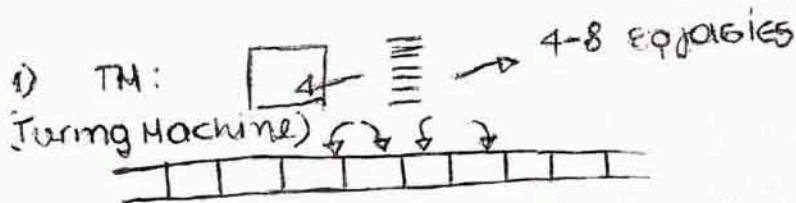
Για οποιαδήποτε απορία : panagis@ceid.upatras.gr

2 ασκήσεις → 1 μέχρι τα χριστούγεννα  
→ 1 για την εφεταστική (μετράει)

- 1) Αλγόριθμοι ταξινόμησης
- 2) Αναζήτηση
- 3) Δέντρα

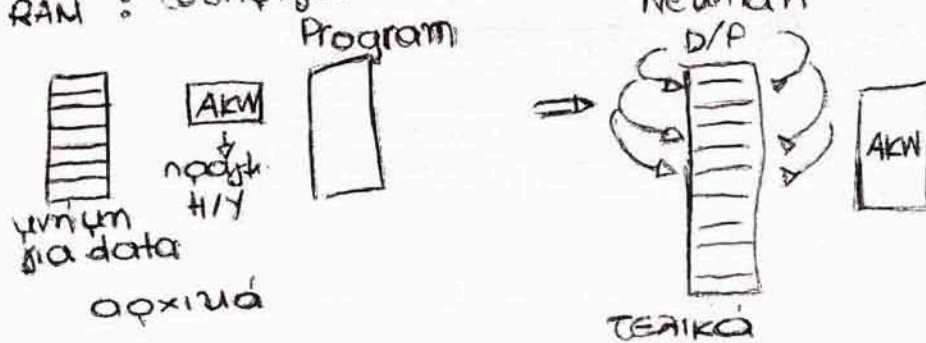
- Μοντέλα Υπολογισμού -

3/11/2003



Υπολογιστής : ορισμός κάποιας μηχανής και μετά υλοποίηση.

2) RAM : (στηρίζεται στο πυρηνό έχει μειωθεί ο χρόνος + έχει αυξηθεί η ταχύτητα)



□  $1\text{cm}^2$  200 transistors (1976)

□  $1\text{cm}^2$  1M transistors (1989) → Siemens

□  $1\text{cm}^2$  1 τρις transistors (1995-1997) → Philips

3) Φωτονικοί Η/Υ (εξανδρασμοί για επιπλέον στο φως)

3) Βιολογικοί Η/Υ ( - II - - II - - II - κυτταρα )

10	1
≤ 1.5	35-5V

Leibniz  
1675

(TM)<sup>3</sup> ≡ RAM

Είναι ζήτημα πώς να χρησιμοποιήσουμε RAM

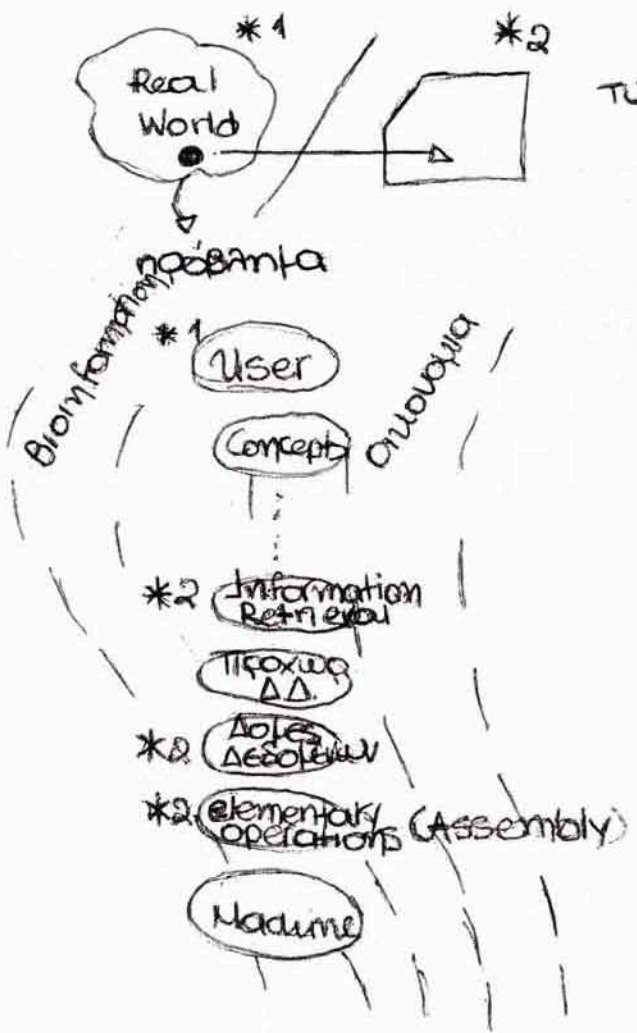
3) PM (Printer Machine)

4) PPM (Pure Printer Machine)

Searching is  $\log \log n = \left( \log \left( \log 2^{35} \right) \right) = 5.5$

$2^5 \cdot 2^{10} \cdot 2^{10} \cdot 2^{10}$   
30.000.000.000 Δισεκατ. αριθμοί  
 $\hookrightarrow 2^{35}$

Distributed  $\Rightarrow$  Δεν είναι όλα τα κομμάτια στον ίδιο server.  
Είναι διασκορπισμένα σε διάφορους servers.



Τυποποιημένος Κόσμος

Bio	Info
DNA 4 Proteins 50	Strings
Protein Clustering	DB-Clusters
Drug Design	Computation Geometry

ΔΙΕΠΙΣΤΗΜΟΝΙΚΟΤΗΤΑ

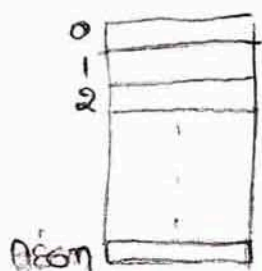
## Ματέλα Μηχανής

1) RAM (RANDOM ACCESS MACHINE)  $\rightarrow$  MIO 16XU90 MOUTERO

↳ Απονομοθέτηση ευδοκία των Η/Υ

υποστηρίζει στοιχειώδεις  
ηράξεις

- 1 ACC
- 3 idx registers



Διότιτα : Ευτοφειδης ο  
υποαρχιεπος διεκωνων

ΕΥΤΟΛΕΣ + ΣΕΒΑΣΜΕΝΑ : ΧΑΡΑΤΕ ΓΕ ΥΙΑ ΘΕΩΝ ΚΥΝΗΤΩΝ

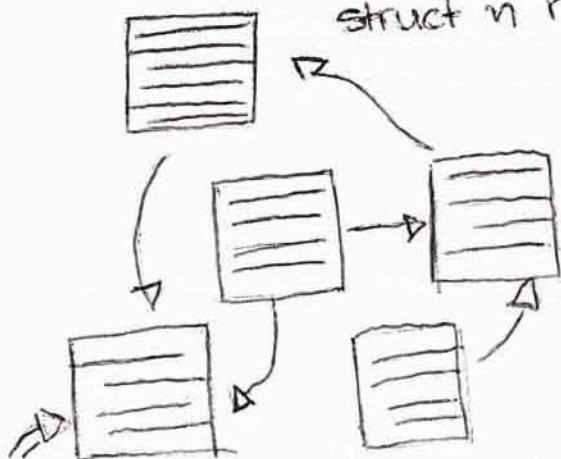
→ arrays pass you multidimensional arrays

2) PM (PINTER MACHINE) → 210 κομμοί αλγορίθμοι

↳ είναι ένα να υπάρχει τIS & μνήμη τIS RAM + να τIS διακοπών στο xwp

Δεν έχει την δυνατότητα του υψώσεως και διεύθυνσης

struct n records;



MAX 5

$$m = AGI; l = j = 1$$

2.  $\text{unlike } (i \neq n) \text{ do } S \rightarrow M$

3. if  $\text{CAGT} \geq m) \{ \rightarrow 1-1$

4.  $m = A[j]$ ;  $z \rightarrow m$

$$5. \quad i = \sqrt{-1}$$

$\rightarrow n-1$

$$j = j+1$$

11. 10-1-1950

$$A[1 \dots n]$$

• Πολλαπλότητα αριθμού σταθμών των  
βημάτων

$$\hookrightarrow T(u) = 3M - 2 + N + 1$$

4) = 511 - 2 + 1111  
- περιπτώσεις πολυωλοκοπότητας -

(a)  $\Pi \times \Pi \rightarrow \chi \epsilon \rho \acute{o} \tau \epsilon \rho \eta \eta \epsilon \rho \acute{o} \tau \omega \nu$ ,  $\mu \epsilon \gamma \iota \sigma \tau \acute{o} \varsigma$   
 $\alpha \rho \iota \theta \mu \acute{o} \varsigma \beta \eta \mu \alpha \tau \omega \nu$ .  $5n-2+1 = 5n-1$

$\alpha\rho\theta\eta\sigma\ \beta\eta\mu\alpha\tau\omega\nu.$   
 $\hookrightarrow \text{A.x. } \pi \times \pi = 3\eta - 2 + 2\eta - 2 + 1 = 5\eta - 3$

(2)  $\text{PNO} \vdash K$  (στη μορφή)

$$(6) \prod_{\text{Gegon opou}} \text{PMO} \vdash \prod_{\text{Gegon opou}} P_k \left[ \prod_{\text{Gegon opou}} \text{PMO} \right] \equiv \prod_{\text{Gegon opou}} \text{PMO}(n)$$
$$T(n)$$
$$\sum_{i=1}^n \geq 3n + \ln n + 12$$
$$5A = 1$$

KK



γ) Επιμερίσμεν τον λογαριθμό (Amortized)  
 η συνολικός χρόνος

$O_{p1}, O_{p2}, \dots, O_{pn}$   
 $C(O_{p1}), C(O_{p2}), \dots, C(O_{pn})$   
 (κόστος)

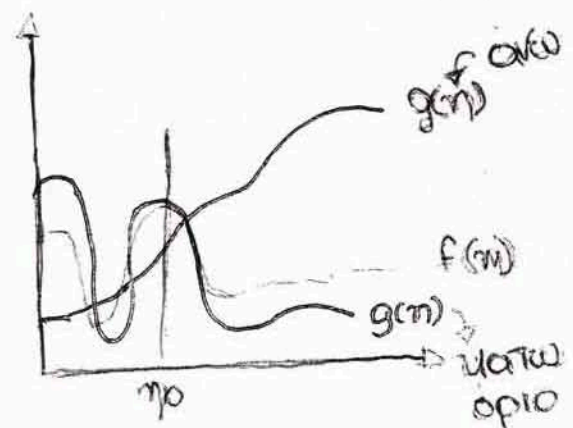
$$EK = \frac{\sum_{i=1}^n C(O_{pi})}{n}$$

ΑΣΥΜΤΩΤΙΚΟΙ ΣΥΜΒΟΛΙΣΜΟΙ

Πάνω όριο:

(α) Ομικρόν

$f(n) = O(g(n)) \Leftrightarrow \exists c > 0, n_0 \Rightarrow \forall n > n_0 \Rightarrow f(n) \leq c \cdot g(n)$



ανω όριο

$$T(n) = 5n - 3 = O(n)$$

MAX

$$T(n) = 15 \log n = O(\log n)$$

$$T(n) = 5n^2 + 4400n + 10 = O(n^2)$$

$$T(n) = 3n + 5 \ln n = O(n)$$

$$3n \log n + 10 = O(n \log n)$$

$$T(n) = 2^{100} = O(1) \quad \text{4 σταθερά}$$

(β) κάτω όριο:

$f(n) = \Omega(g(n))$  όταν  $\exists c > 0$   
 υπάρχει  $n_0 \mid \forall n > n_0 \Rightarrow f(n) \geq c \cdot g(n)$

λόγο μεγάλου και να είναι, είναι ανεξαρτητή του n από ένα σταθερά

δ) Ακριβής Ευρίσκμεν

$$f(n) = \Theta(g(n))$$

$$\left. \begin{array}{l} f(n) = O(n) \\ f(n) = \Omega(n) \end{array} \right\} f(n) = \Theta(g(n))$$

(μεγαλύτερη)

π.χ  $\left. \begin{array}{l} f(n) \leq 5n \\ f(n) \geq 0.5n \end{array} \right\} \Theta(n)$

(μικρότερη)

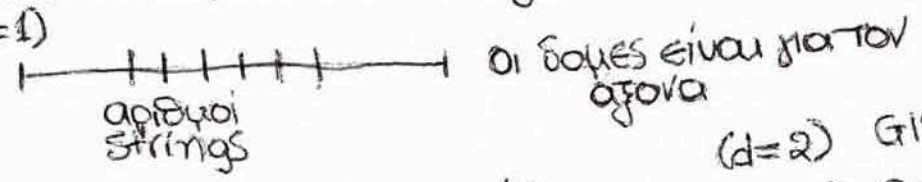
\*Της ίδιας τάξης μεγέθους\*

\*Είναι γραμμική ως προς το μέγεθος εισόδου\*

4/11/2003

Data Manipulation στον άξονα

(d=1)



XML → specification για τα ομάδων ως data

BIOML → Bio data

XML → δηλώση διοίκησης ιδιωτική TI-

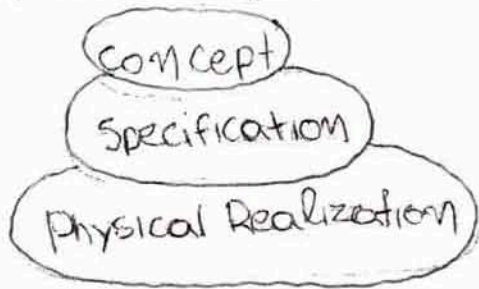
(d=2) GIS, GUI } παρουσίασες Δομές Δεδομένων

(d=3) Graphics

\* οι δ.δ. συμπίζουν παρουσίασες δ.δ. \*

(d>3) DB (Data Base)

Για data base πρέπει να κάνουμε 3 πράγματα:



searching - exact  $\rightarrow$  το μαθημα Δ.Δ.

searching - approximate  $\rightarrow$  για Web Cby similarity, not exact

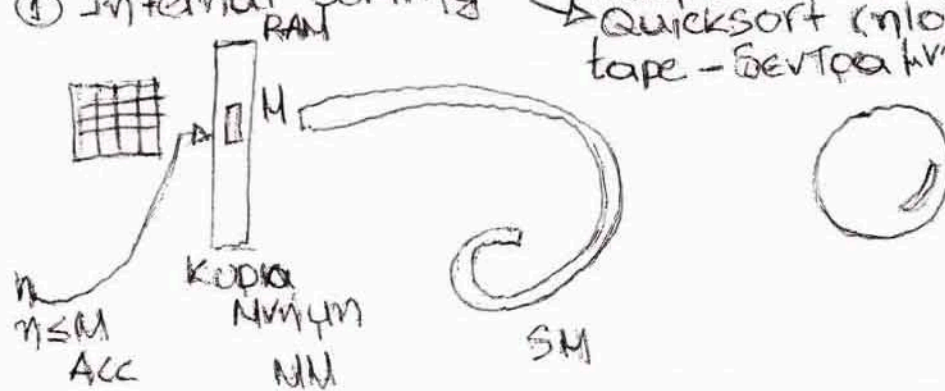
Hashing  $\Leftrightarrow$  Πρόβλημα του λέξικού  
(Διαμοίρασμα)

Κεφ 1 Διατάξη στοιχείων (Sorting)

οποιοι οι αριθμοί χωρούν σε μια θέση memorys και σε μια ενδιαιρέσει το μήκος τους

Handbook of Theoretical Computer Science, MIT PRESS, Elsevier, 1999

① Internal Sorting  $\rightarrow$  Bubble sort ( $n^2$ )  
 $\rightarrow$  Heapsort ( $n \log n$ )  
 $\rightarrow$  Quicksort ( $n \log n$ )  
tape - δεν τρεκνίμει



② External Sorting  
(δεν χωράνε όλα στη μνήμη)  $\rightarrow n \gg M$

Bubblesort

Παραδείγματα:

input  $\left. \begin{array}{l} 4, 3, 8, 6, 2, 5, 7 \\ 3, 4, 6, 2, 5, 7, 8 \\ 2, 3, 4, 5, 6, 7, 8 \end{array} \right\} n \text{ cases}$

Ποιοσυνεκτικότητα  $\rightarrow n^2 \text{ ή } \frac{n^2}{2} \text{ ή } \frac{n^2}{4}$

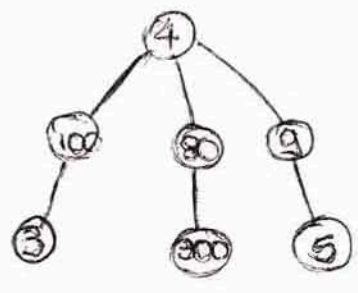


# Heapsort

Heap =



= περιη διατάξη



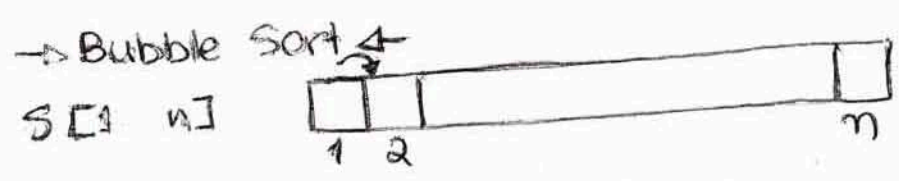
10/11/2003

Πρόβλημα Ταξινότησης : Input  $\rightarrow S = \langle S_1, S_2, \dots, S_n \rangle$   
Output  $\rightarrow S' = \langle S'_1, S'_2, \dots, S'_n \rangle$   
( $S'_1 \leq S'_2 \leq \dots \leq S'_n$ )

Αλγόριθμοι : (a) δύο συγκρίσεις { Bubble Sort  
Insertion Sort  $\Rightarrow \Omega(\log n)$   
Heap Sort  
Merge Sort  
Quick Sort

(b) (+) { radix sort  
bug sort  
counting sort

(c) extended sort



ix

4	3	8	6	2	5	7
3	4	8	6	2	5	7
3	4	8	6	2	5	7
3	4	6	8	2	5	7
3	4	6	2	8	5	7
3	4	6	2	5	8	7
3	4	6	2	5	7	8
3	4	2	6	5	7	8
...						
2	3	4	5	6	7	8



# Bubblesort(S,n) {

```

up ← n;
while up > 1 {
  j ← 0;
  for i = 1 to up - 1 {
    if S[i] > S[i+1] {
      swap(S[i], S[i+1]);
      i++;
    }
  }
  if j == 0 break;
  up--;
}

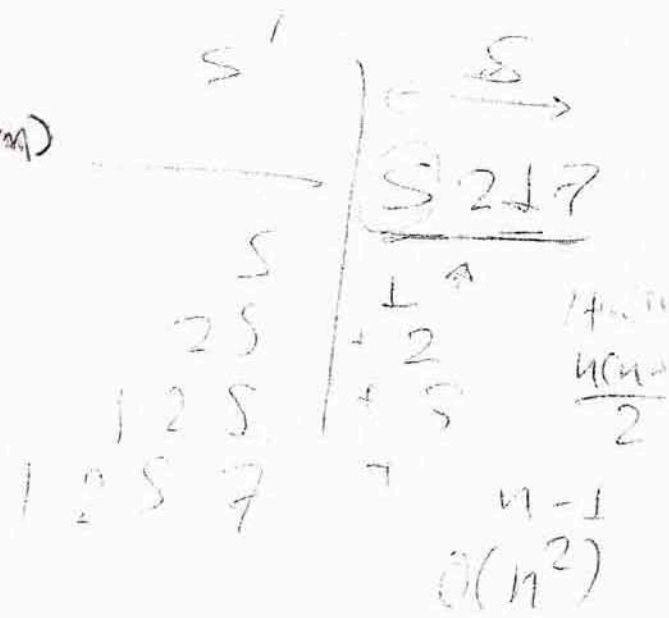
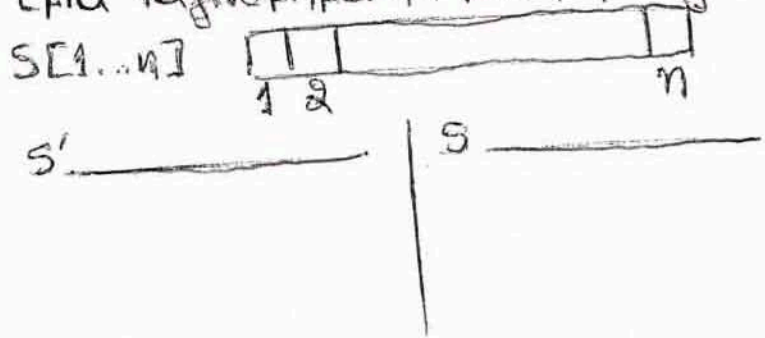
```

Πομπλοκότητα =  $n^2$

while  
 $(n-1) + (n-2) + \dots + (1)$  [αθροίζουμε τα κοστ]  $\rightarrow$   
 $1 + 2 + \dots + n = \frac{n(n+1)}{2}$   
 $O(n^2)$

## Insertion Sort ←

(για ταξινοχητέ n + μια τη ταξινοχητέ n)



π.χ

7	10	30	<del>22</del>	<del>9</del>	<del>22</del>
7	9	10	30		
7	9	10	22	30	

κόστος  $\Rightarrow n^2$

→ Η χειρότερη περίπτωση είναι όταν η λίστα είναι τελείως αταξινοχητή

## Insertion(S,n) {

```

for j ← 2 to n {
  k ← S[j];
  i ← j - 1;
  while i > 0 && k < S[i] {
    S[i+1] ← S[i];
    i--;
  }
  S[i+1] ← k;
}

```

→ Heap Sort →

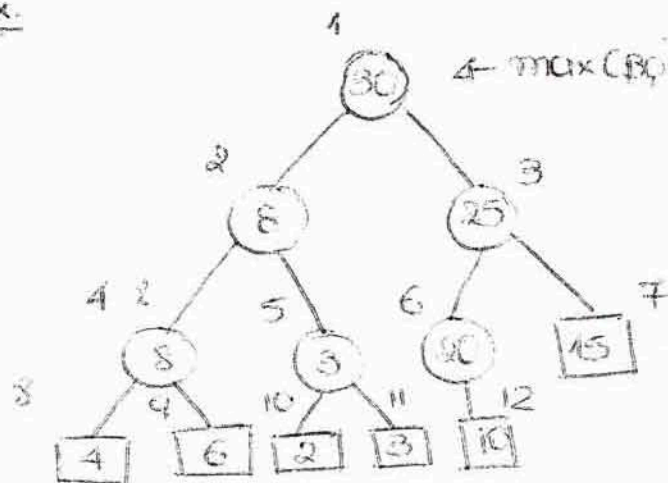
$S[1..n]$



balanced binary heap:

- i)  $\forall u \rightarrow \text{τιμή}(u) = S[i]$
- ii)  $\forall u \rightarrow \text{τιμή}(\text{πατέρα}(u)) \geq \text{τιμή}(u)$
- iii)  $\forall u \rightarrow 0 \text{ ή } 2 \text{ παιδιά εκτός από τα φύλλα}$

π.χ.

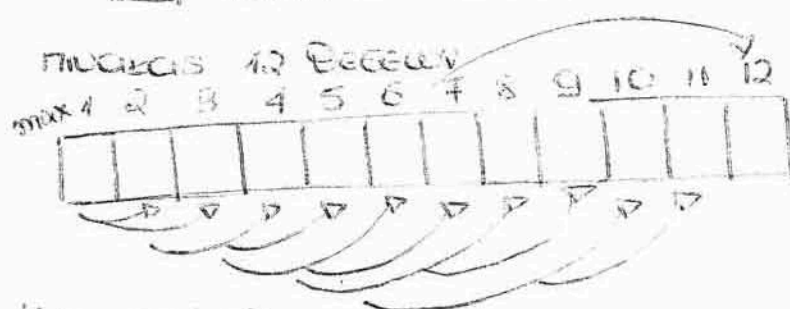


$\leftarrow \text{max (βρίσκεται στο ρίζα)}$

$$\text{αριθμός}(u) = \left\lceil \frac{\text{αριθμός}(l)}{2} \right\rceil = \left\lceil \frac{\text{αριθμός}(r)}{2} \right\rceil$$

○ → εσωτερικά

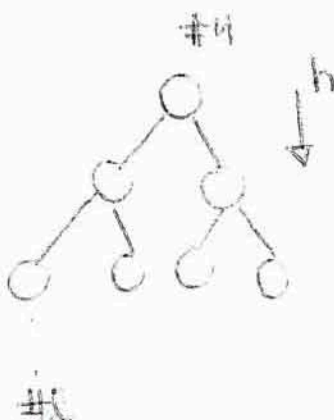
□ → φύλλα



~~Αν έχουμε την τιμή του αριθμού του κόμβου, μπορούμε να βρούμε τον αριθμό του πατέρα του.~~

HeapSort(s)

Build-heap(s),  $(n-1)$   $O(n)$   
 $j \leftarrow n$   $O(n \log n)$   
 for  $i \leftarrow n$  to 2  
     swap( $S[i], S[j]$ )  $O(n \log n)$   
      $j++$   
     Heapify(i);



$$\#n = \#(i-1)$$

$$H(n) = \#n \text{ σε ύψος } h$$

$$\#(h) \leq \frac{n}{2^{h+1}}$$

$$O(n) + O(n \log n)$$

$$\text{αριθμός} \in \begin{matrix} h=1 \\ \swarrow \quad \searrow \\ \text{ } \end{matrix} \quad \#n = \#(i-1)$$

$$h \quad \#n_h = \#(i-1)$$

$$h+1 \quad \#n_{h+1} = \#n_h + \#r_h$$

$$\#r_{h+1} = 2\#r_h$$



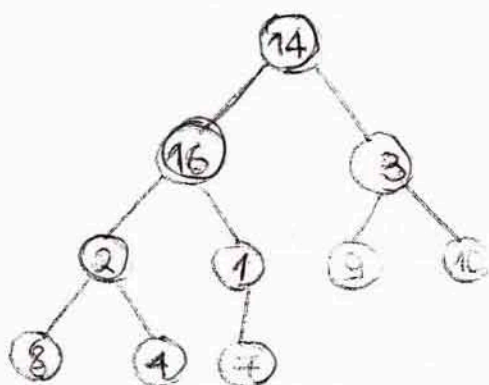
$$C_{\text{Build-heap}} = \sum_{i=1}^n O(h) = \sum_{h=1}^{\lceil \log n \rceil} H(h) \cdot O(h) \leq O\left[\sum_{h=1}^{\log n} \frac{n}{2^{h+1}} \cdot h\right] = O\left(n \sum_{h=1}^{\log n} \frac{h}{2^{h+1}}\right)$$

$$= O(n) \quad \left(\text{ερώση } \sum_{x=0}^{\infty} \frac{x}{x+1} = 2\right)$$

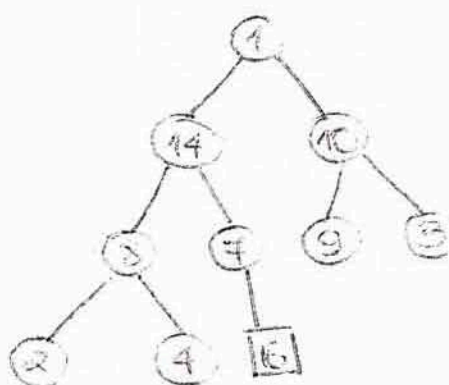
$$h = \lceil \log n \rceil$$

Πχ.

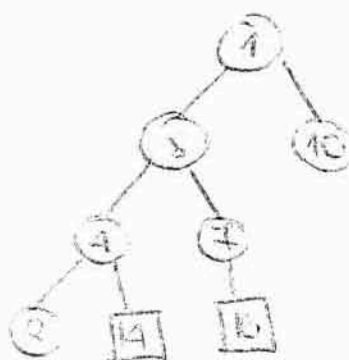
14 16 3 2 1 9 10 8 4 7  
14 16



→ To 1 ή 7 θα ανταλλάξω  
 To 3 ή 2 -||- -||-  
 To 9 ή 10 -||- -||-  
 To 16 ή 14 -||- -||-  
 (όπως το γεγονός ότι  
 τρέφει)

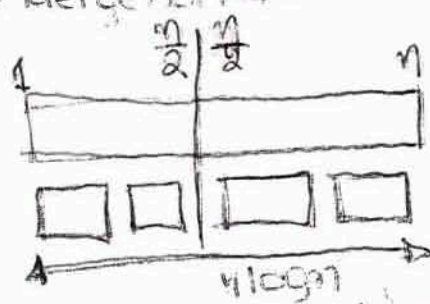


→ To 14 θα αλλάξει με  
 16

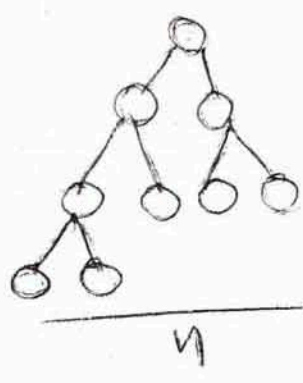


→ Όπως και να μετράμε

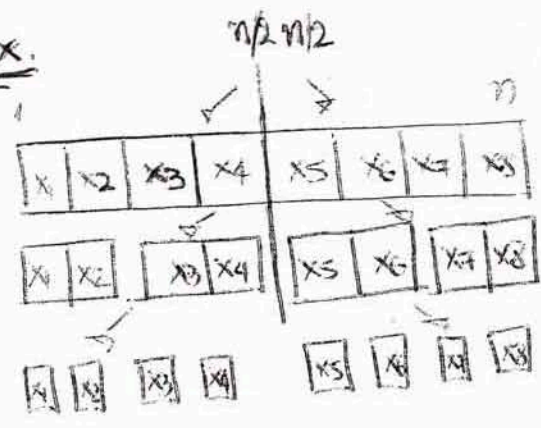
→ Merge sort



TO xwopjw GTM koon



PIX:



$n \log n$  eijkegees

To storagesse aww 2

mergesort( $S, n$ );

$S_1 = S[1 \dots n/2];$

$S_2 = S[n/2 + 1 \dots n];$

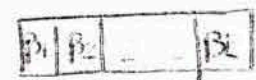
if  $|S_1| > 1$  mergesort( $S_1, n/2$ )

if  $|S_2| > 1$  mergesort( $S_2, n/2$ )

merge( $S_1, S_2$ );

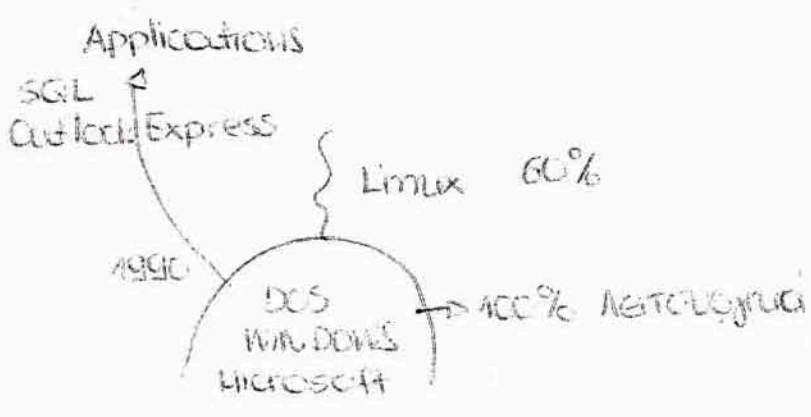
}

i



$a_i b_j - L$

→ H xeiqonegm wepiwneem aww  
 onaw jiwera n eijkegem awaawj  
 jian au gawupe to kdaen kae ju  
 to ewe ←



→ Συμπύκνωση

W.C.

4/1/2022

Bubblesort →  $1 \cdot n (n^2)$

Heapsort →  $1 \cdot n (n \log n)$  → Με τον όρο space εννοούμε το χώρο που καταλαμβάνει στη μνήμη.

Mergesort →  $2 \cdot n$

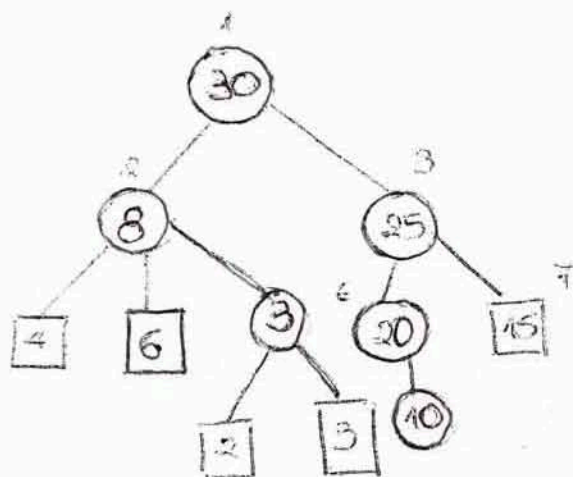
Quicksort →  $1 \cdot n (n^2)$  → Ο χώρος έχει πιο μεγάλη σημασία από ότι ο χρόνος

External Sorting

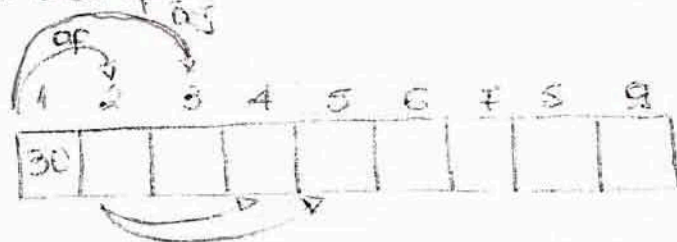
A-Sort (Adapting Sorting) → Αναπτύσσεται το sorting algorithm με το αν η μνήμη είναι μεγαλύτερη ή μικρότερη (space  $4 \cdot n$ )

Αναπαράσταση

Linear Searching

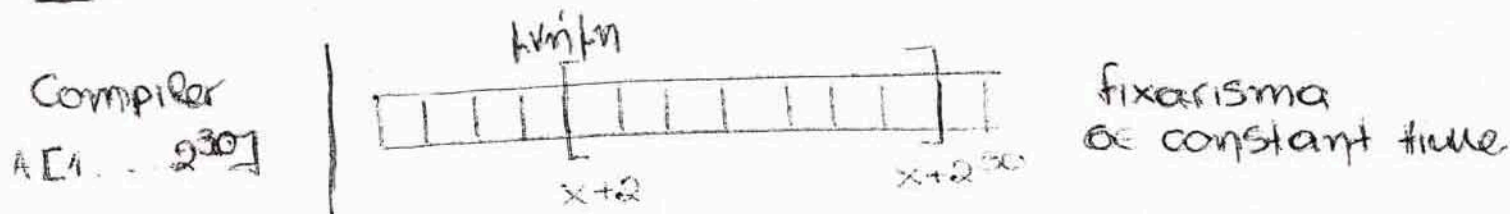


→ Δέντρο



Η P&P λογική βασίζεται στο pointer machine όπου καθεύλας έχει ένα local finger και επηρεάζει το περιβάλλον του.

Θεώρημα 8 Initialization σε constant time



Initialization =  $O(n)$

Χρησιμοποιούμε 2 stacks ώστε να βρίσκουμε το αν όρισα έχω και η αν ήταν πριν αργότερα.

→ Quicksort 4-

αποδοτική αναζήτηση

→ 30 5 10 8 41 60 75 43



χωρίζω σε δύο τμήματα ωαιρώντας ένα στοιχείο και  
βάζοντας στο 1<sup>ο</sup> όλα τα μικρότερα από αυτό και στο 2<sup>ο</sup>  
τα μεγαλύτερα

→ Στην χειρότερη περίπτωση, ο χρόνος που χρειάζεται  
είναι  $\frac{n^2}{2}$ .

> A-Sort &

Ε: ποσα γεγραμια είναι εκτός τάξης

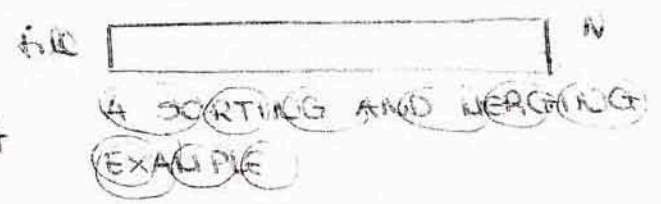
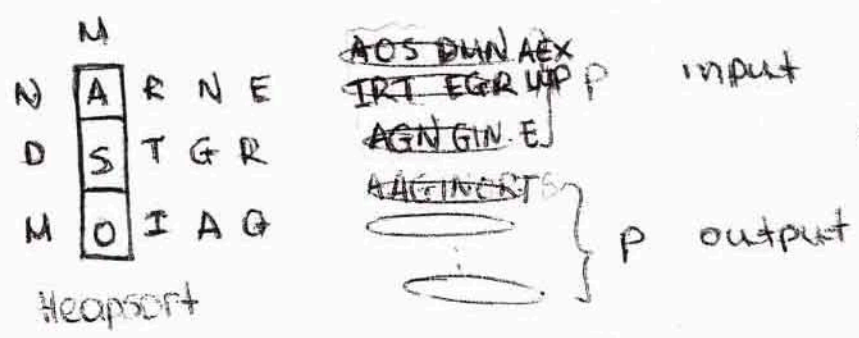
→ External Sorting &

18/11/2003

file μήκος N

Μνήμη -1- M, M < N

2p βοηθητικές ταινίες



→ Η μνήμη μπορεί να τα διατάξει.  
→ Computer to know the limits  
Αν ανα για heapsort χρειάζομεν heap, τότε είναι να  
παράγω files > 3. Computer βοηθάει από το know the limits

n = επόμενο στοιχείο

m = κορυφή Heap?

n ≥ m έστω

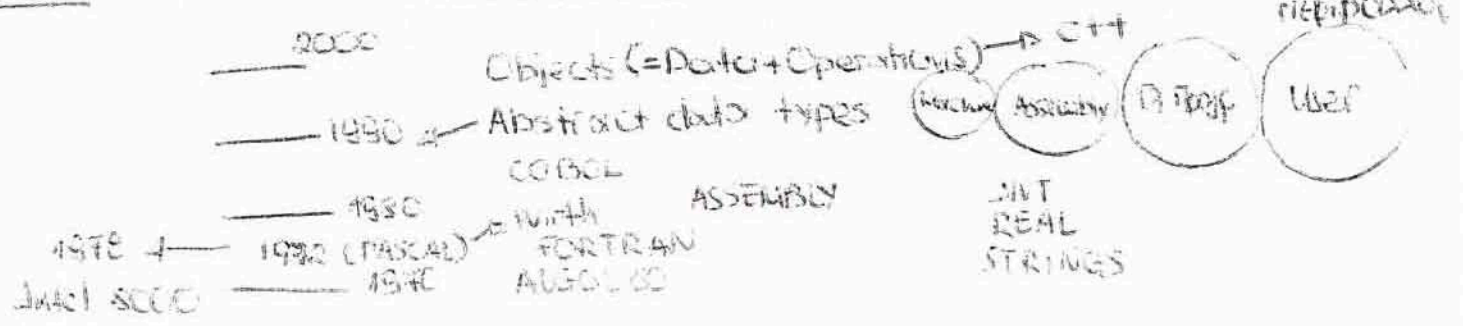
n < m n\* → βοηθάει από τα m κορυφαία

Εργασία

Παράδειγμα

Τύποι στοιχείων

Περιβάλλον



2 QuickSort (A1)

3 QuickSort (A2)

Μετά το Βήμα 1, θα έχουμε:



Για λόγους απλότητας, διαλέγουμε ως  $S$  το 1ο στοιχείο. Ανάγνουμε το  $i$ , συγκρίνουμε για να βρού στοιχείο μεγαλύτερο του  $S$  όπως το  $j$ , αλλά για μικρότερο. Στην συνέχεια, τα ανταλλάσσουμε.



Επανάληψη βήματος τα ίδια



Επανά τον quicksort για  $A_1$  και  $A_2$  (αναδρομική κλήση).

Ο αριθμός τελεστών όταν το μέγεθος γίνει 1 ή 0 ή 2.

Χειρότερη περίπτωση

$$QS(n) = n + 1 + \max_{1 \leq k \leq n} \{ QS(n-1) + QS(n-k) \}$$

αποτελεί το άνω όριο του πίνακα

$$|A_1| = k-1$$

$$|A_2| = n-k$$

Συγκρίνει τα 2 τελευταία στοιχεία να παραμείνουν γοφές

→ το μέγιστο κόστος είναι όταν δε είναι το καλύτερο

$$QS(0) = QS(1) = 0$$

Array $A[1 \dots n]$ RecordFileStack  $\rightarrow$    $\rightarrow$  Heap (LIFO)Queue  $\rightarrow$    $\rightarrow$  FIFOPriority Queue $\hookrightarrow$  επιλογή αναδιάρθρωσης (FIFO / LIFO / etc.)DBes TO  $\mu\kappa\rho\alpha\tau\epsilon\rho\omicron$ 

επιλογή TO - II -

	insert anywhere	min	delete min
Heap	$O(\log n)$	$O(1)$	$O(\log n)$
TS	$O(n \log n)$	$O(1)$	$O(1)$

d = αναδιάρθρωση

and to

 $\mu\kappa\rho\alpha\tau\epsilon\rho\omicron \rightarrow$   $\mu\kappa\rho\alpha\tau\epsilon\rho\omicron$  TS 4SQL server = DB + επιλογή (DB + κατάταξη transactions)Persistent SQL server = DB + history $\hookrightarrow$  Persistent Theory too Terjan

24/11/2003

 $\rightarrow$  Quicksort

Divide &amp; Conquer

Εστω A ένας αποσπασμένος πίνακας1.  $\chi\alpha\lambda\omega\upsilon\sigma\epsilon$  A εμβα S σε

$$A_1 = \{x \in A, x \leq S\} \quad /* \Delta\iota\alpha\lambda\upsilon\sigma\eta */$$

$$A_2 = \{x \in A, x > S\}$$



Το μέγιστο κόστος προκύπτει για  $k=1$  & η ακολουθία είναι ταξινομημένη.

$$QS(n) = n+1 + QS(0) + (n+1+1) + QS(n-2)$$

$$QS(0) + n-1 + QS(n-3) = \dots$$

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

Συμπέρασμα  
FIXIT

Ο Quicksort είναι πιο γρήγορος στη πράξη παρά η Merge Sort

Ο χρόνος  
Heapsort  
Mergesort

Ο χρόνος  
↳ Το υλογν του quicksort είναι πιο μικρό από το αντίστοιχο των Heapsort & Mergesort

Μέση Περίπτωση

1. Τα στοιχεία είναι όλα & διαφορετικά
2. Η κατανομή είναι ισοκύβητη

$$\bar{QS}(n) = n+1 + \sum_{k=1}^n \left[ \Pr \{k=l\} ( \bar{QS}(k-1) + \bar{QS}(n-k) ) \right] = n+1 + \frac{1}{n} \sum_{k=1}^n ( \bar{QS}(k-1) + \bar{QS}(n-k) )$$



$$\bar{QS}(n-k) = n+1 + \frac{2}{n} \sum_{l=0}^{n-1} \bar{QS}(l) \Rightarrow n \bar{QS}(n) = n(n+1) + 2 \sum_{l=0}^{n-1} \bar{QS}(l) \quad (1)$$

$$\left. \begin{aligned} &\bar{QS}(0) + \bar{QS}(n-1) \\ &+ \bar{QS}(1) + \bar{QS}(n-2) \\ &+ \dots \\ &+ \bar{QS}(n-2) + \bar{QS}(1) \\ &+ \bar{QS}(n-1) + \bar{QS}(0) \end{aligned} \right\}$$

$$2 \sum_{k=0}^{n-1} \bar{QS}(k) \quad (n+1) \bar{QS}(n+1) = (n+1)(n+2) + 2 \sum_{k=0}^n \bar{QS}(k) \quad (2)$$

$$(2) - (1) = (n+1) \bar{QS}(n+1) - n \bar{QS}(n) = (n+1)(n+2) - n(n+1) + 2 \bar{QS}(n) \Rightarrow$$

$$(n+1) \bar{QS}(n+1) = 2(n+1) + (n+2) \bar{QS}(n) \Rightarrow \bar{QS}(n+1) = 2 + \frac{(n+2)}{(n+1)} \bar{QS}(n)$$

$$\bar{QS}(n+1) = 2 + \frac{(n+2)}{(n+1)} \left( 2 + \frac{n+1}{n} \bar{QS}(n-1) \right) = \dots = 2 \left( 1 + \frac{(n+2)}{n} \left( \frac{1}{n-1} + \dots + \frac{1}{2} \right) \right)$$

$$H_n = \sum_{i=1}^n \frac{1}{i} \leq \ln(n) + 1$$

$$\sum_{i=2}^n \frac{1}{i} \approx \ln n$$

$$\overline{QS}(n) = 2 \left[ 1 + \ln n \left( \frac{1}{n} + \frac{1}{n-1} + \dots + \frac{1}{2} \right) \right] = 2 + 2(\ln n) \sum_{i=2}^n \frac{1}{i}$$

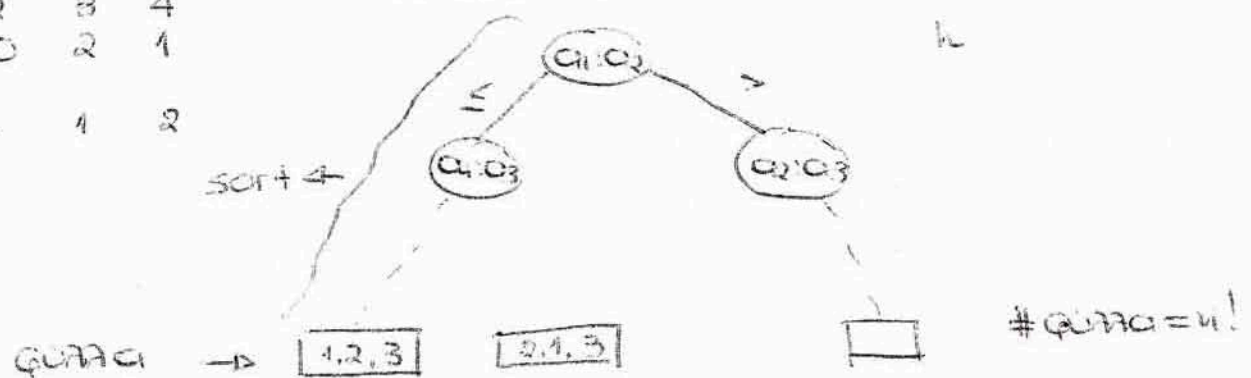
$$\overline{QS}(n) = 2 + 2(\ln n) \ln n \approx 4.44 \ln n$$

Γενικά:  $\overline{QS}(n) = O(n \log n)$

Δέντρο αποφασής:  $\neq$  κόμβος που έχει 2 παιδιά.  
(κάθε σύγκριση)

π.χ.

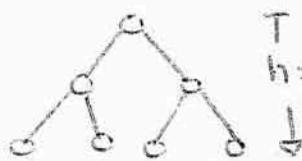
1	2	3	4
3	10	2	1
4	3	1	2



$$2^h = n!$$

$$\Downarrow$$

$$h \geq \log_2 n!$$



$h=2$  (υψος  $\neq$  αριθμος βημάτων που κάνει ένας αλγόριθμος)

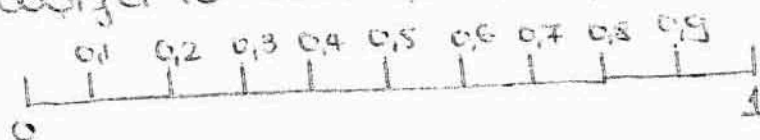
$$\approx \log_2 n! = n \log n \quad (\text{για αρκετά μεγάλα } n) \quad 4-\pi\chi\pi$$

ΣΥΒΟΡΙΣΗ ΤΑΞΙΝΟΜΗΣΗΣ

$$x \in [0, 1)$$

π.χ. [0,205, 0,101, 0,2, 0,47, 0,52, 0,78, 0,59]

Χωρίζει το διάστημα [0,1) σε υποδιαστήματα



Σε  $n$  υποδιαστήματα αντιστοιχίζουμε ένα υψός.

Παραγινε 1-1 αντιστοιχία της εικόνας και τα φινιζουμε ειναι καδους.



Ταξινομούμε  $\neq$  κάθε χρησιμοποιώντας insertion sort.



στη συνέχεια θα δούμε όπως θα ερμηνεύσει τον κώδικα

0,101  
0,2  
0,205  
0,41

Οι εγδοί αναπαριστούν τις αριθμούς και αριθμούς

εγδοί = αριθμοί

$O(n)$  εγδοί

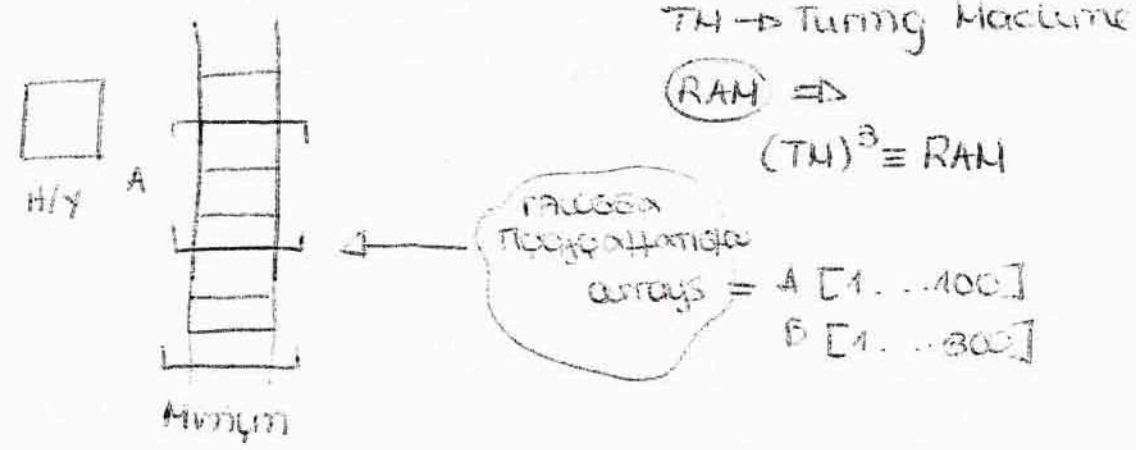
$O(n)$  εγδοί στη θέση περίπτωση

( $\forall$  εγδοί περιέχει εγδοί αριθμοί εγδοί)

25/11/2003

# Trees

↳ Μοντέλα αναπαράστασης



Ο αναπαράσταση των arrays αναφέρεται στο δέντρο

AVL-Tree = Adelson Velsky  
 Landis

- (1) AVL-tree  $\rightarrow$  1961
- (2) B-tree  $\rightarrow$  1962
- (3) BB-tree  $\rightarrow$  Tarjan
- (4) IST-tree  $\rightarrow$  Mc/Ts
- (5) BB(a)  $\rightarrow$  Rieger  
 Nieven  
 Blum/He

Median

nlgn



## External Sorting (SOS)

N = 1000000000

M = 10

M = 10 - 10 - 10

2p ταμίες

## A. SORTING AND MERGING EXAMPLE

↳ διατάσσεται τα κενά

M = 3

2.3 = 6 ταμίες

1η φάση: ASC → δέχεται την ταμιαία → AOS

1η ταμιαία: AOS DMN AEX | RTI → IRT KDN → DMN

2η -11- : IRT EGR <sup>LMP</sup> | Μπορούμε να χρησιμοποιήσουμε μόνο τις

3η -11- : AGN GIN επ ταμίες για να γράφω.

2η φάση: Διαχωρίζουμε τις ταμίες σε ταμίες εγδοού και σε ταμίες εγδοού. Στη συνέχεια, συγχωνεύουμε.

4 AAGINORST

(Διαχωρίζουμε το 10 εγδοού από ταμιαία)

5

6

Μετά:

O A I X G GIN → INO

AAI (ταμιαία + το μεγαλύτερο από ταμιαία 4)

Διαγράφω το A και ταυτόχρονα στη θέση του, ένα άλλο εγδοού και αυτός στη ίδια ταμιαία με το A.

Επαναλαμβάνω μέχρι να τελειώσω το block.

3η φάση: Επαναλαμβάνω για τις ταμίες 1, 2, 3.

1η φάση:  $\lceil N/M \rceil$  γράφω2η -11- : 10 εγδοού →  $\frac{\lceil N/M \rceil}{p}$  γράφω20 -11- →  $\frac{\lceil N/M \rceil}{p^2}$  -11-

⋮

k -11- →  $\frac{\lceil N/M \rceil}{p^k} = 1 \Rightarrow k = \log_p \lceil N/M \rceil$

# Replacement Selection

# EXTERNAL SORTING

$n=5$

Consecutive runs

Heap    Key    Next    Index

EXT	E	E	E
EXT	E	R	EE
RXT	R	N	EER
NXT	T	A	EERT
N*XA*	X	L	EERTX
N*L*A*	A*	S	A*
N*L*S*	L*	O	A*L*
N*O*S*	N*	R	A*L*N*
R*O*S*	O*	T	A*L*N*O*
R*T*S*	R*	I	A*L*N*O*R*
I T*S*	S*	N	A*L*N*O*R*T*
I T*N	T*	G	A*L*N*O*R*T*

Check replacement  
\*, otherwise go ahead  
to merge

in terms

know to advance, pages  
\* are replaced.

... then to merge  
... then ...

IGN → 31 - 11 -

## MEDIAN

Given  $n$  :  $|M|=n$

1- to find median element

$$i = \frac{n}{2}$$

In array :  $O(n \log n)$

2nd - 11 - :  $X \in O(n^2)$   $M \in O(n)$

3rd - 11 - :  $X \in O(n)$

In array → find

1. Divide  $M$  into two sets  $S$ ,  $M_1 = \{x \in M, x < S\}$

$M_2 = \{x \in M, x > S\}$



```

{
  if (|M1| = i - 1) return Si;
  else if (|M1| = i - 1)
    find (i = |M1| - 1, M2)
  else find (i, M1)
}

```

(C A R H - C - M)

i = 1

|M| = 1

$$T(n) = n + 1 + T(n - c) = n + 1 + n - c + 1 + T(n - 2c) = \dots = O(n^2)$$

$$T(n) = \max_i T(n, i)$$

$$T(n) \leq cn$$

$$T(n, i) \leq cn + \frac{1}{n} \left[ \sum_{k=1}^{i-1} T(n-k, i-k) + \sum_{k=i+1}^n T(k-1, i) \right]$$

κόστος  
για διαχωρισμό

κόστος για να  
βρούμε το i-οστό

στοίχείο στο  
αφαιρετικό

αφαιρετικό

αφαιρετικό

αφαιρετικό

αφαιρετικό

αφαιρετικό

αφαιρετικό

αφαιρετικό

αφαιρετικό

αφαιρετικό

αφαιρετικό

αφαιρετικό

αφαιρετικό

αφαιρετικό

αφαιρετικό

αφαιρετικό

αφαιρετικό

αφαιρετικό

αφαιρετικό

αφαιρετικό

αφαιρετικό

αφαιρετικό

αφαιρετικό

αφαιρετικό

αφαιρετικό

αφαιρετικό

αφαιρετικό

αφαιρετικό

$$T(n) = cn + \frac{1}{n} \left[ \sum_{k=1}^{i-1} T(n-k) + \sum_{k=i+1}^n T(k-1) \right]$$

$$= cn + \frac{1}{n} \left[ \sum_{k=1}^{i-1} T(n-k) + \sum_{k=i}^{n-1} T(k) \right]$$

$$T(n) \leq cn \cdot 4$$

$$T(0) = 0$$

$$T(n) \leq cn + \frac{1}{n} \left[ \sum_{k=1}^{i-1} 4c(n-k) + \sum_{k=i}^{n-1} 4ck \right] \leq cn + \frac{4c}{n} \left[ n(n-1) - \frac{(n-1+1)(n-1)}{2} - \frac{i(i-1)}{2} \right] \leq cn + \frac{4c}{n} \left[ n(n-1) - \frac{n(n-1)}{4} \right] \leq cn + \frac{4c}{n} \cdot \frac{3n^2}{4} = \frac{n+1}{2}$$

≤ 4n. Τον quicksort τον υλοποιήσατε αναδρομικά και για  
 δύο διαφορετικές ενώσεις παραγωγής οργάνωσης  
 τον καθόσατε με αρχή αναδρομικά για αυτό και βρήκε  
 ότι η παραγωγικότητα.



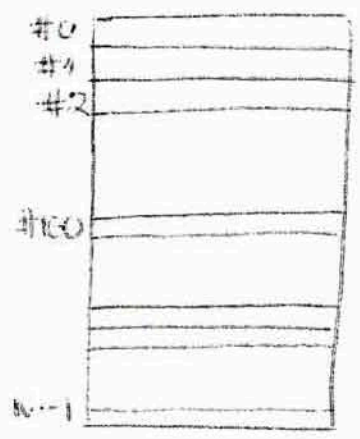
$T(n) = bn + T(n-1)$   
 $T(n) = bn + T(n/2)$  } Πρώτη vs. δεύτερη περίπτωση  
 $n = 10$  } Πρώτη να έχει πολύ περισσότερες φορές  
 $9$  } φορές

Select (i, M)

1. Kave kade diaxaristi  $14 \rightarrow 5 \rightarrow 41$   $1$   $1$   $1$   $1$   
 2. if ( $i > \lfloor M/2 \rfloor$ ) Select ( $i - \lfloor M/2 \rfloor, M/2$ )  $2 \neq 9 \rightarrow 10 \rightarrow 21$   $5$   $20$   $30$   $41$   $50$   $\dots$   $\lfloor \frac{n}{2} \rfloor$   
 else select ( $i, M/2$ )  
 $\downarrow$   
 median  
 (M)

2/12/2003

- Βασικοί και σύνθετοι τύποι στοιχείων
- Πρόσβαση των λεγόμενων
- Στοιχεία συνδυασμού ΔΔ



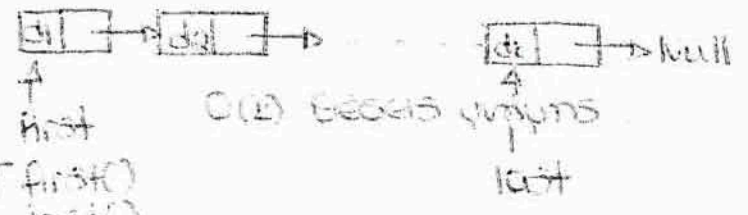
- Βασικοί τύποι δεδομένων { integer, real, char, pointers

• Πινάκες (Arrays)  $A[1 \dots n]$   
 $A[1:5]$

• Εγγραφές (Records)  $\rightarrow$  Pointer Machine  
 struct {

integer  
 pointer  
 }  
 Ηωρα να έχουμε αλληλολογιστικά στοιχεία, ώστε  
 πρόσβαση αλληλολογιστικά των δύο λειτουργιών.

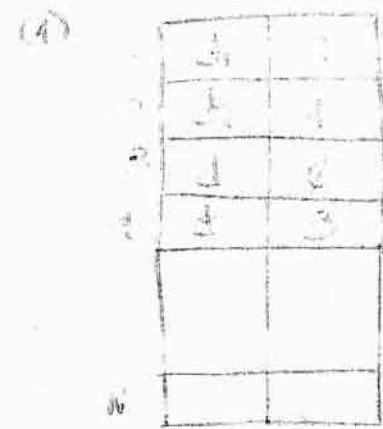
• Λίστα (List)



\* Στο βιβλίο αλφάβητος πρόσβαση

- first()
- last()
- size()  $\rightarrow$  αν επιτρέπει να πάρουμε την τιμή
- insert-after(pointer, data)
- find(data) αν υπάρχει
- concatenate-with(list)  $\rightarrow$  αν συνδυασμός αλφάβητος πρόσβασης
- delete(pointer)

Λιανωμένη Λίστα → (1) πίνακες (2) εγγραφές

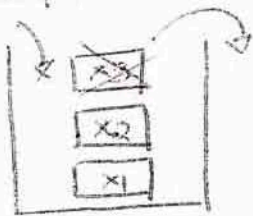


(2) struct {  
data  
pointer  
}  
→ Δεν ε'ι απαραίτητα το ας εγγραφές.

Βιναλές Λιανές



• Στάκος (stack)



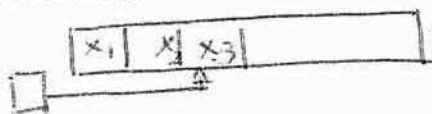
push(data)

pop()

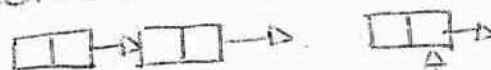
top() → δίνει την κορυφή

is\_empty()

• Πίνακες



• records (Λιανές)



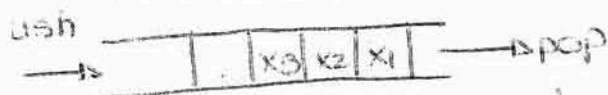
insert-after(last, data)

delete(first)

last()

size()

• Ουρά (Queue)



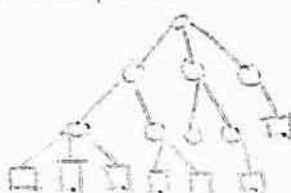
push(data)

pop()

is\_empty()

insert-before(first, data)

• Δένδρα (Trees)



Λιανές

$E < N - 1$

leaf node

$T = (N, E) \quad E = N - 1$

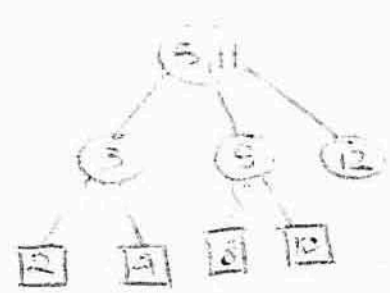
$T' = (N+1, E')$

$T_1 = W$

$T_2 = N+1-m$

Δένδρο → Γράφοι - πολλαπλάσιοι γόνοι  
 - 1 γονιός, 2 γόνοι, 3 γόνοι, ...

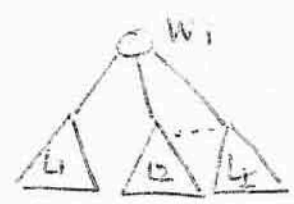
π.χ



{2,4,8,10,12}

• preorder

- $w_1$
- $L_1, L_2, \dots$

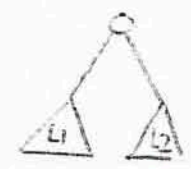


• postorder

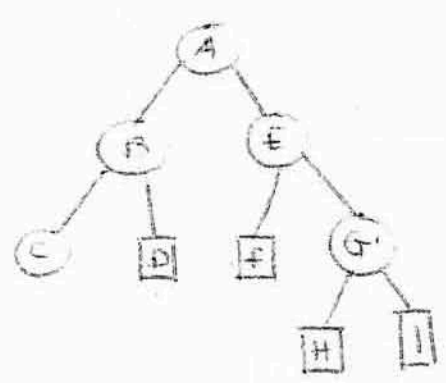
- $L_1, L_2, \dots$

• symmetric order

- $L_1$
- $w_1$
- $L_2$



π.χ



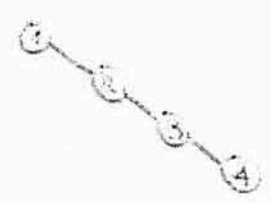
preorder : A B C D E F G H I

postorder : C D B F H I G E A

symmetric order : C B D A F E H G I

search → γειγωτο το έλεος για μια τιμή

1 2 3 4



→ το δένδρο εξελίσσεται σε μια αλυσίδα  
 καλοί συγγραφείς

• Abstract data types → έχει κάποιες ιδιότητες

concrete -||- -||-





Access(x) :  $x \in U$ , inf(x)  $\rightarrow$  O(1) static

Insert(x) :  $x \in U$   $S \rightarrow S + \{x\}$   $\rightarrow$   O(1) dynamic  
 Delete(x) :  $x \in S$   $S \rightarrow S - \{x\}$   $\rightarrow$   O(1) dynamic

O(1)  $\rightarrow$  εφελκυσμός

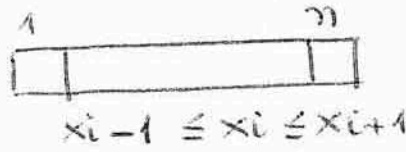
$\rightarrow$  διαρκώς (persistent)  $\rightarrow$  υπάρχουν και όλες οι προηγούμενες (κατασκευές)  $\rightarrow$  κόστος O(1) χρόνος.

persistent  $\rightarrow$  partial (υποσύνολο να ελεγχόμαστε την αλήθεια να διαβάσουμε τις προηγούμενες)  
 $\rightarrow$  fully

O(1)  $\rightarrow$  comparison based  
 $\rightarrow$  representation II -

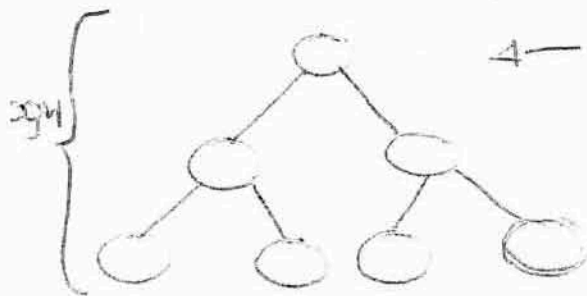
O(1)  $\rightarrow$  explicit  $O(n)$   
 $\rightarrow$  implicit  $n + O(1)$  κόστος που χρησιμοποιείται

$S \subseteq U$ ,  $|S| = n$   $x_i \in S$   
 σταθερή ακολουθία εφελκυσμού



$O(n) \rightarrow$  στην χειρότερη περίπτωση

Access(x)



Binary search

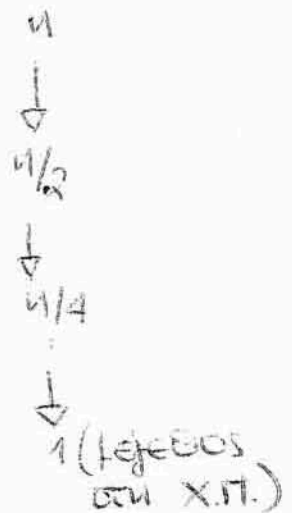


$$T_n = \frac{1}{2} T_{n-1}$$

$$n \left(\frac{1}{2}\right)^h = 1 \rightarrow n = 2^h$$

$$h = \log_2 n \quad \times 11$$

$$\frac{\lceil x - x_1 \rceil}{x_n - x_1}$$



My Recs

Algorithm 2.10

1. if  $(1 \leq |M| \leq 5)$

    return  $M$  (base cases)

2. Return first element of  $M$

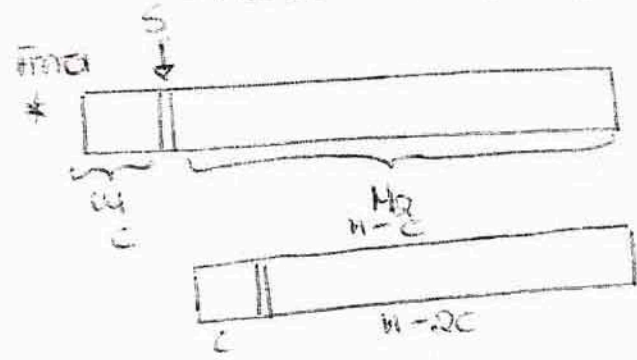
3. Divide  $S$  into two sets  $M_1 = \{x \in M : x \leq S\}$   
 $M_2 = \{x \in M : x > S\}$

4. if  $(1 \leq |M_1|)$

    Select  $(1, M_1)$ ;

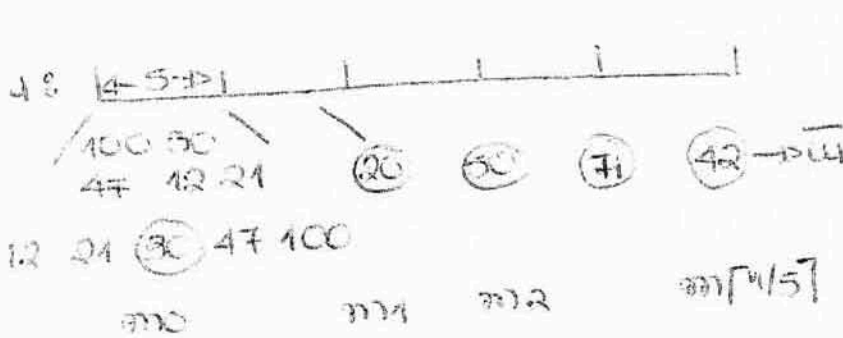
else

    Select  $(1 - |M_1|, M_2)$ ;



$$T(n) = n + T(n-1) = n + n-1 + T(n-2) = \dots = O(n^2)$$

$$T(1) = 0$$

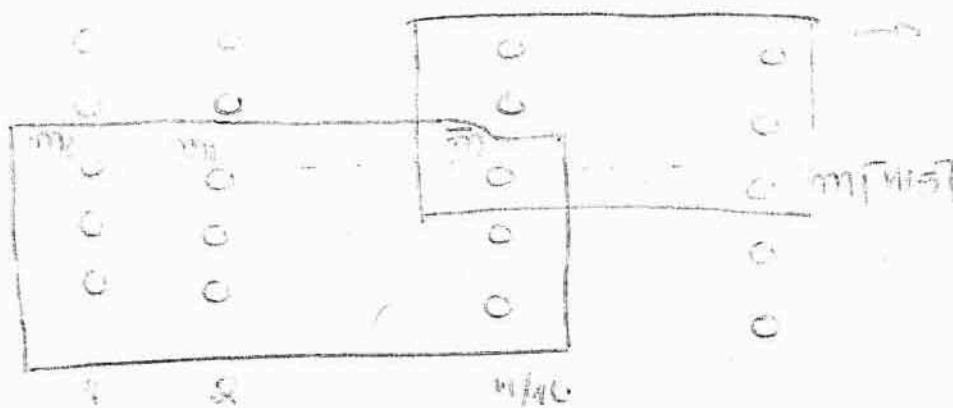


Χωρίζω το array σε πεντάδες  
 Ταξινομώ τα στοιχεία και  
 Επιστρέφω το median

$$\bar{m} = \text{Select}(\lceil \frac{n}{5} \rceil, (m_0, m_1, \dots, m_{\lceil n/5 \rceil}))$$

↳ Επιστρέφω το μέσο των μέσων (πολύ καλύτερο διαχωριστικό)

Βασίζομαι στις πεντάδες ως άλλες 5



→ Στο επόμενο βήμα υπολογίζουμε το μέσο των μέσων, δηλαδή το  $\bar{m}$ .

Τα στοιχεία τα αργότερα είναι υψότερα από τον median.  
Το  $\bar{m}$  είναι ο median των median.

$\frac{3n}{10}$ ,  $n \bmod 10 = 0 \Leftrightarrow 10 | n$

$\frac{3n}{11}$  στοιχεία όταν  $n \bmod 10 \neq 0 \Leftrightarrow 10 \nmid n$   
(μειωότερα από το  $\lceil \frac{3n}{10} \rceil$ )

$\frac{3n}{11}$  στοιχεία το πολύ μεγαλύτερα από  $\bar{m}$

Η χειρότερη περίπτωση είναι όταν λάβει η select για το μεγαλύτερο από τα 2 κομμάτια. (βλ.  $\frac{8n}{11}$ )

$$T(n) = \begin{cases} cn & , n \leq 100 \\ bn + T\left(\frac{8n}{11}\right) + T\left(\frac{21n}{100}\right) & , n > 100 \end{cases}$$

$\downarrow$   $\downarrow$   $\downarrow$   
 Διαχωρισμός  $\downarrow$   $\downarrow$   
 Select  $\downarrow$   $\downarrow$   
 να βρούμε το median των median

$T(n) \leq cn$

- για  $n \leq 100$  προφανές

- για  $n > 100$   $T(n) \leq T\left(\frac{8n}{11}\right) + T\left(\frac{21n}{100}\right) + bn \leq c \cdot \frac{8n}{11} + c \cdot \frac{21n}{100} + bn =$

$$cn \left( \frac{8}{11} + \frac{21}{100} + \frac{b}{c} \right) \leq cn$$

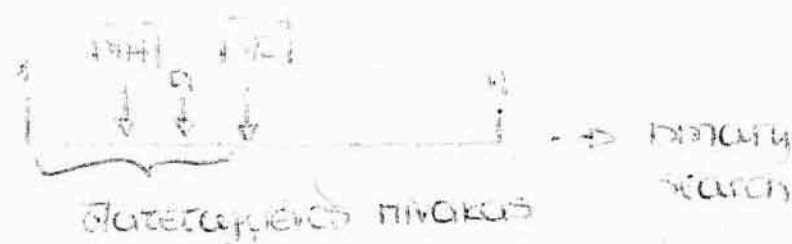
$\downarrow$  πρέπει

$$\frac{8}{11} + \frac{21}{100} + \frac{b}{c} \leq 1 \Rightarrow \boxed{c \geq \frac{1100b}{69}}$$

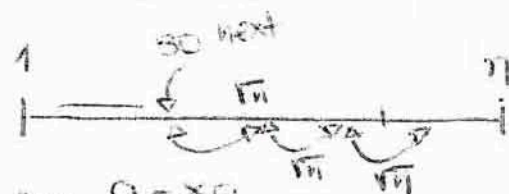
Επιλέγουμε τελικά  $c = \max\left(\frac{1100b}{69}, a\right)$



## Interpolation



1.  $a \leq A[n/2]$
2.  $a \leq A[n/4]$  οχι



$$p = \frac{a - x_0}{x_{n+1} - x_0}$$

$$next = \lceil p \cdot n \rceil$$

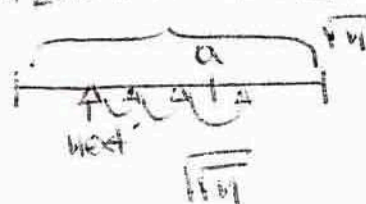
$$a \geq A[next]$$

$$a \geq A[next + \sqrt{n}]$$

Ψάχνω το  $a$

Συνεχίζω μέχρι  $a \leq A[next + (1-\epsilon)\sqrt{n}]$

$$A[next + (1-\epsilon)\sqrt{n}] \leq a \leq A[next + \sqrt{n}]$$



## Μέση Πολυπλοκότητα

$$\log \log n \quad n = 2^{32} \quad \log \log \approx 5$$

$C = \# \text{ ερωτήσεων μέχρι να βρεθεί υπάρχει } \sqrt{n} \quad P_i = P\{\text{χρειάζονται } \geq i \text{ ερωτήσεις}\}$

$$C = \sum_{i \geq 1} (i \cdot P\{\text{καμμι βρήκατε}\}) = \sum_{i \geq 1} i(\varphi_i - \varphi_{i+1}) = \varphi_1 - \varphi_2 + 2\varphi_2 - 2\varphi_3 + 3\varphi_3 - 3\varphi_4 + \dots = \sum_{i \geq 1} \varphi_i$$

$$+ 3\varphi_3 - 3\varphi_4 + \dots = \sum_{i \geq 1} \varphi_i$$

$$\varphi_1 = \varphi_2 = 1$$

$$\varphi_i \leq P\{\text{όχι } a - next \geq (i-2)\sqrt{n}\}, i \geq 3$$

$$P\{|x - \mu| \geq \epsilon\} \leq \frac{\sigma^2}{\epsilon^2} \text{ ανισότητα Chebyshev}$$

Τυχαία μεταβλητή  $X = \{ \text{αριθμός } x_j \leq a \}$

$$Y = \{ \text{αριθμός δεσμών } 4 \leq n \leq 5 \}$$

$$P(Y=2) = \binom{2}{2} (2/6)^2 (1-2/6)^{5-2} \rightarrow \text{για } \mu \text{ αλυσίδα}$$

$P(x=j) = \binom{n}{j} p^j (1-p)^{n-j}$   $t = q \cdot n = n \cdot x$

$p = \frac{x_{i+1} - x_i}{x_{i+1} - x_0}$



αριθμός  
περιόδων

$\sigma^2 = p(1-p)n$   
(τετραγωνό διασποράς)

$p \leq \frac{p(1-p)n}{(1-2)^2 n} \leq \frac{1}{4(1-2)^2}$  επειδή  $p(1-p) \leq \frac{1}{4}, p \in (0,1)$   
Λόγικος λόγος το  $\frac{1}{4}$

$\sum_{i=1}^{\infty} q_i = 2 + \sum_{i=3}^{\infty} q_i = 2 + \sum_{i=3}^{\infty} \frac{1}{4(1-2)^2} = 2 + \frac{\pi^2}{24} \approx 2.4$

$\Rightarrow \boxed{C \approx 2.4}$

$\bar{T}(n) = C + \bar{T}(\sqrt{n}) \approx 2 + C \cdot \log \log n = O(\log \log n)$

↓  
next  
να βρούμε  
το επόμενο  
για να αρχίσω  
την αναζήτηση

↓  
εφαρμογή  
αναδρομικά

Χειρότερη περίπτωση  $\rightarrow \sqrt{n}$

9/12/2003

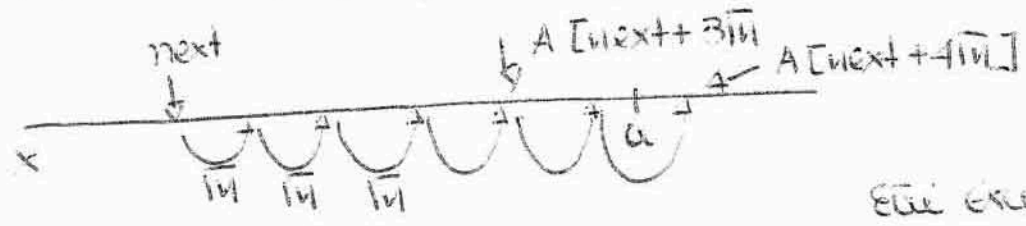
Interpolation Search

ΠΜΠ:  $\bar{T}(n) = O(\log \log n)$  για ομοιόμορφα κατανομημένα στοιχεία

3 5 7 9,5 11

5 15 20 25 26 26.2 26.5 27 27.3 28 32 37 39 41

ΠΜΠ:  $n_{\text{next}} = \lceil \sqrt{n} \rceil$  (αρχική εκτίμηση)



Εδώ εδώ  $i=4$

$A[n_{\text{next}} + (i-1)\sqrt{n}] \leq a \leq A[n_{\text{next}} + i\sqrt{n}]$

$$i \pm \frac{Q}{\sqrt{n}} = \sqrt{n}$$



Για να πάρω  $\sqrt{n}$  επαρκά στο δεξιό άκρο  $\sqrt{n}$ .

Ας α πχπ ε  $n^{1/2} + n^{1/4} + n^{1/8} + \dots = \sqrt{n} = O(\sqrt{n})$

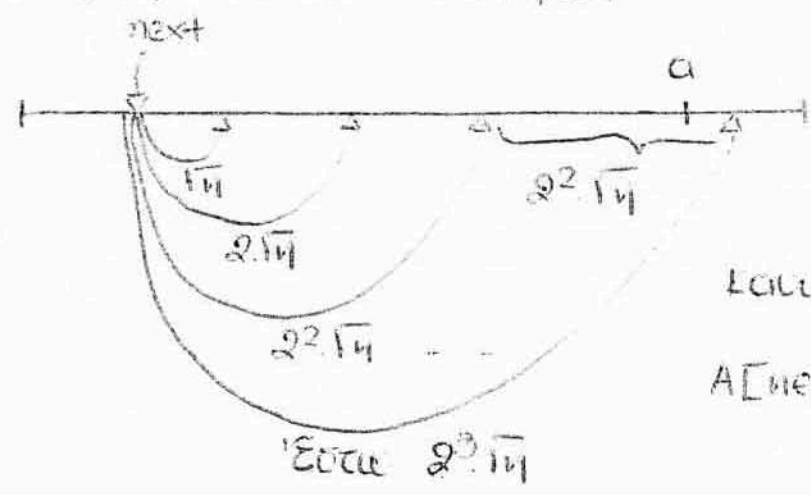
στο 4  
αρχικό  
ερώτημα

2 ε  
ερώτημα

3 ε  
ερώτημα

Σε μια j-προσπαquia, το  $\sqrt{n}$  προσεγγίζεται με το  $\sqrt{n}$

Σταματούμε την εφαρμογή τα αλγόριθμοι, όταν φτάσουμε σε μέγεθος διαστήματος.

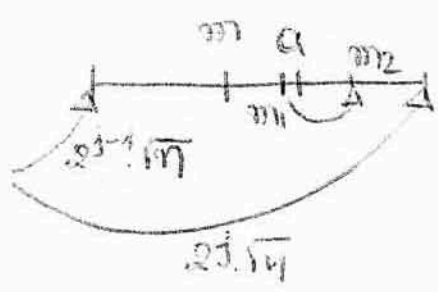


Υπολογίζω την ακριβή  
επιλογή

Εάν  $2^j \cdot \sqrt{n}$  βήματα

$$A[next + 2^{j-1} \cdot \sqrt{n}] \leq a \leq A[next + 2^j \cdot \sqrt{n}]$$

Έστω  $2^j \cdot \sqrt{n}$



πηγαίνω στη θέση και βρίσκω  $A[m] \leq a$ .

Στη συνέχεια, βρίσκω τη θέση του βεγιά διαστήματος.  $A[m] \leq a$ ,  $A[m_2] \geq a$ .

Συνεχίζω μέχρι  $|m_2 - m| \leq \sqrt{n}$

Το ερώτημα είναι αν είναι

Εάν  $2^j$  με  $\frac{1}{\epsilon}$  είναι η μέγεθος του φάσματος στη θέση

$$2^j \sqrt{n} \leq n \Rightarrow j = O(\log n)$$

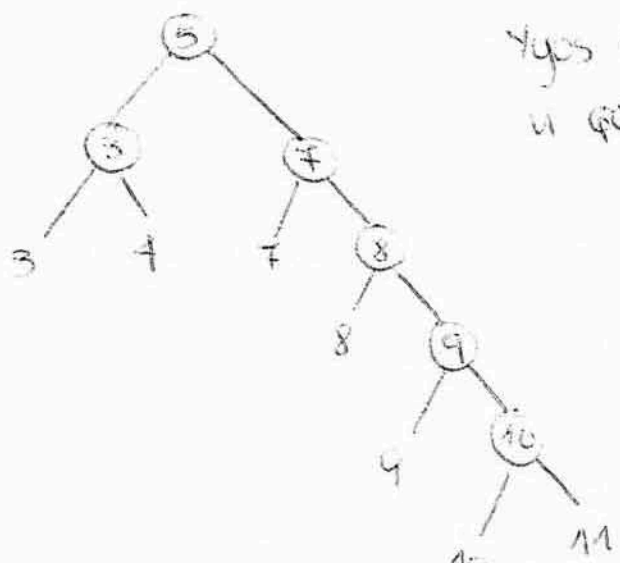


2c ορατότητα :  $2^{h-1} \sqrt{h}$

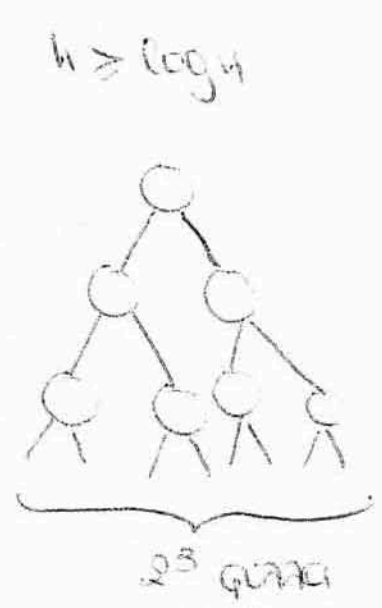
ηδη ποσο  $\log(2^{h-1})$  επιπλέον  $\log(2^{h-1}) = \log(2^{h-1})$

$$\log n + \log n^{1/2} + \log n^{1/4} + \dots = \log n \left(1 + \frac{1}{2} + \frac{1}{4} + \dots\right) = O(\log n)$$

AVL Δέντρο



Υψος  $h$  :  
 $h \geq \log n$



Συμμετρικά Δέντρα η φύλλων  
Υψος :  $\Theta(\log n)$

Υποστηρίγματα  
π.χ. AVL, Red-black  
Βασισμένα

$n=3$   
B-Δέντρο, (a,b)  
-Δέντρα

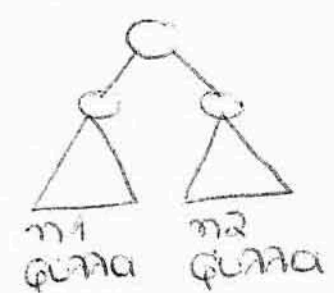
2	10	17	32	41	52	68
---	----	----	----	----	----	----

22 32 41 52 68

Βαρος  $\rightarrow$  αριθμός φύλλων

Πίνακας  
Αναζήτησης  $O(\log n)$

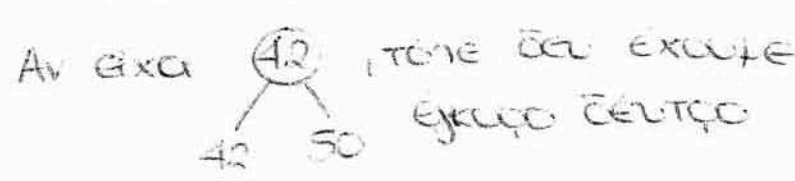
Εισαγωγή  
Διαγραφή?  $O(n)$   $\rightarrow$  χειρότερη περίπτωση



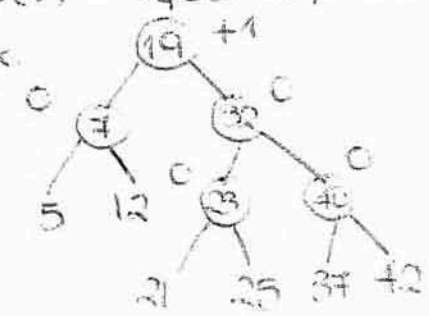
$L \rightarrow O(\log^2 n)$  από search  $O(\log^2 n)$

Υψοποίηση  $v \equiv hb(v) = \text{Υψος δεξιού παιδιού}(v) - \text{Υψος αριστερού παιδιού}(v)$

$hb(v) = +1, 0, -1$  εσωτερικής τιμής π.χ.



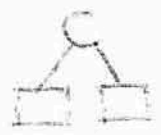
αλλιώς  $hb(42) = 2$

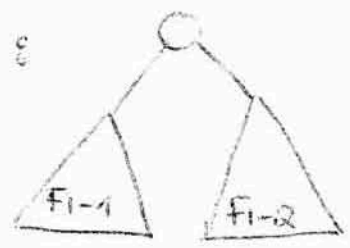


# Αλγόριθμοι Περσινταζών AVL Δέντρων

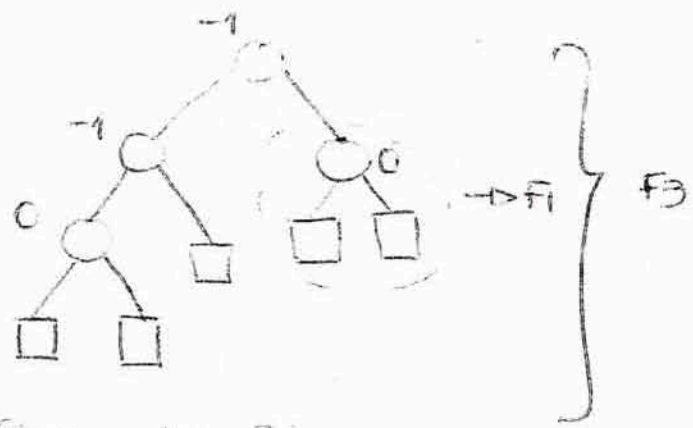
## ↳ ΤΙΜΕΡΙΝΟΙ ΔΕΝΤΡΟΙ

$F_0$  : □ (1 φύλλο)

$F_1$  :  (1 κόμβος + 2 φύλλα)

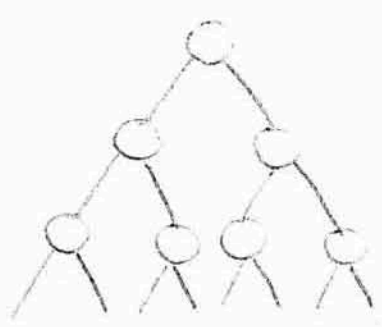
$F_2$  : 

π.χ.  $F_2$  :



Είμαι AVL δέντρο

Έχω νόμους κόμβος  $\neq$  γύρω -1



→ Ελαττώση μεγίστων AVL δέντρων (πάσης)

Πάσης ελαττω δέντρο η φωνάει  $2^h = n \Rightarrow h = \log_2 n$   
(ελαττωση περσινταζών)

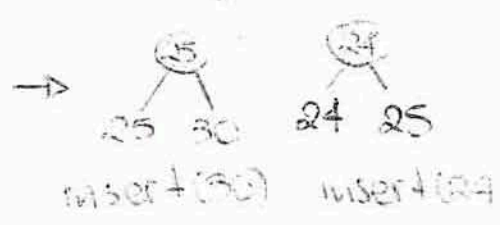
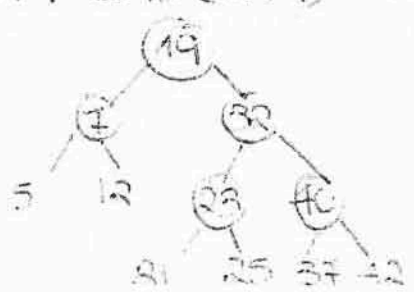
$F_n$  έχει ύψος  $n$  →  $\Phi_{n+1}$  φωνάει διαφορετικοι fibonacci

π.χ.  $F_3 \rightarrow$  ύψος = 3  $\Phi_{n+1} = \Phi_n + \Phi_{n-1}$

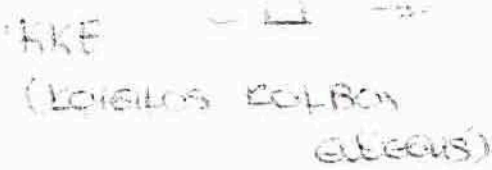
$$\Phi_n = \frac{a^{n+1} - b^{n+1}}{\sqrt{5}} \text{ όπου } a = \frac{1+\sqrt{5}}{2} \text{ και } b = \frac{1-\sqrt{5}}{2} \approx -1$$

$$\text{Εστω } \Phi_{n+1} = n \Rightarrow \frac{a^{n+2} - b^{n+2}}{\sqrt{5}} = n \leq \frac{a^{n+2} - 1}{\sqrt{5}} = n \Rightarrow$$

$$a^{n+2} \leq \sqrt{5}n + 1 \leq \sqrt{5}(n+1) \Rightarrow n \leq \frac{\log \sqrt{5}}{\log a} + \log(n+1) - 2 = O(\log n)$$



Για να κάνω ελαττω



→ Bildung d. des Körpers zur  
Anleitung in Tüfteln, Herabsetzen  
desen, das Drogen, das  
Opium

Diagram illustrating a binary tree structure with nodes labeled with values and operations:

- Root node:  $+1$
- Left child of root:  $+2 \rightarrow \text{ELE}$
- Right child of root:  $-1$
- Left child of  $-1$ :  $-1$
- Right child of  $-1$ :  $+1$
- Left child of  $+1$ :  $\square$
- Right child of  $+1$ :  $\square$
- Left child of  $\square$ :  $\square$
- Right child of  $\square$ :  $\square$

451421200

Definición  $\mathcal{U} := [0, \dots, N-1]$  donde  $N$  está dada

Union  $(A, B, C)$  : edges  $A$  and  $B$  and  $C$

A hand-drawn diagram of two cells, labeled A and B. Cell A is on the left and contains four numbered organelles: 1 (mitochondrion), 2 (nucleus), 3 (vacuole), and 4 (chloroplast). Cell B is on the right and contains two numbered organelles: 5 (mitochondrion) and 6 (nucleus). The labels A and B are at the bottom, and C and D are on the right.

Union (B, C, D)

Find  $(c) = 18$

$\text{Fintel}(0) = D$  (Analogie to Reparaturkosten  
Grund + Jekvate ra  
Unterhaltung)

Τον υπογράφει 15-10

N-1 → Give to every cell individually  
etc. endo control + energetic

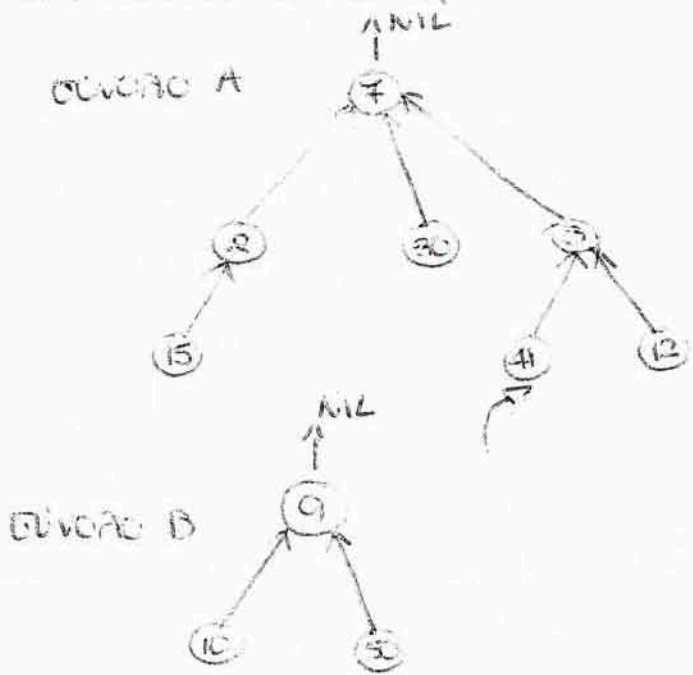
B	M	12	3	3	C	C	E
C	1	3	3	4	5	6	4



Αλγόριθμος  $n-1$  Union  $\rightarrow$   $\log_2 n$

Επικοινωνία  $n-1$  Union  $\rightarrow \frac{\log_2 n - 1}{n-1} = O(\log n)$

Εν Αλγόριθμο Δίνει Εφαρμογή στο Union

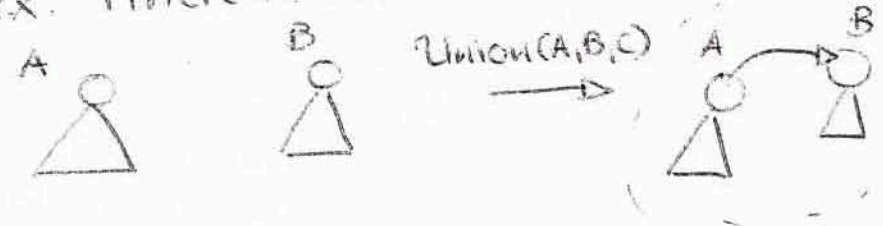


Αναπαριστάται με δυο δέντρα. Ο κάθε κόμβος ~ Ουρά φράγας

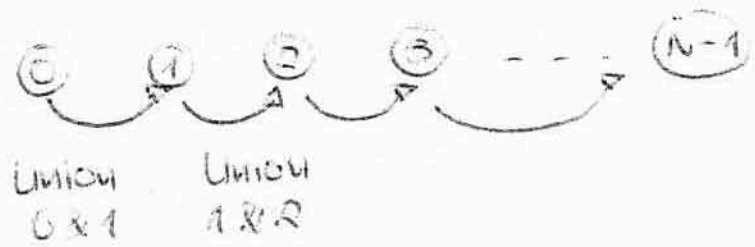
Αναπαριστάται το σύνολο ως δέντρο και αντιστρέφουμε τους δείκτες (τα παιδιά δείχνουν στον πατέρα)

Για να βρούμε find to 41, ξεκινάμε από το 41 και φτάνουμε στην ρίζα όπου καταλήγει το όλη της.

π.χ.  $\text{find}(41) = A$



Κάνουμε την ρίζα του A να δείχνει στο B.



Φαίνεται απλοϊκά από κόμβους (απλή αλυσίδα των ριζών)

$\text{find} = O(N) \rightarrow$  ΔΕΚΕΙΟΣ

Τις έχουμε να το αναβελούμε?

Weighted Union Rule



Το σύνολο με τα λιγότερα στοιχεία δείχνει στο σύνολο με τα περισσότερα στοιχεία.

Αλγόριθμος  $find$  - 1 Union  $\rightarrow$  2 calls

Union  $\rightarrow$  2 calls  $\rightarrow$  4 calls  $\rightarrow$  8 calls  $\rightarrow$  16 calls  $\rightarrow$  32 calls  $\rightarrow$  64 calls  $\rightarrow$  128 calls  $\rightarrow$  256 calls  $\rightarrow$  512 calls  $\rightarrow$  1024 calls  $\rightarrow$  2048 calls  $\rightarrow$  4096 calls  $\rightarrow$  8192 calls  $\rightarrow$  16384 calls  $\rightarrow$  32768 calls  $\rightarrow$  65536 calls  $\rightarrow$  131072 calls  $\rightarrow$  262144 calls  $\rightarrow$  524288 calls  $\rightarrow$  1048576 calls  $\rightarrow$  2097152 calls  $\rightarrow$  4194304 calls  $\rightarrow$  8388608 calls  $\rightarrow$  16777216 calls  $\rightarrow$  33554432 calls  $\rightarrow$  67108864 calls  $\rightarrow$  134217728 calls  $\rightarrow$  268435456 calls  $\rightarrow$  536870912 calls  $\rightarrow$  1073741824 calls  $\rightarrow$  2147483648 calls  $\rightarrow$  4294967296 calls  $\rightarrow$  8589934592 calls  $\rightarrow$  17179869184 calls  $\rightarrow$  34359738368 calls  $\rightarrow$  68719476736 calls  $\rightarrow$  137438953472 calls  $\rightarrow$  274877906944 calls  $\rightarrow$  549755813888 calls  $\rightarrow$  1099511627776 calls  $\rightarrow$  2199023255552 calls  $\rightarrow$  4398046511104 calls  $\rightarrow$  8796093022208 calls  $\rightarrow$  17592186044416 calls  $\rightarrow$  35184372088832 calls  $\rightarrow$  70368744177664 calls  $\rightarrow$  140737488355328 calls  $\rightarrow$  281474976710656 calls  $\rightarrow$  562949953421312 calls  $\rightarrow$  1125899906842624 calls  $\rightarrow$  2251799813685248 calls  $\rightarrow$  4503599627370496 calls  $\rightarrow$  9007199254740992 calls  $\rightarrow$  18014398509481984 calls  $\rightarrow$  36028797018963968 calls  $\rightarrow$  72057594037927936 calls  $\rightarrow$  144115188075855872 calls  $\rightarrow$  288230376151711744 calls  $\rightarrow$  576460752303423488 calls  $\rightarrow$  1152921504606846976 calls  $\rightarrow$  2305843009213693952 calls  $\rightarrow$  4611686018427387904 calls  $\rightarrow$  9223372036854775808 calls  $\rightarrow$  18446744073709551616 calls  $\rightarrow$  36893488147419103232 calls  $\rightarrow$  73786976294838206464 calls  $\rightarrow$  147573952589676412928 calls  $\rightarrow$  295147905179352825856 calls  $\rightarrow$  590295810358705651712 calls  $\rightarrow$  1180591620717411303424 calls  $\rightarrow$  2361183241434822606848 calls  $\rightarrow$  4722366482869645213696 calls  $\rightarrow$  9444732965739290427392 calls  $\rightarrow$  18889465931478580854784 calls  $\rightarrow$  37778931862957161709568 calls  $\rightarrow$  75557863725914323419136 calls  $\rightarrow$  151115727451828646838272 calls  $\rightarrow$  302231454903657293676544 calls  $\rightarrow$  604462909807314587353088 calls  $\rightarrow$  1208925819614629174706176 calls  $\rightarrow$  2417851639229258349412352 calls  $\rightarrow$  4835703278458516698824704 calls  $\rightarrow$  9671406556917033397649408 calls  $\rightarrow$  19342813113834066795298816 calls  $\rightarrow$  38685626227668133590597632 calls  $\rightarrow$  77371252455336267181195264 calls  $\rightarrow$  154742504910672534362390528 calls  $\rightarrow$  309485009821345068724781056 calls  $\rightarrow$  618970019642690137449562112 calls  $\rightarrow$  1237940039285380274899124224 calls  $\rightarrow$  2475880078570760549798248448 calls  $\rightarrow$  4951760157141521099596496896 calls  $\rightarrow$  9903520314283042199192993792 calls  $\rightarrow$  19807040628566084398385987584 calls  $\rightarrow$  39614081257132168796771975168 calls  $\rightarrow$  79228162514264337593543950336 calls  $\rightarrow$  158456325028528675187087900672 calls  $\rightarrow$  316912650057057350374175801344 calls  $\rightarrow$  633825300114114700748351602688 calls  $\rightarrow$  1267650600228229401496703205376 calls  $\rightarrow$  2535301200456458802993406410752 calls  $\rightarrow$  5070602400912917605986812821504 calls  $\rightarrow$  10141204801825835211973625643008 calls  $\rightarrow$  20282409603651670423947251286016 calls  $\rightarrow$  40564819207303340847894502572032 calls  $\rightarrow$  81129638414606681695789005144064 calls  $\rightarrow$  162259276829213363391578010288128 calls  $\rightarrow$  324518553658426726783156020576256 calls  $\rightarrow$  649037107316853453566312041152512 calls  $\rightarrow$  1298074214633706907132624082305024 calls  $\rightarrow$  2596148429267413814265248164610048 calls  $\rightarrow$  5192296858534827628530496329220096 calls  $\rightarrow$  10384593717069655257060992658440192 calls  $\rightarrow$  20769187434139310514121985316880384 calls  $\rightarrow$  41538374868278621028243970633760768 calls  $\rightarrow$  83076749736557242056487941267521536 calls  $\rightarrow$  166153499473114484112975882535043072 calls  $\rightarrow$  332306998946228968225951765070086144 calls  $\rightarrow$  664613997892457936451903530140172288 calls  $\rightarrow$  1329227995784915872903807060280344576 calls  $\rightarrow$  2658455991569831745807614120560689152 calls  $\rightarrow$  5316911983139663491615228241121378304 calls  $\rightarrow$  10633823966279326983230456482242756608 calls  $\rightarrow$  21267647932558653966460912964485513216 calls  $\rightarrow$  42535295865117307932921825928971026432 calls  $\rightarrow$  85070591730234615865843651857942052864 calls  $\rightarrow$  170141183460469231731687303715884105728 calls  $\rightarrow$  340282366920938463463374607431768211456 calls  $\rightarrow$  680564733841876926926749214863536422912 calls  $\rightarrow$  1361129467683753853853498429727072845824 calls  $\rightarrow$  2722258935367507707706996859454145691648 calls  $\rightarrow$  5444517870735015415413993718908291383296 calls  $\rightarrow$  10889035741470030830827987437816582766592 calls  $\rightarrow$  21778071482940061661655974875633165533184 calls  $\rightarrow$  43556142965880123323311949751266331066368 calls  $\rightarrow$  87112285931760246646623899502532662132736 calls  $\rightarrow$  174224571863520493293247799005065324265472 calls  $\rightarrow$  348449143727040986586495598010130648530944 calls  $\rightarrow$  696898287454081973172991196020261297061888 calls  $\rightarrow$  1393796574908163946345982392040522594123776 calls  $\rightarrow$  2787593149816327892691964784081045188247552 calls  $\rightarrow$  5575186299632655785383929568162090376495104 calls  $\rightarrow$  11150372599265311570767859136324180752990208 calls  $\rightarrow$  22300745198530623141535718272648361505980416 calls  $\rightarrow$  44601490397061246283071436545296723011960832 calls  $\rightarrow$  89202980794122492566142873090593446023921664 calls  $\rightarrow$  178405961588244985132285746181186892047843328 calls  $\rightarrow$  356811923176489970264571492362373784095686656 calls  $\rightarrow$  713623846352979940529142984724747568191373312 calls  $\rightarrow$  1427247692705959881058285969449495136382746624 calls  $\rightarrow$  2854495385411919762116571938898990272765493248 calls  $\rightarrow$  5708990770823839524233143877797980545530986496 calls  $\rightarrow$  11417981541647679048466287755595961091061972992 calls  $\rightarrow$  22835963083295358096932575511191922182123945984 calls  $\rightarrow$  45671926166590716193865151022383844364247891968 calls  $\rightarrow$  91343852333181432387730302044767688728495783936 calls  $\rightarrow$  182687704666362864775460604089535377456991567872 calls  $\rightarrow$  365375409332725729550921208179070754913983135744 calls  $\rightarrow$  730750818665451459101842416358141509827966271488 calls  $\rightarrow$  1461501637330902918203684832716283019655932542976 calls  $\rightarrow$  2923003274661805836407369665432566039311865085952 calls  $\rightarrow$  5846006549323611672814739330865132078623730171904 calls  $\rightarrow$  11692013098647223345629478661730264157247460343808 calls  $\rightarrow$  23384026197294446691258957323460528314494920687616 calls  $\rightarrow$  46768052394588893382517914646921056628989841375232 calls  $\rightarrow$  93536104789177786765035829293842113257979682750464 calls  $\rightarrow$  187072209578355573530071658587684226515959365500928 calls  $\rightarrow$  374144419156711147060143317175368453031918731001856 calls  $\rightarrow$  748288838313422294120286634350736906063837462003712 calls  $\rightarrow$  1496577676626844588240573268701473812127674924007424 calls  $\rightarrow$  2993155353253689176481146537402947624255349848014848 calls  $\rightarrow$  5986310706507378352962293074805895248510699696029696 calls  $\rightarrow$  11972621413014756705924586149611790497021399392059392 calls  $\rightarrow$  23945242826029513411849172299223580994042798784118784 calls  $\rightarrow$  47890485652059026823698344598447161988085597568237568 calls  $\rightarrow$  95780971304118053647396689196894323976171195136475136 calls  $\rightarrow$  191561942608236107294793378393788647952342390272950272 calls  $\rightarrow$  383123885216472214589586756787577295904684780545900544 calls  $\rightarrow$  766247770432944429179173513575154591809369561091801088 calls  $\rightarrow$  1532495540865888858358347027150309183618739122183602176 calls  $\rightarrow$  3064991081731777716716694054300618367237478244367204352 calls  $\rightarrow$  6129982163463555433433388108601236734474956488734408704 calls  $\rightarrow$  12259964326927110866866776217202473468949912977468817408 calls  $\rightarrow$  24519928653854221733733552434404946937899825954937634816 calls  $\rightarrow$  49039857307708443467467104868809893875799651909875269632 calls  $\rightarrow$  98079714615416886934934209737619787751599303819750539264 calls  $\rightarrow$  196159429230833773869868419475239575503198607639501078528 calls  $\rightarrow$  392318858461667547739736838950479151006397215279002157056 calls  $\rightarrow$  784637716923335095479473677900958302012794430558004314112 calls  $\rightarrow$  1569275433846670190958947355801916604025588861116008628224 calls  $\rightarrow$  3138550867693340381917894711603833208051177722232017256448 calls  $\rightarrow$  6277101735386680763835789423207666416102355444464034512896 calls  $\rightarrow$  12554203470773361527671578846415332832204710888928069025792 calls  $\rightarrow$  25108406941546723055343157692830665664409421777856138051584 calls  $\rightarrow$  50216813883093446110686315385661331328818843555712276103168 calls  $\rightarrow$  100433627766186892221372630771322662657637687111424552206336 calls  $\rightarrow$  200867255532373784442745261542645325315275374222849104412672 calls  $\rightarrow$  401734511064747568885490523085290650630550748445698208825344 calls  $\rightarrow$  803469022129495137770981046170581301261101496891396417650688 calls  $\rightarrow$  1606938044258990275541962092341162602522202993782792835301376 calls  $\rightarrow$  3213876088517980551083924184682325205044405987565585670602752 calls  $\rightarrow$  6427752177035961102167848369364650410088811975131171341205504 calls  $\rightarrow$  12855504354071922204335696738729300820177623950262342682411008 calls  $\rightarrow$  25711008708143844408671393477458601640355247900524685364822016 calls  $\rightarrow$  51422017416287688817342786954917203280710495801049370729644032 calls  $\rightarrow$  102844034832575377634685573909834406561420991602098741459288064 calls  $\rightarrow$  205688069665150755269371147819668813122841983204197482918576128 calls  $\rightarrow$  411376139330301510538742295639337626245683966408394965837152256 calls  $\rightarrow$  822752278660603021077484591278675252491367932816789931674304512 calls  $\rightarrow$  1645504557321206042154969182557350504982735865633579863348609024 calls  $\rightarrow$  3291009114642412084309938365114701009965471731267159726697218048 calls  $\rightarrow$  6582018229284824168619876730229402019930943462534319453394436096 calls  $\rightarrow$  13164036458569648337239753460458804039861886925068638906788872192 calls  $\rightarrow$  26328072917139296674479506920917608079723773850137277813577744384 calls  $\rightarrow$  52656145834278593348959013841835216159447547700274555627155488768 calls  $\rightarrow$  105312291668557186697918027683670432318895095400549111254310977536 calls  $\rightarrow$  210624583337114373395836055367340864637790190801098222508621955072 calls  $\rightarrow$  421249166674228746791672110734681729275580381602196445017243910144 calls  $\rightarrow$  842498333348457493583344221469363458551160763204392890034487820288 calls  $\rightarrow$  1684996666696914987166688442938726917102321526408785780068975640576 calls  $\rightarrow$  3369993333393829974333376885877453834204643052817571560137951281152 calls  $\rightarrow$  6739986666787659948666753771754907668409286105635143120275902562304 calls  $\rightarrow$  13479973333575319897333507543509815336818572211270286240551805124608 calls  $\rightarrow$  26959946667150639794667015087019630673637144422540572481103610249216 calls  $\rightarrow$  53919893334301279589334030174039261347274288845081144962207220498432 calls  $\rightarrow$  107839786668602559178668060348078522694548577690162289924414440996864 calls  $\rightarrow$  215679573337205118357336120696157045389097155380324579848828881993728 calls  $\rightarrow$  431359146674410236714672241392314090778194310760649159697657763987456 calls  $\rightarrow$  862718293348820473429344482784628181556388621521298319395315527974912 calls  $\rightarrow$  1725436586697640946858688965569256363112777243042596638790631055949824 calls  $\rightarrow$  3450873173395281893717377931138512726225554486085193277581262111899648 calls  $\rightarrow$  6901746346790563787434755862277025452451108972170386555162524223799296 calls  $\rightarrow$  13803492693581127574869511724554050904902217944340773110325048447598592 calls  $\rightarrow$  27606985387162255149739023449108101809804435888681546220650096895197184 calls  $\rightarrow$  55213970774324510299478046898216203619608871777363092441300193790394368 calls  $\rightarrow$  110427941548649020598956093796432407239217743554726184882600387580788736 calls  $\rightarrow$  220855883097298041197912187592864814478435487109452369765200775161577472 calls  $\rightarrow$  441711766194596082395824375185729628956870974218904739530401550323154944 calls  $\rightarrow$  883423532389192164791648750371459257913741948437809479060803100646309888 calls  $\rightarrow$  1766847064778384329583297500742918515827483896875618958121606201292619776 calls  $\rightarrow$  3533694129556768659166595001485837031654967793751237916243212402585239552 calls  $\rightarrow$  7067388259113537318333190002971674063309935587502475832486424805170479104 calls  $\rightarrow$  14134776518227074636666380005943348126619871175004951664972849610340958208 calls  $\rightarrow$  28269553036454149273332760011886696253239742350009903329945699220681916416 calls  $\rightarrow$  56539106072908298546665520023773392506479484700019806659891398441363832832 calls  $\rightarrow$  113078212145816597093331040047546785012958969400039613319782796882727665664 calls  $\rightarrow$  226156424291633194186662080095093570025917938800079226639565593765455331328 calls  $\rightarrow$  452312848583266388373324160190187140051835877600158453279131187530910662656 calls  $\rightarrow$  904625697166532776746648320380374280103671755200316906558262375061821325312 calls  $\rightarrow$  1809251394333065553493296640760748560207343510400633813116524750123642650624 calls  $\rightarrow$  3618502788666131106986593281521497120414687020801267626233049500247285301248 calls  $\rightarrow$  7237005577332262213973186563042994240829374041602535252466099000494570602496 calls  $\rightarrow$  14474011154664524427946373126085988481658748083205070504932198000989141204992 calls  $\rightarrow$  28948022309329048855892746252171976963317496166410141009864396001978282409984 calls  $\rightarrow$  57896044618658097711785492504343953926634992332820282019728792003956564819968 calls  $\rightarrow$  115792089237316195423570985008687907853

U-F is weighted union rule + path compression  
 with union by rank

$C(m, a(m, n)) \rightarrow$  αυθαίρετα Αδελφώματα

αυθαίρετα 2 τεταγμένα

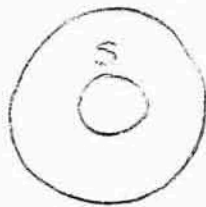
$$A(m, n) = 2^{2^{m-2}} \cdot 2^{n-2} \} \text{ σε } 536$$

$$a(m, n) \leq A^{-1}(m, n) \text{ όταν } \log a(n) < 2^{2^{m-2}} \cdot 2^{n-2} \leq 4$$

12/1/2004

Κατακερτατισμός (Hashing)

$U$  (σύνολο)

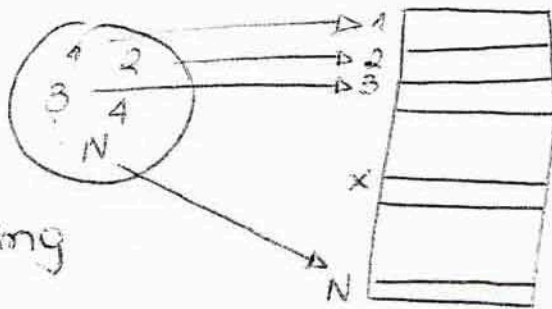


insert(x) :  $x \in U : S = S \cup \{x\}$

delete(x) :  $S = S - \{x\}$

access(x) η search(x)

Εστω  $|U| = N$



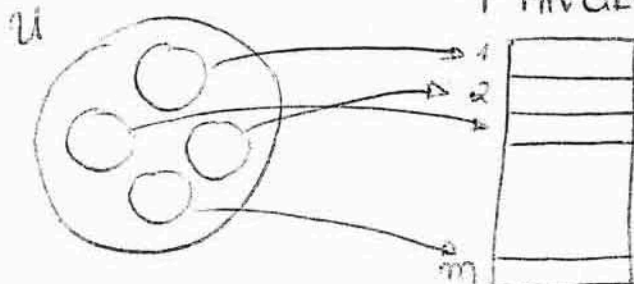
direct  
addressing

(+) insert  $O(1)$   
 delete  
 search

(-) μπορεί να μη δώσει η τιμή  
 $|S| \ll |U|$  σπατάλη τιμών

Εστω ότι σταθερούμε μια περιορισμένη τιμή

$T$  πίνακας κατακερτατισμού



$h : U \rightarrow \{0, \dots, m-1\}$

συνάρτηση κατακερτατισμού

Εάν το  $U$  χωριστεί στοιχεία από δύο ομάδες, τότε έχουμε σύγκρουση (collision).





Ορίζω  $weight(x) = \# \text{ απόγονων } x$  και  $rank(x) = \text{λογος θέσης } x$  στην σειρά.

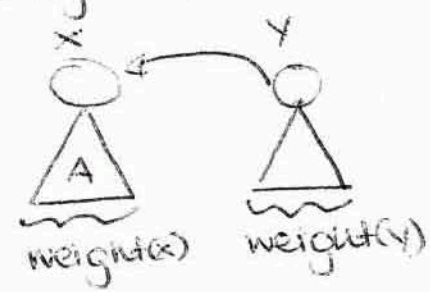
1)  $weight(x) \geq 2^{rank(x)}$

αποδείξτε με επαγωγή

a)  $rank(x)=0$  τότε  $2^{rank(x)} = 1 = weight(x)$

b)  $weight(x) = \# \text{ απόγονων } x \text{ πριν ενώσει } y + \# \text{ απόγονων } y$

$$\begin{aligned} &\geq 2 * weight(y) \geq (\text{ελάχιστο υποδέντρο}) \\ &\geq 2 * 2^{rank(y)} \geq 2 * 2^{rank(x)-1} = \\ &= 2^{rank(x)} \end{aligned}$$



2)  $rank(x) \leq \log n$ ,  $n-1$  unions

Εκεί που δείχνει ότι  $2^{rank(x)} \leq weight(x) \leq n \Rightarrow$

$$rank(x) \leq \log n$$

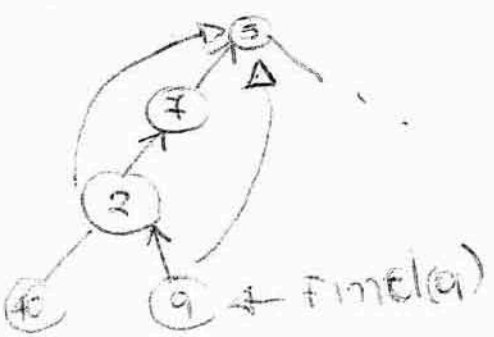
$\Rightarrow$   
 μέγιστο μέγεθος  
 δέντρου με  
 $n-1$  ενώσεις

rank  $\rightarrow$  αριθμός βημάτων που χρειάζονται για να φτάσουμε στο root.

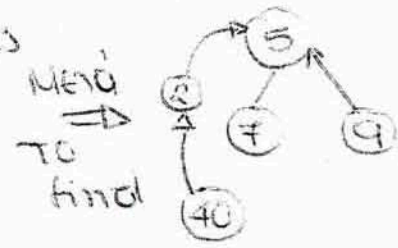
$n-1$  unions και  $m$  find

χρόνος  $\approx O(n + m \log n)$   $O(m + n \log n)$

Path compression



η ταχύτερη μέθοδος για να βρούμε το root



V-F is weighted union rule + path compression  
 w-1 union rule in find union.

$C(m, a(m, n)) \rightarrow$  given access time

conclusion 2 parameters

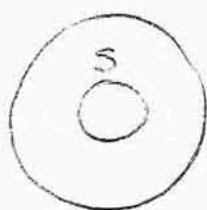
$$A(5, 4) = 2^{2-2} = 1$$

$$A(m, n) \approx A^{-1}(m, n) \text{ then } \log m < 2^{2-2} A(m, n) \leq 4$$

12/1/2004

## Κατακερτάσιος (Hashing)

$U$  (σύνολο)

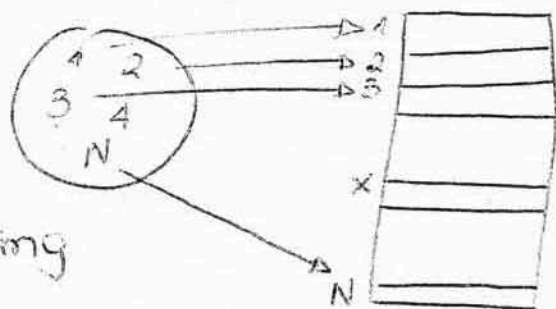


$\text{insert}(x) : x \in U : S = S \cup \{x\}$

$\text{delete}(x) : S = S - \{x\}$

$\text{access}(x) \text{ or } \text{search}(x)$

Εστω  $|U| = N$



direct  
addressing

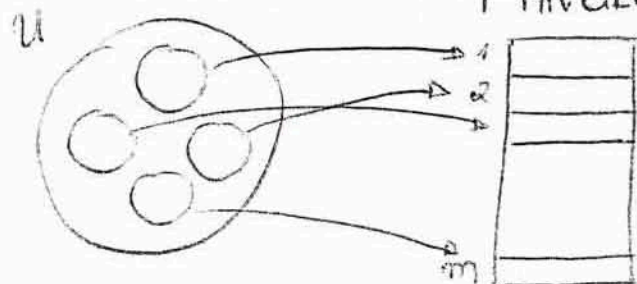
(+) insert  $O(1)$

delete  
search

(-) μπορεί να μη φτάσει η τιμή  
 $|S| \ll |U|$  απαιτείται τιμή

Εστω ότι σταθερούμε μια περιορισμένη τιμή

$T$  πίνακας κατακερτάσιος



$h : U \rightarrow \{0, \dots, m-1\}$

συνάρτηση κατακερτάσιος

Εάν το  $U$  χωριστεί στοιχεία από δύο ομάδες, τότε έχουμε σύγκρουση.  
 (collision).

Κατακεφαλοποιείας. Έστω  $U = \{0, 1, \dots, N-1\}$  και συνόλεος  $T (|T|=m)$  και  $h: U \rightarrow \{0, 1, \dots, m-1\}$  τότε  $x \in U \rightarrow T[h(x)]$

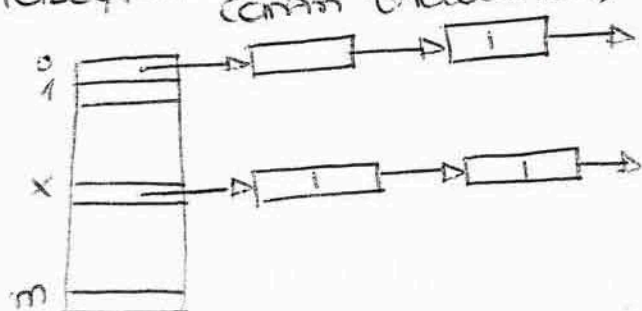
π.χ.  $m=5, U = \{0, 1, 2, \dots, 99\}$

$S = \{3, 15, 20, 22, 24\}$

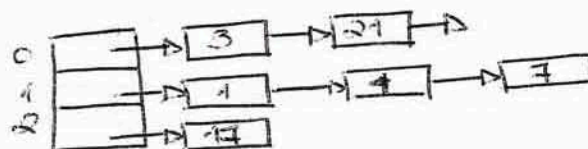
$h = x \bmod 5$

x	h(x)	T
		0 15 *collision
3	3	1
15	0	2 22
20	0	3 3
22	2	4 24
24	4	

Κατακεφαλοποιείας (chained hashing) ή αλυσίδες (αλληλ. συν. συνόλεος)



π.χ.  $m=3, U = \{0, 1, \dots, 99\}$   
 $S = \{1, 3, 4, 7, 10, 17, 21\}$   
 $h = x \bmod 3$



$O(m + |S|)$

Χρησιμοποιώντας αλυσίδες, έχουμε δυνατότητα σε πράξεις

(1) Εισαγωγή  
 $insert(x) \{$

$\rightarrow T[h(x)] \rightarrow push(x)$   
 $\}$

$search(x) \{$   
 $T[h(x)] \rightarrow search(x)$   
 $\}$

$O(|S|)$   
 Λογικός χρόνος

$delete(x) \{$   
 $search(x);$   
 $T[h(x)] \rightarrow delete(x);$   
 $\}$

$O(|S|)$   
 Λογικός χρόνος

Κατακεφαλοποιείας με hash, έχουμε σε πράξεις

Παράδειγμα: ΜΕΘΟΔΟΣ ΤΕΡΜΙΝΩΣΗΣ

Παράδειγμα

$h: U \rightarrow \{0, \dots, m-1\}$  στοιχείο

$K$  πράξεις  $\{$  για από τις βασικές πράξεις που περιγράφονται  $\}$   
 ή την ίδια διαδικασία

Αν για  $K$  πράξεις, απαιτείται χρόνο  $n$ , τότε ο μέσος χρόνος είναι  $n/K$ .



• Κάθε κελύ, σε κάθε ωράνη και στοιχείο, je την ίδια πιθανότητα

Οι λίστες θα αναδιατάσσονται χρονιά-εξοδα.

Θεωρήτα  $\rightarrow$  Οι κ πράξεις  $O(k(1+b/2))$  [Η αναδιατάξη εκτός]

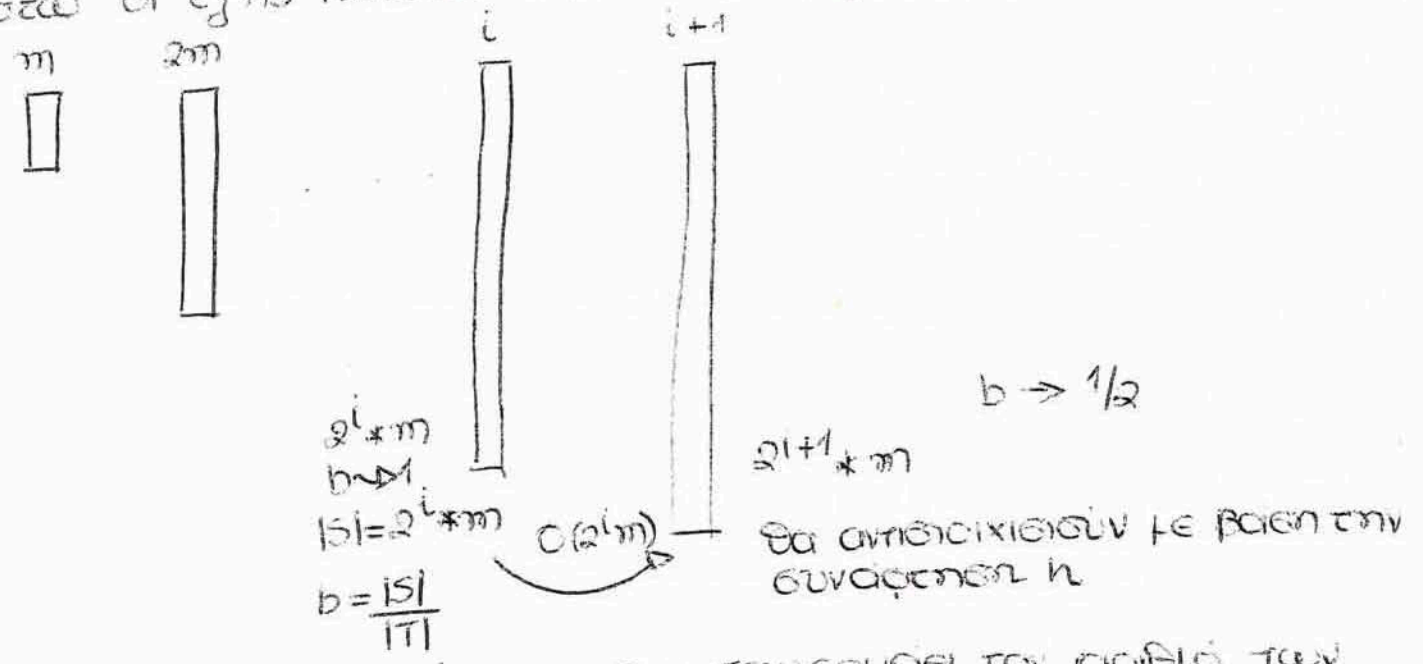
$b = \frac{|S|}{m}$  load factor

•  $b \rightarrow 1$  τότε  $|S| \approx m$ ,  $O(1) \rightarrow O(k)$   $\xrightarrow{\text{ή για πράξη}} O(1)$

$\hookrightarrow$  Δεν είναι κατά για real time εφαρμογές.

•  $b \gg 1$  τότε  $|S| \gg m$  (Οι λίστες μεγαλώνουν σε μήκος  $\rightarrow$  κακή συμπεριφορά)

Έστω οι εγής πίνακες κατακεφαλαίωτοι:



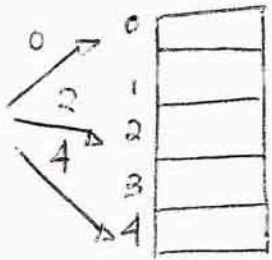
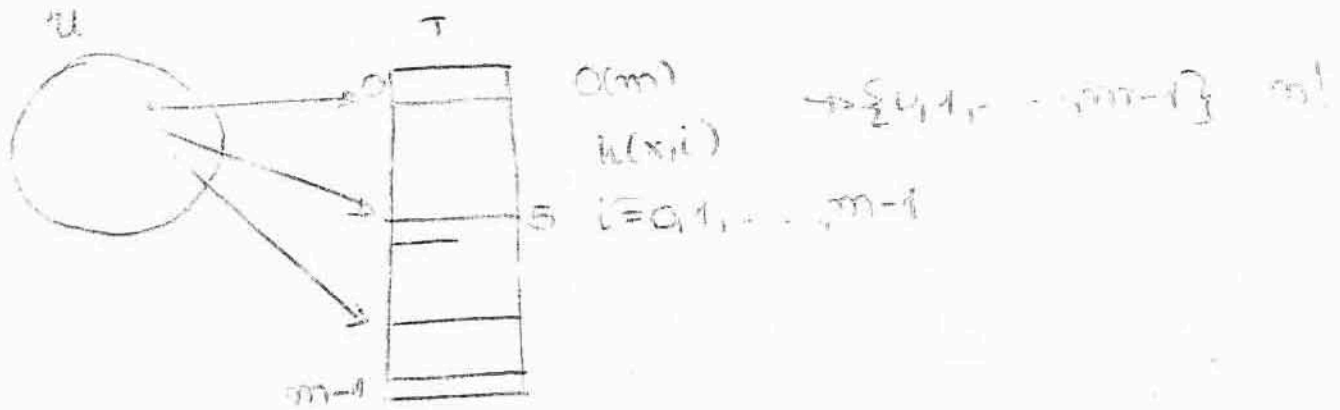
Το κόστος της μεταβάρης δεν θεωρείται τον αριθμό των πράξεων που χρειάζονται για την μεταβάρη

Υπάρχουν 2 τρόποι για να εξελεγεθεί x πράξεις:

- (1) όλες μαζί
- (2) κάθε φορά για χωριστή πράξη και για όλες τις ερωτήσεις (insert, delete, access)

Κατακεφαλαίωτος με ανοικτή διεύθυνση (open addressing hashing)





Insert( $x$ )  $\{$   $O(m)$

$i \leftarrow 0$ ;  
while ( $T[h(x, i)] \neq \text{NULL} \parallel i < m$ )  $i++$ ;

if ( $i == m$ ) fail

else  $T[h(x, i)] = x$ ;

$\}$   $\rightarrow$  loopel ua ghele o wluakos (peroveronta)

search( $x$ )  $\{$

$i \leftarrow 0$ ;

while ( $T[h(x, i)] = \text{NULL} \parallel i < m$ )  $\{$

if ( $T[h(x, i)] == x$ ) return  $h(x, i)$ ;

else  $i++$ ;

$\}$

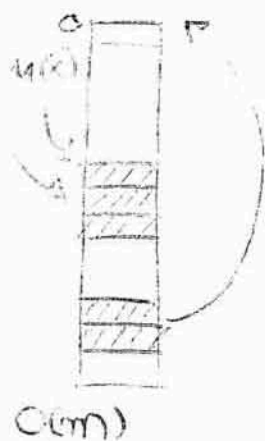
delete( $x$ )  $\{$

if ( $\text{pos} = \text{search}(x) \neq \text{NULL}$ )  $T[\text{pos}] = \text{NULL}$ ;

$\}$

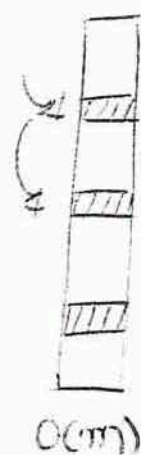
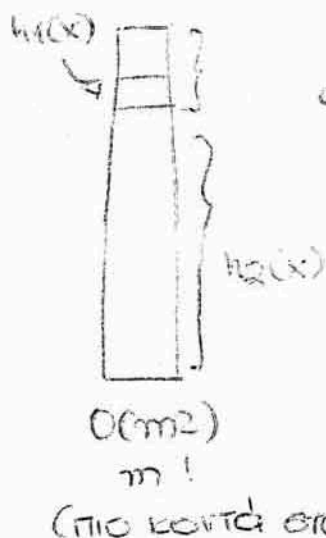
Linear probing

$$h(x, i) = (h_1(x) + i) \bmod m$$



quadratic probing  $h(x,i) = (h(x) + c_1 + c_2 i^2) \bmod m$

double hashing  $h(x,i) = (h_1(x) + i h_2(x)) \bmod m$



Θεωρητά  $\rightarrow$  0 φορές χρόνος αναζήτησης  
έρευνας & 1 φορά χρόνο εγγραφής:

$$1/(1-b) \quad b = \frac{|S|}{m} \quad b \leq 1 \quad |S| \leq m$$

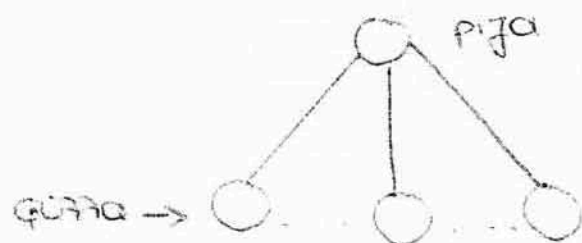
Θεωρητά  $\rightarrow$  0 φορές χρόνο αναζήτησης έρευνας

$$\frac{1}{b} \ln \frac{1}{1-b} + \frac{1}{b}, b < 1$$

$|S| \rightarrow$  μέγεθος ακολουθίας την αξία  $h(x)$   
 $x < S, \nexists x, y \quad h[x] = h[y]$

13/01/2004

Δέντρα  $\Rightarrow$  Δομές δεδομένων



minimum spanning tree  $\rightarrow$  όταν φέρουμε όλες τις ωνυμίες (edges)  
και φέρω ωνυμολογιστική την μικρότερη.

search  
insert  
delete  $\} \leq \log n$

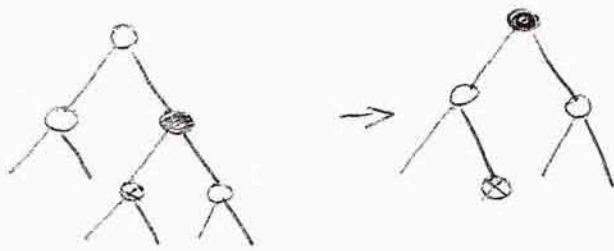
## AVL-tree (1964)

searching

Hashing

IST  $\rightarrow \log \log n$  $O(1)$ BB-tree  $\rightarrow$  Rebalancing operations σε constant time.

BB[+]

 $\rightarrow$  BB-treeφθίς "πράγες"  $\log n$ 

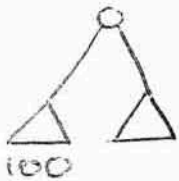
ομαζω δομή  $\rightarrow$  "ακριβή πράγες"  $O(1)$   
 Αλλά, στο AVL-tree (εξού  $O(\log n)$ )

 $\rightarrow$  AVL-tree

- Βαροζυγισμένα  $\rightarrow$  BBA
- Υψοζυγισμένα  $\rightarrow$  IST

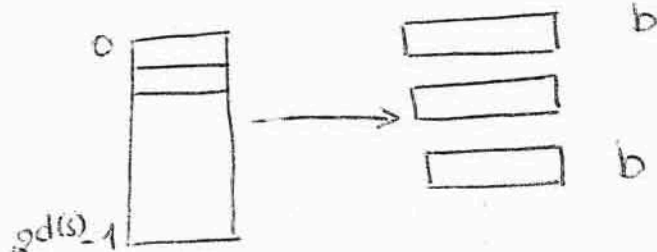
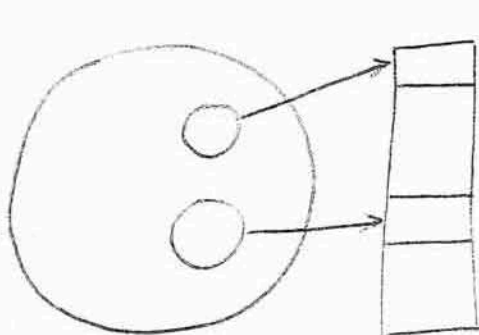
Βάρος  $\rightarrow$  το άθροισμα των ενοχλείων σε ένα δένδρο

π.χ.



βάρος = 100

19/01/2004



Τα ενοχλεία αντιστοιχίζονται  
 σε έναν double αριθμό το  
 καθένα.

Μας ενδιαφέρει ο αριθμός των υποενοχλείων  
 και γράφονται block.

$h: U \rightarrow \{0,1\}^k, \exists x,y \quad h(x)=h(y)$

$h_d: U \rightarrow \{0,1\}^d, d \leq k, x \in U \quad h_d(x) = \text{prefix}_d(h(x))$   
 $\rightarrow 1101 \quad h(1101) = 11$   
 $\rightarrow 1110 \quad h(1110) = 11$

x	$h(x)$	$h_1(x)$	$h_2(x)$	$h_3(x)$
0	0000	0	00	
1	0001	0	00	
4	0100	0	01	
8	1000	1	10	
12	1100	1	11	

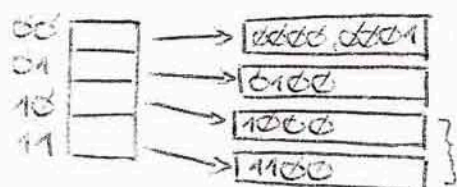
ορίκο βάθος  $d(S)$

όπου  $S$  τα στοιχεία που έχουμε αναθέσει στην δατή.

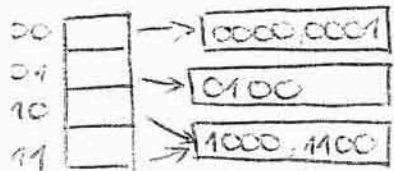
π.χ.  $b=2$

$$S = \{0, 1, 4, 8, 12\}$$

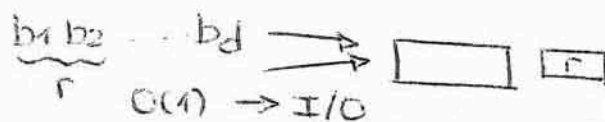
$$h(S) = \{0000, 0001, 0100, 1000, 1100\}$$



για ενοποίηση του πίνακα, μπορούμε να αναθέσουμε αυτά τα στοιχεία στο ίδιο block



Βλέπουμε ότι δεν έχουν το ίδιο πρόθεμα



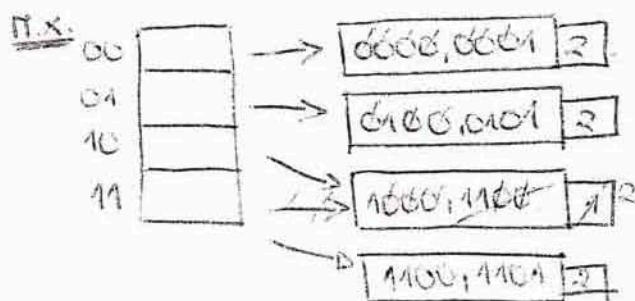
Search(x) {  
 $i_{BIN} = h_d(S)(x)$ ;  
 γράψτε B T[iBIN]  
}

insert(x) {  
 $i_{BIN} = h_d(S)(x)$ ;  
 $B \leftarrow T[i_{BIN}]$   
 if  $(|B| < b)$  επάγει  $x \rightarrow B$   
 else if  $(i_B \neq d(S))$

$B \rightarrow B'$

$O(2^{d+1})$

else  $d++$ , γράψτε T

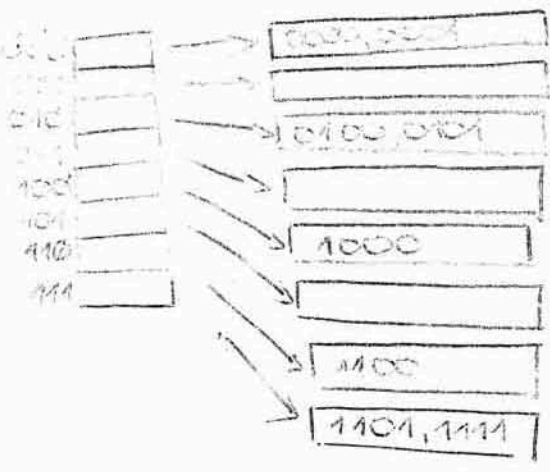


insert(0101)

insert(1101)

Δεν έχουμε πάθος τώρα





Αντιστοιχία με τις  
εξωτερικά φύλλα  
κατακερματισμού

$\mu \in \mathbb{R} \rightarrow b$

delete(x) {  
 $i_{BIN} = hd(s)(x)$ ;  
 $B \leftarrow T[i_{BIN}]$   
 if (x υπάρχει στο B) διαγράψτε x;  
 $B'$  για το οποίο υπάρχει ίδιο πρόβλημα  
 if ( $|B'| + |B| < b$ )  $B = B \cup B'$   
 Αν ( $r_B = r_{B'} = d(s)$ ) γυρνώμεθα ο T,  $d = d - 1$   
 και όχι άλλη επεξεργασία  
}

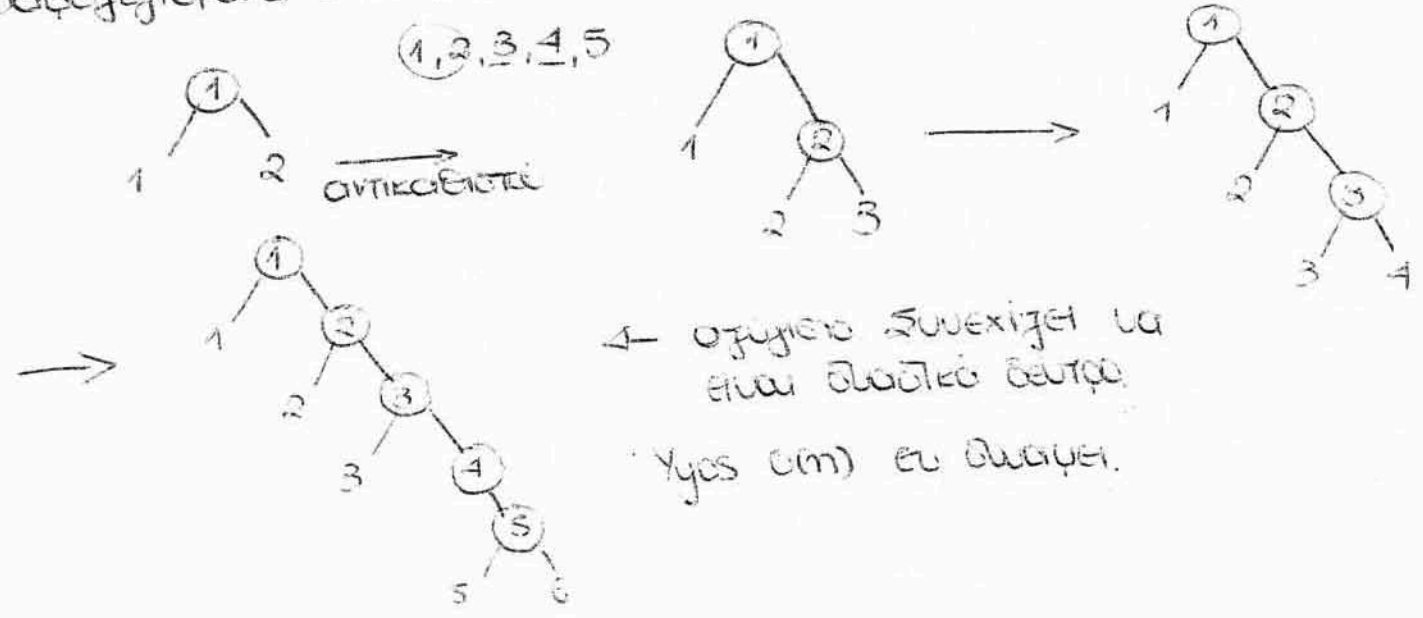
$b_1 b_2 b_3 \dots b_{r-1} \rightarrow$  όπου r το μικρό βάθος

Θ.1  $\rightarrow$  Μέσο αριθμός περιόδων είναι  $n/b \ln 2$  και το μέσο μήκος  
 κατακερματισμού  $|T| = \left( e/b \ln 2 \right) n^{1+1/b}$ .

$\rightarrow$  Red - black δέντρα  $\leftarrow$

Υποχρεωτικά : AVL, Red-black

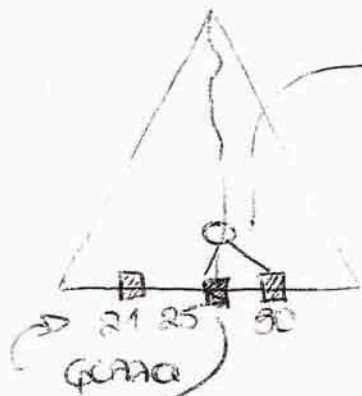
Βασισμένα : BB[a]



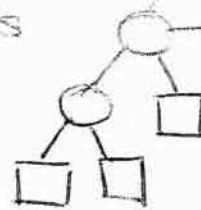
$\leftarrow$  ορίστηκε συνειδητά να  
 είναι ελαστικό δέντρο.  
 Υπάρχει όμως το θέμα.



Πόσο θα διαφέρει η ανώτερη ευρεία σε ένα red-black tree  
 σε σχέση με τον αριθμό φύλλων;  $\Delta h \in (\log n)$  (Εάν το ελάχιστο +  
 το μέγιστο  $\log n$ )

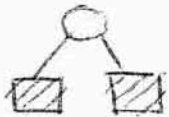


Ο θα μπορούσε  
 ο κόμβος  
 να είναι κόκκινος  
 (προπατήρας)



ΑΝΤΙΜΕΤΩΠΙΣΤΕ  
 ΤΑΥΤΟΣ

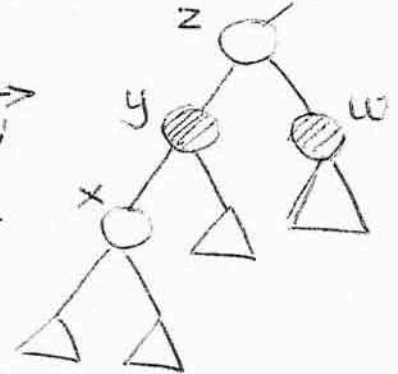
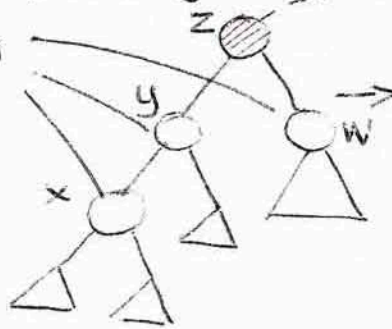
Αντικαθιστάτε  
 το κόμμο ΓΑΡΑ με  
 έναν κόκκινο κόμμο  
 ο οποίος έχει 2 κόμμοι  
 ΓΑΡΑ



→ ΑΝΤΙΜΕΤΩΠΙΣΤΕ ←

1) ΑΝΤΙΜΕΤΩΠΙΣΤΕ ΧΡΩΜΑΤΟΣ

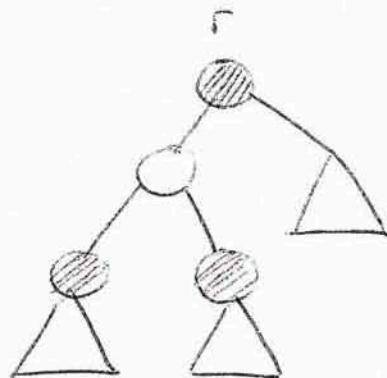
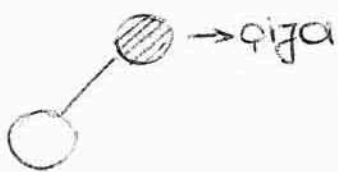
ΚΟΚΚΙΝΟΙ



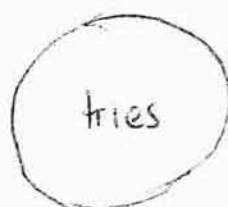
ΚΑΙΝΕ  
 ΕΧΘΡΟΝ  
 ΧΡΩΜΑΤΟΣ

→ ΚΟΚΚΙΝΟ

→ ΚΟΚΚΙΝΟ

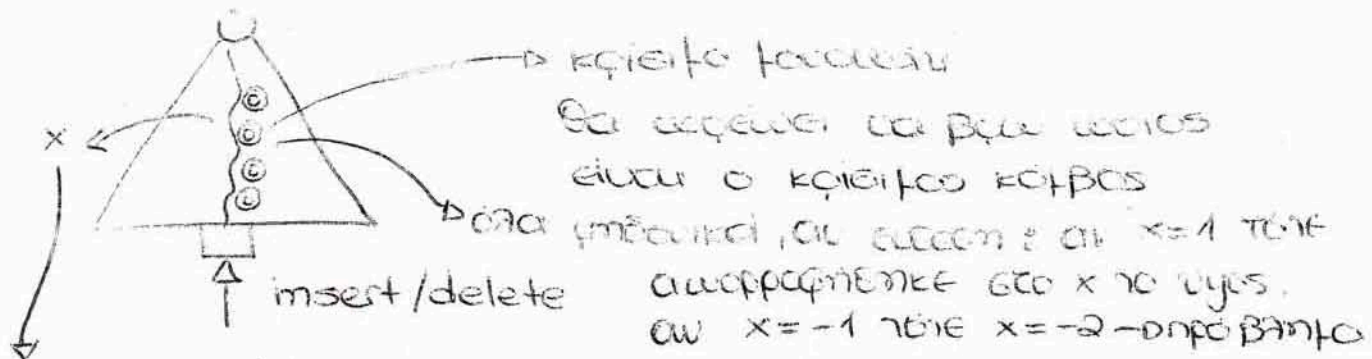


20/04/2004



Βασιλειάδης

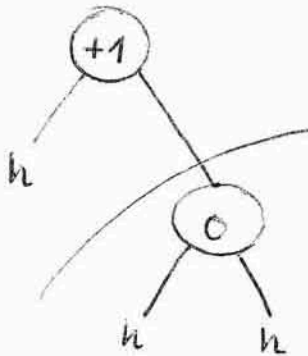
Αποδοτικότητα υψος  $\rightarrow$  το ποσό  $1.44 * \log n$



Κρίσιμος κόμβος εύρεσης

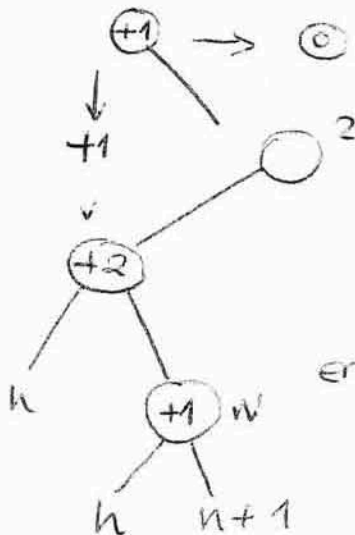
$\rightarrow$  σ' αυτόν ελέγχω, αν το δέντρο χρειάζεται rebalance.

ΚΚΕ

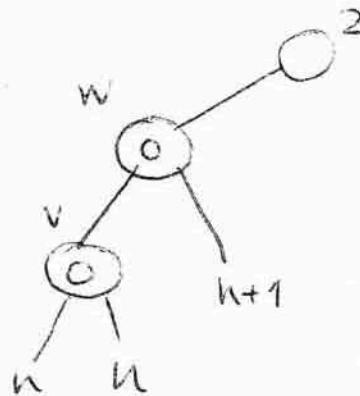


- ο τρέξι εδώ να έχει όλα γινόμενα (ΚΚΕ)

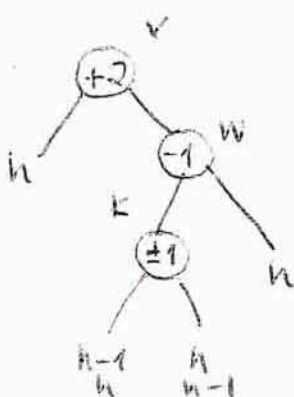
Αν είναι +2, θα αρθρώσει να βάλουμε ελαστικές.



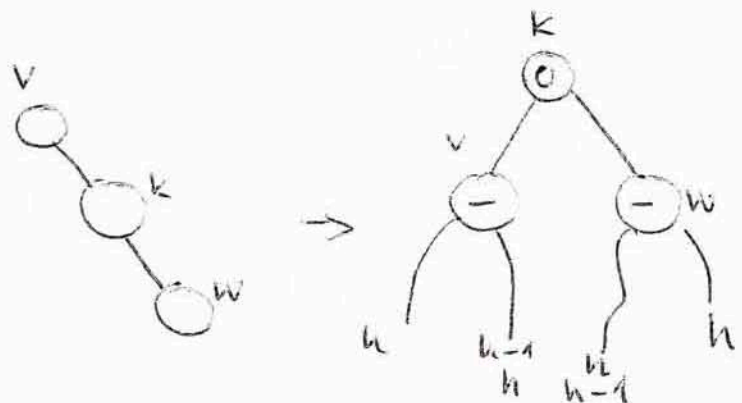
$\Rightarrow$  ελαστική



(αυτήν επεξεργασία)

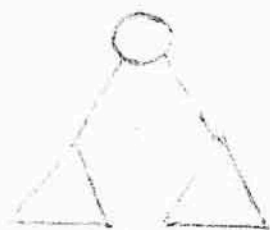


$\rightarrow$



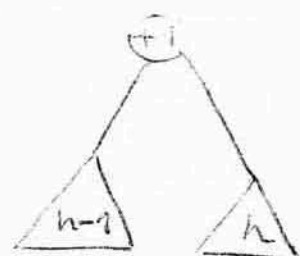
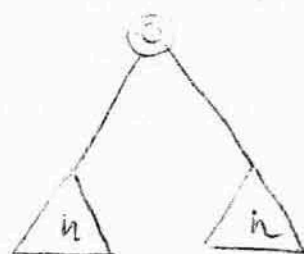
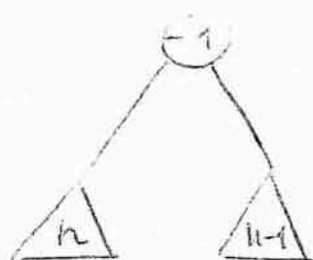
(αυτήν επεξεργασία)





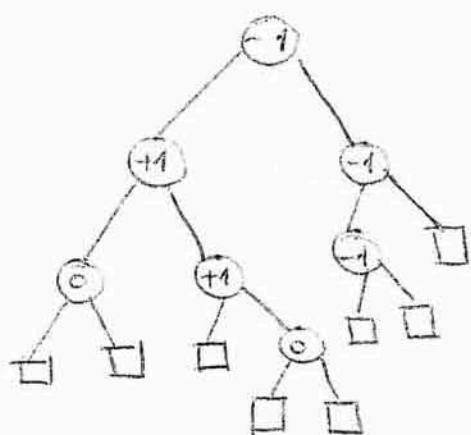
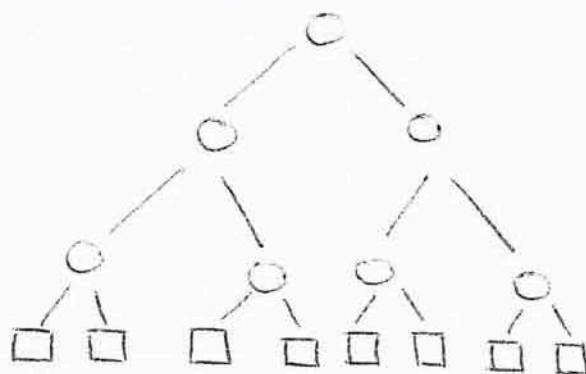
$\pm 1, 0$

→ εγχαρτισμένο δέντρο



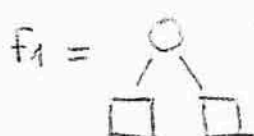
Το δέντρο έχει <sup>①</sup> εγχαρτισμένο υψος, και insert/delete σε  $\log n$  στο worst case <sup>②</sup>

η στοιχεία  $\rightarrow$   
 η ελάχιστη ύψος  
 στο πλήρες  
 εγχαρτισμένο  
 δέντρο.  
 $n = 2^k$



Fibonacci δέντρο

$f_0 = \square$



...  $f_n =$



3) Top-down rebalancing

4) Αν ένας κόμβος βρίσκεται σε ύψος  $h$ , σε μια ακολουθία  $n$  πορτρέτων θα ανακατασκευαστεί  $O(n/h)$  φορές

5) Ξύλον 1-bit

### TRIES

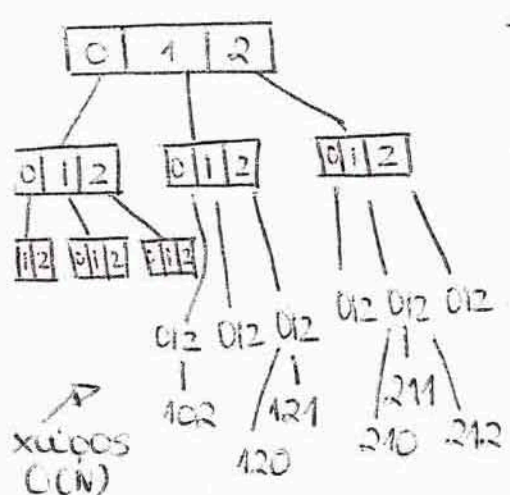
Υδροδομές κόμβων δοσμένων

Εστω  $S = \{121, 102, 211, 120, 210, 212\}$

$S \subseteq V$  (δεν είναι)

$\Sigma^3 = V \Rightarrow \Sigma = \{0, 1, 2\}$

έχουν μήκος 3



→ κάθε επίπεδο του δέντρου, έχει έναν αριθμό 1-bit συγκεκριμένο bit.

→ Η ρίζα έχει τρεις επιλογές όλα είναι τα υψήλια

$$|S| = n$$

$|V| = N$  —————> πλήρες τετράγωνο του στοιχείου που έχω

$N = k^l$  —————> όλοι οι αριθμοί συμπεριλαμβάνονται

↓  
k χαρακτήρες του αλφαριθμητικού

$$\Rightarrow l = \log_k N$$

Αναζήτηση  $O(l) = O(\log_k N)$

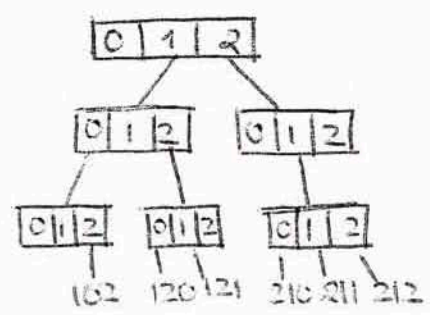
Αντικαθιστά το RAM φαίνεται υπολογιστικό γιατί χρησιμοποιείται μνήμη. Μπορεί να χρησιμοποιηθεί και για το φακέλο pointer machine, χρησιμοποιώντας λίστες.

PM  $O(k \cdot \log_k N)$

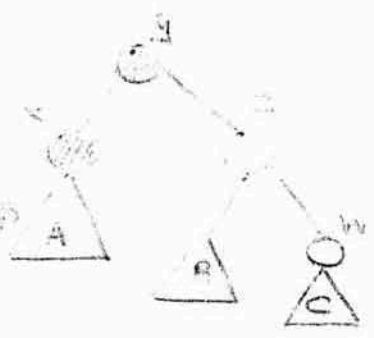
Τι θα μπορούσε να κάνει για να μειώσει το χώρο?

Κόβω τα κλάδα της δομής στα οποία θα αυξάνεται και.

$O(l \cdot k \cdot m)$



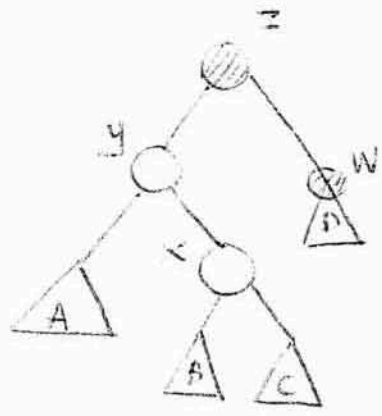
Είσοδος  
ή  
επιβεβαίωση  
της  
working set



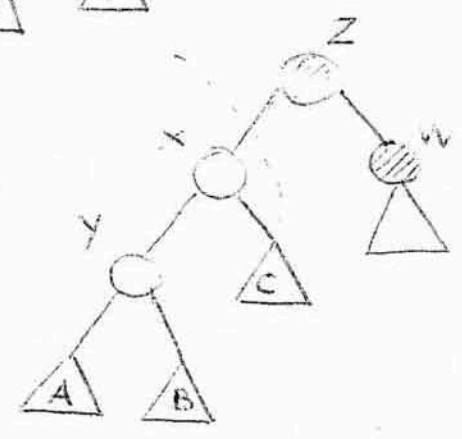
Αφίσχυση  
→  
περίστωση



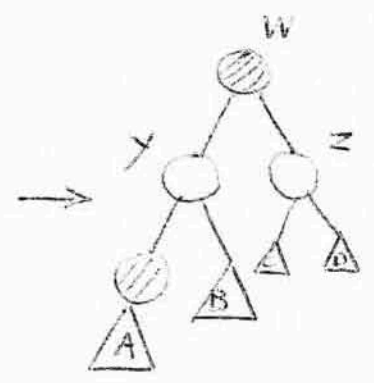
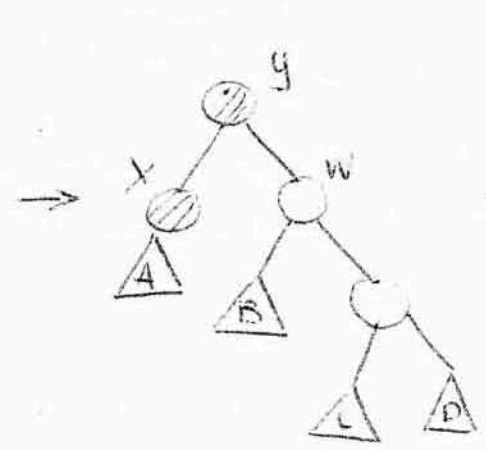
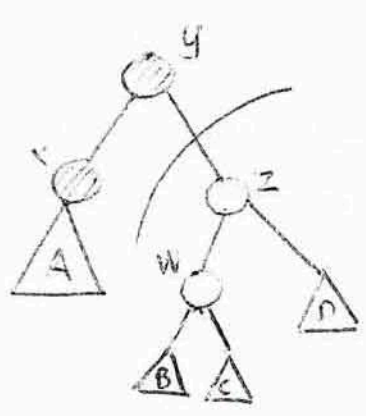
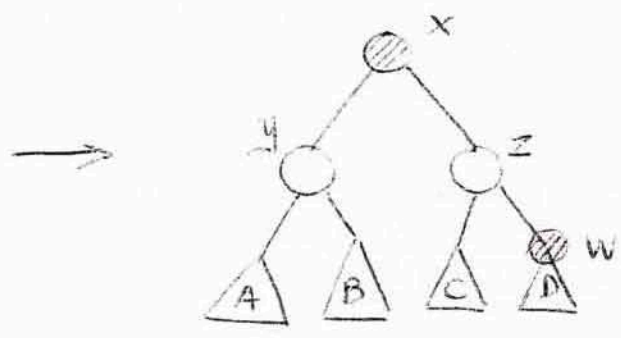
6)



Δίνεται  
→  
Περίστωση  
(xy προς  
αφίσχυση)

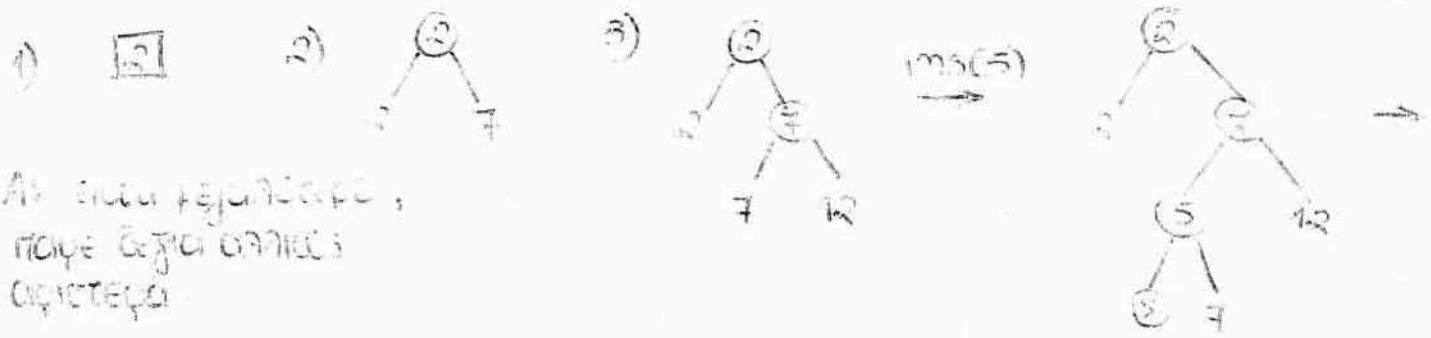


Η επιβεβαίωση δαίνει την άδεια

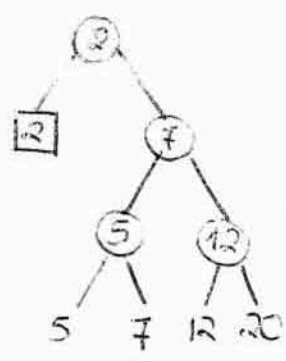


### ΙΔΙΟΤΗΤΕΣ (SOS)

- 1) Μετά από + εισαγωγή/αφαίρεση χρειάζονται O(log n) αλλαγές πρώτης σειράς (m) βήματα.
- 2) Συνολικό κόστος μιας ακολουθίας από m εισαγωγές/αφαίρεσεις είναι O(m).



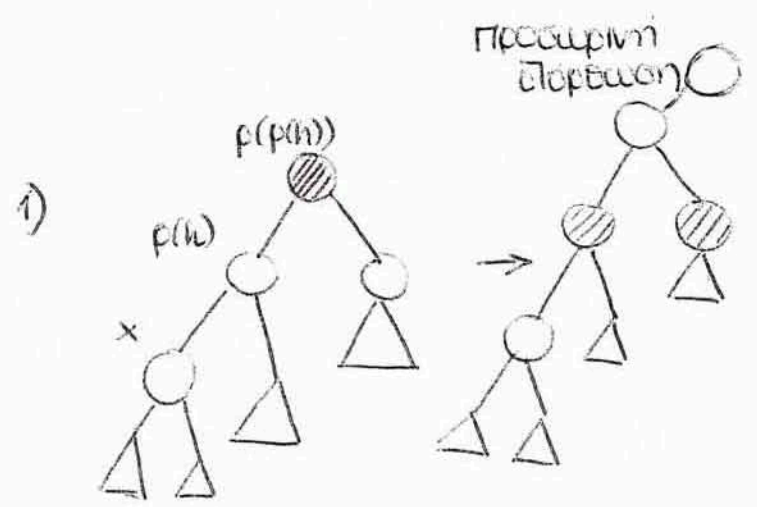
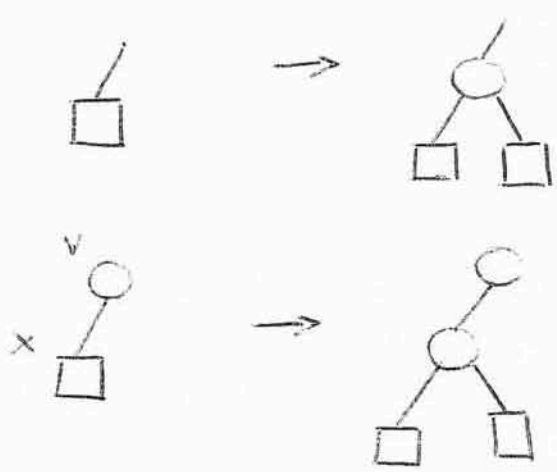
Αν είναι τετραπαική,  
παιρά βέρια αριστερά  
αριστερά



Αλλαγές heights  
 → AVL  $0 \rightarrow \pm 1$   
 → color flip

- S. Sahni  
Data structures in C++
- R. Sedgewick  
Algorithms in C++

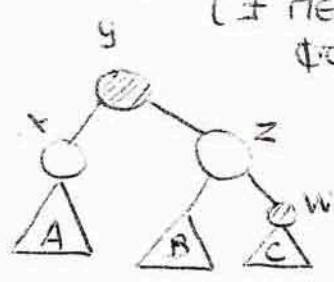
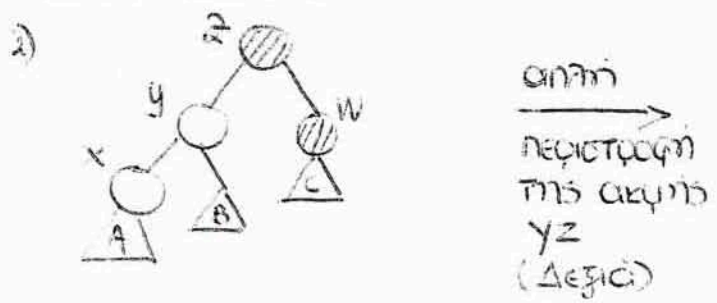
Red-black βέρια



εναλλαγή χρώματος  
(τη τετραπαική)

Όταν ένα είναι τετραπαική,  
επανεξέταση ως προς τα πάνω.  
(Ε περίπτωση να  
φτάσει στη ρίζα)

Δομές αλλαγής (περιτρώφης)



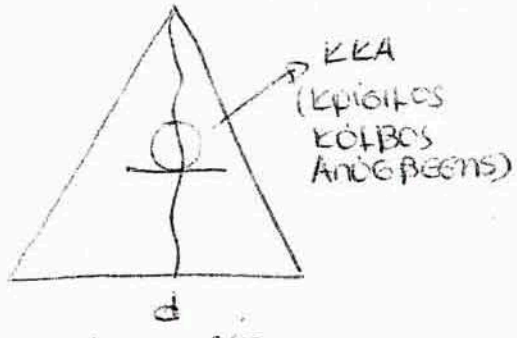
Τετραπαική για τη ρίζα  
ωραφισμένη ταύρα



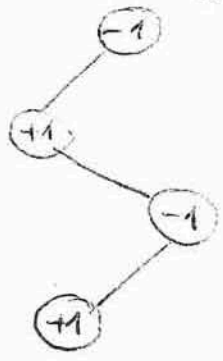
	genus entry	applies entry
i	$\log n$	$O(n)$
d	$\log n$	$\log n$

BB-tree

for the range delete



επιπλέον κόμβος απόψεως

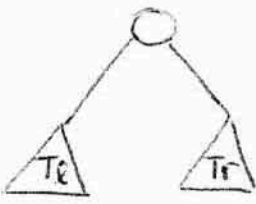


	genus entry	applies entry	amortized
i	$\log n$	$O(n)$	$O(n)$
d	$\log n$	$O(n)$	$O(n)$

BB-tree amortized

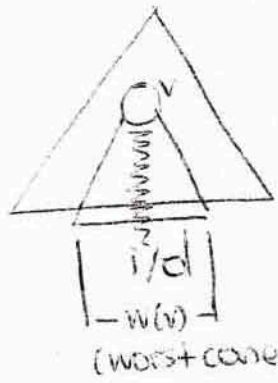
3n 4n

Get a → delete, IST



$$\frac{1}{3} < p = \frac{|T_l|}{|T_r|} < \frac{2}{3}$$

(π-a) (a)



→ case not balanced, then we apply the rebalancing operation (rotation of the tree)



# Interpolation Search Tree

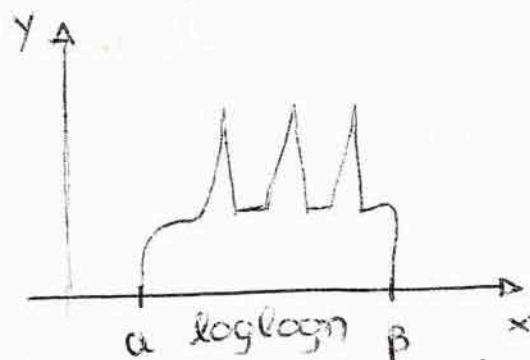
• Melhorn - Tarauvidis (1943) → JALM

Dynamic Interpolation Search

$O(\log \log n)$

## ΔΙΑΙΣΤΗΤΕΣ

1. Χώρος  $O(n)$  και κτιζεται σε  $O(n)$  χρόνο.
2. Αναζήτηση  $O(\log \log n)$  χρόνο στη πέρα περίπτωση για smooth  
[Τα στοιχεία που εισάγουμε, οι από τους υποκείμενους κάποιες  
συγκεκριμένες κατανομές.  
κατανομές.

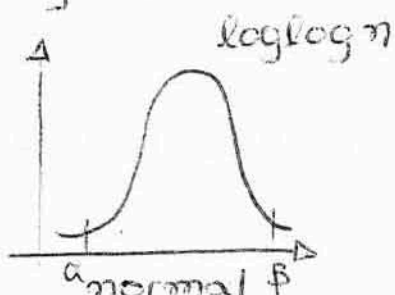


$y \rightarrow$  πιθανότητα εμφάνισης  
κάποιας τιμής

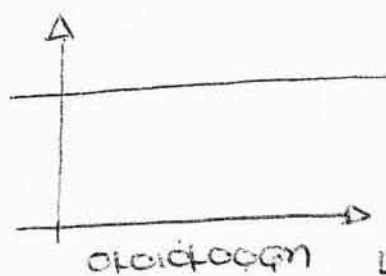
$x \rightarrow$  εύρος τιμών

Η πιθανότητα εμφάνισης είναι σχεδόν σταθερή στη συνολική  
διαμετρία.

Η πιθανότητα είναι φραγμένη



(smooth κατανομή)



4. Ομοιογένεια υπή έχει  
την ίδια πιθανότητα να  
εμφανιστεί από ένα διαμετρία  
τιμών.

↳ η αναζήτηση της παίρνει  
 $\log \log n$

Τα παραπάνω ισχύουν για τις εισαγωγές.

Διαλέγει με την ίδια πιθανότητα όλες τις από  
είναι ώστε να μπορεί να σταθεροποιηθεί.

Τυχόν στοιχείο.

## Διαφορές

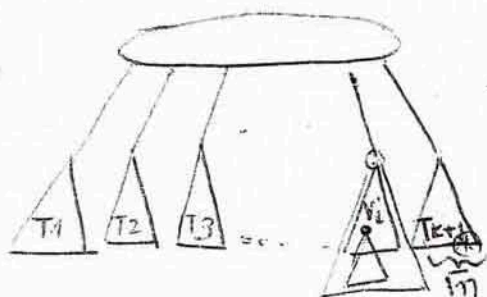


Αν έχουμε  $\overline{m}$  κόμβους, τότε το REP θα έχει  $\overline{m}$  τεύχος και να  
 να μπορεί να μας καθορίσει ποια από τα κομμάτια  
 του μας στις ρίζες να αναμένεται.

Εάν μπορούμε να έχουμε ούτως  $m = \lceil m^a \rceil$  για  $\frac{1}{2} < a \leq 1$ , τα  $S_j$   
 έχουν όλα όμοιοι  $\overline{m}$  τεύχος και τα  $T_1, T_2, \dots, T_{k+1}$  είναι  
 ideal.

Για να έχουμε  $O(n)$  χώρο, αρέσει  $\frac{1}{2} \leq a < 1$

### Εισαγωγές - Διαγραφές



Οι εισαγωγές μπορούν να  
 μετασχηματιστούν υπερβολικά ένα δέντρο  
 Δέντρο έχουμε πρώτους εισαγωγής  
 όπως στο red-black tree.

η Έστω  $w$  την αρχή και  $w + w/4$  την. Εφαρμόζω  
 partial rebuilding. Παίρνει το δέντρο που έχει μετα-  
 σχηματιστεί υπερβολικά και φτιάχνει ένα ideal IST δέντρο  
 με  $w/4 + w$  φύλλα.

(vi) Όταν θέλουμε να διαγράψουμε ένα στοιχείο, επηρεάζουμε  
 size(vi) το φύλλο που την περιέχει δηλαδή αυξάνουμε  
 δέντρο διαγράφουμε το φύλλο, αλλιώς το αγνοούμε.  
 Μας αφήνουν χώρο χωρίς να τους χρειάζαμε.

2η χειρότερη περίπτωση, δέντρο  $O(\log n)$

$H(n) \leq 1 + H(n/2) \rightarrow$  όσο μας προς το κάτω, μειώνεται  
 (αναποδοτισμός) το τεύχος.

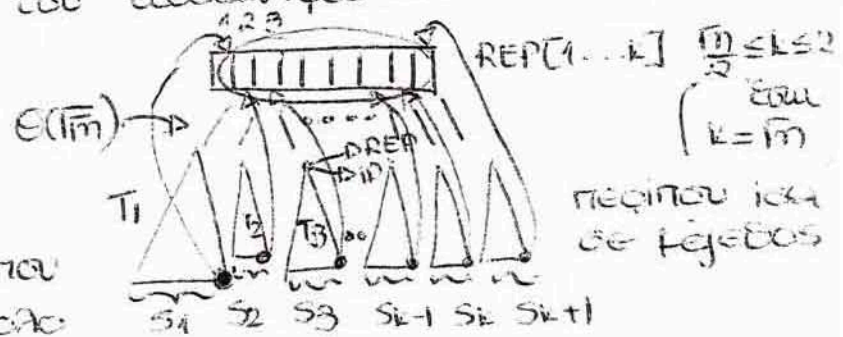
Ορίζουμε σε κάθε κόμβο το size(n) που είναι ο # των φύλλων  
 χωρίς σημασία αν αρέσει να ορίσουμε το isize(vi): isize(vi)  
 = size(vi) (initial size). Εξαναδομηση έτσι ώστε το δέντρο  
 στον κόμβο  $v_i$  να είναι γραμμικό. (Rebuild(vi))  $\rightarrow$  κατασκευά-  
 ζει ένα ideal IST για τα  $m$  επηρεαζόμενα φύλλα  $T_{v_i}$ , ②  
 Έστω  $C(w) = 0$ ,  $\forall w \in T_{v_i}$  και σωστό isize(w).



3. Εισαγωγές/Διαγραφές  
(insert) amortized time  $\log(n)$  (amortized)  
just amortized time  $\log(n)$
4. Αναζήτηση  $O((\log n)^2)$  στη χειρότερη περίπτωση  
έχει βέλτιστα σε  $O(\log n)$  σε μερικούς αλγόριθμους
5. Διαπεράσιμη είναι και διακρίνει διαδοχικά γρήγορα σε  $O(n)$   
χρόνο, επίσης Predecessor, Successor και Min σε  $O(1)$   
χρόνο.

Στα IST δέντρα, ο βαθμός κάθε κόμβου εξαρτάται από  
τον αριθμό των φύλλων του υψόμεντου του.

- Βαθμός κόμβου είναι  $\Theta(\bar{m})$   
 $\frac{\bar{m}}{2} \leq \text{βαθμός} \leq 2\bar{m}$



Κάθε ένα από τα υψόμεντα που  
δημιουργούνται κόμβων το είναι  
τιμής που έχουμε σε κόμβους.

- Ideal IST: κάθε γιος αποθηκεύει  
 $\bar{m}$  φύλλα και έχει βαθμό  $\Theta(\bar{m})$   
(Μας δίνει τρόπο να υλοποιήσουμε το  
υψος του δέντρου)  
 $H(n) = 1 + H(\bar{m})$

στην ρίζα αποθηκεύετε ένα array που το αποθηκεύετε  
REP.  
Το REP το χρησιμοποιείτε για να αποθηκεύετε την αναζήτηση  
για το ελάχιστο υψόμεντο. Ο REP αποθηκεύει τιμές από το φύλλο  
του δέντρου ή αλλιώς από το εύρος  $S: x_1 < x_2 < \dots < x_n \leq$   
 $[a, b]$ . REP είναι  $x_{i1}, x_{i2}, \dots, x_{ik}$  και  $REP[j] = x_{ij}$   
Την τιμή που βρίσκεται στο δεδομένο φύλλο του  $i$  υψόμεντου  
την αποθηκεύετε στο πίνακα REP. Τα  $S_1, S_2, \dots, S_{k+1}$  είναι Ideal  
αποθηκεύονται σε IST δέντρα  $T_1, T_2, \dots, T_{k+1}$ . Η δομή  
αυτή ορίζεται αναδρομικά. Ειδικότερα, έχουμε  $ID[1..m]$   
 $ID = \text{Inverse Distribution}$ . Θέλουμε να υλοποιήσουμε την  
αναδρομική κατανομή των τιμών του εύρους έτσι ώστε να  
εξοικονομούμε την κατανομή.  $ID[j] \Leftrightarrow REP[j] \leq a + (b-a) \cdot \frac{j-1}{m} = REP[j]$   
(ID περιλαμβάνει στην ουσία)

Τα μεγέθη του REP ταξινομούνται με τους όρους με  
χρησιμοποιώντας ένα word interpolation search.



αυξανόν, σε αριθμό στοιχείων

Το array ID τα συντάσσεται με βάση τον αριθμό όρων που γράφονται.  
Συγκεκριμένα,

Το ID έχει ένα word εγχείριση  
από το REP

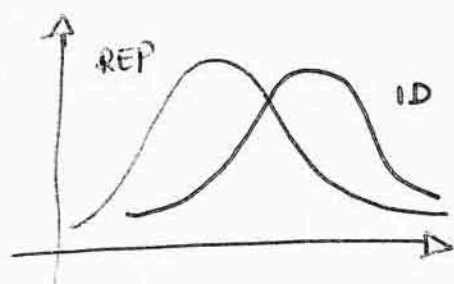
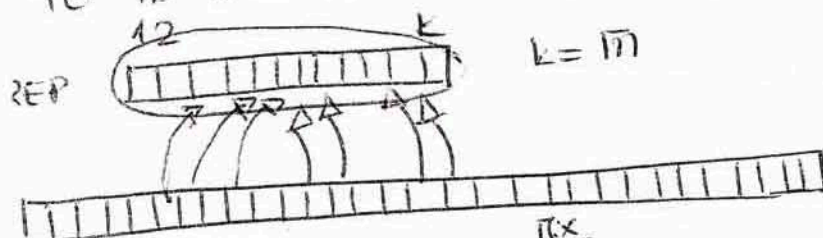
$$1) i = \left\lceil \frac{y-a}{b-a} m \right\rceil$$

$$2) j = ID[i]$$

3) while  $(y \geq REP[j])$  AND  $(j < k)$  do  
 $j = j + 1$

4) Υαζε στο j-οστό μεσοσπασμό.

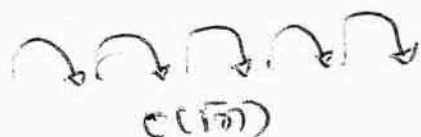
Το ID έχει δείκτη από συγκεκριμένα της του REP



REP	12	15	20	22	40	71	202	400	700
ID	12	15	20	22	40	71	202	400	700

για  
 $n=18$

Εκτελείται από την αρχή της ID  
δείχνουν τους μεσούς της του REP  
ένα συνδυασμό νοσήν της word  
δείχνουν σε μεσούς της του REP.



MMT  
 $O(\log \log n)$

εφαρμογή interpolation  
search στο REP

Με επεξεργασία-αυτο γράφο

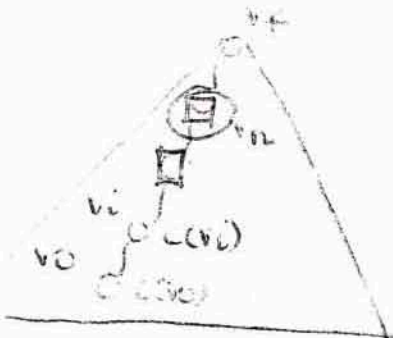
$REP[j], REP[j+2^0], REP[j+2^1], \dots, REP[j+2^{l-1}], REP[j+2^l]$

τ.ω  $REP[j+2^{l-1}] \leq y \leq REP[j+2^l]$

Αυτο γράφο από  $REP[j+2^{l-1}+1], REP[j+2^{l-1}+2], \dots, REP[j+2^l]$   
σε + κοτπο  $O(\log n)$



## Search(x)



Εάν οι κόμβοι αντιστοιχίζονται με αριθμούς από 0, τότε το εύρος  $m$  είναι  $x$ .

1.  $search(x, T) \rightarrow$  2.  $c(v) = c$  και  $size(v)$
3. for  $i=1$  to  $k$
- $c(v_i) = c(v) + 1$

κάθε κόμβος έχει 4. Πιο γρήγορα από υπολογίζει εύρος  $v_h$  είναι τελειώνει 5. Αν  $\exists v_h \text{ Rebuild}(v_h)$

$c(v) \geq size(v)/4 \rightarrow$  έχει τις περισσότερες αλλαγές το υποδέντρο του (υπολογισμός). Στην συνέχεια, διαγράψτε το τελειώνει (πιο γρήγορα) από αυτός.

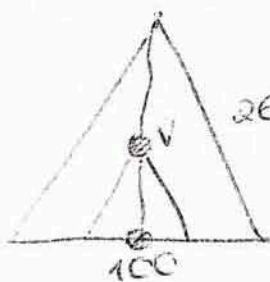
Επειδή είναι το βήμα 5, διαγράφουμε το υποδέντρο  $v_h$  και είναι υποδέντρο του  $v_h$ .

## Διαγραφή(x)

1. —
2. Διαγράψτε φύλλο που περιέχει  $x$ .
3. —
4. —
5. —

Τα υποδέντρα βήματα είναι ίδια με την διαγραφή(x)

Αν μας αναφέρεται κάποιος να έχουμε ανάμεσα άλλα φέει και τις σχέσεις



$$26 \geq 100/4$$

$$26 \geq 12$$

$$26 \geq 14$$

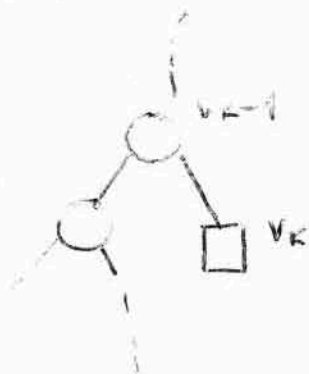
26 φορές  $\rightarrow$  έχει καταρτίσει αυτό το υποδέντρο βήματα rebuild.

## Εύρεση(y)



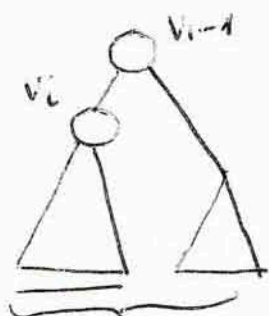
Το REP μας βοηθάει να γράψουμε

$$w_i = \# \text{ γυμνών } T_{v_i}, \quad 0 \leq i \leq k$$



Το παιδί του  $v_{k-1}$  είναι εκπαιδευμένο γυμνό.

$w_{k-1} \geq 2$ .  $w_{k-1} = 2$  όταν το αριστερό του  $v_{k-1}$  είναι φύλλο.



$$\frac{w_i}{w_{i-1}} \leq (1-\alpha) \Rightarrow \boxed{w_i \leq (1-\alpha) w_{i-1}}$$

↓  
από αριστερά  
BBTA]

γυμνοποίηση  
του  $v_{i-1}$

$$2 \leq w_{k-1} \leq (1-\alpha) w_{k-2} \leq (1-\alpha)^2 w_{k-3} \leq \dots \leq (1-\alpha)^{k-1} w_0$$

$$w_0 = \# \text{ γυμνών του δέντρου} = n+1$$

$$\text{Άρα } 2 \leq (1-\alpha)^{k-1} (n+1)$$

Θέλουμε να βρούμε άνω όριο για το  $k$  και γ'αυτό λογαριθμίζουμε:

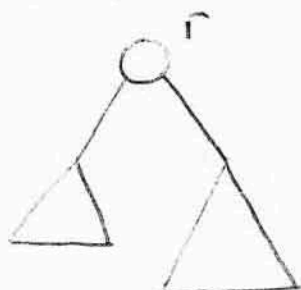
$$1 \leq (k-1) \log(1-\alpha) + \log(n+1) \Rightarrow -(k-1) \log(1-\alpha) \leq \log(n+1) - 1$$

$$\Rightarrow (k-1) \log(1-\alpha)^{-1} \leq \log(n+1) - 1 \Rightarrow k \log(1-\alpha)^{-1} \leq \log(n+1) - 1$$

$$+ \log(1-\alpha)^{-1} \Rightarrow k \leq \frac{\log(n+1) - 1}{\log(1-\alpha)^{-1}} + 1 \Rightarrow k = O(\log n)$$

Έχουμε φέρει το βάθος ενός δέντρου με τον λογαριθμισμό του αριστερά των γυμνών που αυξάνεται.

Κάνοντας εισαγωγές/διαγραφές, χαλάει η γυμνοποίηση με αποτέλεσμα να εωηρεάζονται τα όρια με τα οποία φράσσεται την γυμνοποίηση.



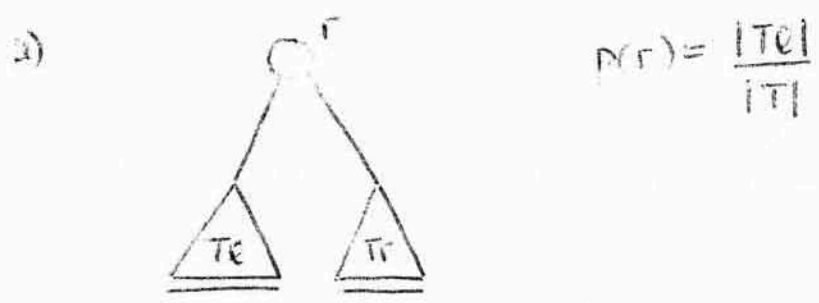
$$1-\alpha \geq p(r) \geq \alpha$$

Χαλάει αν διαγραφούμε 1 γυμνό από το αριστερό υποδέντρο ή αν εισαγάγουμε 1 γυμνό στο δεξί υποδέντρο.



$\epsilon(\log n)$  bytes  
 $\hookrightarrow \epsilon(\log^2 n)$

BB[α] δέντρο

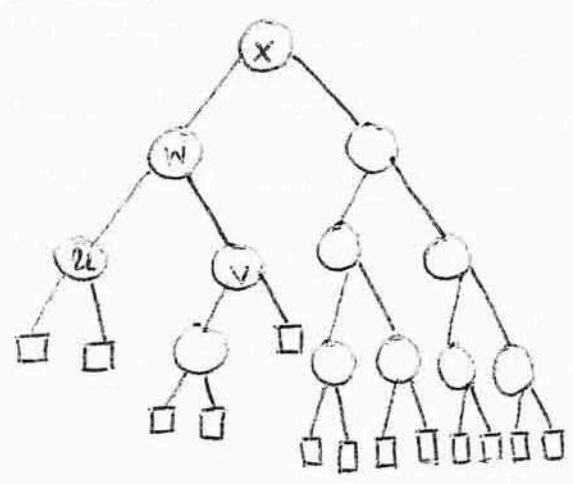


$|T| = 100$  φύλλα  $p(r) = 0.42$   
 $|T_L| = 42 = 11 \cdot$

β) Το T είναι περιορισμένος συγκερασμός αν  $\alpha \leq p(r) \leq 1-\alpha$   
 (r ρίζα του T) και ισχύει  $\forall$  υποδέντρο του T  
 Έστω  $\alpha = 0.3$   $0.3 \leq p(r) \leq 0.7$   $\alpha \in [1/4, 1-1/2]$

γ) Το σύνολο όλων των T περιορισμένος συγκερασμός είναι BB[α] δέντρο.

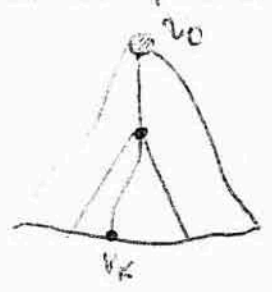
$\alpha = \frac{1}{3}$



$p(x) = 5/14$   
 $p(w) = 2/5$   
 $p(u) = 1/2$   
 $p(v) = 2/3$

$\frac{1}{3} \leq p(v) \leq \frac{2}{3} \quad \forall v \in T$   
 $\downarrow$   
 συγκερασμός

δ) Το T είναι BB[α] με η εσωτ. εκφραση  
 'Απόδειξη με επαγωγή διασυνδέει όλες (n+1 φύλλα)  
 Έστω μονοπάτι  $v_0, v_1, \dots, v_k$

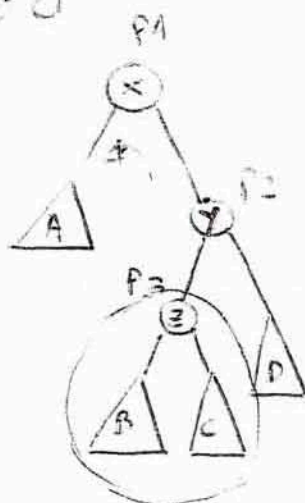


Το k είναι το βάθος του  $v_k$  ( $k = \text{βάθος}(v_k)$ ).  
 Δεν είναι κατ'ανάγκη το τελευταίο φύλλο  
 ώστε να θεωρούμε να ωριμάει να είναι το  
 υψος του T.

$$a \leq j_1, j_2 \leq 1-a$$

Μπορεί να ισχύει να αντιστοιχίζω το κάθε περιεστρώς

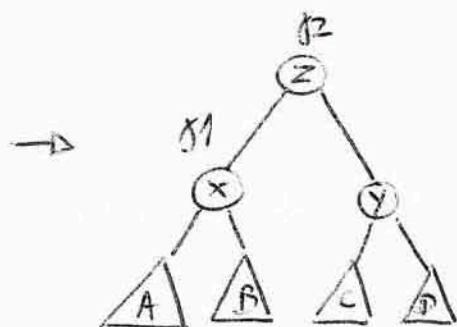
2)  $p_2 \geq 1$



Εέλουμε κατά κάποιο τρόπο να περιορίσουμε τον αριθμό των φύλλων στα όρια του  $x$  έτσι ώστε να χωρίσουμε σταθερικά τα παιδιά του κόμβου  $x$ . Για το λόγο

$p_1 \leq a$  αυτό εφαρμόζουμε πλήρη περιστροφή προς τα αριστερά.

Πλήρη περιστροφή  $\rightarrow 2$  αλληλ. περιστροφές.



(Το βίβλιο το έχει ακόμα)

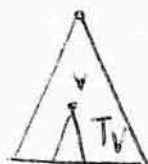
Αυτή την διαδικασία την εφαρμόζουμε στο πρώτο εσωτερικό κόμβο που υψώθηκε προερχόμενα ενομοιογενή.

Εφαρμόζουμε (εχει ολόκληρη) περιστροφές.

505

### ΙΔΙΟΤΗΤΑ ΒΑΡΟΥΣ

Ένας εσωτερικός κόμβος  $v$  θα ενομοιοποιηθεί μετά από  $O(|T_v|)$  πράξεις στο  $T_v$ .

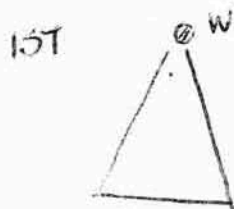


Μετά από 1 περιστροφή, ο  $v$  θα έχει  $|T_v|$  γράμματα.

$O(k)$

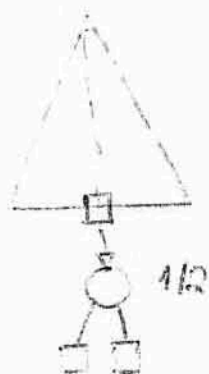
π.χ. έχει  $k$  γράμματα. Θα ενομοιοποιηθεί μετά από  $k/2$  διαγραφές ή εισαγωγές τουλάχιστον που συμβαίνουν εφόσον το ζεύγος.

Στο 1στ, ισχύει επίσης η ιδιότητα βαρους.



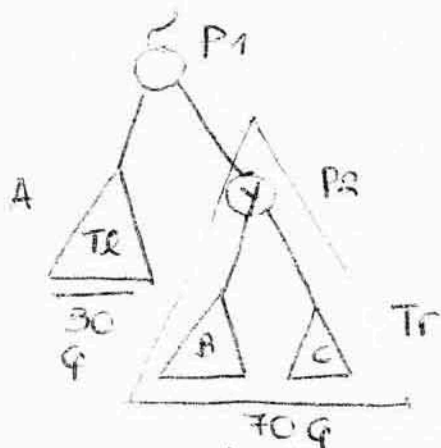
$$|size(w)| = |T_w|$$

θα ενομοιοποιηθεί μετά από  $|T_w|/4$



Αντιστοιχία με το κλάδο 1 = είναι κόμβος με 2 παιδιά

$$2^h = n \Rightarrow h = \log_2 n$$



Εστω ότι βρίσκουμε σε έναν εσωτερικό κόμβο

$$\rightarrow p_1 \geq \alpha \text{ ΠΡΙΝ}$$

$$\rightarrow p_1 < \alpha \text{ ΜΕΤΑ}$$

• ΠΡΙΝ  
(εισαγωγή  
σε εσωτερικό  
κόμβο)

$$\frac{|T_L| + 1}{|T| + 1} \geq \alpha$$

το 1 δείχνει το φύλλο που έχω διαγράψει

• ΠΡΙΝ  
(εισαγωγή  
σε φύλλο  
κόμβο)

$$\frac{|T_L|}{|T| - 1} \geq \alpha$$

$\frac{30}{100}$  OK (στην αρχή, πριν την εφαρμογή αναδιάρθρωσε κόμβου)

$\frac{30}{101} < \alpha$  αλλά  $\frac{30}{101-1} \geq \alpha = \text{OK}$  (εφαρμόζουμε εισαγωγή φύλλου)

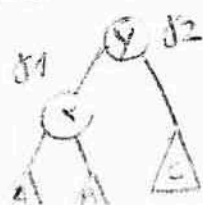
Την χρονική στιγμή 1, είναι  $\frac{|T_L|}{|T|} \geq \alpha$ .

Την -1- -1- -1+1, είναι  $\frac{|T_L|}{|T|+1} < \alpha$

$d > 0$

1)  $p_2 \leq d$  στην περίπτωση

$d \rightarrow$  τοχαίο σταθερά που επιλέγουμε έτσι με  $d \in [\alpha, 1-\alpha]$

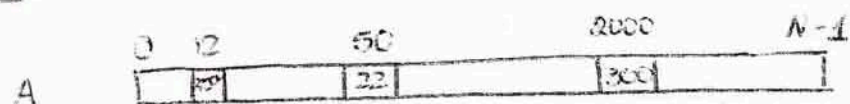


$$p_1 = p_1 + (1-p_1)p_2$$

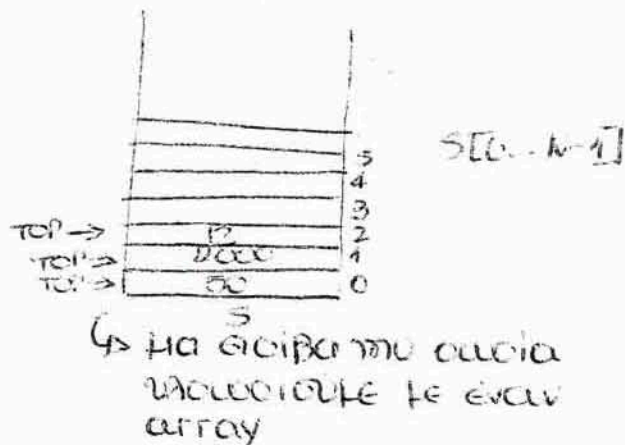
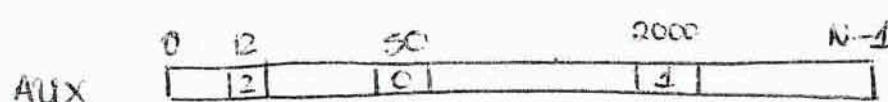
$$p_2 = p_1 / [p_1 + (1-p_1)p_2]$$

Η ιδέα είναι ότι έχουμε ένα array με στοιχεία  
 που θέλουμε να τα ελέγξουμε

505 Array Initialization σε O(1).



Χρησιμοποιούμε 1 extra array με το AUX



Αρχικά, TOP = -1

Βάζω στην αρχή του array 50 to 22.

→ την αλλαγή

Εισαγωγή (v, i) → δέχομαι την αλλαγή

TOP = TOP + 1

AUX[i] = TOP

S[TOP] = i

A[i] = v;

}

Βάζω στην αρχή του array 2000 ενώ αργότερα (έτσι 300)

Η εικόνα που δείχνει την αλλαγή με την αλλαγή είναι η εικόνα που  
 έχει ο AUX την δέχομαι στην εικόνα η αλλαγή ενοποιηθεί  
 έχει γίνει εικόνα

Επειδή δεν ξέρω αν έχω εικόνα για την ή όχι "αλλαγή",  
 ελέγχω τον AUX ως εξής:

Read(i) {

if (S[AUX[i]] = i) AND (0 ≤ AUX[i] ≤ TOP)

OK

else

fail

}