

Κινητός και Διάχυτος Υπολογισμός (Mobile & Pervasive Computing)

Δημήτριος Κατσαρός, Ph.D.

Χειμώνας 2005

Διάλεξη 4η

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

Ιστοσελίδα του μαθήματος

- http://skyblue.csd.auth.gr/~dimitris/courses/mpc_fall05.htm
- Θα τοποθετούνται οι διαφάνειες του επόμενου μαθήματος
- Σταδιακά θα τοποθετηθούν και τα research papers που αντιστοιχούν σε κάθε διάλεξη

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

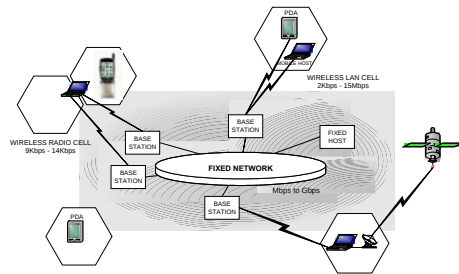
Περιεχόμενα

- **Αρχιτεκτονική κινητού δικτύου**
- Εκπομπή σε πολλαπλά κανάλια
- Caching
- Prefetching
- Ευρετήρια

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

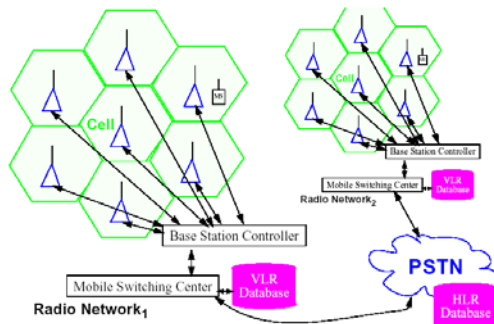
Αρχιτεκτονική κινητού δικτύου



20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

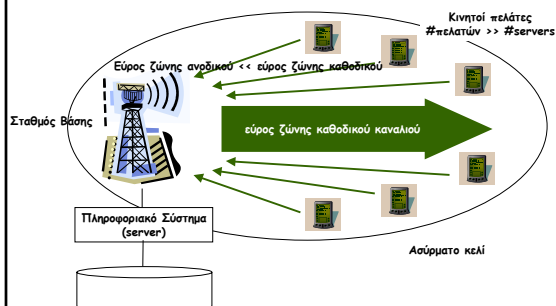
Αρχιτ. Personal Comm. Sys. (PCS)



20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

Γενικό μοντέλο εκπομπής



20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

Περιεχόμενα

- Αρχιτεκτονική κινητού δικτύου
- **Εκπομπή σε πολλαπλά κανάλια**
- Caching
- Prefetching
- Ευρετήρια

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

7

Πολλαπλά κανάλια εκπομπής

Για λόγους όπως:

- Application scalability
 - Μια εφαρμογή αποκτά επιπλέον κανάλια για να εξυπηρετήσει μεγαλύτερο πληθυσμό
- Fault tolerance
 - Τρεις servers εκπέμπουν σε μια γεωγραφική περιοχή σε μη συνεχόμενες συχνότητες, αλλά οι δυο παθαίνουν βλάβη και τα κανάλια τους ανατίθενται στον τρίτο
- Reconfiguration of adjoining cells
 - Γειτονικά κελιά εξυπηρετούνται από διαφορετικούς servers, αλλά τα κελιά συνενώνονται και τα κανάλια ανατίθενται στον έναν από τους δυο
- Heterogeneous clients
 - Πελάτες με ετερογενείς δυνατότητες

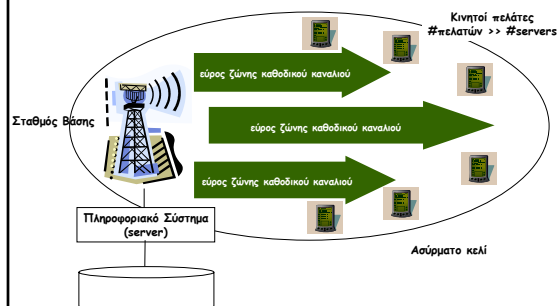
Είναι δυνατόν να υπάρχουν πολλαπλά κανάλια εκπομπής

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

8

Εκπομπή σε Πολλαπλά Κανάλια

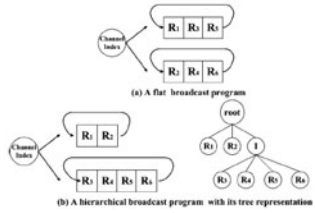


20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

9

Ιεραρχικά προγράμματα εκπομπής



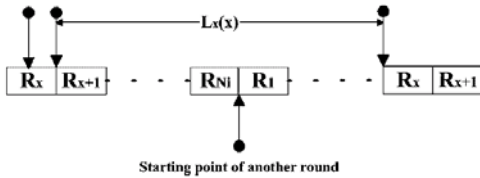
	Access frequency						Average expected delay	
	$P_r(R_1)$	$P_r(R_2)$	$P_r(R_3)$	$P_r(R_4)$	$P_r(R_5)$	$P_r(R_6)$	in figure 1(a)	in figure 1(b)
Case 1	0.167	0.167	0.167	0.167	0.167	0.167	1	1.16
Case 2	0.25	0.25	0.125	0.125	0.125	0.125	1	1
Case 3	0.3	0.3	0.1	0.1	0.1	0.1	1	0.9
Case 4	0.4	0.4	0.05	0.05	0.05	0.05	1	0.7

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

Μέση καθυστέρηση σε ένα κανάλι

Request for data item x



Μέση καθυστέρηση για κάθε αντικείμενο στο κανάλι i είναι:

$$\sum_{x=1}^{N_i} (N_i - x) / N_i$$

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

Δημιουργία ιεραρχικών προγραμμ.

Property 2. Let $t_i = \sum_{h=1}^i N_h$ and $t_0 = 0$. Then, N_i data items, denoted by R_j , $t_{i-1} + 1 \leq j \leq t_i$, are allocated to broadcast disk i , and $d_i = d_{R_j}$ for $j \in [t_{i-1} + 1, t_i]$.

Πρόβλημα
Δημιουργίας
Ιεραρχικού
Προγράμματος
Εκπομπής

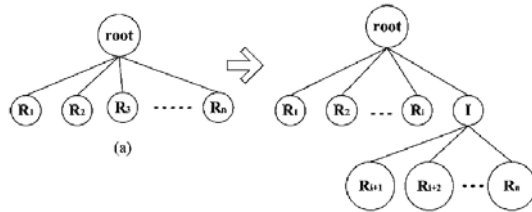
$$\begin{aligned} & \sum_{j=1}^n d_{R_j} \cdot P_r(R_j) \\ &= \sum_{i=1}^K d_i \sum_{j=t_{i-1}+1}^{t_i} P_r(R_j) \\ &= \sum_{i=1}^K \left(\sum_{q=1}^{N_i} \frac{N_i - q}{N_i} \right) \sum_{j=t_{i-1}+1}^{t_i} P_r(R_j), \end{aligned}$$

where $t_i = \sum_{h=1}^i N_h$ and $t_0 = 0$.

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

Δενδρική αναπαράσταση



20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

13

Κόστος επιπέδου του δένδρου

Definition 1. Suppose that level v in the allocation tree has $j - i + 1$ data nodes, $R_i, R_{i+1}, \dots, R_j$. The cost of level v is defined as

$$C_{i,j} = \sum_{k=1}^{j-i+1} \frac{(j-i+1)-k}{j-i+1} \sum_{q=i}^j P_r(R_q),$$

which is equal to $w_v \cdot P_{L_v}$, where w_v and P_{L_v} are, respectively, the weight of leaf nodes and the aggregate access frequency for the leaf nodes in level v of the allocation tree.

Definition 2. Suppose that node R has $j - i + 1$ child data nodes, $R_i, R_{i+1}, \dots, R_j$, which are sorted according to the descending order of $P_r(R_q)$, $i \leq q \leq j$, i.e., $P_r(R_q) \geq P_r(R_y)$ iff $q \leq y$. The reduction gain achieved by grouping nodes $R_{p+1}, R_{p+2}, \dots, R_j$ and attaching them under a new child node, denoted by $\delta(p)$, can be formulated as $\delta(p) = C_{i,j} - (C_{i,p} + C_{p+1,j})$.

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

14

Αλγόριθμος VF^K

Input. Assume that $R_1, \dots, R_n$ have been sorted according to the descending order of $P_r(R_j)$, $1 \leq j \leq n$, i.e., $P_r(R_q) \geq P_r(R_y)$ iff $q \leq y$. K is the number of broadcast disks in a broadcast disk array.

Output. The resulting allocation tree.

begin

1. Create table AT with K rows;
2. $AT(1).B = 1$; /* $AT(1).B$ records the beginning of level 1 */
3. $AT(1).E = n$; /* $AT(1).E$ records the end of level 1 */
4. $AT(1).LC = C_{1,n}$; /* $AT(1).LC$ records the cost of level 1 */
5. **for** each row i in table AT and $i \geq 2$
6. **begin**
7. $AT(i).B = 0$; /* $AT(i).B$ records the beginning of level i */
8. $AT(i).E = 0$; /* $AT(i).E$ records the end of level i */
9. $AT(i).LC = 0$; /* $AT(i).LC$ records the cost of level i */
10. **end**
11. $pivot = 1$;

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

15

Αλγόριθμος VF^K

```

12. repeat
13.   begin
14.     Choose row  $i$  from table  $AT$  such that  $AT(i).LC$  is maximal
        among all unmarked rows;
15.     if  $(i == 1 \text{ or } i == pivot) \wedge$  Upward and downward partition  $\wedge$ 
16.       begin
17.          $j = \text{Partition}(R_{AT(i).B}, R_{AT(i).B+1}, \dots, R_{AT(i).E})$ ;
18.         {Update table  $AT$  accordingly and unmark all rows;
19.          $pivot++$ ;}
20.       end
21.     else  $\wedge$  Middle partition  $\wedge$ 
22.       begin
23.          $j = \text{Partition}(R_{AT(i).B}, R_{AT(i).B+1}, \dots, R_{AT(i).E})$ ;
24.         if  $(AT(i-1).E - AT(i-1).B) < (j - AT(i).B)$ 
25.           {Update table  $AT$  accordingly and unmark all rows;
26.            $pivot++$ ;}
27.         else {
28.           Mark row  $i$ ;
29.           Merge  $(R_{AT(i).B}, R_{AT(i).B+1}, \dots, R_j)$  and
               $(R_{j+1}, R_{j+2}, \dots, R_{AT(i).E})$  together;
30.         }
31.       end
32.   until  $(pivot == K)$ 
end

```

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

Αλγόριθμος VF^K

Procedure Partition($R_i, R_{i+1}, \dots, R_j$).

1. Determine p^* such that

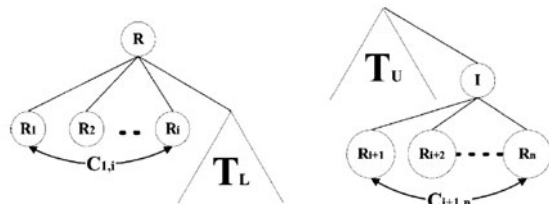
$$\delta(p^*) = \max_{p \in \{i, i+(j-i+1)/2-1\}} \{\delta(p)\}.$$

2. Attach nodes $R_{p^*+1}, R_{p^*+2}, \dots, R_j$ under a new node I in the tree.
3. Return p^* .

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

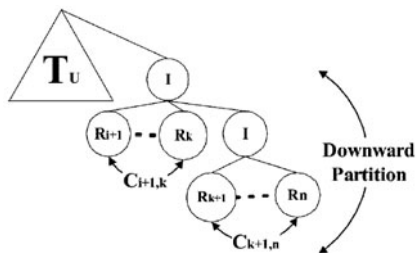
Αλγόριθμος VF^K



20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

Αλγόριθμος VF^K

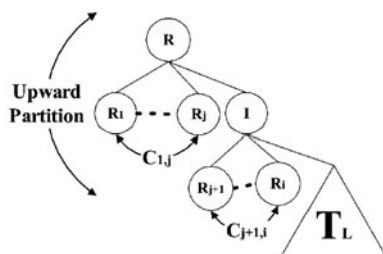


20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

19

Αλγόριθμος VF^K



20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

20

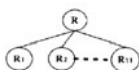
Παράδειγμα του VF^K

Data record	R_1	R_2	R_3	R_4	R_5	R_6	R_7	R_8	R_9	R_{10}	R_{11}
$P_T(R_i)$	0.237	0.211	0.132	0.132	0.08	0.05	0.05	0.027	0.027	0.027	0.027

(a) Partition $(R_1, R_2, \dots, R_{11})$ is selected and decomposed to $(R_1, \dots, R_4)$ and $(R_5, \dots, R_{11})$.

Level i	$AT(i).B$	$AT(i).E$	$AT(i).LC$
1	1	11	5*
2	0	0	0
3	0	0	0
4	0	0	0

p	1	2	3	4	5
$C_{1,11}$	5	5	5	5	5
$C_{1,p} + C_{p+1,11}$	3,4335	2,484	2,05	1,932	2,104
$\delta(p)$	1,5665	2,516	2,95	3,068*	2,896



20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

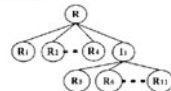
21

Παράδειγμα του VF^K

(b) Partition $(R_1, R_2, \dots, R_4)$ is selected and decomposed to (R_1, R_2) and (R_3, R_4) .

Level i	$AT(i).B$	$AT(i).E$	$AT(i).LC$
1	1	4	1.068*
2	5	11	0.864
3	0	0	0
4	0	0	0

p	1	2
$C_{1,4}$	1.068	1.068
$C_{1,p} + C_{p+1,4}$	0.475	0.356
$\delta(p)$	0.593	0.712*



20/10/2005

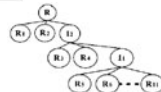
Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

Παράδειγμα του VF^K

(c) Partition $(R_5, R_6, \dots, R_{11})$ is selected and decomposed to $(R_5, \dots, R_7)$ and $(R_8, \dots, R_{11})$.

Level	$AT(i).B$	$AT(i).E$	$AT(i).LC$
1	1	2	0.224
2	3	4	0.132
3	5	11	0.864*
4	0	0	0

p	5	6	7
$C_{5,11}$	0.864	0.864	0.864
$C_{5,p} + C_{p+1,11}$	0.52	0.381	0.342
$\delta(p)$	0.344	0.483	0.522*



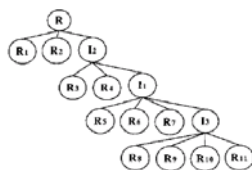
20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

Παράδειγμα του VF^K

(d) The final result of table AT.

Level i	$AT(i).B$	$AT(i).E$	$AT(i).LC$
1	1	2	0.224
2	3	4	0.132
3	5	7	0.18
4	8	11	0.162



20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

Το πρόβλημα Windows Scheduling

- Έστωσαν **h slotted κανάλια** και **n αντικείμενα (σελίδες)**, που στην κάθε μια αντιστοιχεί **ένα παράθυρο (window) $w_i \geq 1$** , όπου w_i είναι ακέραιος.
- Είναι δυνατό να εκπέμψουμε τις n σελίδες στα h κανάλια, μια σελίδα σε κάθε κανάλι σε κάθε slot (χρονική στιγμή), ώστε *το κενό μεταξύ δυο συνεχόμενων εμφανίσεων της σελίδας i να είναι όχι μεγαλύτερο από w_i* ?
- Εάν για δεδομένο h και $W(w_1, w_2, \dots, w_n)$, η απάντηση στο παραπάνω ερώτημα είναι καταφατική, το πρόγραμμα εκπομπής που λύνει το πρόβλημα αποκαλείται **$\langle h, W \rangle$ -πρόγραμμα**
- Το πρόβλημα βελτιστοποίησης είναι να βρούμε το ελάχιστο h που θα το σημειώνουμε ως $H(W)$, ώστε να υπάρχει το **$\langle h, W \rangle$ -πρόγραμμα**
- Το πρόβλημα αυτό αποκαλείται **optimal windows scheduling πρόβλημα**

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

25

Το πρόβλημα Windows Scheduling

- Το πρόβλημα αυτό ανήκει στην κλάση των NP-Hard ακόμα και στην περίπτωση που $h=1$.
- Παράδειγμα:
 - $W = \langle 2, 4, 5 \rangle$
 - $\langle 1, W \rangle = [2 \ 4 \ 2 \ 5 \ 2 \ 4 \ 2 \ 5]$
 - Σημειώστε ότι η πρώτη σελίδα εμφανίζεται κάθε 2 slots, η δεύτερη κάθε 4 slots και η Τρίτη κάθε 4(<5) slots

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

26

Περιεχόμενα

- Αρχιτεκτονική κινητού δικτύου
- Εκπομπή σε πολλαπλά κανάλια
- **Caching**
- Prefetching
- Ευρετήρια

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

27

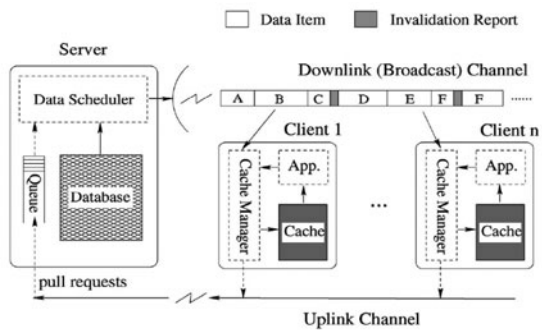
Caching στους κινητούς πελάτες

- Προγράμματα εκπομπής
 - Βασίζονται στη μέση πιθανότητα προσέλασης: μέσος όρος πάνω σε ΟΛΟΥΣ τους πελάτες
 - Όχι αναγκαστικά βέλπστη για κάθε έναν πελάτη
- Πώς μπορεί κάθε πελάτης να υποβοηθήσει τον εαυτό του?
 - **Caching**: προσωρινή αποθήκευση των δεδομένων που λαμβάνει
 - Πολιτική caching: όταν εξαναγκάζεται να αντικαταστήσει κάποιο (επειδή η cache είναι πλήρης), αντικαθιστά εκείνα που είναι λιγότερο πιθανό να χρησιμεύσουν στο μέλλον

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

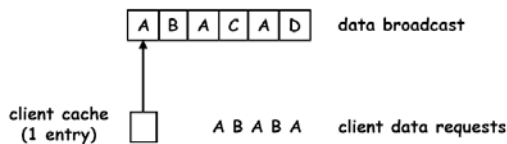
Το γενικό μοντέλο caching



20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

Caching στους κινητούς πελάτες



- Ποια **πολιτική αντικατάστασης** (replacement strategy ή policy ή algorithm) θα πρέπει να χρησιμοποιούν οι πελάτες;

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

31

Least Recently Used (LRU)

9 Steps

LRU: Αντικατάσταση του αντικειμένου που έχει χρησιμοποιηθεί παλιότερα στο παρελθόν

20/10/2005 Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

32

Most Probable Accessed (MPA)

8 Steps

MPA: Αντικατάσταση του αντικειμένου που θα χρησιμοποιηθεί λιγότερο συχνά στο μέλλον

20/10/2005 Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

33

Probability Inv. Broadc. Frequency

5 Steps !

PIX: Αντικατάσταση του αντικειμένου με το μικρότερο P/X , όπου

- **P:** είναι η πιθανότητα προσπέλασης του αντικειμένου
- **X:** είναι η συχνότητα εκπομπής του αντικειμένου

$PIX(A) = \frac{1}{2} / \frac{1}{2} = 1, PIX(B) = \frac{1}{3} / \frac{1}{6} = 2, PIX(C) = \frac{1}{6} / \frac{1}{6} = 1$

20/10/2005 Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

Η πολιτική LIX (1/3)

- Η πολιτική PIX δεν είναι πρακτική
 - Απαιτεί τέλεια γνώση των πιθανοτήτων προσπέλασης
 - Απαιτεί σύγκριση των τιμών PIX όλων των αντικειμένων στην cache (σειριακή σάρωση των αντικειμένων)
- Ιδέα: προσέγγιση της πολιτικής PIX με έναν αλγόριθμο του τύπου LRU (δηλ., τον LIX) που λαμβάνει υπόψη του πιθανότητες προσπέλασης
- LRU (Least Recently Updated)
 - Τα Cached δεδομένα διατηρούνται σε μια λίστα
 - Εάν ένα δεδομένο προσπελαστεί μετακινείται στην κορυφή/αρχή της λίστας
 - Όταν συμβεί cache miss (απαιτείται να “κατεβεί” από το κανάλι και να μπει στην cache), το δεδομένο στο τέλος της λίστας εκδιώκεται από τη λίστα

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

34

Η πολιτική LIX (2/3)

- Χρήση τροποποιημένου LRU για κάθε δίσκο εκπομπής χωριστά
 - Ο LIX διατηρεί μια λίστα με αντικείμενα για κάθε δίσκο εκπομπής με συχνότητα f_j
 - Τα αντικείμενα μπαίνουν στη λίστα του αντίστοιχου δίσκου εκπομπής όπου ανήκουν
 - Όταν ένα αντικείμενο προσπελάζεται, τοποθετείται στην κορυφή/αρχή της λίστας
 - Μια εκτίμηση της πιθανότητας προσπέλασης p_i αναπροσαρμόζεται οποτεδήποτε το αντικείμενο d_i προσπελάζεται
 - Όταν ένα αντικείμενο d_i πρόκειται να εκδιωχτεί, υπολογίζεται μια LIX τιμή lix_i (που προσεγγίζει την PIX τιμή) για κάθε αντικείμενο στο τέλος κάθε λίστας
 - Το αντικείμενο με τη χαμηλότερη τιμή, τελικά εκδιώκεταιThe data item with lowest value lix_i is evicted

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

35

Η πολιτική LIX (3/3)

Computing the access probability estimate p_i for d_i :

Initial value $p_i = 0$

If most recent access to d_i was at time t_i (initially $t_i = 0$) and t is the time of the current access then

$$p_i := \frac{c}{t - t_i} + (1 - c)p_i$$

c : constant ($0 < c < 1$)

Approximation of PIX value
(d_i belongs to broadcast disk with frequency f_j)

$$lix_i = p_i / f_j$$

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

36

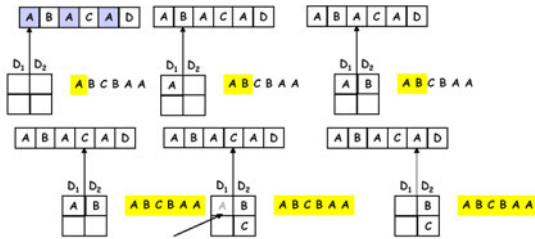
Παράδειγμα LIX

Δύο δίσκοι εκπομπής

$D_1 = \{A\}$, $D_2 = \{B, C, D\}$, συχνότητες $f_1 = 3$, $f_2 = 1$, $c = 1/2$

Η cache μπορεί να αποθηκεύσει 2 αντικείμενα.

p	1	2	3	4	5
A	1	1	1	1	1
B	0	1/4	1/4	1/4	1/4
C	0	0	0	1/8	1/8
D	0	0	0	0	0



evict: $lix_A = \frac{1}{3} = 1/6$, $lix_B = \frac{1}{2} = 1/4$

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

Περιεχόμενα

- Αρχιτεκτονική κινητού δικτύου
- Εκπομπή σε πολλαπλά κανάλια
- Caching
- **Prefetching**
- Ευρετήρια

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

Κίνητρο για prefetching

- Οι PIX/LIX αποθηκεύουν στην cache αντικείμενα, μόνο μετά αφού ζητηθούν
- Εναλλακτικά, ο κινητός πελάτης μπορεί να “κατεβάζει” αντικείμενα από το κανάλι, αφού ούτως ήγει άλλως το ακούει
- Ο στόχος είναι να ελαττώσει το χρόνο απόκρισης
- Μέθοδοι prefetching:
 - Tag Team Caching
 - Ευρεστικό Prefetching

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

Η έννοια του tag team caching

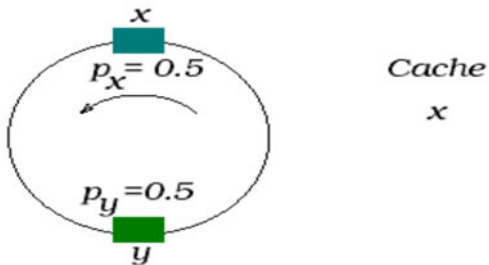
- Tag Team Caching – Τα αντικείμενα συνεχώς αντικαθιστούν το ένα το άλλο μέσα στην cache
- Για παράδειγμα,
 - έστωσαν δυο αντικείμενα x και y τα οποία εκπέμπονται από το κανάλι
 - Ο πελάτης caches το x όταν εκπέμπεται στο κανάλι
 - Εκδιώχνει το x και caches το y , όταν εκπέμπεται το y

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

40

Παράδειγμα tag team caching (1/2)



20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

41

Αναμεν. Καθυστέρη. “Demand Driven”

- Υποθέτουμε ότι ένας πελάτης ενδιαφέρεται να προσπελάσει το x και το y $p_x = p_y = 0.5$ και ότι έχει μια cache με μια μόνο θέση
- Στο μοντέλο demand driven, κάνει cache το x και εάν χρειάζεται το y , περιμένει για το y και αντικαθιστά το x στην cache με το y
- Η αναμενόμενη καθυστέρηση σε cache miss είναι $\frac{1}{2}$ της περιστροφής του δίσκου
- Η αναμενόμενη καθυστέρηση πάνω σε όλες τις προσπελάσεις είναι
- $C_i * M_i * D_i$, όπου C είναι η πιθανότητα προσπέλασης, M είναι η πιθανότητα ενός cache miss και D είναι η αναμενόμενη καθυστέρηση εκπομπής για το αντικείμενο i
- Για τα αντικείμενα x και y , είναι ίση με: $0.5 * 0.5 * 0.5 + 0.5 * 0.5 * 0.5 = 0.25$

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

42

Αναμεν. Καθυστέρ. “Tag team caching”

- $0.5 \cdot 0.5 \cdot 0.25 + 0.5 \cdot 0.5 \cdot 0.25 = 0.125$, δηλαδή, το μέσο κόστος είναι το $\frac{1}{2}$ του αντίστοιχου κόστους στο σχήμα demand driven
- Γιατί? Ένα miss μπορεί να συμβεί σε οποιαδήποτε στιγμή στο μοντέλο demand driven, ενώ τα misses συμβαίνουν μόνο κατά τη διάρκεια του μισού broadcast στο tag team caching

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

Ευρεστικό Prefetching

- Απλό Ευρεστικό Prefetching
- Εκτελεί έναν υπολογισμό για κάθε αντικείμενο που εκπέμπεται στο κανάλι με βάση την πιθανότητα προσπέλασης P για το αντικείμενο και το ποσό του χρόνου T που θα περάσει μέχρι να εμφανιστεί ξανά η σελίδα στο κανάλι εκπομπής
- Εάν η $P \cdot T$ τιμή των σελίδων που εκπέμπονται είναι υψηλότερες από αυτές των σελίδων στην cache, τότε εκκείνες με τη χαμηλότερη τιμή $P \cdot T$, εκδιώκεται από την cache

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

Ευρεστικό Prefetching

Page	Access Probability	Broadcast Frequency (per Period)	PTN Value
A	P	2	$P/2$
B	$P/2$	2	$P/4$
C	$P/2$	1	$P/2$

Table 2: Access and Frequency Values for the pt Example

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

Ευρεστικό Prefetching

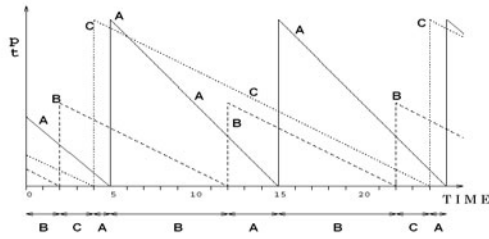


Figure 7: pt Value vs. Time

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

Περιεχόμενα

- Αρχιτεκτονική κινητού δικτύου
- Εκπομπή σε πολλαπλά κανάλια
- Caching
- Prefetching
- **Ευρετήρια**

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

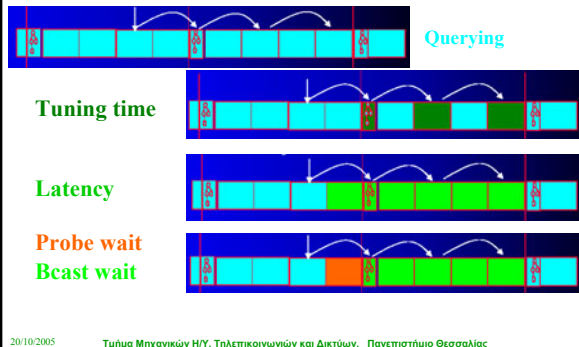
Παράμετροι ενδιαφέροντος (1/2)

- **Tuning time:** Ο χρόνος που ο κινητός πελάτης δαπανά “ακούγοντας” το κανάλι. Προσδιορίζει την κατανάλωση ενέργειας για την απόκτηση των δεδομένων
- **Latency (Access time):** Ο χρόνος που περνάει (κατά μέσο όρο) από τη στιγμή που ο κινητός πελάτης “κάνει αίτηση” για κάποια δεδομένα μέχρι τη στιγμή που τα δεδομένα αυτά έρχονται στην κατοχή του πελάτη
 - **Probe wait:** Ο μέσος χρόνος από τη στιγμή συντονισμού στο κανάλι μέχρι να βρει τον δείκτη (pointer) για τον επόμενο index. Είναι ίσος με το μισό της απόστασης μεταξύ δυο τμημάτων index.
 - **Bcast wait:** Ο μέσος χρόνος από τη στιγμή που βρίσκεται ο πρώτος index μέχρι να “κατεβούν” όλα τα δεδομένα

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

Παράμετροι ενδιαφέροντος (2/2)



Οργάνωση του καναλιού εκπομπής

- **Packet:** η βασική (μικρότερη) μονάδα μεταφοράς μηνυμάτων στα δίκτυα
- **Bucket:** η μικρότερη λογική μονάδα εκπομπής. Αποτελείται από σταθερό αριθμό packets. Όλα τα buckets έχουν το ίδιο μέγεθος
 - Index buckets
 - Data buckets
- **Index Segment:** σύνολο συνεχόμενων index buckets
- **Data Segment:** σύνολο συνεχόμενων data buckets

20/10/2005 Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

Οργάνωση του καναλιού εκπομπής

- Περιεχόμενα bucket
 - **Bucket_id:** το offset του bucket από την αρχή του κύκλου εκπομπής
 - **Bcast_pointer:** το offset μέχρι την αρχή του επόμενου κύκλου εκπομπής
 - **Index_pointer:** το offset μέχρι την αρχή του επόμενου index segment
 - **Bucket_type:** data bucket ή index bucket
- Index bucket: είναι μια ακολουθία της μορφής:
 - (**attribute_value, offset**): offset είναι ένας δείκτης στο bucket που περιέχει εγγραφή που προσδιορίζεται από την attribute_value

20/10/2005 Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

Clustering index

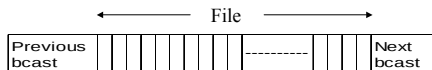
- **Clustering index**: ένα ευρετήριο (index) είναι clustered πάνω σε ένα attribute, εάν όλες οι εγγραφές με την ίδια τιμή για το attribute αυτό, εμφανίζονται συνεχόμενες σε ένα “αρχείο”
- Τα δυο άκρα στην βελτιστοποίηση Tuning time και Access time
 - Latency_opt
 - Tune_opt

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

52

Latency OPT



Latency είναι η βέλτιστη: Δεν υπάρχει επιβάρυνση για το index

$$\text{Latency} = \text{Data}/2 + C$$

$$\text{Tuning time} = \text{Data}/2 + C$$

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

53

Tuning OPT



Tuning time είναι ο βέλτιστος:

$$\begin{aligned} \text{Latency} &= (\text{Data} + \text{Index}) / 2 + (\text{Data} + \text{Index}) / 2 + C \\ &= \text{Data} + \text{Index} + C \end{aligned}$$

$$\text{Tuning time} = k + C$$

k: number of levels in the index tree

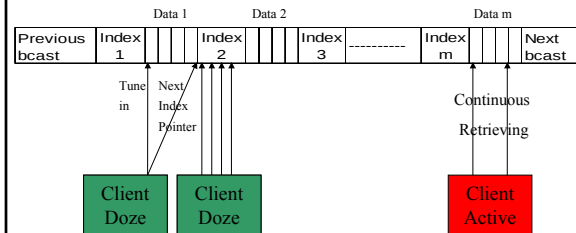
20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

54

(1, m) Indexing

Το ευρετήριο (index) εκπεμπεται m φορές κατά τη διάρκεια μιας εκπομπής του αρχείου.
Πολυεπίπεδο ευρετήριο (index), δηλαδή δένδρο
Ολόκληρος ο index εκπέμπεται πριν από το 1/m κομμάτι του αρχείου.



20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

Ανάλυση (1, m) Indexing

- Η κατανομή πιθανότητας του initial probe για τους πελάτες είναι ομοιόμορφη
- Data: το μέσο μέγεθος του αρχείου
- C: η coarseness του index attribute
- Ο index έχει δείκτες μόνο στην πρώτη εμφάνιση εγγραφής με συγκεκριμένη τιμή για το attribute
- Άρα, κατασκευάζουμε ευρετήριο μόνο για (Data/C) data buckets
- n, η χωρητικότητα ενός bucket, δηλ., ο αριθμός των ζευγών (attribute_value, offset) που μπορεί να στεγάσει
- k: ο αριθμός των επιπέδων του ευρετηρίου
- Index: ο αριθμός των buckets του

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

Ανάλυση (1, m) Indexing

Όταν το δένδρο είναι πλήρως ισοζυγισμένο $k = \left\lceil \log_n \left(\frac{Data}{C} \right) \right\rceil$ $Index = \sum_{i=0}^{k-1} n^i$

Latency: The probe wait is $\frac{1}{2} * (Index + \frac{Data}{m})$ and the bcast wait is $\frac{1}{2} * ((m * Index) + Data) + C$. Hence, the latency is

$$\frac{1}{2} * \left(Index + \frac{Data}{m} \right) + \frac{1}{2} * ((m * Index) + Data) + C,$$

i.e.,

$$\frac{1}{2} * \left((m + 1) * Index + \left(\frac{1}{m} + 1 \right) * Data \right) + C.$$

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

Ανάλυση (1, m) Indexing

Tuning time: $1 + k + C$

•Βέλτιστο m για ελαχιστοποίηση Latency:

- Παραγωγή την εξίσωση της Latency ως προς m
- Εξίσωση με 0
- Επίλυση ως προς m

$$m^* = \sqrt{\frac{\text{Data}}{\text{Index}}}$$

20/10/2005

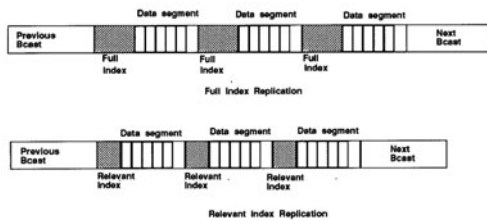
Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

58

Distributed Indexing

- Βελτίωση του (1,m) index ελαττώνοντας τη replication του index

- Δεν χρειάζεται να εκπέμπεις ολόκληρο τον index μεταξύ διαδοχικών data segments, αλλά μόνο το κομμάτι του που δεικτοδοτεί τα data που έπονται.

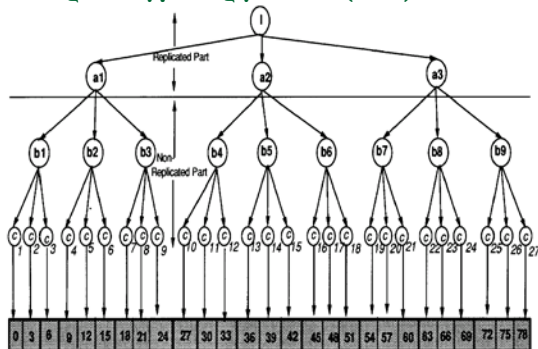


20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

59

Παράδειγμα εργασίας (1/2)



20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

60

Παράδειγμα εργασίας (2/2)

$NRR = \{b1, b2, b3, b4, b5, b6, b7, b8, b9\}$
 $Rep(b1) = \{1, a1\}$ $Ind(b1) = \{b1, c1, c2, c3\}$ $Data(b1) = \{0, \dots, 8\}$
 $Rep(b2) = \{a1\}$ $Ind(b2) = \{b2, c4, c5, c6\}$ $Data(b2) = \{9, \dots, 17\}$
 $Rep(b3) = \{a1\}$ $Ind(b3) = \{b3, c7, c8, c9\}$ $Data(b3) = \{18, \dots, 26\}$
 $Rep(b4) = \{1, a2\}$ $Ind(b4) = \{b4, c10, c11, c12\}$ $Data(b4) = \{27, \dots, 35\}$
 $Rep(b5) = \{a2\}$ $Ind(b5) = \{b5, c13, c14, c15\}$ $Data(b5) = \{36, \dots, 44\}$
 $Rep(b6) = \{a2\}$ $Ind(b6) = \{b6, c16, c17, c18\}$ $Data(b6) = \{45, \dots, 53\}$
 $Rep(b7) = \{1, a3\}$ $Ind(b7) = \{b7, c19, c20, c21\}$ $Data(b7) = \{54, \dots, 62\}$
 $Rep(b8) = \{a3\}$ $Ind(b8) = \{b8, c22, c23, c24\}$ $Data(b8) = \{63, \dots, 71\}$
 $Rep(b9) = \{a3\}$ $Ind(b9) = \{b9, c25, c26, c27\}$ $Data(b9) = \{72, \dots, 80\}$

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

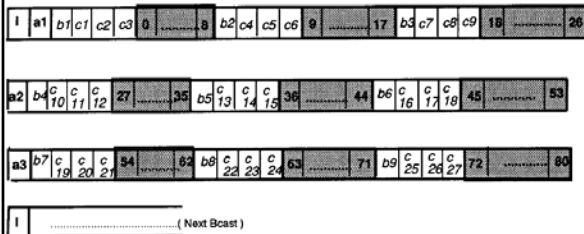
Εναλλ. μέθοδοι index distribution

- **Nonreplicated Distribution:** διαφορετικά κομμάτια index segments είναι διακριτά, δηλ., δεν υπάρχει replication.
- **Entire Path Replication:** Το μονοπάτι από τη ρίζα μέχρι ένα index bucket B γίνεται replication ακριβώς πριν την εμφάνιση του B.
- **Partial Path Replication:** Έστωσαν δυο index buckets B και B'. Είναι αρκετό να κάνουμε replication το μονοπάτι από τον ελάχιστο κοινό πρόγονο (Least Common Ancestor) των B και B', πριν την εμφάνιση του B', δεδομένου ότι προσθέτουμε κάποια επιπρόσθετη indexing πληροφορία για "προσανατολισμό".

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

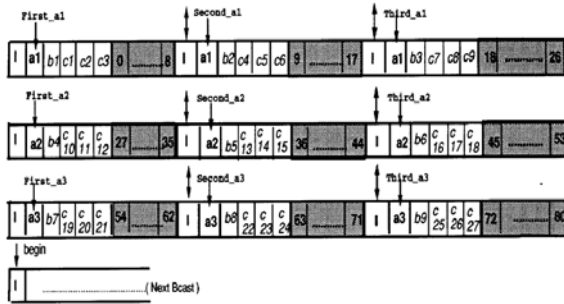
Nonreplicated Distribution



20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

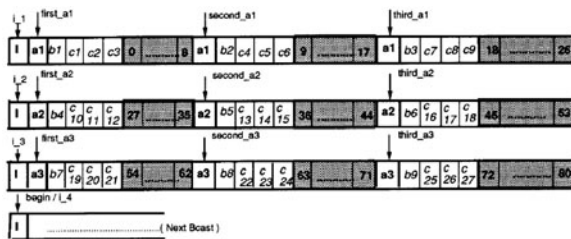
Entire Path Replication



20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

Distributed Indexing



20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

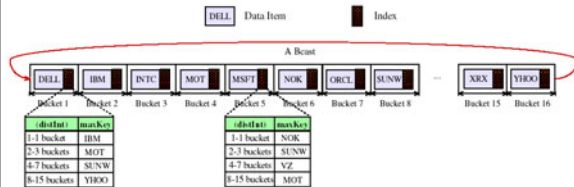
Exponential Index (1/2)

- Data buckets
 - Data part
 - Index table
 - Index entries:
 - κάθε entry indexes ένα segment από buckets και έχει τη μορφή $\{distInt, maxKey\}$
 - » $distInt$: καθορίζει την απόσταση των buckets από το τρέχον bucket (μετρημένο σε αριθμό buckets)
 - » $maxKey$: είναι η τιμή του μέγιστου κλειδιού αυτών των buckets
 - Τα μεγέθη των segments αυξάνουν εκθετικά (δυνάμεις του 2). Η i-οστή entry περιγράφει το segment των buckets τα οποία είναι σε απόσταση 2^{i-1} έως $2^i - 1$
 - Προφανώς οι τιμές $distInt$ δεν χρειάζονται αφού μπορούν εύκολα να συναχθούν

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

Exponential Index (2/2)



20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

Κυρτά πρότυπα προσπέλασης

- Τι γίνεται όταν το πρότυπο προσπέλασης στις εγγραφές δεν είναι ομοιόμορφο?
- Τα ευρετήρια που παρουσιάσαμε είναι κατάλληλα για ομοιόμορφα πρότυπα προσπέλασης
- n : αριθμός εγγραφών
- R_i : i -οστή εγγραφή
- $Pr(R_i)$: πιθανότητα προσπέλασης εγγραφής R_i
- $I_{pi}(R_i)$: αριθμός index nodes μέχρι να φτάσουμε στην εγγραφή ή index node R_i
- a_i : αναπαριστά έναν index node
- $d(a_i)$: fanout του κόμβου a_i
- $Path(R_i)$: σύνολο των index nodes από τη ρίζα μέχρι την εγγραφή R_i
- $f(a_i)$: κόστος "ενεργοποίησης" index node a_i . Συνήθως $f(a_i)=d(a_i)$.

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

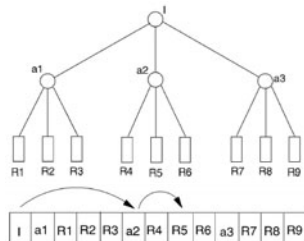
Ισοζυγισμένα δεινδρικά ευρετήρια

Έστω ότι το μέσο κόστος εντοπισμού μιας εγγραφής εκφράζεται ως:

$$\sum_{1 \leq i \leq n} Pr(R_i) \sum_{a_j \in Path(R_i)} f(a_j)$$

Το διπλανό ευρετήριο είναι βέλτιστο για ομοιόμορφη πιθανότητα προσπέλασης.

Μέσο κόστος: $C(T_{d=3}^B) = 6$



20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

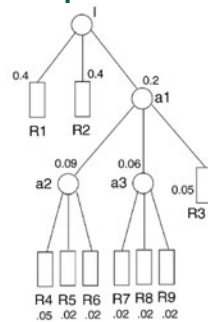
Μη ισοζυγισμένα δενδρικά ευρετήρια σταθερού fanout [π.χ., d=3]

Data record	R1	R2	R3	R4	R5	R6	R7	R8	R9
$Pv(R_i)$	0.4	0.4	0.05	0.05	0.02	0.02	0.02	0.02	0.02
$I_{a0}(R_i)$ in $T_{d=3}^I$	2	2	2	2	2	2	2	2	2
$I_{a0}(R_i)$ in $T_{d=3}^H$	1	1	2	3	3	3	3	3	3

$$\begin{aligned}
 C(T_{d=3}^I) &= 0.4*3 + 0.4*3 + \quad [\text{για } R_1 \text{ και } R_2] \\
 &\quad 0.05*(3+3) + \quad [\text{για } R_3] \\
 &\quad 0.05*(3+3+3) + \quad [\text{για } R_4] \\
 &\quad 5*0.02*(3+3+3) \quad [\text{για υπόλοιπες}] \\
 &= 0.4*6 + 0.05*15 + 45*0.02 = \mathbf{4.28}
 \end{aligned}$$

Σειρά εκπομπής στο ασύρματο κανάλι

I	R1	R2	a1	R3	a2	R4	R5	R6	a3	R7	R8	R9
---	----	----	----	----	----	----	----	----	----	----	----	----



20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

Αλγόριθμος κατασκευής $T_{d=j}^I$

Algorithm CF: Use access frequencies to build an index tree with a fixed fanout d .

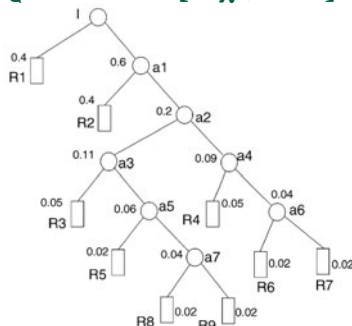
- Step 1:** Initially, we have a forest of n subtrees, each of which is a single node labeled with the corresponding access frequency.
- Step 2:** Attach the d subtrees with the smallest labels to a new node. Label the resulting subtree with the sum of all labels from its d child subtrees.
- Step 3:** $n = n - d + 1$. (Then, n is the number of remaining subtrees.) If $n = 1$ stop else goto Step 2.

Κατασκευάζει ένα Huffman δένδρο με fanout = d .
Στο βασικό Huffman δένδρο, fanout = 2.

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

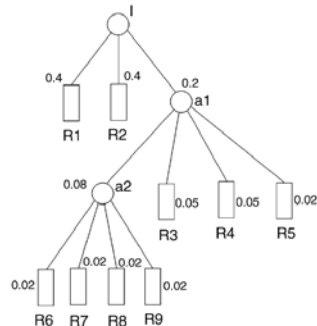
Μη ισοζυγισμένα δενδρικά ευρετήρια σταθερού fanout [π.χ., d=2]



20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

Μη ισοζυγισμένα δενδρικά ευρετήρια σταθερού fanout [π.χ., $d=4$]



20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

Μη ισοζυγισμένα δένδρα σταθερού fanout

- Δεν υπάρχει μονοτονική σχέση, ούτε αύξουσα ούτε φθίνουσα, για τις τιμές κόστους $C(T_{d=j}^I)$ σε σχέση με το επιτρεπτό fanout j .
- Αυτό το γεγονός, σε συνδυασμό με την ιεραρχική φύση της κατασκευής, υπονοεί ότι, επιτρέποντας μεταβλητά fanouts στους κόμβους του δένδρου, μπορούμε να ελαττώσουμε ακόμα περισσότερο το μέσο κόστος εντοπισμού των εγγράφων

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

Βασική ιδέα (1/2)

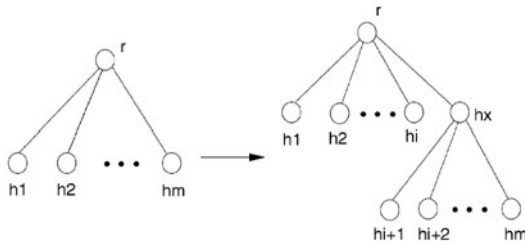
Lemma 1. Suppose that node r has m child nodes, $h_1, h_2, \dots$, and h_m , which are sorted according to descending order of $Pr(h_j)$, $1 \leq j \leq m$, i.e., $Pr(h_j) \geq Pr(h_k)$ if and only if $j \leq k$. Then, the average cost of index probes can be reduced by grouping nodes $h_{i+1}, h_{i+2}, \dots$, and h_m and attaching them under a new child node if and only if

$$(m - i - 1) \sum_{1 \leq j \leq i} Pr(h_j) > \sum_{i+1 \leq j \leq m} Pr(h_j).$$

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

Βασική ιδέα (2/2)



20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

76

Μη ισοζυγισμένα δένδρα μεταβλητού fanout T_V^I : Αλγόριθμος κατασκευής

Algorithm VF:

- Step 1:** Assume that $R_1, R_2, \dots$, and R_m have been sorted according to descending order of $Pr(R_j)$, $1 \leq j \leq m$, i.e., $Pr(R_j) \geq Pr(R_k)$ iff. $j \leq k$.
- Step 2:** Partition $(R_1, R_2, \dots, R_m)$.
- Step 3:** Report the resulting index tree.

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

77

Μη ισοζυγισμένα δένδρα μεταβλητού fanout T_V^I : Αλγόριθμος κατασκευής

Procedure *Partition* $(h_1, h_2, \dots, h_m)$:

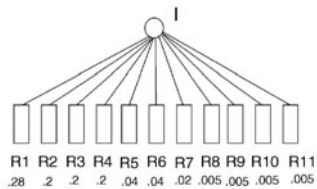
- Let $y(i) = (m - i - 1) \sum_{1 \leq j \leq i} Pr(h_j) - \sum_{i+1 \leq j \leq m} Pr(h_j)$. Determine i^* such that $y(i^*) = \max_{i \in \{1, m-2\}} \{y(i)\}$.
- If $y(i^*) \leq 0$, then return.
- Attach nodes $h_{i^*+1}, h_{i^*+2}, \dots, h_m$ under a new index node hx in the index tree.
- Partition $(h_1, h_2, \dots, h_{i^*})$.
- Insert hx into the ordered list $(h_1, h_2, \dots, h_{i^*})$ and relabel them as $(h_1, h_2, \dots, h_{i^*+1})$ according to descending order of $Pr(h_j)$, $1 \leq j \leq i^* + 1$.
- Partition $(h_1, h_2, \dots, h_{i^*+1})$.
- Return.

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

78

Παράδειγμα κατασκευής T_V^I (1/6)

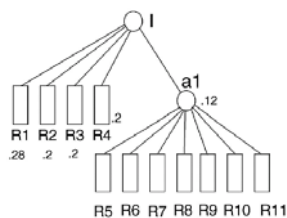


i	1	2	3	4	5	6	7	8	9
$(10-i) \sum_{1 \leq j \leq i} Pr(h_j)$	9*.208	8*.048	7*.068	6*.088	5*.092	4*.096	3*.098	2*.0985	0.99
$\sum_{i+1 \leq j \leq 11} Pr(h_j)$	0.72	0.52	0.32	0.12	0.08	0.04	0.02	0.015	0.01
$y(i)$	1.8	3.32	4.44	5.16	4.32	3.8	2.92	1.955	0.98

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

Παράδειγμα κατασκευής T_V^I (2/6)

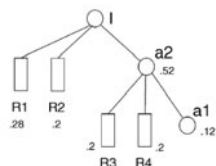


20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

Παράδειγμα κατασκευής T_V^I (3/6)

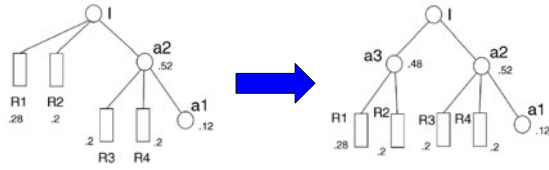
i	1	2	3
$(4-i) \sum_{1 \leq j \leq i} Pr(h_j)$	3*.28	2*.048	0.68
$\sum_{i+1 \leq j \leq 5} Pr(h_j)$	0.72	0.52	0.32
$y(i)$	0.12	0.44	0.32



20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

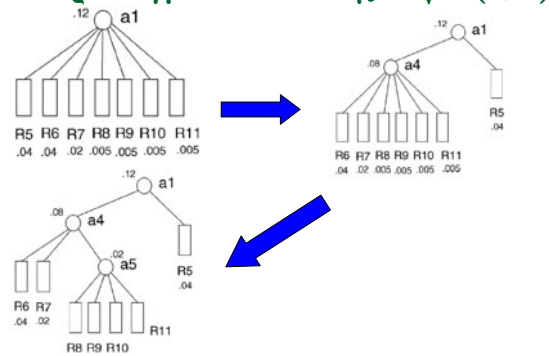
Παράδειγμα κατασκευής T^I_V (4/6)



20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

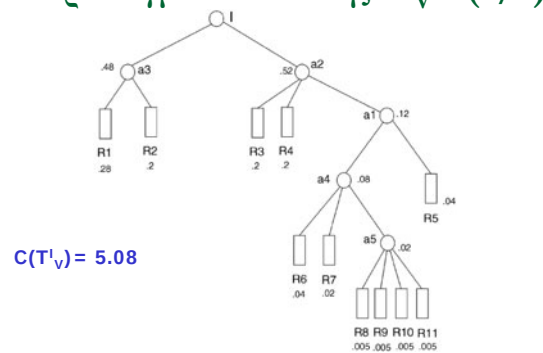
Παράδειγμα κατασκευής T^I_V (5/6)



20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

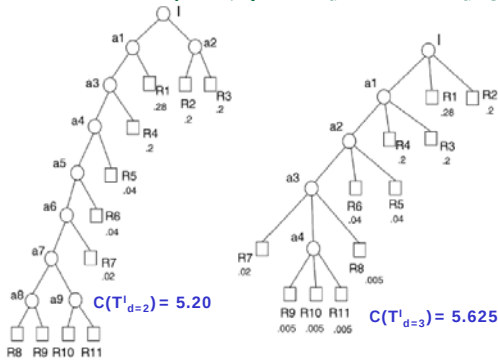
Παράδειγμα κατασκευής T^I_V (6/6)



20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

Αντιδιαστολή T^I_V με $T^I_{d=2}$ και $T^I_{d=3}$



20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

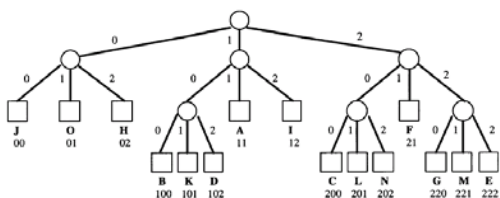
“Αλφαβητικά” δενδρικά ευρετήρια

- Ομοιόμορφη προσπέλαση & ταξινομημένα κλειδιά
 - (1,m) indexing
 - Distributed indexing
 - Exponential indexing
- Μη ομοιόμορφη προσπέλαση & μη ταξινομημένα κλειδιά
 - $T^I_{d=j}$
 - T^I_V
- Μη ομοιόμορφη προσπέλαση & ταξινομημένα κλειδιά ????

20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

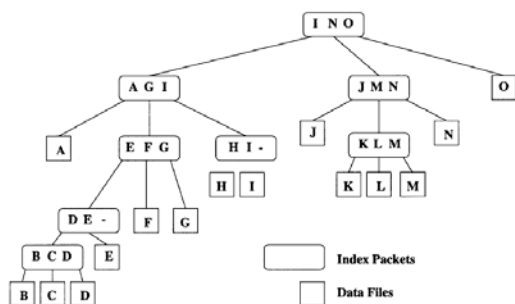
Το πρόβλημα των Huffman δένδρων



20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

Τα Alphabetic δένδρα



20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

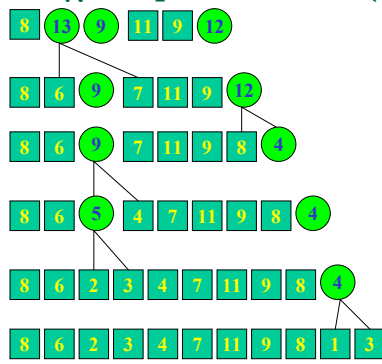
Παράδειγμα κατασκευής δυναμικού αλφαβητικού δένδρου



20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

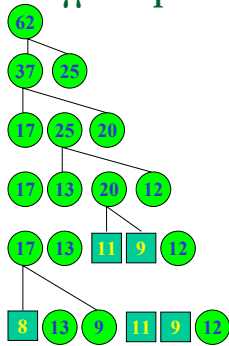
Παράδειγμα Alphabetic tree (1/4)



20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

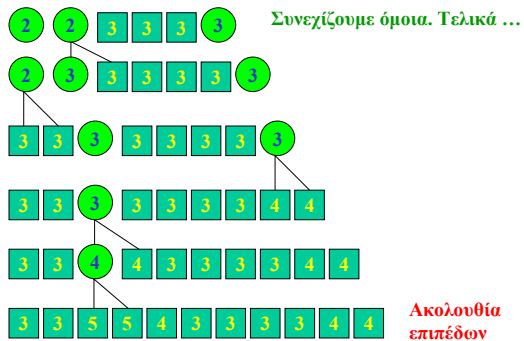
Παράδειγμα Alphabetic tree (2/4)



20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

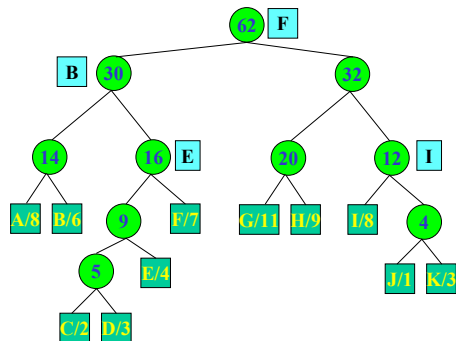
Παράδειγμα Alphabetic tree (3/4)



20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας

Alphabetic tree (4/4)



20/10/2005

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, Πανεπιστήμιο Θεσσαλίας
