Κινητός και Διάχυτος Υπολογισμός (Mobile & Pervasive Computing)

Δημήτριος Κατσαρός, Ph.D.

Χειμώνας 2005

Διάλεξη 3η

Ιστοσελίδα του μαθήματος

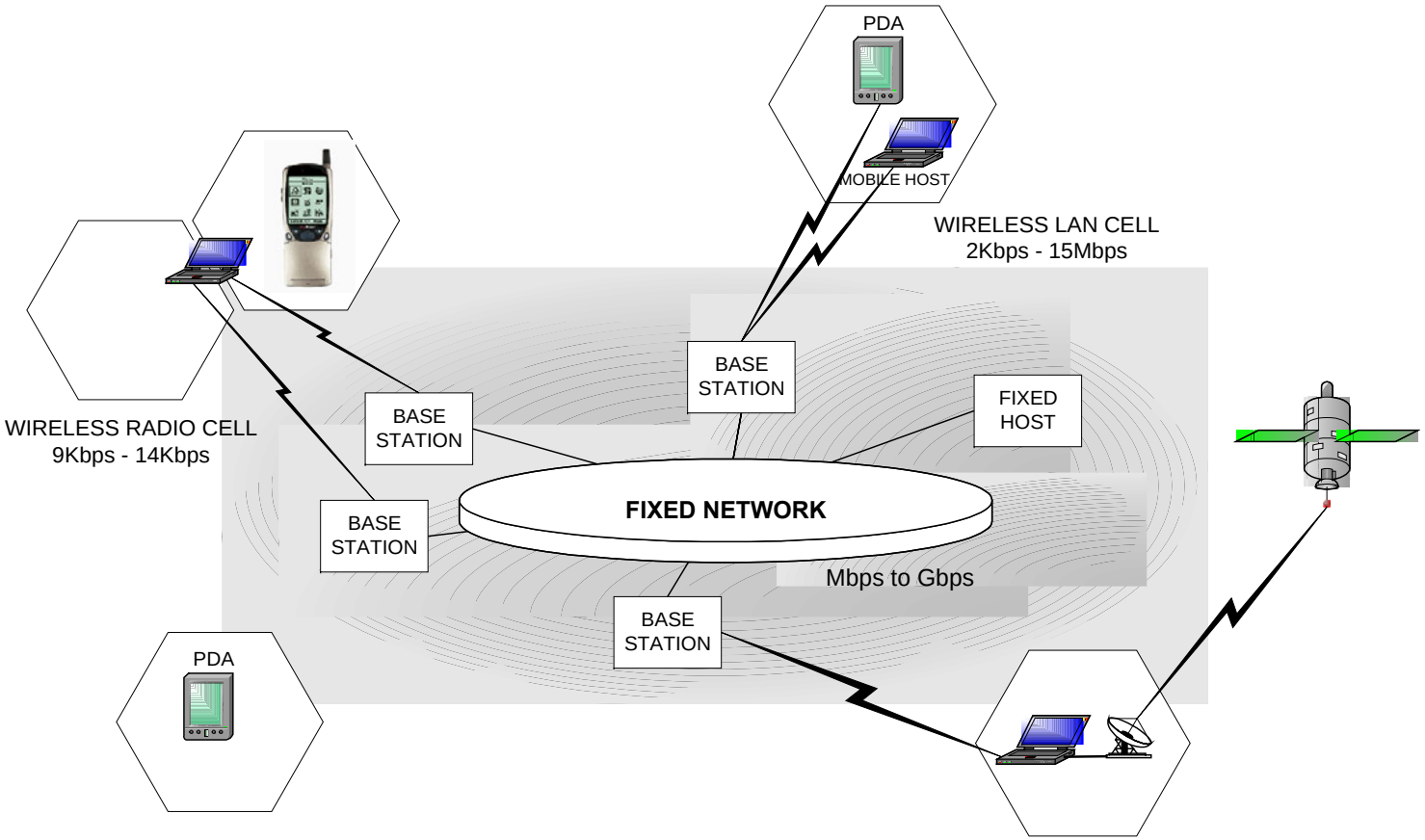
- **http://skyblue.csd.auth.gr/~dimitris/courses/mpc_fall05.htm**

- Θα τοποθετούνται οι διαφάνειες του επόμενου μαθήματος
- Σταδιακά θα τοποθετηθούν και τα research papers που αντιστοιχούν σε κάθε διάλεξη

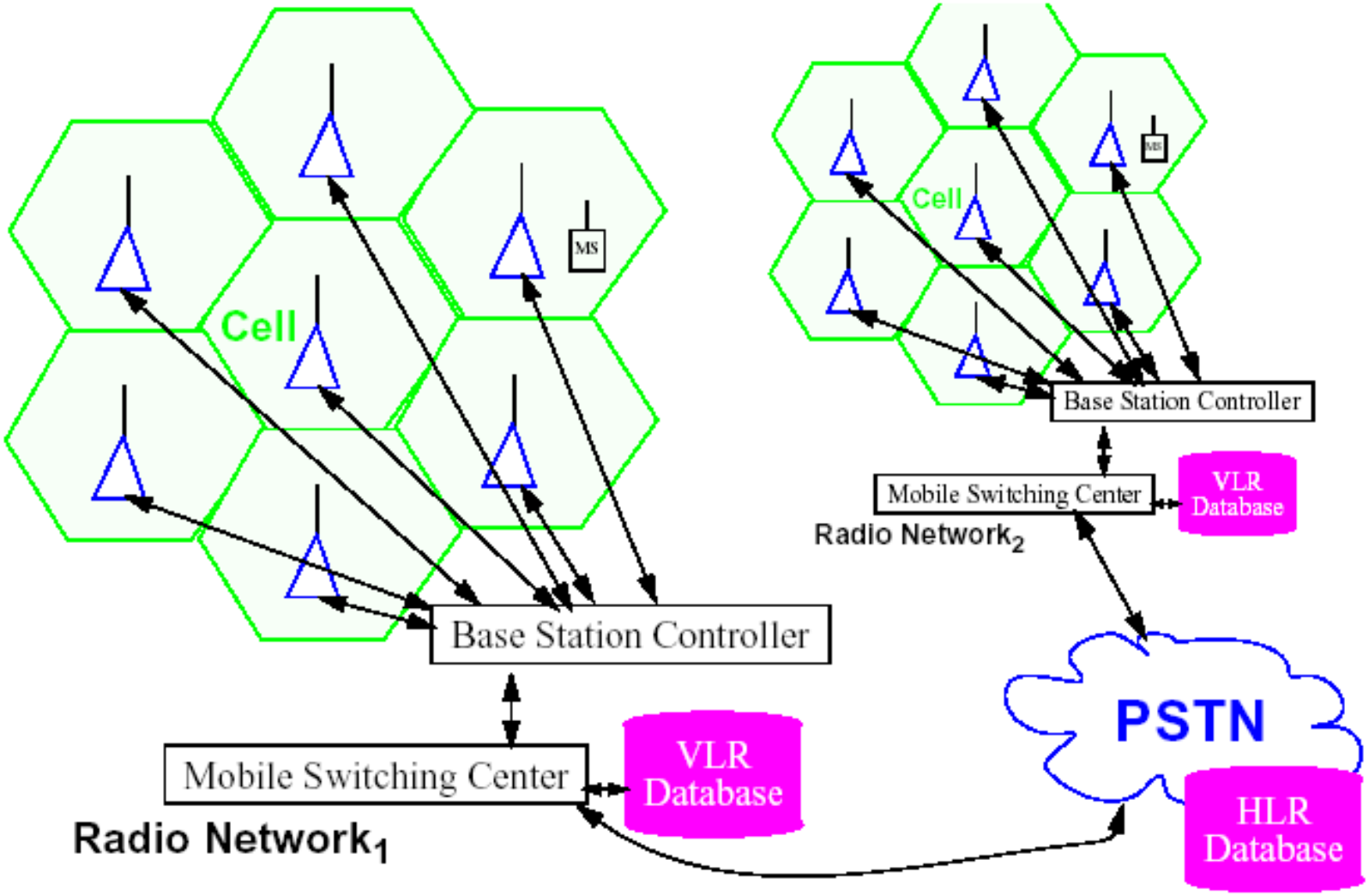
Περιεχόμενα

- **Αρχιτεκτονική κινητού δικτύου**
- Ασύμμετρο περιβάλλον επικοινωνίας χωρίς Ανοδικό Κανάλι
- Δίσκοι Εκπομπής (Broadcast Disks)
- Αλγόριθμοι για Καθαρή Εκπομπή (Pure Broadcast)
- Ασύμμετρο περιβάλλον επικοινωνίας με Ανοδικό Κανάλι
- Αλγόριθμοι για Υβριδική Εκπομπή (Hybrid Broadcast)
- Αλγόριθμοι για Κατ' Απαίτηση Εκπομπή (On-Demand Broadcast)
- Εκπομπή σε πολλαπλά κανάλια

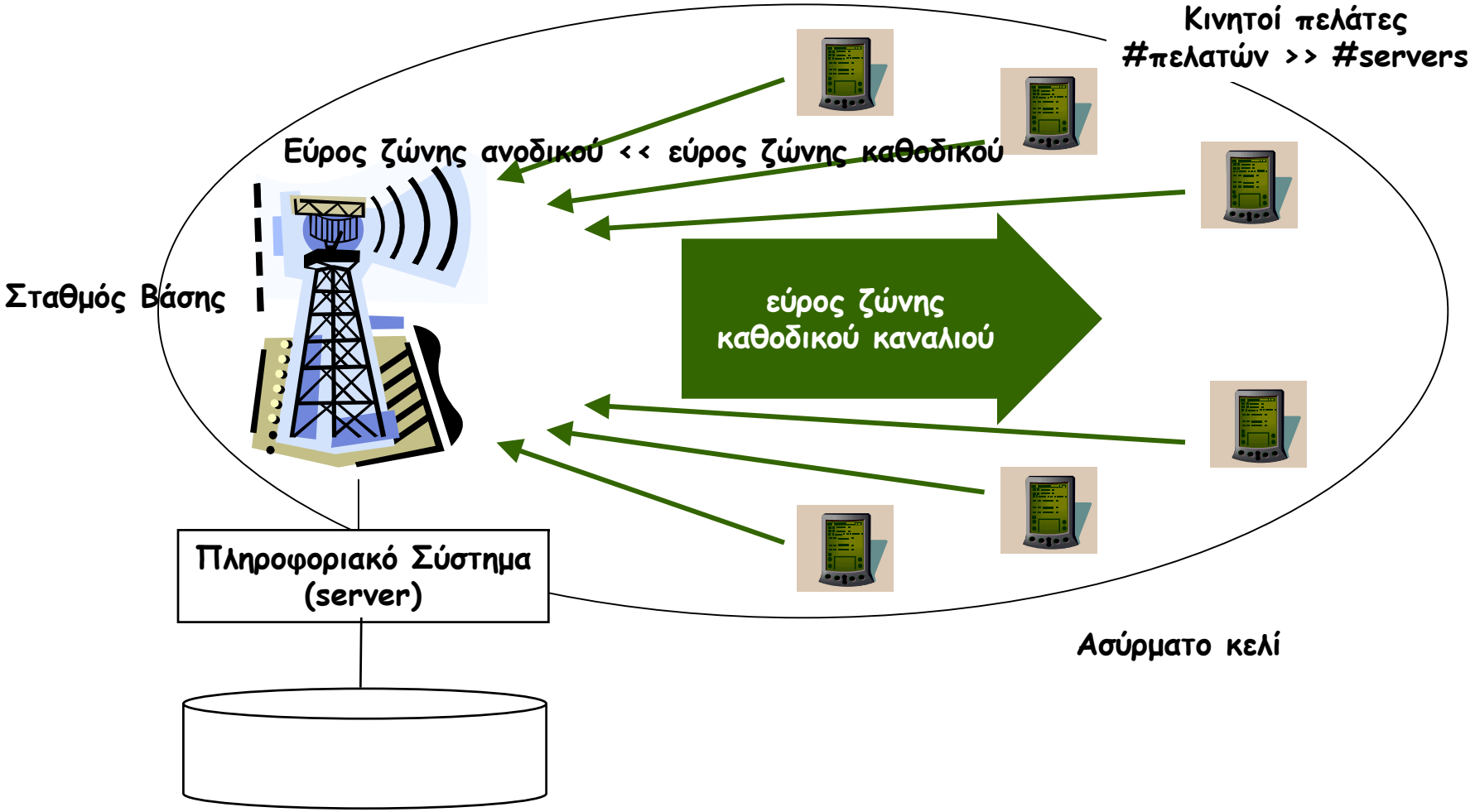
Αρχιτεκτονική κινητού δικτύου



Αρχιτ. Personal Comm. Sys. (PCS)



Γενικό μοντέλο εκπομπής



Αρχιτεκτονικές

- **Pure Pull,** δηλ. point-to-point
  - Ο πελάτης συντονίζεται στο κανάλι και κάνει την αίτηση. Ο server απαντά και κλείνει τη σύνδεση.
- **Καθαρή Εκπομπή (Pure Push)** δηλ., one-to-all or broadcast
  - Ο server "αποφασίζει" να στείλει κάποια δεδομένα και τα εκπέμπει συνεχώς & επαναλαμβανόμενα. Οι πελάτες συντονίζονται, και τα λαμβάνουν.
- **Υβριδική & Κατ' Απαίτηση Εκπομπή (Hybrid & on-demand)** broadcast
  - Κάθε πελάτης στέλνει την αίτησή του και ο server προσαρμόζει την καθαρή εκπομπή του & τις "ομαδοποιεί" και τις εκπέμπει.

Χαρακτηριστικά Pure Pull

- Μειονεκτήματα
  - Δεν μπορεί να κλιμακωθεί (scale) σε πολύ μεγάλο αριθμό κινητών χρηστών
  - Σπαταλά το εύρος ζώνης, καθώς εκπέμπει το ίδιο αντικείμενο πολλές φορές στο κανάλι (για διαφορετικούς χρήστες), όταν υπάρχει επικάλυψη στα ενδαφέροντά τους
  - Ο server χτίζεται με overcapacity, αλλά δεν αξιοποιείται όταν δεν υπάρχει μεγάλος φόρτος
- Πλεονεκτήματα
  - Είναι "διαλογικό", και συνεπώς αποφεύγει τη "σειριακή φύση" του καναλιού εκπομπής

Περιεχόμενα

- Αρχιτεκτονική κινητού δικτύου
- **Ασύμμετρο περιβάλλον επικοινωνίας χωρίς Ανοδικό Κανάλι**
- Δίσκοι Εκπομπής (Broadcast Disks)
- Αλγόριθμοι για Καθαρή Εκπομπή (Pure Broadcast)
- Ασύμμετρο περιβάλλον επικοινωνίας με Ανοδικό Κανάλι
- Αλγόριθμοι για Υβριδική Εκπομπή (Hybrid Broadcast)
- Αλγόριθμοι για Κατ' Απαίτηση Εκπομπή (On-Demand Broadcast)
- Εκπομπή σε πολλαπλά κανάλια

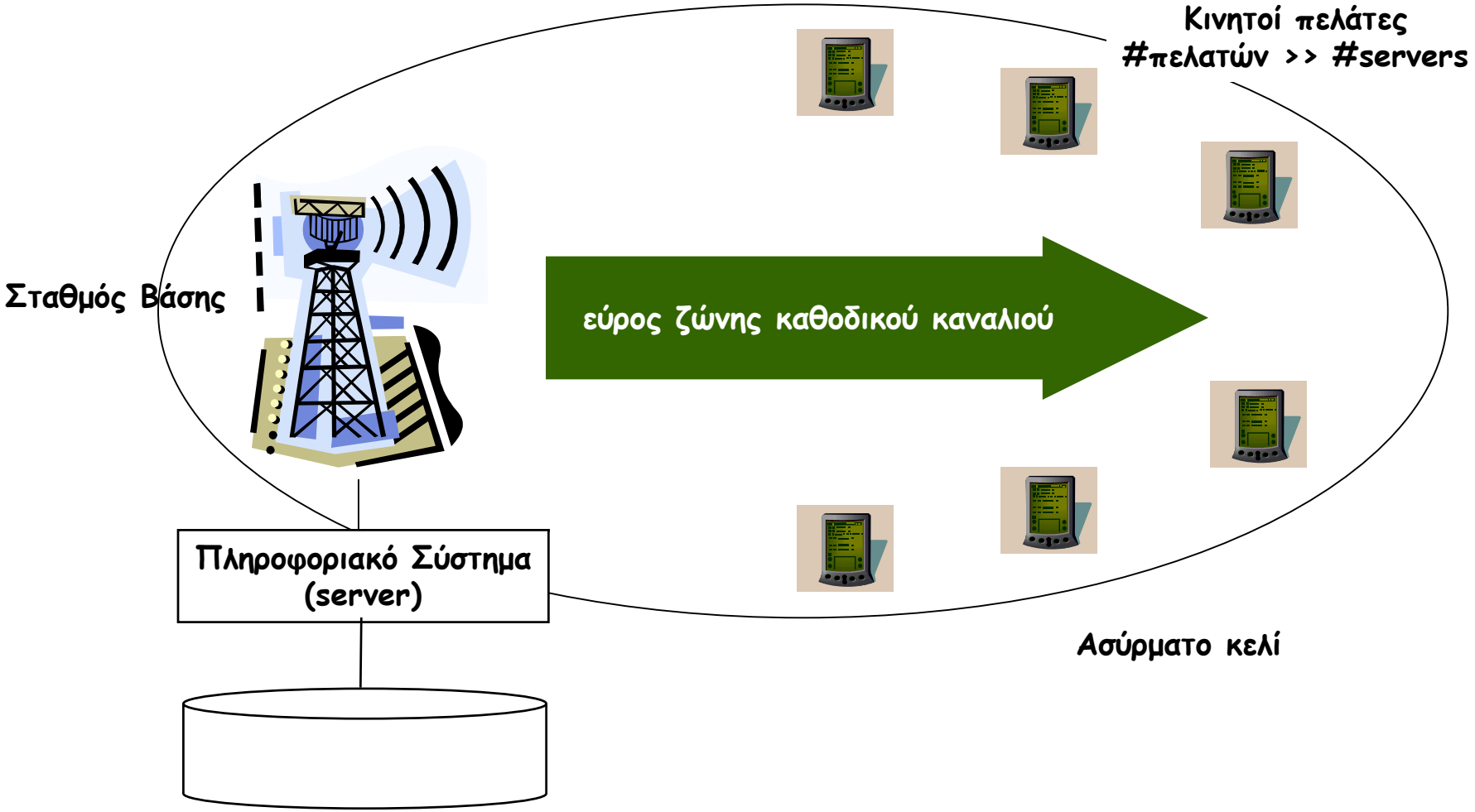
Ασύμμετρο περιβάλλον επικοινωνίας

- Σε πολλές υπάρχουσες αλλά και αναπτυσσόμενες εφαρμογές, η χωρητικότητα του καθοδικού (downstream) καναλιού επικοινωνίας από τους servers προς τους πελάτες είναι πολύ μεγαλύτερη από τη χωρητικότητα του καναλιού από τους πελάτες προς τους servers
- Ασυμμετρία επικοινωνίας μπορεί να προκύψει για δυο λόγους
  - Οι περιορισμοί στο εύρος ζώνης του φυσικού μέσου επικοινωνίας. Π.χ., οι στατικοί servers έχουν ισχυρούς (ανα)μεταδότες, ενώ οι κινητοί πελάτες έχουν μικρή ή καθόλου δυνατότητα μετάδοσης.
  - Εξαιτίας του προτύπου ροής πληροφορίας στην εφαρμογή. Π.χ., ένα σύστημα ανάκτησης πληροφορίας όπου ο αριθμός των πελατών είναι πολύ μεγαλύτερος από τον αριθμό των servers είναι ασύμμετρο, επειδή δεν υπάρχει αρκετή χωρητικοτητα (είτε στο δίκτυο είτε στους servers) για να εξυπηρετηθούν όλες οι ταυτόχρονες αιτήσεις που μπορεί να συμβούν.

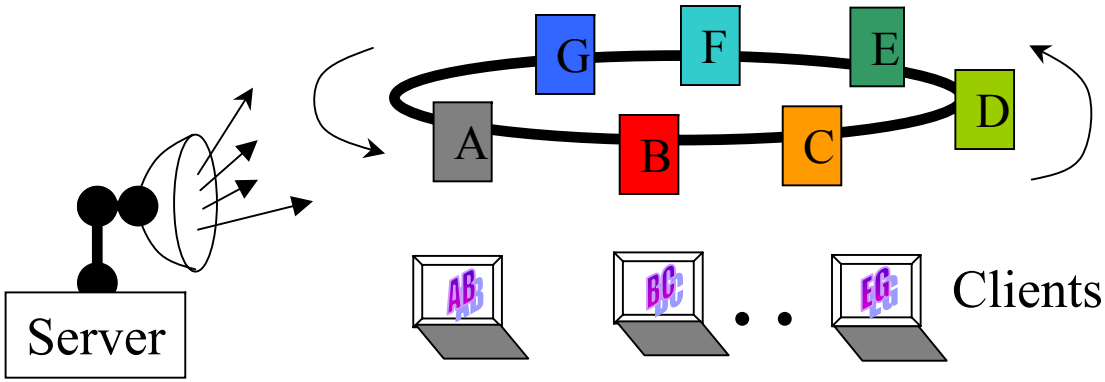
Χαρακτηριστικά περιβάλλοντος

- Στη διάλεξη αυτή, εστιάζουμε σε ένα "περιορισμένο" περιβάλλον εκπομπής
- Η πληθυσμός των κινητών χρηστών και οι προτιμήσεις τους σε δεδομένα δεν αλλάζουν με ταχείς ρυθμούς
- Τα δεδομένα είναι προς ανάγνωση μόνο και έχουν το ίδιο μέγεθος
- Οι πελάτες παίρνουν τα δεδομένα τους από το κανάλι εκπομπής, δεν υπάρχει prefetching
- Δεν χρησιμοποιούν (ακόμα και στην περίπτωση που το διαθέτουν) το upstream κανάλι επικοινωνίας

Μοντέλο Καθαρής Εκπομπής

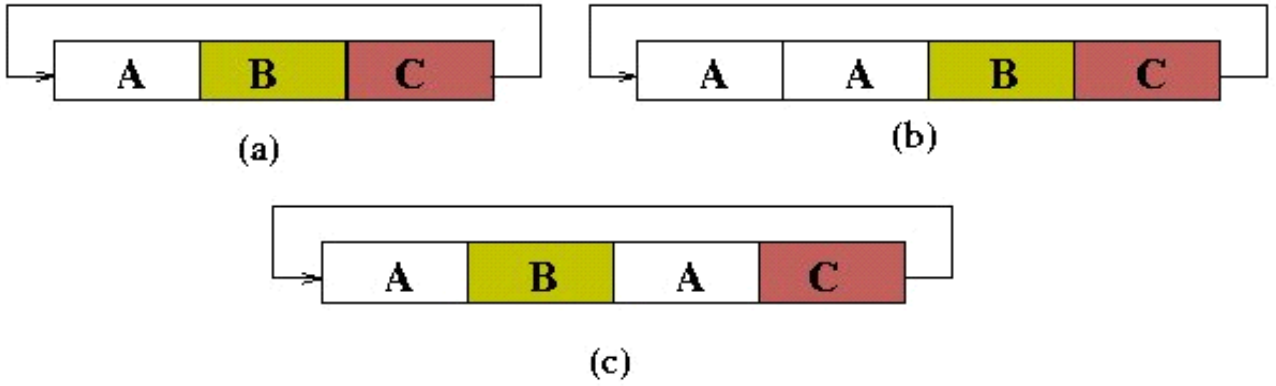


Pure push



- Πόσες φορές θα σταλεί κάποιο μέχρι να ολοκληρωθεί ένας κύκλος εκπομπής;
- Ποια δεδομένα θα αποστείλω;

Προγράμματα εκπομπής



- Τρία διαφορετικά προγράμματα εκπομπής
  a) **Επίπεδο** (Flat)
  b) **Κυρτό** (Skewed)
  c) **Πολλαπλών δίσκων** (Multi-disk)
- Κάποια εισαγάγουν replication
- Ποιο είναι το καλύτερο;
- Η **expected delay** υπολογίζεται πολ/ζοντας την πιθανότητα προσπέλασης για κάθε σελίδα επί την αναμενόμενη καθυστέρηση για τη σελίδα αυτή και αθροίζοντας τα αποτελέσματα

Μέση καθυστέρηση πρόσβασης

Πιθανότητα προσπέλασης			(Μέση) Αναμενόμενη καθυστέρηση		
A	B	C	Flat	Skewed	Multi-disk
0.333	0.333	0.333	1.50	1.75	1.67
0.50	0.25	0.25	1.50	1.63	1.50
0.75	0.125	0.125	1.50	1.44	1.25
0.90	0.05	0.05	1.50	1.33	1.10
1.0	0.0	0.0	1.50	1.25	1.00

Παρατηρήσεις

- Ο πίνακας δείχνει τρία κύρια σημεία:
  - Για ομοιόμορφες πιθανότητες προσπέλασης (1/3 η κάθε μια), το flat disk μοντέλο έχει την καλύτερη επίδοση. Το γεγονός αυτό δείχνει το θεμελιώδη περιορισμό των Δίσκων Εκπομπής, **increasing the broadcast rate of one item must necessarily decrease the broadcast rate of one or more other items**.
  - Καθώς οι πιθανότητες προσπέλασης κυρτώνονται (skewed), τα non-flat προγράμματα είναι καλύτερα.
  - Το Multi-disk πρόγραμμα πάντα είναι καλύτερο από το skewed program. Αυτό οφείλεται στο **Bus Stop Paradox**. Εάν ο inter-arrival rate (i.e., broadcast rate) μιας σελίδας είναι σταθερός, τότε η αναμενόμενη καθυστέρηση για μια αίτηση που γίνεται σε τυχαίο χρόνο είναι ίση με το μισό του χρόνου μεταξύ διαδοχικών εκπομπών. Εάν υπάρχει διακύμανση, τότε τα κενά θα έχουν διαφορετικά μήκη και η πιθανότητα να φτάσει μια αίτηση κατά τη διάρκεια μεγάλου κενού είναι μαγαλύτερη απ΄την πιθανότητα να φτάσει κατά τη διάρκεια μικρού κενού. Επομένων, η αναμενόμενη καθυστέρηση αυξάνει καθώς αυξάνει η διακύμανση.

Επιθυμητές ιδιότητες εκπομπής

- Οι inter-arrival times διαδοχικών εμγανίσεων πρέπει να είναι σταθεροί για το κάθε αντικείμενο.
- Πρέπει να υπάρχει σαφής διάκριση αρχής και τέλους του προγράμματος και το πρόγραμμα να επαναλαμβάνειται μετά το τέλος του, δηλ., να είναι περιοδικό.

Περιεχόμενα

- Αρχιτεκτονική κινητού δικτύου
- Ασύμμετρο περιβάλλον επικοινωνίας χωρίς Ανοδικό Κανάλι
- **Δίσκοι Εκπομπής (Broadcast Disks)**
- Αλγόριθμοι για Καθαρή Εκπομπή (Pure Broadcast)
- Ασύμμετρο περιβάλλον επικοινωνίας με Ανοδικό Κανάλι
- Αλγόριθμοι για Υβριδική Εκπομπή (Hybrid Broadcast)
- Αλγόριθμοι για Κατ' Απαίτηση Εκπομπή (On-Demand Broadcast)
- Εκπομπή σε πολλαπλά κανάλια

Δίσκοι εκπομπής (Broadcast Disks)

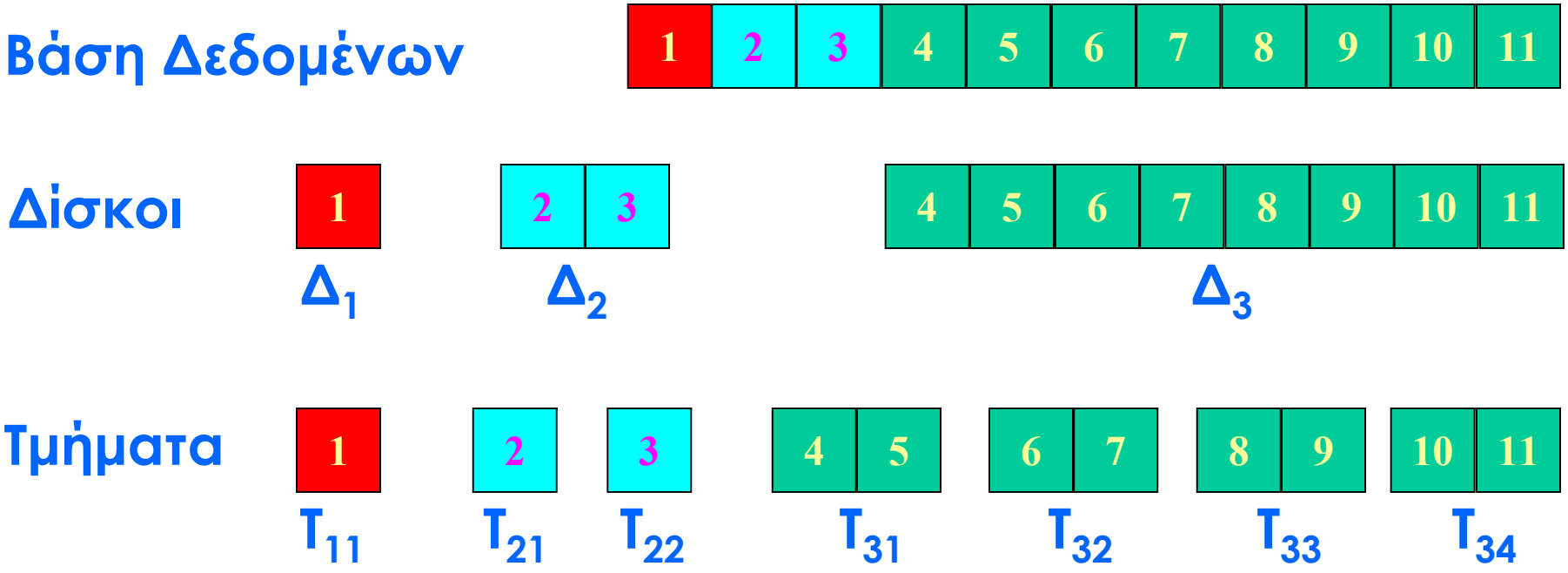
1. **Διατάσσουμε** τις σελίδες (αντικείμενα) από το πιο δημοφιλές (hottest) στο λιγότερο δημοφιλές.
2. **Διαμερίζουμε** τη λίστα των σελίδων σε πολλαπλές διαμερίσεις, όπου η κάθε διαμέριση περιέχει σελίδες με παρόμοιες πιθανότητες προσπέλασης. Αυτές οι διαμερίσεις θα αποκαλούνται **Δίσκοι**.
3. **Επιλέγουμε** τις σχετικές **συχνότητες εκπομπής** του κάθε Δίσκου. Ο μόνος περιορισμός στις σχετικές συχνότητες είναι ότι πρέπει να είναι ακέραιοι. Για παράδειγμα, δεδομένων δυο Δίσκων, ο Δίσκος 1 μπορεί να εκπέμπεται τρεις φορές για κάθε δυο φορές που εκπέμπεται ο Δίσκος 2, thus, rel.freq(l)=3, και rel_freq(2)=2.
4. **Διασπάμε** κάθε Δίσκο σε έναν αριθμό μικρότερων μονάδων. Αυτές οι μονάδες αποκαλούνται **chunks** (το C_{ij} αναφέρεται στο j-οστό chunlk του Δίσκου i). Πρώτα, υπολογίζουμε το **max_chunks** ως το Ε.Κ.Π. των σχετικών συχνοτήτων. Κατόπιν, διασπάμε κάθε Δίσκο disk i σε num_chunks(i)=max_chunks/rel_freq(i) chunks. Στο προηγούμενο παράδειγμα, το num_chunks(l) θα ισούται με 2, και το num-chunks(2) θα ισούται με 3.

Δίσκοι εκπομπής (Broadcast Disks)

- Το πρόγραμμα εκπομπής δημιουργείται με τη "συνύφανση" chunks του κάθε δίσκου με τον ακόλουθο τρόπο

```
01 for i := 0 to max_chunks - 1
02     for j := 1 to num_disks
03         Broadcast chunk C_{j,(i mod num_chunks(j))}
04     endfor
05 endfor
```

Δίσκοι Εκπομπής (παράδειγμα)



Παρατηρήσεις

- Ο αλγόριθμος παράγει μη περιοδικό πρόγραμμα εκπομπής με σταθερό inter-arrival times για κάθε σελίδα.
- Μερικά broadcast μπορεί να είναι αχρησιμοποιήτα, εάν δεν είναι δυνατό να διαιρέσουμε ακριβώς έναν δίσκο στα αντίστοιχα chunks (π.χ., στο βήμα 4 του αλγορίθμου).
- Αναμένεται ότι ο αριθμός των δίσκων θα είναι μικρός, (συνήθως 2 ή 5) και ο αριθμός των σελίδων προς εκπομπή πάρα πολύ μεγάλος, ώστε ο αριθμός των μη χρησιμοποιημένων slots να είναι ασήμαντος.

Παρατηρήσεις

- Τρεις παράγοντες μπορούν να χρησιμοποιηθούν για να επηρεάσουν το σχήμα του προγράμματος
  - Ο αριθμός των δίσκων καθορίζει τον αριθμό των διαφορετικών συχνοτήτων με τις οποίες θα εκπεμφθούν οι σελίδες.
  - Για κάθε δίσκο, ο αριθμός των σελίδων του.
  - Οι σχετικές συχνότητες καθορίζουν το μέγεθος του κύκλου εκπομπής και τον ρυθμό άφιξης κάθε σελίδας.
- Διαισθητικά
  - Οι γρήγοροι δίσκοι θα έχουν λίγες σελίδες, αν και δεν επιβάλεται από το μοντέλο
  - Ο μόνος περιορισμός αφορά τις σχετικές συχνότητες των δίσκων: να είναι ακέραιοι
  - Είναι πιθανό να έχουμε έναν δίσκο που περιστρέφεται 141 φορές για κάθε 98 φορές που περιστρέφεται ένας αργός δίσκος. Όμως, αυτό το κλάσμα έχει ως αποτέλεσμα μεγάλη περίοδο (δηλ., σχεδόν 14,000 περιστροφές του γρήγορου δίσκου). Επιπλέον, θα πρέπει ο αργός δίσκος να μπορεί να διασπαστεί σε 141 περίπου ίσα chunks. Άλλωστε, τέτοια κλάσματα δεν επιφέρουν σημαντική βελτίωση.

Κριτική των Δίσκων Εκπομπής (ΔΕ)

- Παρέχουν έναν κομψό τρόπο για τη δημιουργία προγράμματος εκπομπής (broadcast schedule), δημιουργώντας μια εναέρια μνήμη
- Δεν μας δίνουν τη μεθοδολογία για την επιλογή των παραμέτρων του συστήματος

Μαθηματική θεμελίωση ΔΕ

- Υποθέτουμε ότι ένας server εκπέμπει τα περιεχόμενα μιας βάσης, η οποία περιέχει N αντικείμενα που όλα έχουν το ίδιο μέγεθος.
- Ο χρόνος διαιρείται σε μονάδες που αποκαλούνται ticks, και κάθε αντικείμενο απαιτεί ένα tick για να μεταδοθεί.
- Οι κινητοί πελάτες αιτούνται αντικείμενα (όχι συλλογές τους).
- Κάθε αντικείμενο d_i $(1 \leq i \leq N)$ έχει πιθανότητα προσπέλασης p_i.
- Οι αιτήσεις είναι εκθετικά κατανεμημένες – η πιθανότητα ότι κάποιοα αντικείμενο θα ζητηθεί σε κάθε χρονικό διάστημα είναι σταθερή.
- Ένα πρόγραμμα εκπομπής είναι μια ακολουθία από L αντικείμενα, όπου $L \geq N$. Όλα τα αντικείμενα πρέπει να εκπεμφθούν μια φορά τουλάχιστον, και μερικά μπορεί να εκπεμφθούν περισσότερες από μια φορά κατά τη διάρκεια οτυ προγράμματος.
- Στόχος είναι η δημιουργία ενός προγράμματος που ελαχιστοποιεί την αναμενόμενη καθυστέρηση (expected delay) μιας αίτησης κάτω από τους ακόλουθους περιορισμούς:
  1. Το διάστημα μεταξύ δυο διαδοχικών εκπομπών του ίδιου αντικειμένου είναι σταθερό (fixed interarrival time between all successive transmissions of a given data item).
  2. Υπάρχει η έννοια του κύκλου – δηλ., το πρόγραμμα έχει αρχή και τέλος.

Μαθηματική θεμελίωση ΔΕ

- Υποθέτοντας ότι το διάστημα μεταξύ δυο διαδοχικών εκπομπών του ίδιου αντικειμένου είναι σταθερό, τότε με βάση ένα βασικό αποτέλεσμα της θεωρία πιθανοτήτων:

  Η αναμενόμενη καθυστέρηση (expected delay) ελαχιστοποιείται όταν έχουμε εκθετικά κατανεμημένες αιτήσεις και γνωστό ρυθμό αφίξεων των αιτήσεων.

- Για ένα δεδομένο πρόγραμμα εκπομπής, η μέση αναμενόμενη καθυστέρηση για όλα τα αντικείμενα είναι επομένως:

$$\text{average expected delay} = \frac{1}{2}\sum_{i=1}^{N} p_i w_i$$

Μαθηματική θεμελίωση ΔΕ

- Προσαρμόζοντας την προηγούμενη εξίσωση τους Δίσκους Εκπομπής και υποθέτοντας ότι έχουμε K Δίσκους

$$\text{broadcast disk average expected delay} = \frac{K}{2}\left(\sum_{i=1}^{K} N_i \sum_{d_j \in C_i} p_j\right)$$

- Για δεδομένο K, πρέπει να ελαχιστοποιήσουμε το: $\sum_{i=1}^{K} N_i \sum_{d_j \in C_i} p_j$

- Συνεπώς:

Αντιμετωπίζουμε ένα πρόβλημα διαμέρισης (partitioning)

Βέλτιστη διαμέριση

- **Βέλτιστη επίλυση προβλήματος διαμέρισης**
- Επίλυση με Δυναμικό Προγραμματισμό
- Μέση καθυστέρηση α_{ij} για όλα τα αντικείμενα σε ένα δίσκο που περιέχει τα αντικείμενα από i μέχρι j:

$$a_{ij} = \frac{1}{2}(j - i + 1)\sum_{m=i}^{j} p_m \,, \quad \mathrm{j} \geq \mathrm{i}$$

- Συμβολίζοντας με $a_{optimal\_minimal}(i,j)$ τη βελτιστη λυση (οηλ., ελάχιστη μέση καθυστέρηση προσπέλασης) για την ανάθεση των αντικειμένων από το i μέχρι το N σε j Δίσκους, είναι προφανές ότι $a_{optimal\_minimal}(i,1)=a_{ij}$ (προηγούμενη εξίσωση).
- Η αναδρομική εξίσωση του δυναμικού προγραμματισμού είναι η παρακάτω:

$$a_{optimal\_minimal}(i,k) = \min_{l \in \{i, i+1, \ldots, n\}} \{a_{il} + a_{optimal\_minimal}(l, k-1)\}$$

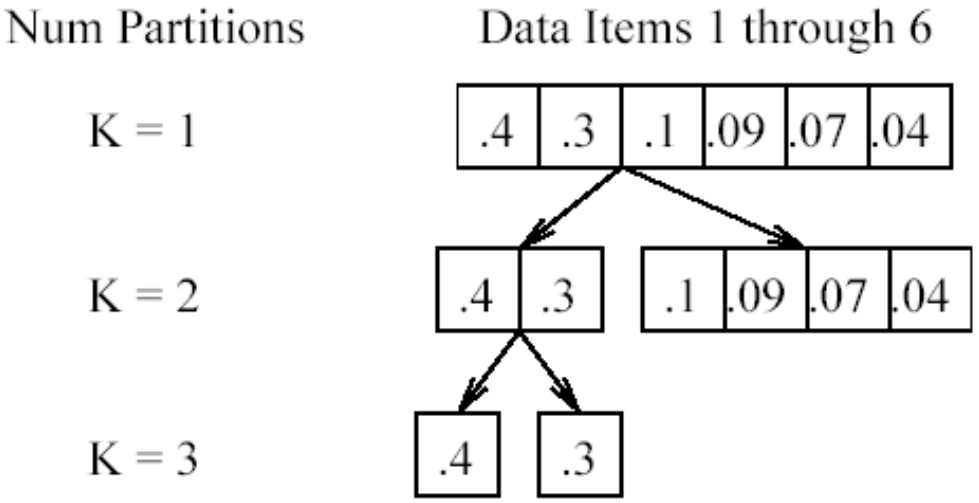
Προσεγγιστικ. αλγόριθμος GREEDY

- **Προσεγγιστική επίλυση του προβλήματος διαμέρισης** με βάση το Δυναμικό Προγραμματισμό. Είναι μέθοδος Top-Down και λέγεται **GREEDY**. Επιχειρεί να βρει το καλύτερο σημείο διαμέρισης, διασπώντας διαδοχικά με βάση τη σχέση:

$$C^s_{ij} = C_{is} + C_{s+1,j} - C_{ij}, \quad i \le s < j$$

```
Αλγόριθμος Greedy (int n, int k, float vector P)
BEGIN
numPartitions = 1;
while (numPartitions < k)
        for each partition r with data items i through j {
         // Find the best point to split in partition r
            for(s = i;  s<=j; s=s+1)   // Initialize the best split point for this partition
                                       //as the first data item. If we find a better one
                                       //subsequently, update the best split point.
                if ( (s == i) OR  (localChange > C_ij^s) )
                    localS = s;
                    localChange = C_ij^s;    // Initialize the best solution as the one for the first
                                             // partition. If we find a better one subsequently,
                                             // update the best solution.
            if ( (r == 1) OR (globalChange > localChange) )
                globalChange = localChange;
                globalS = localS;
                bestpart = r;
        }
        split partition bestpart at point globalP;
        numPartitions = numPartitions + 1;
END
```

Παράδειγμα αλγορίθμου GREEDY



By Equation 2, initial cost is 3 ticks.

Best split is at data item 2.
New cost is .7 + .6 = 1.3 ticks.

Best split is at data item 4.
New cost is .2 + .15 + .6 = .95 ticks.

Προσεγγ. αλγ. Growing Segments

- Η μέθοδος **Growing Segments** ξεκινά με μια αρχική "ελάχιστη" διαμέριση αναθέοντας ένα αντικείμενο σε κάθε δίσκο. Αυτή η διαμέριση αποτελεί τη διαμέριση "σπόρο (seed)"
- Κατόπιν, επαυξάνει κάθε τμήμα της διαμέρισης περιλαμβάνοντας τόσα αντικείμενα όσα καθορίζονται από την παράμετρο *increment* και υπολογίζει ποια από αυτές τις επαυξήσεις δίνει τη μεγαλύτερη ελάττωση στη μέση καθυστέρηση.
- Κατόπιν, επιλέγει αυτή τη διαμέριση ως το νέο σπόρο και συνεχίζει τη διαδικασίαμέχρι να καλύψει όλα τα αντικείμενα.
- Η παράμετρος *increment* είναι πολύ σημαντική και έχει το ακόλουθο trade-off: όσο μεγαλύτερη είναι η τιμή της, τόσο μικρότερη είναι η πολυπλοκότητα χρόνου εκτέλεσης του αλγορίθμου αλλά και μικρότερη η ποιότητα του παραγόμενου προγράμματος εκπομπής.

Προσεγγ. αλγ. Growing Segments

Αλγόριθμος Growing Segments (int **n**, int **k**, float vector **P**)

BEGIN

```
increment= getNextIncrement();
r0 = 1;
for(i=1; i<=k; ++i)        // initial setup of (r0, r1, · · · , rk)
    ri = ri−1+increment;
while(rk < n)
    increment = getNextIncrement();
    for(i=1; i<=k; ++i)
        p0 = r0;
        for(x=1; x<=k; ++x)
            if (x ≥ i) px = rx+increment;
            else px = rx;
        Si = (p0, p1, ... , pk);
Find a partition (r0, r1, · · · , rk) among Si's such that
     minDelay(r0, r1, · · · , rk)=min{ minDelay(Sx) } for x = 1, 2, ..., k.
return (r0, r1, ... , rk);
```

END

Περιεχόμενα

- Αρχιτεκτονική κινητού δικτύου
- Ασύμμετρο περιβάλλον επικοινωνίας χωρίς Ανοδικό Κανάλι
- Δίσκοι Εκπομπής (Broadcast Disks)
- **Αλγόριθμοι για Καθαρή Εκπομπή (Pure Broadcast)**
- Ασύμμετρο περιβάλλον επικοινωνίας με Ανοδικό Κανάλι
- Αλγόριθμοι για Υβριδική Εκπομπή (Hybrid Broadcast)
- Αλγόριθμοι για Κατ' Απαίτηση Εκπομπή (On-Demand Broadcast)
- Εκπομπή σε πολλαπλά κανάλια

Γενίκευση των Δίσκων Εκπομπής

- Αίρουμε την υπόθεση ότι όλα τα αντικείμενα έχουν το ίδιο μέγεθος
- Εξακολουθούμε να επιδιώκουμε το γεγονός ότι οι διαδοχικές εκπομπές του ίδιου αντικειμένου είναι ίσες, διότι:

  ΛΗΜΜΑ. Το πρόγραμμα εκπομπής με την *ελάχιστη συνολική μέση καθυστέρηση* είναι εκείνο στο οποίο οι εμφανίσεις κάθε αντικειμένου απέχουν πάντα το ίδιο (**equally spaced criterion**)
- Αναζητούμε πόσο συχνά πρέπει να εκπέμπεται κάθε αντικείμενο

Ο κανόνας "Τετράγωνο της Ρίζας"

- **ΘΕΩΡΗΜΑ**. Υποθέτοντας ότι οι εμφανίσεις κάθε αντικειμένου είναι equally spaced, η *ελάχιστη συνολική μέση καθυστέρηση* επιτυγχάνεται όταν η απόσταση s_i μεταξύ διαδοχικών εμφανίσεων του αντικειμένου i είναι ανάλογη προς την τετραγωνική ρίζα του μήκους του και αντιστρόφως ανάλογη προς την τετραγωνική ρίζα της πιθανότητας προσπέλασής του. Δηλαδή

$$s_i \propto \sqrt{l_i / p_i}$$

Στην περίπτωση αυτή, η *ελάχιστη συνολική μέση καθυστέρηση* είναι ίση με:

$$\frac{1}{2}\left(\sum_{i=1}^{M}\sqrt{p_i l_i}\right)^2$$

Απόδειξη του κανόνα Square-root

- Έστω ότι s_i είναι το spacing για το αντικείμενο i.
- Η *μέση καθυστέρηση πρόσβασης* στο i είναι $t_i=s_i/2$
- Η συνολική μέση καθυστέρηση πρόσβασης για όλα τα αντικείμενα είναι $$t = \frac{1}{2}\sum_{i=1}^{N} p_i * s_i$$
- Έστω ότι $r_i = l_i / s_i$, δηλ. είναι το κλάσμα του εύρους ζώνης που ανατίθεται στο αντικείμενο i, λόγου του κριτηρίου equal-spacing.
- Επομένως: $$\sum_{i=1}^{N} r_i = 1$$
- Άρα: $$t = \frac{1}{2}\sum_{i=1}^{N} \frac{p_i * l_i}{r_i}$$

Απόδειξη του κανόνα Square-root

- Αφού $\sum_{i=1}^{N} r_i = 1$
- Μόνο N-1 από τα r_i μπορούν να αλλαχτούν ανεξάρτητα. Η βέλτιστη τιμή για τα r_i επιτυγχάνεται εάν $\partial t/\partial r_i = 0$, για κάθε i. Επιλύουμε τις εξισώσεις αυτές, ξεκινώντας με $0 = \partial t/\partial r_1$.

$$
\begin{aligned}
0 = \frac{\partial t}{\partial r_1} &= \frac{1}{2}\frac{\partial}{\partial r_1}\left(\sum_{i=1}^{M}\frac{p_i l_i}{r_i}\right) \\
&= \frac{1}{2}\frac{\partial}{\partial r_1}\left(\frac{p_1 l_1}{r_1} + \sum_{i=2}^{M-1}\frac{p_i l_i}{r_i} + \frac{p_M l_M}{\left(1-\sum_{i=1}^{M-1} r_i\right)}\right) \\
&= \frac{1}{2}\left(-\frac{p_1 l_1}{r_1^2} + \frac{p_M l_M}{\left(1-\sum_{i=1}^{M-1} r_i\right)^2}\right) \\
&\Rightarrow \frac{p_1 l_1}{r_1^2} = \frac{p_M l_M}{(1-\sum_{i=1}^{M-1} r_i)^2}.
\end{aligned}
$$

Απόδειξη του κανόνα Square-root

- Όμοια ισχύει ότι: $\frac{p_2 l_2}{r_2^2} = \frac{p_M l_M}{\left(1 - \sum_{i=1}^{M-1} r_i\right)^2}$

- Από αυτές τις εξισώσεις προκύπτει ότι:

$$\frac{p_1 l_1}{r_1^2} = \frac{p_2 l_2}{r_2^2} \Rightarrow \frac{r_1}{r_2} = \sqrt{\frac{p_1 l_1}{p_2 l_2}}$$

- Και γενικά ότι:

$$\frac{r_i}{r_j} = \sqrt{\frac{p_i l_i}{p_j l_j}}, \quad \forall i, j$$

Απόδειξη του κανόνα Square-root

- Αυτό σημαίνει ότι η βέλτιστη τιμή του r_i πρέπει να είναι γραμμικά ανάλογη προς το: $\sqrt{l_i * p_i}$
- Εύκολα διαπιστώνουμε ότι υπάρχει η σταθερά αναλογίας α ίση με : $a = 1 / \sum_{j=1}^{N} \sqrt{l_j * p_j}$
- ώστε $r_i = a * \sqrt{l_i * p_i}$ είναι η μόνη δυνατή λύση για τις εξισώσεις $\partial t/\partial r_i = 0$
- Με αντικατάσταση των r_i στη σχέση που δίνει το t βρίσκουμε τη βέλτιστη τιμή του.

Αλγόριθμος με βάση τον κανόνα S-r

- Το Θεώρημα πρακτικά υπονοεί ότι για να επιτύχουμε τη βέλτιστη επίδοση (ελάχιστη συνολική μέση καθυστέρηση), πρέπει να ισχύει για το spacing s_i κάθε αντικειμένου i ότι:

$$\frac{s_i^2 p_i}{l_i} = \text{constant}$$

- Ορίζοντας για κάθε αντικείμενο j τη συνάρτηση:

$$G(j) = (Q - R(j))^2 * p_j / l_j$$

- όπου Q είναι η τρέχουσα χρονική στιγμή και R(j) η στιγμή τελευταίας εκπομπής του αντικειμένου j, έχουμε τον επόμενο αλγόριθμο εκπομπής:

Αλγόρ. εκπομπής Vaidya-Hameed

Broadcast scheduling algorithm A

Step 1. Determine maximum value of $G(j)$ over all items j, $1 \leqslant j \leqslant M$. Let $G_{\max}$ denote the maximum value of $G(j)$.

Step 2. Choose item i such that $G(i) = G_{\max}$. If this equality holds for more than one item, choose any one of them arbitrarily.

Step 3. Broadcast item i at time Q.

Step 4. $R(i) = Q$.

Παράδειγμα Vaidya-Hammed

Example 1. Consider a database containing 3 items such that $p_1 = 1/2$, $p_2 = 3/8$, and $p_3 = 1/8$. Assume that items have lengths $l_1 = 1$, $l_2 = 2$ and $l_3 = 4$ time units. Figure 1 shows the items recently broadcast by the server (up to time < 100). The above algorithm is called to determine the item to be transmitted at time 100. Thus, $Q = 100$. Also, from figure 1, observe that $R(1) = 95$, $R(2) = 93$, and $R(3) = 96$. The algorithm evaluates function $G(j) = (Q - R(j))^2 p_j / l_j$ for $j = 1, 2, 3$ as 12.5, $147/16$ $(= 9.1875)$ and 0.5, respectively. As $G(j)$ is the largest for $j = 1$, item 1 is transmitted at time 100.

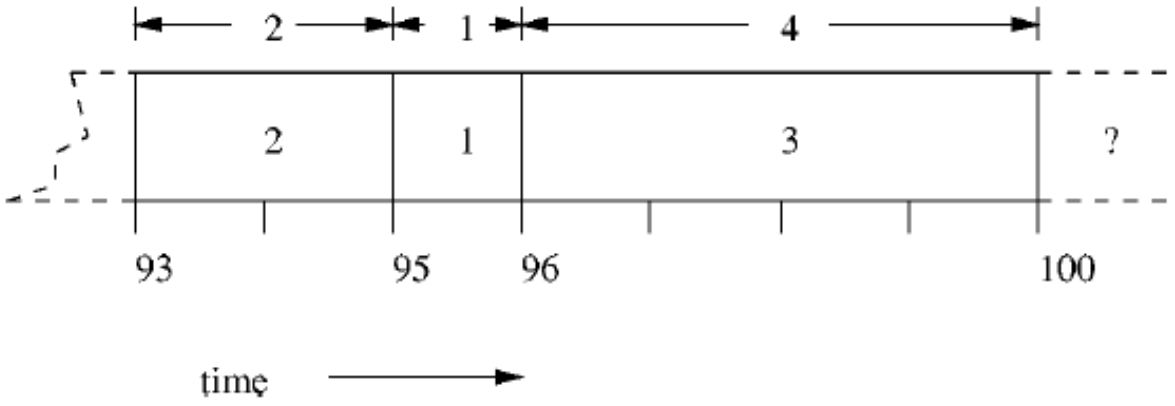


Figure 1. Example 1.

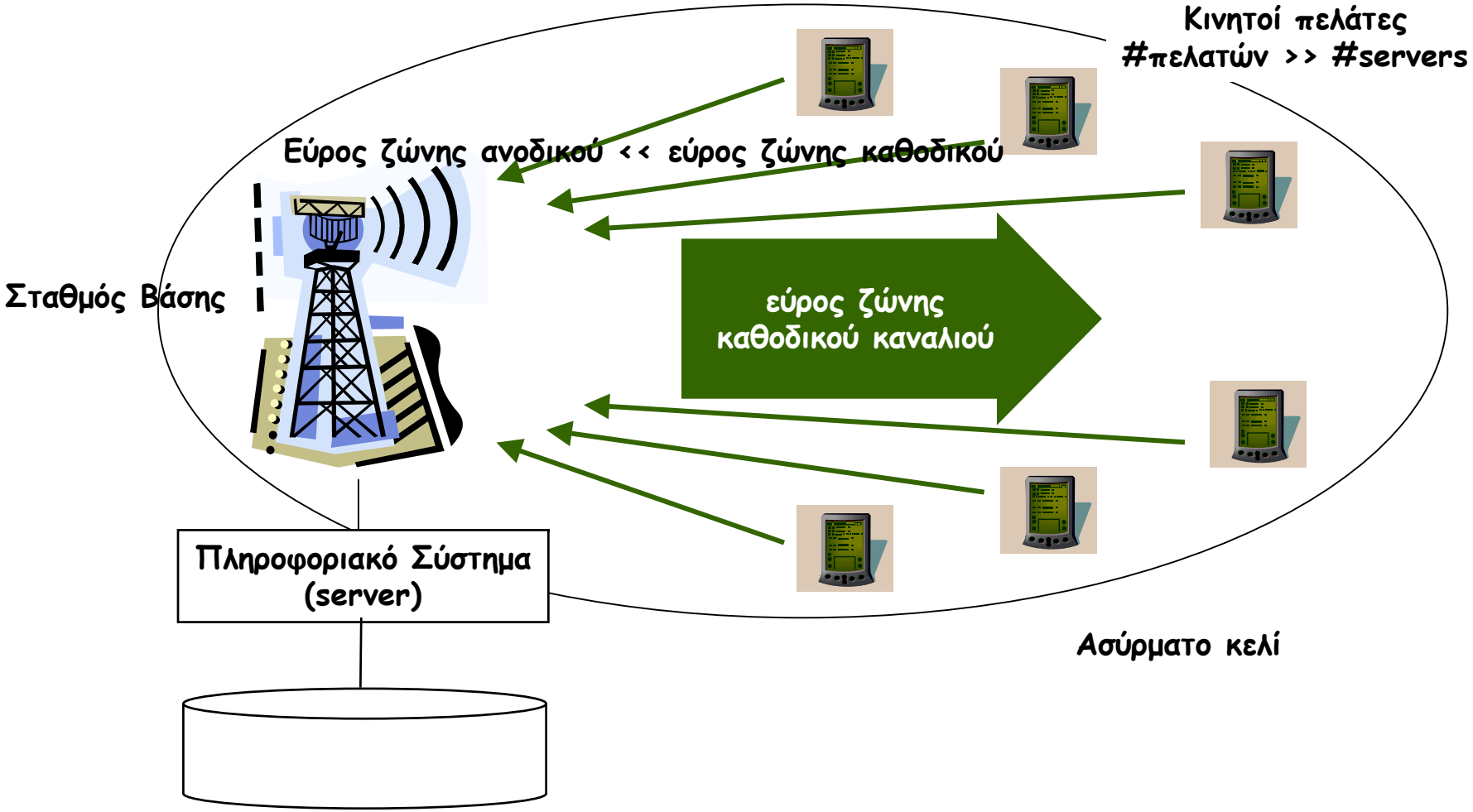
Περιεχόμενα

- Αρχιτεκτονική κινητού δικτύου
- Ασύμμετρο περιβάλλον επικοινωνίας χωρίς Ανοδικό Κανάλι
- Δίσκοι Εκπομπής (Broadcast Disks)
- Αλγόριθμοι για Καθαρή Εκπομπή (Pure Broadcast)
- **Ασύμμετρο περιβάλλον επικοινωνίας με Ανοδικό Κανάλι**
- Αλγόριθμοι για Υβριδική Εκπομπή (Hybrid Broadcast)
- Αλγόριθμοι για Κατ' Απαίτηση Εκπομπή (On-Demand Broadcast)
- Εκπομπή σε πολλαπλά κανάλια

Μειονεκτήματα Καθαρής Εκπομπής

- Υποθέτει ότι τα ενδιαφέροντα των χρηστών (hot spots) είναι σχετικά σταθερά
  - Οι ανάγκες σε δεδομένα δεν μπορούν να προβλεφθούν
  - Οι ανάγκες σε δεδομένα δεν είναι σταθερές
- Πώς θα ενημερώνεται ο server για την επιτυχία του προγράμματος εκπομπής;
- Προσαρμογή της Καθαρής Εκπομπής με ενημέρωση για τις ανάγκες σε δεδομένα

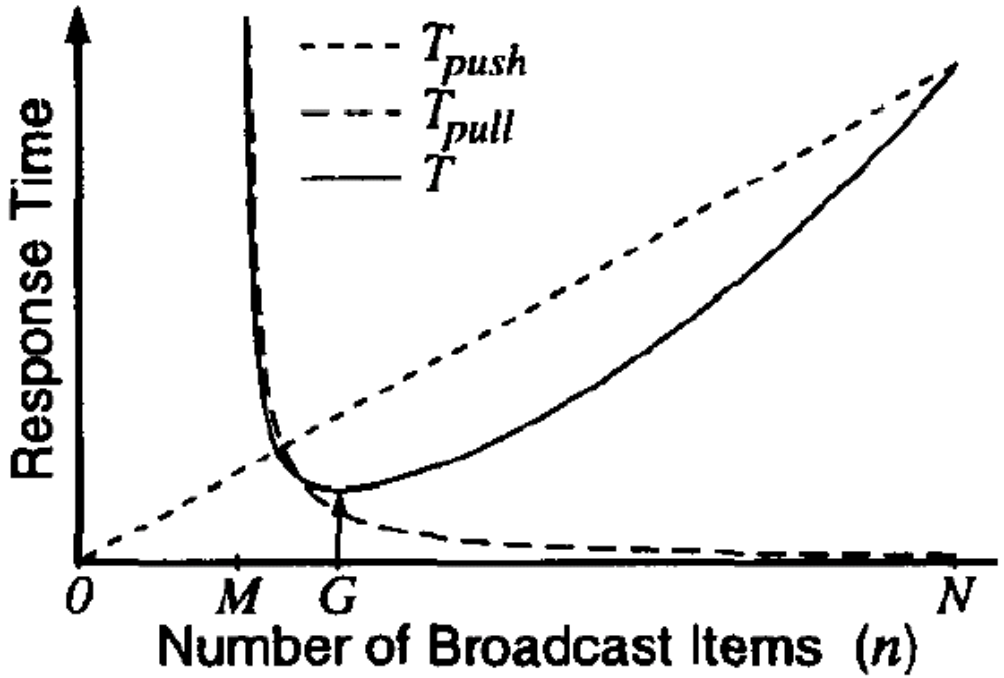
Εκπομπή με Ανοδικό Κανάλι



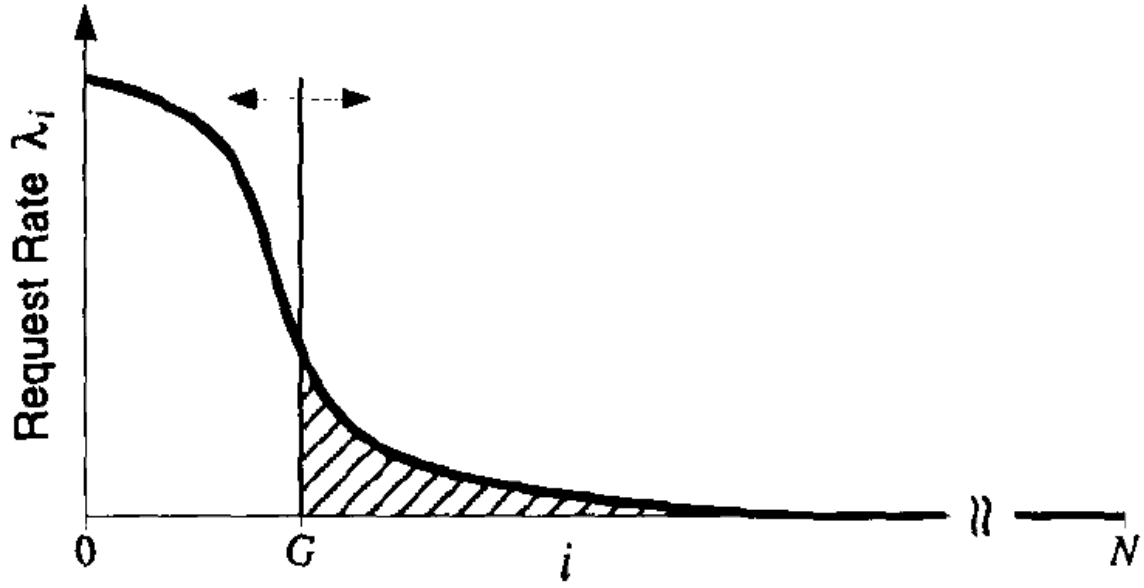
Κίνητρο για νέο μοντέλο εκπομπής

- Βάση N αντικειμένων, λ_i Poisson ρυθμός αιτήσεων ($\lambda_1 > \lambda_2 > \lambda_3 > \lambda_4 \ldots > \lambda_N$) και ο μέσος ρυθμός εξυπηρέτησης είναι 1/μ. Ο Server έχει κανάλι εκπομπής εύρους B και αποφασίζει να εκπέμψει τα πρώτα n και τα υπόλοιπα με pull. Έστω $\Lambda_k=\Sigma\lambda_i$ (1<=i<=k). Τότε:

$$T_{pull} = \frac{1}{\mu-(\Lambda_N-\Lambda_n)} \qquad T_{push} = \frac{n\,S}{2\,B}$$



Κύρτωση στο πρότυπο προσπέλασης



- **Στόχος**: Εύρεση του βέλτιστου σημείου G για να στείλουμε με Καθαρή Εκπομπή τα δεδομένα αριστερά του G και με Pull τα δεδομένα δεξιά του G.
- **Ιδιότητες του G**:
  - Η ουρά να διατηρείται *κάτω* από την pull capacity
  - Η κεφαλή να αρκετά *πλατιά*, ώστε να περικλείει τα hot αντικείμενα, αλλά όχι πολύ πλατιά για να μην περικλείει όσα ζητούνται σπάνια.

Περιεχόμενα

- Αρχιτεκτονική κινητού δικτύου
- Ασύμμετρο περιβάλλον επικοινωνίας χωρίς Ανοδικό Κανάλι
- Δίσκοι Εκπομπής (Broadcast Disks)
- Αλγόριθμοι για Καθαρή Εκπομπή (Pure Broadcast)
- Ασύμμετρο περιβάλλον επικοινωνίας με Ανοδικό Κανάλι
- **Αλγόριθμοι για Υβριδική Εκπομπή (Hybrid Broadcast)**
- Αλγόριθμοι για Κατ' Απαίτηση Εκπομπή (On-Demand Broadcast)
- Εκπομπή σε πολλαπλά κανάλια

Χαρακτηρισμός δεδομένων

- Ατμός (Vapor)
  - Ζητούνται πάρα πολύ
- Υγρά (Liquid)
  - Δεν στέλνοται από τον server με καθαρή εκπομπή, αλλά ο server έχει λάβει μέτριο ή μικρό αριθμό αιτήσεων γι' αυτά
- Παγωμένα (Frigid)
  - Δεν έχουν ζητηθεί για αρκετό διάστημα και η "θερμοκρασία" τους λ_i έχει πέσει πρακτικά στο ΜΗΔΕΝ.
- **Ζήτημα**: Πώς θα διαχωρίσουμε τα Vapor από τα Liquid, αφού για τα πρώτα δεν έρχονται αιτήσεις στον server;

Υβριδική Εκπομπή

- Κάθε δεδομένο έχει μια *Θερμοκρασία*
- Μια ουρά περιέχει όλα τα Vapor δεδομένα
- Εκπέμπεται πάντα αυτό στην κεφαλή της ουράς
- Όταν εκπεμφθεί, η Θερμοκρασία του ελαττώνεται κατά CoolingFactor (0,1) και τοποθετείται στο τέλος της ουράς
- Όταν εκπεμφθούν όλα, γίνεται εκτίμηση των περιεχομένων της ουράς
  - Αναγνωρίζονται τα Vapor δεδομένα που θα γίνουν Liquid
  - Αναγνωρίζονται τα Liquid δεδομένα που θα γίνουν Vapor

Υβριδική Εκπομπή

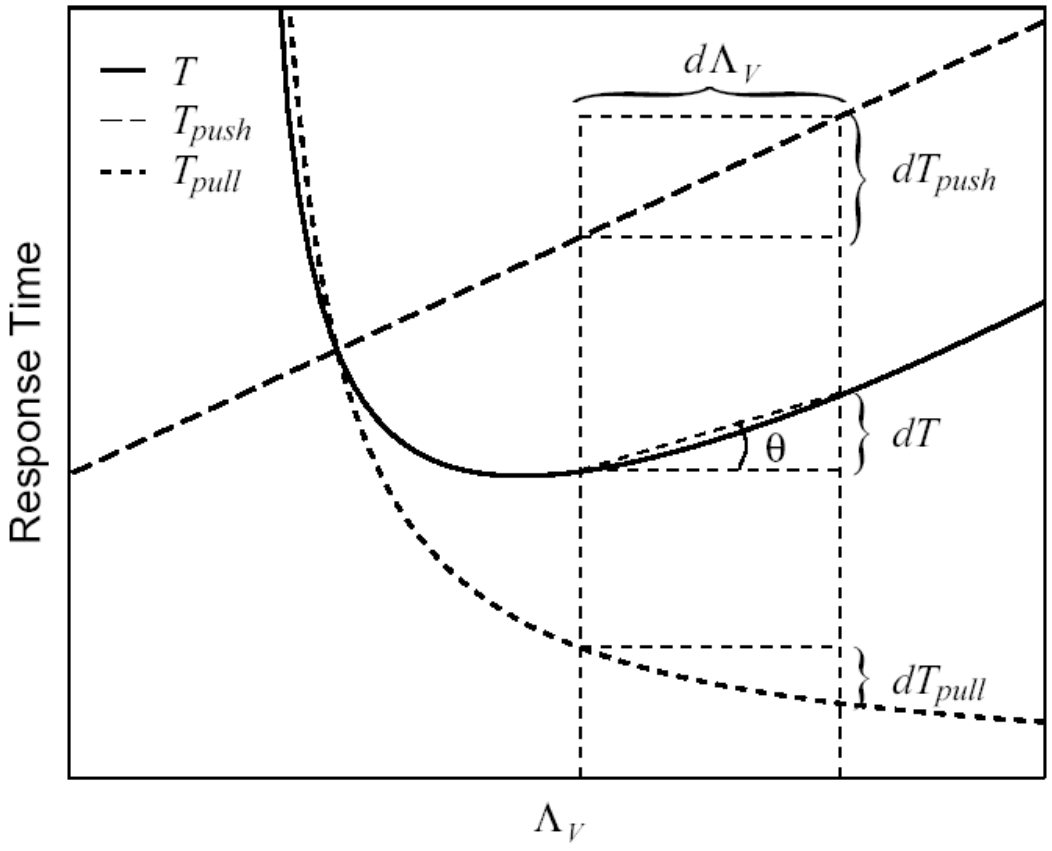
- Υπολογίζουμε το *συνολικό αναμενόμενο περιθωριακό κέρδος* dT

$$dT = \frac{\Lambda_V \, dT_{push} + \Lambda_L \, dT_{pull}}{\Lambda_V + \Lambda_L}$$

- Όπου

$$dT_{push} = A \, \frac{S_i + (2N_V + A)\, I}{2\, B}$$

$$dT_{pull} = T_{pull} \, \frac{d\Lambda_V}{\mu - \Lambda_L + d\Lambda_V}$$



Αλγόριθμος Υβριδικής Εκπομπής

l = hottest liquid item
v = coldest vapor item
while $(\lambda_v < \lambda_l)$
    demote v
    v = coldest vapor item
end
while $(-dT/\lambda_v > tan\theta_0)$
    demote v
    v = coldest vapor item
end
l = hottest liquid item
while $(dT/\lambda_l < tan\theta_0)$
    promote l
    l = hottest liquid item
end

Ο PLPCHD Υβριδικός αλγόριθμος

Push Less Pull the Current Highest Demanded Item

- Βάση D ισομεγεθών αντικειμένων ταξινομημένων με φθίνουσα πιθανότητα προσπέλασης. Τα K πρώτα επιλέγονται για Καθαρή Εκπομπή και τα υπόλοιπα D-K για Pure Pull. Ο μέσος χρόνος αναμονής είναι:

$$T_{exp-hyb} = T_{exp-acc} + T_{exp-res} = \sum_{i=1}^{K} P_i \cdot \overline{T_{acc,i}} + \sum_{i=K+1}^{D} P_i \cdot \overline{T_{res,i}}.$$

Επειδή $S_i = \frac{\sum_{j=1}^{K} \sqrt{\hat{P}_j}}{\sqrt{\hat{P}_i}}$ Κανονικοποιώντας $\hat{P}_i = \frac{P_i}{\sum_{j=1}^{K} P_j}$

Και $T_{res,i}$= D-K (δηλ., το μέγιστο μήκος της ουράς, έχουμε ότι

$$T_{exp-hyb}(K) = \sum_{i=1}^{K} S_i P_i + \sum_{i=K+1}^{D} P_i * (D - K)$$

Ο PLPCHD Υβριδικός αλγόριθμος

Integer function CUT-OFF POINT $(D, P = P_1, P_2 ... P_D) : K$

/* D: Total No. Of items in the Database of the server

P: Sorted vector of access probability of items in decreasing order

K: Optimal Cut off Point */

$K := 1; T_{exp-hyb}(0) := T_{exp-hyb}(1) := D;$

while $K \leq D$ **and** $T_{exp-hyb}(K-1) \geq T_{exp-hyb}(K)$ **do**

begin

Set $s_i = \frac{\sum_{j=1}^{k} \sqrt{\hat{P}_j}}{\sqrt{\hat{P}_i}}$, **where** $\hat{P}_i = \frac{P_i}{\sum_{j=1}^{K} P_j}$,

$T_{exp-hyb}(K) = \sum_{i=1}^{K} S_i P_i + \sum_{i=K+1}^{D} P_i * (D - K);\ K := K + 1;$

end

return $(K-1)$

Ο PLPCHD Υβριδικός αλγόριθμος

Procedure SCHEDULING

while true **do**
begin
compute an item from the push scheduling and broadcast it;
if the pull-queue is not empty **then**
extract the most requested item from the pull-queue,
clear the number of pending requests for that item, and pull-it
end;

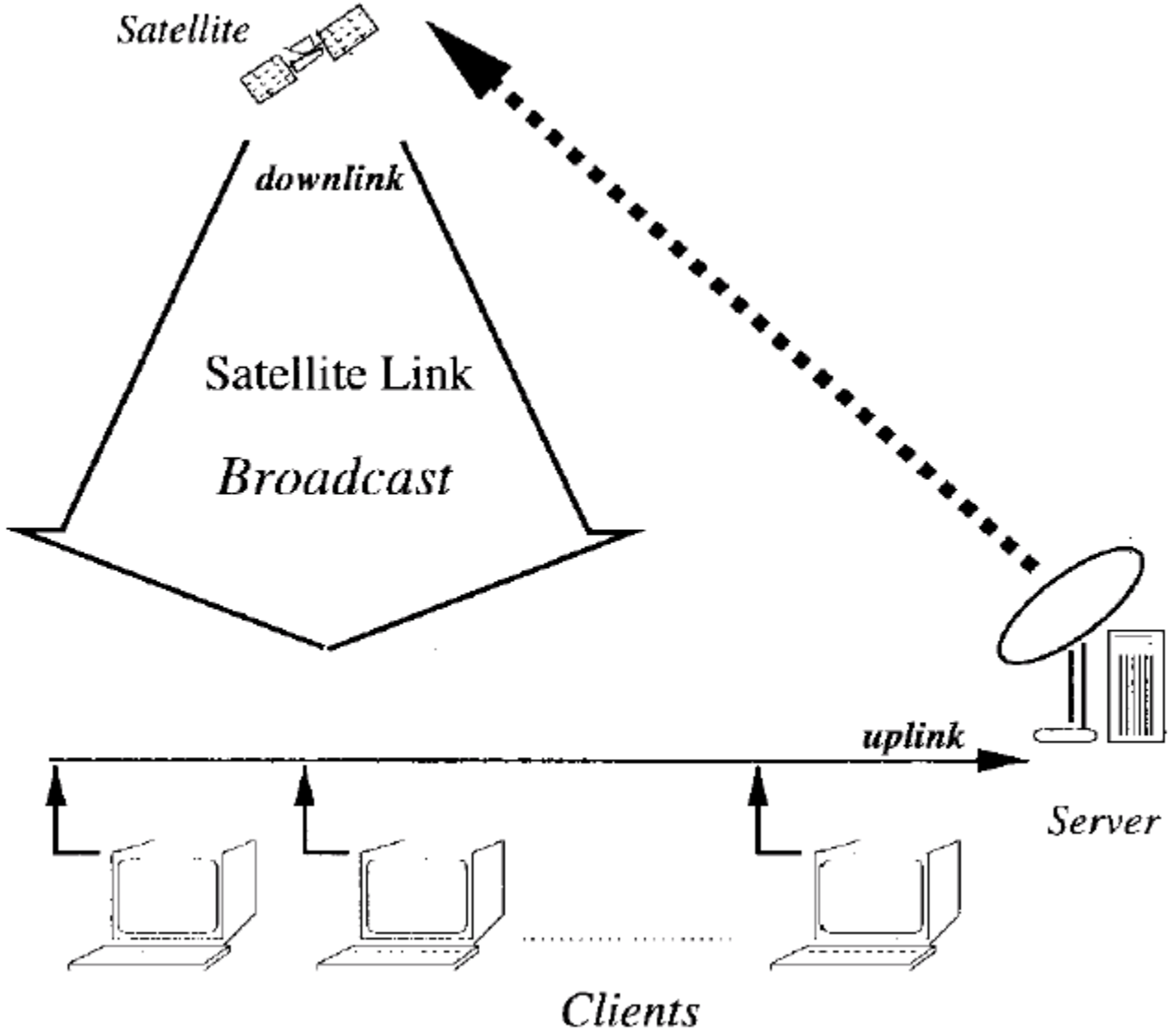
Περιεχόμενα

- Αρχιτεκτονική κινητού δικτύου
- Ασύμμετρο περιβάλλον επικοινωνίας χωρίς Ανοδικό Κανάλι
- Δίσκοι Εκπομπής (Broadcast Disks)
- Αλγόριθμοι για Καθαρή Εκπομπή (Pure Broadcast)
- Ασύμμετρο περιβάλλον επικοινωνίας με Ανοδικό Κανάλι
- Αλγόριθμοι για Υβριδική Εκπομπή (Hybrid Broadcast)
- **Αλγόριθμοι για Κατ' Απαίτηση Εκπομπή (On-Demand Broadcast)**
- Εκπομπή σε πολλαπλά κανάλια

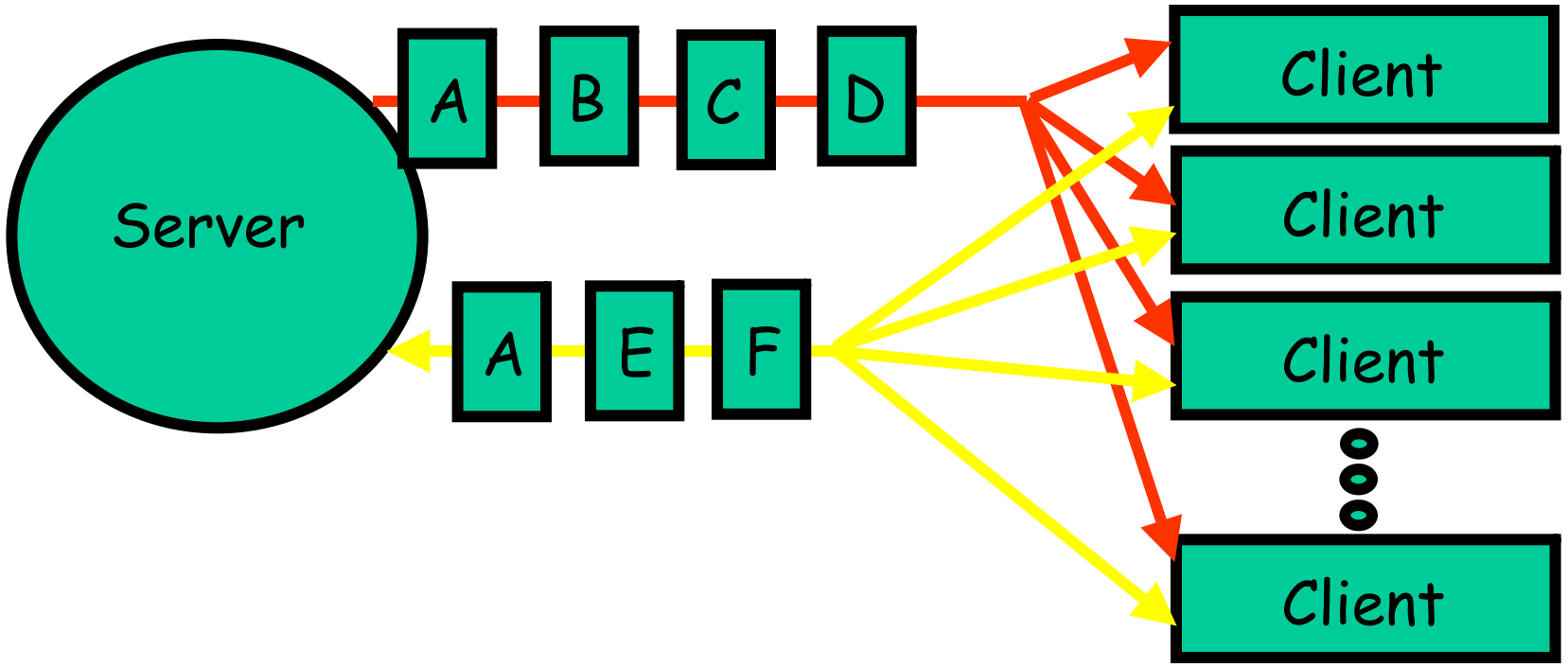
Μειονεκτήματα Υβριδικής Εκπομπής

- Η συγγένειά της με την Καθαρή Εκπομπή
  - Προσπαθεί να εκτιμήσει τις πραγματικές πιθανότητες προσπέλασης των αντικειμένων
  - Δεν κάνει βέλτιστη χρήση του καναλιού εκπομπής
- Στηρίζεται σε παραμέτρους που ρυθμίζονται διοικητικά
- Δεν έχει άμεση αντίδραση σε ραγδαίες αλλαγές στο πρότυπο προσπέλασης

Ύπαρξη Ανοδικού Καναλιού



Μοντέλο Κατ' Απαίτηση Εκπομπής



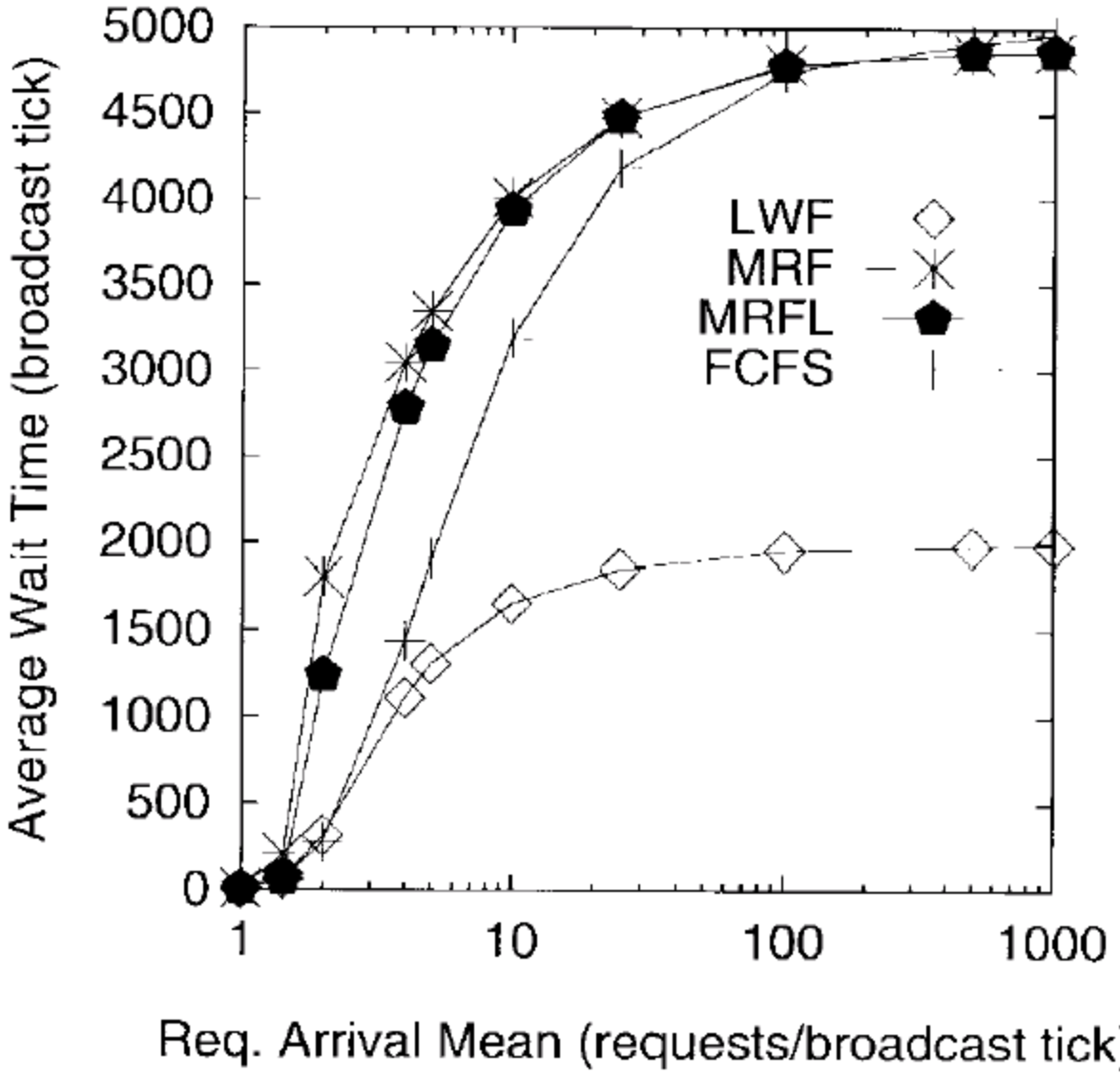
Απαιτήσεις

- **Γρήγορη απόκριση** (responsiveness)
  - Μικρή καθυστέρηση (latency) των χρηστών
    - Όχι **starvation**
    - Κόστος επιλογής του αντικειμένου προς εκπομπή
- **Ικανότητα κλιμάκωσης** (scalability)
  - Ρυθμό άφιξης των αιτήσεων
  - Μέγεθος βάσης δεδομένων
  - Ρυθμό εκπομπής
- **Ευρωστία**
  - Αλλαγές στο πρότυπο προσπέλασης

Υπάρχοντες αλγόριθμοι

- **FCFS** (First-Come-First-Served)
  - Εκπέμπει τα αντικείμενα με τη σειρά που έρχονται οι αιτήσεις γι' αυτά. Εάν υπάρχει ήδη στην ουρά δεν προστίθεται νέα είσοδος (entry) στην ουρά
- **MRF** (Most Requests First)
  - Εκπέμπει το αντικείμενο με τις περισσότερες εκκρεμείς αιτήσεις
- **MRFL** (Most Requests First Lowest)
  - Όπως και ο MRF, αλλά σπάει τις ισοπαλίες προς χάριν του αντικειμένου που δεν έχει εκπεμφθεί για το μεγαλύτερο διάστημα
- **LWF** (Longest Wait First)
  - Εκπέμπει το αντικείμενο για το οποίο ο συνολικός χρόνος αναμονής όλων των αιτήσεων γι' αυτό είναι ο μεγαλύτερος (δηλ. το άθροισμα του χρόνου αναμονής στην ουρά όλων των αιτήσεων γι'αυτό

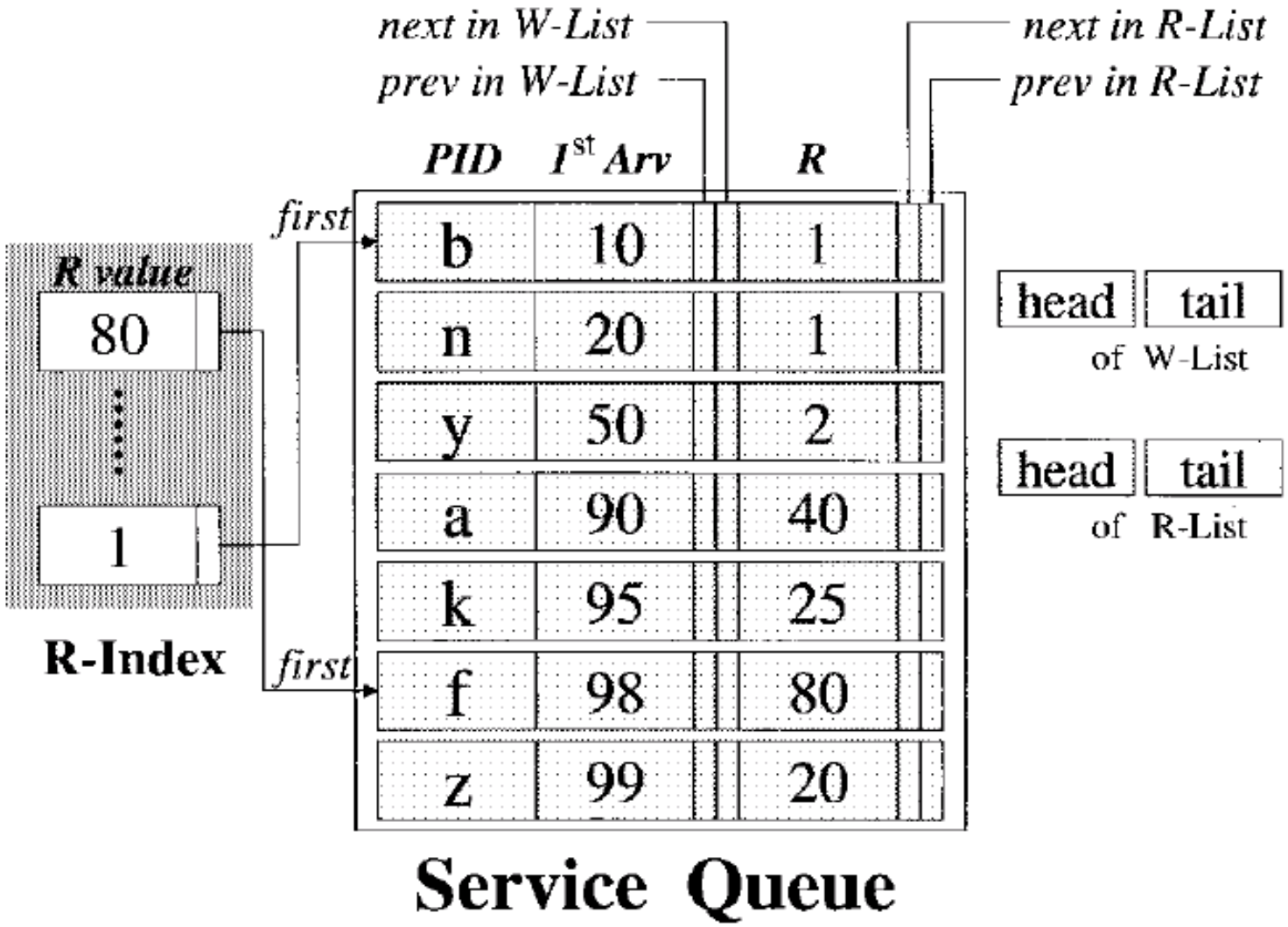
Επίδοση υπαρχόντων αλγορίθμων



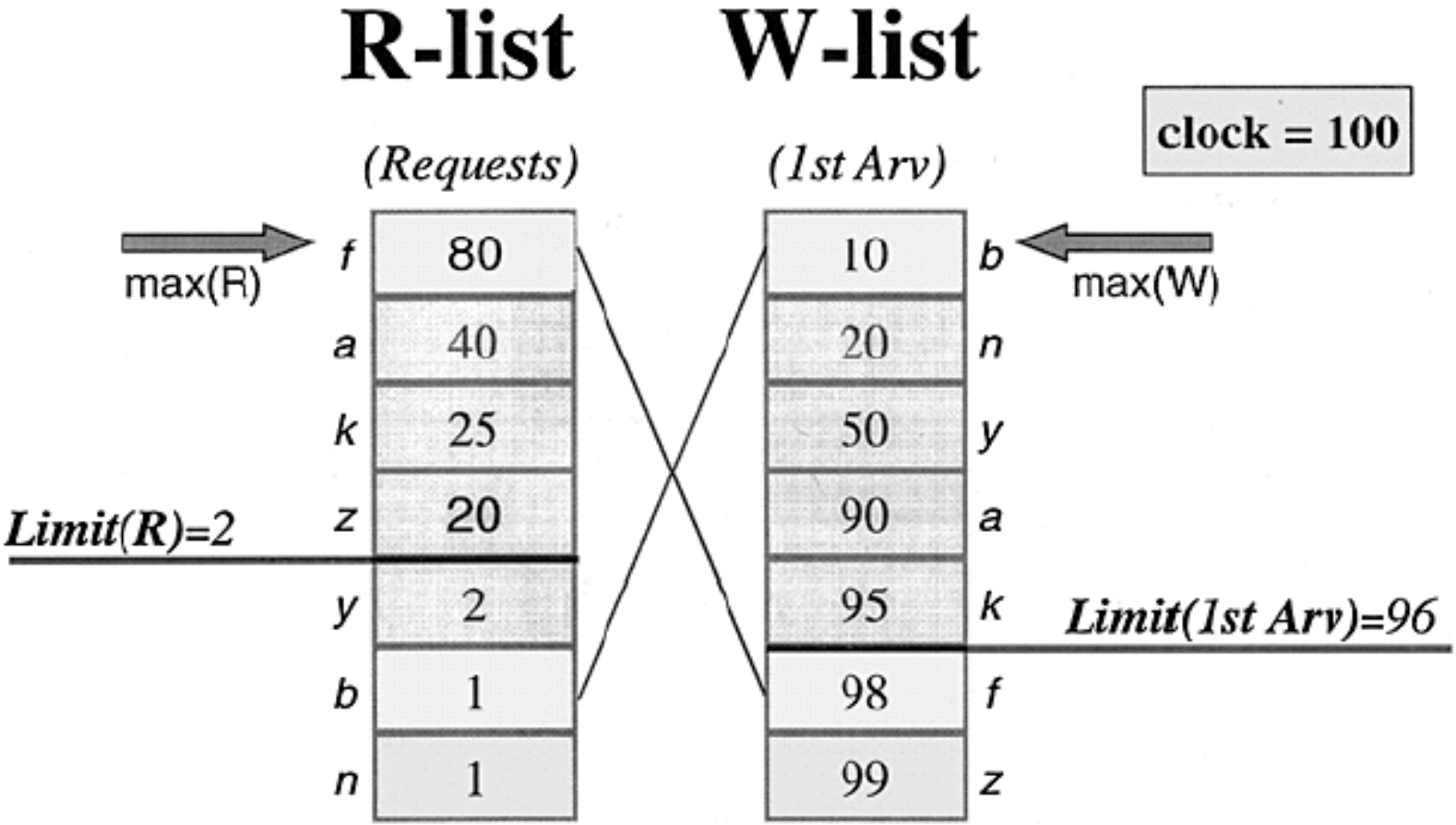
RxW: Συνδυασμός MRF & FCFS

- Εκπέμπει τη σελίδα με το μεγαλύτερο γινόμενο RxW
  - R: είναι ο αριθμός των εκκρεμών αιτήσεων για τη σελίδα
  - W: είναι ο μέγιστος χρόνος αναμονής μέσα στην ουρά, δηλ. η αίτηση πιο παλιά στο χρόνο
- Εκπέμπει μια σελίδα είτε επειδή είναι δημοφιλής είτε επειδή δεν έχει εκπεμφθεί για αρκετό χρονικό διάστημα

Η ουρά εξυπηρέτησης του RxW



Βελτίωση στην αναζήτηση του RxW



Βελτίωση στην αναζήτηση του RxW

- Εξέτασε τη σελίδα στην κορυφή της R-list
- Θέσε MAX = R x W αυτής της σελίδας
- Οι υπόλοιπες τιμές W μπορούν να περιοριστούν αξιοποιώντας την τιμή R', δηλ., την R τιμή της επόμενης στη R-list σελίδας.
- Για να έχει μια άλλη σελίδα RxW μεγαλύτερο από το τρέχον MAX, πρέπει το αντίστοιχο W να ικανοποιεί την ανίσωση W > MAX/R'
- Έτσι θέτουμε ένα όριο στην 1stARV ως εξής

  limit(1stARV) = clock – MAX/R'
- Κατόπιν εξετάζουμε τη σελίδα στην κορυφή της W-list και κάνουμε ανάλογες ενέργειες περιορίζοντας το εύρος αναζήτησης στη R-list

Παράδειγμα βελτιωμένου RxW

The R-List and the W-List are shown as two separate lists and the current clock value is 100 ticks.

First, the entry for page *f* (the top of the R-List) is examined resulting in MAX being set to 160 and *limit(1stARV)* being set to 96.

Next, the entry for page *b* (the top of the W-List) is checked. RxW of b is less than MAX (90 versus 160) so MAX is left unchanged, but *limit(R)* is set to 2.

The algorithm then checks page *α*, which has an RxW value of 400, and so MAX is updated to 400, and *limit(1stARV)* is set to 84. The algorithm continues searching until page *y* is examined, at which point the limit on the -List is reached and the algorithm stops. In this example, page *a* has the highest value, so it is chosen to be broadcast.

Περιεχόμενα

- Αρχιτεκτονική κινητού δικτύου
- Ασύμμετρο περιβάλλον επικοινωνίας χωρίς Ανοδικό Κανάλι
- Δίσκοι Εκπομπής (Broadcast Disks)
- Αλγόριθμοι για Καθαρή Εκπομπή (Pure Broadcast)
- Ασύμμετρο περιβάλλον επικοινωνίας με Ανοδικό Κανάλι
- Αλγόριθμοι για Υβριδική Εκπομπή (Hybrid Broadcast)
- Αλγόριθμοι για Κατ' Απαίτηση Εκπομπή (On-Demand Broadcast)
- **Εκπομπή σε πολλαπλά κανάλια**

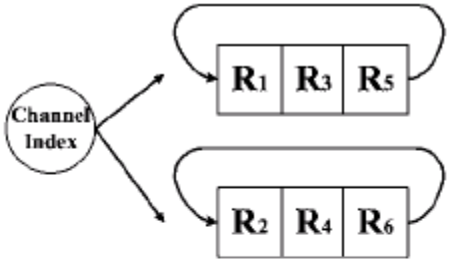
Πολλαπλά κανάλια εκπομπής

Για λόγους όπως:

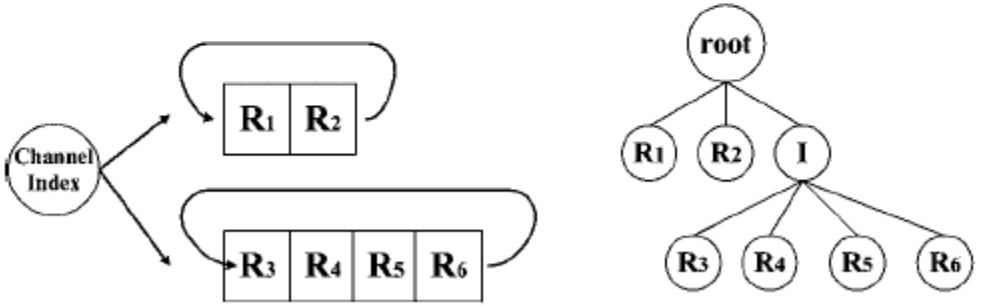
- Application scalability
  - Μια εφαρμογή αποκτά επιπλέον κανάλια για να εξυπηρετήσει μεγαλύτερο πληθυσμό
- Fault tolerance
  - Τρεις servers εκπέμπουν σε μια γεωγραφική περιοχή σε μη συνεχόμενες συχνότητες, αλλά οι δυο παθαίνουν βλάβη και τα κανάλια τους ανατίθονται στον τρίτο
- Reconfiguration of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

Είναι δυνατόν να υπάρχουν πολλαπλά κανάλια εκπομπής

Ιεραρχικά προγράμματα εκπομπής



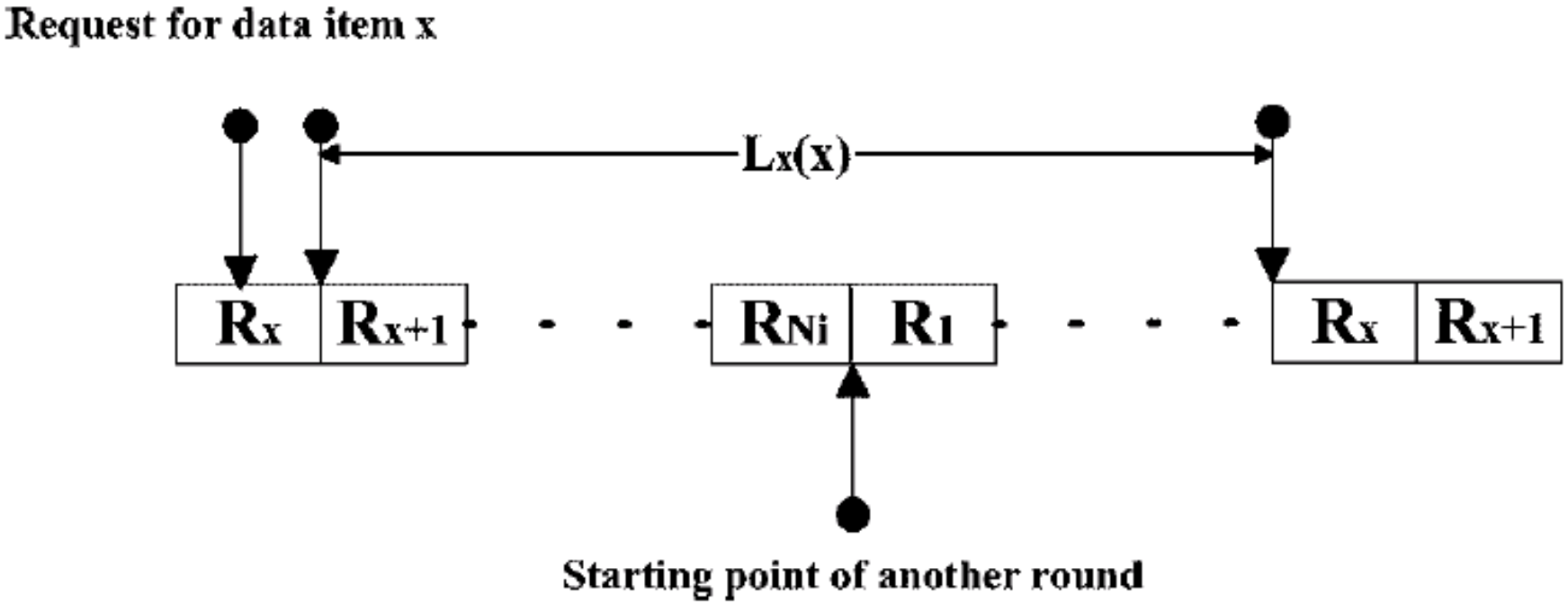
(a) A flat broadcast program



(b) A hierarchical broadcast program with its tree representation

	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

Μέση καθυστέρηση σε ένα κανάλι



Μέση καθυστέρηση για κάθε αντικείμενο στο κανάλι i είναι:

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

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, $t_{i-1} + 1 \leqslant j \leqslant t_i$, 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

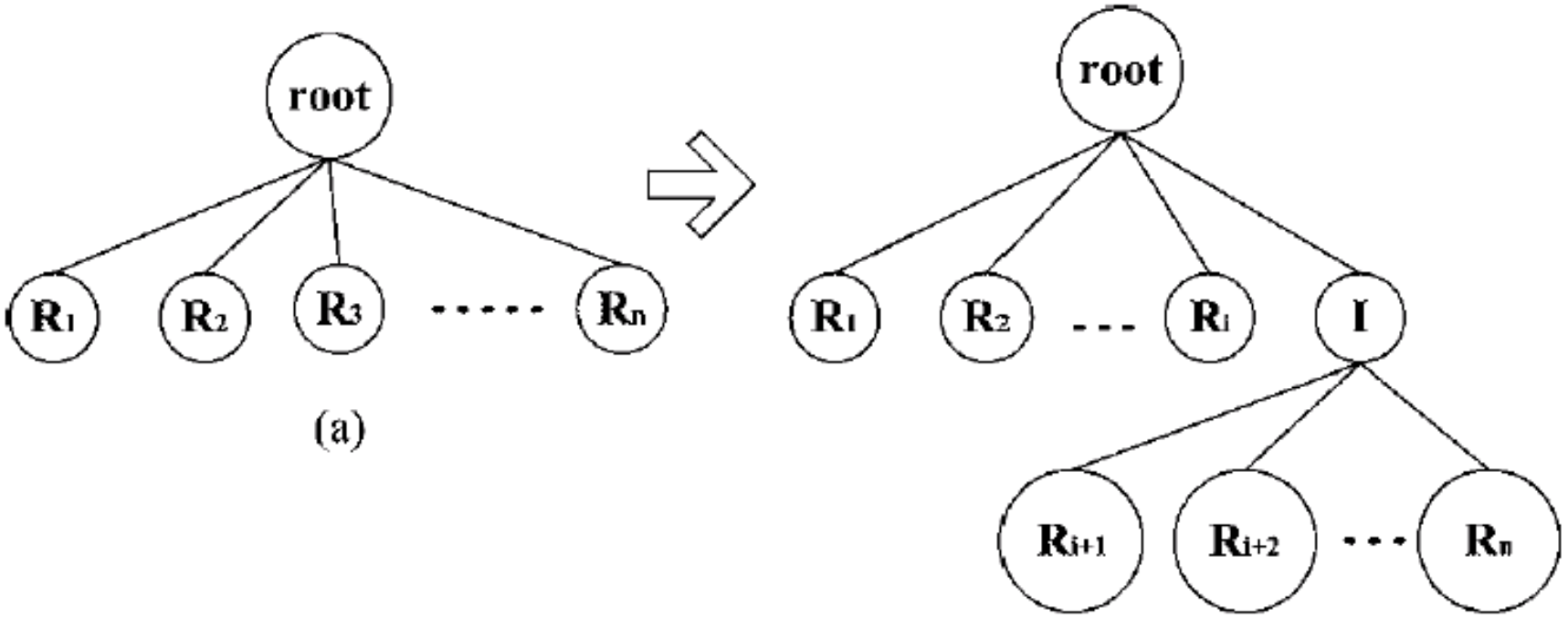
$$\sum_{j=1}^{n} d_{R_j} \cdot Pr(R_j)$$

$$= \sum_{i=1}^{K} d_i \sum_{j=t_{i-1}+1}^{t_i} Pr(R_j)$$

$$= \sum_{i=1}^{K} \left(\sum_{q=1}^{N_i} \frac{N_i - q}{N_i} \right) \sum_{j=t_{i-1}+1}^{t_i} Pr(R_j),$$

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

Δενδρική αναπαράσταση



Κόστος επιπέδου του δένδρου

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

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

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

Αλγόριθμος VF^K

Input. Assume that $R_1, \ldots, R_n$ have been sorted according to the descending order of $P_r(R_j)$, $1 \leqslant j \leqslant n$, i.e., $P_r(R_q) \geqslant P_r(R_y)$ iff $q \leqslant y$. 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 $i \geqslant 2$
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. $pivot = 1$;

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 or i == pivot) /* Upward and downward partition */
16.         begin
17.             j = Partition(R_AT(i).B, R_AT(i).B+1, ..., R_AT(i).E);
18.             {Update table AT accordingly and unmark all rows;
19.             pivot++;}
20.         end
21.     else /* Middle partition */
22.         begin
23.             j = Partition(R_AT(i).B, R_AT(i).B+1, ..., 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, ..., R_j) and
                    (R_j+1, R_j+2, ..., R_AT(i).E) together;}
30.         end
31.     end
32. until (pivot == K)
end
```

Αλγόριθμος VF^K

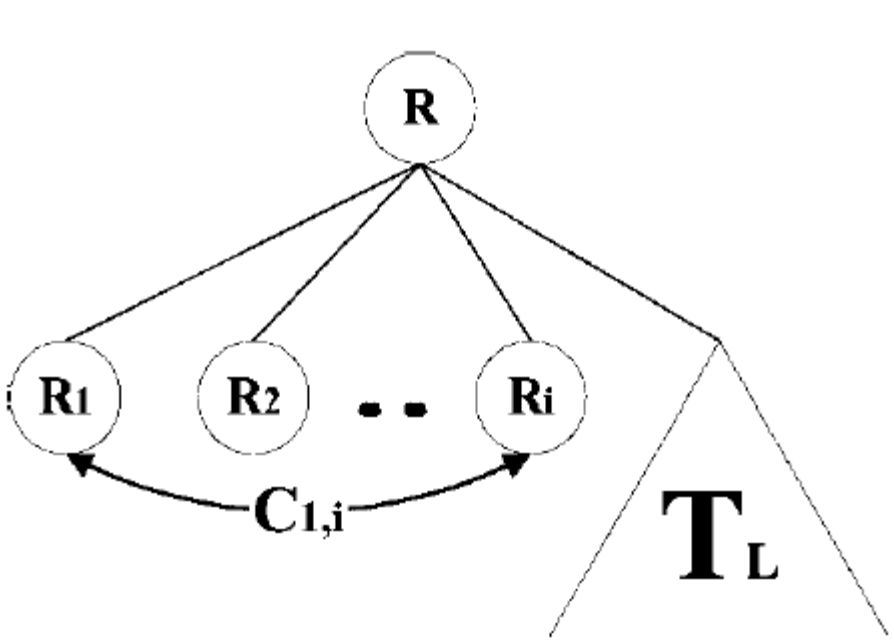
Procedure *Partition*$(R_i, R_{i+1}, \ldots, R_j)$.

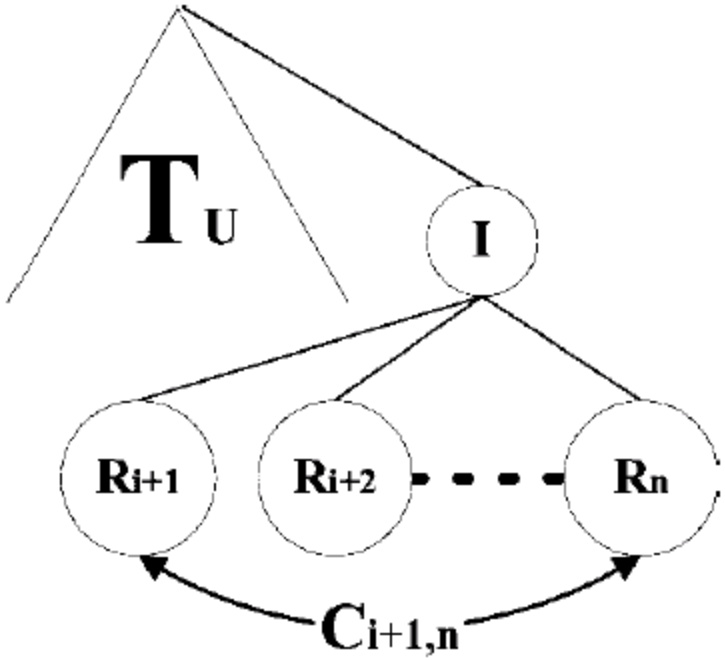
1. Determine p^* such that

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

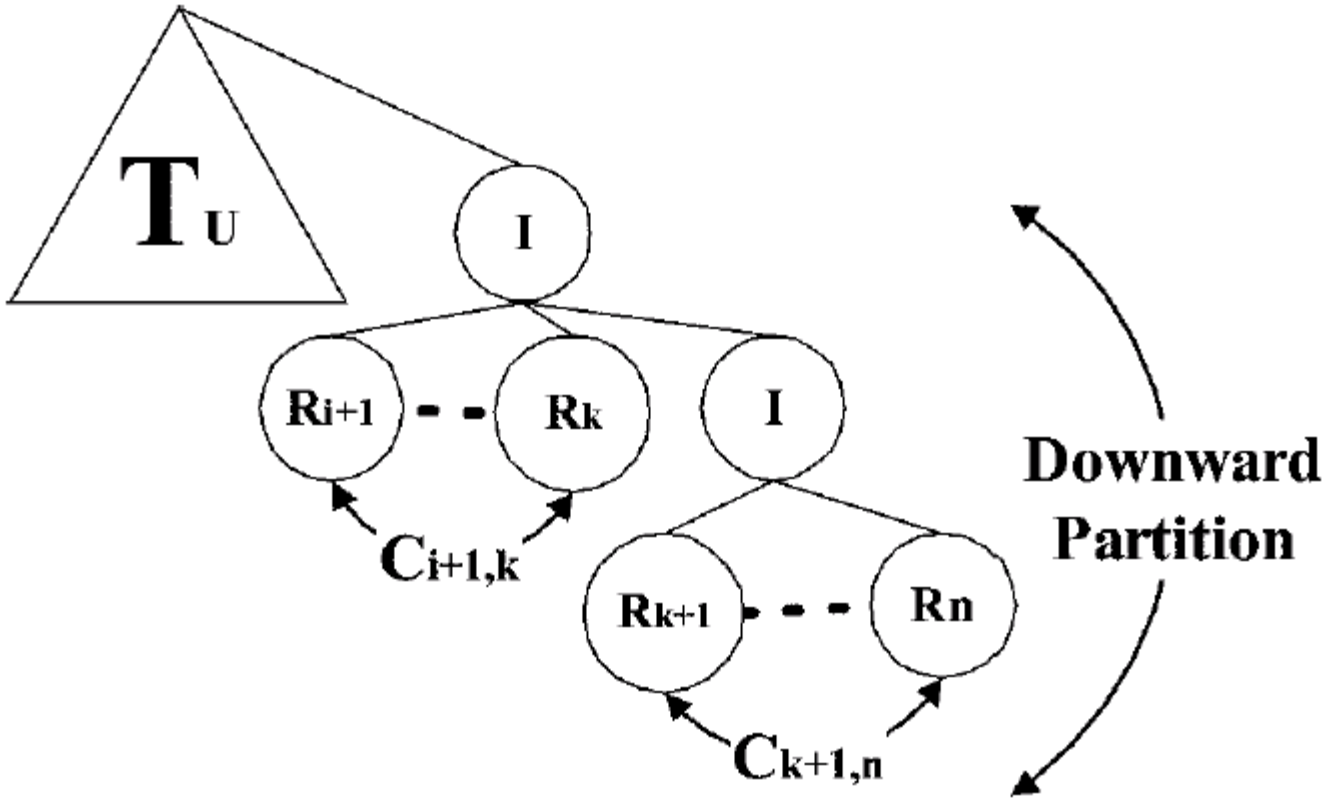
2. Attach nodes $R_{p^*+1}, R_{p^*+2}, \ldots, R_j$ under a new node I in the tree.
3. Return p^*.

Αλγόριθμος VF^K

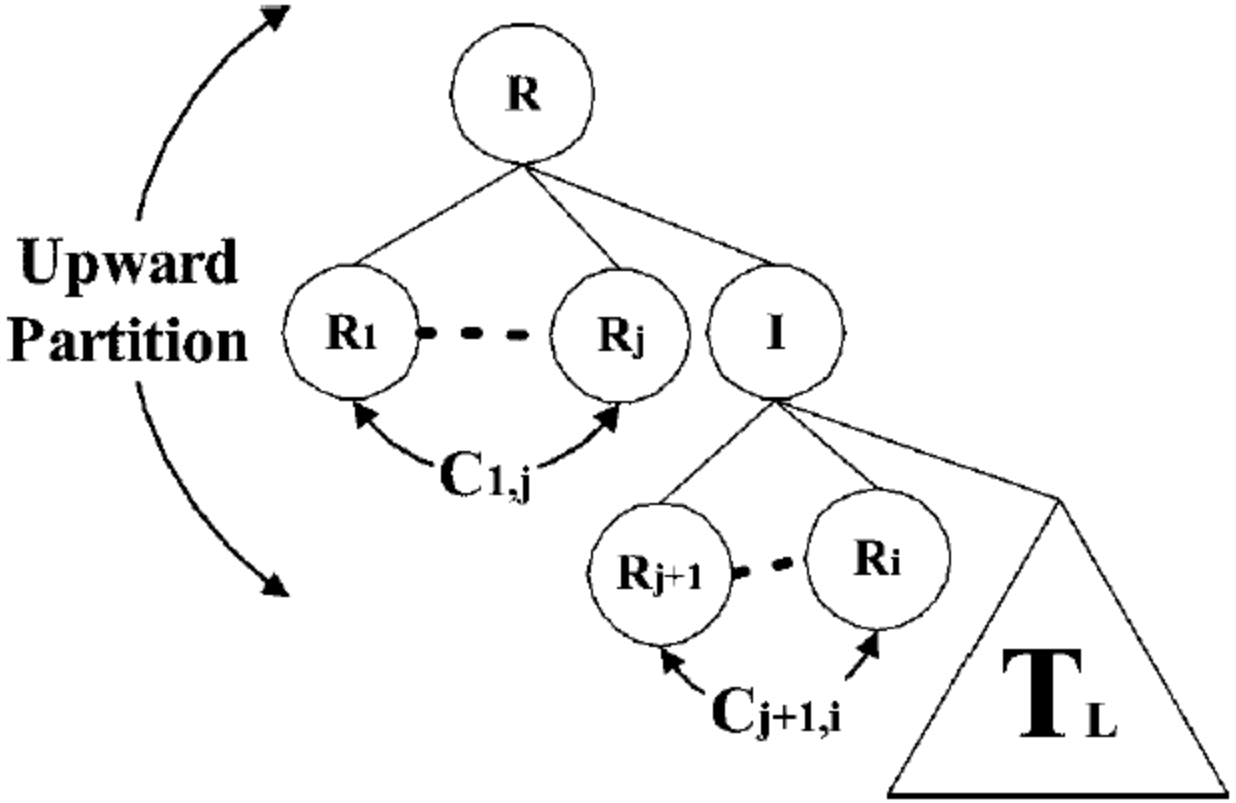




Αλγόριθμος VF^K



Αλγόριθμος VF^K



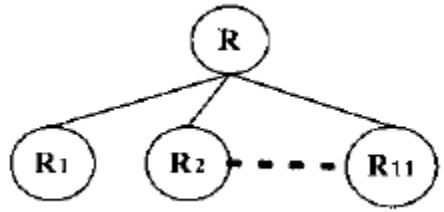
Παράδειγμα του VF^K

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

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

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

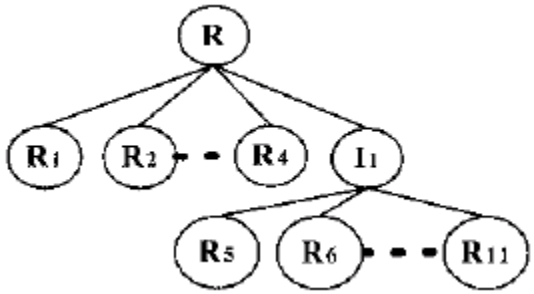


Παράδειγμα του VF^K

(b) Partition $(R_1, R_2, \ldots, R_4)$ 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*

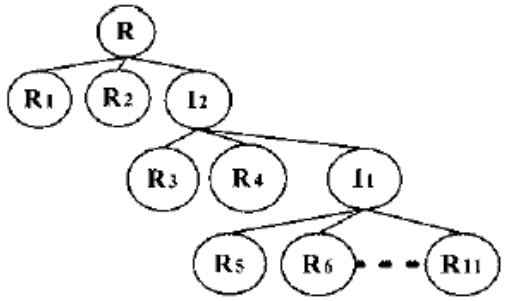


Παράδειγμα του VF^K

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

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*



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