Πανεπιστήμιο Κρήτης, Τμήμα Επιστήμης Υπολογιστών
Άνοιξη 2008

HY463 - Συστήματα Ανάκτησης Πληροφοριών
Information Retrieval (IR) Systems

Parallel and Distributed IR
Παράλληλη και Κατανεμημένη ΑΠ

Γιάννης Τζίτζικας

Διάρθρωση Περιεχομένου

Μέρος Α: Παράλληλη Ανάκτηση Πληροφοριών (Parallel IR)

Μέρος Β: Κατανεμημένη Ανάκτηση Πληροφοριών (Distributed IR)
- Επιλογή Πηγής
- Ενοποίηση Αποτελεσμάτων

Μέρος Γ: Ανάκτηση Πληροφοριών σε Ομότιμα Συστήματα (Peer-to-Peer Systems)

Μέρος Β

Κατανεμημένη Ανάκτηση Πληροφοριών
(Distributed Information Retrieval)

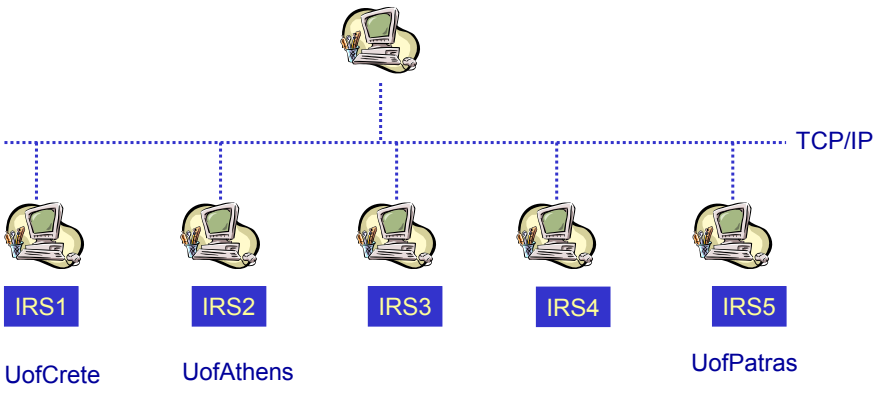
Κατανεμημένη Ανάκτηση Πληροφοριών: Διάρθρωση

- Κίνητρο
- Σχέση μεταξύ Παράλληλης και Κατανεμημένης Αν. Πληροφοριών
- Σχεδιαστικά Ζητήματα
- Διαμέριση Συλλογών και Εγγράφων
- Επιλογή Πηγής
- Ενοποίηση Αποτελεσμάτων

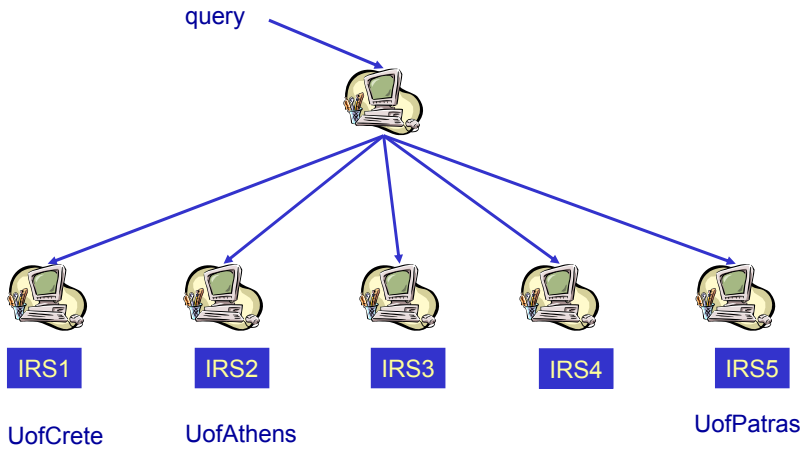
Κατανεμημένη ΑΠ: Κίνητρο

- Το κίνητρο για Παράλληλη ΑΠ ήταν η **βελτίωση της απόδοσης**
- Για την Κατανεμημένη ΑΠ δεν είναι μόνο αυτό.
  - Είναι και η ανάγκη **ενοποιημένης πρόσβασης** στα έγγραφα πολλών συστημάτων ανάκτησης πληροφοριών

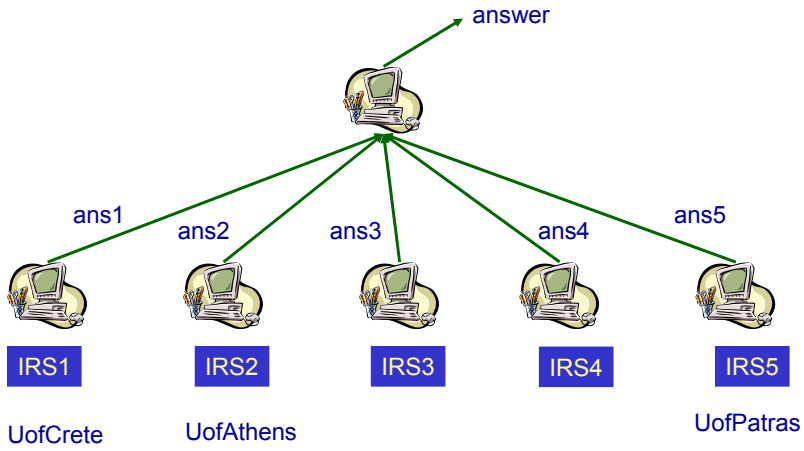
Κατανεμημένη Ανάκτηση Πληροφοριών



Κατανεμημένη Ανάκτηση Πληροφοριών

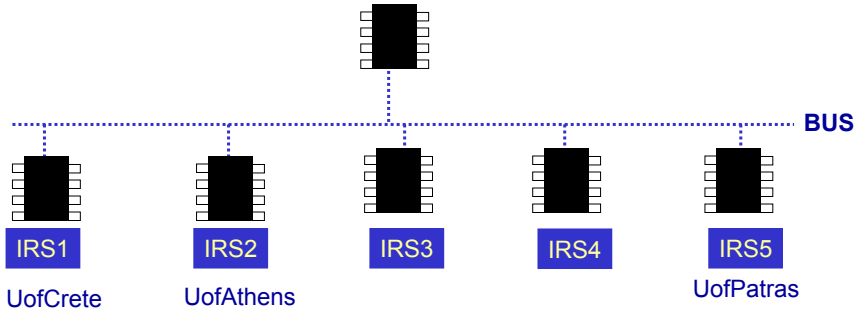


Κατανεμημένη Ανάκτηση Πληροφοριών



Ποια η Σχέση μεταξύ Παράλληλης και Κατανεμημένης Ανάκτησης Πληροφοριών;

- Η κατανεμημένη μοιάζει με την παράλληλη αρχιτεκτονική MIMD
- Διαφορές με την MIMD
  - το **κανάλι επικοινωνίας** μεταξύ των επεξεργαστών είναι πολύ πιο αργό
  - δεν έχουμε τους ίδιους επεξεργαστές (όπως σε μια παράλληλη μηχανή)
  - στην κατανεμημένη ο μεσίτης (broker) συχνά επιλέγει να χρησιμοποιήσει μόνο ένα υποσύνολο των υποκείμενων συστημάτων

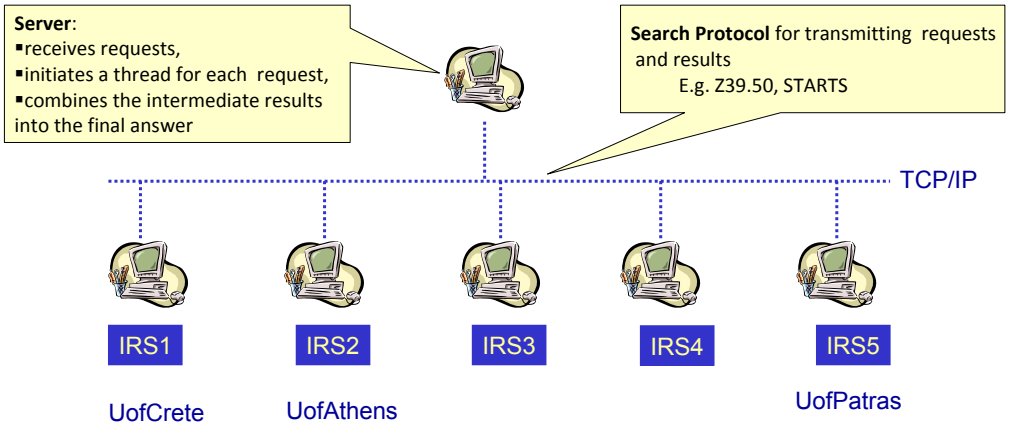


Σχέση μεταξύ Παράλληλης και Κατανεμημένης Αν. Πλ.

Ποια προσέγγιση του **Partitioned Parallel Processing** είναι κατάλληλη για την Κατανεμημένη Ανάκτηση;

- **document partitioning:** ενδείκνυται για την κατανεμημένη ανάκτηση
- **term partitioning**: δεν είναι πολύ καλή διότι απαιτεί περισσότερη επικοινωνία (για αυτό σπάνια υιοθετείται από ένα κατανεμημένο σύστημα)

Κατανεμημένη Ανάκτηση: Σχεδιαστικά ζητήματα



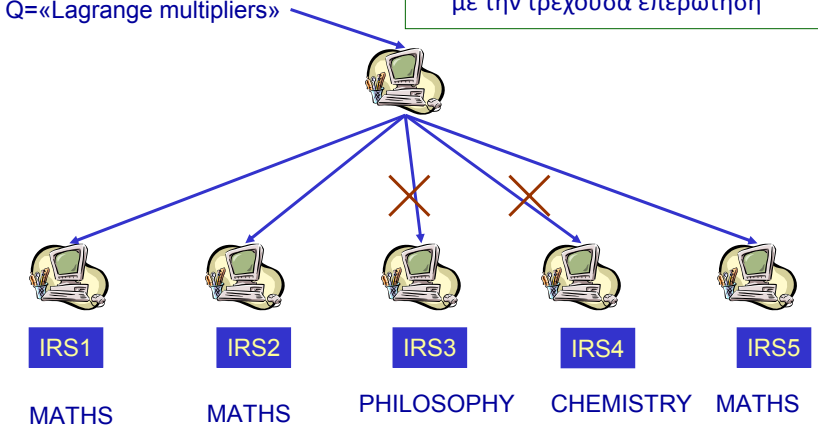
- **distribute** documents across servers
- **selection** of the servers to receive a particular request
- **combine** the results of multiple servers

Διαμερισμός Συλλογών (Collection Partitioning)

- Δεν τίθεται τέτοιο ζήτημα αν τα υποκείμενα συστήματα είναι ετερογενή
- Σενάρια για την περίπτωση που υπάρχει κεντρικός έλεγχος:
  - **Semantic**-based partition of collections to servers
    - Search Servers focusing on a particular subject area
      - E.g. Maths, Physics, etc
  - **Semantic**-based partition of documents to servers
    - π.χ. με χρήση ενός αλγορίθμου ομαδοποίησης (clustering)
  - **Replications** of collections to all servers
    - για βελτίωση throughput (μοιάζει με το multitasking)
    - όταν οι συλλογές δεν είναι μεγάλες
  - **Τυχαία διανομή** εγγράφων στους servers
    - για βελτίωση της απόδοσης στην περίπτωση που η συλλογή είναι πολύ μεγάλη

Επιλογή Πηγής (Source Selection)

Source Selection: Επιλογή των συλλογών που είναι πιθανόν να έχουν συναφή έγγραφα με την τρέχουσα επερώτηση



Για ποιο λόγο να κάνουμε Επιλογή Πηγής;

Η αναζήτηση σε κάθε συλλογή μπορεί:

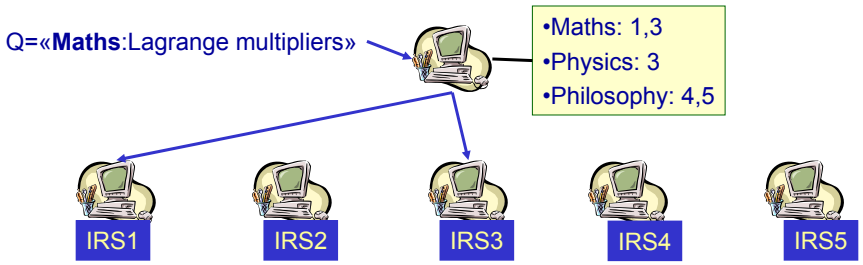
- να είναι **ακριβή σε χρόνο** (αφού μπορεί να έχουμε εκατοντάδες συλλογές)
- να είναι **ακριβή σε χρήμα** (η αναζήτηση μπορεί να έχει χρηματικό κόστος)
- να καθορίσει την **αποτελεσματικότητα** (effectiveness) της ανάκτησης

Επιλογή Πηγής: Επιλογή **Όλων**

- Κατάλληλη κυρίως για διαμερισμό μιας μεγάλης συλλογής πάνω από ένα **τοπικό δίκτυο**
- Εύκολη ενοποίηση αποτελεσμάτων για το Boolean model
  - answer(q) = ans1(q) ∪ … ∪ ansk(q)
- Η ενοποίηση αποτελεσμάτων για τα στατιστικά μοντέλα είναι πιο δύσκολη (*το ζήτημα αυτό θα μελετηθεί παρακάτω*)

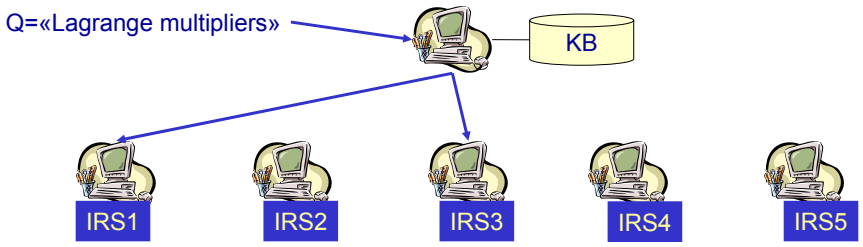
Επιλογή Πηγής: **Χειρονακτική Ομαδοποίηση** και Επιλογή

- Θεματική οργάνωση συλλογών (χειρονακτικώς)
  - πχ ***μαθηματικά, φυσική, ειδήσεις***, κλπ
  - προβλήματα
    - χρονοβόρα διαδικασία, ευάλωτη σε ασυνέπειες/παραλείψεις, δεν θα δουλέψει καλά για μη-συνηθισμένες επερωτήσεις
- Ο χρήστης επιλέγει τη θεματική κατηγορία



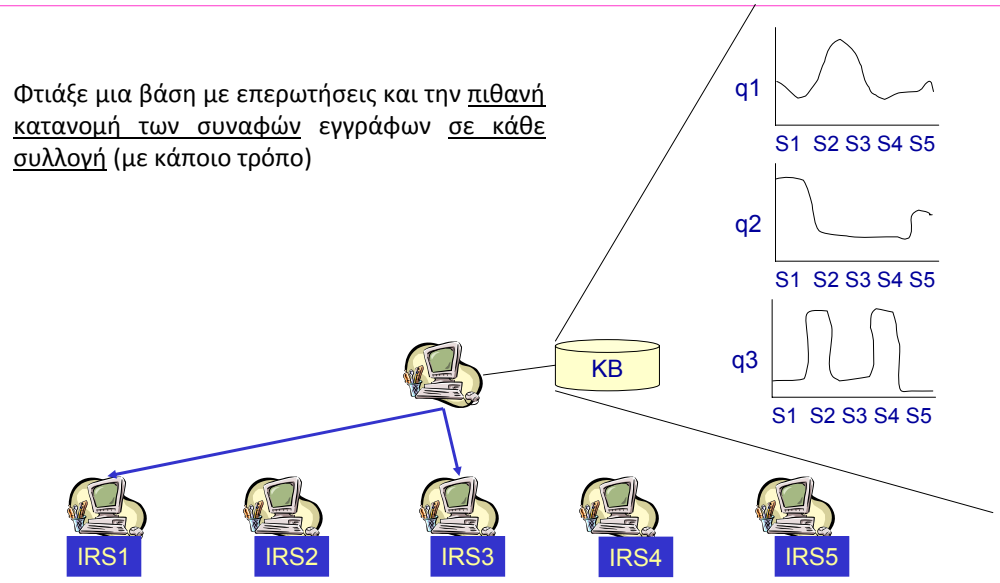
Επιλογή Πηγής βάσει **Κανόνων** (Rule-based)

- Τα περιεχόμενα κάθε συλλογής περιγράφονται σε μια **Βάση Γνώσης**
- Ένα Σύστημα Κανόνων επιλέγει τις πηγές για κάθε εισερχόμενη επερώτηση
- Αδυναμίες
  - κόστος συγγραφής κανόνων
  - ανάγκη συντήρησης των κανόνων (αν οι συλλογές είναι δυναμικές)



Επιλογή Πηγής: **Κατανομή Συναφών** Εγγράφων (Relevant Document Distribution (RDD))

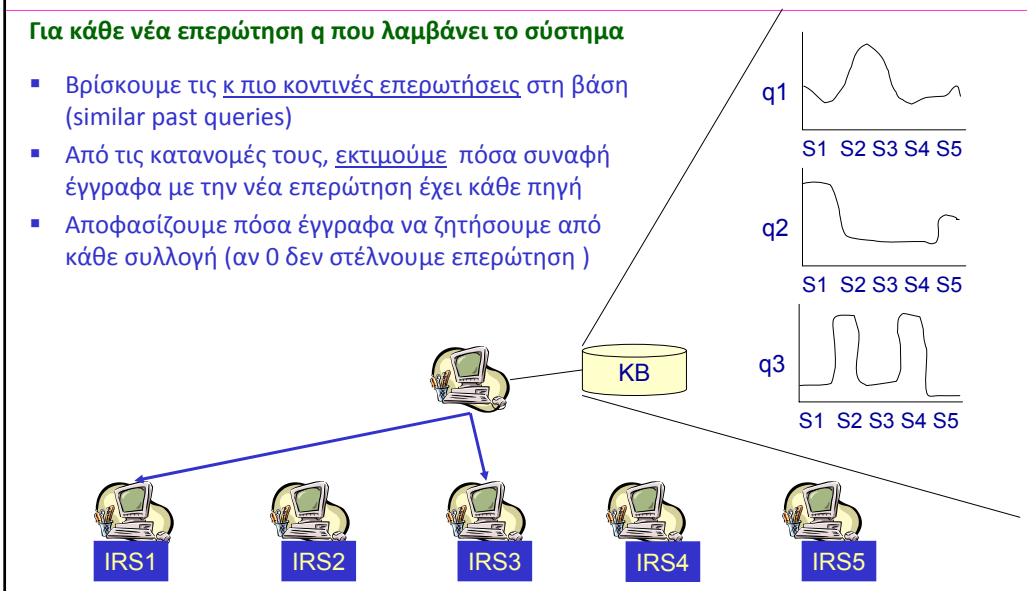
Φτιάξε μια βάση με επερωτήσεις και την πιθανή κατανομή των συναφών εγγράφων σε κάθε συλλογή (με κάποιο τρόπο)



Επιλογή Πηγής: **Κατανομή Συναφών** Εγγράφων (Relevant Document Distribution (RDD))

Για κάθε νέα επερώτηση q που λαμβάνει το σύστημα

- Βρίσκουμε τις κ πιο κοντινές επερωτήσεις στη βάση (similar past queries)
- Από τις κατανομές τους, εκτιμούμε πόσα συναφή έγγραφα με την νέα επερώτηση έχει κάθε πηγή
- Αποφασίζουμε πόσα έγγραφα να ζητήσουμε από κάθε συλλογή (αν 0 δεν στέλνουμε επερώτηση)



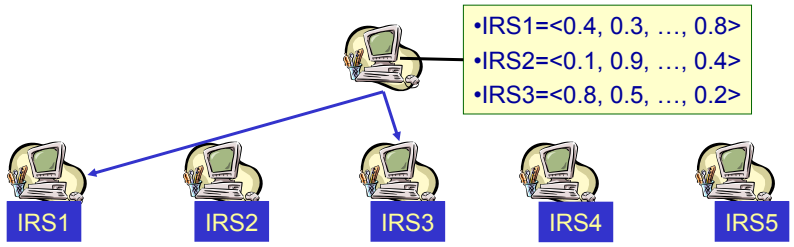
Επιλογή Πηγής: **Επερώτηση Βολιδοσκόπησης** (Query Probing)

- Στέλνουμε μια **επερώτηση βολιδοσκόπησης (probing query)** σε κάθε συλλογή (που μπορεί να περιλαμβάνει μερικούς από τους όρους της επερώτησης)
  - Κάθε συλλογή απαντά με στατιστικές πληροφορίες
    - πχ: μέγεθος συλλογής, πόσα έγγραφα έχουν τον κάθε όρο, πόσα έγγραφα έχουν όλους τους όρους της επερώτησης, κλπ
  - Βάσει αυτών των στοιχείων επιλέγουμε την πηγή

Υποθέσεις

- η επεξεργασία των επερωτήσεων βολιδοσκόπησης είναι πολύ φθηνότερη
  - περιέχουν λίγους όρους, δεν χρειάζεται να υπολογίσουμε βαθμούς συνάφειας ή να διατάξουμε τα έγγραφα ως προς τη συνάφεια τους

Επιλογή Πηγής με **Διανύσματα Πηγών**



- Βλέπουμε **κάθε συλλογή** ως ένα **μεγάλο έγγραφο**
- Φτιάχνουμε ένα **διάνυσμα για κάθε συλλογή** (τύπου TF-IDF)
  - tfij: συνολικές εμφανίσεις του όρου i στη συλλογή j
  - idfi: log(N/ni), όπου N το πλήθος των συλλογών, και ni το πλήθος των συλλογών που έχουν τον όρο i
- Υπολογίζουμε το **βαθμό ομοιότητας κάθε νέας επερώτησης με το διάνυσμα κάθε συλλογής** (π.χ. ομοιότητα συνημίτονου)
- Διατάσσουμε τις συλλογές και επιλέγουμε τις κορυφαίες

Επιλογή Πηγής με **Διανύσματα Πηγών (II)**

- Μια αδυναμία:
  - Μπορεί ο βαθμός ομοιότητας με μία συλλογή να είναι μεγάλος, αλλά να μην υπάρχει κανένα έγγραφο εκεί με μεγάλο βαθμό συνάφειας
- Ένας τρόπος αντιμετώπισης:
  - Για κάθε συλλογή φτιάξε N/B διανύσματα, δηλαδή ένα διάνυσμα για κάθε B έγγραφα της συλλογής
  - Αν B = 1 τότε ο server είναι σαν να έχει το ευρετήριο όλων των συστημάτων
  - Αν B = N τότε έχουμε ένα διάνυσμα για κάθε συλλογή

Επιλογή Πηγής: **GlOSS**
(**Gl**ossary of **S**ervers **S**erver)

Estimate the number of potentially relevant documents in a collection C for a Boolean AND query Q as:

$$|C| \cdot \prod_{t \in Q} \frac{df_t}{|C|}$$

df_t : number of docs in C that contain t

|C| the number of documents in the collection C

Requires that for each collection *C we* have an entry in a centralized index
centralized index is small, easy to maintain

Επιλογή Πηγής: **gGlOSS** και **hGlOSS**

- **gGlOSS**
  - Extends the GlOSS approach to the vector space model
    - Each collection is represented by its centroid vector
    - Standard inner product similarity measure of query to each collection
    - Rank collections accordingly
- **hGlOSS** (hierarchical GlOSS)
  - Extends the gGlOSS approach to sets of gGlOSS indexes
  - Each gGlOSS index is represented by its centroid vector

Επιλογή Πηγής: **Σύνοψη**

Προσεγγίσεις για μικρό αριθμό συλλογών

- **Επιλογή Όλων**
- **Χειρονακτική Ομαδοποίηση (και χειρονακτική επιλογή)**
- **Επιλογή βάσει Κανόνων (Rule-based selection)**
- **Κατανομή Συναφών Εγγράφων (Relevant document distribution (RDD))**
- **Βολιδοσκόπηση Επερώτησης (Query Probing)**
- **Διανύσματα Πηγών**

Προσεγγίσεις για μεγάλο αριθμό συλλογών

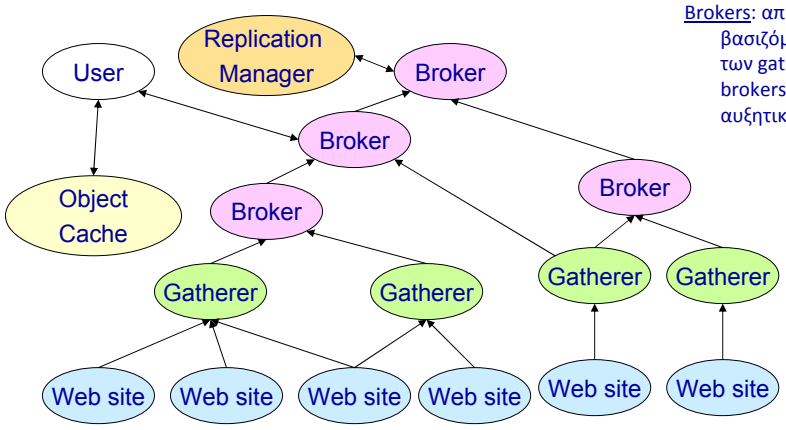
- **Διανύσματα Πηγών**
- **GlOSS**

HARVEST

A distributed architecture to gather and distribute data
- used by CIA, NASA, US National Academy of Sciences

Gatherers: ευρετηριάζουν >=1 Web Server περιοδικά

Brokers: απαντούν επερωτήσεις βασιζόμενοι στα ευρετήρια των gatherers ή άλλων brokers (και ενημερώνουν αυξητικά τα ευρετήρια τους)

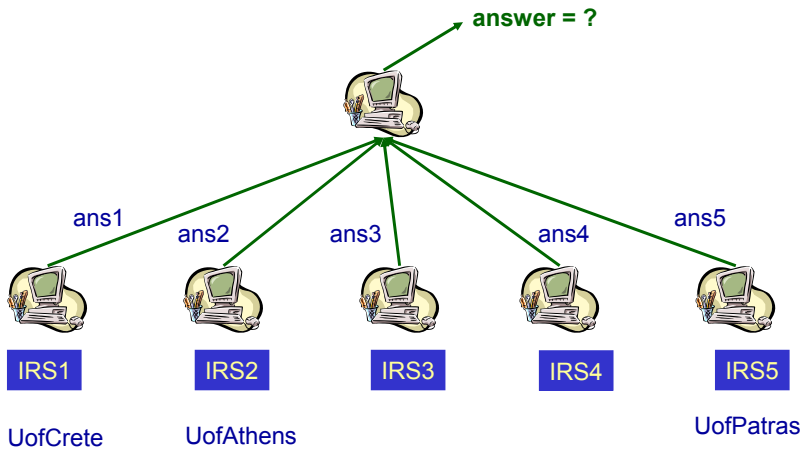


Ενοποίηση Αποτελεσμάτων
(… Results Merging, Fusion, Rank Aggregation, …)

Ενοποίηση Αποτελεσμάτων
Διάρθρωση

- Κατηγορίες Τεχνικών Ενοποίησης: Απομονωμένες και Ολοκληρωμένες
- Τεχνικές Ενοποίησης
  - Round Robin interleaving
  - Score-based
  - Weighted Score-based
  - Global-statistics
- Μετα-Μηχανές Αναζήτησης
- Ενοποίηση Διατάξεων (Rank-Aggregation)
  - Επιθυμητές Ιδιότητες
  - Ενοποίηση κατά Borda
  - Ενοποίηση κατά Condorcet
  - Το Θεώρημα του Ανέφικτου του K. Arrow (Arrow's Impossibility theorem)
  - Ενοποίηση κατά Kemeny
  - Αποδοτικοί αλγόριθμοι υπολογισμού των κορυφαίων κ στοιχείων της ενοποιημένης διάταξης (Top-K Rank Aggregation)

Ενοποίηση Αποτελεσμάτων



Περιπτώσεις

- Ενοποίηση **Συνόλων** (π.χ. απαντήσεων σε Exact Match Queries)
  - answer(q) = ans1(q) ∪ … ∪ ansk(q)
  - Άρα η ενοποίηση αποτελεσμάτων για το Boolean model είναι εύκολη
- Ενοποίηση **Διατάξεων** (απαντήσεων Partial Match Queries)
  - Η ενοποίηση αποτελεσμάτων είναι πιο δύσκολη
    - οι διατάξεις/σκορ **δεν είναι πάντα συγκρίσιμες**, αφού εξαρτώνται από τα στατιστικά της συλλογής του κάθε συστήματος (e.g. idf)
    - υπάρχουν πολλοί διαφορετικοί τρόποι συνάθροισης διατάξεων
  - Συχνά μας αρκεί η εύρεση των κορυφαίων στοιχείων της ενοποιημένης διάταξης

Κατηγορίες Στρατηγικών Ενοποίησης Διατάξεων

(A) **Ολοκληρωμένες** Τεχνικές (**Integrated**)

- Οι πηγές παρέχουν **επιπρόσθετη πληροφορία** που χρησιμοποιείται κατά την ενοποίηση
- Αδυναμίες:
  - Στενό πεδίο εφαρμογής - απαιτούν συμφωνία μεταξύ των πηγών (e.g. protocol)
  - Συχνά λαμβάνουν υπόψη τους μέτρα όπως Precision/Recall, τα οποία δεν είναι πάντα «αντικειμενικά» ή συγκρίσιμα.

(B) **Απομονωμένες** Μέθοδοι (**Isolated**)

- Δεν απαιτούν **καμία επιπλέον πληροφορία** από τις πηγές (μπορούν να εφαρμοστούν και στις μετα-μηχανές αναζήτησης)
- Είναι ανεξάρτητες των τεχνικών ευρετηρίασης και των μοντέλων ανάκτησης των υποκείμενων συστημάτων
- Άρα κατάλληλες για δυναμικά περιβάλλοντα όπου υπάρχουν πολλά συστήματα των οποίων η λειτουργία εξελίσσεται συχνά και απρόβλεπτα
- Σχετικές τεχνικές: round robin interleaving, score-based, Borda, Condorcet, download and re-index the contents of the objects (web pages)

Ενοποίηση Διατάξεων: **Round Robin interleaving** (isolated)

(δηλαδή merge sort)

Παράδειγμα:

- ans1(q) = <d10,d2, d30, d7>
- ans2(q) = <d4, d12, d5, d9>
- ANS(q) = < {d10,d4}, {d2,d12}, {d30,d5}, {d7,d9}>

Προβλήματα

- στην πραγματικότητα όλα τα έγγραφα του ans1(q) μπορεί να είναι καλύτερα (πιο συναφή) από το 1ο στοιχείο της ans2(q)

Ενοποίηση Διατάξεων: **Score-based** (isolated)

Παράδειγμα:

- ans1(q) = < (d3,0.8), (d2,0.7) >
- ans2(q) = < (d5,0.6), (d6,0.3) >
- ans3(q) = < (d4,0.9) >

- ANS(q) = < d4, d3, d2, d5, d6>

Προβλήματα

- τα σκορ διαφορετικών συστημάτων *δεν είναι συγκρίσιμα* (κανονικοποιημένα), αφού εξαρτώνται από τα στατιστικά της συλλογής του κάθε συστήματος (e.g. idf).

Ενοποίηση Διατάξεων: **Weighted Score-based**

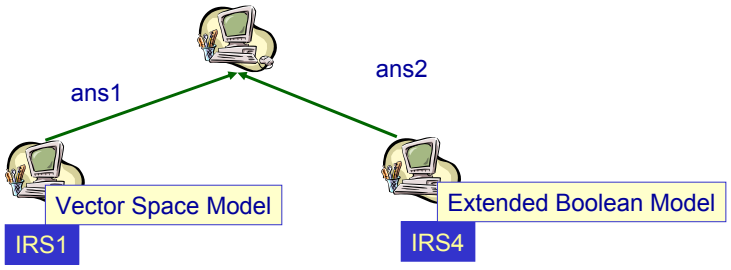
- Λαμβάνουμε υπόψη το σκορ της πηγής που υπολογίσαμε όταν κάναμε *Επιλογή Πηγής (source selection)*

Πχ

- Score(IRS1) = 0.9 // υπολογίστηκε στη φάση επιλογής πηγής
- Score(IRS2) = 0.5 // υπολογίστηκε στη φάση επιλογής πηγής
- ans1(q) = <(d1, 0.7)>
- ans2(q) = <(d2, 0.9)>
- ANS(q) = < (d1, 0.63), (d2, 0.45)> // 0.63 = 0.9*0.7

- Εδώ πολλαπλασιάσαμε το σκορ της πηγής με το σκορ των εγγράφων.
- Υπάρχουν και άλλες παραλλαγές (π.χ. [Callan94,95])

Ενοποίηση Διατάξεων: **Download and re-index/re-score** (isolated)

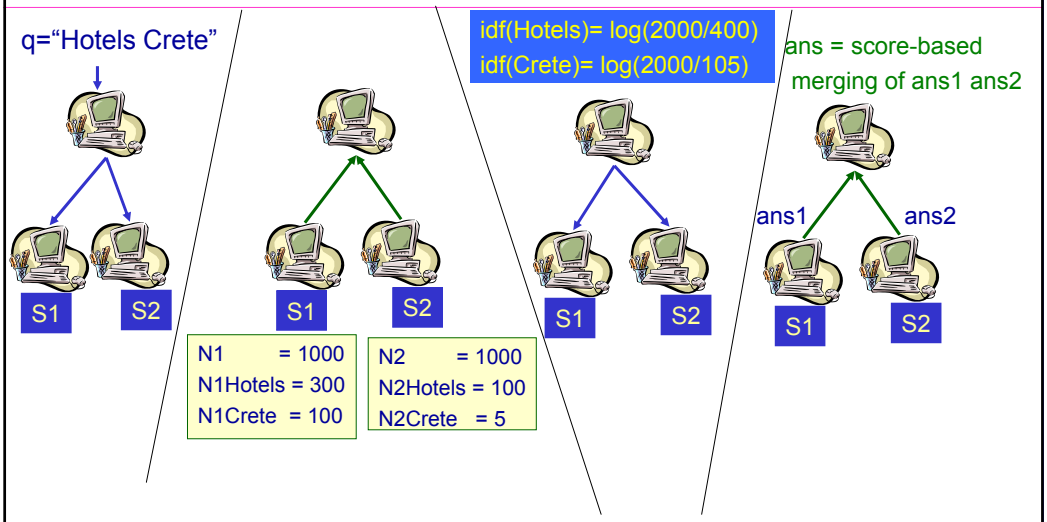


- Εδώ ανακτούμε τα έγγραφα των απαντήσεων κάθε πηγής, τα επαναευρετηριάζουμε και επαναυπολογίζουμε το βαθμό συνάφειας τους
- Μειονέκτημα
  - Χρονοβόρα διαδικασία

Ενοποίηση Διατάξεων: **Global term statistics (integrated)**

- Μπορούμε να κάνουμε συγκρίσιμα τα σκορ διαφορετικών συστημάτων αν επιβάλουμε τα ίδια στατιστικά στοιχεία σε όλα τα συστήματα (global statistics)
- Τα στατιστικά αυτά στοιχεία μπορούν να αποκτηθούν στη φάση της επιλογής πηγής (πχ Διανύσματα Πηγής, Probe Queries, …)
- Αποτίμηση Επερωτήσεων σε 2 φάσεις
  - στην 1η συλλέγονται τα στατιστικά (ο server στέλνει την επερώτηση και οι πηγές απαντούν με τα στατιστικά των όρων που περιέχονται στην επερώτηση)
  - στην 2η ο server στέλνει σε κάθε πηγή την επερώτηση μαζί με τα καθολικά στατιστικά των όρων της
  - κάθε πηγή αποτιμά την επερώτηση με τα καθολικά στατιστικά και επιστρέφει την απάντηση
  - Ο server λαμβάνει έτοιμα σκορ και απλά τα ενοποιεί (merge sort)

Ενοποίηση Διατάξεων: Global term statistics
Παράδειγμα



Ενοποίηση Αποτελεσμάτων
Διάρθρωση

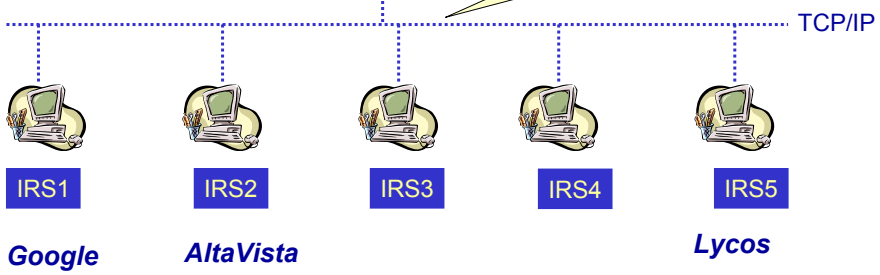
- Κατηγορίες Τεχνικών Ενοποίησης: Απομονωμένες και Ολοκληρωμένες
- Τεχνικές Ενοποίησης
  - Round Robin interleaving
  - Score-based
  - Weighted Score-based
  - Global-statistics
- Μετα-Μηχανές Αναζήτησης
- Ενοποίηση Διατάξεων (Rank-Aggregation)
  - Επιθυμητές Ιδιότητες
  - Ενοποίηση κατά Borda
  - Ενοποίηση κατά Condorcet
  - Το Θεώρημα του Ανέφικτου του K. Arrow (Arrow's Impossibility theorem)
  - Ενοποίηση κατά Kemeny
  - Αποδοτικοί αλγόριθμοι υπολογισμού των κορυφαίων κ στοιχείων της ενοποιημένης διάταξης (Top-K Rank Aggregation)

Μέτα-μηχανές Αναζήτησης

Μετα-Μηχανές Αναζήτησης

Server:
- receives requests,
- initiates a thread for each request,
- combines the intermediate results into the final answer

«Search Protocol»: HTTP/HTML



Μετα-Μηχανή Αναζήτησης: Μηχανή αναζήτησης που προωθεί την επερώτηση σε πολλές μηχανές αναζήτησης και ενοποιεί τα αποτελέσματα που επιστρέφουν

Γιατί φτιάχνουμε μετα-μηχανές αναζήτησης;

- Καλύτερη **κάλυψη**:
  - Το σύνολο των σελίδων που είναι γνωστές (ευρετηριασμένες) σε κάθε μηχανή είναι διαφορετικό
- **Διάταξη Πλειοψηφούσας Γνώμης** (consensus ranking)
  - Η διαθεσιμότητα πολλών μηχανών μας δίνει την δυνατότητα να ορίσουμε ένα αθροιστικό (πλειοψηφικό) μέτρο συνάφειας
    - Ενοποίηση αποτελεσμάτων = Πρόβλημα απόφασης ομάδας (group decision problem)
- **Μείωση spam**:
  - Δύσκολα μια spam σελίδα μπορεί να ξεγελάσει όλες τις μηχανές

Μετα-Μηχανές Αναζήτησης

- Ενδεικτικές μηχανές:
  - Dogpile (http://www.dogpile.com/)
    - over Google, Yahoo!, msn, Ask Jeaves
  - SurfWax (http://www.surfwax.com/)
  - http://www.jux2.com/
  - Metacrawler, SavvySearch,
- Βήματα Λειτουργίας
  - Submit queries to host sites.
  - **Parse resulting HTML pages** to extract search results.
  - **Integrate multiple rankings into a "consensus" ranking**.
  - Present integrated results to user.
- Διαφορές με την Κατανεμημένη Ανάκτηση Πληροφοριών
  - οι υποκείμενες μηχανές δεν παρέχουν term-statistics, άρα μπορούμε να χρησιμοποιήσουμε μόνο απομονωμένες (isolated) τεχνικές ενοποίησης αποτελεσμάτων
  - οι υποκείμενες μηχανές δεν υποστηρίζουν την ίδια ερωτηματική γλώσσα

Ενοποίηση Διατάξεων

Ενοποίηση Διατάξεων: **Rank Aggregation (or Meta-Ranking)** (isolated)

Διατύπωση του Προβλήματος

- ***D***: ένα σύνολο αντικειμένων (π.χ. εγγράφων)
- ***S1,…Sk***: ένα σύνολο διατάξεων του D

Σκοπός: Ενοποίηση των διατάξεων *S1,..Sk σε μία*

The metaphor: **elections**

- Objects → Candidates
- Sources → Electors
- Ordering by a system → Elector's voting ticket
- Fused ordering → Election list

Διάρθρωση

- Ενοποίηση
  - κατά Borda
  - κατά Condorcet
  - κατά Kemeny
- Επιθυμητές Ιδιότητες Τεχνικών Ενοποίησης Διατάξεων
- Το Θεώρημα του Ανέφικτου του Arrow
- Αποδοτικοί αλγόριθμοι υπολογισμού των κορυφαίων κ στοιχείων της ενοποιημένης διάταξης (Top-K Rank Aggregation)

Plurality Ranking (Απλή Πλειοψηφία)

Ο υποψήφιος με τις περισσότερες πρώτες θέσεις είναι ο νικητής.

Έστω 6 πηγές (S1,…,S6) και 4 σελίδες a,b,c,d.
Κάθε σύστημα επιστρέφει μια γραμμική διάταξη των σελίδων:
S1: <a,c,d,b>
S2: <a,b,c,d>
S3: <b,c,a,b>
S4: <b,a,d,c>
S5: <a,d,c,b>
S6: <c,a,b,d>
Μετράμε ***πόσες πρώτες θέσεις*** κατέλαβε κάθε σελίδα
a: 3
b: 2
c: 1
d: 0
Άρα η τελική κατάταξη είναι η **<a,b,c,d>**

Plurality Ranking (Απλή Πλειοψηφία)
Κάποια προβλήματα

3 συστήματα <a,c,d,b>
6 συστήματα <a,d,c,b>
3 συστήματα <b,c,d,a>
5 συστήματα <b,d, c, a>
2 συστήματα <c,b,d,a>
5 συστήματα <c,d,b,a>
2 συστήματα <d,b,c,a>
4 συστήματα <d,c,b,a>

a:9
b:8
c:7
d:6
Τελική διάταξη: **<a,b,c,d>**

Απόσυρση του d
(που ήταν τελευταίο
στην ενοποιημένη διάταξη)

3 συστήματα <a,c,b>
6 συστήματα <a,c,b>
3 συστήματα <b,c,a>
5 συστήματα <b,c, a>
2 συστήματα <c,b,a>
5 συστήματα <c,b,a>
2 συστήματα <b,c,a>
4 συστήματα <c,b,a>

a:9
b:10
c:11
Τελική διάταξη: **<c,b,a>**
Αντίστροφη της αρχικής!

Plurality Ranking (Απλή Πλειοψηφία)
Κάποια προβλήματα

3 συστήματα <a,c,d,b>
6 συστήματα <a,d,c,b>
3 συστήματα <b,c,d,a>
5 συστήματα <b,d, c, a>
2 συστήματα <c,b,d,a>
5 συστήματα <c,d,b,a>
2 συστήματα <d,b,c,a>
4 συστήματα <d,c,b,a>

a:9
b:8
c:7
d:6
Τελική διάταξη: **<a,b,c,d>**

Απόσυρση του d
Τελική διάταξη: **<c,b,a>**

Απόσυρση του a
Τελική διάταξη: **<d,c,b>**

Απόσυρση του b
Τελική διάταξη: **<d,c,a>**

Απόσυρση του c
Τελική διάταξη: **<d,b,a>**

Ενοποίηση Διατάξεων κατά **Borda**
[Jean-Charles Borda 1770]

Reinvented (for the context of Meta-Searching) in [Tzitzikas 2001]

The votes of an object o $V(o) = \sum_{i=1..k} r_i(o)$

$r_i(o)$: the position of the object o in the ordering of system S_i

The fused ordering M is derived by ordering the objects in *ascending* order wrt to their *votes*

Example:

$S_1 : < o_1, o_2, o_3 >$ $V(o_1) = 1+1+2 = 4$
$S_2 : < o_1, o_3, o_2 >$ $V(o_2) = 2+3+3 = 8$
$S_3 : < o_3, o_1, o_2 >$ $V(o_3) = 3+2+1 = 6$

$M : < o_1, o_3, o_2 >$

If each source S_i returns an ordered *subset* O_i of Obj.

$$r_i(o_j) = \begin{cases} \text{position of } o_j \text{ in } O_i, & \text{if } o_j \in O_i \\ F+1 & \text{otherwise} \end{cases} \quad \text{where } F = \max\{|O_1|,...,|O_k|\}$$

Γιατί;

Ενοποίηση Διατάξεων κατά Borda
[Tzitzikas, 2001] **Βαθμός Συμφωνίας**

The *distance* between two orderings i and j: $dist(i,j) = \sum_{o \in O} |r_i(o) - r_j(o)|$

The *mean distance* of the fused ordering 0 $Dem = \frac{\sum_{i=1..k} dist(0,i)}{k}$

The ***level of agreement*** of the fused ordering 0:

- linear transformation $LA = \frac{C - Dem}{C}$ C: Max possible mean distance
- inversion transformation $LA = C^{-Dem}$ $C > 1$, e.g. C = 2

- **High** level may drive the user to read only the very first documents since probably they are the more relevant
- **Low** level may drive the user to read more documents

Ενοποίηση Διατάξεων κατά **Condorcet** [1785]

Condorcet: the winner is a candidate that defeats every other candidate in pairwise majority-rule election

S1: <a,b,c>
S2: <b,a,c>
S2: <c,a,b>

a:b 2:1// το a νικά το b δύο φορές (και χάνει μία)
a:c 2:1// το a νικά το c δύο φορές (και χάνει μία)

Αρα η τελική κατάταξη κατά Condorset είναι: **<a,b,c>**

Ενοποίηση Διατάξεων κατά **Condorcet** [1785]

S1: <a,b,c>
S2: <b,c,a>
S3: <c,a,b>

a:b 2:1 // άρα το b δεν μπορεί να είναι ο νικητής
a:c 1:2 // άρα το a δεν μπορεί να είναι ο νικητής
c:b 1:2 // άρα το c δεν μπορεί να είναι ο νικητής

Δεν υπάρχει πάντα Condorcet νικητής!

Borda vs Condorcet

S1: <a,b,c>

S2: <b,a,c>

S2: <c,a,b>

- Condorcet
  - a:b 2:1
  - a:c 2:1
  - Condorcet ordering: <a,b,c>
- Borda
  - a: 1+2+2 = 5
  - b: 2+1+3 = 6
  - c: 3 + 3 + 1 = 7
  - Borda ordering: <a,b,c>

Borda ≠ Condorcet

Borda (1770)

- Member of French Academy of Sciences
- Noted for work in hydraulics, optics, navigation instrument
- Purpose: Reforming the election procedure of French Academy.
- Criticize plurality method

Condorcet (1785)

- Viewed Borda as an enemy
- Finding best ordering by hypothesis testing
- Switch to propose Condorcet winner

Borda ≠ Condorcet

S1: <a,b,c,d,e>
S2: <b,c,e,d,a>
S3: <e,a,b,c,d>
S4: <a,b,d,e,c>
S5: <b,a,d,e,c>

- Borda
  - a: 1 + 5 + 2 + 1 + 2 = 11
  - b: 2 + 1+ 3 + 2 + 1 = 9
  - c: 3 + 2 + 4 + 5 + 5 =19
  - d: 4 + 4 + 5 + 3 + 3 = 19
  - e: 5+3 +1 + 4 + 4 = 17
  - **Borda winner : b**
- Condorcet
  - a:b 3:2
  - a:c 4:1
  - a:d 4:1
  - a:e :3:2
  - **Condorcet winner a**

Prurality ≠ Borda ≠ Condorcet

	49 votes	48 votes	3 votes
1*st*	*x*	*y*	*z*
2*nd*	*y*	*z*	*y*
3*rd*	*z*	*x*	*x*

- Prurality winner: x
- Borda winner: y
- Condorcet: z> x

Condorcet and Order

- Θεωρείστε την περίπτωση τριών υποψηφίων (a,b,c) και 13 εκλεκτόρων

	a	*b*	*c*
a	–	8	6
b	5	–	11
c	7	2	–

Έχουμε συνοψίσει τις διατάξεις που έδωσαν οι εκλέκτορες κατασκευάζοντας έναν πίνακα C, όπου το C[i,j] εκφράζει πόσες φορές το i νικά το j

Μπορούμε να υπολογίσουμε τη στήριξη (support) κάθε πιθανής γραμμικής διάταξης αθροίζοντας τη στήριξη της κάθε συσχέτισής της.

- <a,b,c> has support 25
  - a>b:8, a>c:6, b>c:11
- <b,c,a> has support 23
  - a<b:5, c>a:7, b>c:11

Ενοποίηση Διατάξεων κατά **Kemeny** (1959)

(Kemeny developed BASIC language)

- Απόσταση (distance) μεταξύ δυο διατάξεων = πλήθος των διαφωνιών στη διάταξη ζευγαριών
- Παράδειγμα 1
  - r1 = <a,b,c>
  - r2 = <b, a, c>
  - K(r1, r2) = 1
    - $(a >_{r1} b, a <_{r2} b)$
- Παράδειγμα 2
  - r1 = <a, b, c, d>
  - r2 = <b, d, a, c>
  - K(r1, r2) = 3
    - $(a >_{r1} b, a <_{r2} b)$ $(a >_{r1} d, a <_{r2} d)$ $(c >_{r1} d, c <_{r2} d)$

Ενοποίηση Διατάξεων κατά **Kemeny** (1959)

Kemeny Optimal Aggregation

- Η καλύτερη ενοποιημένη διάταξη είναι εκείνη που απέχει το λιγότερο από όλες τις διατάξεις
- Έστω n διατάξεις: r1, r2, …, rn
- Ενοποιημένη διάταξη $r = \arg\min \sum K(r,ri)$
- Η εύρεση της ενοποιημένης διάταξης είναι ακριβή
  - (πρόβλημα NP-hard)
- Reconciles Borda and Condorcet

Ενοποίηση Διατάξεων: **Επιθυμητές Ιδιότητες**

- Ουδετερότητα (Neutrality)
  - Καμία εναλλακτική δεν πρέπει να ευνοείται
- Pareto Optimality
  - Αν X > Y (σε όλες τις διατάξεις) τότε X > Y (στην τελική)
- Μονοτονία (Monotonicity) // Ranking higher should not hurt a candidate
  - αν ο X προηγείται του Y (X > Y) στην τελική διάταξη, αλλαγή ενός ψηφοδελτίου YZX → XYZ, ο X θα εξακολουθήσει να προηγείται
- Ανεξαρτησία από άσχετες εναλλακτικές (Independence from Irrelevant Alternatives)
  - X > Y (στην τελική), αλλαγή ενός ψηφοδελτίου XZY → ZXY, το X>Y παραμένει στην τελική
- Συνέπεια (Consistency)
  - Αν οι ψηφοφόροι διαιρεθούν σε δύο ομάδες και κάθε ομάδα αναδείξει τον ίδιο νικητή, τότε ο τελικός νικητής (αν λάβουμε υπόψη τις ψήφους και των 2 ομάδων) πρέπει να είναι ο ίδιος

Arrow's Impossibility Theorem

Kenneth J. Arrow, *Social Choice and Individual Values* (1951). Won Nobel Prize in 1972

No voting scheme over three or more alternatives can satisfy the following conditions

- Universality (no restriction on individual ordering. All orderings are achievable. Deterministic)
- Monotonicity
- Independence of irrelevant alternatives
- Pareto Optimality
- Non-dictatorship (No voter in the society is a dictator in the sense that, there does not exist a single voter *i* in the society such that for every set of orderings in the domain and every pair of distinct social states *x* and *y*, if voter *i* strictly prefers *x* over *y*, *x* is socially selected over *y*.)

Συμπέρασμα: δεν υπάρχει μια απολύτως ικανοποιητική συνάρτηση ενοποίησης διατάξεων

Διάρθρωση

- Ενοποίηση
  - κατά Borda
  - κατά Condorcet
  - κατά Kemeny
- Επιθυμητές Ιδιότητες Τεχνικών Ενοποίησης Διατάξεων
- Το Θεώρημα του Ανέφικτου του Arrow
- Αποδοτικοί αλγόριθμοι υπολογισμού των κορυφαίων κ στοιχείων της ενοποιημένης διάταξης (Top-K Rank Aggregation)

Top-k Rank Aggregation

- Έχουμε **N** αντικείμενα και τους **βαθμούς** τους βάσει **m** διαφορετικών **κριτηρίων**.
- Έχουμε έναν τρόπο να συνδυάζουμε τα m σκορ κάθε αντικειμένου σε ένα ενοποιημένο σκορ
  - π.χ. min, avg, sum
- Στόχος: Βρες τα **κ** αντικείμενα με το υψηλότερο ενοποιημένο σκορ.

Εφαρμογές:

Υπολογισμός των κορυφαίων-κ στοιχείων της απάντησης

- ενός ΣΑΠ που βασίζεται στο διανυσματικό μοντέλο (τα m κριτήρια είναι οι m όροι της επερώτησης)
- ενός μεσίτη (π.χ. μετα-μηχανής αναζήτησης) πάνω από m Συστήματα Ανάκτησης Πληροφοριών
- μιας επερώτησης σε μια Βάση Πολυμέσων
  - κριτήρια (και συνάμα χαρακτηριστικά/features): χρώμα, μορφή, υφή, …

Άλλο ένα παράδειγμα εφαρμογής

Ενοποίηση απαντήσεων σε Μεσολαβητές (middleware) πάνω από πηγές που αποθηκεύουν δομημένες πληροφορίες

- έστω μια υπηρεσία εύρεσης εστιατορίων βάσει τριών κριτηρίων:
  - απόσταση από ένα σημείο
  - κατάταξη εστιατορίου
  - τιμή γεύματος, και άλλα
- όπου ο χρήστης μπορεί να ορίσει τον επιθυμητό τρόπο υπολογισμού του ενοποιημένου σκορ ενός εστιατορίου
  - π.χ. Σκορ(εστΧ) = Stars(εστΧ)*0.25 + 0.75*DistanceFromHome(εστΧ)
- η υπηρεσία αυτή υλοποιείται με χρήση τριών απομακρυσμένων υπηρεσιών
  - (α) getRestaurantsByStars
    - επιστρέφει όλα τα εστιατόρια σε φθίνουσα σειρά ως προς τα αστέρια που έχουν (κάθε εστιατόριο συνοδεύεται με ένα σκορ)
  - (β) getRestaurantsByDistance(x,y)
    - επιστρέφει όλα τα εστιατόρια σε φθίνουσα σειρά ως προς την απόσταση τους από ένα συγκεκριμένο σημείο με συντεταγμένες (x,y) // κάθε εστιατόριο συνοδεύεται από την απόσταση του από το (x,y)

Πως μπορώ να ελαχιστοποιήσω το πλήθος των στοιχείων που πρέπει να διαβάσω από την απάντηση της κάθε υπηρεσίας, προκειμένου να βρω τα κορυφαία 5 εστιατόρια (βάσει σκορ όπως υπολογίζεται από της συνάρτηση βαθμολόγησης που έδωσε ο χρήστης);

Εύρεση των κ-κορυφαίων
Απλοϊκός Αλγόριθμος

1/ Ανέκτησε ολόκληρες τις m λίστες
2/ Υπολόγισε το ενοποιημένο σκορ του κάθε αντικειμένου
3/ Ταξινόμησε τα αντικείμενα βάσει του σκορ και επέλεξε τα πρώτα κ

Παρατηρήσεις
- Κόστος γραμμικό ως προς το μήκος των λιστών
- Δεν αξιοποιεί το γεγονός ότι οι λίστες είναι ταξινομημένες

Εύρεση των κ-κορυφαίων
Παράδειγμα: Απλοϊκός Τρόπος

Έστω οτι θέλουμε να συναθροίσουμε τις διατάξεις που
επιστρέφουν 3 πηγές S1, S2, S3 και ο τρόπος συνάθροισης είναι το άθροισμα.

S1 = < **A** 0.9, **C** 0.8, **E** 0.7, **B** 0.5, **F** 0.5, **G** 0.5, **H** 0.5 >
S2 = < **B** 1.0, **E** 0.8, **F** 0.7, **A** 0.7, **C** 0.5, **H** 0.5, **G** 0.5 >
S3 = < **A** 0.8, **C** 0.8, **E** 0.7, **B** 0.5, **F** 0.5, **G** 0.5, **H** 0.5 >

Ο Απλοϊκός Τρόπος

Score(**A**) = 0.9 + 0.7 + 0.8 = 2.4
Score(**B**) = 0.5 + 1.0 + 0.5 = 2
Score(**C**) = 0.8 + 0.5 + 0.8 = 2.1
Score(**E**) = 0.7 + 0.8 + 0.7 = 2.2
Score(**F**) = 0.5 + 0.7 + 0.5 = 1.7
Score(**G**) = 0.5 + 0.5 + 0.5 = 1.5
Score(**H**) = 0.5 + 0.5 + 0.5 = 1.5
Τελική διάταξη: < **A, E, C, B, F, G, H>**

Εύρεση των κ-κορυφαίων
Πιο Αποδοτικοί Αλγόριθμοι

- Γενική ιδέα: ***Άρχισε να διαβάζεις τις διατάξεις από την κορυφή. Προσπάθησε να καταλάβεις πότε πρέπει να σταματήσεις***.
- Αλγόριθμοι
  - **Fagin Algorithm (FA)** [Fagin 1999, J. CSS 58]
  - **Threshold Algorithm (TA)** [Fagin et al., PODS'2001]

Εύρεση των κ-κορυφαίων
Πιο Αποδοτικοί Αλγόριθμοι

Υποθέσεις

1. Υποθέτουμε ότι έχουμε στη διάθεση μας 2 τρόπους πρόσβασης στα αποτελέσματα μιας πηγής:
   Σειριακή πρόσβαση στις διατάξεις: φθίνουσα ως προς το σκορ
   Τυχαία προσπέλαση: Δυνατότητα εύρεσης του σκορ ενός συγκεκριμένου αντικειμένου με μία πρόσβαση
2. Συναρτήσεις βαθμολόγησης (σκορ)
   Τα σκορ ανήκουν στο διάστημα [0,1]
   Η συνάρτηση ενοποιημένου σκορ είναι **μονότονη**
   αν όλα (m) τα σκορ ενός αντικειμένου Α είναι μεγαλύτερα ή ίσα των αντίστοιχων σκορ ενόςαντικειμένου Β, τότε σίγουρα το ενοποιημένο σκορ του Α είναι μεγαλύτερο ή ίσο του σκορ του Β

Εύρεση των κ-κορυφαίων

Ο Αλγόριθμος του Fagin (FA) [1999]

1.α/ Κάνε σειριακή ανάκτηση αντικειμένων από κάθε λίστα (αρχίζοντας από την κορυφή), έως ότου η τομή των αντικειμένων από κάθε λίστα να έχει κ αντικείμενα

1.β/ Για κάθε αντικείμενο που ανακτήθηκε (στο 1.α) συνέλεξε τα σκορ που λείπουν (με χρήση του μηχανισμού τυχαίας προσπέλασης)

2/ Υπολόγισε το ενοποιημένο σκορ του κάθε αντικειμένου

3/ Ταξινόμησε τα αντικείμενα βάσει του ενοποιημένου σκορ και επέλεξε τα πρώτα κ

Σχόλια

Αξιοποιεί (α) το γεγονός ότι οι λίστες είναι ταξινομημένες και (β) ότι η συνάρτηση ενοποίησης είναι μονότονη

[-] Το πλήθος των αντικειμένων που θα ανακτηθούν μπορεί να είναι μεγάλο

Για οποιοδήποτε μη επιλεγμένο αντικείμενο υπάρχουν (τουλάχιστον) κ που είναι καλύτερα από αυτό

Εύρεση των κ-κορυφαίων

Παράδειγμα: Αλγόριθμος του Fagin (FA)

S1 = < **A** 0.9, **C** 0.8, **E** 0.7, **B** 0.5, **F** 0.5, **G** 0.5, **H** 0.5 >

S2 = < **B** 1.0, **E** 0.8, **F** 0.7, **A** 0.7, **C** 0.5, **H** 0.5, **G** 0.5 >

S3 = < **A** 0.8, **C** 0.8, **E** 0.7, **B** 0.5, **F** 0.5, **G** 0.5, **H** 0.5 >

Έστω ότι θέλω το Top-1

Το E εμφανίζεται σε όλες

(**μονοτονία => δεν μπορεί κάποιο δεξιότερο του E να είναι καλύτερο του E**

Το E δεν είναι σίγουρα ο νικητής.

Υποψήφιοι νικητές = {A, B, C, E, F}. Κάνουμε τυχαίες προσπελάσεις για να βρούμε τα σκορ που μας λείπουν

getScore(S2,A), getScore(S1,B), getScore(S3,B), getScore(S2,C), …

Πράγματι, top-1= {A}

Εύρεση των κ-κορυφαίων

Παράδειγμα: Αλγόριθμος του Fagin (FA)

S1 = < **A** 0.9, **C** 0.8, **E** 0.7, **B** 0.5, **F** 0.5, **G** 0.5, **H** 0.5 >

S2 = < **B** 1.0, **E** 0.8, **F** 0.7, **A** 0.7, **C** 0.5, **H** 0.5, **G** 0.5 >

S3 = < **A** 0.8, **C** 0.8, **E** 0.7, **B** 0.5, **F** 0.5, **G** 0.5, **H** 0.5 >

Έστω ότι θέλω το Top-2

Το E, B (και το A) εμφανίζονται σε όλες

(**μονοτονία => δεν μπορεί κάποιο δεξιότερο του B να είναι καλύτερο του B**

Εύρεση των κ-κορυφαίων

Ο Αλγόριθμος TA (Threshold Algorithm) [Fagin et al. 2001]

Ιδέα:

Υπολόγισε το μέγιστο σκορ που μπορεί να έχει ένα αντικείμενο που δεν έχουμε συναντήσει ακόμα.

1/ Κάνε σειριακή ανάκτηση αντικειμένων από κάθε λίστα (αρχίζοντας από την κορυφή) και με χρήση *τυχαίας προσπέλασης* βρες όλα τα σκορ κάθε αντικειμένου

2/ Ταξινόμησε τα αντικείμενα (βάσει του ενοποιημένου σκορ) και κράτησε τα καλύτερα κ

3/ Σταμάτησε την σειριακή ανάκτηση όταν τα σκορ των παραπάνω κ αντικειμένων δεν μπορεί να είναι μικρότερα του μέγιστου πιθανού σκορ των απαρατήρητων αντικειμένων (threshold).

Εύρεση των κ-κορυφαίων
Παράδειγμα: Αλγόριθμος TA:

S1 = < **A** 0.9, **C** 0.8, **E** 0.7, **B** 0.5, **F** 0.5, **G** 0.5, **H** 0.5 >
S2 = < **B** 1.0, **E** 0.8, **F** 0.7, **A** 0.7, **C** 0.5, **H** 0.5, **G** 0.5 >
S3 = < **A** 0.8, **C** 0.8, **E** 0.7, **B** 0.5, **F** 0.5, **G** 0.5, **H** 0.5 >

Έστω ότι θέλω το Top-1

Score(A) = 0.9 + 0.7 + 0.8 = 2.4
Score(B) = 0.5 + 1.0 + 0.5 = 2
UpperBound = 0.9 + 1.0 + 0.8 = 2.7
αφού 2.7 > 2.4 συνεχίζω

Score(C) = 0.8 + 0.5 + 0.8 = 2.1
Score(E) = 0.7 + 0.8 + 0.7 = 2.2
UpperBound = 0.8 + 0.8 + 0.8 = 2.4
αφού 2.4 δεν είναι μεγαλύτερο του 2.4 (σκορ του A) σταματάω.

Σύγκριση: Fagin vs. TA

- Ο FA ποτέ δεν τερματίζει νωρίτερα του TA
- Ο TA χρειάζεται μόνο έναν μικρό (k) ενταμιευτή (buffer)
- Ο TA μπορεί όμως να κάνει περισσότερες τυχαίες προσπελάσεις

Ο TA είναι βέλτιστος για όλες τις μονότονες συναρτήσεις σκορ
– Συγκεκριμένα, είναι "instant optimal": είναι καλύτερος πάντα (όχι μόνο στην χειρότερη περίπτωση ή στην μέση περίπτωση)

- Επεκτάσεις
  - Αλγόριθμος NRA (Non Random Access)
    - Έκδοση του TA για την περίπτωση που η τυχαία πρόσβαση είναι αδύνατη. Επίσης "instant optimal".
      - Do sequential access until there are *k* objects whose lower bound no less than the upper bound of all other objects
    - Αλγόριθμος CA (Combined Algorithm)
      - Έκδοση του TA που θεωρεί τις τυχαίες προσπελάσεις ακριβότερες των σειριακών.