Introduction to

Information Retrieval

ΠΛΕ70: Ανάκτηση Πληροφορίας

Διδάσκουσα: Ευαγγελία Πιτουρά

Διαλέξεις 6-7: Επανάληψη Διάταξης Εγγράφων. Θέματα Υλοποίησης. Περίληψη Αποτελεσμάτων.

1

Περίληψη διαβάθμισης

- Βαθμολόγηση και κατάταξη εγγράφων
- Στάθμιση όρων (term weighting)
- Αναπαράσταση εγγράφων και ερωτημάτων ως διανύσματα

2

Κατάταξη εγγράφων (Ranked retrieval)

- Μέχρι τώρα, τα ερωτήματα που είδαμε ήταν Boolean.
  - Τα έγγραφα ήταν ταίριαζαν, είτε όχι
- Κατάλληλη *για ειδικούς* με σαφή κατανόηση των αναγκών τους και της συλλογής
  - Αλλά, όχι κατάλληλη για την πλειοψηφία των χρηστών
- Το πρόβλημα με τα πάρα πολλά ή τα πολύ λίγα αποτελέσματα

Μοντέλα διαβαθμισμένης ανάκτησης

- Αντί ενός **συνόλου** εγγράφων που ικανοποιούν το ερώτημα, η διαβαθμισμένη ανάκτηση (ranked retrieval) επιστρέφει μια **διάταξη** των (κορυφαίων) για την ερώτηση εγγράφων της συλλογής
- Συνήθως μαζί με ερωτήματα **ελεύθερου κειμένου** (Free text queries)
- Πως διατάσουμε-διαβαθμίζουμε τα έγγραφα μιας συλλογής με βάση ένα ερώτημα
  - Αναθέτουμε ένα **βαθμό (score)** ***score(d, q)*** μετρά πόσο καλά το έγγραφο d "ταιριάζει" (match) με το ερώτημα q

Διαβαθμισμένη ανάκτηση

- Όταν το σύστημα παράγει ένα διατεταγμένο σύνολο αποτελεσμάτων, τα μεγάλα σύνολα δεν αποτελούν πρόβλημα
  - Δείχνουμε απλώς τα κορυφαία (top) *k* (≈ 10) αποτελέσματα
  - Δεν παραφορτώνουμε το χρήστη

Προϋπόθεση: ο αλγόριθμος διάταξης δουλεύει σωστά

Βαθμός ταιριάσματος ερωτήματος-εγγράφου

- Χρειαζόμαστε ένα τρόπο για να αναθέσουμε ένα βαθμό σε κάθε ζεύγος ερωτήματος(q)/εγγράφου(d)

score(d, q)

Επιθυμητές ιδιότητες:

- Αν *κανένας όρος του ερωτήματος* δεν εμφανίζεται στο έγγραφο, τότε ο βαθμός θα πρέπει να είναι 0
- Όσο πιο συχνά εμφανίζεται κάποιος όρος του ερωτήματος σε ένα έγγραφο, τόσο μεγαλύτερος θα πρέπει να είναι ο βαθμός

Στάθμιση με Log-συχνότητας

- Η συχνότητα όρου $tf_{t,d}$ του όρου t σε ένα έγγραφο d ορίζεται ως αριθμός των φορών που το t εμφανίζεται στο d.
- *Επειδή η συνάφεια (Relevance) δεν αυξάνει αναλογικά με τη συχνότητα όρου*, στάθμιση με χρήση του λογάριθμου της συχνότητα (log frequency weight) του όρου t στο d είναι

$$w_{t,d} = \begin{cases} 1 + \log_{10} \mathrm{tf}_{t,d}, & \text{if } \mathrm{tf}_{t,d} > 0 \\ 0, & \text{otherwise} \end{cases}$$

  - 0 → 0, 1 → 1, 2 → 1.3, 10 → 2, 100->3, 1000 → 4, κλπ.

Στάθμιση με Log-συχνότητας

- Ο βαθμός για ένα ζεύγος εγγράφου-ερωτήματος: άθροισμα όλων των κοινών όρων:

$$score = \sum_{t \in q \cap d} (1 + \log \mathrm{tf}_{t,d})$$

- Ο βαθμός είναι 0 όταν κανένας από τους όρους του ερωτήματος δεν εμφανίζεται στο έγγραφο

Συχνότητα εγγράφων (Document frequency)

Οι σπάνιοι όροι δίνουν περισσότερη πληροφορία από τους συχνούς όρους

- Θυμηθείτε τα stop words (διακοπτόμενες λέξεις)

- Θεωρείστε έναν όρο σε μια ερώτηση που είναι σπάνιος στη συλλογή (π.χ., *arachnocentric*)
  - Το έγγραφο που περιέχει αυτόν τον όρο είναι πιο πιθανό να είναι πιο σχετικό με το ερώτημα από ένα έγγραφο που περιέχει ένα λιγότερο σπάνιο όρο του ερωτήματος

→ Θέλουμε να δώσουμε μεγαλύτερο βάρος στους σπάνιους όρους – αλλά πως; **df**

Βάρος idf

- df_t είναι η συχνότητα εγγράφων του t: ο αριθμός (πλήθος) των εγγράφων της συλλογής που περιέχουν το t
  - df_t είναι η αντίστροφη μέτρηση της πληροφορίας που παρέχει ο όρος t
  - $\mathrm{df}_t \leq N$
- Ορίζουμε την αντίστροφη συχνότητα εγγράφων idf (inverse document frequency) του t ως

$$\mathrm{idf}_t = \log_{10}(N/\mathrm{df}_t)$$

  - Χρησιμοποιούμε $\log(N/\mathrm{df}_t)$ αντί για N/df_t για να «ομαλοποιήσουμε» την επίδραση του idf.

Παράδειγμα idf, έστω N = 1 εκατομμύριο

term	df_t	idf_t
calpurnia	1	6
animal	100	4
sunday	1,000	3
fly	10,000	2
under	100,000	1
the	1,000,000	0

$$\text{idf}_t = \log_{10}(N/\text{df}_t)$$

✓ Κάθε όρος στη συλλογή έχει μια τιμή idf

Στάθμιση tf-idf

Το tf-idf βάρος ενός όρου είναι το γινόμενο του βάρους tf και του βάρους idf.

$$w_{t,d} = \log(1+\text{tf}_{t,d}) \times \log_{10}(N/\text{df}_t)$$

- Το *πιο γνωστό* σχήμα διαβάθμισης στην ανάκτηση πληροφορίας -- Εναλλακτικά ονόματα: tf.idf, tf x idf
- Αυξάνει με τον αριθμό εμφανίσεων του όρου στο έγγραφο
- Αυξάνει με τη σπανιότητα του όρου στη συλλογή

Βαθμός εγγράφου και ερώτησης

$$\text{Score}(q,d) = \sum_{t \in q \cap d} \text{tf.idf}_{t,d}$$

Υπάρχουν πολλές άλλες παραλλαγές

- Πως υπολογίζεται το "tf" (με ή χωρίς log)
- Αν δίνεται βάρος και στους όρους του ερωτήματος

Δυαδική μήτρα σύμπτωσης (binary term-document incidence matrix)

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	1	1	0	0	0	1
Brutus	1	1	0	1	0	0
Caesar	1	1	0	1	1	1
Calpurnia	0	1	0	0	0	0
Cleopatra	1	0	0	0	0	0
mercy	1	0	1	1	1	1
worser	1	0	1	1	1	0

Κάθε έγγραφο αναπαρίσταται ως ένα δυαδικό διάνυσμα $\in \{0,1\}^{|V|}$(την αντίστοιχη στήλη)

Ο πίνακας με μετρητές

Κάθε έγγραφο είναι ένα διάνυσμα μετρητών (συχνότητα εμφάνισης του όρου στο έγγραφο) στο $\mathbb{N}^{|v|}$: μια στήλη παρακάτω

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	157	73	0	0	0	0
Brutus	4	157	0	1	0	0
Caesar	232	227	0	2	1	1
Calpurnia	0	10	0	0	0	0
Cleopatra	57	0	0	0	0	0
mercy	2	0	3	5	5	1
worser	2	0	1	1	1	0

Ο πίνακας με βάρη

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	5.25	3.18	0	0	0	0.35
Brutus	1.21	6.1	0	1	0	0
Caesar	8.59	2.54	0	1.51	0.25	0
Calpurnia	0	1.54	0	0	0	0
Cleopatra	2.85	0	0	0	0	0
mercy	1.51	0	1.9	0.12	5.25	0.88
worser	1.37	0	0.11	4.15	0.25	1.95

Θεωρούμε το tf-idf βάρος του όρου:

- Κάθε έγγραφο είναι ένα διάνυσμα tf-idf βαρών στο $R^{|v|}$

Τα έγγραφα ως διανύσματα

Έχουμε ένα |V|-διάστατο διανυσματικό χώρο

- Οι όροι είναι οι άξονες αυτού του χώρου
- Τα έγγραφα είναι σημεία ή διανύσματα σε αυτόν τον χώρο

- Πολύ μεγάλη διάσταση: δεκάδες εκατομμύρια διαστάσεις στην περίπτωση της αναζήτησης στο web
- Πολύ αραιά διανύσματα – οι περισσότεροι όροι είναι 0

Τα ερωτήματα ως διανύσματα

- Βασική ιδέα 1: Εφαρμόζουμε το ίδιο και για τα ερωτήματα, δηλαδή, *αναπαριστούμε και τα ερωτήματα ως διανύσματα στον ίδιο χώρο*

- Βασική ιδέα 2: Διαβάθμιση των εγγράφων με βάση το πόσο κοντά είναι στην ερώτηση σε αυτό το χώρο
  - Κοντινά = ομοιότητα διανυσμάτων
  - Ομοιότητα ≈ αντίθετο της απόστασης

Ομοιότητα διανυσμάτων

Πρώτη προσέγγιση: απόσταση μεταξύ δυο διανυσμάτων

- Ευκλείδεια απόσταση;
  - Δεν είναι καλή ιδέα – είναι μεγάλη για διανύσματα διαφορετικού μήκους

cosine(query, document)

Dot product Unit vectors

$$\cos(\vec{q},\vec{d}) = \frac{\vec{q}\bullet\vec{d}}{|\vec{q}||\vec{d}|} = \frac{\vec{q}}{|\vec{q}|}\bullet\frac{\vec{d}}{|\vec{d}|} = \frac{\sum_{i=1}^{|V|} q_i d_i}{\sqrt{\sum_{i=1}^{|V|} q_i^2}\sqrt{\sum_{i=1}^{|V|} d_i^2}}$$

q_i είναι το tf-idf βάρος του όρου i στην ερώτηση
d_i είναι το tf-idf βάρος του όρου i στο έγγραφο

$\cos(\vec{q}, \vec{d})$ είναι η ομοιότητα συνημίτονου των $\vec{q}$ και $\vec{d}$, που ορίζεται ως το συνημίτονο της γωνίας μεταξύ των $\vec{q}$ και $\vec{d}$.

Κανονικοποίηση του μήκους

- Ένα διάνυσμα μπορεί να κανονικοποιηθεί διαιρώντας τα στοιχεία του με το μήκος του, με χρήση της L_2 νόρμας:

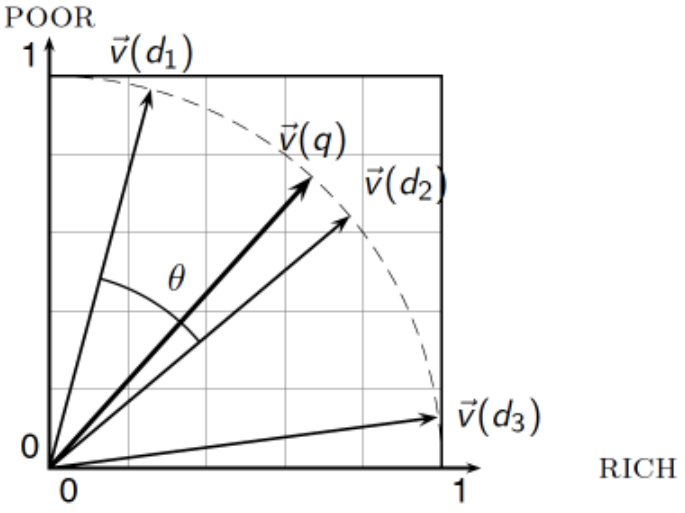
$$\|\vec{x}\|_2 = \sqrt{\sum_i x_i^2}$$

- Διαιρώντας ένα διάνυσμα με την L_2 νόρμα το κάνει μοναδιαίο
  - *Ως αποτέλεσμα, μικρά και μεγάλα έγγραφα έχουν συγκρίσιμα βάρη*
- Για διανύσματα που έχουμε κανονικοποιήσει το μήκος τους (length-normalized vectors) το συνημίτονο είναι απλώς το εσωτερικό γινόμενο (dot or scalar product):

$$\cos(\vec{q},\vec{d}) = \vec{q} \bullet \vec{d} = \sum_{i=1}^{|V|} q_i d_i$$

21

Ομοιότητα συνημιτόνου



22

Παραλλαγές της tf-idf στάθμισης

Term frequency		Document frequency		Normalization	
n (natural)	$\mathrm{tf}_{t,d}$	n (no)	1	n (none)	1
l (logarithm)	$1+\log(\mathrm{tf}_{t,d})$	t (idf)	$\log\frac{N}{\mathrm{df}_t}$	c (cosine)	$\frac{1}{\sqrt{w_1^2+w_2^2+\ldots+w_M^2}}$
a (augmented)	$0.5+\frac{0.5\times\mathrm{tf}_{t,d}}{\max_t(\mathrm{tf}_{t,d})}$	p (prob idf)	$\max\{0,\log\frac{N-\mathrm{df}_t}{\mathrm{df}_t}\}$	u (pivoted unique)	$1/u$
b (boolean)	$\begin{cases}1 & \text{if } \mathrm{tf}_{t,d}>0\\ 0 & \text{otherwise}\end{cases}$			b (byte size)	$1/CharLength^{\alpha}$, $\alpha<1$
L (log ave)	$\frac{1+\log(\mathrm{tf}_{t,d})}{1+\log(\mathrm{ave}_{t\in d}(\mathrm{tf}_{t,d}))}$				

Augmented: θεωρούμε τη συχνότητα του πιο συχνού όρου στο έγγραφο και κανονικοποιούμε με αυτήν
Το 0.5 είναι ένας τελεστές στάθμισης (εξομάλυνσης)

Στάθμιση ερωτημάτων και εγγράφων

- Πολλές μηχανές αναζήτησης σταθμίζουν διαφορετικά τις ερωτήσεις από τα έγγραφα
- Συμβολισμό: *ddd.qqq,* με χρήση των ακρονύμων του πίνακα (πρώτα 3 γράμματα έγγραφο- επόμενα 3 ερώτημα)
- Συχνό σχήμα : lnc.ltc
- Έγγραφο: logarithmic tf (l), no idf (n), cosine normalization (c)

Γιατί;

idf: ολικό μέγεθος

Παράδειγμα

Ποια είναι οι ομοιότητα μεταξύ των έργων

SaS: *Sense and Sensibility*

PaP: *Pride and Prejudice*, and

WH: *Wuthering Heights*?

Συχνότητα όρων (μετρητές)

όρος	SaS	PaP	WH
affection	115	58	20
jealous	10	7	11
gossip	2	0	6
wuthering	0	0	38

Παράδειγμα (συνέχεια)

Για απλοποίηση δε θα χρησιμοποιήσουμε τα idf βάρη

lnc (logarithmic, none, normalized cosine)

όρος	SaS	PaP	WH
affection	115	58	20
jealous	10	7	11
gossip	2	0	6
wuthering	0	0	38

Log frequency βάρος

όρος	SaS	PaP	WH
affection	3.06	2.76	2.30
jealous	2.00	1.85	2.04
gossip	1.30	0	1.78
wuthering	0	0	2.58

Μήκος SAS $= \sqrt{3.06^2 + 2.00^2 + 1.3^2 + 0^2} \approx 3.88$

Μετά την κανονικοποίηση

όρος	SaS	PaP	WH
affection	0.789	0.832	0.524
jealous	0.515	0.555	0.465
gossip	0.335	0	0.405
wuthering	0	0	0.588

Παράδειγμα (συνέχεια)

όρος	SaS	PaP	WH
affection	0.789	0.832	0.524
jealous	0.515	0.555	0.465
gossip	0.335	0	0.405
wuthering	0	0	0.588

όρος	SaS	PaP	WH
affection	115	58	20
jealous	10	7	11
gossip	2	0	6
wuthering	0	0	38

cos(SaS,PaP) ≈
0.789 × 0.832 + 0.515 × 0.555 + 0.335 × 0.0 + 0.0 × 0.0
≈ 0.94
cos(SaS,WH) ≈ 0.79
cos(PaP,WH) ≈ 0.69

Γιατί cos(SaS,PaP) > cos(SaS,WH)?

Στάθμιση ερωτημάτων και εγγράφων

Συχνό σχήμα : lnc.ltc

- Έγγραφο: logarithmic tf, no idf, cosine normalization
- Ερώτημα: logarithmic tf (l), idf (t), cosine normalization (c)

Παράδειγμα

Έγγραφο: *car insurance auto insurance*
Ερώτημα: *best car insurance*

N = 1000K

Όρος	Ερώτημα (Query)						Έγγραφο				Prod
	tf-raw	tf-wt	df	idf	wt	n'lize	tf-raw	tf-wt	wt	n'lize	
auto	0	0	5000	2.3	0	0	1	1	1	0.52	0
best	1	1	50000	1.3	1.3	0.34	0	0	0	0	0
car	1	1	10000	2.0	2.0	0.52	1	1	1	0.52	0.27
insurance	1	1	1000	3.0	3.0	0.78	2	1.3	1.3	0.68	0.53

Μήκος Εγγράφου = $\sqrt{1^2 + 0^2 + 1^2 + 1.3^2} \approx 1.92$ lnc.ltc

Score = 0+0+(0.52*0.52=)27+(0.78*0.68=)0.53 = 0.8

Μοντέλο Σάκου Λέξεων (Bag of words model)

- Η διανυσματική αναπαράσταση δεν εξετάζει τη διάταξη των λέξεων σε ένα έγγραφο
  - *John is quicker than Mary* και
  - *Mary is quicker than John*

  Έχουν τα ίδια διανύσματα

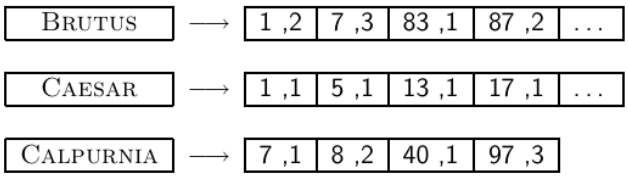
- Αυτό λέγεται μοντέλου σάκου λέξεων (bag of words model).

Περίληψη βαθμολόγησης στο διανυσματικό χώρο

- Αναπαράσταση του ερωτήματος ως ένα διαβαθμισμένο tf-idf διάνυσμα
- Αναπαράσταση κάθε εγγράφου ως ένα διαβαθμισμένο tf-idf διάνυσμα
- Υπολόγισε το συνημίτονο για κάθε ζεύγος ερωτήματος, εγγράφου
- Διάταξε τα έγγραφα με βάση αυτό το βαθμό
- Επέστρεψε τα κορυφαία K (π.χ., K =10) έγγραφα στο χρήστη

Μερικά θέματα υλοποίησης

Τροποποίηση ευρετηρίου



- Συχνότητες όρων

Σε κάθε καταχώρηση, αποθήκευση του $tf_{t,d}$ επιπρόσθετα του docID $_d$

Τροποποίηση ευρετηρίου

Σε κάθε καταχώρηση, αποθήκευση του $tf_{t,d}$ επιπρόσθετα του docID $_d$

Ως *ακέραια συχνότητα* όχι (log-)σταθμισμένο πραγματικό αριθμό γιατί οι πραγματικοί αριθμοί είναι δύσκολο να συμπιεστούν .

- Χρήση Unary code
- Επιπρόσθετος χώρος μικρός, λιγότερο από ένα byte ανά καταχώρηση με bitwise συμπίεση ή ένα byte ανά καταχώρηση με μεταβλητού μεγέθους byte code

Υπολογισμός cosine βαθμού

Υπολογισμός ανά-όρο (ένας-όρος-τη-φορά - a-term-at-a-time)

- Η απλούστερη περίπτωση είναι να επεξεργαστούμε όλη τη λίστα καταχωρήσεων του πρώτου όρου
- Δημιουργούμε ένα συσσωρευτή των βαθμών για κάθε docID εγγράφου που βρίσκουμε
- Μετά επεξεργαζόμαστε πλήρως τη λίστα καταχωρήσεων για τον δεύτερο όρο κοκ

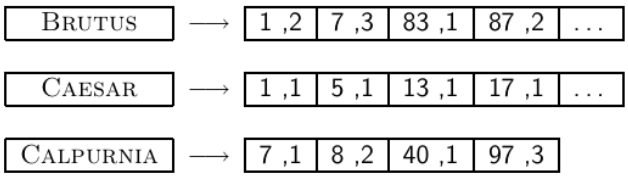
Υπολογισμός ανά όρο (term-at-a-time)

```
CosineScore(q)
 1  float Scores[N] = 0
 2  float Length[N]
 3  for each query term t
 4  do calculate w_{t,q} and fetch postings list for t
 5     for each pair(d, tf_{t,d}) in postings list
 6     do Scores[d] += w_{t,d} × w_{t,q}
 7  Read the array Length
 8  for each d
 9  do Scores[d] = Scores[d]/Length[d]
10  return Top K components of Scores[]
```

Για κάθε όρο *t* του ερωτήματος *q*

Λέμε τα στοιχεία του πίνακα Scores, συσσωρευτές (accumulators)

Παράδειγμα



- Ερώτημα: [Brutus Caesar]:
- Συσσωρευτές για τα: 1, 5, 7, 13, 17, 83, 87
- Δε χρειαζόμαστε για τα 8, 40, 85

✓ *Εξετάζουμε μόνο τα έγγραφα που έχουν μη μηδενικό συνημίτονο*

Υπολογισμός βαρών

- Η σχετική διάταξη των εγγράφων δεν επηρεάζεται από την κανονικοποίηση ή όχι του διανύσματος του q, επίσης αν κάθε όρος μόνο μια φορά στο ερώτημα το $w_{t,q}$ μπορεί να αγνοηθεί, οπότε μπορούμε απλώς να αθροίζουμε τα $w_{t,s}$
- (document-at-a-time) Μπορούμε να διατρέχουμε τις λίστες των όρων του ερωτήματος παράλληλα όπως στην περίπτωση της Boolean ανάκτησης (merge sort)
  - Αυτό έχει ως αποτέλεσμα λόγω της διάταξης των εγγράφων στις λίστες καταχωρίσεων τον υπολογισμό του βαθμού ανά έγγραφο

Πως υπολογίζουμε τα κορυφαία *k* αποτελέσματα;

Σε πολλές εφαρμογές, δε χρειαζόμαστε την πλήρη κατάταξη, αλλά μόνο τα κορυφαία *k*, για κάποιο μικρό *k*, π.χ., *k* = 100

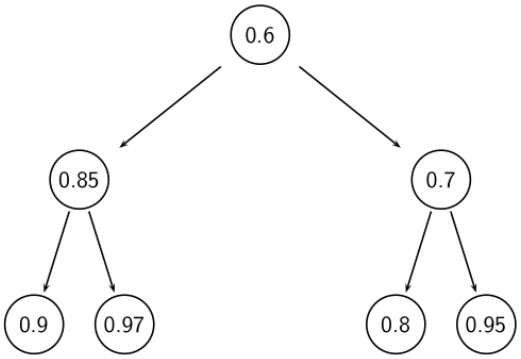
- Απλοϊκός τρόπος:
  - Υπολόγισε τους βαθμούς για όλα τα N έγραφα
  - Sort
  - Επέστεψε τα κορυφαία *k*

Αν δε χρειαζόμαστε όλη τη διάταξη, υπάρχει αποδοτικός τρόπος να υπολογίσουμε μόνο τα κορυφαία k;

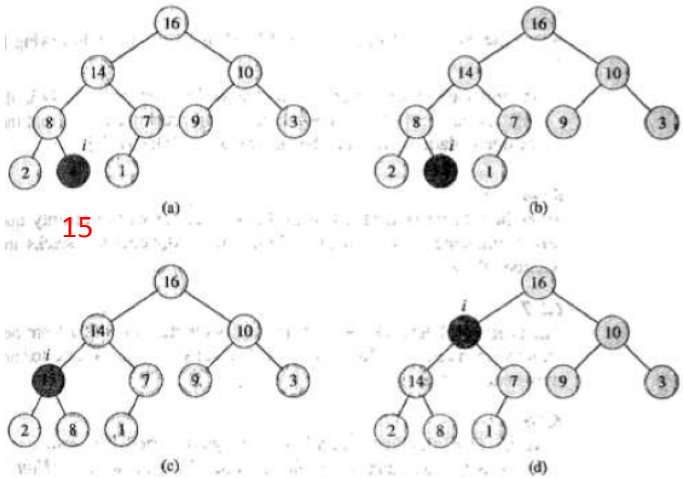
- *Έστω J τα έγγραφα με μη μηδενικό συνημίτονο. Μπορούμε να βρούμε τα K καλύτερα χωρίς ταξινόμηση όλων των J εγγράφων;*

Χρήση min heap

- Χρήση δυαδικού min heap
- Ένα δυαδικό min heap είναι ένα δυαδικό δέντρο που η τιμή ενός κόμβου είναι μικρότερη από την τιμή των δύο παιδιών του.



Παράδειγμα εισαγωγής (max heap)



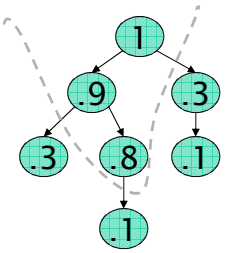
Επιλογή των κορυφαίων k σε $O(N \log k)$

Στόχος: Διατηρούμε τα καλύτερα k που έχουμε δει μέχρι στιγμής

- Χρήση δυαδικού min heap
- Για την επεξεργασία ενός νέου εγγράφου d' με score s':
  - Get *current minimum* h_m of heap ($O(1)$)
  - If $s' < h_m$ skip to next document /* υπάρχουν k καλύτερα */
  - If $s' > h_m$ heap-delete-root ($O(\log k)$) /* καλύτερο, σβήσε τη ρίζα
    heap-add d'/s' ($O(\log k)$) και βάλτο στο heap */

Χρήση max heap

- *2J* πράξεις για την κατασκευή του, βρίσκουμε τους *K* "winners" σε 2log *J* βήματα.
- Για *J*=1M, *K*=100, 10% του κόστους της ταξινόμησης.



Ακόμα πιο αποδοτικός υπολογισμός;

Η ταξινόμηση έχει πολυπλοκότητα χρόνου $O(N)$ όπου N ο αριθμός των εγγράφων (ή, ισοδύναμα J).

Βελτιστοποίηση κατά ένα σταθερό όρο, αλλά ακόμα θέλουμε $O(N)$, $N > 10^{10}$

Υπάρχουν sublinear αλγόριθμοι;

- Αυτό που ψάχνουμε στην πραγματικότητα αντιστοιχεί στο να λύνουμε το πρόβλημα των k-πλησιέστερων γειτόνων (*k*-nearest neighbor (kNN) problem) στο διάνυσμα του ερωτήματος (= query point).
- *Δεν υπάρχει γενική λύση σε αυτό το πρόβλημα που να είναι sublinear. (ειδικά για πολλές διαστάσεις)*

44

Γενική Προσέγγιση

- Βρες ένα σύνολο *A* από υποψήφια έγγραφα (*contenders*), όπου *K* < |*A*| << *N*
  - *Το A δεν περιέχει απαραίτητα όλα τα* top *K, αλλά περιέχει αρκετά καλά έγγραφα και πολλά από τα* top *K*
- Επέστρεψε τα top K έγγραφα του A

Το A είναι ένα ψαλίδισμα (pruning) των μη υποψηφίων

✓ *Έτσι και αλλιώς το συνημίτονο είναι μόνο μια «εκτίμηση» της συνάφειας*

Θα δούμε σχετικούς ευριστικούς

Περιορισμός του ευρετηρίου

- Ο βασικός αλγόριθμος υπολογισμού του συνημίτονου θεωρεί έγγραφα που περιέχουν *τουλάχιστον έναν όρο του ερωτήματος*
- Μπορούμε να επεκτείνουμε αυτήν την ιδέα;
  - Εξετάζουμε μόνο τους όρους του ερωτήματος με *μεγάλο idf*
  - Εξετάζουμε μόνο έγγραφα που περιέχουν *πολλούς από τους όρους του ερωτήματος*

Μόνο όροι με μεγάλο idf

Παράδειγμα: Για το ερώτημα «*catcher in the rye*»

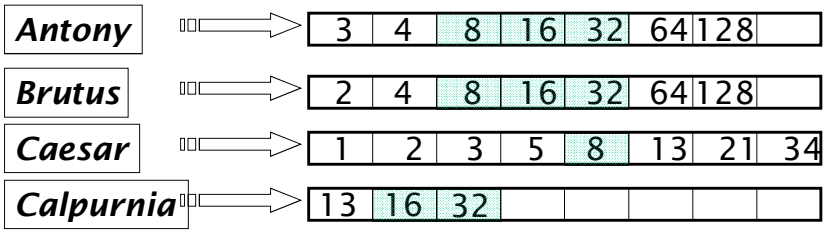
- Αθροίζουμε μόνο το βαθμό για τους όρους catcher και rye
- Γιατί; : οι όροι **in** και **the** έχουν μικρή συνεισφορά στο βαθμό και άρα δεν αλλάζουν σημαντικά τη διάταξη

- Όφελος:
  - Οι καταχωρήσεις των όρων με μικρά idf περιέχουν πολλά έγγραφα → αυτά τα (πολλά) έγγραφα δε μπαίνουν ως υποψήφια στο σύνολο A

Έγγραφα με πολλούς όρους του ερωτήματος

- Κάθε έγγραφο που έχει *τουλάχιστον έναν όρο* του ερωτήματος είναι υποψήφιο για τη λίστα με τα κορυφαία K έγγραφα
- Για ερωτήματα με πολλούς όρους, υπολογίζουμε τους βαθμούς μόνο των εγγράφων που περιέχουν αρκετούς από τους όρους του ερωτήματος
  - Για παράδειγμα, τουλάχιστον 3 από τους 4 όρους
  - Παρόμοιο με ένα είδος μερικής σύζευξη (“soft conjunction”) στα ερωτήματα των μηχανών αναζήτησης (αρχικά στη Google)
- Εύκολα να υλοποιηθεί κατά τη διάσχιση των καταχωρήσεων

3 από τους 4 όρους του ερωτήματος



Υπολογισμοί βαθμών μόνο για τα έγγραφα 8, 16 και 32

Λίστες πρωταθλητών

- **Προ-υπολογισμός** για κάθε όρο *t* του λεξικού, των ***r*** εγγράφων με το μεγαλύτερο βάρος ανάμεσα στις καταχωρήσεις του *t* -> *λίστα πρωταθλητών* (champion list , fancy list or top docs for *t*)
  - Αν tf.idf, είναι αυτά με το καλύτερο tf
- *Κατά την ώρα του ερωτήματος*, πάρε ως Α την ένωση των λιστών πρωταθλητών για τους όρους του ερωτήματος, υπολόγισε μόνο τους βαθμούς για τα έγγραφα της Α και διάλεξε τα Κ ανάμεσα τους
- Το *r* πρέπει να επιλεγεί κατά τη διάρκεια της κατασκευής του ευρετηρίου
  - Έτσι, είναι πιθανόν ότι $r < K$

Επεξεργασία Ανά-Έγγραφο και Ανά-Όρο

- ***Υπολογισμός ανά-όρο (term-at-a-time processing):*** *Υπολογίζουμε για κάθε όρο της ερώτησης, για κάθε έγγραφο που εμφανίζεται στη λίστας καταχώρησης του ένα βαθμό και μετά συνεχίζουμε με τον επόμενο όρο της ερώτησης*
- ***Υπολογισμός Ανά Έγγραφο (document-at-a-time processing):*** *Τελειώνουμε τον υπολογισμό του βαθμού ομοιότητας ερωτήματος-εγγράφου για το έγγραφο d_i πριν αρχίσουμε τον υπολογισμό βαθμού ομοιότητας ερωτήματος-εγγράφου για το έγγραφο d_{i+1}.*

Διάταξη με βάση την ποιότητα των εγγράφων

Μέχρι στιγμής η διάταξη των εγγράφων στις λίστες καταχωρήσεων γίνεται με βάση το docID.

✓ Συχνά υπάρχει ένας *ανεξάρτητος του ερωτήματος (στατικός)* χαρακτηρισμός της καταλληλότητας ("goodness", authority) του εγγράφου

Για παράδειγμα:

- Στις μηχανές αναζήτησης (στο Google) το PageRank $g(d)$ μιας σελίδας d μετρά το πόσο «καλή» είναι μια σελίδα με βάση το πόσες «καλές» σελίδες δείχνουν σε αυτήν, ή
- wikipedia σελίδες ή
- άρθρα σε μια συγκεκριμένη εφημερίδα, κλπ

Διάταξη με βάση την ποιότητα των εγγράφων

Αν υπάρχει μια διάταξη της καταλληλότητας τότε **ο συγκεντρωτικός βαθμός (net-score) ενός εγγράφου *d*** είναι ένας συνδυασμός της καταλληλότητας του εγγράφου (που έστω ότι δίνεται από μια συνάρτηση g στο [0, 1]) και της συνάφειας του με το ερώτημα *q* (που εκφράζεται από το συνημίτονο) π.χ.:

$$\text{net-score}(q, d) = g(d) + \cos(q, d)$$

Θέλουμε να επιλέξουμε σελίδες που είναι και γενικά σημαντικές (authoritative) και συναφείς ως προς την ερώτηση (το οποίο μας δίνει το συνημίτονο)

- Πως μπορούμε να επιτύχουμε γρήγορο τερματισμό (early termination); Δηλαδή να μην επεξεργαστούμε όλη τη λίστα καταχωρήσεων για να βρούμε τα καλύτερα *k*.

Διάταξη με βάση την ποιότητα των εγγράφων

- Διατάσουμε τις λίστες καταχωρήσεων με βάση την καταλληλότητα (π.χ., PageRank) των εγγράφων:

$$g(d_1) > g(d_2) > g(d_3) > \ldots$$

Η διάταξη των εγγράφων είναι ίδια για όλες τις λίστες καταχωρήσεων

✓ Τα «καλά» έγγραφα στην αρχή της κάθε λίστας, οπότε αν θέλουμε να βρούμε γρήγορα καλά αποτελέσματα μπορούμε να δούμε μόνο την αρχή της λίστας

Διάταξη με βάση την ποιότητα των εγγράφων

Υπενθύμιση net-score(q, d) = $g(d)$ + cos(q, d) και τα έγγραφα σε κάθε λίστα σε διάταξη με βάση το g

Επεξεργαζόμαστε ένα έγγραφο τη φορά – δηλαδή, για κάθε έγγραφο υπολογίζουμε πλήρως το net-score του (για όλους τους όρους του ερωτήματος)

- Έστω $g \rightarrow [0, 1]$,

  το τελευταίο k-κορυφαίο έγγραφο έχει βαθμό **1.2**

  και για το έγγραφο d που επεξεργαζόμαστε $g(d) < 0.1$, άρα και για όλα τα υπόλοιπα συνολικός βαθμός < 1.1.

  => δε χρειάζεται να επεξεργαστούμε το υπόλοιπο των λιστών

Διάταξη με βάση το βάρος του όρου στο έγγραφο

Ιδέα: δεν επεξεργαζόμαστε τις καταχωρήσεις που θα συνεισφέρουν λίγο στο τελικό βαθμό

Διάταξη των εγγράφων με βάση το βάρος (weight) $wf_{t,d}$

✓*Όχι κοινή διάταξη των εγγράφων σε όλες τις λίστες*

- Η απλούστερη περίπτωση, normalized tf-idf weight
- Τα κορυφαία k έγγραφα είναι πιθανόν να βρίσκονται στην αρχή αυτών των ταξινομημένων λιστών.

→ γρήγορος τερματισμός ενώ επεξεργαζόμαστε τις λίστες καταχωρήσεων *μάλλον* δε θα αλλάξει τα κορυφαία k έγγραφα

Υπολογισμός ανά όρο

```
CosineScore(q)
1  float Scores[N] = 0
2  float Length[N]
3  for each query term t
4  do calculate w_{t,q} and fetch postings list for t
5     for each pair(d, tf_{t,d}) in postings list
6     do Scores[d]+ = w_{t,d} × w_{t,q}
7  Read the array Length
8  for each d
9  do Scores[d] = Scores[d]/Length[d]
10 return Top K components of Scores[]
```

Μη φέρεις όλη τη λίστα καταχωρήσεων, μόνο τα πρώτα στοιχεία της

58

1. Πρόωρος τερματισμός

- Κατά τη διάσχιση των καταχωρήσεων ενός όρου *t*, σταμάτα νωρίς αφού:
  - Δεις ένα προκαθορισμένο αριθμό r από έγγραφα
  - Το $wf_{t,d}$ πέφτει κάτω από κάποιο κατώφλι
- Πάρε την ένωση του συνόλου των εγγράφων που προκύπτει
  - Ένα σύνολο για κάθε όρο
- Υπολόγισε τους βαθμούς μόνο αυτών των εγγράφων

2. idf-διατεταγμένοι όροι

Κατά την επεξεργασία των όρων του ερωτήματος

- Τους εξετάζουμε με φθίνουσα διάταξη ως προς idf
  - Όροι με μεγάλο idf πιθανών να συνεισφέρουν περισσότερο στο βαθμό
- Καθώς ενημερώνουμε τη συμμετοχή στο βαθμό κάθε όρου
  - Σταματάμε αν ο βαθμός των εγγράφων δεν μεταβάλλεται πολύ

Κλάδεμα συστάδων

Προ-επεξεργασία

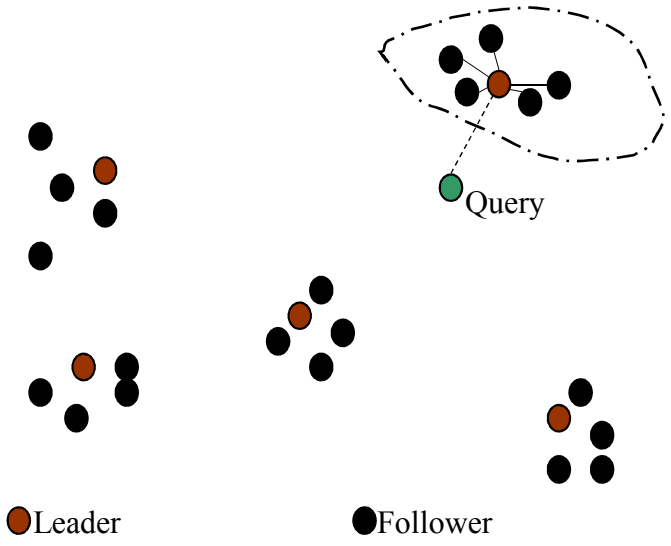
- Επέλεξε τυχαία $\sqrt{N}$ έγγραφα: τα οποία τα ονομάζουμε ηγέτες (*leaders)*
- Για κάθε άλλο έγγραφο, προ-υπολογίζουμε τον κοντινότερο ηγέτη του
  - Αυτά τα έγγραφα καλούντα ακόλουθοι (followers);
  - Ο αναμενόμενος αριθμός είναι: ~ $\sqrt{N}$ ακόλουθοι ανά ηγέτη

Κλάδεμα συστάδων

Για κάθε ερώτημα *q*

- Βρες τον πιο κοντινό ηγέτη *L*.
- Ψάξε για τα *K* πλησιέστερα έγγραφα ανάμεσα στους ακολούθους του *L*.

Κλάδεμα συστάδων



Κλάδεμα συστάδων

Γιατί τυχαία δείγματα;

- Γρήγορη
- Οι ηγέτες αντανακλούν την πραγματική κατανομή

Κλάδεμα συστάδων

Γενικές παραλλαγές (b1-b2)

- Κάθε ακόλουθος συνδέεται με *b1*=3 (έστω) πλησιέστερους ηγέτες.
- Για ένα ερώτημα, βρες *b2*=4 (έστω) κοντινότερους ηγέτες και τους ακολούθους τους

Βαθμιδωτά (διαστρωματωμένα) ευρετήρια (Tiered indexes)

Βασική ιδέα:

- Κατασκευάζουμε διάφορα επίπεδα/βαθμίδες από ευρετήρια, όπου το καθένα αντιστοιχεί στη σημαντικότητα των όρων
- Κατά τη διάρκεια της επεξεργασίας του ερωτήματος,
  - αρχίζουμε από την υψηλότερη βαθμίδα
  - Αν το ευρετήριο της υψηλότερης βαθμίδας, έχει τουλάχιστον k (π.χ., *k* = 100) αποτελέσματα: σταμάτα και επέστρεψε αυτά τα αποτελέσματα στο χρήστη
  - Αλλιώς, αν έχουμε βρει $< k$ ταιριάσματα: επανέλαβε την αναζήτηση στην επόμενη βαθμίδα

Βαθμιδωτά ευρετήρια

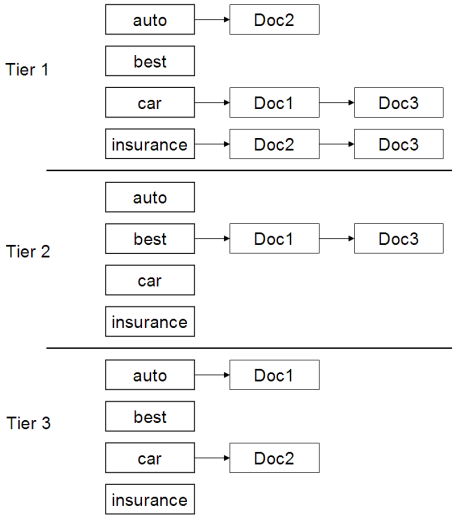
Παράδειγμα

Έστω 2 βαθμίδες

- Βαθμίδα 1: Ευρετήριο για όλους τους τίτλους (ή με τα έγγραφα με μεγάλο tf.idf)
- Βαθμίδα 2: Ευρετήριο για τα υπόλοιπο έγγραφο (ή με τα έγγραφα με μικρό tf.idf)

Οι σελίδες που περιέχουν του όρους αναζήτησης στον τίτλο είναι καλύτερα ταιριάσματα από τις σελίδες που περιέχουν τους όρους στο σώμα του εγγράφου

Βαθμιδωτά ευρετήρια



Βαθμιδωτά ευρετήρια

- Η χρήση βαθμιδωτών ευρετηρίων θεωρείται ως ένας από τους λόγους που η ποιότητα των αποτελεσμάτων του Google ήταν αρχικά σημαντικά καλύτερη (2000/01) από αυτήν των ανταγωνιστών τους.
- (μαζί με το PageRank, τη χρήση του anchor text και περιορισμών θέσεων (proximity constraints))

Συνδυασμός διανυσματικής ανάκτησης

- Πως συνδυάζουμε την ανάκτηση φράσεων (και γενικά την εγγύτητα όρων – proximity queries) με τη διανυσματική ανάκτηση;
  - Window: το μικρότερο παράθυρο που περιέχονται όλοι οι όροι του ερωτήματος μετρημένο ως το πλήθος λέξεων του παραθύρου
- Πως συνδυάζουμε την Boolean ανάκτηση με τη διανυσματική ανάκτηση;
  - Π.χ., AND ή NOT
- Πως συνδυάζουμε τα * με τη διανυσματική ανάκτηση;

Επεξεργασία ερωτήματος

- Αναλυτής ερωτημάτων (query parser)

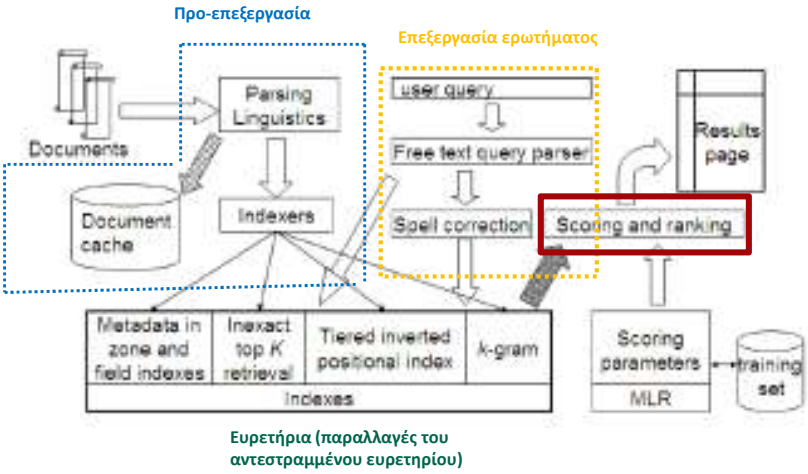
Παράδειγμα rising interest rates

1. Εκτέλεσε την ερώτημα ως ερώτημα φράσης “rising interest rates” και κατάταξε τα αποτελέσματα χρησιμοποιώντας διανυσματική βαθμολόγηση
2. Αν δεν υπάρχουν αρκετά αποτελέσματα, εκτέλεσε το ερώτημα ως 2 ερωτήματα φράσεις: “rising interest” και “interest rates” και κατάταξε τα αποτελέσματα χρησιμοποιώντας διανυσματική βαθμολόγηση
3. Αν δεν υπάρχουν αρκετά αποτελέσματα, εκτέλεσε το ερώτημα ως διάνυσμα και κατάταξε τα αποτελέσματα χρησιμοποιώντας διανυσματική βαθμολόγηση

Μπορούμε τώρα για τα έγγραφα που εμφανίζονται σε παραπάνω από ένα από τα παραπάνω βήματα να συνδυάσουμε (αθροίσουμε) τους βαθμούς

Μια εικόνα του γενικού συστήματος ΑΠ

Πλήρες σύστημα αναζήτησης



Ποια συστατικά έχουμε ήδη δει

- Προ-επεξεργασία των εγγράφων
- Ευρετήρια θέσεων (Positional indexes)
- Βαθμιδωτά ευρετήρια (Tiered indexes)
- Διορθώσεις ορθογραφικές (Spelling correction)
- *Ευρετήρα k*-γραμμάτων (για ερωτήματα με * και ορθογραφικές διορθώσεις)
- Επεξεργασία ερωτημάτων
- Βαθμολόγηση εγγράφων

Ποια συστατικά δεν έχουμε δει ακόμα

- Cache για τα έγγραφα
- Ευρετήρια ζώνης: χωρίζουν τα ευρετήρια σε διαφορετικές ζώνες: π.χ., το σώμα του κειμένου, όλο τα υπογραμμισμένο κείμενο, κείμενο άγκυρας (anchor text), κείμενο στα πεδία των μεταδεδομένων, κλπ
- Συναρτήσεις διαβάθμισης βασισμένη σε μηχανική μάθηση
- Διαβάθμιση με βάση τη γειτονικότητα (Proximity ranking)(π.χ., κατάταξε τα έγγραφα στα οποία οι όροι του ερωτήματος εμφανίζονται στο ίδιο τοπικό παράθυρο πιο ψηλά από τα έγγραφα όπου οι όροι εμφανίζονται μακριά ο ένας από τον άλλον)

Τι (άλλο) θα δούμε σήμερα;

- Περιλήψεις αποτελεσμάτων
  - Κάνοντας τα καλά αποτελέσματα χρήσιμα

Πως παρουσιάζουμε τα αποτελέσματα στο χρήστη;

Περιλήψεις αποτελεσμάτων

- Αφού έχουμε διατάξει τα έγγραφα που ταιριάζουν με το ερώτημα, θέλουμε να τα παρουσιάσουμε στο χρήστη
- Πιο συχνά ως μια λίστα από τίτλους εγγράφων, URL, μαζί με μια μικρή περίληψη, aka "10 blue links"

78

Περιλήψεις αποτελεσμάτων

- Η περιγραφή του εγγράφου είναι κρίσιμη γιατί συχνά οι χρήστες βασίζονται σε αυτήν για να αποφασίσουν αν το έγγραφο είναι σχετικό
  - Δε χρειάζεται να διαλέξουν ένα-ένα τα έγγραφα με τη σειρά

Ο τίτλος αυτόματα από μεταδεδομένα, αλλά πώς να υπολογίσουμε τις περιλήψεις;

79

Περιλήψεις αποτελεσμάτων

Δύο βασικά είδη περιλήψεων

- Μια στατική περίληψη (static summary) ενός εγγράφου είναι πάντα η ίδια ανεξάρτητα από το ερώτημα που έθεσε ο χρήστης
- Μια δυναμική περίληψη (dynamic summary) εξαρτάται από το ερώτημα (query-dependent). Προσπαθεί να εξηγήσει γιατί το έγγραφο ανακτήθηκε για το συγκεκριμένο κάθε φορά ερώτημα

Στατικές Περιλήψεις

- Σε ένα τυπικό σύστημα η στατική περίληψη είναι ένα υποσύνολο του εγγράφου
- Απλός ευριστικός: οι πρώτες περίπου 50 λέξεις του εγγράφου

  cached κατά τη δημιουργία του ευρετηρίου
- Πιο εξελιγμένες μέθοδοι – βρες από κάθε έγγραφο κάποιες σημαντικές προτάσεις
  - Απλή γλωσσολογική επεξεργασία (NLP) με ευριστικά για να βαθμολογηθεί κάθε πρόταση
  - Η περίληψη αποτελείται από τις προτάσεις με το μεγαλύτερο βαθμό
  - Ή και πιο περίπλοκη γλωσσολογική επεξεργασία για τη σύνθεση/δημιουργία περίληψης

Δυναμικές Περιλήψεις

- Παρουσίασε ένα ή περισσότερα «παράθυρα» (windows, snippets) μέσα στο έγγραφο που να περιέχουν αρκετούς από τους όρους του ερωτήματος
  - "KWIC" snippets: Keyword in Context presentation

82

Δυναμικές Περιλήψεις

- Για τον υπολογισμό των εγγράφων χρειαζόμαστε τα ίδια τα έγγραφα (δεν αρκεί το ευρετήριο)
- Cache εγγράφων – που πρέπει να ανανεώνεται
- Συχνά όχι όλο το έγγραφο αν είναι πολύ μεγάλο, αλλά κάποιο πρόθεμα του
- Βρες μικρά παράθυρα στα έγγραφα που περιέχουν όρους του ερωτήματος
  - Απαιτεί γρήγορη αναζήτηση παράθυρου στην cache των εγγράφων

83

Δυναμικές Περιλήψεις

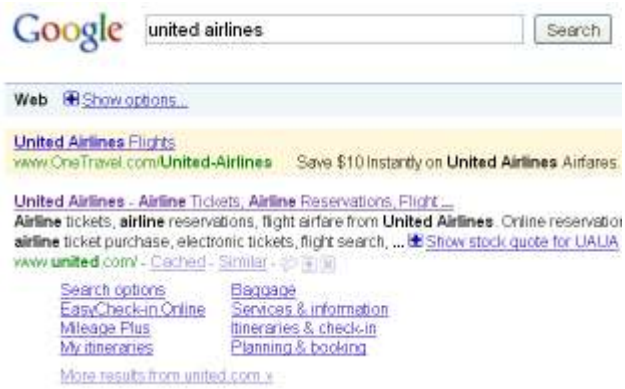
- Βαθμολόγησε κάθε παράθυρο ως προς το ερώτημα
  - Με βάση διάφορα χαρακτηριστικά το πλάτος του παραθύρου, τη θέση του στο έγγραφο, κλπ
  - Συνδύασε τα χαρακτηριστικά
- Δύσκολο να εκτιμηθεί η ποιότητα
- Positional indexes (words vs bytes)
- Ο χώρος που διατίθεται για τα παράθυρα είναι μικρός

Δυναμικές Περιλήψεις

Query: "new guinea economic development" Snippets (in bold) that were extracted from a document: . . . **In recent years, Papua New Guinea has faced severe economic difficulties and** economic growth has slowed, partly as a result of weak governance and civil war, and partly as a result of external factors such as the Bougainville civil war which led to the closure in 1989 of the Panguna mine (at that time the most important foreign exchange earner and contributor to Government finances), the Asian financial crisis, a decline in the prices of gold and copper, and a fall in the production of oil. **PNG's economic development record over the past few years is evidence that** governance issues underly many of the country's problems. Good governance, which may be defined as the transparent and accountable management of human, natural, economic and financial resources for the purposes of equitable and sustainable development, flows from proper public sector management, efficient fiscal and accounting mechanisms, and a willingness to make service delivery a priority in practice. . . .

Quicklinks

- Για *navigational query* όπως ***united airlines*** οι χρήστες πιθανόν να ικανοποιούνται από τη σελίδα www.united.com
- Quicklinks παρέχουν navigational cues σε αυτή τη σελίδα



86

87

Εναλλακτικές αναπαραστάσεις;

ΤΕΛΟΣ 6-7ου Μαθήματος

Ερωτήσεις?

Χρησιμοποιήθηκε κάποιο υλικό των:
- ✓*Pandu Nayak and Prabhakar Raghavan, CS276:Information Retrieval and Web Search (Stanford)*
- ✓*Hinrich Schütze and Christina Lioma, Stuttgart IIR class*