ΜΥΕ003: Ανάκτηση Πληροφορίας

Διδάσκουσα: Ευαγγελία Πιτουρά

Επεκτάσεις Ευρετηρίου. Στατιστικά Συλλογής. Συμπίεση

Ακαδημαϊκό Έτος 2024-2025

Περιεχόμενα

- Σύντομη επανάληψη
- Επεκτάσεις ανεστραμμένου ευρετηρίου
  - Skip pointers
  - Ευρετήρια θέσεων
  - Biwords
- Στατιστικά
- Συμπίεση

Τι είναι η Ανάκτηση Πληροφορίας (Information Retrieval);

Μηχανές Αναζήτησης

Φάση 1: Δημιουργία της μηχανής αναζήτησης

- Δημιουργία συλλογής (αν δεν υπάρχει):
  - crawl, scrap, use specific APIs (social media)
- Ανάλυση του κειμένου (parsing)
  - Ποιοι θα είναι οι όροι, περιλαμβάνει και γλωσσολογική επεξεργασία
- Κατασκευή ευρετηρίων (indexing)
  - Ανεστραμμένο ευρετήριο (Inverted index)
    - Λίστες καταχωρήσεων (posting lists) – μία για κάθε όρο
    - Καταχώρηση – ένα στοιχείο της λίστας
      Κάθε λίστα είναι διατεταγμένη με το DocID
  - Λεξιλόγιο (Vocabulary): το σύνολο των όρων
  - Λεξικό (Dictionary) δομή δεδομένων για τους όρους

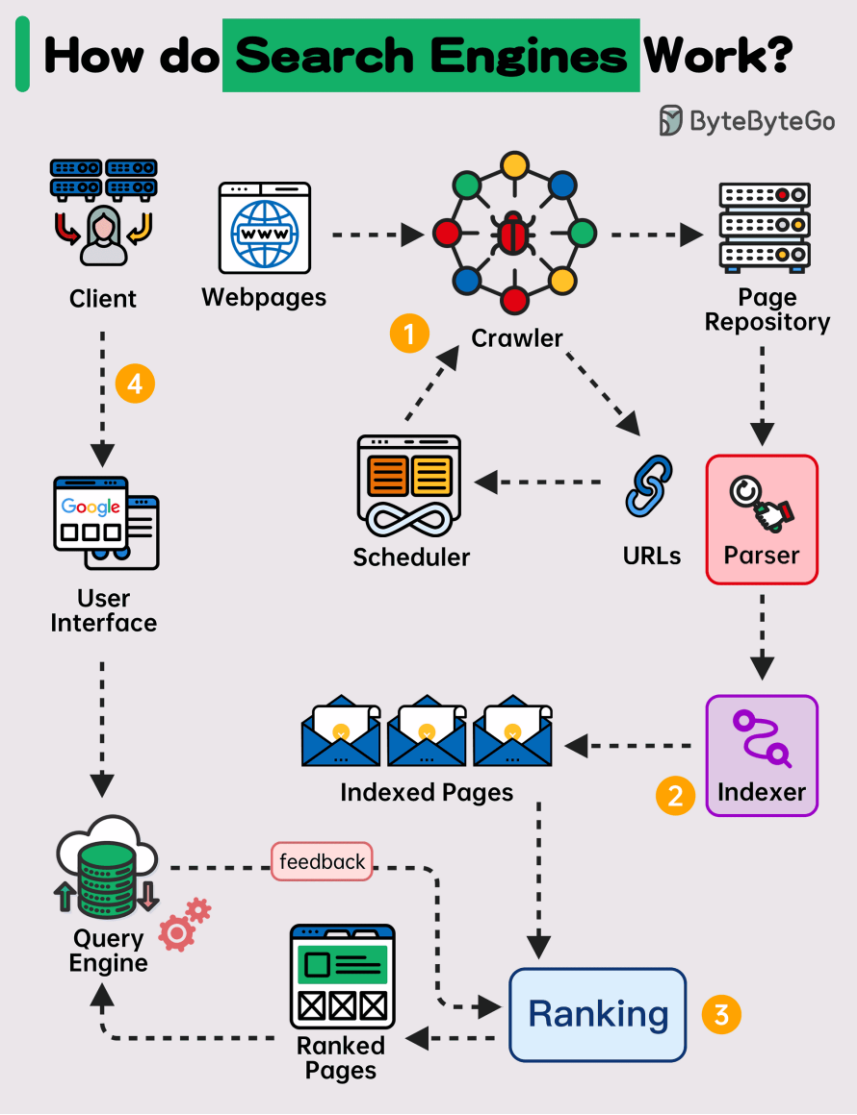
Μηχανές Αναζήτησης

Φάση 2: Λειτουργία

Ο χρήστης υποβάλει ένα ερώτημα

Επεξεργασία του ερωτήματος

Χρήση του ευρετηρίου για να βρούμε (retrieve) τα συναφή έγγραφα και να τα διατάξουμε με βάση τη συνάφεια



https://blog.bytebytego.com/p/ep104-how-do-search-engines-work

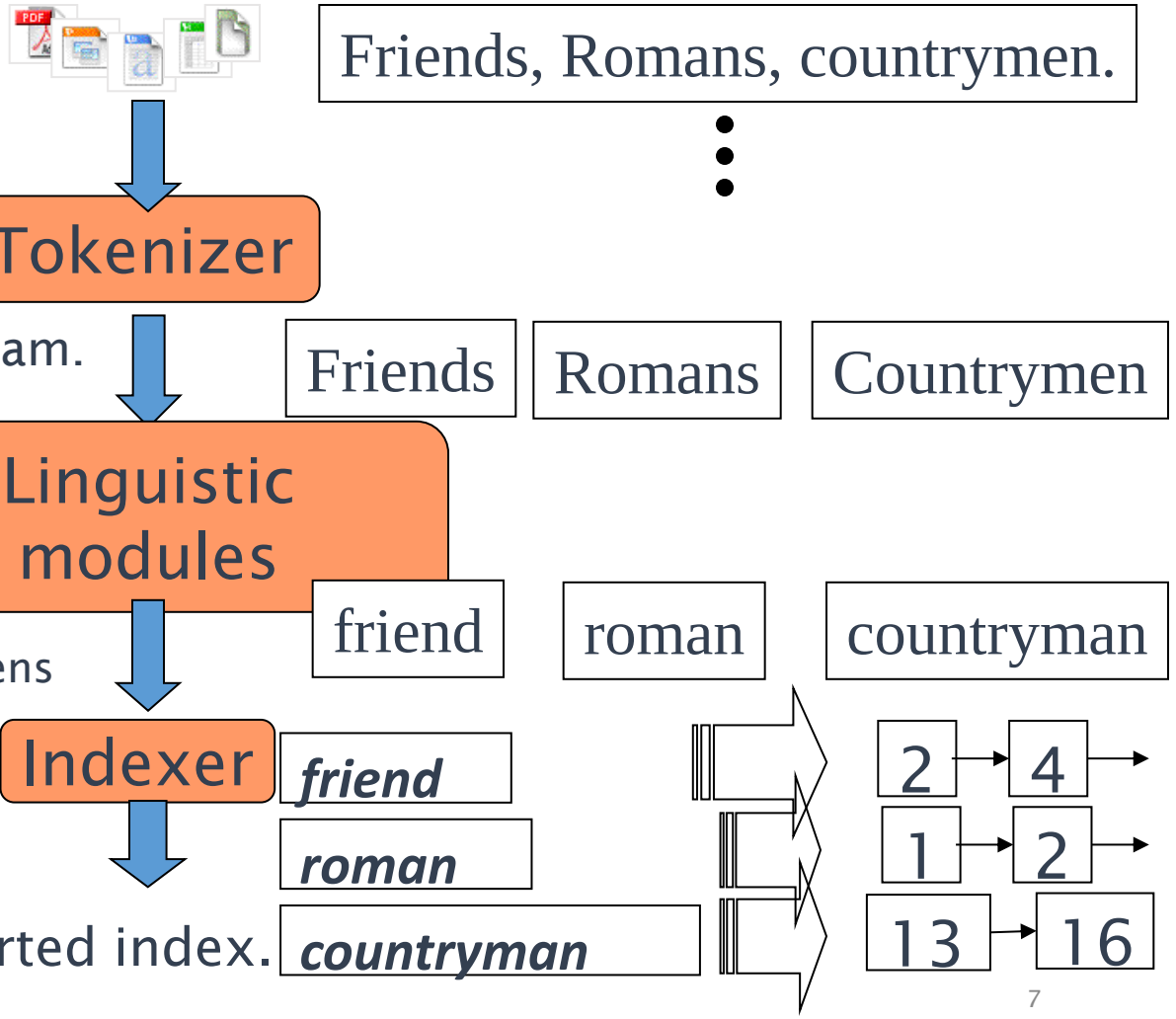
Τα βασικά βήματα ανάλυσης κειμένου

1. Συλλέγουμε τα έγγραφα που θέλουμε να συμπεριλάβουμε στο ευρετήριο

2. Διαιρούμε το κείμενο σε γλωσσικά σύμβολα **(token)**

3. Γλωσσολογική προ-επεξεργασία των συμβόλων

4. Ευρετηριάζουμε τα έγγραφα στα οποία περιλαμβάνεται κάθε όρος



Περίληψη

Ακολουθία εγγράφων

Έγγραφο 1 (d1) : Το Παν. Ιωαννίνων ιδρύθηκε το 1970.

Έγγραφο 2 (d2) : Τα Ιωάννινα είναι η μεγαλύτερη πόλη της Ηπείρου.

Έγγραφο 3 (d3) : Η πτυχιακή εξεταστική στο Τμήμα Μηχ. Η/Υ και Πληροφορικής θ΄ αρχίσει την 1η Φεβρουαρίου.

Έγγραφο 4 (d4) : Οι μαθητές των Ιωαννίνων αρίστευσαν στις εξετάσεις για την εισαγωγή στα Πανεπιστήμια.

Έγγραφο 5 (d5): Το 2017 ιδρύθηκε Πολυτεχνική Σχολή στο ΠΙ.

Granularity: Μονάδα εγγράφου

↓

Token (λεκτικές μονάδες)

Θέματα

- Που σταματάμε: κενό/σημείο στίξης αλλά και απόστροφοι/όχι κενό/παύλα, κλπ
- Stop words (το, και?)
- Κανονικοποίηση
  - Κεφαλαία/μικρά
  - Τόνοι
  - Κανόνες vs Λίστες ισοδυναμίας

↓

Όροι (terms) που θα εισαχθούν στο ευρετήριο

- **Περιστολή (stemming)** περικοπή καταλήξεων
- **Λημματοποίηση (lemmatization)** γλωσσική/μορφολογική επεξεργασία και αναγωγή της λέξης στη ρίζα της

✓ Ίδια πολιτική και στο κείμενο και στην ερώτηση

Δομές Δεδομένων για το Λεξικό

- Δυο βασικές επιλογές:
  - Πίνακες Κατακερματισμού (Hashtables)
  - Δέντρα (Trees)

- Μερικά συστήματα ανάκτησης πληροφορίας χρησιμοποιούν πίνακες κατακερματισμού άλλα δέντρα

Πίνακες Κατακερματισμού

Κάθε όρος του λεξιλογίου κατακερματίζεται σε έναν ακέραιο

+:

- Η αναζήτηση είναι πιο γρήγορη από ένα δέντρο: O(1)

- :

- Δεν υπάρχει εύκολος τρόπος να βρεθούν μικρές παραλλαγές ενός όρου
  - judgment/judgement, *resume* vs. *résumé*
- Μη δυνατή η προθεματική αναζήτηση [ανεκτική ανάκληση]
- Αν το λεξιλόγιο μεγαλώνει συνεχώς, ανάγκη για να γίνει κατακερματισμός από την αρχή

Δέντρα

- Το απλούστερο: δυαδικό δέντρο
- Το πιο συνηθισμένο: B-δέντρα
- Τα δέντρα απαιτούν ένα δεδομένο τρόπο διάταξης των χαρακτήρων (αλλά συνήθως υπάρχει ή μπορεί να οριστεί)

+:

- Λύνουν το πρόβλημα προθέματος (π.χ., όροι που αρχίζουν με *hyp*)
- Πλεονεκτούν όταν το λεξικό αποθηκεύεται στο δίσκο (τότε τα *a* και *b* καθορίζονται από το μέγεθος του block)

-:

- Πιο αργή: O(log *M*) [και αυτό απαιτεί (*ισοζυγισμένα* (*balanced)* δέντρα)
- Η επανα-ισοζύγιση (rebalancing) των δυαδικών δέντρων είναι ακριβή
  - Αλλά τα B-δέντρα καλύτερα

Περιεχόμενα

- Σύντομη επανάληψη
- Επεκτάσεις ανεστραμμένου ευρετηρίου
  - Skip pointers
  - Ευρετήρια θέσεων
  - Biwords
- Στατιστικά
- Συμπίεση

ΜΥΕ003: Ανάκτηση Πληροφορίας

Διδάσκουσα: Ευαγγελία Πιτουρά

Λίστες Καταχωρήσεων: Επεκτάσεις

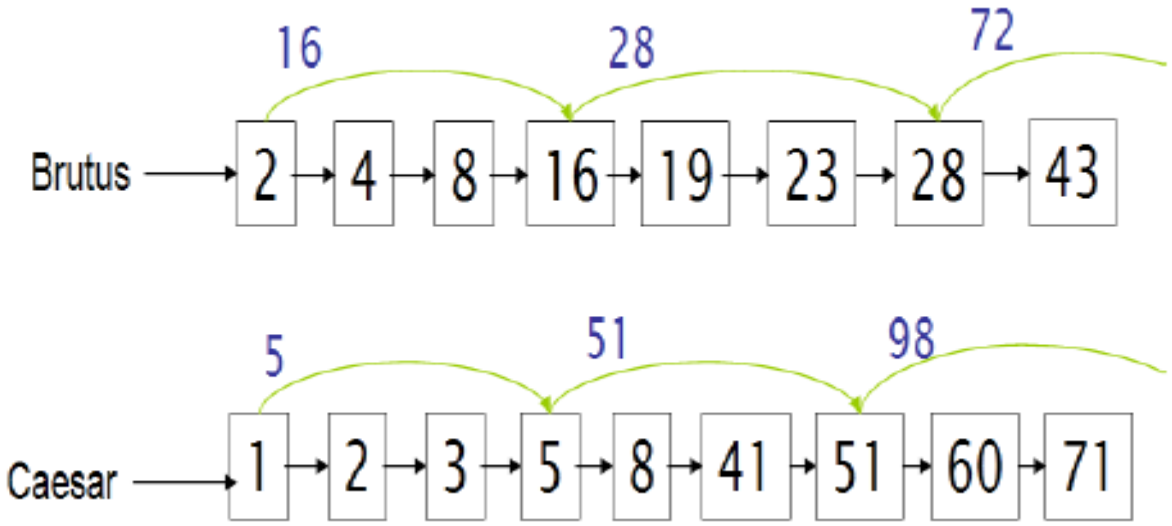
Ακαδημαϊκό Έτος 2023-2024

Τι θα δούμε σήμερα;

Ανεστραμμένο ευρετήριο

- Γρηγορότερη συγχώνευση: *Λίστες Παράβλεψης* (skip lists)
- Λίστες καταχωρήσεων *με πληροφορίες θέσεων* (*positional postings*) και ερωτήματα φράσεων (*phrase queries*)

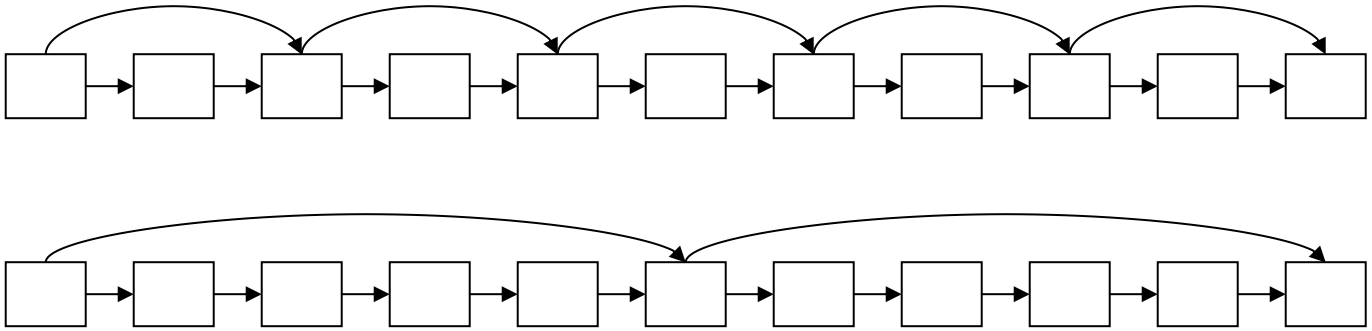
Επεξεργασία ερωτήματος με skip pointers



Αριθμός συγκρίσεων χωρίς και με χρήση δεικτών παράβλεψης

Που να τοποθετήσουμε τους δείκτες?

- Tradeoff:
  - Πολλοί δείκτες παράβλεψης $\rightarrow$ μικρότερα διαστήματα παράβλεψης $\Rightarrow$ μεγαλύτερη πιθανότητα παράβλεψης. Πολλές συγκρίσεις για να παραλείψουμε δείκτες.
  - Λιγότεροι δείκτες παράβλεψης $\rightarrow$ λιγότερες συγκρίσεις δεικτών αλλά μεγαλύτερα διαστήματα $\Rightarrow$ λίγες επιτυχημένες παραβλέψεις.



Ευρετήρια φράσεων

Ερωτήματα Φράσεων (phrase queries)

- Θέλουμε να μπορούμε να απαντάμε σε ερωτήματα όπως “***stanford university***” – ως φράση
- Οπότε το έγγραφο *“I went to university at Stanford”* δεν αποτελεί ταίριασμα.
  - Η έννοια των ερωτημάτων φράσεων έχει αποδειχθεί πολύ δημοφιλής και εύκολα κατανοητή από τους χρήστες
  - Από τις λίγες μορφές αναζήτησης πέρα της βασικής που υιοθετήθηκαν (ερωτήσεις με «» αποτελούν το 10%)
  - Ακόμα περισσότερες είναι *έμμεσα* ερωτήματα φράσεων
- Για να τα υποστηρίξουμε, δεν αρκούν εγγραφές της μορφής

  <*term* : *docs*>

Μια πρώτη προσέγγιση: Ευρετήρια ζευγών λέξεων (Biword indexes)

- Εισήγαγε στο ευρετήριο *κάθε διαδοχικό ζεύγος όρων* στο κείμενο ως φράση
- Για παράδειγμα το κείμενο “Friends, Romans, Countrymen” παράγει τα biwords
  - ***friends romans***
  - ***romans countrymen***
- Κάθε τέτοιο biword είναι τώρα ένας όρος του ευρετηρίου
- Επιτρέπει την επεξεργασία ερωτημάτων φράσεων με δύο λέξεις.

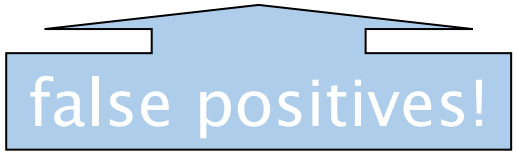
Μεγαλύτερες φράσεις

- Οι μεγαλύτερες φράσεις με κατάτμηση:

stanford university palo alto μπορεί να διασπαστεί ως ένα Boolean ερώτημα με biwords:

stanford university *AND* ***university palo*** *AND* ***palo alto***

Χωρίς να εξετάσουμε τα έγγραφα, δεν μπορούμε να εξακριβώσουμε ότι τα έγγραφα που ικανοποιούν το παραπάνω ερώτημα περιέχουν τη φράση.



Παράδειγμα

d1: a b b c
d2: b b c a
d3: b c d c
d4: a d c b
d5: a b c a b

“a b”

“a b c”

Διευρυμένα biwords

- Επεξεργασία του κειμένου και εκτέλεση part-of-speech-tagging (POST).
- Ομαδοποιούμε τους όρους σε ουσιαστικά- Nouns (N) και άρθρα/προθέσεις (X).

- Διευρυμένο biword: κάθε ακολουθία όρων της μορφής NX*N
  - Κάθε τέτοιο διευρυμένο biword είναι τώρα ένας όρος του λεξικού

Διευρυμένα biwords

- Παράδειγμα: ***catcher in the rye***

  N X X N

- Επεξεργασία ερωτήματος: χώρισε το σε N και X
  - Διαίρεσε την ερώτηση σε διευρυμένα biwords
  - Αναζήτησε στο ευρετήριο το: ***catcher rye***

- Παράδειγμα: ***cost overruns on a power plant***
  - “cost overruns” “overruns power” “power plant”

Θέματα

- False positives
- Περισσότερους από 2 όρους -> Phrase index (ευρετήριο φράσης)
- Δημιουργούνται πολύ μεγάλα λεξικά
  - Δεν είναι δυνατόν για μεγαλύτερες φράσεις από 2 λέξεις, μεγάλα ακόμα και για αυτές
- Τα ευρετήρια biword δεν είναι η συνήθης λύση (για όλα τα biwords) αλλά χρησιμοποιούνται *ως μέρος πιο σύνθετων λύσεων*

Ευρετήρια θέσεων (positional indexes)

Positional indexes (Ευρετήρια Θέσεων)

- Στις καταχωρήσεις, με κάθε όρο, αποθηκεύουμε και τη θέση (θέσεις) όπου εμφανίζονται τα tokens του:

<***term***, number of docs containing ***term***;
doc1: position1, position2 … ;
doc2: position1, position2 … ;
etc.>

Παράδειγμα

<***be***: 993427;
1: 7, 18, 33, 72, 86, 231;
2: 3, 149;
4: 17, 191, 291, 430, 434;
5: 363, 367, …>

Επεξεργασία ερωτήματος φράσης

- Βρες τις εγγραφές του ευρετηρίου για τους όρους του ερωτήματος
- Αλλά τώρα δεν αρκεί η ισότητα των doc id
- Συγχώνευσε τις doc: position λίστες για απαρίθμηση όλων των πιθανών θέσεων
- Δύο συγχωνεύσεις: όρο και θέση

Παράδειγμα

“a b”

d1: a b b c

d2: b b c a “a b c”

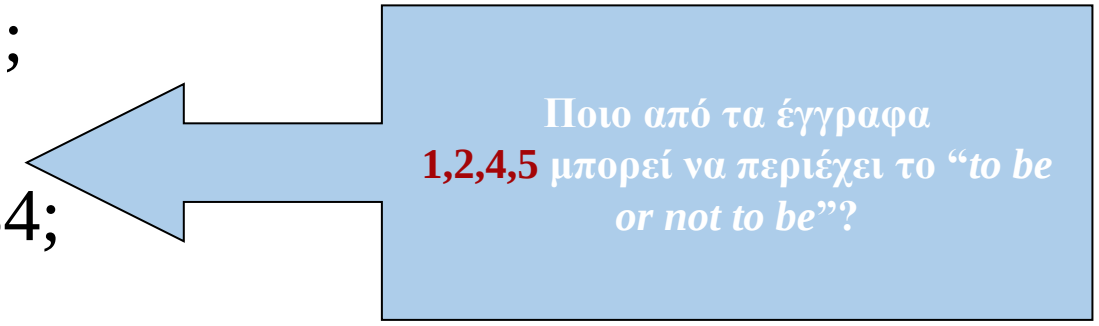
d3: b c d c

d4: a d c b

d5: a b c a b

Παράδειγμα

<***be***: 993427;
1: 7, 18, 33, 72, 86, 231;
2: 3, 149;
4: 17, 191, 291, 430, 434;
5: 363, 367, …>



Επεξεργασία ερωτήματος φράσης

Παράδειγμα ερωτήματος: *“to*$_1$ *be*$_2$ *or*$_3$ *not*$_4$ *to*$_5$ *be*$_6$*”*

TO, 993427:

‹ **1**: ‹7, 18, 33, 72, 86, 231›; **2**: ‹1, 17, 74, 222, 255›; **4**: ‹8, 16, 190, 429, 433›; 5: ‹363, 367›; 7: ‹13, 23, 191›; . . . ›

BE, 178239:

‹ **1**: ‹17, 25›; **4**: ‹17, 191, 291, 430, 434›; **5**: ‹14, 19, 101›; . . . ›

Ερωτήματα γειτονικότητας (Proximity queries)

- Η ίδια γενική μέθοδος για ερωτήματα γειτονικότητας (proximity searches)

- LIMIT! /3 STATUTE /3 FEDERAL /2 TORT
  - Πάλι, */k* means “within *k* words of”.
- Μπορούμε να χρησιμοποιήσουμε ευρετήρια θέσεων αλλά όχι ευρετήρια biword.

Πολυπλοκότητα ερώτησης

- Αυξάνει την πολυπλοκότητα της ερώτησης από O(T), T αριθμός εγγράφων σε O(N), N αριθμός token.

Μέγεθος ευρετηρίου

- Μπορούμε να συμπιέσουμε τα position values/offsets
- Παρόλα αυτά, σημαντική αύξηση του χώρου αποθήκευσης των λιστών καταχωρήσεων
- Αλλά χρησιμοποιείται ευρέως

Η σχετική θέση των όρων χρησιμοποιείται και εμμέσως για την κατάταξη των αποτελεσμάτων.

Μέγεθος ευρετηρίου

- Χρειάζεται μια εγγραφή για κάθε εμφάνιση στο έγγραφο αντί για μια για κάθε έγγραφο
- Το *μέγεθος του ευρετηρίου* εξαρτάται από το *μέσο μέγεθος του αρχείου*
  - Μέσο μέγεθος web σελίδας < 1000 όροι
  - SEC filings, books, ακόμα και μερικά επικά ποιήματα … πάνω από 100,000 όρους
- Έστω ένας όρος με συχνότητα 0.01% (1 ανά 1000 όρους) σε ένα έγγραφο

Document size	Postings	Positional postings
1000	1	1
100,000	1	?

? Α 1
Β 100
C 1000

Rules of thumb

- Ένα ευρετήριο θέσεων είναι 2–4 μεγαλύτερο από ένα απλό ευρετήριο
- Το μέγεθος του *συμπιεσμένου* ευρετηρίου είναι το 35–50% του όγκου του αρχικού κειμένου
- Αυτά αφορούν την Αγγλική (και παρόμοιες) γλώσσες

Συνδυαστικές μέθοδοι

- Αυτές οι δυο προσεγγίσεις μπορεί να συνδυαστούν
  - Για συγκεκριμένες φράσεις (***"Michael Jackson", "Britney Spears"***) η συνεχής συγχώνευση καταχωρήσεων ευρετηρίου θέσεων δεν είναι αποδοτική
    - Ακόμα περισσότερο για φράσεις όπως ***"The Who"***

Πότε biwords αντί για positional indexes?

- Αυτά που συναντώνται συχνά
- Τις ποιο «ακριβές»

Περιεχόμενα

- Σύντομη επανάληψη
- Επεκτάσεις ανεστραμμένου ευρετηρίου
  - Skip pointers
  - Ευρετήρια θέσεων
  - Biwords
- Στατιστικά
- Συμπίεση

ΜΥΕ003: Ανάκτηση Πληροφορίας

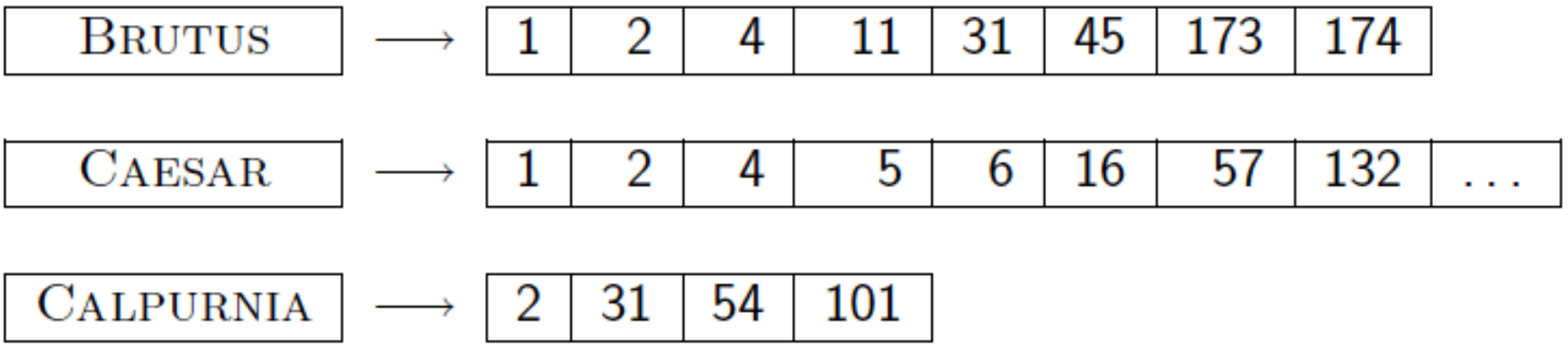
Διδάσκουσα: Ευαγγελία Πιτουρά

Κεφάλαιο 5: Στατιστικά Συλλογής. Συμπίεση.

Ακαδημαϊκό Έτος 2024-2025

Στατιστικά

- Πόσο μεγάλο είναι το λεξικό και οι καταχωρήσεις;



Η συλλογή RCV1

- Μερικά στατιστικά

- Θα χρησιμοποιήσουμε τη συλλογή RCV1.
  - Είναι ένας χρόνος του κυκλώματος ειδήσεων του Reuters (Reuters newswire) (μέρος του 1995 και 1996)
  - 1GB κειμένου

- Δεν είναι πολύ μεγάλη, αλλά είναι διαθέσιμη στο κοινό.

Ένα έγγραφο της συλλογής Reuters RCV1

REUTERS

You are here: Home > News > Science > Article

Go to a Section: U.S. International Business Markets Politics Entertainment Technology Sports Oddly Enoug

Extreme conditions create rare Antarctic clouds

Tue Aug 1, 2006 3:20am ET

Email This Article | Print This Article | Reprints

[-] Text [+]

SYDNEY (Reuters) - Rare, mother-of-pearl colored clouds caused by extreme weather conditions above Antarctica are a possible indication of global warming, Australian scientists said on Tuesday.

Known as nacreous clouds, the spectacular formations showing delicate wisps of colors were photographed in the sky over an Australian meteorological base at Mawson Station on July 25.

Η συλλογή RCV1: στατιστικά

N	documents	800,000
L	tokens per document	200
M	terms (= word types)	400,000
	bytes per token (incl. spaces/punct.)	6
	bytes per token (without spaces/punct.)	4.5
	bytes per term (= word type)	7.5
T	non-positional postings	100,000,000

- Γιατί κατά μέσο ένα term είναι μεγαλύτερο από ένα token;
- Πόσες είναι οι λίστες καταχωρήσεων;
- Πόσο μεγάλος είναι ο πίνακας;
- Πόσες μη μηδενικές τιμές στον πίνακα;

Μέγεθος ευρετηρίου

size of1.	word types (terms)			non-positional postings			positional postings		
	dictionary			non-positional index			positional index		
	Size (K)	Δ%	cumul %	Size (K)	Δ %	cumul %	Size (K)	Δ %	cumul %
Unfiltered	484								
A. No numbers									
B. Case folding									
C. 30 stopwords									
D. 150 stopwords									
E. stemming									

Ποια από τα παραπάνω πιστεύετε ότι θα προκαλεί τη μεγαλύτερη μείωση στο μέγεθος του *λεξικού*?

Μέγεθος ευρετηρίου

size of	word types (terms)			non-positional postings			positional postings		
	dictionary			non-positional index			positional index		
	Size (K)	Δ%	cumul %	Size (K)	Δ %	cumul %	Size (K)	Δ %	cumul %
Unfiltered	484			109,971					
A. No numbers	474	-2	-2						
B. Case folding	392	-17	-19						
C. 30 stopwords	391	-0	-19						
D. 150 stopwords	391	-0	-19						
E. stemming	322	-17	-33						

Ποια από τα παραπάνω πιστεύετε ότι θα προκαλεί τη μεγαλύτερη μείωση στο μέγεθος του *ανεστραμμένου ευρετηρίου*?
(συνολικός αριθμός καταχωρήσεων)

Μέγεθος ευρετηρίου

size of	word types (terms)			non-positional postings			positional postings		
	dictionary			non-positional index			positional index		
	Size (K)	Δ%	cumul %	Size (K)	Δ %	cumul %			
Unfiltered	484			109,971			197,879		
A. No numbers	474	-2	-2	100,680	-8	-8			
B. Case folding	392	-17	-19	96,969	-3	-12			
C. 30 stopwords	391	-0	-19	83,390	-14	-24			
D. 150 stopwords	391	-0	-19	67,002	-30	-39			
E. stemming	322	-17	-33	63,812	-4	-42			

Το *stemming* μειώνει το μέγεθος του *positional index*

A. Σωστό
B. Λάθος

Μέγεθος ευρετηρίου

size of	word types (terms)			non-positional postings			positional postings		
	dictionary			non-positional index			positional index		
	Size (K)	Δ%	cumul %	Size (K)	Δ %	cumul %	Size (K)	Δ %	cumul %
Unfiltered	484			109,971			197,879		
A. No numbers	474	-2	-2	100,680	-8	-8	179,158	-9	-9
B. Case folding	392	-17	-19	96,969	-3	-12	179,158	0	-9
C. 30 stopwords	391	-0	-19	83,390	-14	-24	121,858	-31	-38
D. 150 stopwords	391	-0	-19	67,002	-30	-39	94,517	-47	-52
E. stemming	322	-17	-33	63,812	-4	-42	94,517	0	-52

Γιατί 0;

1 12 16 75

d13: windows … window … windows …. windows

No stemming

window → … d13

windows → … d13

Stemming

window → … d13

No stemming

window → … d13: 1, 12

windows → … d13: 16, 75

Stemming

window → … d13: 1, 12, 16, 75

Μέγεθος ευρετηρίου

size of	word types (terms)			non-positional postings			positional postings		
	dictionary			non-positional index			positional index		
	Size (K)	Δ%	cumul %	Size (K)	Δ %	cumul %			
Unfiltered	484			109,971			197,879		
A. No numbers	474	-2	-2	100,680	-8	-8			
B. Case folding	392	-17	-19	96,969	-3	-12			
C. 30 stopwords	391	-0	-19	83,390	-14	-24			
D. 150 stopwords	391	-0	-19	67,002	-30	-39			
E. stemming	322	-17	-33	63,812	-4	-42			

Το κέρδος από την αφαίρεση των *stopwords* πολύ μεγαλύτερο, στα positional από ότι στα *non-positional*

Α. Σωστό

Β. Λάθος

Μέγεθος ευρετηρίου

size of	word types (terms)			non-positional postings			positional postings		
	dictionary			non-positional index			positional index		
	Size (K)	Δ%	cumul %	Size (K)	Δ %	cumul %	Size (K)	Δ %	cumul %
Unfiltered	484			109,971			197,879		
A. No numbers	474	-2	-2	100,680	-8	-8	179,158	-9	-9
B. Case folding	392	-17	-19	96,969	-3	-12	179,158	0	-9
C. 30 stopwords	391	-0	-19	83,390	-14	-24	121,858	-31	-38
D. 150 stopwords	391	-0	-19	67,002	-30	-39	94,517	-47	-52
E. stemming	322	-17	-33	63,812	-4	-42	94,517	0	-52

Μέγεθος ευρετηρίου

size of	word types (terms)			non-positional postings			positional postings		
	dictionary			non-positional index			positional index		
	Size (K)	Δ%	cumul %	Size (K)	Δ %	cumul %	Size (K)	Δ %	cumul %
Unfiltered	484			109,971			197,879		
No numbers	474	-2	-2	100,680	-8	-8	179,158	-9	-9
Case folding	392	-17	-19	96,969	-3	-12	179,158	0	-9
30 stopwords	391	-0	-19	83,390	-14	-24	121,858	-31	-38
150 stopwords	391	-0	-19	67,002	-30	-39	94,517	-47	-52
stemming	322	-17	-33	63,812	-4	-42	94,517	0	-52

Λεξιλόγιο και μέγεθος συλλογής

Νόμος του Heaps:

$$M = k\,T^b$$

M είναι το μέγεθος του λεξιλογίου (αριθμός όρων, terms), T ο αριθμός των tokens στη συλλογή

- περιγράφει πόσο μεγαλώνει το λεξιλόγιο όσο μεγαλώνει η συλλογή (το συνολικό μήκος των εγγράφων)

- Συνήθης τιμές: $30 \leq k \leq 100$ (εξαρτάται από το είδος της συλλογής) και $b \approx 0.5$

Λεξιλόγιο και μέγεθος συλλογής

Ο νόμος του Heaps απαντά πχ στο:
Πόσο περιμένω να μεγαλώσει το λεξικό, δηλαδή, πόσοι περιμένω να είναι οι νέοι όροι στο καινούργιο έγγραφο

Νόμος του Heaps:

$$M = k\,T^b$$

M είναι το μέγεθος του λεξιλογίου (αριθμός όρων - terms), T ο αριθμός των tokens στη συλλογή

- Συνήθης τιμές: $30 \leq k \leq 100$ (εξαρτάται από το είδος της συλλογής) και $b \approx 0.5$

Παράδειγμα: Έχω μια συλλογή με έγγραφα που το καθένα έχει περίπου *1000 λέξεις (tokens)*. Έστω ότι έχω *500* έγγραφα και έρχεται ακόμα 1 έγγραφο. Έστω b = 0.5

Πριν T = 500,000 tokens άρα M = k*707.1 όροι
Μετά T’ = 501,000 tokens άρα M’ = k*707.8 όροι

Λεξιλόγιο και μέγεθος συλλογής

Ο **νόμος του Heaps**:

$$M = k\,T^b$$

M είναι το μέγεθος του λεξιλογίου (αριθμός όρων), T ο αριθμός των tokens στη συλλογή

- Σε log-log plot του μεγέθους Μ του λεξιλογίου με το Τ, ο νόμος προβλέπει γραμμή με κλίση b

$$\log(M) = \log(k) + b\log(T)$$

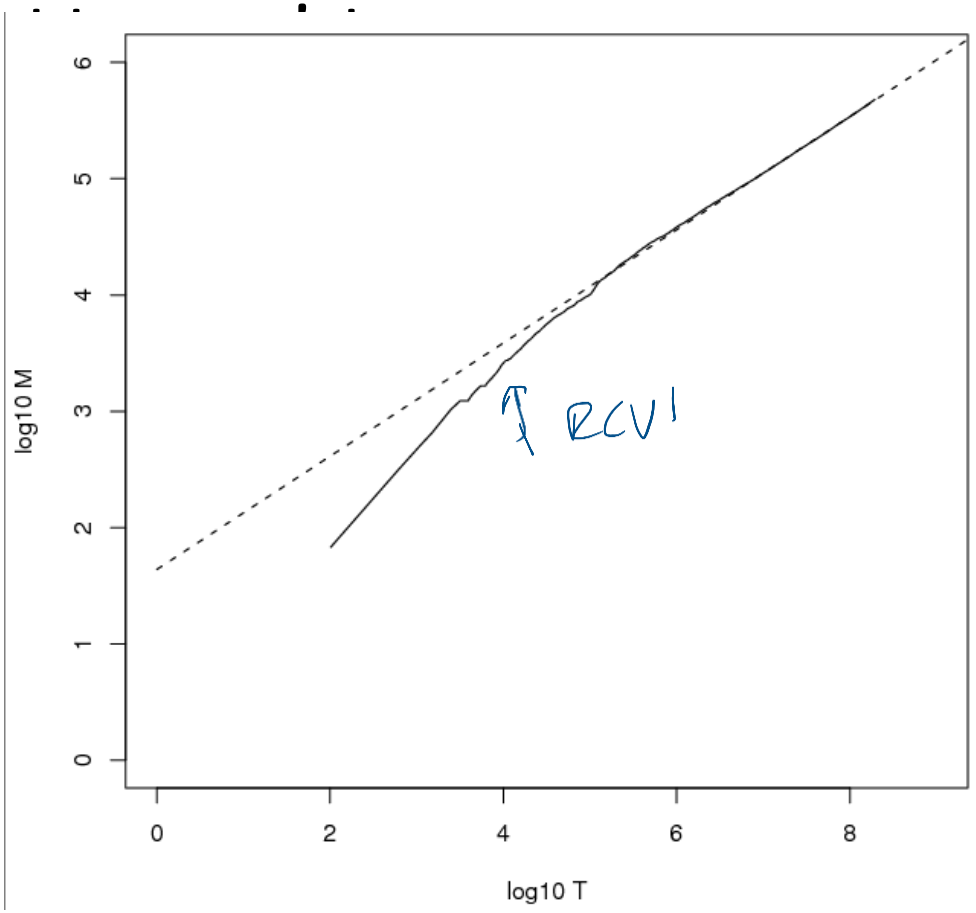
Για το RCV1, η διακεκομμένη γραμμή

$\log_{10} M = 0.49 \log_{10} T + 1.64$ (best least squares fit)

Οπότε, $M = 10^{1.64} T^{0.49}$, άρα $k = 10^{1.64} \approx 44$ και $b = 0.49$.

Καλή προσέγγιση για το Reuters RCV1!

Για το πρώτα 1,000,020 tokens, ο νόμος προβλέπει 38,323 όρους, στην πραγματικότητα 38,365



Λεξιλόγιο και μέγεθος συλλογής

- Diminishing returns: μπορούμε γρήγορα να καλύψουμε μέρος του λεξιλογίου, αλλά γίνεται όλο και πιο δύσκολο να το καλύψουμε όλο

- Σε log-log plot του μεγέθους M του λεξιλογίου με το T, ο νόμος προβλέπει γραμμή με κλίση περίπου ½

Ο νόμος του Heaps

Τα παρακάτω επηρεάζουν το μέγεθος του λεξικού (και την παράμετρο k):

- Stemming
- Including numbers
- Spelling errors
- Case folding

Ο νόμος του Zipf

- Ο νόμος του Heaps μας δίνει το μέγεθος του λεξιλογίου μιας συλλογής (σε συνάρτηση του μεγέθους της συλλογής)
- Θα εξετάσουμε τη *σχετική συχνότητα* των όρων
- Στις φυσικές γλώσσες, υπάρχουν λίγοι πολύ συχνοί όροι και πάρα πολύ σπάνιοι

Ο νόμος του Zipf

Ο **νόμος του Zipf:** Ο i-οστός πιο συχνός όρος έχει συχνότητα ανάλογη του 1/*i* .

$$cf_i \propto 1/i$$

$$cf_i = m\, i^{-k}$$

cf_i collection frequency: ο αριθμός εμφανίσεων του όρου t_i στη συλλογή (διαφορετικό από το df: document frequency)

Η συχνότητα εμφάνισης ενός όρου είναι αντιστρόφως ανάλογη της θέσης του στη διάταξη με βάση τις συχνότητες

- Για k = 1
- Αν ο πιο συχνός όρος (ο όρος *the*) εμφανίζεται cf_1 φορές
- Τότε ο δεύτερος πιο συχνός (*of*) εμφανίζεται $cf_1/2$ φορές
- Ο τρίτος (*and*) $cf_1/3$ φορές
- …

Ο νόμος του Zipf

Η συχνότητα εμφάνισης του *i*-οστού όρου:

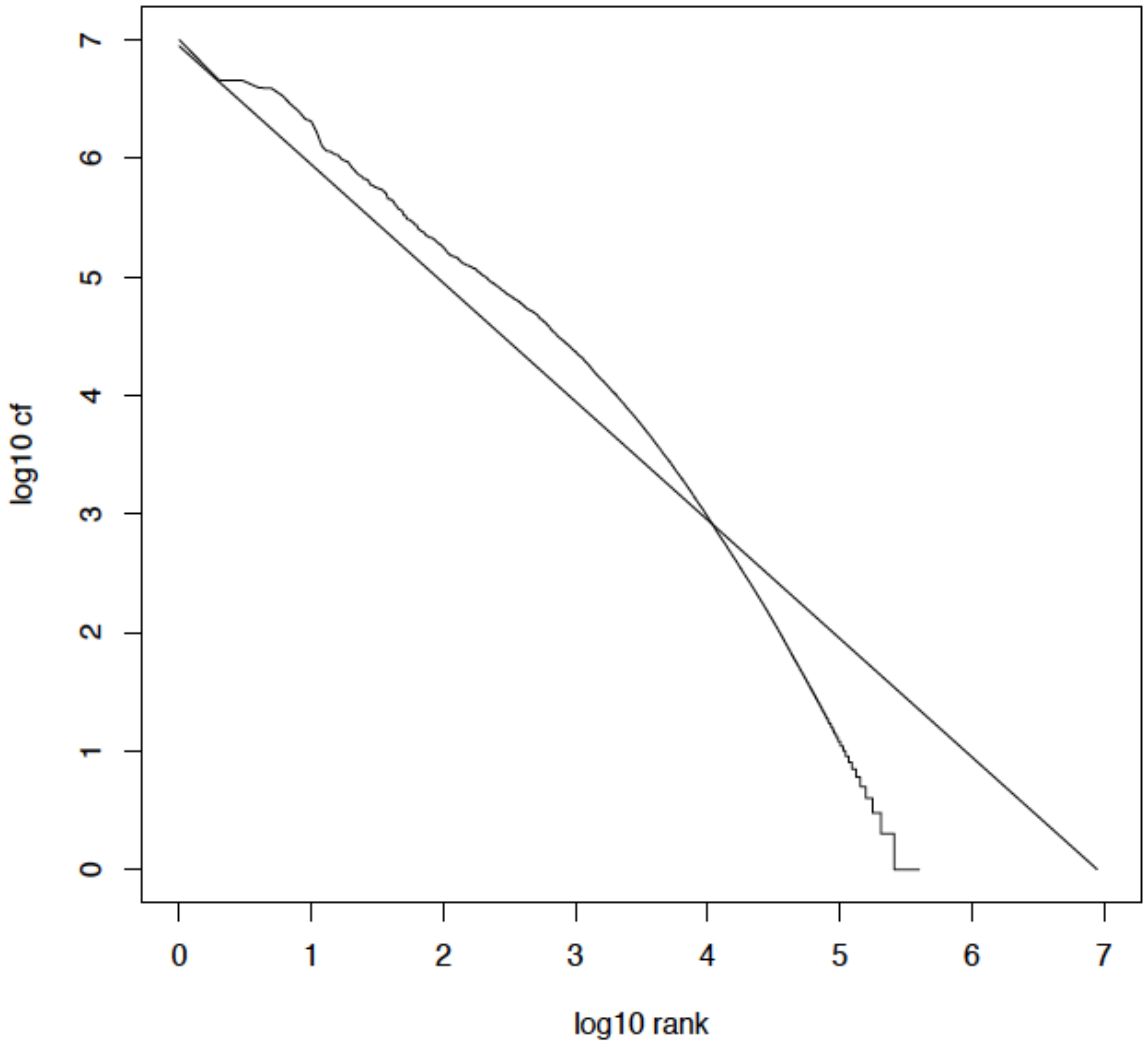
$$cf_i = mi^{-k}$$
$$\log cf_i = \log m - k \log i$$

- Γραμμική σχέση μεταξύ log cf_i και log *i*

$cf_i = mi^{-k}$, k = 1

power law σχέση (εκθετικός νόμος)

Zipf's law for Reuters RCV1



ΣΥΜΠΙΕΣΗ

Συμπίεση

- Θα δούμε μερικά θέματα για τη συμπίεση το *λεξικού* και των *λιστών καταχωρήσεων*
- Βασικό Boolean ανεστραμμένο ευρετήριο, χωρίς πληροφορία θέσης κλπ

Γιατί συμπίεση;

- Λιγότερος *χώρος στη μνήμη*
  - Λίγο πιο οικονομικό
- Κρατάμε περισσότερα πράγματα στη μνήμη
  - Αύξηση της ταχύτητας
- Αύξηση της ταχύτητας μεταφοράς δεδομένων από το δίσκο στη μνήμη
  - [διάβασε τα συμπιεσμένα δεδομένα| αποσυμπίεσε] γρηγορότερο από [διάβασε μη συμπιεσμένα δεδομένα]
  - Προϋπόθεση: Γρήγοροι αλγόριθμοι αποσυμπίεσης

Απωλεστική και μη συμπίεση

- Lossless compression: (μη απωλεστική συμπίεση) Διατηρείτε όλη η πληροφορία
  - Αυτή που κυρίως χρησιμοποιείται σε ΑΠ
- Lossy compression: (απωλεστική συμπίεση) Κάποια πληροφορία χάνεται
  - Πολλά από τα *βήματα προ-επεξεργασίας* (μετατροπή σε μικρά, stop words, stemming, number elimination) μπορεί να θεωρηθούν ως απωλεστική συμπίεση
  - Μπορεί να είναι αποδεκτή στην περίπτωση π.χ., που μας ενδιαφέρουν μόνο τα κορυφαία από τα σχετικά έγγραφα

Περιεχόμενα

- Σύντομη επανάληψη
- Επεκτάσεις ανεστραμμένου ευρετηρίου
  - Skip pointers
  - Ευρετήρια θέσεων
  - Biwords
- Στατιστικά
- Συμπίεση

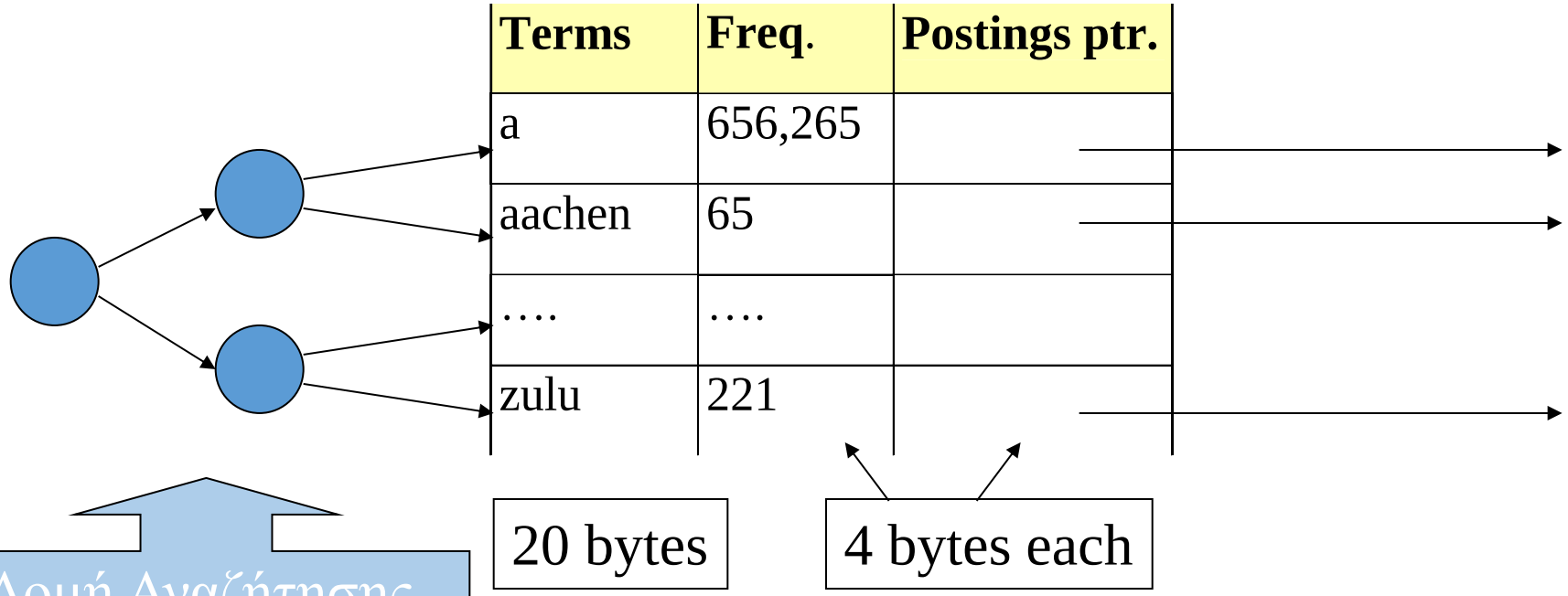
ΣΥΜΠΙΕΣΗ ΛΕΞΙΚΟΥ

Συμπίεση λεξικού

- Η αναζήτηση αρχίζει από το λεξικό -> Θα θέλαμε να το κρατάμε στη μνήμη
- Συνυπάρχει με άλλες εφαρμογές (memory footprint competition)
- Κινητές/ενσωματωμένες συσκευές μικρή μνήμη
- Ακόμα και αν όχι στη μνήμη, θα θέλαμε να είναι μικρό για γρήγορη αρχή της αναζήτησης

Αποθήκευση λεξικού

- Κάθε εγγραφή: τον όρο, συχνότητα εμφάνισης, δείκτη
- Θα θεωρήσουμε *την πιο απλή αποθήκευση*, ως ταξινομημένο πίνακα εγγραφών σταθερού μεγέθους (array of fixed-width entries)
  - ~400,000 όροι; 28 bytes/term = 11.2 MB.



(20+4+4) 400,000=11,2MB

Θα την αγνοήσουμε

4 bytes pointers -> 4GB address space (more bytes may be needed for larger collections)

Αποθήκευση λεξικού

Σπατάλη χώρου

- Πολλά από τα bytes στη στήλη **Term** *δεν* χρησιμοποιούνται – δίνουμε 20 bytes για όρους με 1 χαρακτήρα
  - Και δε μπορούμε να χειριστούμε το *supercalifragilisticexpialidocious* ή *hydrochlorofluorocarbons (λέξεις με πάνω από 20 χαρακτήρες)*
- Μέσος όρος των λέξεων στο λεξικό για τα Αγγλικά: ~8 χαρακτήρες

Θα μειώσουμε το χώρο για την αποθήκευση των terms

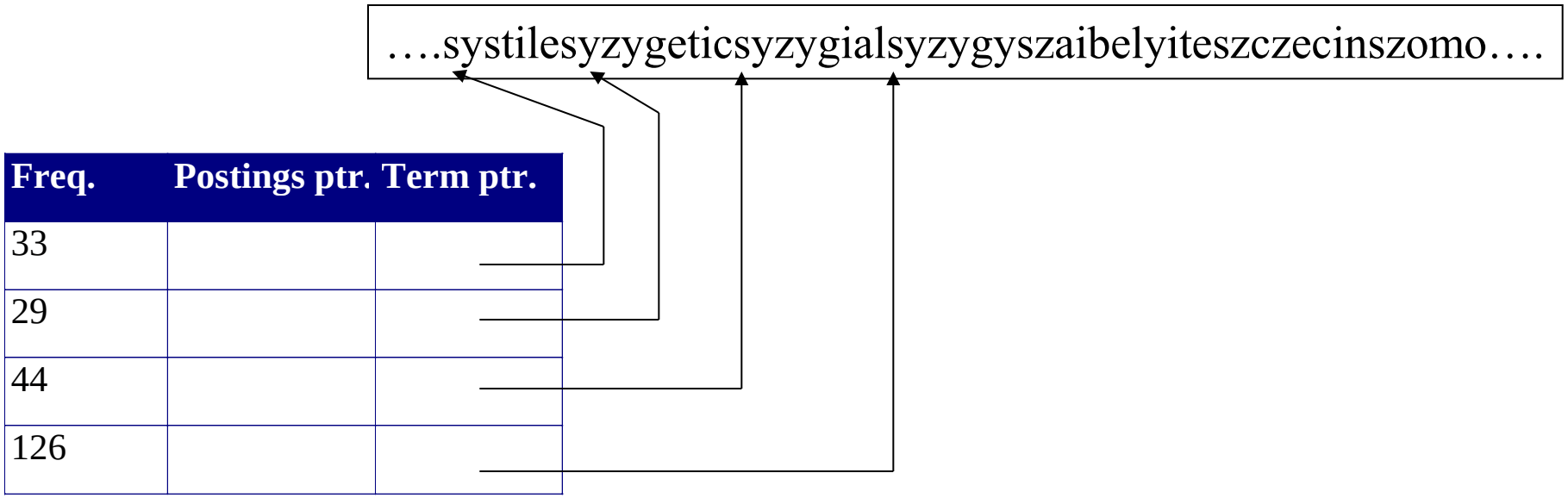
Πως;

Όλοι οι όροι σε ένα μεγάλο string

Συμπίεση της λίστας όρων: Λεξικό-ως-Σειρά-Χαρακτήρων

Αποθήκευσε το λεξικό ως ένα (μεγάλο) string χαρακτήρων:

- Ένας δείκτης δείχνει στο τέλος της τρέχουσας λέξης (αρχή επόμενης)



δυαδική αναζήτηση όπως πριν, τώρα στο string

Θα μειώσουμε το χώρο για την αποθήκευση των terms

Αρχικά

20 per term

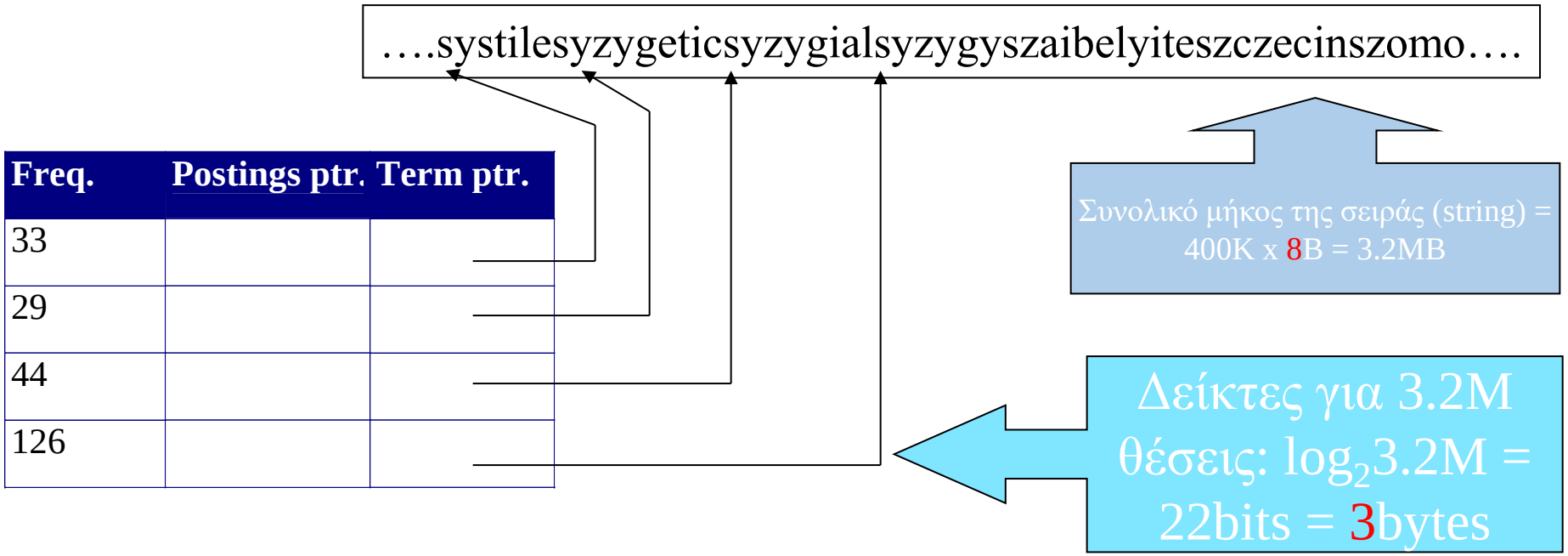
Σύνολο: 20 x 400,000

Τώρα

Ένα μεγάλο string +

Ένα δείκτη per term (που δείχνει στη θέση του στο μεγάλο string)

Συμπίεση της λίστας όρων: Λεξικό-ως-Σειρά-Χαρακτήρων



δυαδική αναζήτηση όπως πριν, τώρα στο string

Χώρος για το λεξικό ως string

- 4 bytes ανά όρο για το Freq.
- 4 bytes ανά όρο για δείκτες σε Postings.
- 3 bytes ανά term pointer

} 11

- Κατά μέσο όρο 8 bytes ανά όρο στο string (3.2MB)

- 400K όροι x 19 ⇒ 7.6 MB (έναντι 11.2MB για σταθερό μήκος λέξης)

11+8

  - Εξοικονόμηση 60% του χώρου

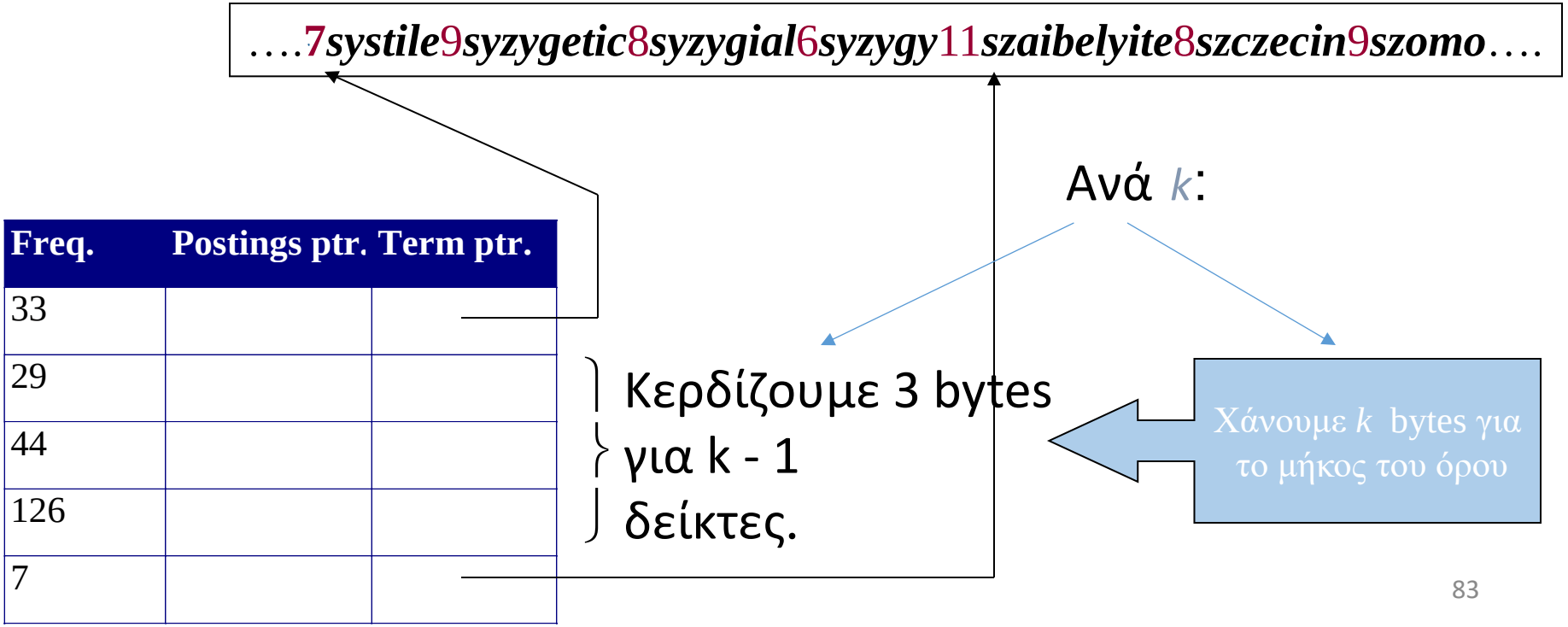
Θα μειώσουμε το χώρο για την αποθήκευση των terms

Πως;
Όλοι οι όροι σε ένα μεγάλο string +

Ένας δείκτης στο string ανά ***k*** όρους

Blocking (Δείκτες σε ομάδες)

- Διαίρεσε το string σε ομάδες (blocks) των *k* όρων
- Διατήρησε ένα δείκτη σε κάθε ομάδα
  - Παράδειγμα: *k* = 4.
- Χρειαζόμαστε και το μήκος του όρου (1 extra byte)



Blocking

Συνολικό όφελος για block size $k = 4$

- Χωρίς blocking 3 bytes/pointer
  - 3 x 4 = 12 bytes, (ανά block)

Τώρα 3 + 4 = 7 bytes.

Εξοικονόμηση ακόμα ~0.5MB. Ελάττωση του μεγέθους του ευρετηρίου από 7.6 MB σε 7.1 MB.

- Γιατί όχι ακόμα μεγαλύτερο *k*;
- Σε τι χάνουμε;

Θα μειώσουμε το χώρο για την αποθήκευση των terms

Πως;
Όλοι οι όροι σε ένα μεγάλο string +

Ένας δείκτης στο string ανά ***k*** όρους +

Συγχώνευση των κοινών προθεμάτων

Εμπρόσθια κωδικοποίηση (Front coding)

Οι λέξεις συχνά έχουν μεγάλα κοινά προθέματα – αποθήκευση μόνο των διαφορών

8*automata*8*automate*9*automatic*10*automation*

→8*automat***a*1◊*e*2◊*ic*3◊*ion*

Encodes ***automat***

Extra length beyond ***automat.***

Περίληψη συμπίεσης για το λεξικό του RCV1

Τεχνική	Μέγεθος σε MB
Fixed width	11.2
Dictionary-as-String with pointers to every term	7.6
Also, blocking $k = 4$	7.1
Also, Blocking + front coding	5.9

ΣΥΜΠΙΕΣΗ ΤΩΝ ΚΑΤΑΧΩΡΗΣΕΩΝ

Συμπίεση των καταχωρήσεων

- Το αρχείο των καταχωρήσεων είναι *πολύ μεγαλύτερο* αυτού του λεξικού - τουλάχιστον 10 φορές.
- Βασική επιδίωξη: *αποθήκευση κάθε καταχώρησης συνοπτικά*
- Στην περίπτωση μας, μια καταχώρηση είναι το αναγνωριστικό ενός εγγράφου (docID).
  - Για τη συλλογή του Reuters (800,000 έγγραφα), μπορούμε να χρησιμοποιήσουμε 32 bits ανά docID αν έχουμε ακεραίους 4-bytes.
  - Εναλλακτικά, $\log_2$ 800,000 ≈ 20 bits ανά docID.

Η συλλογή RCV1: στατιστικά

N	documents	800,000
L	tokens per document	200
M	terms (= word types)	400,000
	bytes per token (incl. spaces/punct.)	6
	bytes per token (without spaces/punct.)	4.5
	bytes per term (= word type)	7.5
T	non-positional postings	100,000,000

Συμπίεση των καταχωρήσεων

- Μέγεθος της συλλογής
  - 800,000 (έγγραφα)×200 (token)× 6 bytes = 960 MB
- Μέγεθος του αρχείου καταχωρήσεων (ευρετηρίου)
  - 100,000,000 (καταχωρήσεις)×20/8 bytes = 250MB

Μπορούμε να το μειώσουμε;

Συμπίεση των καταχωρήσεων

- Αποθηκεύουμε τη λίστα των εγγράφων σε αύξουσα διάταξη των docID.
  - ***computer***: 33, 47, 154, 159, 202 …
- Συνέπεια: αρκεί να αποθηκεύουμε τα διάκενα (*gaps*).
  - 33, 14, 107, 5, 43 …

- Γιατί; Τα περισσότερα διάκενα μπορεί να κωδικοποιηθούν/αποθηκευτούν με πολύ λιγότερα από 20 bits.

Παράδειγμα

	encoding	postings list									
THE	docIDs	...		283042		283043		283044		283045	...
	gaps				1		1		1		...
COMPUTER	docIDs	...		283047		283154		283159		283202	...
	gaps				107		5		43		...
ARACHNOCENTRIC	docIDs	252000		500100							
	gaps	252000	248100								

Παρόμοια ιδέα και για *positional indexes* (κωδικοποίηση των κενών ανάμεσα στις θέσεις)

Συμπίεση των καταχωρήσεων

- Ένας όρος όπως ***arachnocentric*** εμφανίζεται ίσως σε ένα έγγραφο στο εκατομμύριο.
- Ένας όρος όπως ***the*** εμφανίζεται σχεδόν σε κάθε έγγραφο, άρα 20 bits/εγγραφή πολύ ακριβό

Κωδικοποίηση μεταβλητού μεγέθους (Variable length encoding)

Στόχος:

- Για το ***arachnocentric***, θα χρησιμοποιήσουμε εγγραφές ~20 bits/gap.
- Για το ***the***, θα χρησιμοποιήσουμε εγγραφές ~1 bit/gap entry.

- Αν το μέσο κενό για έναν όρο είναι *G*, θέλουμε να χρησιμοποιήσουμε εγγραφές ~$\log_2 G$ bits/gap.
- Βασική πρόκληση: κωδικοποίηση κάθε ακεραίου (gap) *με όσα λιγότερα bits* είναι απαραίτητα για αυτόν τον ακέραιο.
- Αυτό απαιτεί κωδικοποίηση μεταβλητού μεγέθους -- *variable length encoding*
- Αυτό το πετυχαίνουμε χρησιμοποιώντας σύντομους κώδικες για μικρούς αριθμούς

Κωδικοί μεταβλητών Byte (Variable Byte (VB) codes)

- Κωδικοποιούμε κάθε διάκενο με ακέραιο αριθμό από bytes
- Το πρώτο bit κάθε byte χρησιμοποιείται ως *bit συνέχισης* (continuation bit)
  - 0, αν ακολουθεί και άλλο byte
  - 1, αλλιώς (αν το τελευταίο)
    - Είναι 0 σε όλα τα bytes εκτός από το τελευταίο, όπου είναι 1
  - Χρησιμοποιείται για να σηματοδοτήσει το τελευταίο byte της κωδικοποίησης

Κωδικοί μεταβλητών Byte (Variable Byte (VB) codes)

- Ξεκίνα με ένα byte για την αποθήκευση του *G*
- Αν *G* ≤127, υπολόγισε τη δυαδική αναπαράσταση με τα 7 διαθέσιμα bits and θέσε *c* =1
- Αλλιώς, κωδικοποίησε τα 7 lower-order bits του G και χρησιμοποίησε επιπρόσθετα bytes για να κωδικοποιήσεις τα higher order bits με τον ίδιο αλγόριθμο
- Στο τέλος, θέσε το bit συνέχισης του τελευταίου byte σε 1, c = 1 και στα άλλα σε 0, *c* = 0.

5 101
10000101

824 1100111000
00000110 10111000

Παράδειγμα

docIDs	824	829	215406
gaps		5	214577
VB code	00000110 10111000	10000101	00001101 00001100 10110001

Postings stored as the byte concatenation

000001101011100010000101000011010000110010110001

Key property: VB-encoded postings are uniquely *prefix-decodable*.

For a small gap (5), VB uses a whole byte.

Άλλες κωδικοποιήσεις

- Αντί για bytes, δηλαδή 8 bits, άλλες μονάδες πχ 32 bits (words), 16 bits, 4 bits (nibbles).

Compression ratio vs speed of decompression

  - Με byte χάνουμε κάποιο χώρο αν πολύ μικρά διάκενα – nibbles καλύτερα σε αυτές τις περιπτώσεις.
  - Μικρές λέξεις, πιο περίπλοκος χειρισμός

- Οι κωδικοί VB χρησιμοποιούνται σε πολλά εμπορικά/ερευνητικά συστήματα

5 101

1101

824 1100111000

0001 0100 0111 1000

Συμπίεση του RCV1

Data structure	Size in MB
dictionary, fixed-width	11.2
dictionary, term pointers into string	7.6
with blocking, k = 4	7.1
with blocking & front coding	5.9
collection (text, xml markup etc)	3,600.0
collection (text)	960.0
Term-doc incidence matrix	40,000.0
postings, uncompressed (32-bit words)	400.0
postings, uncompressed (20 bits)	250.0
postings, variable byte encoded	116.0
postings, γ–encoded	*101.0*

Συμπεράσματα

- Μπορούμε να κατασκευάσουμε ένα ευρετήριο για Boolean ανάκτηση πολύ αποδοτικό από άποψη χώρου
- Μόνο 4% του συνολικού μεγέθους της συλλογής
- Μόνο το 10-15% του συνολικού κειμένου της συλλογής
- Βέβαια, έχουμε αγνοήσει την πληροφορία θέσης (positional indexes)
  - Η εξοικονόμηση χώρου είναι μικρότερη στην πράξη
  - Αλλά, οι τεχνικές είναι παρόμοιες – χρησιμοποίηση gaps και για τις θέσεις στο έγγραφο

ΤΕΛΟΣ 5ου Κεφαλαίου

Ερωτήσεις?

Χρησιμοποιήθηκε κάποιο υλικό των:

- ✓ *Pandu Nayak and Prabhakar Raghavan, CS276: Information Retrieval and Web Search (Stanford)*