


Indexing and Query Processing

CS 510 Winter 2007

1



What will we cover?

- Key concepts and terminology
- Inverted index structures
 - Organization, creation, maintenance
 - Compression
 - Distribution
- Answering queries with inverted indexes
 - Boolean, cosine, phrase (not probabilistic)
 - Optimizations
 - Fielded queries
- Other approaches: signatures, suffix trees

CS 510 Winter 2007

2



Some Scenarios

- Web search
 - Lots of storage and CPU
 - Frequent new documents
 - Update of existing documents
 - Lots & lots of queries
 - Absolutely current?

CS 510 Winter 2007

3



Scenarios (2)

- Company repository
 - Servers might be provisioned for collection size
 - Documents might be fairly static
 - Moderate to low query rate
- Desktop search
 - What % of your disk is documents?
 - Low query rates, but don't want to hog disk, CPU
 - Would like to be very up to date

CS 510 Winter 2007

4



Why isn't this a database problem?

- Indexing selected columns vs. indexing nearly everything
 - In both cases, want to be able to read an "entry" rapidly
- Identifying the "parts" of the "record" to index
 - words, stems, phrases
 - column values in a row

CS 510 Winter 2007

5



DB vs. IR indexing (cont.)

- Kinds of queries supported
 - $A=v$ $A<v$
 - $\{kw1, kw2, kw3\}$ "kw1 kw2 kw3"
- What's involved in producing a prefix of the answer: ranking vs. sorting
- Update patterns
 - Change a row: two index entries
 - Change a document: many index entries

CS 510 Winter 2007

6



What's a document?

- For these discussions, a document is a sequence of terms
- It has an identifier
 - Text processing has been done: parsing, punctuation, stemming, stop words taken care of

CS 510 Winter 2007

7



What do we want to speed up?

- Boolean queries
Need to know which documents contain particular words; combine lists of documents
- Cosine (or other) similarity
Finding documents in order of similarity to a query
- Phrase (or proximity) queries
Determining adjacent (or nearby) word occurrences

CS 510 Winter 2007

8



Cosine computation

Notation (Zobel & Moffat)

$f_{d,t}$ - frequency of term t in document d

f_t - number of documents containing t

N - total number of documents

$w_{q,t}$ - weight of term t in query q

$w_{d,t}$ - weight of term t in document d

W_d - weight (vector length) of d

W_q - weight (vector length) of q

CS 510 Winter 2007

9



One weight model

Used by Zobel & Moffat (not sure exactly why the fudge factors)

Main thing here is what the inputs are

$$w_{q,t} = \ln(1 + N/f_t)$$

$$w_{d,t} = 1 + \ln(f_{d,t})$$

CS 510 Winter 2007

10



Recall the cosine formula

$$W_d = (\sum_t (w_{d,t})^2)^{1/2}$$

$$W_q = (\sum_t (w_{q,t})^2)^{1/2}$$

$$S_{q,d} = (\sum_t w_{d,t} * w_{q,t}) / W_d * W_q$$

CS 510 Winter 2007

11



What can we calculate in advance?

- $f_{d,t}$, hence $w_{d,t}$, hence W_d
- Note that $w_{q,t}$ only depends on t
 So can compute query term weights in advance for all terms
 (then select the ones we need for a given query)

CS 510 Winter 2007

12

What changes on update?

- $f_{d,t}$, $w_{d,t}$, W_d for updated document d
- What about $w_{q,t}$?

Recall $w_{q,t} = \ln(1 + N/f_t)$

- Consider adding one new document
- New ratio is $(N+1)/f_t$ or $(N+1)/(f_t+1)$
- 1314/65

CS 510 Winter 2007

13

Can omit W_q factor sometimes

If we are just comparing $S_{q,d}$ values, then we can ignore W_q , since it is the same for every d .

(but perhaps not if we are combining $S_{q,d}$ with other measures)

CS 510 Winter 2007

14



Naive calculation

```

For each d in D
  S ← 0
  For each t in q
    S ← S + (wd,t * wq,t)
  Sd,q ← S/Wd*Wq

```

Note that not every query term needs to be in the document

Could modify to set $S_{d,q} \leftarrow 0$ if any query term is missing

CS 510 Winter 2007

15



Better calculation scheme

```

For each t in q
  For each d containing t
    ...

```

Will need to keep a partial result (accumulator) for multiple documents

CS 510 Winter 2007

16



Phrase queries

- Need to tell “eggs and ham” from “ham and eggs”
- Options
 - Search for terms, then post-process by checking actual documents
 - Remember positions for each term in each document
 - Index phrases or partial phrases (for example, all two-term combinations)

CS 510 Winter 2007

17



Compressing an increasing list

Group exercise: You want to represent some initial sequence of primes. How many bits using small ints? What is a more space-efficient representation?

2 3 5 7 11 13 17 19 23 29 31 37 41

CS 510 Winter 2007

18



Sample Documents

d1

Would you like them
in a house?
Would you like them
with a mouse?

d2

You may like them.
You will see.
You may like them
in a tree!

d3

I would not, could not in a tree.
Not in a car! You let me be.

d4

Say!
In the dark?
Here in the dark?
Would you, could you,
in the dark?

d5

You do not like them.
So you say.
Try them! Try them!
And you may.
Try them and you may, I say

CS 510 Winter 2007

19



Inverted List Structure

Term dictionary, with frequencies

t a be car could dark here house I in let

f_t 3 1 1 2 1 1 1 2 4 1

t like may me mouse not say see so the

f_t 3 2 1 1 2 2 1 1 1

t them tree try will with would you

f_t 3 2 1 1 1 3 5

CS 510 Winter 2007

20

Inverted List Structure (2)

Document frequencies, for each term

t: $\langle d, f_{d,t} \rangle, \dots$

a: $\langle d1, 2 \rangle, \langle d2, 1 \rangle, \langle d3, 2 \rangle$

be: $\langle d3, 1 \rangle$

...

in: $\langle d1, 1 \rangle, \langle d2, 1 \rangle, \langle d3, 2 \rangle, \langle d4, 3 \rangle$

...

you: $\langle d1, 2 \rangle, \langle d2, 3 \rangle, \langle d3, 1 \rangle, \langle d4, 2 \rangle, \langle d5, 4 \rangle$

Inverted list structure (3)

Document info: location, W_d

For example, d3

I would not, could not in a tree.

Not in a car! You let me be.

I would not could in a tree car you

1 1 3 1 2 2 1 1 1

let me be

1 1 1

Do we need the document contents?



Example similarity calculation

$q = \{\text{not}, \text{in}, \text{tree}\}$

$w_{q,t} = 1.25 \ 0.81 \ 1.25$

$W_q = (1.25^2 + 0.81^2 + 1.25^2)^{1/2}$
 ≈ 1.94

$S_{q,d3} = \frac{(1.25*2.099 + 0.81*1.693 + 1.25*1)}{4.37*1.94}$

$\approx (2.62 + 1.37 + 1.25)/2.25$

$\approx \mathbf{2.33}$

CS 510 Winter 2007

23



Indexing word positions

Might want to list all word positions
in an index

for phrase or proximity queries

For term t , let $f_{d,t} = k$

$\langle d; k; p_1, p_2, \dots, p_k \rangle$

in: $\langle d1; 1; 5 \rangle, \langle d2; 1; 12 \rangle,$

$\langle d3; 2; 6,10 \rangle, \langle d4; 3; 2,6,13 \rangle$

Might associate info with each occurrence:

In title? Capitalized? Font size?

CS 510 Winter 2007

24



Problem with word positions

Slows down search when all you need is word frequency

Could have two indexes, with and without word positions

CS 510 Winter 2007

25



Phrase index

Count of occurrences of phrase

- There are a lot of phrases
- How to choose which ones?
- One possibility: index just two-word phrases where the first word is common.

red: "red flag" "red light"
"red hot" "red square"

Then "red hot coals" is processed as
"red hot" + coals, plus position check

CS 510 Winter 2007

26