Lecture 2: Datastructures and Algorithms for Indexing

Information Retrieval
Computer Science Tripos Part II

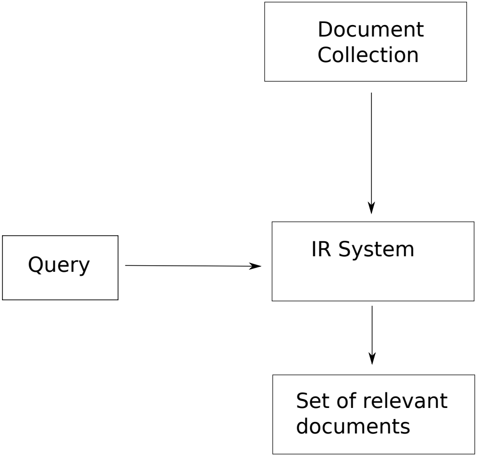
Simone Teufel

Natural Language and Information Processing (NLIP) Group
UNIVERSITY OF CAMBRIDGE

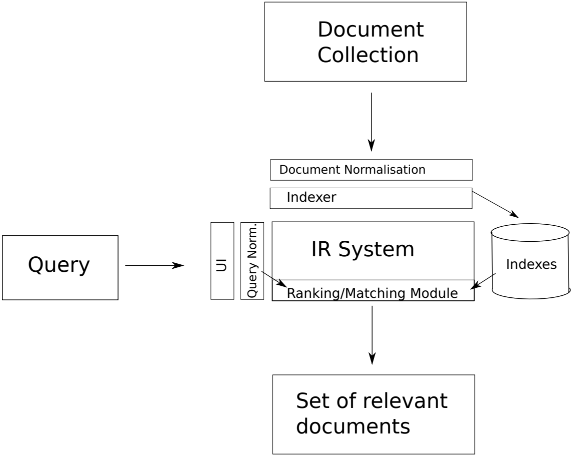
Simone.Teufel@cl.cam.ac.uk

Lent 2014

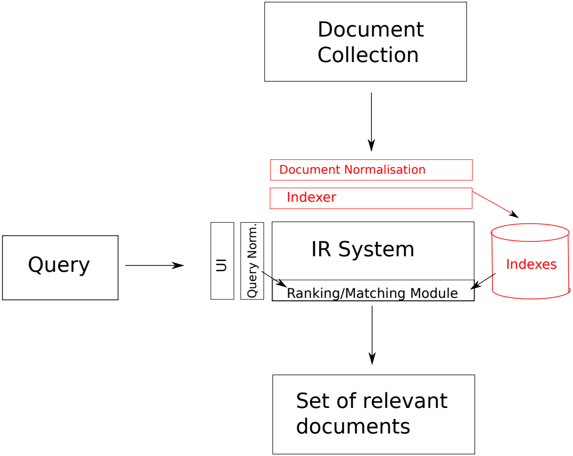
IR System Components



IR System Components



IR System Components



Today: the indexer

Overview

1. Index construction

2. Document and Term Normalisation
   - Documents
   - Terms

3. Other types of indexes
   - Biword indexes
   - Positional indexes

Index construction

Brutus → 1 → 2 → 4 → 11 → 31 → 45 → 173 → 174

Caesar → 1 → 2 → 4 → 5 → 6 → 16 → 57 → 132 → 179

Calpurnia → 2 → 31 → 54 → 101

The major steps in inverted index construction:

- Collect the documents to be indexed.
- Tokenize the text.
- Perform linguistic preprocessing of tokens.
- Index the documents that each term occurs in.

Definitions

- Word: a delimited string of characters as it appears in the text.
- Term: a "normalised" word (case, morphology, spelling etc); an equivalence class of words
- Token: an instance of a word or term occurring in a document.
- Type: an equivalence class of tokens (same as "term" in most cases)

Example: index creation by sorting

Doc 1:
I did enact Julius Caesar: I was killed i' the Capitol;Brutus killed me.

Doc 2:
So let it be with Caesar. The noble Brutus hath told you Caesar was ambitious.

$\Longrightarrow$ Tokenisation

Term	docID
I	1
did	1
enact	1
julius	1
caesar	1
I	1
was	1
killed	1
i'	1
the	1
capitol	1
brutus	1
killed	1
me	1
so	2
let	2
it	2
be	2
with	2
caesar	2
the	2
noble	2
brutus	2
hath	2
told	2
you	2
caesar	2
was	2
ambitious	2

$\Longrightarrow$ Sorting

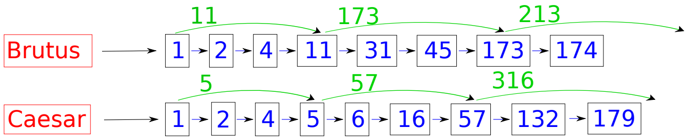
Term (sorted)	docID
ambitious	2
be	2
brutus	1
brutus	2
capitol	2
caesar	1
caesar	2
caesar	2
did	1
enact	1
hath	1
I	1
I	1
i'	1
it	2
julius	1
killed	1
killed	2
let	2
me	1
noble	2
so	2
the	1
the	2
told	2
you	2
was	1
was	1
with	2

Index creation; grouping step ("uniq")

Term	doc. freq.		Postings list
ambitious	1	→	2
be	1	→	2
brutus	2	→	1 → 2
capitol	1	→	1
caesar	2	→	1 → 2
did	1	→	1
enact	1	→	1
hath	1	→	2
I	1	→	1
i'	1	→	1
it	1	→	2
julius	1	→	1
killed	1	→	1
let	1	→	2
me	1	→	1
noble	1	→	2
so	1	→	2
the	2	→	1 → 2
told	1	→	2
you	1	→	2
was	2	→	1 → 2
with	1	→	2

- Primary sort by term (dictionary)
- Secondary sort (within postings list) by document ID
- Document frequency (= length of postings list):
  - for more efficient Boolean searching (cf. lecture 1)
  - for term weighting (lecture 4)
- keep dictionary in memory
- keep postings list (much larger) on disk

Optimisation: Skip Lists



- Some postings lists can contain several million entries
- Enter skip lists
- Check skip list if present, in order to skip multiple entries
- Tradeoff: How many skips to place?
  - More skips: each pointer skips only a few items, but we can frequently use it.
  - Fewer skips: each skip pointer skips many items, but we can not use it very often.
- Workable heuristic: place $\sqrt{L}$ skips evenly for a list of length L.
- With today's fast CPUs, skip lists don't help that much anymore.

Overview

1. Index construction
2. Document and Term Normalisation
   - Documents
   - Terms
3. Other types of indexes
   - Biword indexes
   - Positional indexes

To build an inverted index, we need to get from Input

Friends, Romans, countrymen. So let it be with Caesar...

to Output

friend | roman | countryman | so

- Each token is a candidate for a postings entry.
- What are valid tokens to emit?

Parsing a document

- Up to now, we assumed that
  - We know what a document is
  - We can easily "machine-read" each document
- We need do deal with format and language of each document
  - Format could be excel, latex, HTML ...
  - Document could be compressed or in binary format (excel, word)
  - Character set could be Unicode, UTF-8, Big-5, XML (`&amp`)
  - Language could be French email with Spanish quote or attachment
- Each of these is a statistical classification problem
- Alternatively we can use heuristics

Format/Language: Complications

- A single index usually contains terms of several languages.
- Documents or their components can contain multiple languages
- What is the document unit for indexing?
  - a file?
  - an email?
  - an email with 5 attachments?
  - an email thread?
- Also might have to deal with XML/hierarchies of HTML documents etc.
- Answering the question "What is a document?" is not trivial.
- Smaller units raise precision, drop recall

Normalisation

- Need to normalise words in the indexed text as well as query terms to the same form
- Example: We want to match U.S.A. to USA
- We most commonly implicitly define equivalence classes of terms.
- Alternatively, we could do asymmetric expansion:

window $\rightarrow$ window, windows
windows $\rightarrow$ Windows, windows, window
Windows $\rightarrow$ Windows

- Either at query time, or at index time
- More powerful, but less efficient

Tokenisation

Mr. O'Neill thinks that the boys' stories about Chile's capital aren't amusing.

neill
oneill
o'neill
o' neill
o neill

?

aren't
arent
aren t
are n't

?

Tokenisation problems: One word or two? (or several)

Hewlett-Packard

State-of-the-art

co-education

the hold-him-back-and-drag-him-away maneuver

data base

San Francisco

Los Angeles-based company

cheap San Francisco-Los Angeles fares

York University vs. New York University

Numbers

20/3/91
3/20/91
Mar 20, 1991

B-52
6-year-old

100.2.86.144

(800) 234-2333
800.234.2333

.74189359872398457

- Older IR systems may not index numbers...
- ... but generally it's a useful feature.

Chinese: No Whitespace

莎拉波娃现在居住在美国东南部的佛罗里达。今年 4 月 9 日，莎拉波娃在美国第一大城市纽约度过了 1 8 岁生日。生日派对上，莎拉波娃露出了甜美的微笑。

- Need to perform word segmentation
- Use a lexicon or supervised machine-learning
- Ambiguity 和尚
  - As one word, means "monk"
  - As two words, means "and" and "still"

Script-related Problems

ノーベル平和賞を受賞したワンガリ・マータイさんが名誉会長を務めるＭＯＴＴＡＩＮＡＩキャンペーンの一環として、毎日新聞社とマガジンハウスは「私 の、もったいない」を募集します。皆様が日ごろ「もったいない」と感じて実践していることや、それにまつわるエピソードを８００字以内の文章にまとめ、簡 単な写真、イラスト、図などを添えて１０月２０日までにお送りください。大賞受賞者には、５０万円相当の旅行券とエコ製品２点の副賞が贈られます。

- Different scripts (alphabets) might be mixed in one language.
- e.g., Japanese has 4 scripts: kanji, katakana, hiragana, romanji
- no spaces

استقلت الجزائر في سنة 1962 بعد 132 عاما من الاحتلال الفرنسي.

← → ← → ← START

'Algeria achieved its independence in 1962 after 132 years of French occupation.'

- Scripts can incorporate different reading directions.
- e.g., Arabic script and bidirectionality
- Rendering vs. conceptual order

Other cases of "no whitespace": Compounding

Compounding in Dutch, German, Swedish

German

Lebensversicherungsgesellschaftsangestellter
leben+s+versicherung+s+gesellschaft+s+angestellter

Other cases of "no whitespace": Agglutination

"Agglutinative" languages do this not just for compounds:

Inuit

tusaatsiarunnangittualuujunga
(= "I can't hear very well")

Finnish

epäjärjestelmällistyttämättömyydellänsäkäänköhän
(= "I wonder if – even with his/her quality of not having been made unsystematized")

Turkish

Çekoslovakyalılaştıramadıklarımızdanmşçasına
(= "as if you were one of those whom we could not make resemble the Czechoslovacian people")

Casefolding, accents, diacritics

- Casefolding can be semantically distinguishing:

Fed vs. fed
March vs. march
Turkey vs. turkey
US vs. us

- Though in most cases it's not.
- Accents and Diacritics can be semantically distinguishing:

Spanish

peña = cliff, pena = sorrow

- Though in most cases they are not (résumé vs. resume)
- Most systems case-fold (reduce all letters to lower case) and throw away accents.
- Main decision criterion: will users apply it when querying?

Stop words

- Extremely common words which are of little value in helping select documents matching a user need

a, an, and, are, as, at, be, by, for, from, has, he, in, is, it, its, of, on, that, the, to, was, were, will, with

- Used to be standardly non-indexed in older IR systems.
- Need them to search for the following queries:

to be or not to be
prince of Denmark
bamboo in water

- Length of practically used stoplists has shrunk over the years.
- Most web search engines do index stop words.

Lemmatisation

- Reduce inflectional/variant forms to base form

am, are, is → be
car, car's, cars', cars → car
the boy's cars are different colours → the boy car be different color

- Lemmatisation implies doing "proper" reduction to dictionary headword form (the lemma)
- Inflectional morphology (cutting → cut)
- Derivational morphology (destruction → destroy)

Stemming

- Stemming is a crude heuristic process that chops off the ends of words in the hope of achieving what “principled” lemmatisation attempts to do with a lot of linguistic knowledge.

automate, automation, automatic $\rightarrow$ automat

- language dependent, but fast and space-efficient
- does not require a stem dictionary, only a suffix dictionary
- Often both inflectional and derivational

Porter Stemmer

- M. Porter, "An algorithm for suffix stripping", Program 14(3):130-137, 1980
- Most common algorithm for stemming English
- Results suggest it is at least as good as other stemmers
- Syllable-like shapes + 5 phases of reductions
- Of the rules in a compound command, select the top one and exit that compound (this rule will have affecte the longest suffix possible, due to the ordering of the rules).

Stemming: Representation of a word

[C] (VC){m}[V]

C : one or more adjacent consonants
V : one or more adjacent vowels

[ ] : optionality
( ) : group operator
{**x**} : repetition x times
m : the "measure" of a word

shoe	$[sh]_C[oe]_V$	m=0
Mississippi	$[M]_C([i]_V[ss]_C)([i]_V[ss]_C)([i]_V[pp]_C)[i]_V$	m=3
ears	$([ea]_V[rs]_C)$	m=1

Notation: measure *m* is calculated on the word **excluding** the suffix of the rule under consideration

Porter stemmer: selected rules

SSES → SS
IES → I
SS → SS
S →

caresses → caress
cares → care

(m>0) EED → EE

feed → feed
agreed → agree
BUT: freed, succeed

(*v*) ED →

plastered → plaster
bled → bled

Three stemmers: a comparison

Such an analysis can reveal features that are not easily visible from the variations in the individual genes and can lead to a picture of expression that is more biologically transparent and accessible to interpretation.

Porter Stemmer

such an analysi can reveal featur that ar not easili visibl from the variat in the individu gene and can lead to a pictur of express that is more biolog transpar and access to interpret

Lovins Stemmer

such an analys can reve featur that ar not eas vis from th vari in th individu gen and can lead to a pictur of expres that is mor biolog transpar and acces to interpres

Paice Stemmer

such an analys can rev feat that are not easy vis from the vary in the individ gen and can lead to a pict of express that is mor biolog transp and access to interpret

Does stemming improve effectiveness?

- In general, stemming increases effectiveness for some queries and decreases it for others.

Example queries where stemming helps

tartan sweaters → sweater, sweaters
sightseeing tour san francisco → tour, tours

Example queries where stemming hurts

operational research	→ oper	= operates, operatives, operate, operation, operational, operative
operating system	→ oper	
operative dentistry	→ oper	

More equivalence classing

- Thesauri: semantic equivalence, car = automobile
- Soundex: phonetic equivalence, Muller = Mueller

Overview

Phrase Queries

- We want to answer a query such as [cambridge university] – as a phrase.

- None of these should be a match:

The Duke of Cambridge arriving at St John's College, Cambridge alongside Leszek Borysiewicz Vice Chancellor University of Cambridge, Polly Coutice Director of Cambridge Programme Sustainability and Proffessor Christopher Dobso Photo: PA

The Duke of Cambridge was welcomed by University of Cambridge officials as he began a 10-week course on Tuesday.

- But this one is OK:

Prince William begins agricultural course at Cambridge University

Phrase Queries

- About 10% of web queries are phrase queries.
- Consequence for inverted indexes: no longer sufficient to store docIDs in postings lists.
- Two ways of extending the inverted index:
  - biword index
  - positional index

- Index every consecutive pair of terms in the text as a phrase.

Friends, Romans, Countrymen

Generates two biwords:

friends romans

romans countrymen

- Each of these biwords is now a vocabulary term.
- Two-word phrases can now easily be answered.

Longer phrase queries

- A long phrase like cambridge university west campus can be represented as the Boolean query

cambridge university AND university west AND west campus

- We need to do post-filtering of hits to identify subset that actually contains the 4-word phrase.

Issues with biword indexes

- Why are biword indexes rarely used?
- False positives, as noted above
- Index blowup due to very large term vocabulary

Positional indexes

- Positional indexes are a more efficient alternative to biword indexes.
- Postings lists in a nonpositional index: each posting is just a docID
- Postings lists in a positional index: each posting is a docID and a list of positions (offsets)

Positional indexes: Example

Query: "to_1 be_2 or_3 not_4 to_5 be_6"

TO, 993427:
$\langle$ 1: $\langle 7, 18, 33, 72, 86, 231\rangle$;
2: $\langle 1, 17, 74, 222, 255\rangle$;
4: $\langle 8, 16, 190, 429, 433\rangle$;
5: $\langle 363, 367\rangle$;
7: $\langle 13, 23, 191\rangle$; $\ldots\rangle$

BE, 178239:
$\langle$1: $\langle 17, 25\rangle$;
4: $\langle 17, 191, 291, 430, 434\rangle$;
5: $\langle 14, 19, 101\rangle$; $\ldots\rangle$

Positional indexes: Example

Query: "to_1 be_2 or_3 not_4 to_5 be_6"

TO, 993427:
$\langle$ 1: $\langle$7, 18, 33, 72, 86, 231$\rangle$;
2: $\langle$1, 17, 74, 222, 255$\rangle$;
4: $\langle$8, 16, 190, 429, 433$\rangle$;
5: $\langle$363, 367$\rangle$;
7: $\langle$13, 23, 191$\rangle$; . . . $\rangle$

BE, 178239:
$\langle$1: $\langle$17, 25$\rangle$;
4: $\langle$17, 191, 291, 430, 434$\rangle$;
5: $\langle$14, 19, 101$\rangle$; . . . $\rangle$

As always: docid, term, doc freq; new: offsets

Positional indexes: Example

Query: "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⟩; . . . ⟩

Document 4 is a match!

Complexity of search with positional index

- Unfortunately, $\Theta(T)$ rather than $\Theta(N)$
  - T ... number of tokens in document collection
  - N ... number of documents in document collection
- Combination scheme:
  - Include frequent biwords as vocabulary terms in the index ("Cambridge University", "Britney Spears")
  - Resolve all other phrases by positional intersection

Proximity search

- We just saw how to use a positional index for phrase searches.
- We can also use it for proximity search.

employment /4 place

- Find all documents that contain employment and place within 4 words of each other.
- HIT: Employment agencies that place healthcare workers are seeing growth.
- NO HIT: Employment agencies that have learned to adapt now place healthcare workers.

Proximity search with positional index

- Simplest algorithm: look at cross-product of positions of (i) "employment" in document and (ii) "place" in document
- Note that we want to return the actual matching positions, not just a list of documents.
- Very inefficient for frequent words, especially stop words
- More efficient algorithm in book

Take-away

- Understanding of the basic unit of classical information retrieval systems: words and documents: What is a document, what is a term?
- Tokenization: how to get from raw text to terms (or tokens)
- More complex indexes for phrase and proximity search
  - biword index
  - positional index

Reading

- MRS Chapter 2.2
- MRS Chapter 2.4