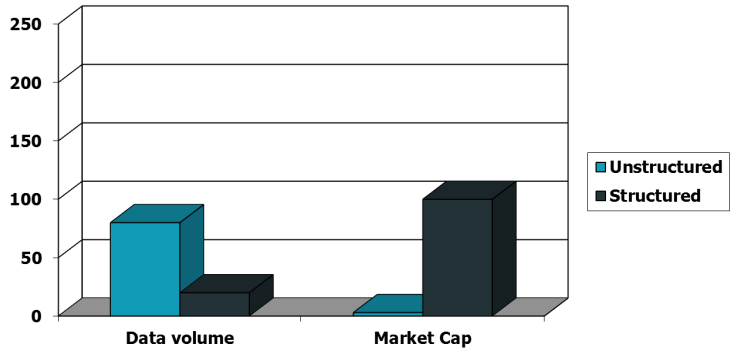
DD2476 Search Engines and Information Retrieval Systems Lecture 1: Introduction

Hedvig Kjellström
hedvig@kth.se
https://www.kth.se/social/course/DD2476

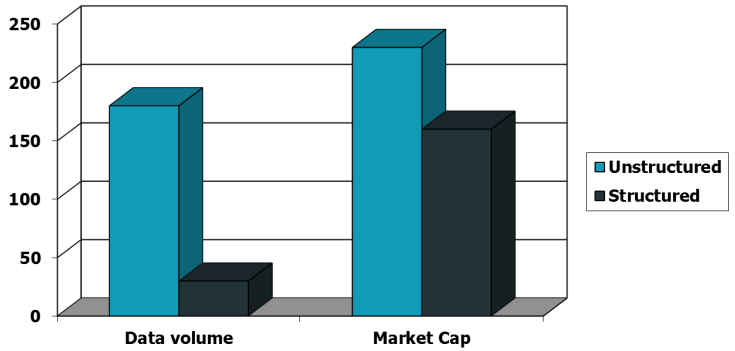
Definition

- Information Retrieval (IR) is finding material (usually documents) of an unstructured nature (usually text) that satisfies an information need from within large collections (usually stored on computers).

Unstructured (text) vs structured (database) data in the mid-nineties



Unstructured (text) vs structured (database) data today



How good are the retrieved docs?

- Precision: Fraction of retrieved docs that are relevant to the user's information need
- Recall: Fraction of relevant docs in collection that are retrieved

More in Lecture 3

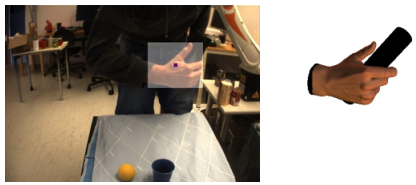
Today

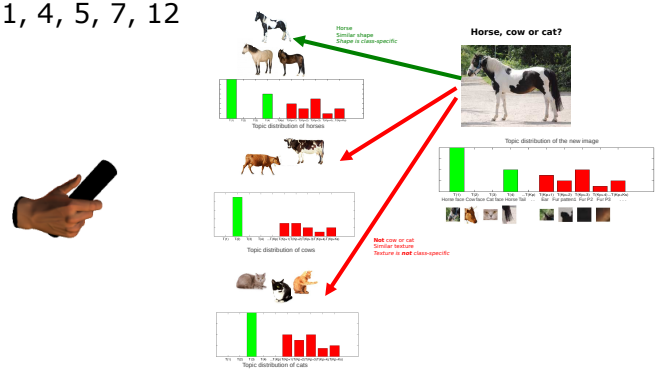
- Presentation of lecturers
- Course practicalities
  - Curriculum
  - Examination
  - Course homepage: https://www.kth.se/social/course/DD2476
- Boolean retrieval (Manning Chapter 1)
  - Building an inverted index
  - Boolean queries
- Term vocabulary (Manning Chapter 2)
  - Elements of text

Presentation of Lecturers

Hedvig Kjellström

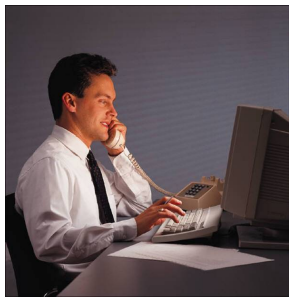
- Associate Professor at CSC
- Researcher in Robotics at CVAP, CSC
- Lecture 1, 4, 5, 7, 12

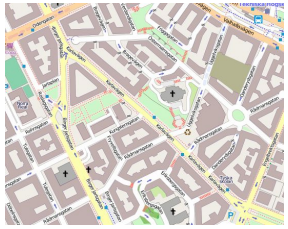




Johan Boye

- Associate Professor at CSC
- Researcher in Language Technology at TCS, CSC
- Lecture 1-3, 5





Jussi Karlgren

- Founder of Gavagai AB, Adjunct Professor at CSC
- Researcher in Language Technology at TCS, CSC
- Lecture 3, 6

Viggo Kann

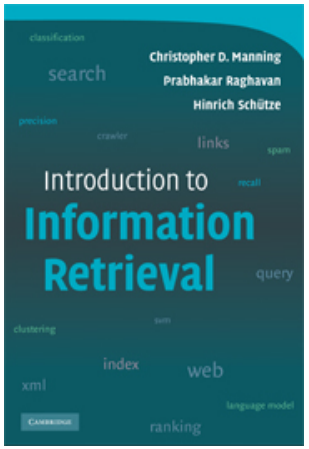
- Professor at CSC
- Researcher in Theoretical Computer Science at TCS, CSC
- Lecture 8

Course Practicalities

Curriculum

- C. D. Manning, P. Raghavan and H. Schütze, *Introduction to Information Retrieval*, Cambridge University Press, 2008

- Preliminary version available online in pdf format
  - See course homepage: https://www.kth.se/social/course/DD2476



Curriculum

- The whole book will be covered
- Depth according to learning outcomes
  - See course homepage: https://www.kth.se/social/course/DD2476
- Reading on your own necessary
  - Lectures cover only highlights, very high pace
  - Examination on whole curriculum
- Course given for the fourth time
  - More focus on evaluation in the assignments, one more lecture on evaluation
  - Less focus on implementational details

Examination

- Three computer assignments (6 ECTS, A-F)
  - Individually
  - Lab 1 (Lecture 1-3 readings) February 18
  - Lab 2 (Lecture 4-6 readings) March 18
  - Lab 3 (Lecture 7-8 readings) April 1
- Please register in Rapp: rapp.csc.kth.se/rapp

- Project (3 ECTS, A-F)
  - Groups of four-five students
  - Presentation (Whole curriculum) May 16

Course Homepage

- News!
- Schedule with readings and examination deadlines
- Contact information
- Computer assignment and project descriptions

- https://www.kth.se/social/course/DD2476

Boolean Retrieval

(Manning Chapter 1)

A First Information Retrieval Example

- Ad hoc retrieval: Find documents in a collection of documents (corpus), relevant to a certain user need
- Boolean retrieval model: Model in which queries are posed as Boolean expressions
- Example: Shakespeare
  - Find all Shakespeare plays that contain the words

BRUTUS AND CAESAR AND NOT CALPURNIA

BRUTE Force Approach

- One could grep all of Shakespeare's plays for BRUTUS and CAESAR, then strip out plays containing CALPURNIA
  - Unix command `grep`, linear search
- Why is that not the answer?
  - Slow (for large corpora)
  - Other operations (e.g., find the word ROMANS NEAR COUNTRYMEN) not feasible
  - Ranked retrieval (best documents to return)
- Instead, organize beforehand

Term-Document Incidence Matrix

Document = play

Term = word

	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

1 if play contains word, 0 otherwise

Bitwise Operations

Document = play

Term = word

	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

BRUTUS AND CAESAR AND NOT CALPURNIA
110100 AND 110111 AND NOT 010000
110100 AND 110111 AND 101111
= 100100 (**Antony and Cleopatra**, **Hamlet**)

Answers to Query

- **Antony and Cleopatra**, Act III, Scene ii
  Agrippa [Aside to Domitius Enobarbus]:
  Why, Enobarbus,
  When Antony found Julius CAESAR dead,
  He cried almost to roaring; and he wept
  When at Philippi he found BRUTUS slain.

- **Hamlet**, Act III, Scene ii
  Lord Polonius:
  I did enact Julius CAESAR: I was killed i' the Capitol; BRUTUS killed me.

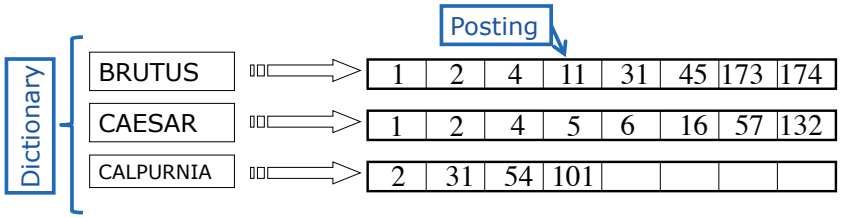
Exercise 5 Minutes

- Consider 10^6 documents, each with ~10^3 words.
- Avg 6 bytes/word including spaces/punctuation
  - 6GB of data.
- Say there are $0.5*10^6$ *distinct* terms among these.
- Normal size collection!

- Discuss in pairs:
  - What are the problems with using the term-document incidence matrix on a collection this size?
  - How can the method be adapted to solve these problems?

Inverted Index

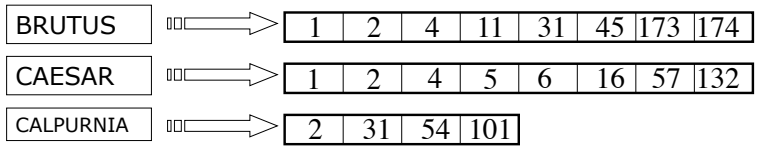
- For each term *t*, store a list of all documents that contain *t*.
  - Identify each by a docID, a document serial number



- Can we use fixed-size arrays for this?
- What happens if the term CAESAR is added to document 14?

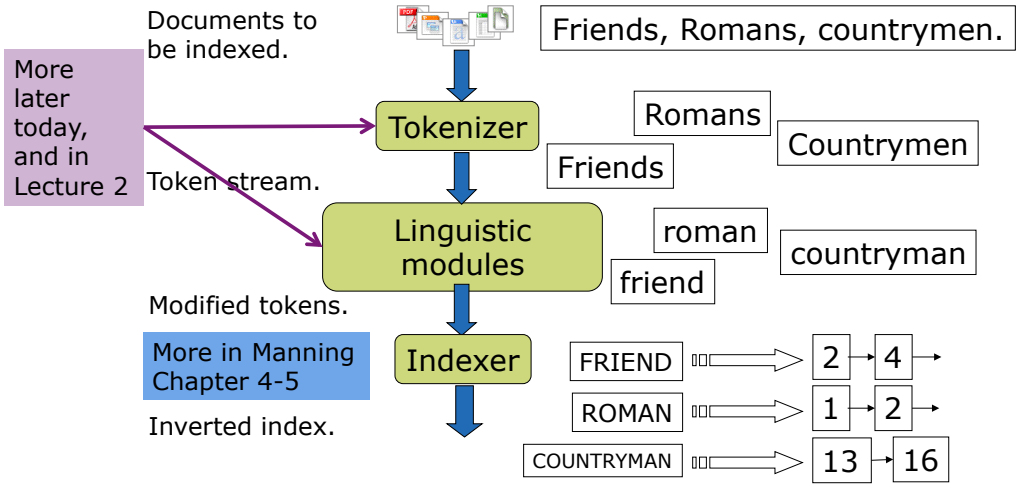
Inverted Index

- Need variable-size posting lists
- Implementational details
  - trade-off storage size/ease of insertion
  - Sort lists wrt DocID



More in Manning Chapter 4-5

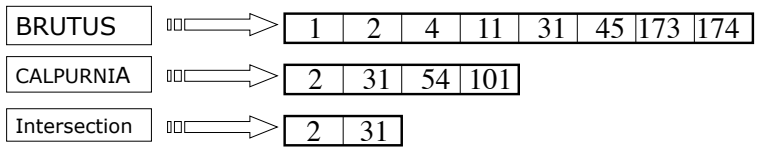
Building an Inverted Index



Query Processing with Inverted Index

- Boolean queries are processed as with the incidence matrix

BRUTUS AND CALPURNIA



- NOT can also be handled with search
- Organizing this work (sorting, evaluation order): query optimization

More in Manning Chapter 1

Beyond Term Search

More in Lecture 2

- Allow compounds, e.g., phrases "…"
  - "FRIENDS, ROMANS, COUNTRYMEN!"
- Additional operators, e.g., NEAR
  - CAESAR NEAR CALPURNIA
  - Index has to capture term proximity
- Zones in documents
  - (author = SHAKESPEARE) AND (text contains WORSER)

More in Manning Chapter 10

Beyond Term Search

- Not only presence/absence, but also term frequency
  - 0 vs 1 hit
  - 1 vs 2 hits
  - 2 vs 3 hits
  - Usually, more is better

More in Lecture 4

Exercise 5 Minutes

- Try the search feature at www.rhymezone.com/shakespeare
  - Who has an open browser? Find someone nearby, or come up to me.
- Discuss in groups:
  - What could it do better?
  - Write down

IR vs Databases: Structured vs Unstructured Data

Employee	Manager	Salary
Smith	Jones	50000
Chang	Smith	60000
Ivy	Smith	50000

- Typically allows numerical range and exact match (for text) queries, e.g.,
  Salary ≥ 60000 AND *Manager* = Smith.

Unstructured Data

More in Lectures 9,12

- Typically refers to free text
  - Images
  - Other media files
- Allows
  - Keyword queries including operators
  - More sophisticated "concept" queries e.g.,
    find all web pages dealing with "drug abuse"
  - Classic model for searching text documents

More in Lecture 4

- No data is truly unstructured
  - Grammar
  - Semistructured search, e.g., XML

More in Lecture 7

More in Manning Chapter 10

Organizing Data

More in Manning Chapter 16-17

- Boolean queries only give inclusion or exclusion of docs.
- Clustering: Given a set of docs, group them into clusters based on their contents.

More in Manning Chapter 13-14

- Classification: Given a set of topics, plus a new doc *D*, decide which topic(s) *D* belongs to.
- Ranking: Can we learn how to best order a set of documents, e.g., a set of search results

More in Lecture 4

The Web and Its Challenges

- Unusual and diverse documents
- Unusual and diverse users, queries, information needs
- Beyond terms, exploit ideas from social networks
  - E.g. link analysis

  More in Lectures 5, 6
- How do search engines work? And how can we make them better?

  More in Lectures 5, 6, 8, 10, 11

Next

- HOUR 2: Johan Boye
- Lecture 2 (February 7, 10.15-12.00)
  - B1
  - Readings: Manning Chapter 2, 3
- Computer Assignment 1 (now – February 18)
  - Register in Rapp: rapp.csc.kth.se/rapp
  - Assignment description: https://www.kth.se/social/course/DD2476

Indexing pipeline

Documents

Byte stream

F r i e n d s , r o m a n s , a n d c o u n t r y m e n

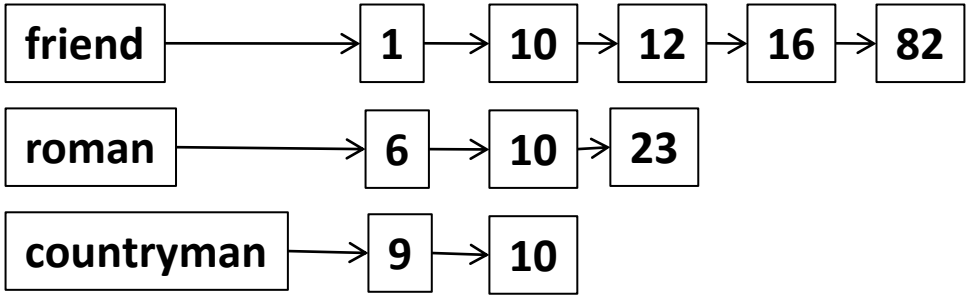
Token stream

Friends romans countrymen

Term stream

friend roman countryman

Inverted index



Documents

- **What is a document, anyway?**
  - a **file?**
  - an **e-mail?**
  - an **e-mail** with **attachments?**
  - a **group of files (PPT or LaTeX as HTML pages)?**
  - a **book**?
  - a **chapter**?
  - a **paragraph**?
  - a **sentence**?

Documents

- Many **different formats** (html, text, Word, Excel, PDF, PostScript, …), **languages** and **character sets**
- **Multilinguality**
  - **Swedish e-mail** with **English attachment**

Character formats

- **Text encodings**
  - **ASCII** (de-facto standard from 1968), 7-bit (=128 chars, 94 printable). Most common on the www until Dec 2007.
  - **Latin-1** (**ISO-8859-1**), 8-bit, ASCII + 128 extra chars
  - **Unicode** (109 000 code points)
  - **UTF-8** (variable-length encoding of Unicode)
- **Page Description Languages**
  - **PostScript** (really a programming language)
  - **PDF** (open standard since 1 July, 2008)
  - **DVI, DOC, …**

Tokenization

- Input: ”**Friends, romans and countrymen**”
- Output: tokens
  - **Friends**
  - **romans**
  - **countrymen**
- Usually, **spaces** and **punctuation** delimits tokens
- But not always:
  - **San Francisco**, **Richard III**, **et cetera**, …
  - **http://www.kth.se**, **jboye@nada.kth.se**
  - **:-)**

More tokenization issues

- Apostrophes:
  - **Finland's → Finland's? Finlands? Finland? Finland s?**
  - **don't → don't ? don t ? do not ? don t?**
- One token or several?
  - **state-of-the-art → state-of-the-art?**
    state of the art?
    state art?
  - **this is a *don't-want-to-leave-bed* day**
  - **Microsoft Word**
  - **Hewlett-Packard**
  - **b-flat**
  - **the *San Francisco-Los Angeles* flight**

Numbers

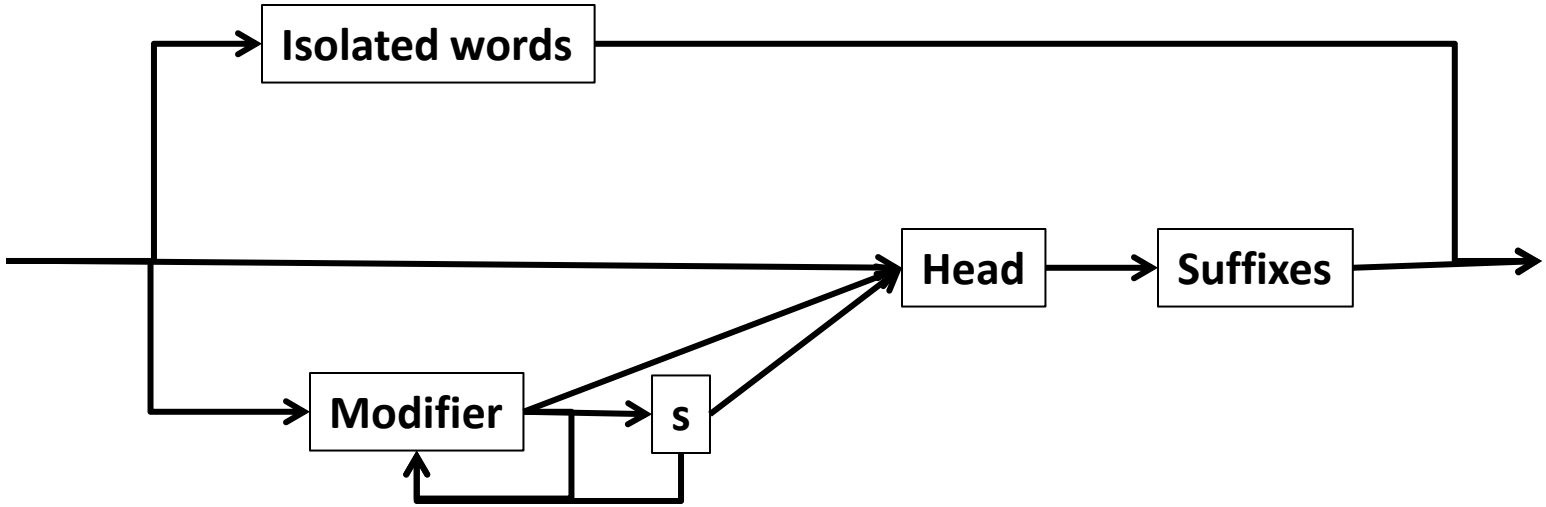
- Can contain spaces or punctuation
  - **123 456.7** or **123,456.7** or **123 456,7**
  - Often useful to index numbers (looking up error codes etc. on the web)
- **+46 (8) 790 60 00**
- **3/20/91 Mar. 12, 1991 20/3/91**
- **B-52**
- My PGP key is **324a3df234cb23e**
- **131.169.25.10**

Language-specific issues

- French:
  - **L'ensemble** → **Le** ? **L** ? **L'** ?
  - want **un ensemble** to match **l'ensemble**
- German and Swedish:
  - compound words are not segmented
  - ***Lebensversicherungsgesellschaftsangestellter*** (German)
  - "Life insurance company employee"
  - ***Försäkringsbolagsanställd*** (Swedish)
  - beneficial to use a compound splitter

Compound splitting

Can be achieved with finite-state techniques.



Compound splitting

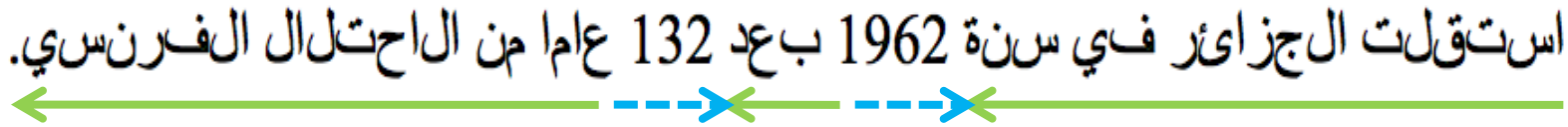
- In Swedish: **försäkringsbolag** (insurance company)
  - **bolag** is the head
  - **försäkring** is a modifier
  - the **s** is an infix
- This process can be recursive:
  - försäkringsbolagslagen (the insurance company law)
  - **en** is a suffix indicating definite form
  - **lag** is the head
  - the **s** is an infix
  - **försäkringsbolag** is the modifier

Language-specific issues

- Chinese and Japanese have no spaces between words:
  - 莎拉波娃现在居住在美国东南部的佛罗里达。
  - Not always guaranteed a unique tokenization
- Japanese have several alphabets
  - **Katakana** and **Hiragana** (syllabic)
  - **Kanji** (Chinese characters)
  - **Romaji** (Western characters)
  - All of these may be intermingled in the same sentence

Language-specific issues

- Right-to-left languages
  - Arabic, Hebrew, Farsi, Urdu, Pashtu, …
  - Some tokens (numbers, years, …) are read left-to-right



  - ‘Algeria achieved its independence in 1962 after 132 years of French occupation.’
  - With Unicode, surface form is complex but stored form is straightforward

Stop words

- Exclude the most common words
  - In English: **the**, **a**, **and**, **to**, **for**, **be**, …
  - Little semantic content
  - ~30% of postings for top 30 words
- However:
  - ”**Let it be**”, ”**To be or not to be**”, ”**The Who**”
  - ”**King of Denmark**”
  - ”**Flights to London**” vs ”**Flights from London**”
  - Trend is to keep stop words: compression techniques means that space requirements are small

Normalization

- Tokens → **terms**
- Term = normalized word form
- Terms are the **atomic elements** of **indexing** and **search**
- Normalization determine equivalence classes of words
  - by deleting full stops: **U.S.A** → **USA**
  - by deleting hyphens: **co-operation** → **cooperation**

More normalization issues

- Diacritica:
  - **å, ä, ö, à, é, ê, ë, ç ,ñ, č, ł, …**
- Umlaut:
  - **Tübingen** and **Tuebingen**, **Österreich** and **Oesterriech**
- Case folding: convert all letters to lowercase
  - Even for **Microsoft Word**, **UN**, **NATO**, **EU**, etc.
- Assymetric normalization
  - Enter: ***window*** Search: ***window, windows***
  - Enter: ***windows*** Search: ***Windows, windows, window***
  - Enter: ***Windows*** Search: ***Windows***

Lemmatization

- Map **inflected form** to its **lemma** (=base form)
- ”The boys’ cars are different colours” → ”The boy car be different color”
- Requires language-specific linguistic analysis
  - part-of-speech tagging
  - morphological analysis
- Useful in morphologically rich languages, like Finnish:
  - ***järjestelmättömyydellänsäkäänköhän***
  - ”with its lack of organisation”

Part-of-speech tagging

- ”*He usually quarrels*” →
  He **pers. pronoun**
  usually **adverb**
  quarrels **verb-pres-3rd pers**
- ”*His usual quarrels*” →
  His **poss. pronoun**
  usual **adjective**
  quarrels **noun-plur-nom**

Stemming

- Don’t do morphological or syntactic analysis, just **chop off the suffixes**
  - No need to know that ”foxes” is plural of ”fox”
- Much **less expensive** than lemmatization, but **can be very wrong** sometimes
  - stocks → stock, stockings → stock
- Stemming usually improves **recall** but lowers **precision**

Porter's algorithm

- Rule-based stemming for English
  - ATIONAL → ATE
  - SSES → SS
  - ING → ε
- Some context-sensitivity
- (W>1) EMENT → ε
  - REPLACEMENT → REPLAC
  - CEMENT → CEMENT

Sum-up

- **Reading**, **tokenizing** and **normalizing** contents of documents
  - **File types** and **character encodings**
  - Tokenization issues: **punctuation, compound words, word order, stop words**
  - Normalization issues: **diacritica, case folding, lemmatization, stemming**
- We're ready for **indexing**