Information retrieval I

Introduction, efficient indexing, querying

Clovis Galiez

Mastère Big Data

November 14, 2019

Objectives of the course

- Acquire a culture in information retrieval
- Master the basics concepts allowing to understand:
  - what is at stake in novel IR methods
  - what are the technical limits

This will allow you to have the basics tools to analyze current limitations or lacks, and imagine novel solutions.

Proceedings of the lecture

- Heterogeneous audience
- No lecture handbook, only slides and materials of the practicals session.
- So... take notes and ask questions!
- Evaluation: exam and optional project (bonus)

Outline of the lectures

- Indexing, basic querying
- Vector-space model, latent semantics, ranking
- Modern IR techniques: artificial intelligence, embeddings.
  Hands-on session: programming a search engine (Python)
- Hands-on Part-II.

Today's outline

- What is information retrieval (in general)?
- Querying (correctness) and ranking (relevance)
- IR in the context of the web
  - Elements of web protocols and languages
  - Gathering data on the web: crawling
  - What data size is at stake?
- How to represent the information?
  - Indexing
  - Sparse representations
  - Reverse indexing
- Practicals: understanding and selling your patent!

What is information retrieval (IR)?

Definition

Answering a query by extracting **relevant information** from **a collection of documents**.

Typical example

Google.

Some open-source tools for in-house IR

IR tools:

- Lucene
- elastic

NLP tools:

- NLTK (Python)
- spaCy

(far to be exhaustive!)

What is "document" and "information"?

Information retrieval

Answering a query by extracting **relevant information** from **a collection of documents**.

Here, **documents** are web pages, images, pdf, etc.

How would you define **information** in the context of information retrieval?

Information

Subset of documents relevant to a query.

How could you qualify or measure information, e.g. relevance?

Correctness, relevance and truth

When was the last US presidential elections?

correct or **incorrect**

- Blue
- 42:17

true, **false** or**...**

- 1st Sept. 2018
- Nov. 2016

Relevance

- Same time as the previous ones, but 5 years later
- during the 21st Century
- 1478563200s since Unix Epoch[a]

[a]Number of seconds elapsed since 1st of January 1970

Querying and ranking systems

Collection of documents

$\downarrow$

Query $\longrightarrow$ $\boxed{\text{Querying system}}$

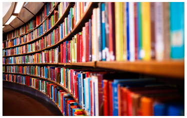
$\downarrow$

Correct answers

$\downarrow$

Ranking of answers (relevance)

Querying systems deal with correctness



Collection of documents

$\downarrow$

Query $\longrightarrow$ $\boxed{\text{Querying system}}$

$\downarrow$

Correct answers

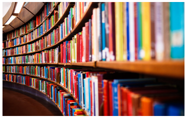
$\downarrow$

Ranking of answers (relevance)

Filter documents that *correctly* answers a given query

- Boolean queries
  - Checks if a word is present or not in a document
- Vector-based models
- Probabilistic models
  - Naïve Bayes model

Ranking systems deal with relevance



Collection of documents

$\downarrow$

Query $\longrightarrow$ $\boxed{\text{Querying system}}$

$\downarrow$

Correct answers

$\downarrow$

Ranking of answers (relevance)

Ranking methods:

- structure-agnostic algorithms
- unsupervised ranking
  - PageRank
- supervised ranking
  - machine learning

What are the pitfalls?

Exercise

Take a few minutes to list what could be the different pitfalls for querying and ranking systems.

- Complexity of natural language
- Ambiguity of natural language
- Size of the data
- ...

IR specific to the World Wide Web

IR and the web

Collection of documents

$\downarrow$

Query $\longrightarrow$ $\boxed{\text{Querying system}}$

$\downarrow$

Correct answers

$\downarrow$

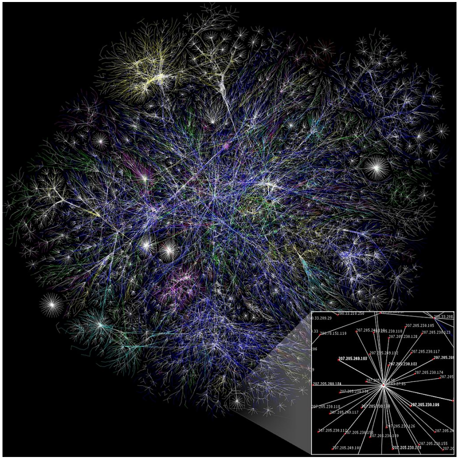
Ranking of answers (relevance)

2 specificities:

- Building the collection of documents
  - **Crawling** the web
  - **Indexing** documents
- Ranking the documents (next lecture)

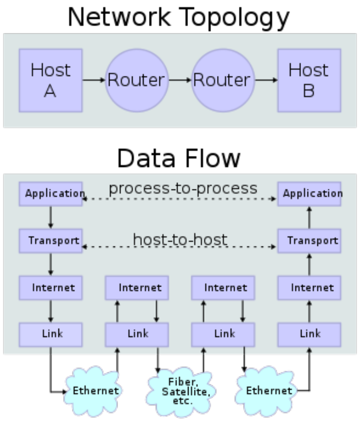
Gathering data on the web (crawling)

The web structure: a huge graph



Initiated in 70's with ARPANET. In 2017, >8 Billion "nodes"

The web protocols

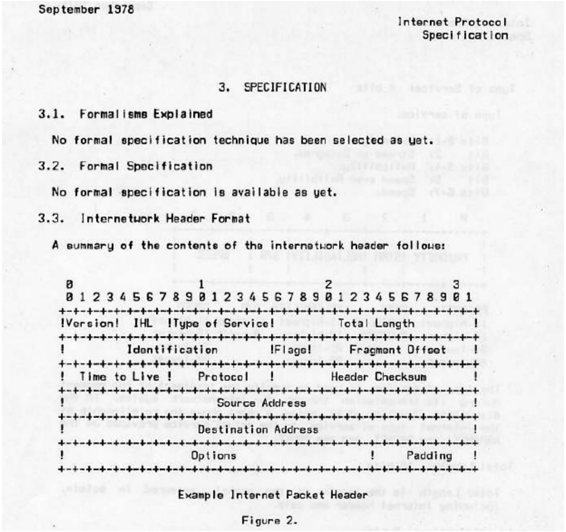


The textual web uses the HTTP[1] over TCP/IP protocol[2].

[1]T. Berners-Lee in 90 at CERN

[2]Cerf and Kahn, 74

Edge-technology: Internet



September 1978

Internet Protocol
Specification

3. SPECIFICATION

3.1. Formalisms Explained

No formal specification technique has been selected as yet.

3.2. Formal Specification

No formal specification is available as yet.

3.3. Internetwork Header Format

A summary of the contents of the internetwork header follows:

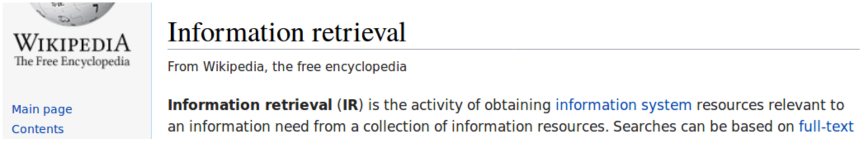
```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!Version!  IHL  !Type of Service!          Total Length         !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!         Identification        !Flags!      Fragment Offset    !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!  Time to Live !    Protocol   !         Header Checksum       !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                       Source Address                          !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                    Destination Address                        !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                    Options                    !    Padding    !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Example Internet Packet Header

Figure 2.

The web structure: languages

HTML (*HyperText Markup Language*) is the main language for describing a web page. From `https://en.wikipedia.org/wiki/Information_retrieval`:



Information retrieval

From Wikipedia, the free encyclopedia

Information retrieval (**IR**) is the activity of obtaining information system resources relevant to an information need from a collection of information resources. Searches can be based on full-text

HTML source code behind:

```
... obtaining <a href="/wiki/Information_system" title="Information system">
information system</a>  resources relevant to an ...
```

How would you collect information from the web?

The web structure: crawling

Hopping from link to link, one can collect/process data on the web:

Web crawling
JumpStation (1993)

⚠ Must keep track of already visited pages (e.g. trie, hash table).

Parsing links from a web page

With regular expressions (regex) for instance.

Regex	Match examples	
`a*`	aaa	a
`M.x`	Max	Mix
`M.*x`	Max	Matrix
`M[^a]*[xn]`	Mix	Moon

Regex is a simple pattern matching formalism.

Regex: groups

Regex	data	1st group
`M(.)x`	Max	a
`M(.)x`	Mix	i
`M(.*)x`	Matrix	atri

Exercise

Find a regex that extracts the URL of an HTML link.

```
<a href="http://www.wikipedia.org">The linked text</a>
```

Extend your regex to extract both the URL and the linked text.

⚠ Quality of the data: HTML errors, difficult parsing.

What is the size of the crawled data?

From technical solution to practice... Quizz:

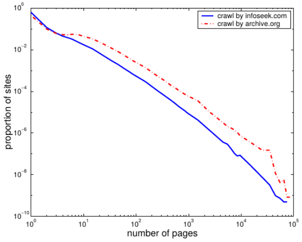
Number of pages indexed by Google	$\sim 10^{11}$
Data size crawled by Google	$> 10^{8}$Gb
Number of pages with $> k/2$ incoming links	$\sim 2^{\gamma}.N_k$ (Zipf's law)
Number of pages of length $> L/2$	$\sim 2^{\gamma}.N_L$ (Zipf's law)

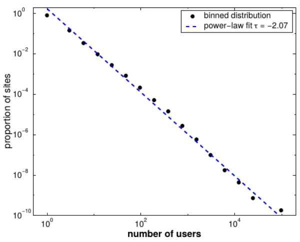
6 degrees of separation law.

Between 0.2% and 4% of the web is accessible by crawling[3].
What is "uncrawlable" is coined the **deep web**.

[3][Pandya et al. IJIRST 2017]

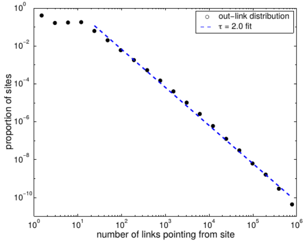
Experimental evidence of the Zipf's law

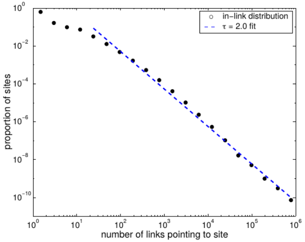




c)

d)





[Adamic et al. *Glottometrics*, 2002]

From gathering to representation

Collection of documents

$\downarrow$

Query $\longrightarrow$ $\boxed{\text{Querying system}}$

$\downarrow$

Correct answers

$\downarrow$

Ranking of answers (relevance)

Representations of a web document

How to query for correct documents?

Exercise

Take a few minutes to think how you would retrieve the web documents corresponding to the query:

```
result last elections president united states
```

You may have encounter the following issues:

- how to correctly match words in the document (tokenization)
- how to match equivalent word (e.g. plural)
- how to implement it

Tokenization

Process of chopping the text of a document in atomic elements:

Brian is in the kitchen	$\rightarrow$	$\underline{\text{Brian}}$ $\underline{\text{is}}$ $\underline{\text{in}}$ $\underline{\text{the}}$ $\underline{\text{kitchen}}$
United States president	$\rightarrow$	$\underline{\text{United States}}$ $\underline{\text{president}}$

Usually, tokenizers remove the punctuation.

⚠ May be difficult: `United States` $\neq$ `United + States`!

- Data mining approaches help to extract the right tokens: if two words are significantly seen one after the other, may be consider as a token.
- Some languages are agglutinative (e.g. Turkish).

Python library for NLP: nltk

```
>>> import nltk
>>> sentence = """At eight o'clock on Thursday morning
... Arthur didn't feel very good."""
>>> tokens = nltk.word_tokenize(sentence)
>>> tokens
['At', 'eight', "o'clock", 'on', 'Thursday', 'morning',
'Arthur', 'did', "n't", 'feel', 'very', 'good', '.']
```

https://www.nltk.org/

Stemming

Language-specific rules defining equivalent words up to a usual transformation (e.g. -ing, -ed, -s, etc.).
For instance, we would transform:

plural forms:	elections	$\rightarrow$	election
substantive/adjectival forms:	presidential	$\rightarrow$	president
stop-words removal:	in	$\rightarrow$	`NULL`

⚠ Again, same difficulties could appear:

- ambiguity: police, policy $\rightarrow$ polic
- Non-conflating: mother, maternal

Implementation of stemming in Python

```
>>> from nltk.stem.porter import *
>>> stemmer = SnowballStemmer("english")
>>> print(stemmer.stem("running"))
run
```

https://www.nltk.org/

After tokenizing and stemming: querying

Query: `result last elections president united states`

Stemmed tokens:
`result, last, election, president, united states`

How to query your documents?

Naive representation: vector of tokens

Matrix with the occurrence of tokens in documents.

	tok 1	tok 2	tok 3	tok 4	tok 5	...
	election	president	crazy	united	United States	...
doc 1doc 1	1	1	0	0	1	...
doc 2	0	1	1	0	1	...
doc 3doc 3	1	1	1	0	1	...
...	...	...	...	...	...	...
Query	1	1	0	0	1	...

Exercise

Write down a simple algorithm that extracts given a doc-tok matrix the documents matching a query.
Can you foresee any practical problem? What is the size of the matrix?
Can it fit in **memory**?

Sparse representation

Since documents contain only a small fraction of existing tokens, most of the vector of token entries are null.
We can use a sparse encoding of the same information:

tok 1	tok 2	tok 3	tok 4	tok 5
election	president	crazy	united	United States

doc 1$\rightarrow$tok 1, tok 2, tok 5
doc 2$\rightarrow$tok 2, tok 3, tok 5
doc 3$\rightarrow$tok 1, tok 2, tok 3 ,tok 5

Exercise

What is the size of the sparse encoding data structure?

Hmmm

Write an algorithm extracting the matching documents from a sparse encoding doc-tok.**How long** would it take to process a query?

Elements in complexity

Definition

The complexity is the measure the of the size an algorithm needs of memory and the time it takes to process.

It is usually measured in terms of order of magnitude of the size of the input data.

Examples: Let N be the number of indexed documents, T the total number of tokens, and k the average number of token per document[4].

- Memory complexity of the naive indexing algorithm? $\mathcal{O}(N.T)$
- Memory complexity of the sparse indexing algorithm? $\mathcal{O}(N.k)$
- Time complexity of your search algorithm in a sparse index? $\mathcal{O}(N.k)$

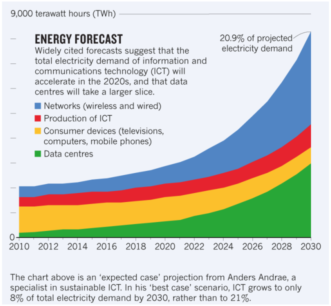
Can we do better?

[4] k can be related to the length of the document through Heap's law: $k = K.L^{\beta}$. In practice, $K = 50, \beta = 0.5$

Complexity matters

Action	Complexity
Sorting	$\mathcal{O}(N \log N)$
Searching a sorted list	$\mathcal{O}(\log N)$
Accessing an element of a matrix	$\mathcal{O}(1)$

Complexity matters!



Electric energy consumption in France per year per person: 0.067GWh
Electric energy of Google per year: 2.500GWh[5].
Equivalent consumption of 40.000 people.

Exercise

Imagine a way of improving how to query a big set of documents.

[5] according to Google. Other sources say $4\times$ more

Inverse sparse index

Idea

Inverting the sparse representation and sorting by document allows to reduce the complexity.

tok 1	tok 2	tok 3	tok 4	tok 5
election	president	crazy	united	United States

doc 1→tok 1,tok 2,tok 5
doc 2→tok 2,tok 3,tok 5
doc 3→tok 1,tok 2,tok 3,tok 5

tok 1→doc 1,doc 3,...
tok 2→doc 1,doc 2,doc 3,...
tok 3→doc 2,doc 3,...
tok 4→doc 102,...
tok 5→doc 1,doc 2,doc 3,...

Exercise

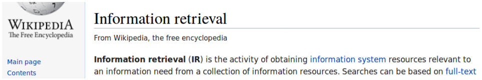
Write an algorithm that indexes a document using a reverse sparse index.
Compute the time complexity for querying an inverse sparse index.

Building an inverted index in practice

The full sparse index does not fit in memory.
Block Sort-Based Indexing is a simple algorithm allows to invert big dictionaries that do not fit in main memory, at low cost, and that can even be parallelized[6].

[6] `https://westmont.instructure.com/files/51060/download?download_frd=1`

Summary


WIKIPEDIA
The Free Encyclopedia

Main page
Contents

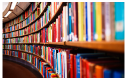
Information retrieval

From Wikipedia, the free encyclopedia

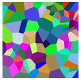
Information retrieval (**IR**) is the activity of obtaining information system resources relevant to an information need from a collection of information resources. Searches can be based on full-text

$\downarrow$

$\downarrow$



$\downarrow$



$\downarrow$

Glimpse of next week: the boolean queries are not flexible

Example

Query: `result elections United States`
Doc title: "White House election: live results!"

With a good stemming and tokenization, we will match `result` and `election`... we miss the match between `United States` and `White House` :-/

Any solution?

- Use semantics (ontologies)
- Use query expansion (add related terms to the query)
- Next week: Use a more flexible querying system

Glimpse of next week: the boolean queries do not rank

Example

Query: `result elections United States`
matching results: 718,698,789

How to pick up most relevant results first?

- Next week: With richer querying and representation of the information
- Next week: By exploiting the graph structure of the web (Google)

Some open-source tools for in-house IR

IR tools:

- Lucene
- elastic

NLP tools:

- NLTK (Python)
- spaCy

Everything is already done?

… but always room for new ideas!

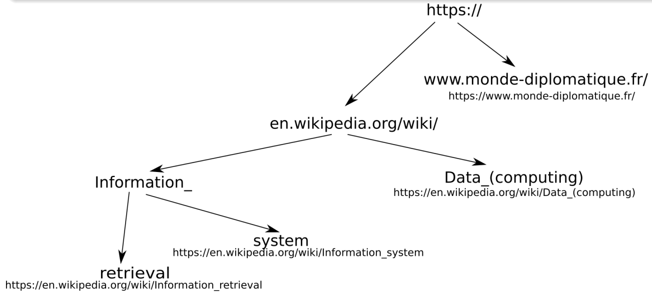
Practicals: Patent analysis.

Extras

How to store already crawled URLs?

Exercise

What is the cost of checking if the crawler already visited a web page? Is it reasonable?



https://
www.monde-diplomatique.fr/
https://www.monde-diplomatique.fr/
en.wikipedia.org/wiki/
Information_
Data_(computing)
https://en.wikipedia.org/wiki/Data_(computing)
system
https://en.wikipedia.org/wiki/Information_system
retrieval
https://en.wikipedia.org/wiki/Information_retrieval

A *trie* structure.

Exercise

What is the complexity in time?

Examples of (successful) companies in IR

ElasticSearch	Distributed, RESTful[7] search and analytics engine capable of solving a growing number of use cases. [...] centrally stores your data so you can discover the expected and uncover the unexpected.
swiftype	All-in-one relevance, lightning-fast setup and unprecedented control.
blekko	Now in IBM Watson

[7] i.e. based on HTTP