

# Technologies du Web

## Master COMASIC

### Search Engines

Antoine Amarilli<sup>1</sup>

December 2, 2014

---

<sup>1</sup>Course material adapted from the slides of the *Web Data Management* book, <http://webdam.inria.fr/Jorge/files/slwebsearch.pdf>. Thanks to Pierre Senellart for proofreading this.

# Search engines

- In the very beginning: **centralized** list of Web servers.
- Later: only **links** (and URL exchange) to find information!
- Early efforts: **Wanderer**, **W3Catalog** (1993).
- First search engine with crawling, indexing and searching: **JumpStation** (December 1993).
- All text indexing: **WebCrawler** (April 1994; brand still active).
- **Lycos** (1994).
- More successful than hand-curated **portals**.
- Nowadays: 93% of Internet traffic from search engines.<sup>2</sup>

---

<sup>2</sup>Forrester Research, 2006 – often quoted but couldn't find a reference to the original study...

# Important search engines

- **Yahoo!** (1994): Initially a **directory**.
- **Excite** (1995).
- **AltaVista** (1995): In 1998, 20 machines, 130 GB RAM, 500 GB HDD, 13M queries/day (150 queries/second).<sup>3</sup>
- **Netscape** (1996): Yahoo!, Magellan, Lycos, Infoseek, Excite.
- **Ask Jeeves** (1996): Tentative **natural language** support.
- **Yandex** (1997): Popular in Russia (60%).<sup>4</sup>
- **Google Search** (1997): PageRank.
- **GoTo** (1998): Pay to appear for certain keywords.
- **MSN Search** (1998).
- **Baidu** (2000): Popular in China (63% of revenue).<sup>5</sup>

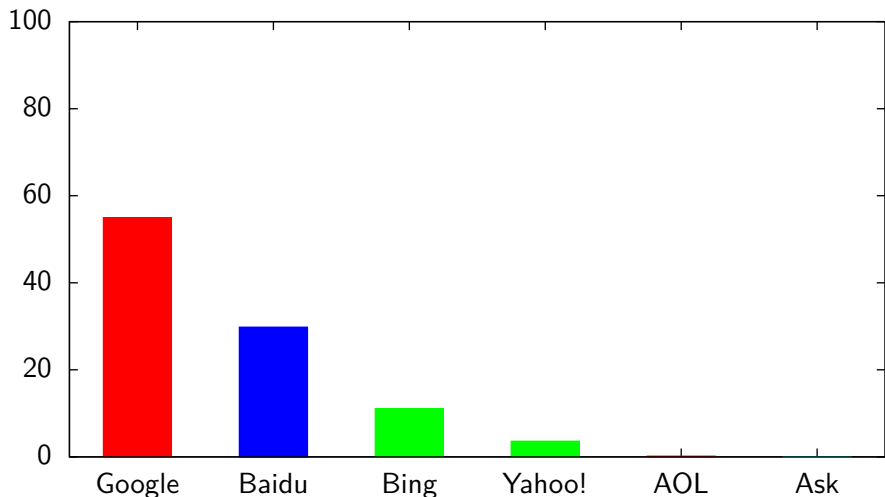
---

<sup>3</sup> Ricardo Baeza-Yates and Berthier Ribeiro-Neto (1999). *Modern Information Retrieval*. Addison-Wesley/ACM Press, pp. 374, 390.

<sup>4</sup> <http://www.liveinternet.ru/stat/ru/searches.html?slice=ru>

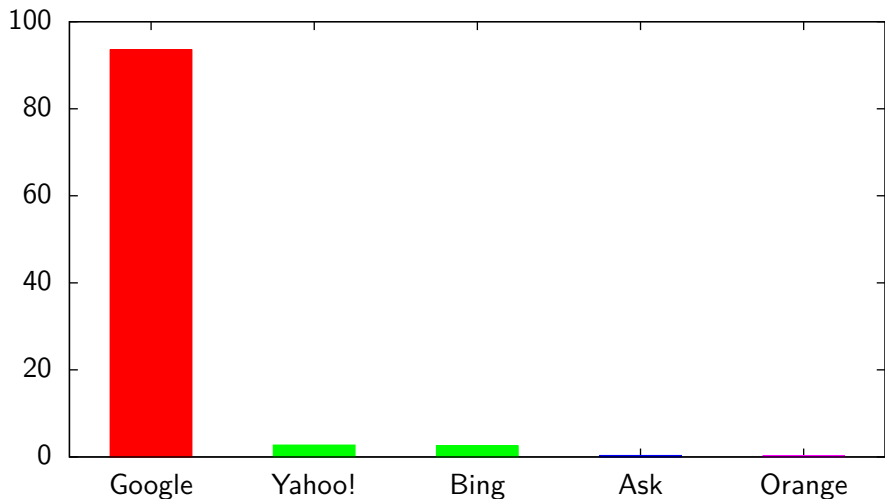
<sup>5</sup> [http://www.nytimes.com/2010/01/14/technology/companies/14baidu.html?\\_r=0](http://www.nytimes.com/2010/01/14/technology/companies/14baidu.html?_r=0)

# Search engine market share (world, October 2014)



Source: <https://www.netmarketshare.com/search-engine-market-share.aspx?qprid=4&qpcustomid=0>

# Search engine market share (France, October 2014)



Source: <http://www.atinternet.com/documents/barometre-des-moteurs-octobre-2014/>

# Table of contents

- 1 Introduction
- 2 Crawling**
- 3 Indexing
- 4 PageRank
- 5 Business
- 6 Innovation

# Crawling basics

- **Crawling**: automated exploration of the Web.
  - Software: crawler, spider, robot, bot.
- Start at a set of **initial** URLs.
- Explore by following **links** found on pages.
- **Scheduling** problem:
  - **Infinite** graph (dynamic content), no termination.
  - Need to **refresh** pages from time to time.
- **Hidden Web**, not reachable from links!
  - Hidden behind web **forms**.
  - `http://www.google.com/webmasters/tools/submit-url`
- **Common Crawl**: free crawl of 6 billion pages
  - Useful if you do not want to crawl (or cannot afford it).

# Finding new URLs

- In an HTML page:
  - Hyperlinks** `<a href="...">`
  - Media** ``, `<audio src="...">`,  
`<video src="...">`, `<source src="...">`
  - Frames** `<iframe src="...">`
  - JavaScript** `window.open("...")` – undecidable in general
  - Page text** Use **regexps**.
- In **other kinds** of files (PDFs...).
- In **sitemaps** provided specifically to crawlers.
- In **server logs** available online (could include Referer).
- From **existing URLs**: parent folder.
  - Also possible: change GET parameters...



# Duplicate pages

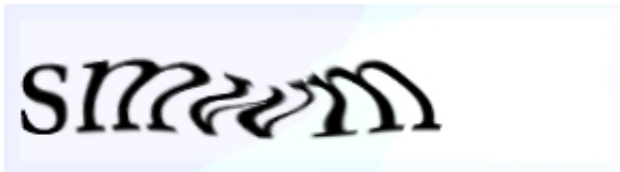
- Prevent **multiple indexing** and penalize **content farms**.
- Duplicate **URLs**: **canonicalization**.
  - `http://example.com:80/foo`
  - `http://example.com/foo/../bar`
  - `http://www.example.com/`
- Duplicate **pages**: use a **hash function**.
- **Near-duplicates** (dates, etc.): use a **similarity function**.
  - Must be **indexed!** e.g., Broder's **MinHash** (1997, used in AltaVista).
- Preferably, when you have multiple legitimate copies:
  - Use **301 redirects** to remove unneeded copies.
  - Specify the canonical copy using:  
`<link rel="canonical" href="http://example.com/">`

# Robot control (honor-based)

- **Robot Exclusion Standard:** `http://example.com/robots.txt`
    - Only at root level (not available for subfolders).
    - Filtering by **User-agent**.
    - Disallow directive to forbid certain pages.
    - Also: Allow, Crawl-delay, Host, Sitemap.
  - **HTTP header:** X-Robots-Tag (less support):
    - X-Robots-Tag: noindex
  - **Meta tag:** `<meta name="robots" content="noindex">`
    - Also nofollow, nosnipped, noarchive...
  - **Links:**
    - `<a href="secret/" rel="nofollow">`
  - **Engine-specific** interfaces (e.g., Google Webmaster Tools).
- **No guarantees!**

# Mandatory robot control with CAPTCHAs

How can we discriminate against robots?



- Completely Automated Public Turing test to tell Computers and Humans Apart (trademarked by CMU).
- Making a computer able to recognize humans.
- Accessibility problems (sound CAPTCHAs for the blind...).
- Used to digitize books (reCAPTCHA, Google) or show ads.
- Sweatshops, or relaying CAPTCHAs to your own website.

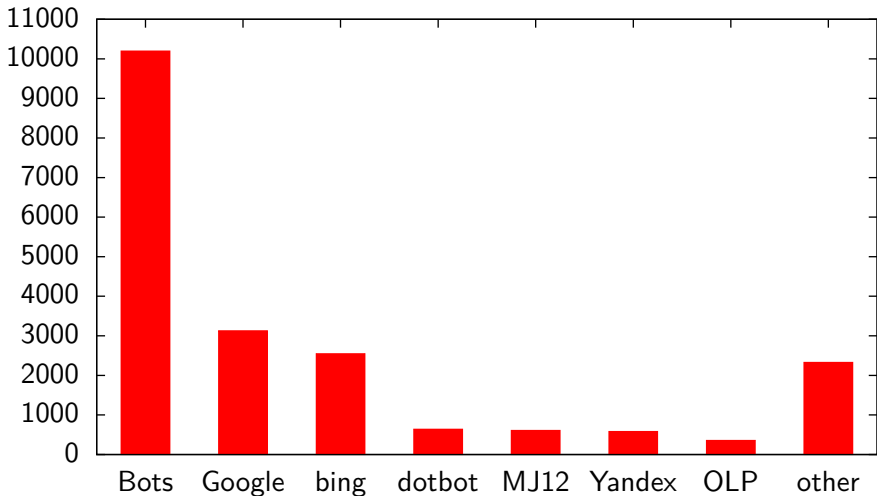
# Crawl scheduling

- Wait a minimal **delay** between requests to the same server.
  - Depends on the **server** (wikipedia.org vs your laptop).
  - Depends on the **resource** (large files...).
  - Generally, waiting at least **one second** is preferable.
- Requests to different servers can be **parallelized**.
- Requests should be run **asynchronously**.
- The HTTP connection should remain **open**.
- Requests can be **distributed** across multiple machines.
- Crawlers represent about 20% of Web traffic.<sup>6</sup>
  - On a3nm.net in September 2013: 16%.

---

<sup>6</sup><http://www.zdnet.com/blog/foremski/report-51-of-web-site-traffic-is-non-human-and-mostly-malicious/2201>

## Crawler traffic



a3nm.net traffic, November 2014 (out of 63175 requests).  
Bots identified by User-Agent containing "bot".

# Table of contents

- 1 Introduction
- 2 Crawling
- 3 Indexing**
- 4 PageRank
- 5 Business
- 6 Innovation

## Content to index

- First, the **contents** of the page.
- Importance could be **weighted**: headers, emphasized text...
- The `<title>`.
- The page **URL**.
- Also: **meta tags**:
  - Historically **indexed**, not anymore.
  - `<meta name="description" content="My website">`  
(used for Google snippets).<sup>7</sup>
  - `<meta name="keywords" content="great,website">`  
(not used by Google).
  - **Engine-specific**: `<meta name="google" ...>`.
- Not only Web pages! Various **file formats**.

---

<sup>7</sup><https://support.google.com/webmasters/answer/79812?hl=en>

# Language identification

- Use **meta-information** (may not be reliable).
- Look at **characters** (not specific enough).
- Look at the **frequent  $k$ -grams** and classify.
  - Surprisingly reliable.

Compare:

th awarting problecon inge pria notobe inguarguarew for andes so coluage to an  
th anciped the cons ors lessolat reatencle behas nuarocialso by in aselartion  
the re laturocalis com ad ble of th the in congto res wortionnetesers a th fory

with:

encer desemps our parcellecon le de grarts is bale diatincyclossibles ligne  
inats res ne de le catictiont pectionne cofon les des sour dand communes gral  
mation parchivell pedique de dontanduismes ce sour les pages ent inse



## Example document collection

- $d_1$  The jaguar is a New World mammal of the Felidae family.
- $d_2$  Jaguar has designed four new engines.
- $d_3$  For Jaguar, Atari was keen to use a 68K family device.
- $d_4$  The Jacksonville Jaguars are a professional US football team.
- $d_5$  Mac OS X Jaguar is available at a price of US \$199 for Apple's new "family pack".
- $d_6$  One such ruling family to incorporate the jaguar into their name is Jaguar Paw.
- $d_7$  It is a big cat.

# Tokenization

- Separate text into **words** (tokens).
- Challenges:
  - No **whitespace** to separate words (Chinese, Japanese...).
  - **Entities**: acronyms, numbers, URLs, emails, units, etc.
  - **Word boundaries** (“host name” or “hostname”?).
- Also eliminate **punctuation** and **case**.

## Example tokenization

*d*<sub>1</sub> the<sub>1</sub> jaguar<sub>2</sub> is<sub>3</sub> a<sub>4</sub> new<sub>5</sub> world<sub>6</sub> mammal<sub>7</sub> of<sub>8</sub> the<sub>9</sub> felidae<sub>10</sub> family<sub>11</sub>

*d*<sub>2</sub> jaguar<sub>1</sub> has<sub>2</sub> designed<sub>3</sub> four<sub>4</sub> new<sub>5</sub> engines<sub>6</sub>

*d*<sub>3</sub> for<sub>1</sub> jaguar<sub>2</sub> atari<sub>3</sub> was<sub>4</sub> keen<sub>5</sub> to<sub>6</sub> use<sub>7</sub> a<sub>8</sub> 68k<sub>9</sub> family<sub>10</sub> device<sub>11</sub>

*d*<sub>4</sub> the<sub>1</sub> jacksonville<sub>2</sub> jaguars<sub>3</sub> are<sub>4</sub> a<sub>5</sub> professional<sub>6</sub> us<sub>7</sub> football<sub>8</sub> team<sub>9</sub>

*d*<sub>5</sub> mac<sub>1</sub> os<sub>2</sub> x<sub>3</sub> jaguar<sub>4</sub> is<sub>5</sub> available<sub>6</sub> at<sub>7</sub> a<sub>8</sub> price<sub>9</sub> of<sub>10</sub> us<sub>11</sub> \$199<sub>12</sub>  
for<sub>13</sub> apple's<sub>14</sub> new<sub>15</sub> family<sub>16</sub> pack<sub>17</sub>

*d*<sub>6</sub> one<sub>1</sub> such<sub>2</sub> ruling<sub>3</sub> family<sub>4</sub> to<sub>5</sub> incorporate<sub>6</sub> the<sub>7</sub> jaguar<sub>8</sub> into<sub>9</sub>  
their<sub>10</sub> name<sub>11</sub> is<sub>12</sub> jaguar<sub>13</sub> paw<sub>14</sub>

*d*<sub>7</sub> it<sub>1</sub> is<sub>2</sub> a<sub>3</sub> big<sub>4</sub> cat<sub>5</sub>

# Stemming

- **Merge** similar words or word forms into **stems**.
- **Morphological**:
  - **Remove**: plural, gender, tense, mood...
  - **Irregularities**: “mouse”/“mice”.
  - **Ambiguities**: “stocking”, “rose”, “couvent” (French).
- **Lexical**:
  - **Derived terms**: “policy”, “politics”, “political”...
  - **Merge** synonyms or near-synonyms. (Not always relevant.)

## Example stemming

- d*<sub>1</sub> the<sub>1</sub> jaguar<sub>2</sub> **be**<sub>3</sub> a<sub>4</sub> new<sub>5</sub> world<sub>6</sub> mammal<sub>7</sub> of<sub>8</sub> the<sub>9</sub> felidae<sub>10</sub> family<sub>11</sub>
- d*<sub>2</sub> jaguar<sub>1</sub> **have**<sub>2</sub> **design**<sub>3</sub> four<sub>4</sub> new<sub>5</sub> **engine**<sub>6</sub>
- d*<sub>3</sub> for<sub>1</sub> jaguar<sub>2</sub> atari<sub>3</sub> **be**<sub>4</sub> keen<sub>5</sub> to<sub>6</sub> use<sub>7</sub> a<sub>8</sub> 68k<sub>9</sub> family<sub>10</sub> device<sub>11</sub>
- d*<sub>4</sub> the<sub>1</sub> jacksonville<sub>2</sub> **jaguar**<sub>3</sub> **be**<sub>4</sub> a<sub>5</sub> professional<sub>6</sub> us<sub>7</sub> football<sub>8</sub> team<sub>9</sub>
- d*<sub>5</sub> mac<sub>1</sub> os<sub>2</sub> x<sub>3</sub> jaguar<sub>4</sub> **be**<sub>5</sub> available<sub>6</sub> at<sub>7</sub> a<sub>8</sub> price<sub>9</sub> of<sub>10</sub> us<sub>11</sub> \$199<sub>12</sub>  
for<sub>13</sub> **apple**<sub>14</sub> new<sub>15</sub> family<sub>16</sub> pack<sub>17</sub>
- d*<sub>6</sub> one<sub>1</sub> such<sub>2</sub> **rule**<sub>3</sub> family<sub>4</sub> to<sub>5</sub> incorporate<sub>6</sub> the<sub>7</sub> jaguar<sub>8</sub> into<sub>9</sub>  
their<sub>10</sub> name<sub>11</sub> **be**<sub>12</sub> jaguar<sub>13</sub> paw<sub>14</sub>
- d*<sub>7</sub> it<sub>1</sub> **be**<sub>2</sub> a<sub>3</sub> big<sub>4</sub> cat<sub>5</sub>

# Stop word removal

- Remove **uninformative** words.
  - Storing the index takes **less space**.
  - Processing is **faster**.
- Words to remove:
  - Conjunctions** “that”, “and”...
  - Function verbs** “be”, “have”, “make”, ...
  - Determiners** “a”, “the”, “this”...

## Example stop word removal

- d*<sub>1</sub> jaguar<sub>2</sub> new<sub>5</sub> world<sub>6</sub> mammal<sub>7</sub> felidae<sub>10</sub> family<sub>11</sub>
- d*<sub>2</sub> jaguar<sub>1</sub> design<sub>3</sub> four<sub>4</sub> new<sub>5</sub> engine<sub>6</sub>
- d*<sub>3</sub> jaguar<sub>2</sub> atari<sub>3</sub> keen<sub>5</sub> 68k<sub>9</sub> family<sub>10</sub> device<sub>11</sub>
- d*<sub>4</sub> jacksonville<sub>2</sub> jaguar<sub>3</sub> professional<sub>6</sub> us<sub>7</sub> football<sub>8</sub> team<sub>9</sub>
- d*<sub>5</sub> mac<sub>1</sub> os<sub>2</sub> x<sub>3</sub> jaguar<sub>4</sub> available<sub>6</sub> price<sub>9</sub> us<sub>11</sub> \$199<sub>12</sub> apple<sub>14</sub>  
new<sub>15</sub> family<sub>16</sub> pack<sub>17</sub>
- d*<sub>6</sub> one<sub>1</sub> such<sub>2</sub> rule<sub>3</sub> family<sub>4</sub> incorporate<sub>6</sub> jaguar<sub>8</sub> their<sub>10</sub> name<sub>11</sub>  
jaguar<sub>13</sub> paw<sub>14</sub>
- d*<sub>7</sub> big<sub>4</sub> cat<sub>5</sub>

# Inverted index

- For each **term**, store the **document** where the term occurs.
- **Distribute** the index:
  - Each term is **mapped** to a machine responsible for it.
  - Use a **hash function** rather than a database.
- Of course, use **concise IDs** to represent the documents.
- Can store also the **occurrence position** in the index.
- Update the index in **bulk** (not step by step).



## Example inverted index

family  $d_1/11, d_3/10, d_5/16, d_6/4$   
football  $d_4/8$   
jaguar  $d_1/2, d_2/1, d_3/2, d_4/3, d_5/4, d_6/8 + 13$   
new  $d_1/5, d_2/5, d_5/15$   
rule  $d_6/3$   
us  $d_4/7, d_5/11$   
world  $d_1/6$   
...

# Relevance

- So far, we have focused on **Boolean** search:
  - A document **either** matches the conditions **or** doesn't.
- More fuzzy notion of **relevance** of a document for a query.
- Given a (human-given) reference set of relevant documents, we **evaluate** the system output as:

$$\text{precision} = \frac{|\text{reference}| \cap |\text{output}|}{|\text{output}|} \quad (1)$$

$$\text{recall} = \frac{|\text{reference}| \cap |\text{output}|}{|\text{reference}|} \quad (2)$$

- **Tradeoffs:**
  - Return **all documents**: high recall but low precision.
  - Return the **very best** document: high precision but low recall.
  - **F-measure** (harmonic mean of precision and recall).

# TF-IDF

- **Term frequency**  $tf(t, d)$ : proportion of terms in  $d$  that are  $t$ .  
→ The more  $t$  appears in  $d$ , the more  $d$  is **relevant** for  $t$ .
- **Inverse document frequency**  $idf(t)$ : log of the inverse proportion of documents where  $t$  occurs.  
→ The rarer  $t$  is, the most **informative** it is.
- **TF-IDF**:

$$tfidf(t, d) = \frac{n_{t,d}}{\sum_{t'} n_{t',d}} \cdot \log \left( \frac{|D|}{|\{d' \in D \mid n_{t,d'} > 0\}|} \right)$$

- Order documents by **decreasing** weight. (Most relevant first.)

# Example TF-IDF

family  $d_1/11/.13, d_3/10/.13, d_6/4/.08, d_5/16/.07$   
football  $d_4/8/.47$   
jaguar  $d_1/2/.04, d_2/1/.04, d_3/2/.04, d_4/3/.04, d_6/8 + 13/.04,$   
 $d_5/4/.02$   
new  $d_2/5/.24, d_1/5/.20, d_5/15/.10$   
rule  $d_6/3/.28$   
us  $d_4/7/.30, d_5/11/.15$   
world  $d_1/6/.47$   
...

# Queries

- **Single-keyword**: use the index.
- **Multi-keyword**:
  - Combine the indexes for the various keywords.
  - Use set operations (intersection, union, difference).
  - Combine the scores somehow.
- For **top-K**, we can just look at the **top** of the relevant lists (threshold algorithm).
- **Position queries**: postprocess the results to filter them.

# MapReduce

- Framework for **distributed computing**.
- Published by **Google** in 2004: model and proprietary implementation.
- Apache **Hadoop**: open source implementation.
- Express your task in terms of:
  - **Map** Operations to run in parallel on each record.
  - **Reduce** Operations to aggregate records.
- The MapReduce implementation takes care of:
  - **Scheduling** the various operations.
  - **Dispatching** the jobs to the various workers.
  - **Monitoring** the jobs and **restarting** failed jobs.
  - **Moving** the computation results around.

# TF-IDF in MapReduce

```
function map(id, document)
  foreach distinct token in tokenize(document)
    stem := stemming(term)
    if (stem not in stopWords)
      tf := count(term, document)/length(document)
      output (term, (id, tf))

function reduce(term, list)
  foreach (id, tf) in list
    output (term, tf * log(nbDocuments/length(list)))
```

# Table of contents

- 1 Introduction
- 2 Crawling
- 3 Indexing
- 4 PageRank**
- 5 Business
- 6 Innovation

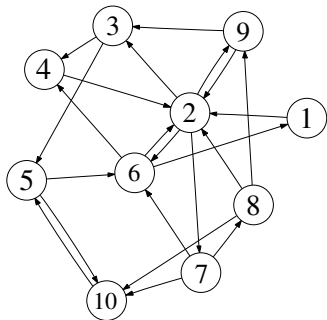


# PageRank intuition

- Developed by **Larry Page** and **Sergey Brin**, 1996. (Initially, a research project at **Stanford**.)
  - **Patent** (assigned to Stanford University, but Google has exclusive license rights).
  - Related: **hubs and authorities**.
- **Idea**: Use the **links** on the Web to estimate the **importance** of pages. (Not just the page content!)
- Precompute this **independently** from the query.
- (Also: distribute it massively on **cheap** machines.)

# Web graph

- Vertices: **pages**.
- Edges: **links**.
- Transition **matrix**: row  $i$  and column  $j$  is the weight from  $i$  to  $j$ , namely  $1/n$  with  $n$  the number of outgoing links of  $i$ .

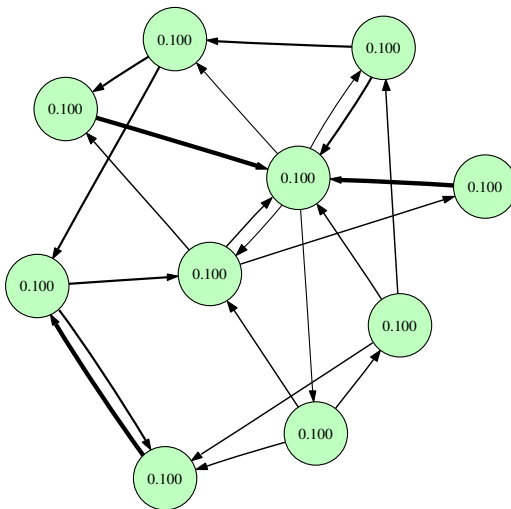


$$G = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{4} & 0 & 0 & \frac{1}{4} & \frac{1}{4} & 0 & \frac{1}{4} & 0 \\ 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & \frac{1}{2} \\ \frac{1}{3} & \frac{1}{3} & 0 & \frac{1}{3} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{3} & 0 & \frac{1}{3} & 0 & \frac{1}{3} \\ 0 & \frac{1}{3} & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{3} & \frac{1}{3} \\ 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

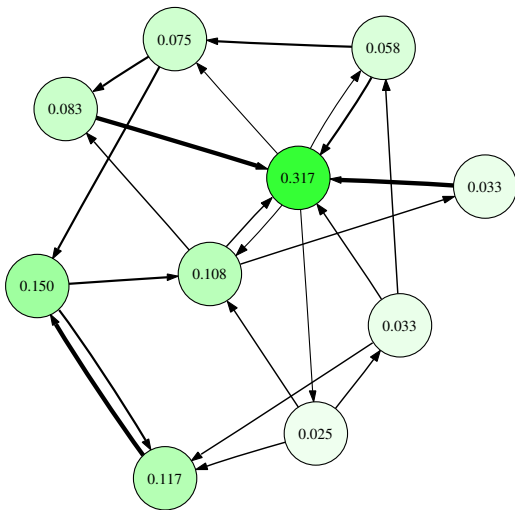
# PageRank

- Give a **query-independent** importance **weight** to pages.
- Mathematically:
  - **Dominant eigenvector** of the transition matrix.
  - **Stationary distribution** of the associated Markov chain.
- Intuitively:
  - The **probability** that a random surfer is currently on the page.
  - The **proportion** of the random surfer's time spent on this page.
- Computationally:
  - Start with a **uniform** importance weight.
  - Iteratively:  $w(p) = \sum_{q \rightarrow p} M(q, p)w(q)$ .
- May not **converge** but we can add a **damping factor**: the surfer jumps to a random page with a low probability.
- Bonus: the **text** of backlinks gives clues about the relevance of the page for specific queries.

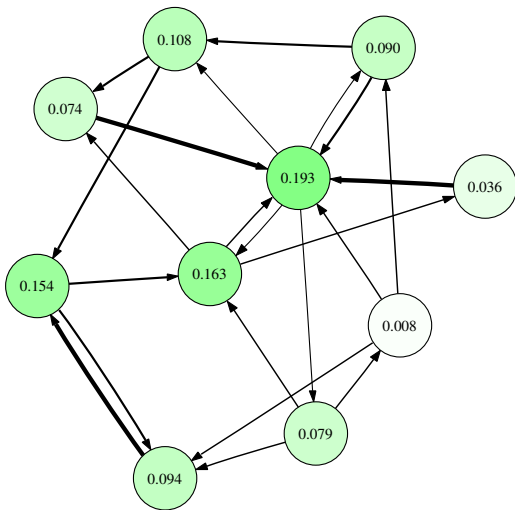
# PageRank example



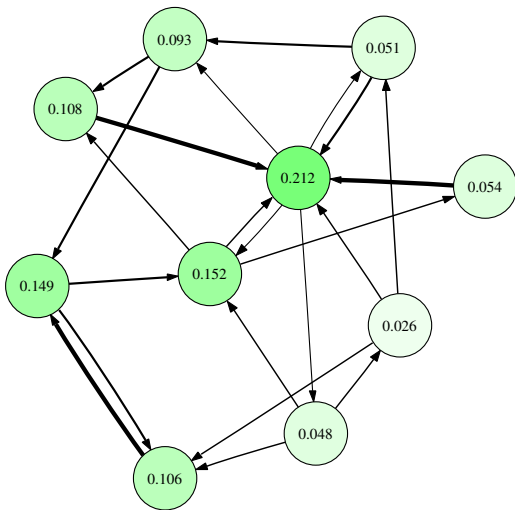
# PageRank example



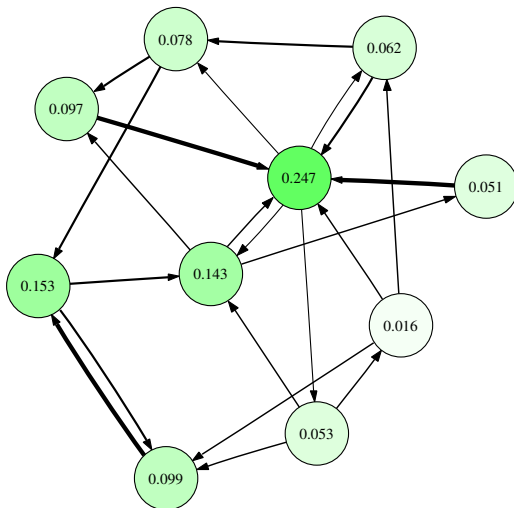
# PageRank example



# PageRank example

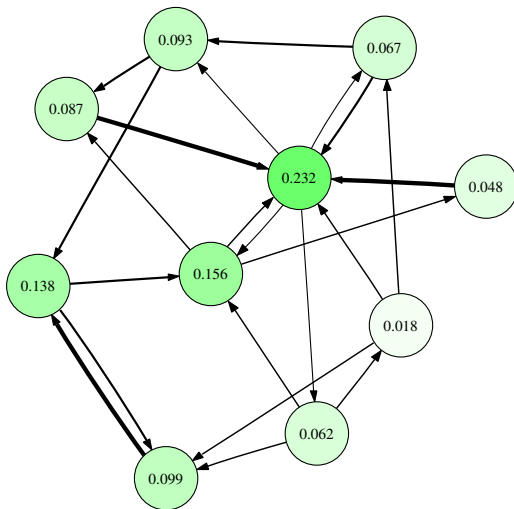


# PageRank example

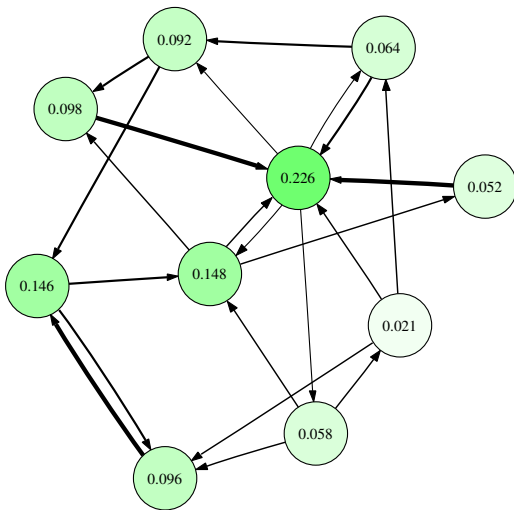




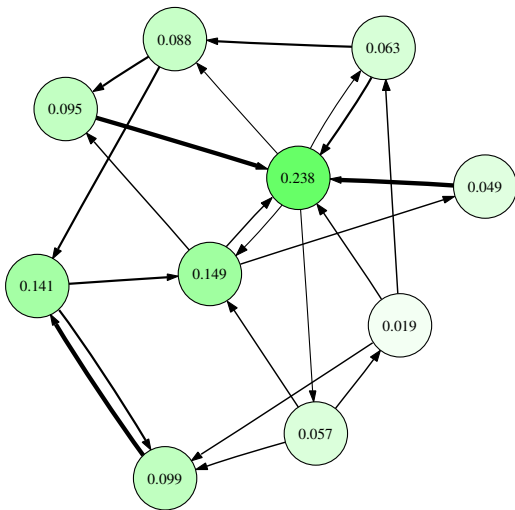
# PageRank example



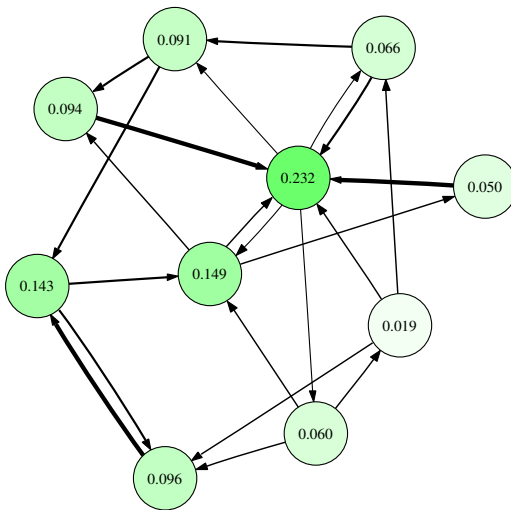
# PageRank example



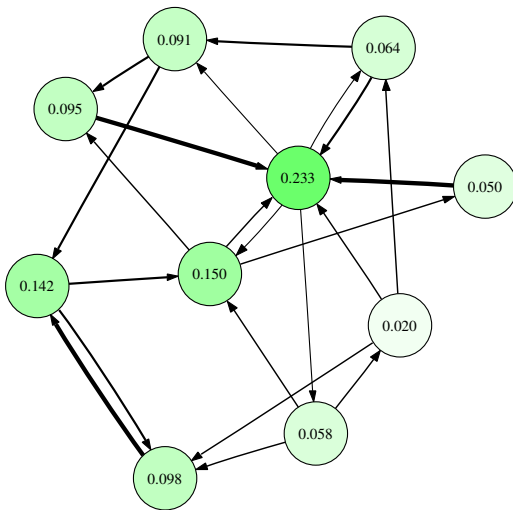
# PageRank example



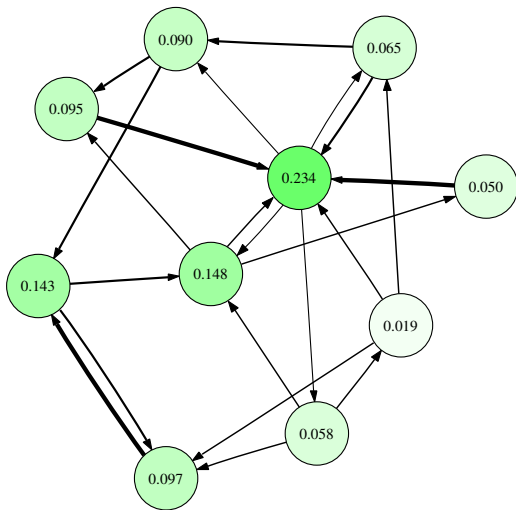
# PageRank example



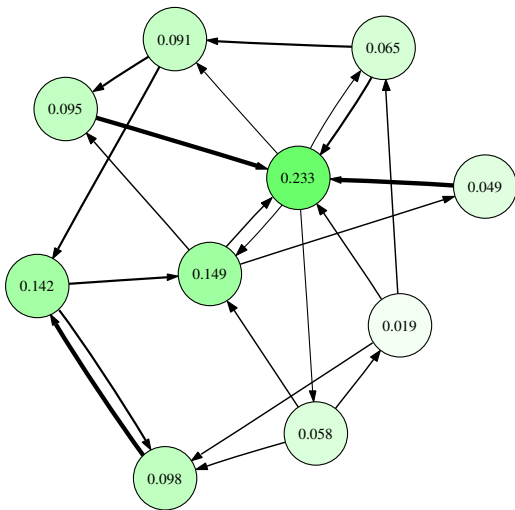
# PageRank example



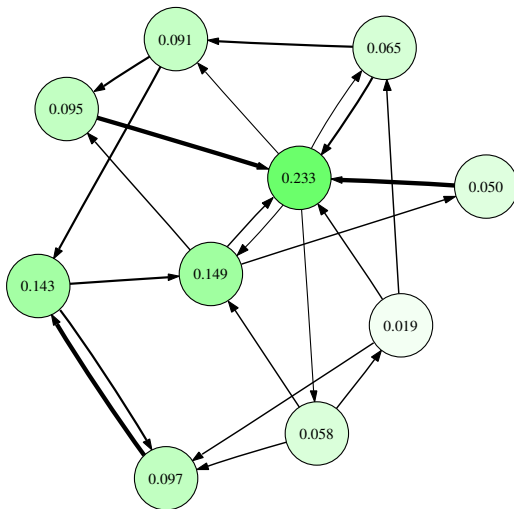
# PageRank example



# PageRank example

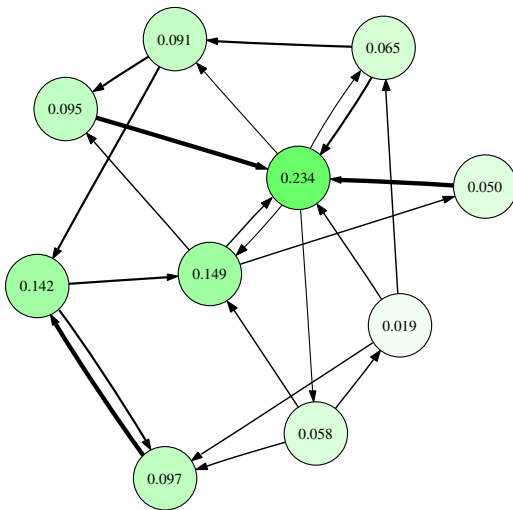


# PageRank example





# PageRank example



# Table of contents

- 1 Introduction
- 2 Crawling
- 3 Indexing
- 4 PageRank
- 5 Business**
- 6 Innovation

# Organic and paid results

- Two kinds of results:
  - **Organic** Automatically generated from the Web.
  - **Paid** Auctioned to publishers.
- Organic results are the **impartial** output of algorithms and cannot be **bought** (however, who chooses the algorithms?).
- Paid results are just **advertising** (online advertising: 36 billion dollar revenue in the US in 2012<sup>8</sup>):
  - **Targeting** based on the keywords (of course) but also the geographical location, the type of device...
  - **Cost per click** (CPC): announcers are billed based on clicks, the search engine tries to optimize revenue.
  - **Cost per action** (CPA): announcers are billed based on conversions (registration, sale, etc.).
  - Risks of **abuse** (click farming...).

---

<sup>8</sup>[http://www.iab.net/media/file/IAB\\_Internet\\_Advertising\\_Revenue\\_Report\\_FY\\_2012\\_rev.pdf](http://www.iab.net/media/file/IAB_Internet_Advertising_Revenue_Report_FY_2012_rev.pdf)

# SEO

- Details of search engine algorithms are **proprietary**.
  - They are not **fixed** with time: large impact on businesses.
  - Many attempts to prevent **abuse** and remove spam from results (Google Panda, Google Penguin).
  - **Manual** spam filtering, requests for reconsideration.
- Search engines are certainly **not just** as PageRank + TF-IDF!
- **White hat** techniques, or recommended practices.
  - **Black hat** techniques, discouraged by search engines.

## General SEO advice

- Make sure that all pages can be reached following **static links**.
- Ensure that your site contains appropriate **keywords**.
- Use **text** rather than images or Flash.
- Use **alt** text for images (also for accessibility).
- Have a **clean** URL structure (no messy GET parameters).
- Avoid being **hacked**.
- Remove user-generated **spam**.

## PageRank tweaks (from white to black)

- When changing a website's URL, a **301 redirect** must be put in place to transfer PageRank.
- Links with `rel="nofollow"` are not **accounted for** in PageRank: use them to link without **endorsing**.
- **Buying** links on high-PageRank websites, or **exchanging** links.
- **Inserting links** in forum posts, wikis, etc.
  - Bots to submit **spammy** content in all forms they find.
- Create backlinks with **misleading** text: **Google bombing**.

## Indexing tweaks (from grey to black)

- Putting keywords in the **URL**:  
`http://example.com/articles/42-a-long-title.`
- **Keyword stuffing** in the page.
- Abusing CSS, JavaScript, redirects, etc., to make crawlers **see** different content.
- Using the User-Agent, IP ranges, etc. to **serve** different content to crawlers.
- **Scrape** existing content, optionally rewriting it (automatically or manually), or translating it.

# Table of contents

- 1 Introduction
- 2 Crawling
- 3 Indexing
- 4 PageRank
- 5 Business
- 6 Innovation**



# Domain search and enterprise search

- Indexes relevant to a specific **domain**:
  - Focused crawling.
  - Information extraction from important sources.
- **Enterprise** search engines.
  - Search appliances.
- Example: **Exalead** (French company, 2000).

# Publish/subscribe

- **Crawling** wastes resources and induces delays between page update and indexing.
- A **push notification** can indicate to a search engine that content has changed.
- Notifications can be **propagated** between interested parties.
- **Manual** resubmission in Google Webmaster Tools.
- **Shady** third-party services.
- **PubSubHubBub** for RSS.

# Deep Web

- Pages that have no **links** to them.
- For instance, **result pages** from a search.
- 2001 estimate: the deep Web is hundreds of times larger than the reachable Web.<sup>9</sup>
- **Web form probing**:
  - Need to figure out form **constraints**.
  - Need to come up with **keywords**.
  - Idea: **feed back** words from the website into the form.
- **Risks?** (Semantics of POST.)

---

<sup>9</sup>Bergman, Michael K (August 2001). "The Deep Web: Surfacing Hidden Value". *The Journal of Electronic Publishing* 7 (1)

# Multimedia indexing

- Other things than **text** on the Web.
- Historical approach: index multimedia content using **text clues**: surrounding text, file names, alt text, etc.
  - <http://google.com/search?tbm=isch&q=241543903>
- Better: index the **content** of an image (or video, or speech).
  - **Voxalead**: search in **videos**, uses text to speech.
- Search by **image**: Google Images, Google Goggles.
- Search by **sound**:
  - query by **humming**: Musipedia, SoundHound.
  - query by **fingerprint**: Shazam.

## P2P search

- Usual search engines have a **centralized** index (and a proprietary back-end).
  - **P2P search**: distribute the crawling and indexing between (untrusted) **peers**.
  - Propagate **queries** to the various peers to answer them.
  - Also used on **darknets**: RetroShare.
  - Current implementations: **Seeks**, **YaCy**.
- Doesn't work very well yet.

# Meta search engine

- Run queries through **multiple** search engines.
- **Aggregate** the results.
- Also known as **federated search**.
- **Historically**: Copernic Desktop Search, Ixquick...
- **Harder** nowadays: there aren't many independent search engines...

# User feedback

- If a user **clicks** on a search result, it means that it is **relevant**.  
→ **Track** response page clicks.
- Other possible signals (pure speculation):
  - **New search** from the same user?
  - **Time** spent on a page (Google Analytics?)
  - Feedback from **social media** (Like button, Google +1).

[example.com - Wikipedia, the free encyclopedia](#)

en.wikipedia.org/wiki/**Example.com** ▾

**Example.com**, **example.net**, **example.org**, and **example.edu** are second-level domain names reserved for documentation purposes and **examples** of the use of ...

[RFC 2606 - IETF](#)

google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=2&...FQjCNGOxLfOSn8hS

# Search result personalisation

- Rank results **differently** depending on the user.
  - **Language** (crucial).
  - **Position** (especially on mobile).
  - **Past searches** (Google Web History, no account required).
  - **Social graph** (searches by friends, feedback from friends).
  - **Similar users** (clustering).
- Risk of a **filter bubble**:
  - You see content that only **confirms** your preferences.
  - No **alternative** viewpoints.
  - No **objective** notion of relevance.
- Search engines which **advertise** the absence of filtering (DuckDuckGo, Startpage.com by Ixquick...).



# Structured data

- Can be **extracted** from webpages automatically.
- Can be **provided** by the website publisher: Microformats, Microdata, RDFa.

```
<div itemscope itemtype="http://data-vocabulary.org/Event">  
  <a href="http://www.example.com/events/rms" itemprop="url">  
    <span itemprop="summary">A free digital society</span>  
  </a>  
    
  <!-- ... -->  
</div>
```

## [Tacoma Dome Calendar - Tacoma Dome Events | Eventful](#) 🔍

View **Tacoma Dome's** upcoming **event** schedule and profile - Tacoma, WA. City/neighborhood: Tacoma Disabled access: No obstacles to access.

Wed, Jun 29 [Britney Spears - Jessie and ...](#) - Tacoma Dome, Tacoma, WA, 98421

Fri, Jul 8 [Matthew Morrison](#) - Tacoma Dome, Tacoma, WA, 98421

Fri, Jul 8 [NKOTBSB](#) - Tacoma Dome, Tacoma, WA, 98421

[eventful.com/tacoma/venues/tacoma-dome-/V0-001.../events](#) - Cached - Similar

# Zero-click information

- Calculator, pronunciation, definitions, unit conversion...
- Wikipedia-like **infoboxes**:
  - Google Knowledge Graph
  - DuckDuckGo

The screenshot shows a DuckDuckGo search interface. The search bar contains the text "telecom paristech" and features a duck logo on the left, a search button with a magnifying glass, and a "More" dropdown menu. Below the search bar, a knowledge panel for "Télécom ParisTech" is displayed. The panel includes a title, a brief description: "Télécom ParisTech (also known as ENST or Télécom or École nationale supérieure des télécommunications) is one of the most prestigious and selective grandes écoles in France.", and two links: "More at Wikipedia" and "École nationale supérieure des télécommunications de Bretagne". To the right of the text is the Telecom ParisTech logo, which consists of the text "TELECOM ParisTech" above a red graphic of a building facade.

# Query language

- Identify **typos** in queries:
  - **Edit distance** approaches (spellchecking).
  - **Statistical** approaches: two queries with small edit distance in quick succession by the same user.
- Query **autocompletion**.
- **Natural language** queries.
- **Voice recognition**: Google Voice Search, Siri...
- Refine a query, use current query as **context**.
- Targeting **specific websites**:
  - e.g. “!wiki Search engine” on DuckDuckGo.
- Structured data visualization: **WolframAlpha**.