Technologies du Web
Master COMASIC
Search Engines

Antoine Amarilli[1]

October 10, 2013

[1] 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.[2]

[2] Forrester Research, 2006 – often quoted but couldn't find a reference to the original study...

Important search engines

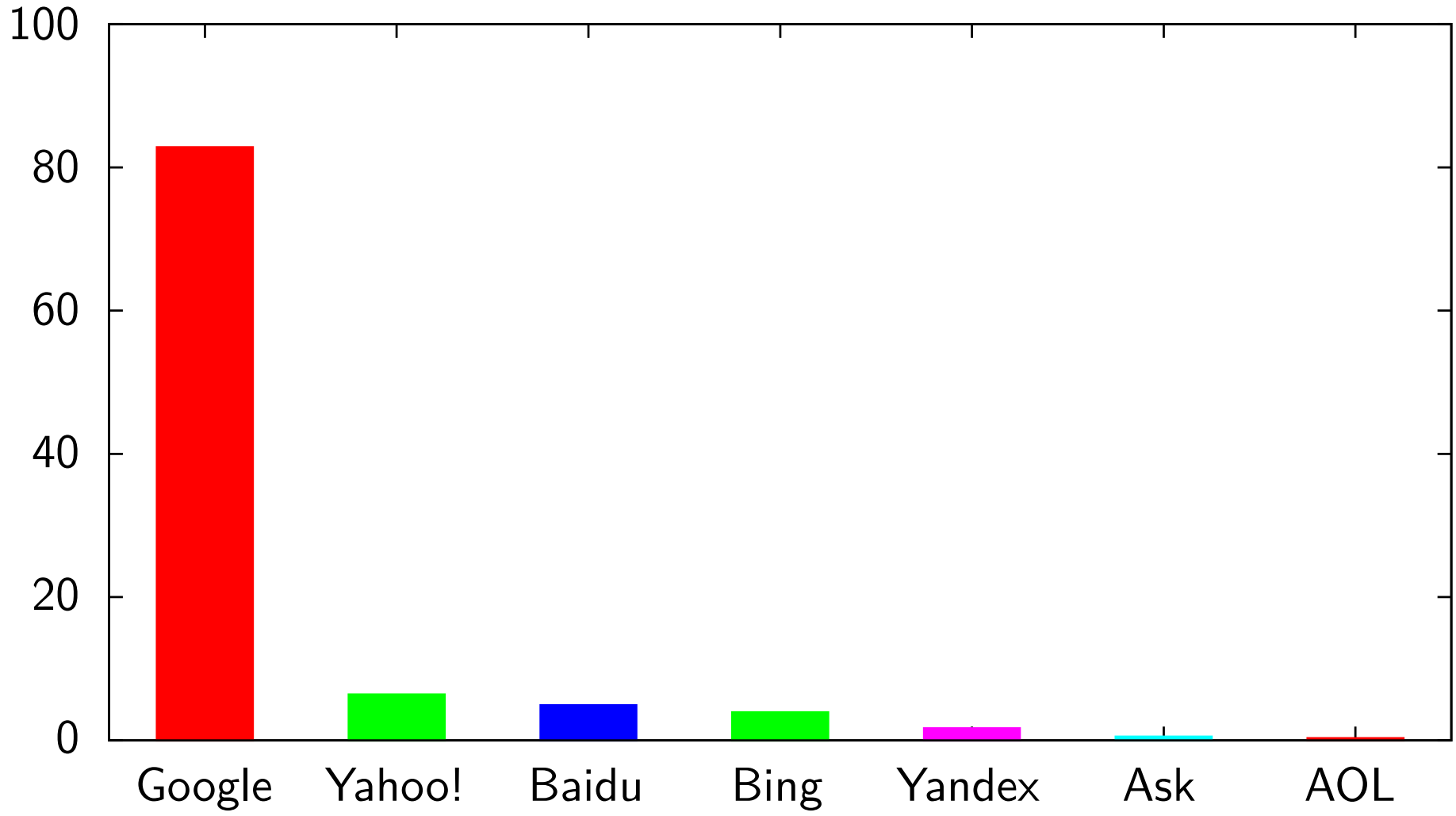
- Yahoo! (1994): Initially a directory.
- Excite (1995).
- AltaVista (1995): In 1998, 20 machines, 130 GB RAM, 500 GB HDD, 13M queries/day (150 queries/second).[3]
- Netscape (1996): Yahoo!, Magellan, Lycos, Infoseek, Excite.
- Ask Jeeves (1996): Tentative natural language support.
- Yandex (1997): Popular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

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

[4] `http://www.liveinternet.ru/stat/ru/searches.html?slice=ru`

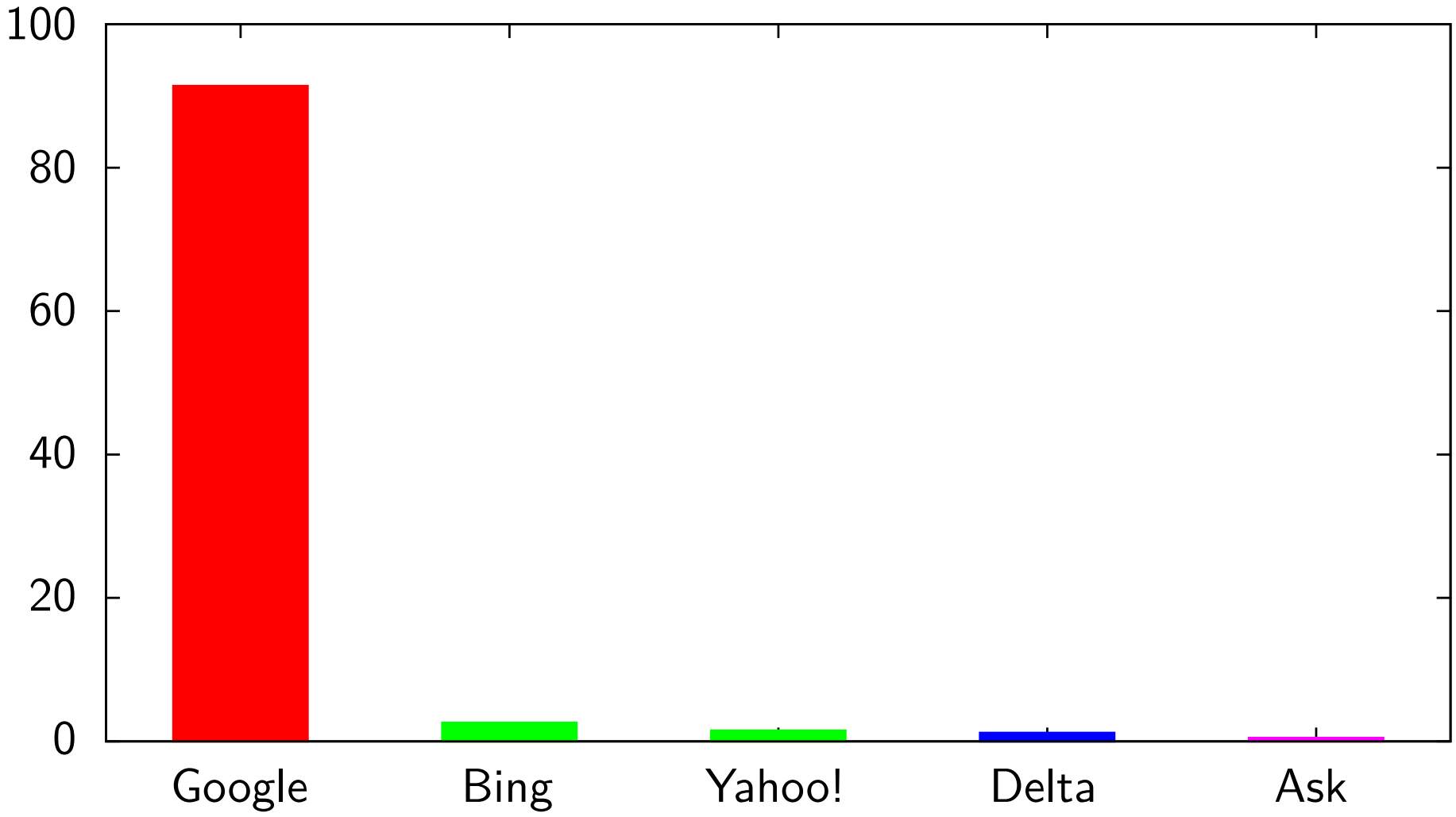
[5] `http://www.nytimes.com/2010/01/14/technology/companies/14baidu.html?_r=0`

Search engine market share (world, May 2011)



Source: https://en.wikipedia.org/w/index.php?title=Web_search_engine&oldid=575728900#Market_share

Search engine market share (France, August 2013)



Source: http://www.atinternet.com/documents/barometre-des-moteurs-de-recherche-aout-2013/

Table of contents

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 `<img src="...">`, `<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?

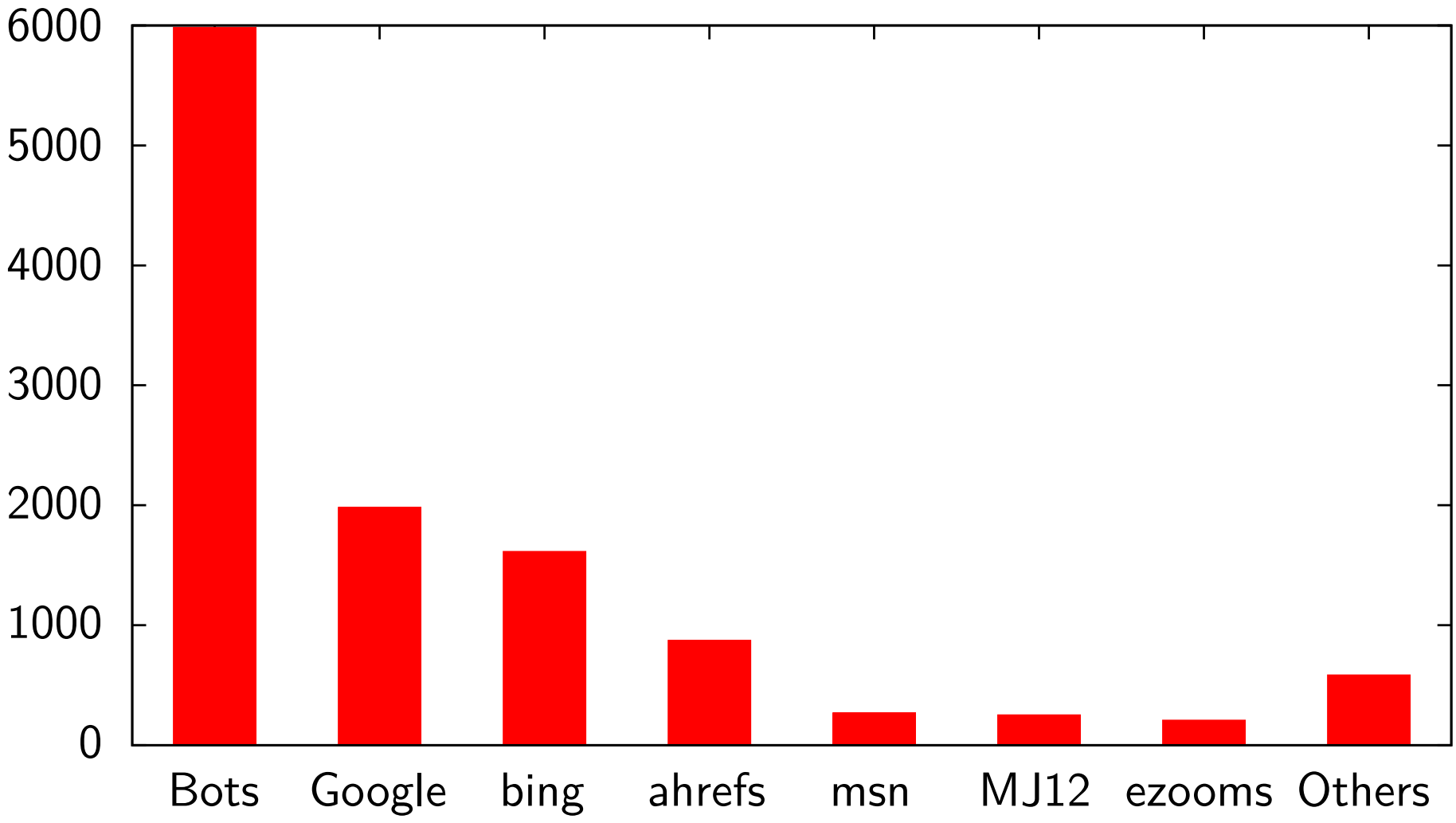
- Compteletly Automated Public Turing test to tell Computers and Humans Apart (trademarked by CMU).
- Making a computer able to recognize humans.
- Accessibility problems (sound CAPTCHAs for the blind...).
- Used to digitize books (reCAPTCHA, Google) or show ads.
- Sweatshops, or relaying CAPTCHAs to 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.[6]
  - ⇒ On `a3nm.net` in September 2013: 16%.

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

Crawler traffic



`a3nm.net` traffic, September 2013 (out of 36593 requests).
Bots identified by a User-Agent including a contact email or URL.

Table of contents

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).[7]
  - <meta name="keywords" content="great,website"> (not used by Google).
  - Engine-specific: <meta name="google" ...>.
- Not only Web pages! Various file formats.

[7] 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.
  $\Rightarrow$ 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_1 the$_1$ jaguar$_2$ is$_3$ a$_4$ new$_5$ world$_6$ mammal$_7$ of$_8$ the$_9$ felidae$_{10}$ family$_{11}$

d_2 jaguar$_1$ has$_2$ designed$_3$ four$_4$ new$_5$ engines$_6$

d_3 for$_1$ jaguar$_2$ atari$_3$ was$_4$ keen$_5$ to$_6$ use$_7$ a$_8$ 68k$_9$ family$_{10}$ device$_{11}$

d_4 the$_1$ jacksonville$_2$ jaguars$_3$ are$_4$ a$_5$ professional$_6$ us$_7$ football$_8$ team$_9$

d_5 mac$_1$ os$_2$ x$_3$ jaguar$_4$ is$_5$ available$_6$ at$_7$ a$_8$ price$_9$ of$_{10}$ us$_{11}$ \$199$_{12}$ for$_{13}$ apple's$_{14}$ new$_{15}$ family$_{16}$ pack$_{17}$

d_6 one$_1$ such$_2$ ruling$_3$ family$_4$ to$_5$ incorporate$_6$ the$_7$ jaguar$_8$ into$_9$ their$_{10}$ name$_{11}$ is$_{12}$ jaguar$_{13}$ paw$_{14}$

d_7 it$_1$ is$_2$ a$_3$ big$_4$ cat$_5$

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_1 the$_1$ jaguar$_2$ be$_3$ a$_4$ new$_5$ world$_6$ mammal$_7$ of$_8$ the$_9$ felidae$_{10}$ family$_{11}$

d_2 jaguar$_1$ have$_2$ design$_3$ four$_4$ new$_5$ engine$_6$

d_3 for$_1$ jaguar$_2$ atari$_3$ be$_4$ keen$_5$ to$_6$ use$_7$ a$_8$ 68k$_9$ family$_{10}$ device$_{11}$

d_4 the$_1$ jacksonville$_2$ jaguar$_3$ be$_4$ a$_5$ professional$_6$ us$_7$ football$_8$ team$_9$

d_5 mac$_1$ os$_2$ x$_3$ jaguar$_4$ be$_5$ available$_6$ at$_7$ a$_8$ price$_9$ of$_{10}$ us$_{11}$ \$199$_{12}$ for$_{13}$ apple$_{14}$ new$_{15}$ family$_{16}$ pack$_{17}$

d_6 one$_1$ such$_2$ rule$_3$ family$_4$ to$_5$ incorporate$_6$ the$_7$ jaguar$_8$ into$_9$ their$_{10}$ name$_{11}$ be$_{12}$ jaguar$_{13}$ paw$_{14}$

d_7 it$_1$ be$_2$ a$_3$ big$_4$ cat$_5$

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_1 jaguar$_2$ new$_5$ world$_6$ mammal$_7$ felidae$_{10}$ family$_{11}$
d_2 jaguar$_1$ design$_3$ four$_4$ new$_5$ engine$_6$
d_3 jaguar$_2$ atari$_3$ keen$_5$ 68k$_9$ family$_{10}$ device$_{11}$
d_4 jacksonville$_2$ jaguar$_3$ professional$_6$ us$_7$ football$_8$ team$_9$
d_5 mac$_1$ os$_2$ x$_3$ jaguar$_4$ available$_6$ price$_9$ us$_{11}$ \$199$_{12}$ apple$_{14}$ new$_{15}$ family$_{16}$ pack$_{17}$
d_6 one$_1$ such$_2$ rule$_3$ family$_4$ incorporate$_6$ jaguar$_8$ their$_{10}$ name$_{11}$ jaguar$_{13}$ paw$_{14}$
d_7 big$_4$ cat$_5$

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:
  $\Rightarrow$ 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.

  $\Rightarrow$ F-measure (harmonic mean of precision and recall).

TF-IDF

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

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

- 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

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.

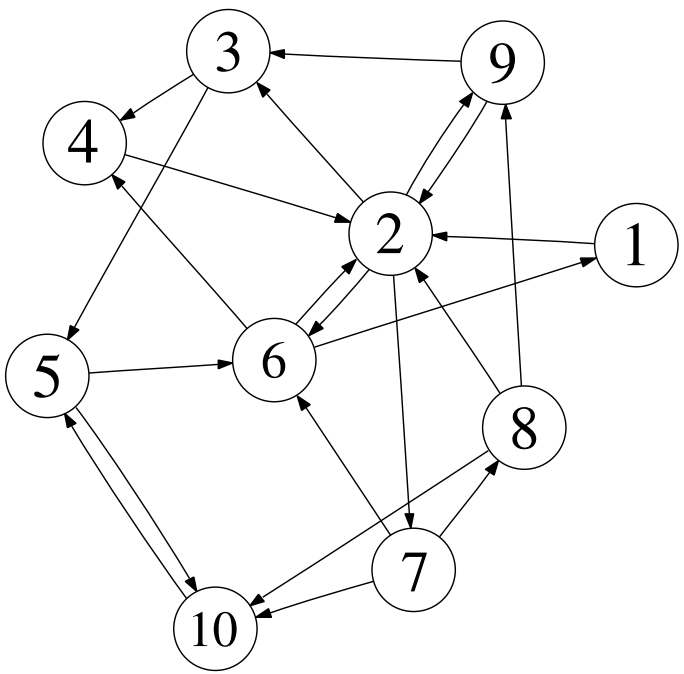
$\Rightarrow$ Idea: Use the links on the Web to estimate the importance of pages. (Not just the page content!)

$\Rightarrow$ Precompute this independently from the query.

$\Rightarrow$ (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.

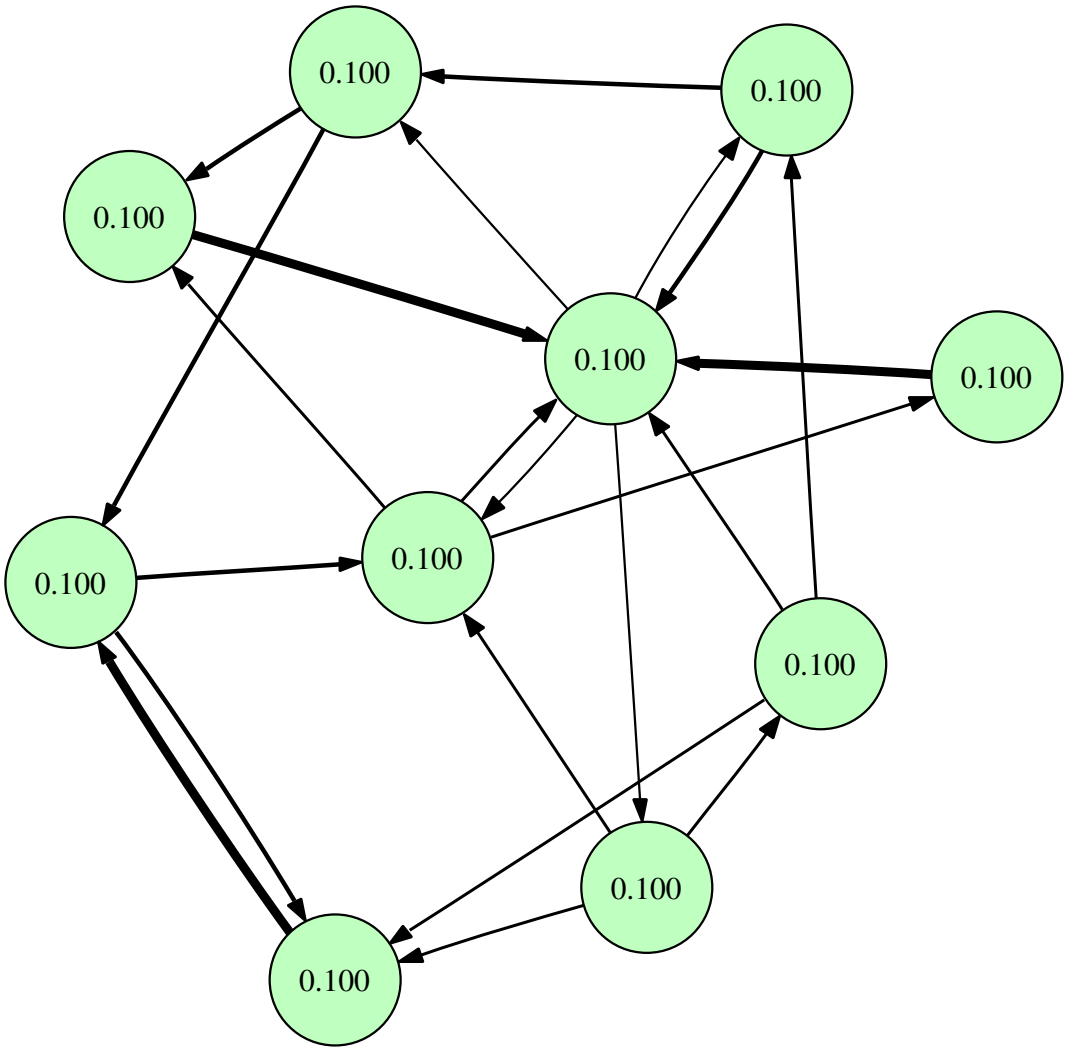


$$G = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{4} & 0 & 0 & \frac{1}{4} & \frac{1}{4} & 0 & \frac{1}{4} & 0 \\ 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & \frac{1}{2}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

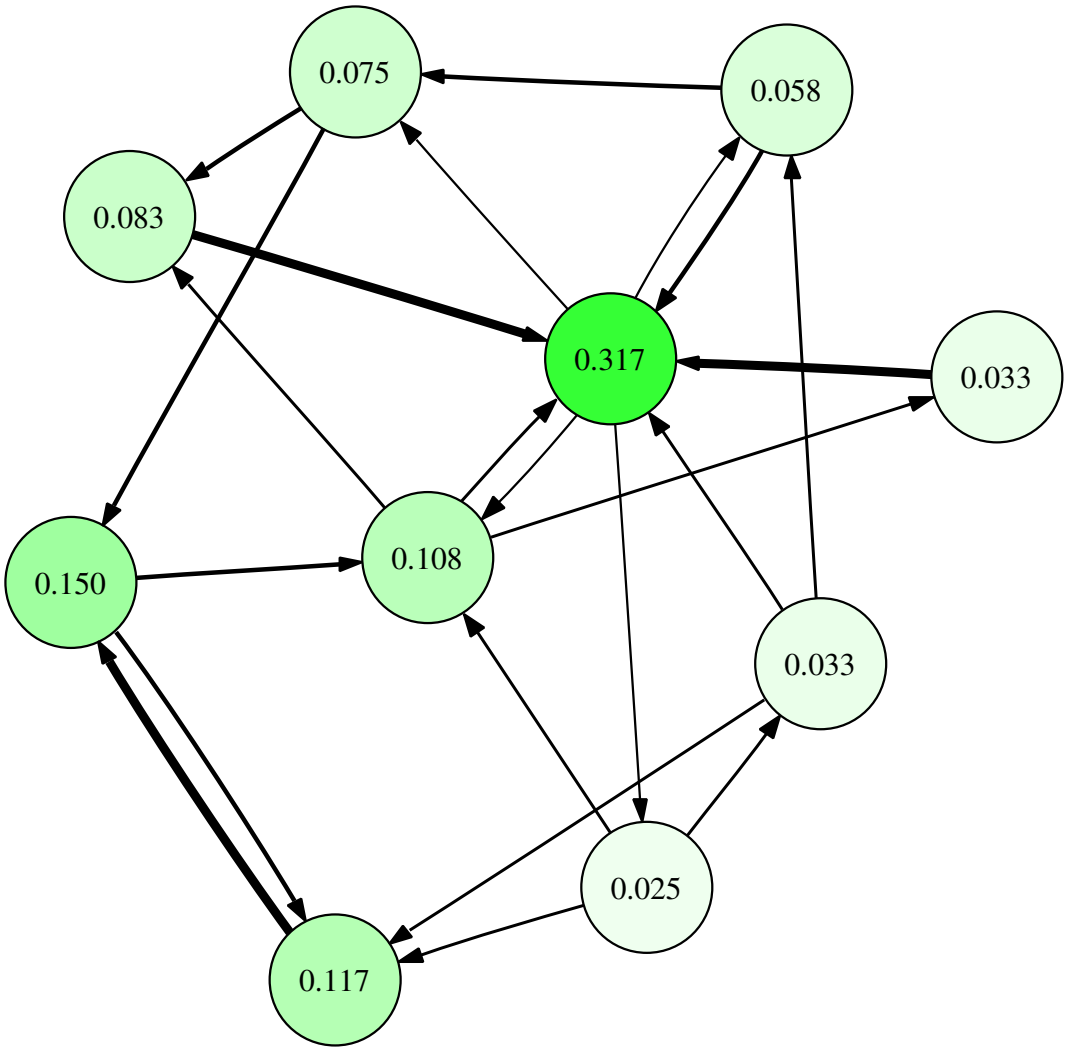
PageRank

- 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: $w(p) = \sum_{q \to p} M(q, p) w(q)$.
- 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.

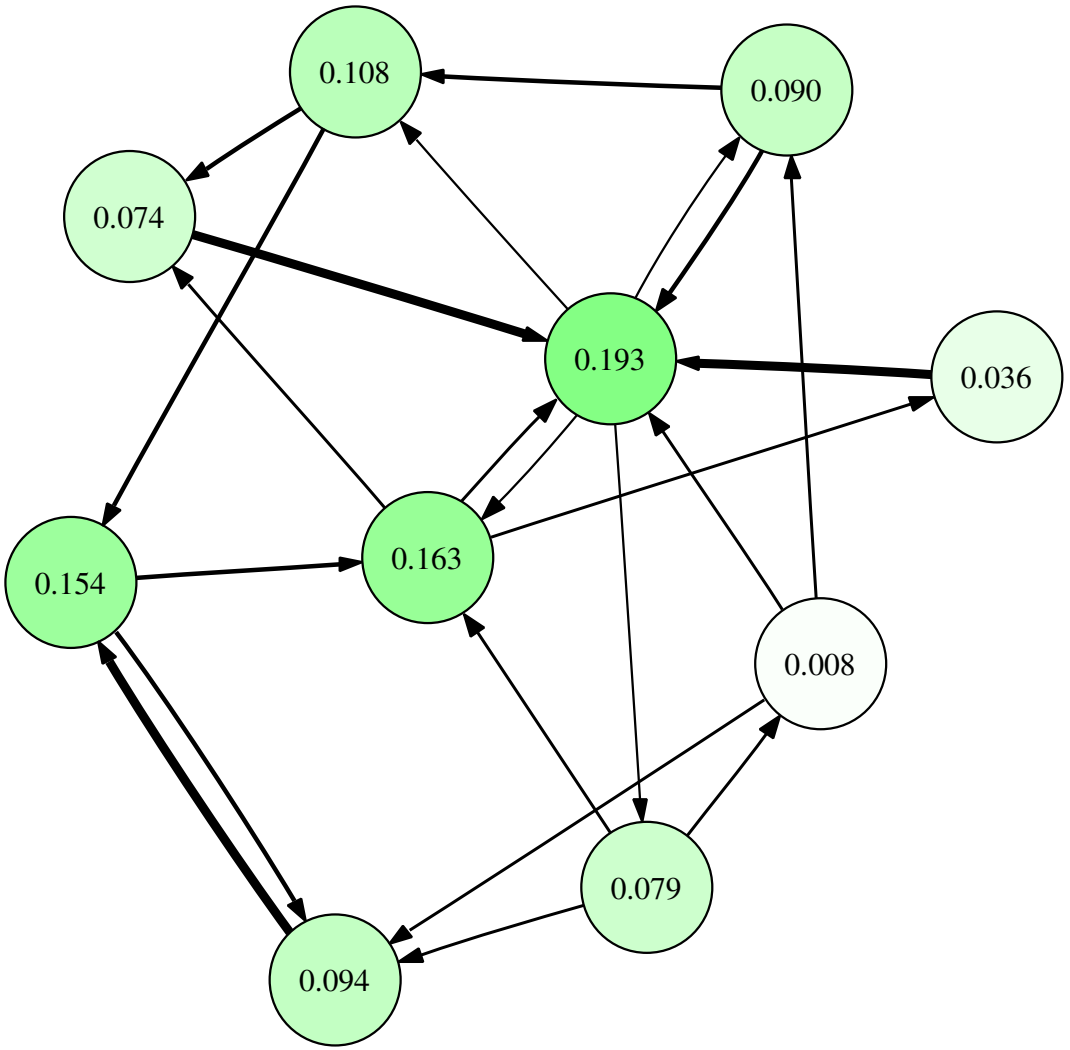
PageRank example



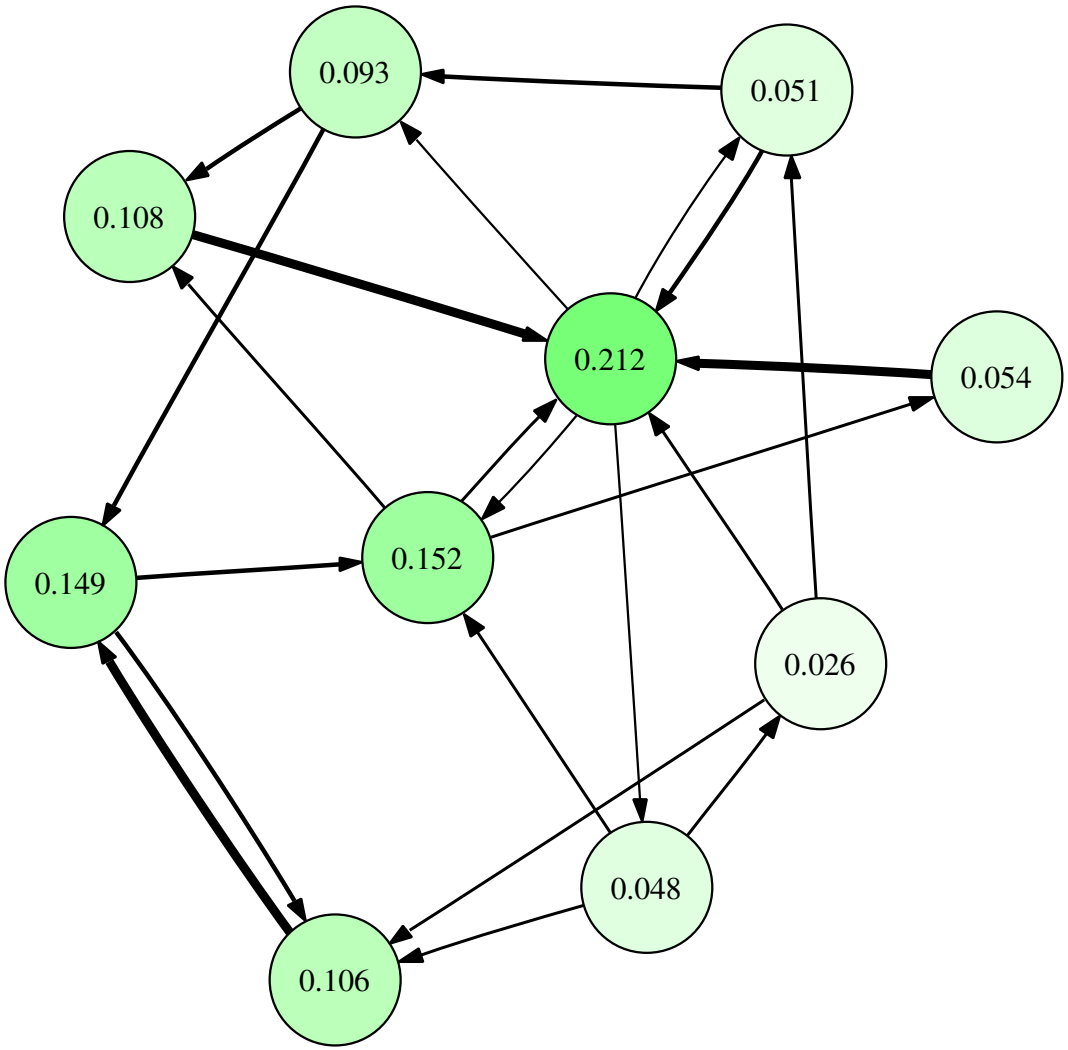
PageRank example



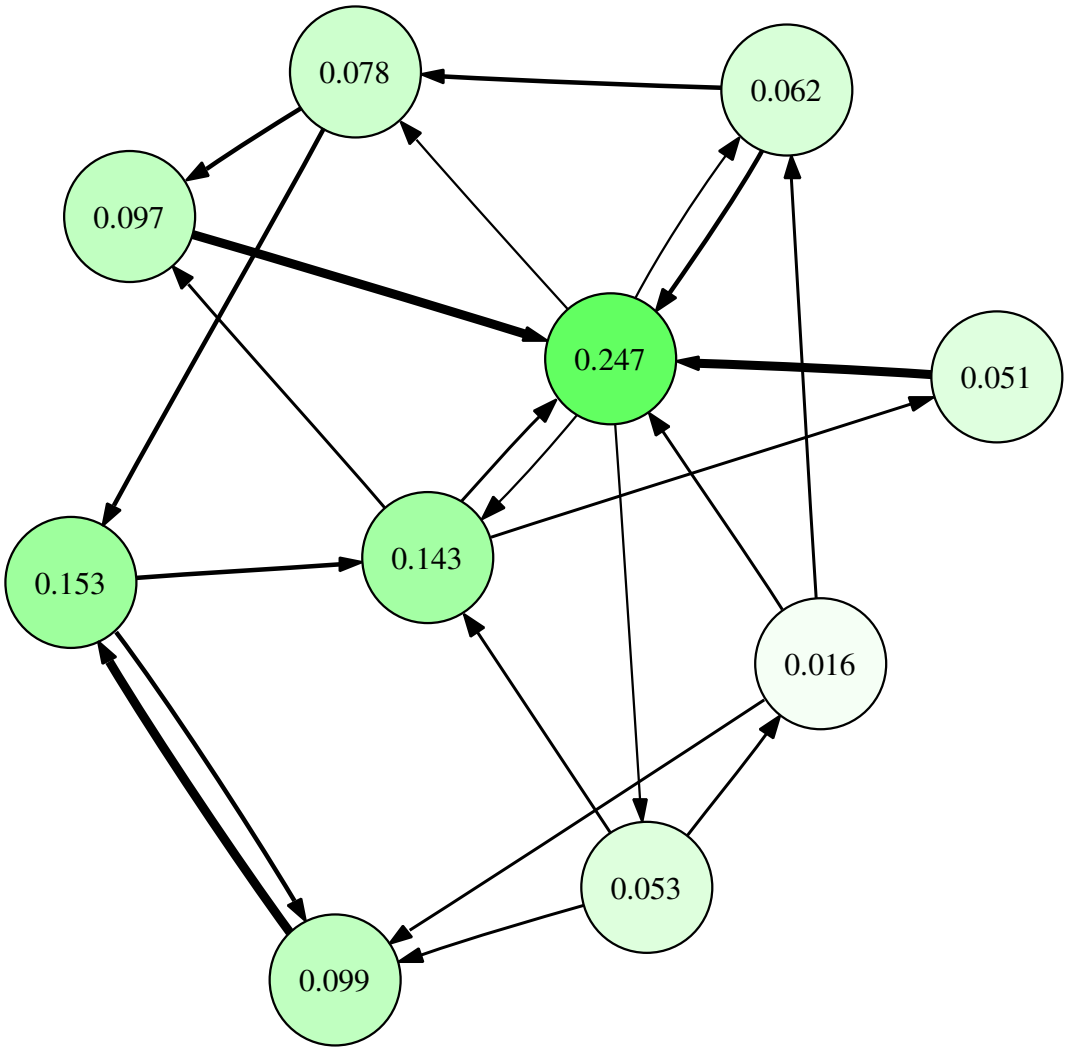
PageRank example



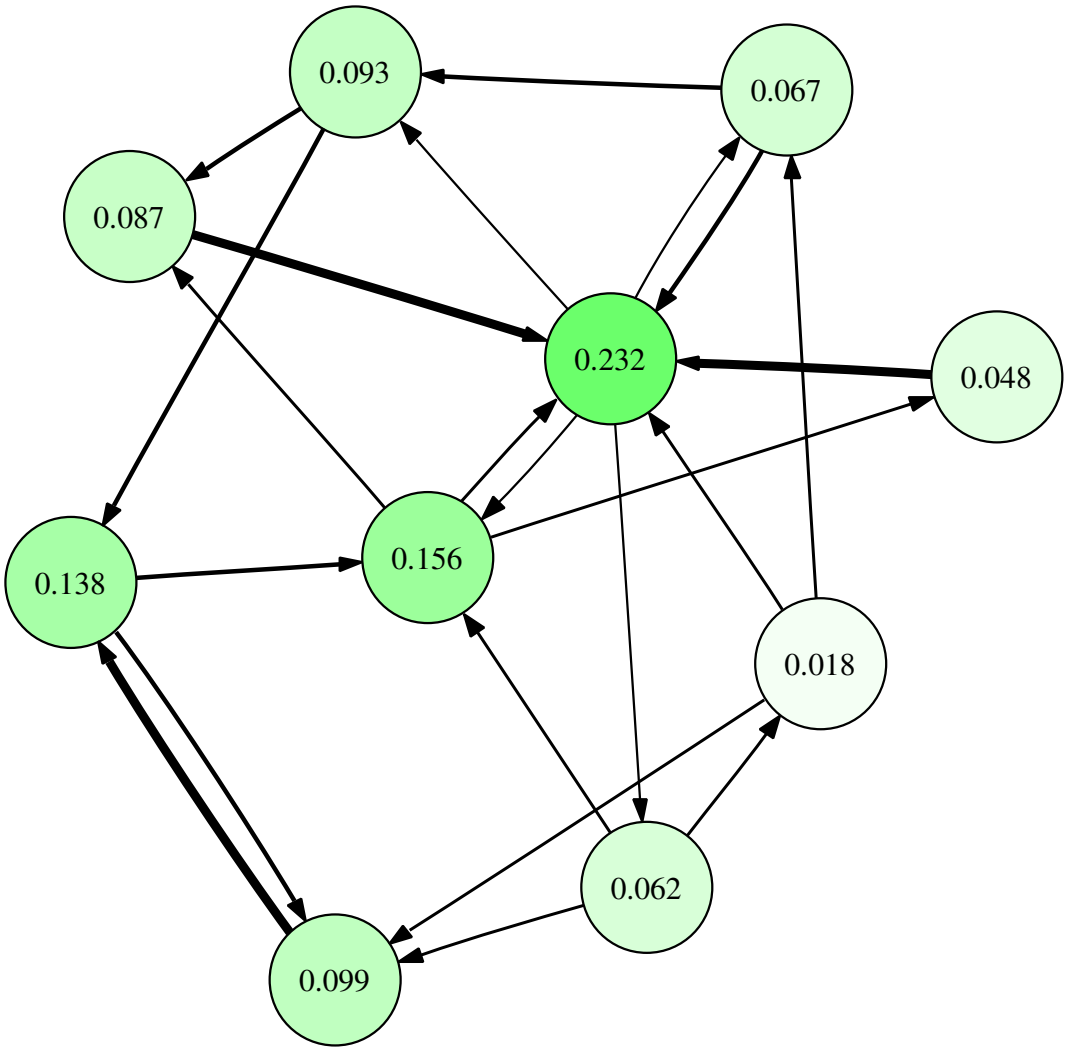
PageRank example



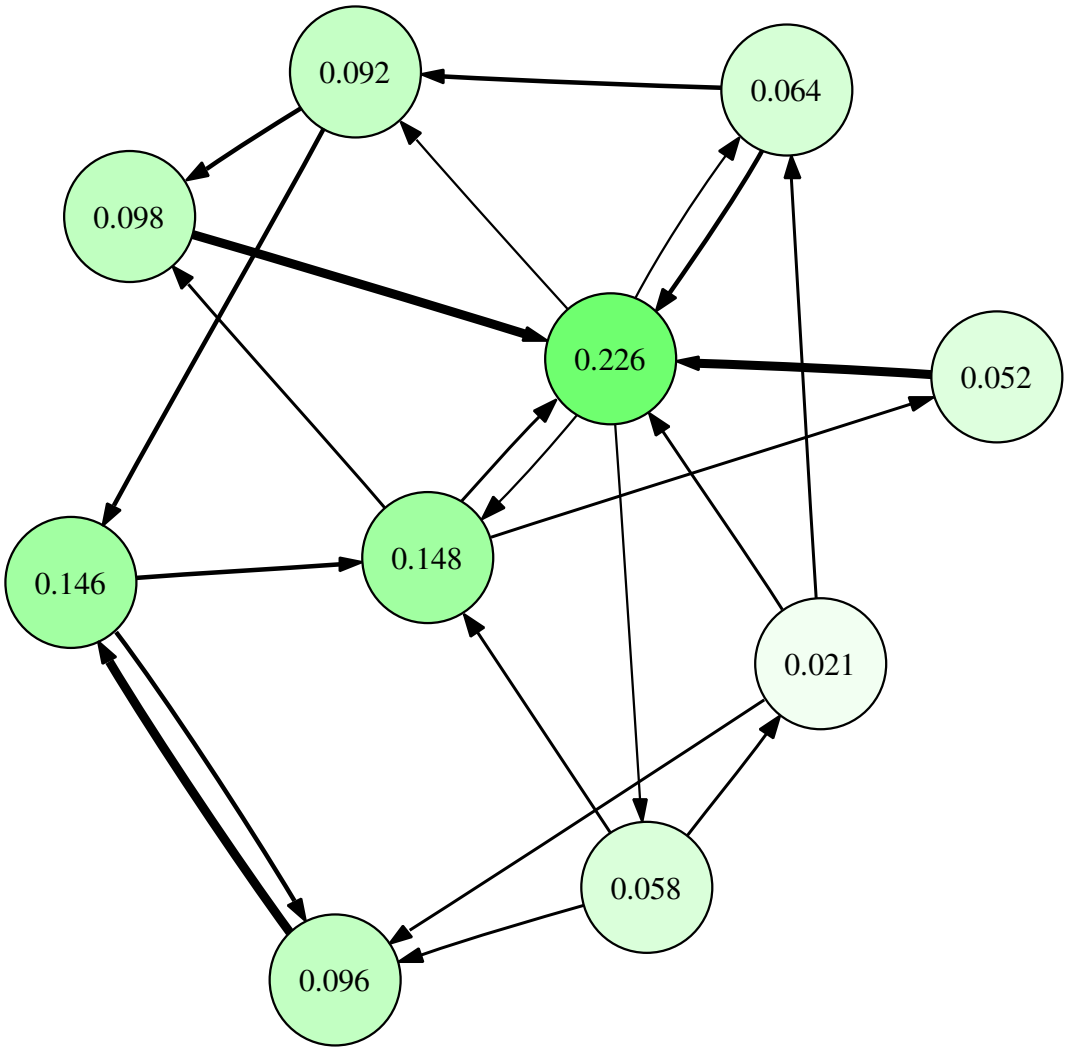
PageRank example



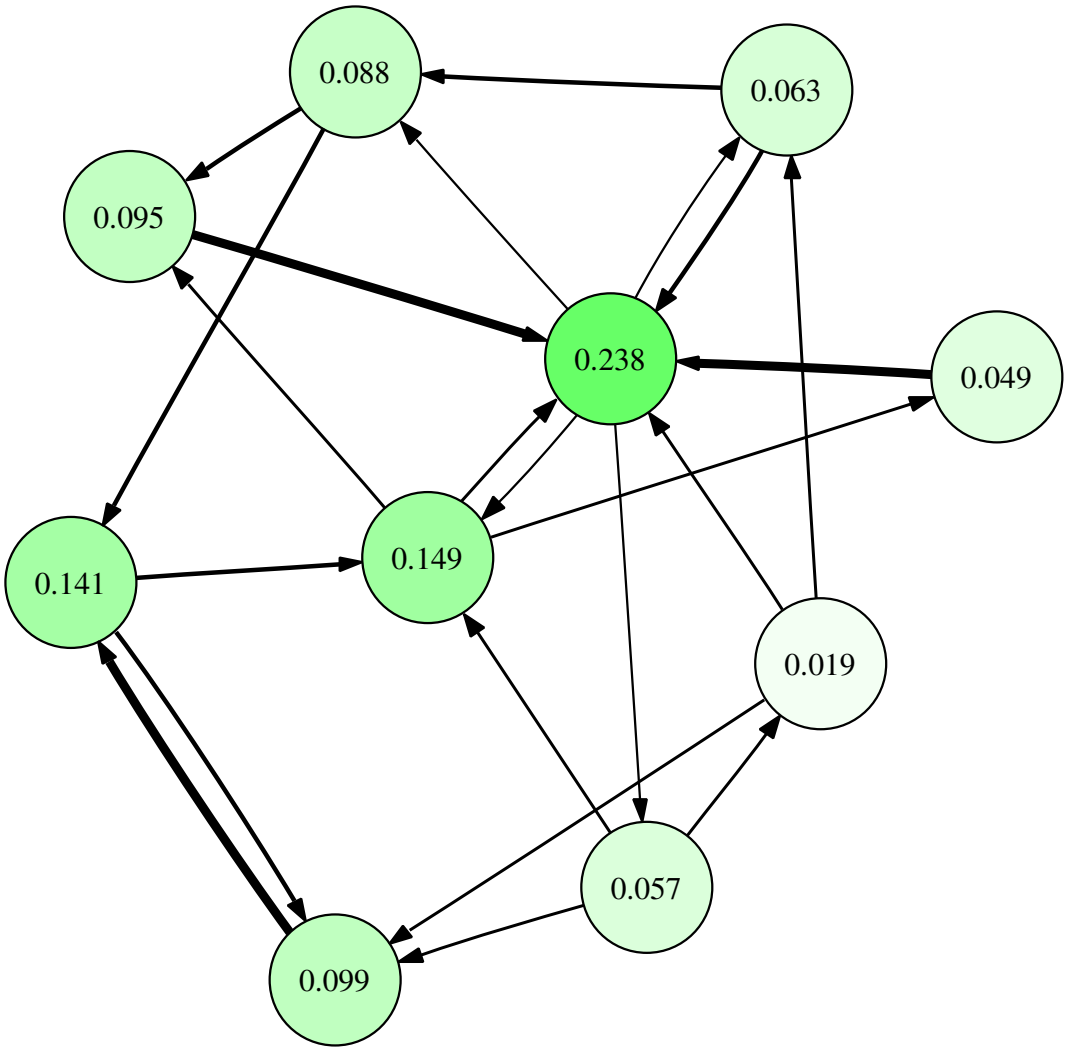
PageRank example



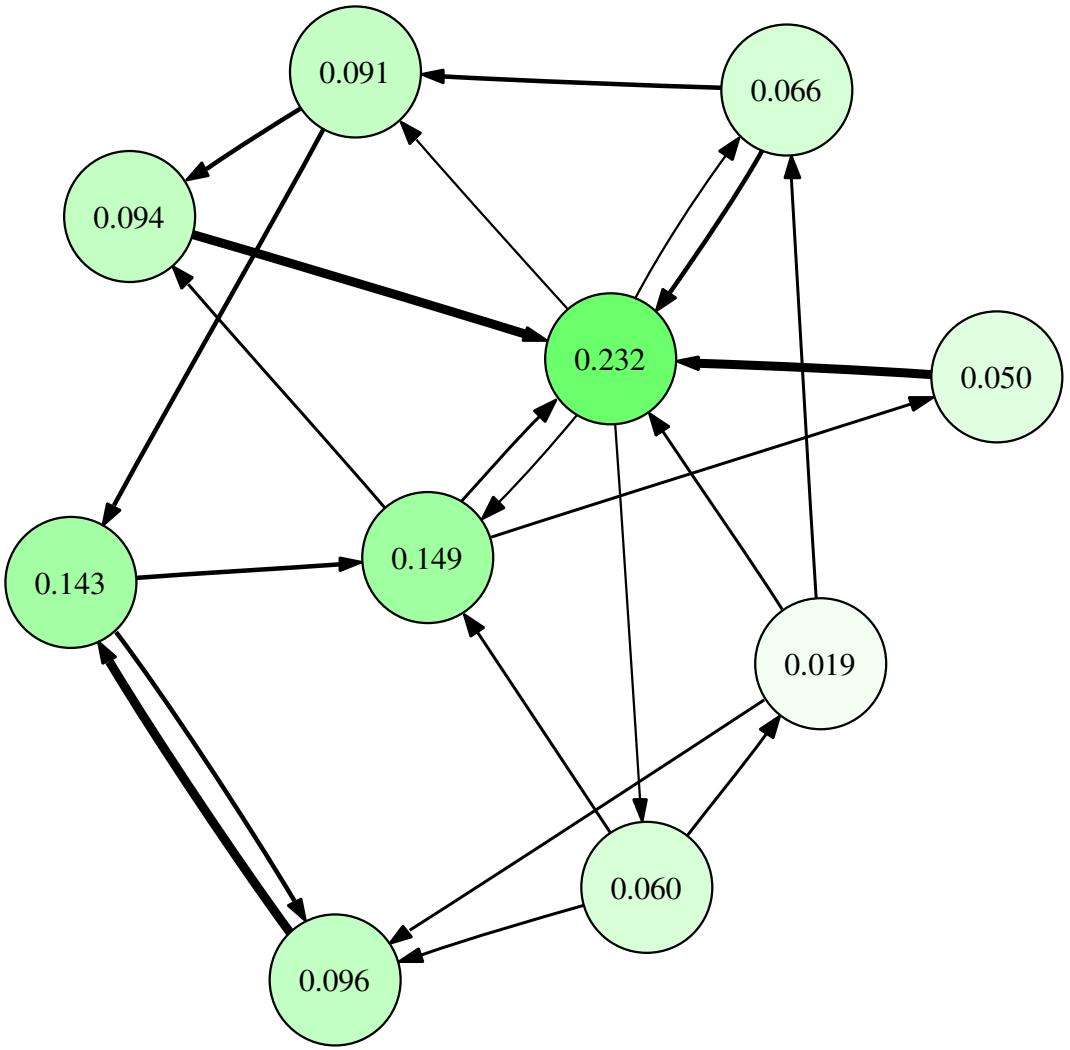
PageRank example



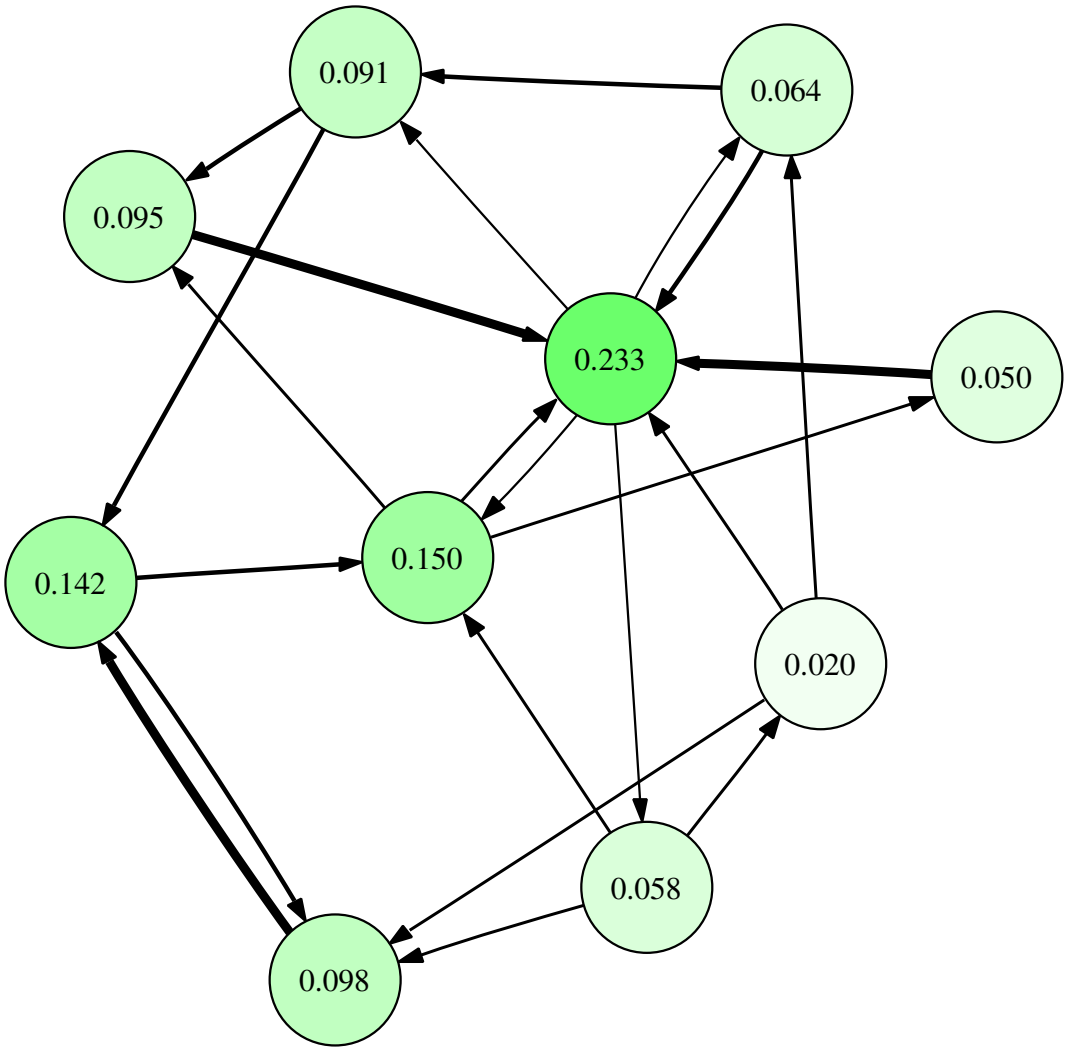
PageRank example



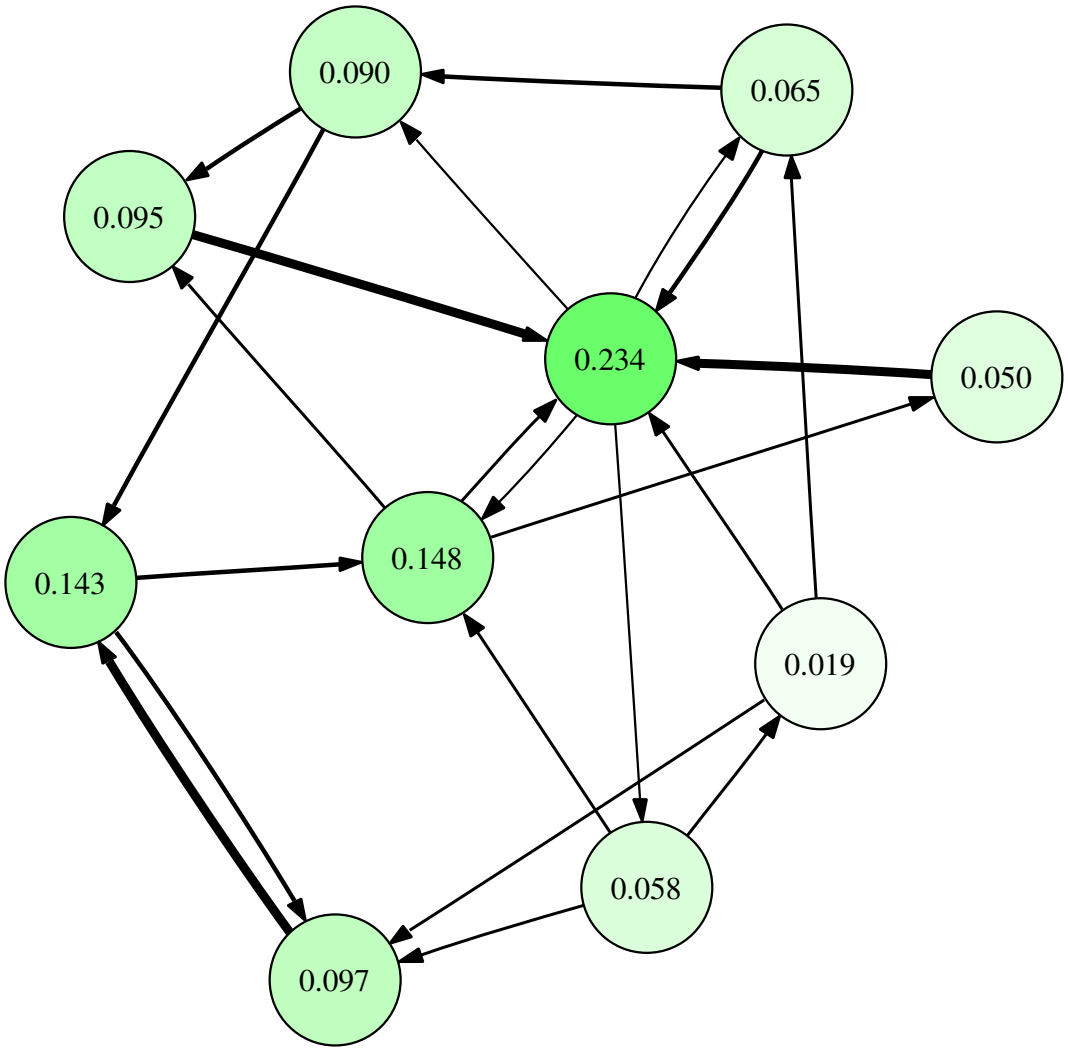
PageRank example



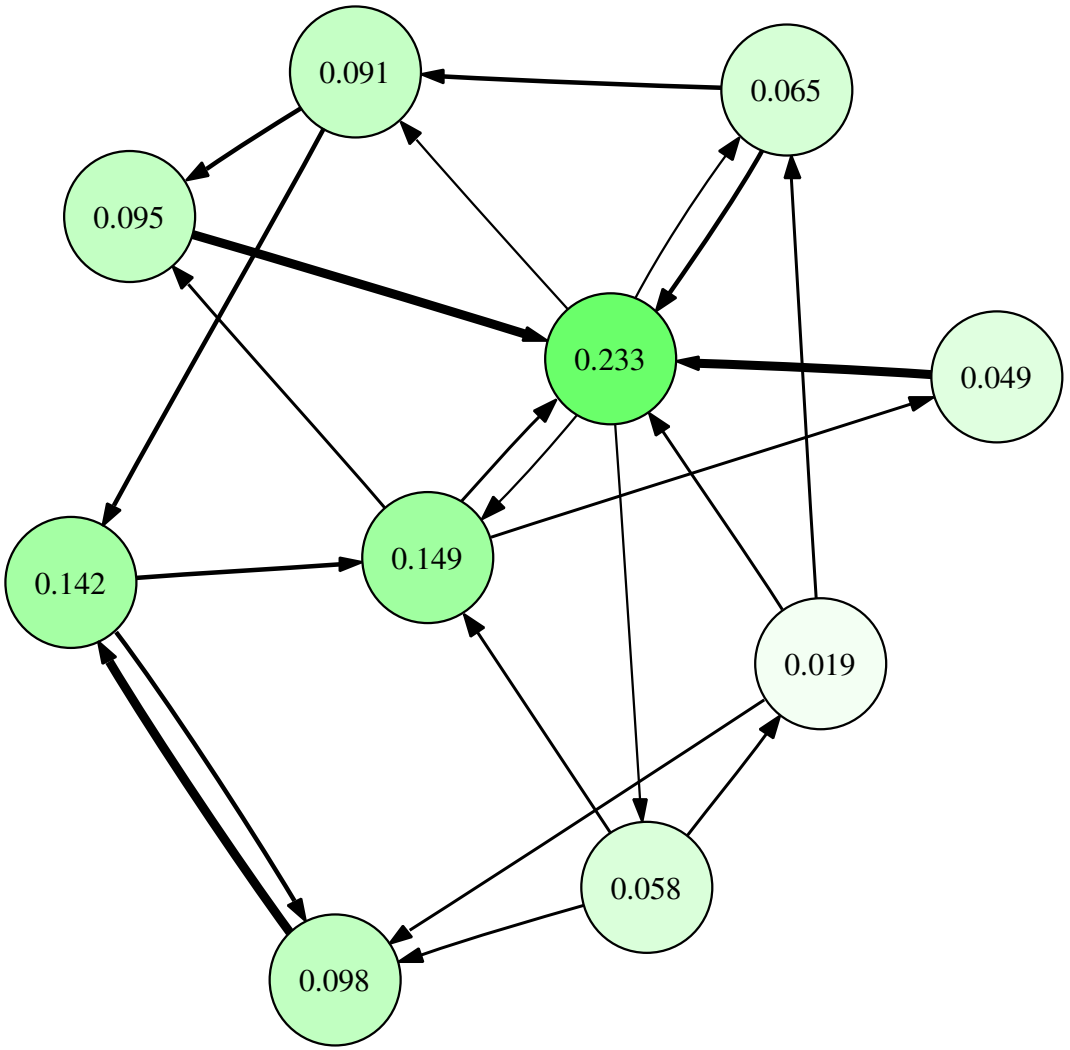
PageRank example



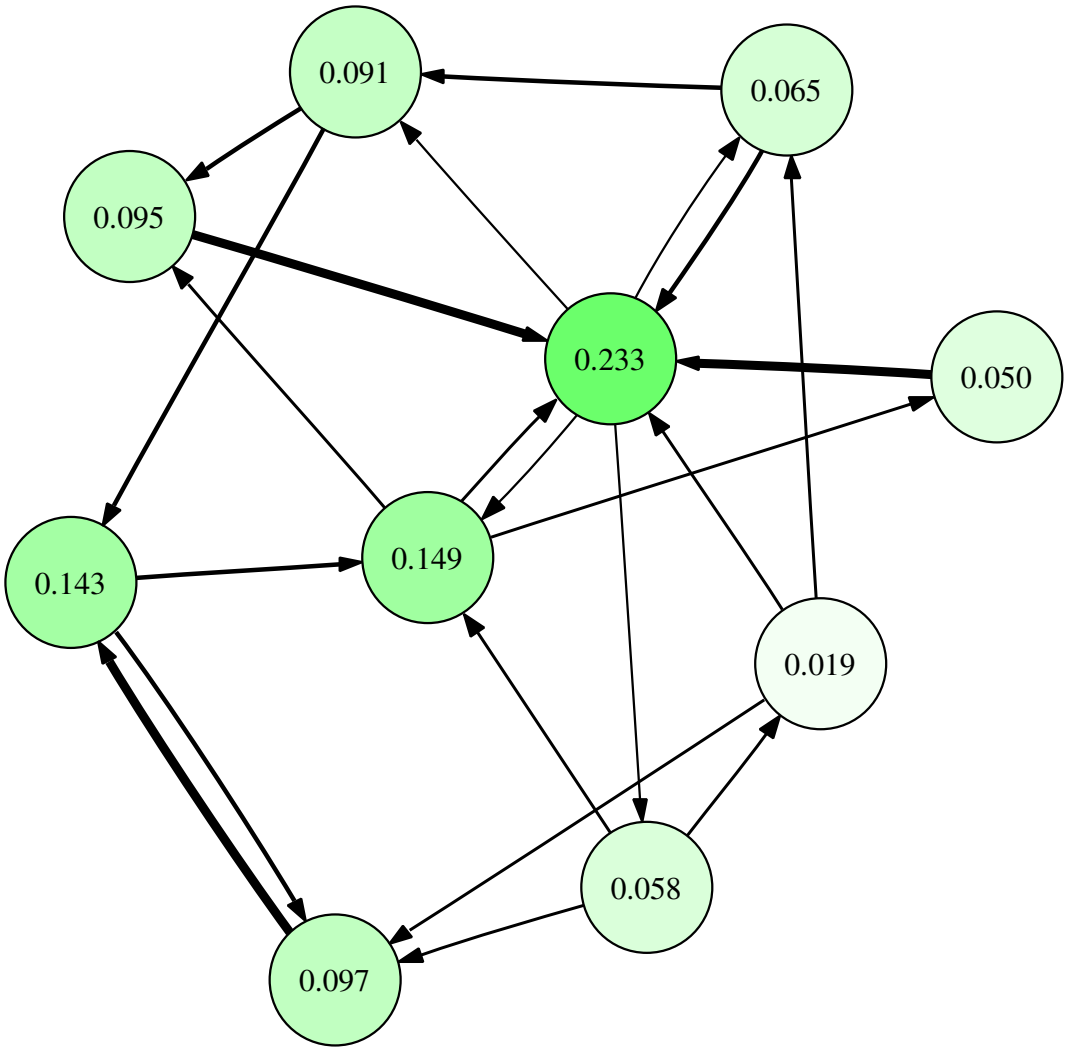
PageRank example



PageRank example



PageRank example



PageRank example

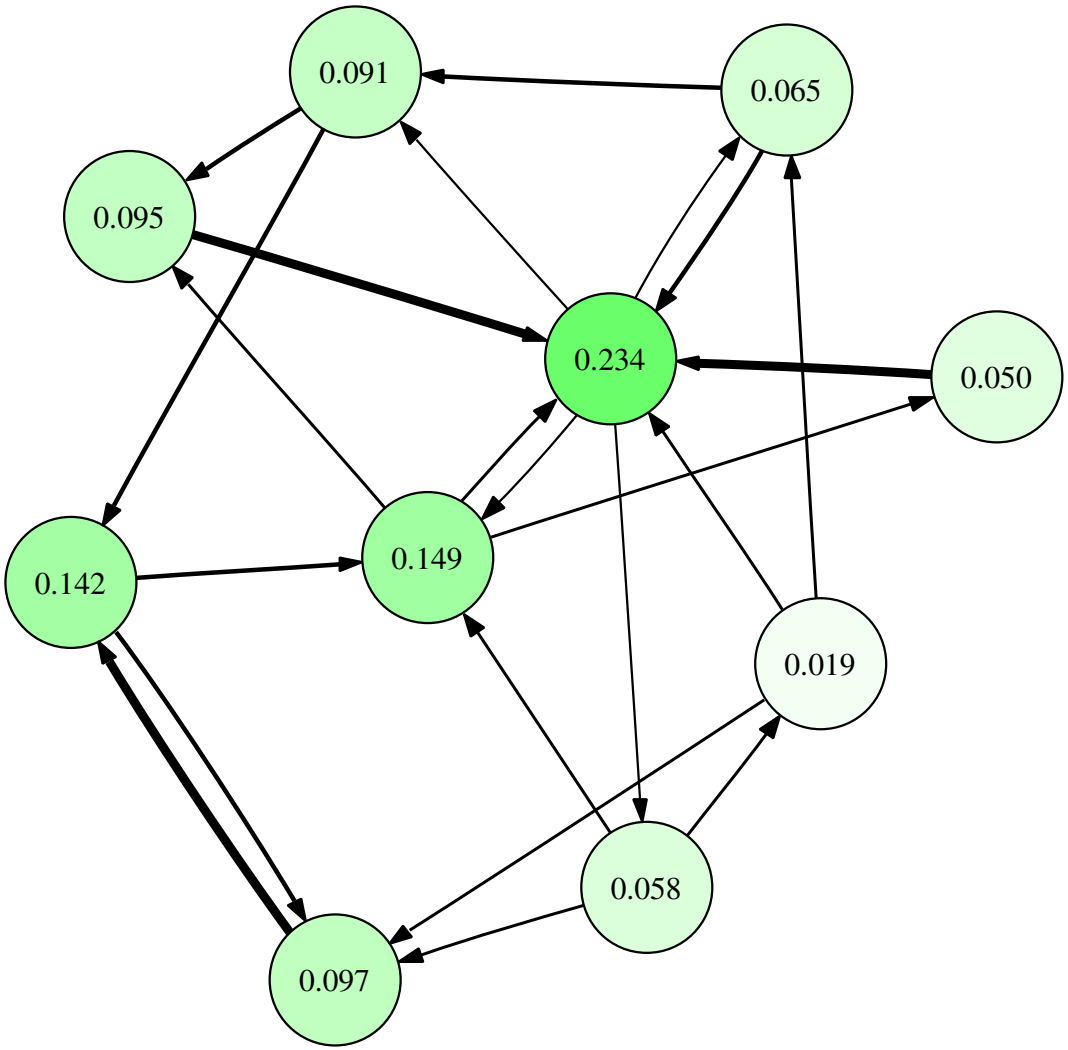


Table of contents

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[8]):
  - 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...).

[8] 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.

$\Rightarrow$ 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.
  $\Rightarrow$ 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

Domain search and enterprise search

- Indexes relevant to a specific domain:
  - Focused crawling.
  - Information extraction from important sources.
- Enterprise search engines.

  $\Rightarrow$ 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.[9]
- 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`.)

[9] 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.

$\Rightarrow$ 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).

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).

$\Rightarrow$ Risk of a filter bubble:

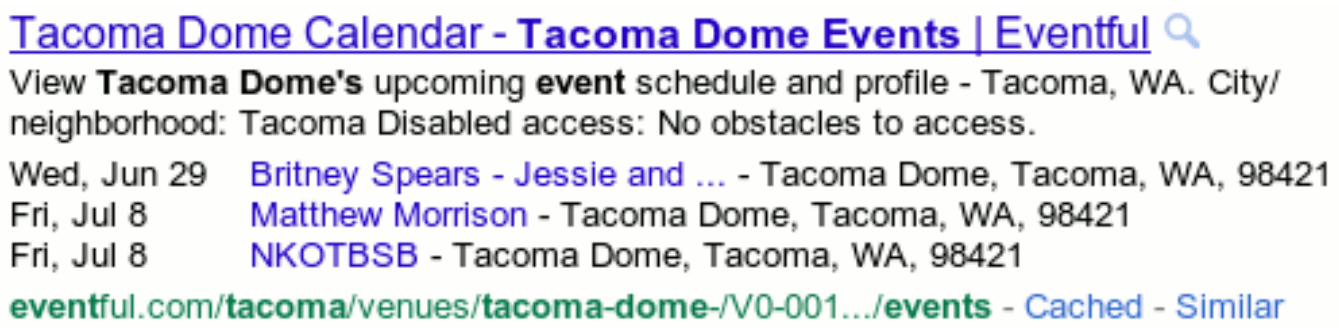
- You see content that only confirms your preferences.
- No alternative viewpoints.
- No objective notion of relevance.

$\Rightarrow$ 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>
  <img itemprop="photo" src="rms.jpg" />
  <!-- ... -->
</div>
```



Zero-click information

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

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.