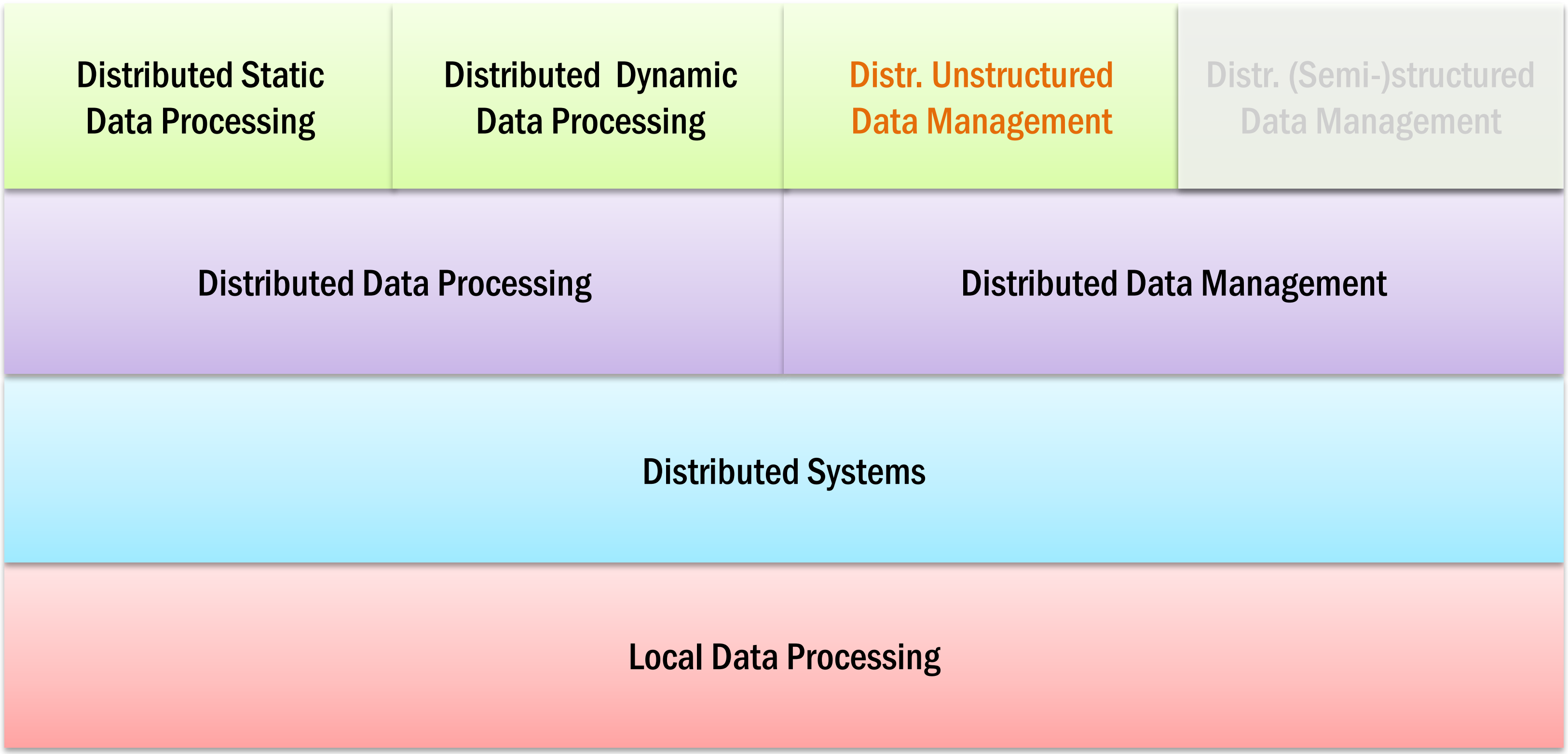
CC5212-1

Procesamiento Masivo de Datos
Otoño 2023

Lecture 8

Information Retrieval: Ranking

Aidan Hogan
aidhog@gmail.com


Distributed Static Data Processing
Distributed Dynamic Data Processing
Distr. Unstructured Data Management
Distr. (Semi-)structured Data Management
Distributed Data Processing
Distributed Data Management
Distributed Systems
Local Data Processing

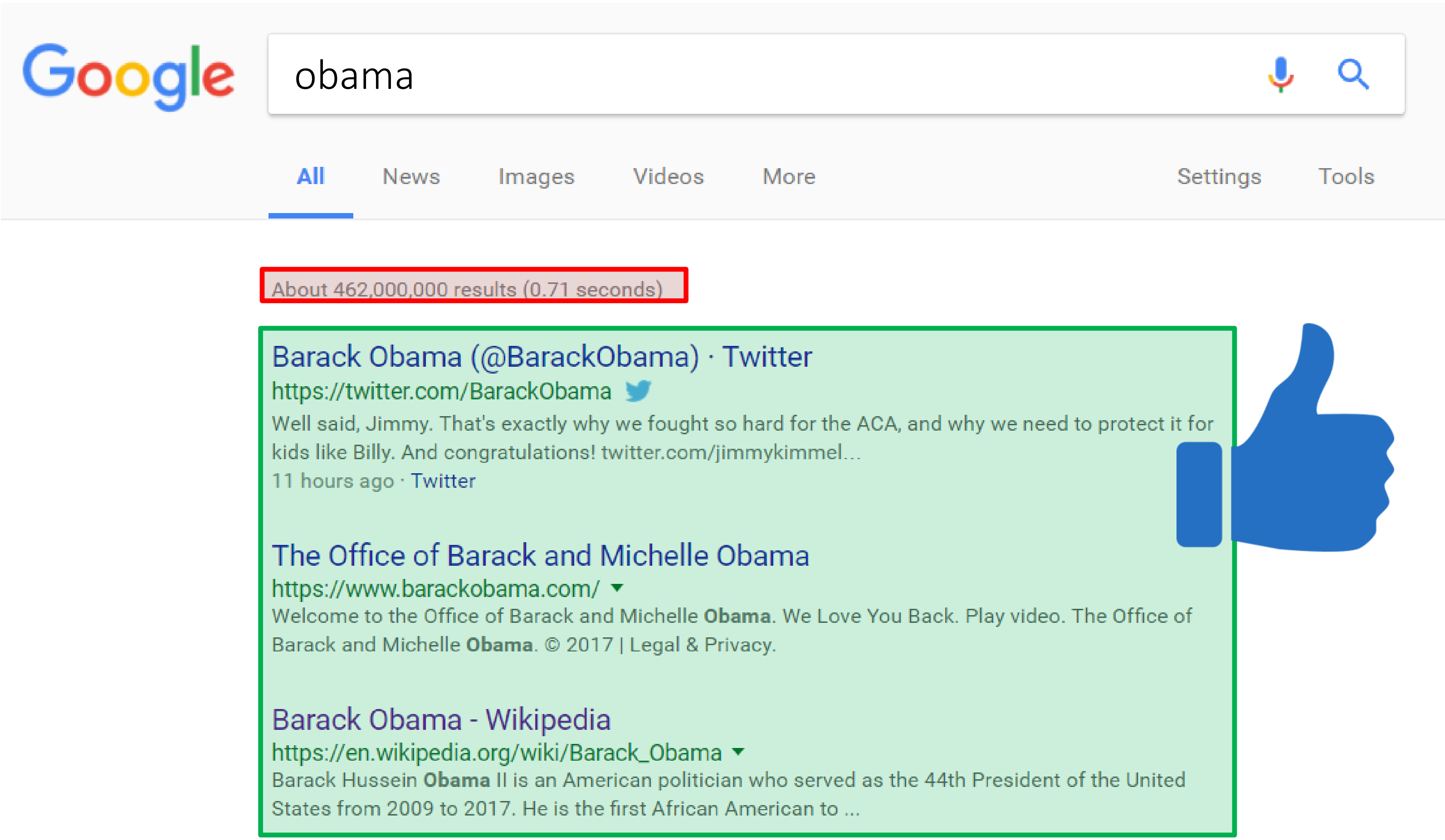
Apache Lucene

Tasks | Console

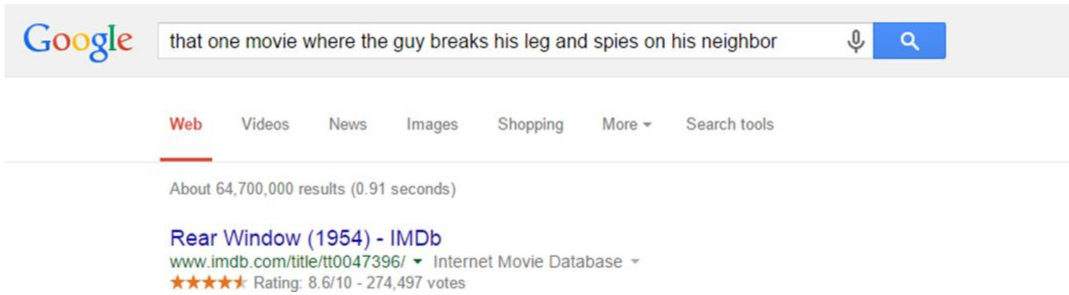
```
SearchWikiIndex [Java Application] C:\Program Files\Java\jre1.8.0_65\bin\javaw.exe (03-05-2017 12:45:22 a. m.)
Opening directory at  lucene
Enter a keyword search phrase:
obama
Running query: obama
Parsed query: TITLE:obam^5.0 ABSTRACT:obam
Matching documents: 255
Showing top 10 results
1       http://es.wikipedia.org/wiki/Obama_Republican   Obama Republican
2       http://es.wikipedia.org/wiki/Obama_(Fukui)      Obama (Fukui)
3       http://es.wikipedia.org/wiki/Republicanos_por_Obama     Republicanos por Obama
4       http://es.wikipedia.org/wiki/Engonga_Obame      Engonga Obame
5       http://es.wikipedia.org/wiki/Barack_Obama       Barack Obama
6       http://es.wikipedia.org/wiki/Michelle_Obama     Michelle Obama
7       http://es.wikipedia.org/wiki/Cartel_%22Hope%22_de_Obama Cartel "Hope" de Obama
8       http://es.wikipedia.org/wiki/Transición_presidencial_de_Barack_Obama    Transición presidencial de Barack Obama
9       http://es.wikipedia.org/wiki/Por_qué_Obama_ganará_en_2008_y_en_2012     Por qué Obama ganará en 2008 y en 2012
10      http://es.wikipedia.org/wiki/Ricardo_Mangue_Obama_Nfubea        Ricardo Mangue Obama Nfubea
```

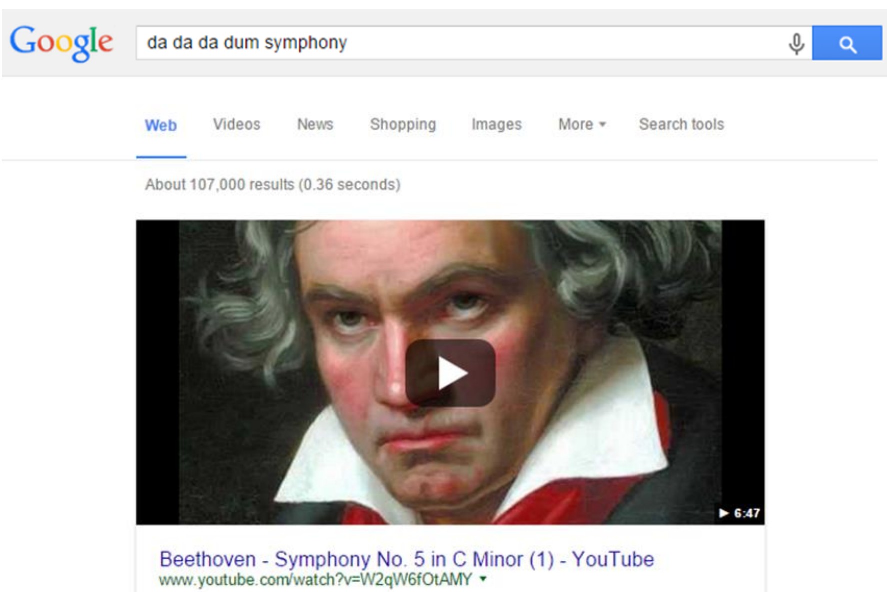
Information Retrieval: Ranking

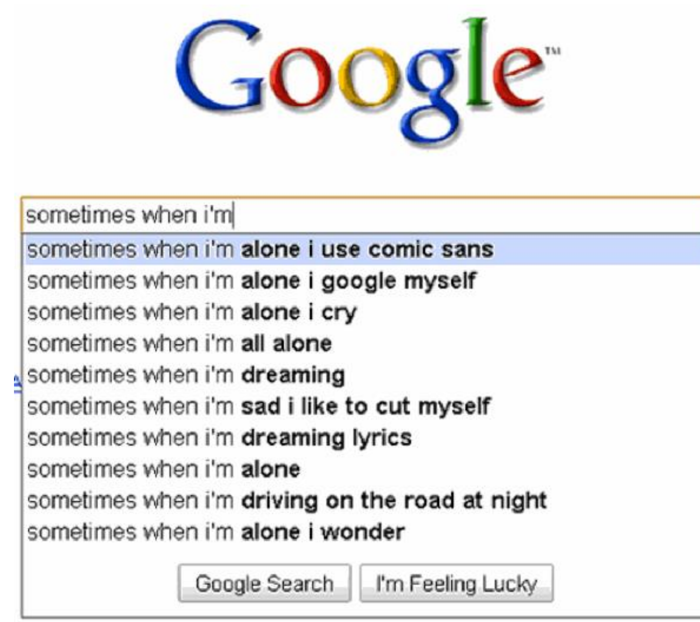
How Does Google Get Such Good Results?



How does Google Get Such Good Results?





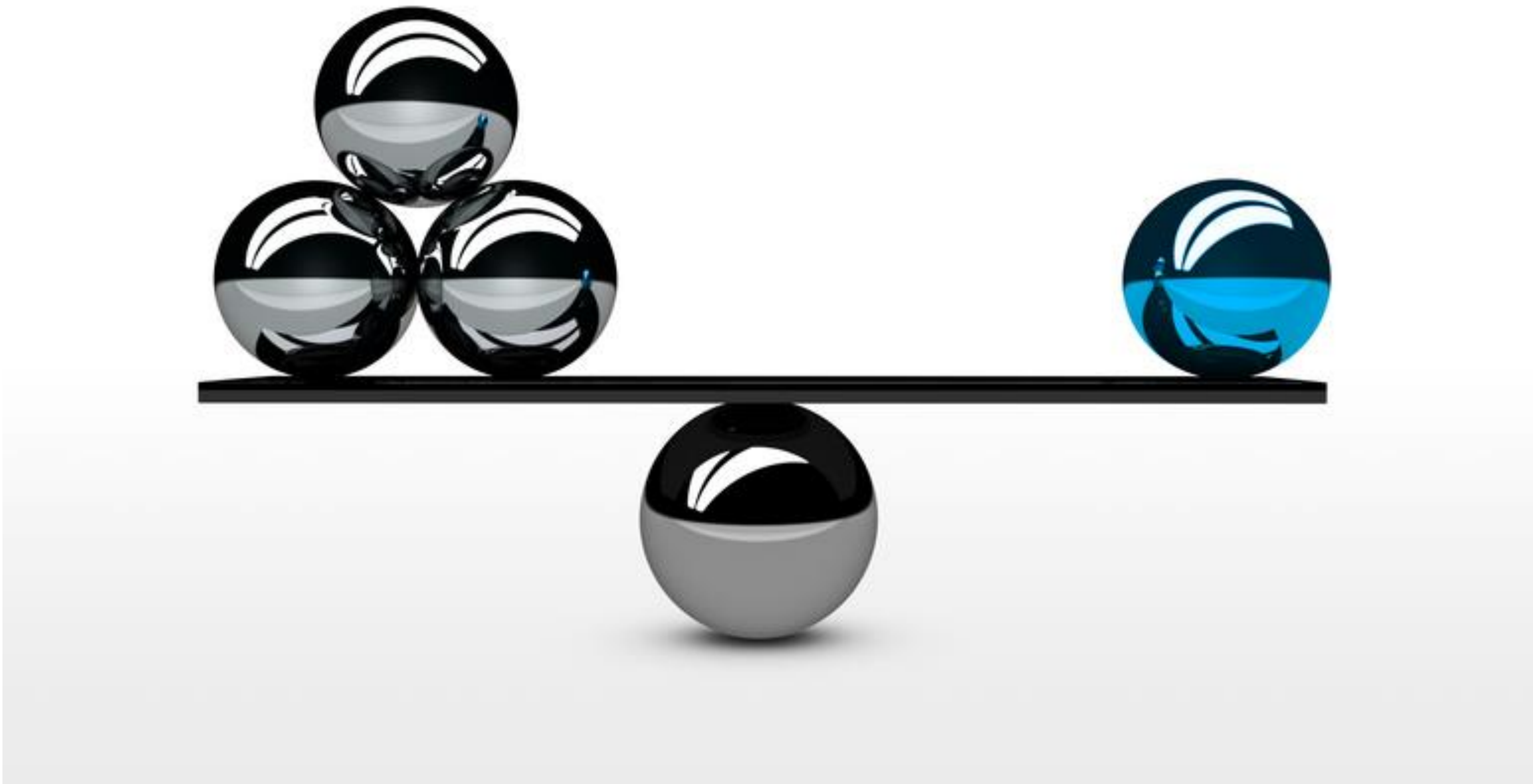


Two aspects of ranking:
Relevance vs. Importance

Two Sides to Ranking: Relevance

Two Sides to Ranking: Importance

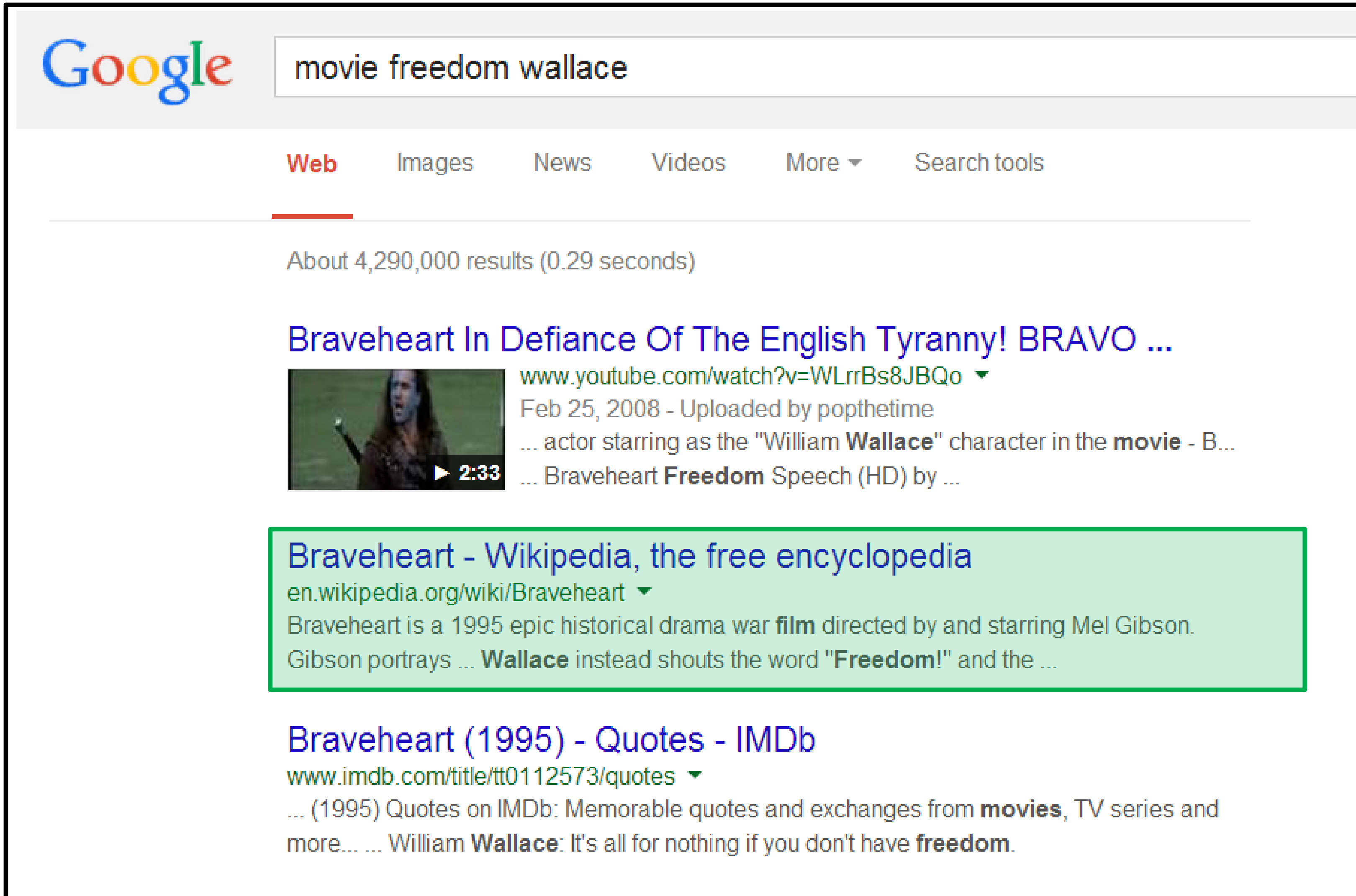
Relevance vs. Importance: A Balancing Act



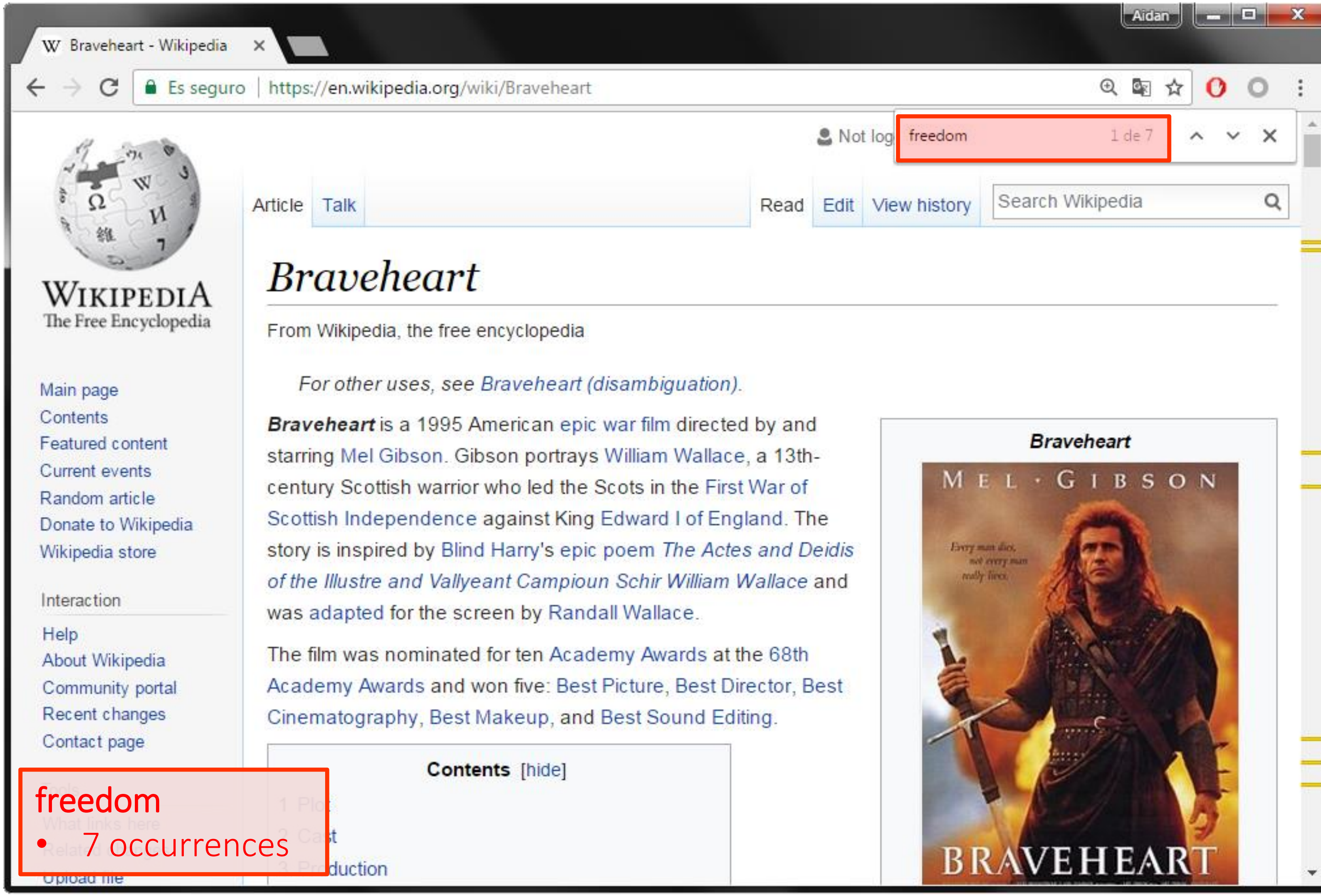
Ranking:
Relevance

Example Query

Which of these three keyword terms is most "important"?



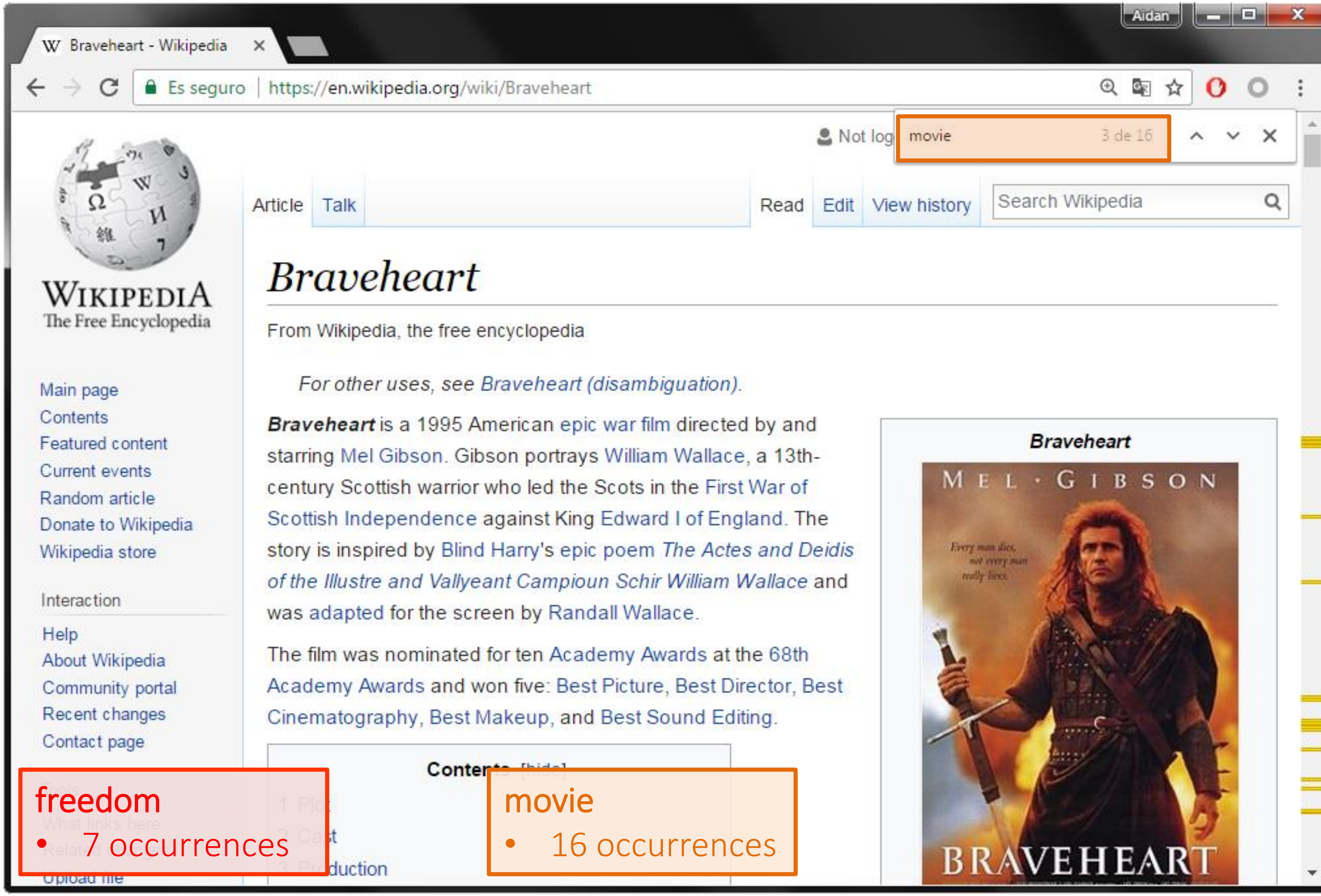
Matches in a Document



freedom

- 7 occurrences

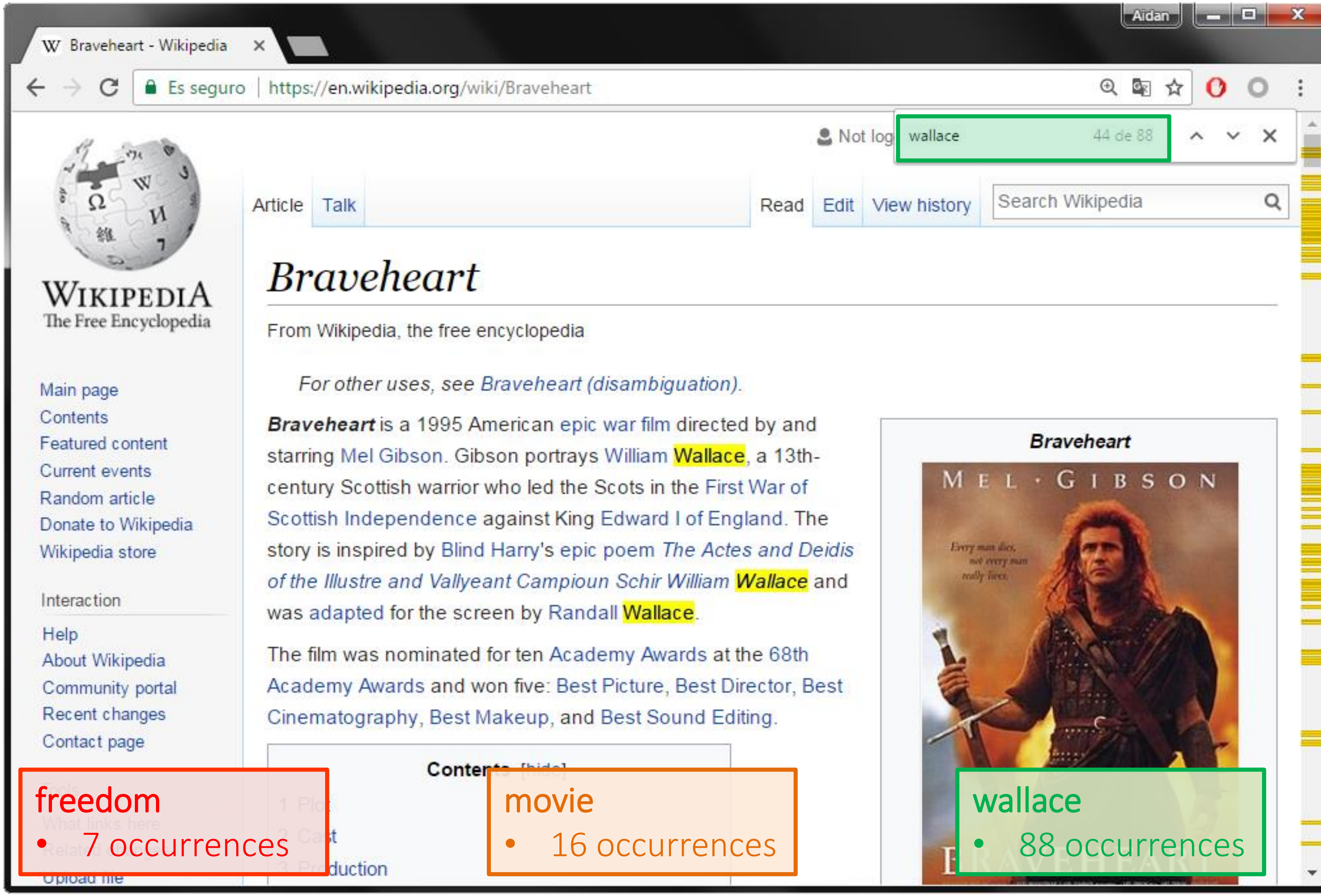
Matches in a Document



freedom
- 7 occurrences

movie
- 16 occurrences

Matches in a Document



freedom

- 7 occurrences

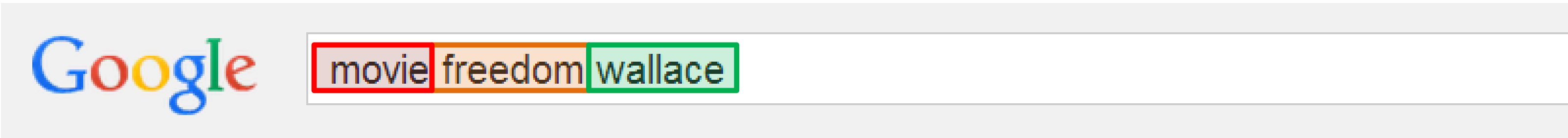
movie

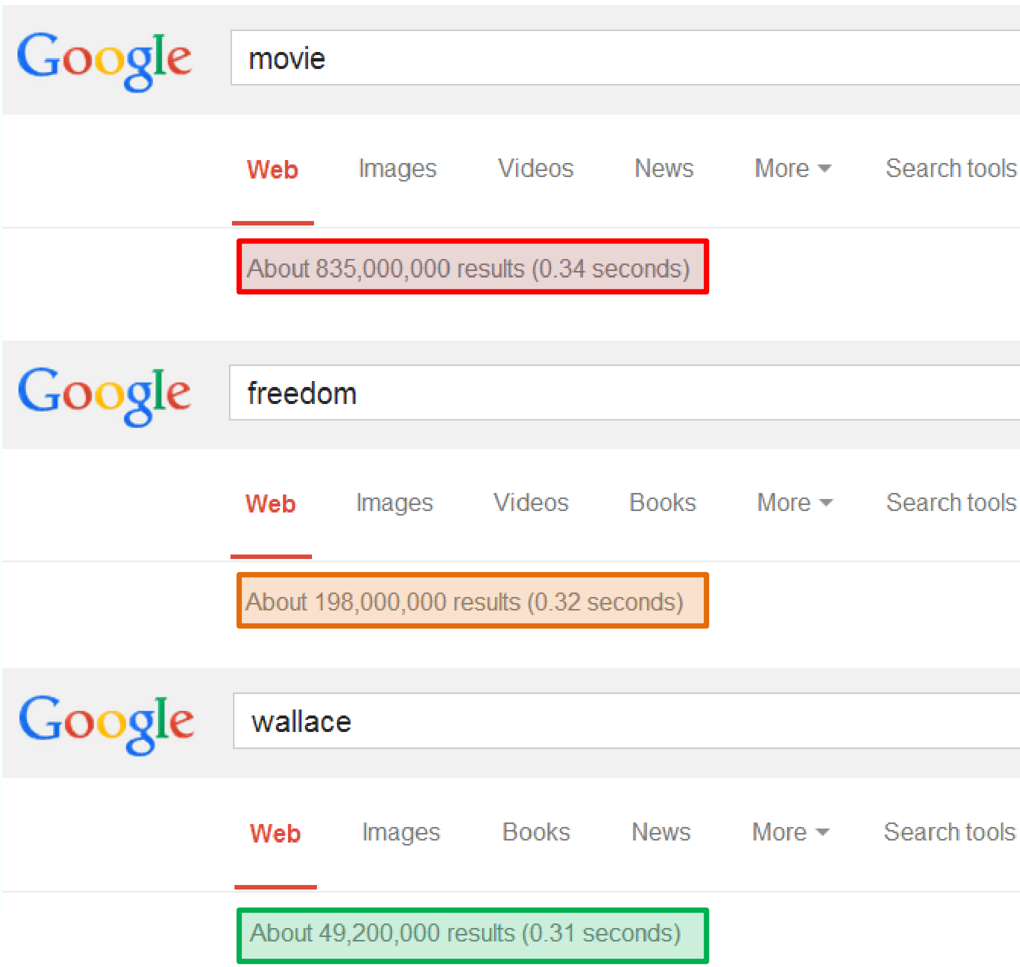
- 16 occurrences

wallace

- 88 occurrences

Usefulness of Words





movie

- occurs very frequently

freedom

- occurs frequently

wallace

- occurs occassionally

Estimating Relevance

- Rare words more important than common words
  - wallace (49M) more important than freedom (198M) more important than movie (835M)
- Words occurring more frequently in a document indicate higher relevance
  - wallace (88) more matches than movie (16) more matches than freedom (7)

Relevance Measure: TF–IDF

- TF: Term Frequency
  - Measures occurrences of a term in a document
  - $\mathrm{tf}(t,d)$ ... various options
    - Raw count of occurrences

$$\boxed{\mathrm{tf}(t,d) = \mathrm{count}(t,d)}$$

    - Logarithmically scaled

$$\boxed{\mathrm{tf}(t,d) = \log(\mathrm{count}(t,d) + 1)}$$

    - Normalised by document length

$$\boxed{\mathrm{tf}(t,d) = \frac{\mathrm{count}(t,d)}{\sum_{t' \in d} \mathrm{count}(t',d)}}$$

$$\boxed{\mathrm{tf}(t,d) = \frac{\mathrm{count}(t,d)}{\max_{t' \in d} \mathrm{count}(t',d)}}$$

    - A combination / something else ☺

Relevance Measure: TF–IDF

- IDF: Inverse Document Frequency
  - How common a term is across **all** documents
  - $\mathrm{idf}(t, D)$ ...
    - Logarithmically scaled document occurrences

$$\boxed{\mathrm{idf}(t, D) = \log\left(\frac{|D|+1}{|\{d \in D : t \in d\}|+1}\right)}$$

    - Note: The more rare, the larger the value

Relevance Measure: TF–IDF

- TF–IDF: Combine Term Frequency and Inverse Document Frequency:

$$\text{tf-idf}(t, d) = \text{tf}(t, d) \times \text{idf}(t, D)$$

- Score for a query
  - Let query $q = (t_1, \ldots, t_n)$
  - Score for a query: $\text{score}(q, d) = \sum_{t \in q} \text{tf-idf}(t, d)$

(There are other possibilities)

Relevance Measure: TF–IDF

Term Frequency

$$\boxed{\mathrm{tf}(t,d) = \mathrm{count}(t,d)}$$

Inverse Document Frequency

$$\boxed{\mathrm{idf}(t,D) = \log_2\left(\frac{|D|+1}{|\{d \in D : t \in d\}|+1}\right)}$$

$$\mathrm{tf\text{-}idf}(t,d) = \boxed{\mathrm{tf}(t,d)} \times \boxed{\mathrm{idf}(t,D)}$$

t	$\mathrm{tf}(t,d)$
movie	16
freedom	7
wallace	43

Relevance Measure: TF–IDF

Term Frequency

$$\mathrm{tf}(t,d) = \mathrm{count}(t,d)$$

Inverse Document Frequency

$$\mathrm{idf}(t,D) = \log_2\left(\frac{|D|+1}{|\{d \in D : t \in d\}|+1}\right)$$

$$\text{tf-idf}(t,d) = \boxed{\mathrm{tf}(t,d)} \times \boxed{\mathrm{idf}(t,D)}$$

t	$\mathrm{tf}(t,d)$	$\lvert\{d \in D : t \in d\}\rvert$
movie	16	835,000,000
freedom	7	198,000,000
wallace	43	49,200,000

Relevance Measure: TF–IDF

Term Frequency

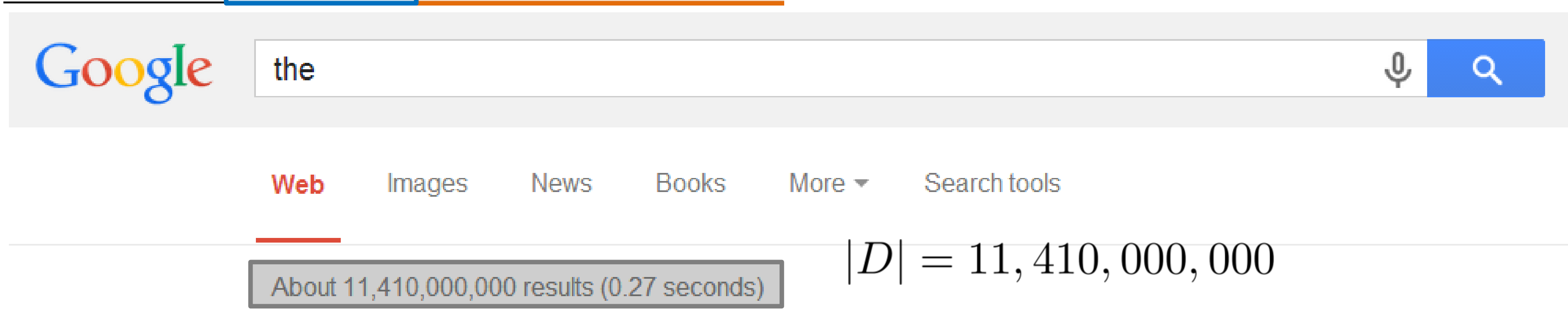
$$\boxed{\mathrm{tf}(t,d) = \mathrm{count}(t,d)}$$

Inverse Document Frequency

$$\boxed{\mathrm{idf}(t,D) = \log_2\left(\frac{|D|+1}{|\{d \in D : t \in d\}|+1}\right)}$$

$$\text{tf-idf}(t,d) = \boxed{\mathrm{tf}(t,d)} \times \boxed{\mathrm{idf}(t,D)}$$

t	$\mathrm{tf}(t,d)$	$\lvert\{d \in D : t \in d\}\rvert$	$\frac{\lvert D\rvert+1}{\lvert\{d \in D : t \in d\}\rvert+1}$
movie	16	835,000,000	
freedom	7	198,000,000	
wallace	43	49,200,000	



$$|D| = 11,410,000,000$$

Relevance Measure: TF–IDF

Term Frequency

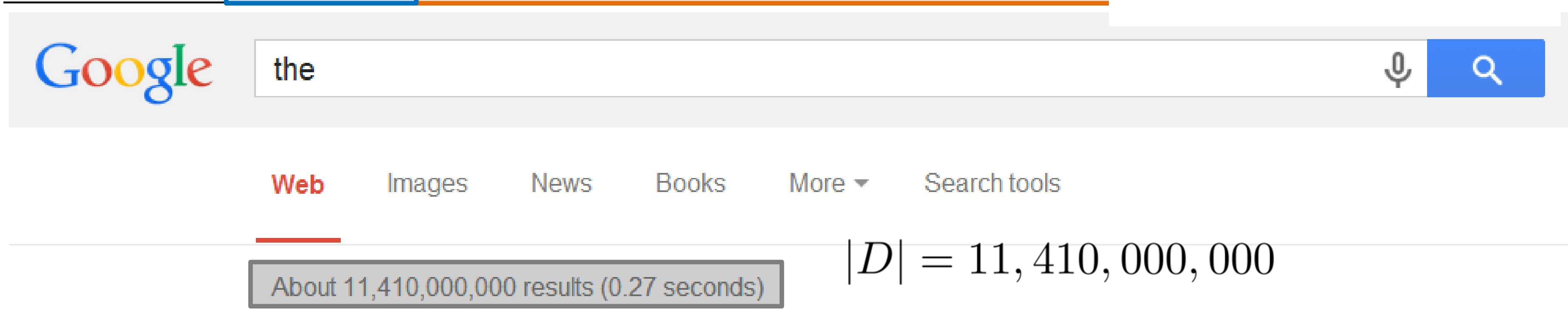
$$\boxed{\mathrm{tf}(t,d) = \mathrm{count}(t,d)}$$

Inverse Document Frequency

$$\boxed{\mathrm{idf}(t,D) = \log_2\left(\frac{|D|+1}{|\{d \in D : t \in d\}|+1}\right)}$$

$$\text{tf-idf}(t,d) = \boxed{\mathrm{tf}(t,d)} \times \boxed{\mathrm{idf}(t,D)}$$

t	$\mathrm{tf}(t,d)$	$\lvert\{d \in D \,:\, t \in d\}\rvert$	$\frac{\lvert D\rvert+1}{\lvert\{d\in D : t\in d\}\rvert+1}$
movie	16	835,000,000	13.66
freedom	7	198,000,000	57.63
wallace	43	49,200,000	231.91



$$|D| = 11{,}410{,}000{,}000$$

Relevance Measure: TF–IDF

Term Frequency

$$\boxed{\mathrm{tf}(t,d) = \mathrm{count}(t,d)}$$

Inverse Document Frequency

$$\boxed{\mathrm{idf}(t,D) = \log_2\left(\frac{|D|+1}{|\{d \in D : t \in d\}|+1}\right)}$$

$$\text{tf-idf}(t,d) = \boxed{\mathrm{tf}(t,d)} \times \boxed{\mathrm{idf}(t,D)}$$

t	$\mathrm{tf}(t,d)$	$\lvert\{d \in D : t \in d\}\rvert$	$\frac{\lvert D\rvert+1}{\lvert\{d \in D : t \in d\}\rvert+1}$	$\mathrm{idf}(t,d)$
movie	16	835,000,000	13.66	3.77
freedom	7	198,000,000	57.63	5.85
wallace	43	49,200,000	231.91	7.86

Relevance Measure: TF–IDF

Term Frequency

$$\boxed{\mathrm{tf}(t,d) = \mathrm{count}(t,d)}$$

Inverse Document Frequency

$$\boxed{\mathrm{idf}(t,D) = \log_2\left(\frac{|D|+1}{|\{d\in D : t\in d\}|+1}\right)}$$

$$\text{tf-idf}(t,d) = \boxed{\mathrm{tf}(t,d)} \times \boxed{\mathrm{idf}(t,D)}$$

t	$\mathrm{tf}(t,d)$	$\lvert\{d \in D : t \in d\}\rvert$	$\frac{\lvert D\rvert+1}{\lvert\{d\in D : t\in d\}\rvert+1}$	$\mathrm{idf}(t,d)$	$\text{tf-idf}(t,d)$
movie	16	835,000,000	13.66	3.77	60.36
freedom	7	198,000,000	57.63	5.85	40.94
wallace	43	49,200,000	231.91	7.86	337.87

Relevance Measure: TF–IDF

Term Frequency

$$\boxed{\mathrm{tf}(t,d) = \mathrm{count}(t,d)}$$

Inverse Document Frequency

$$\boxed{\mathrm{idf}(t,D) = \log_2\left(\frac{|D|+1}{|\{d \in D : t \in d\}|+1}\right)}$$

$$\mathrm{tf\text{-}idf}(t,d) = \boxed{\mathrm{tf}(t,d)} \times \boxed{\mathrm{idf}(t,D)}$$

t	$\mathrm{tf}(t,d)$	$\lvert\{d \in D : t \in d\}\rvert$	$\frac{\lvert D\rvert+1}{\lvert\{d\in D : t\in d\}\rvert+1}$	$\mathrm{idf}(t,d)$	$\mathrm{tf\text{-}idf}(t,d)$
movie	16	835,000,000	13.66	3.77	60.36
freedom	7	198,000,000	57.63	5.85	40.94
wallace	43	49,200,000	231.91	7.86	337.87

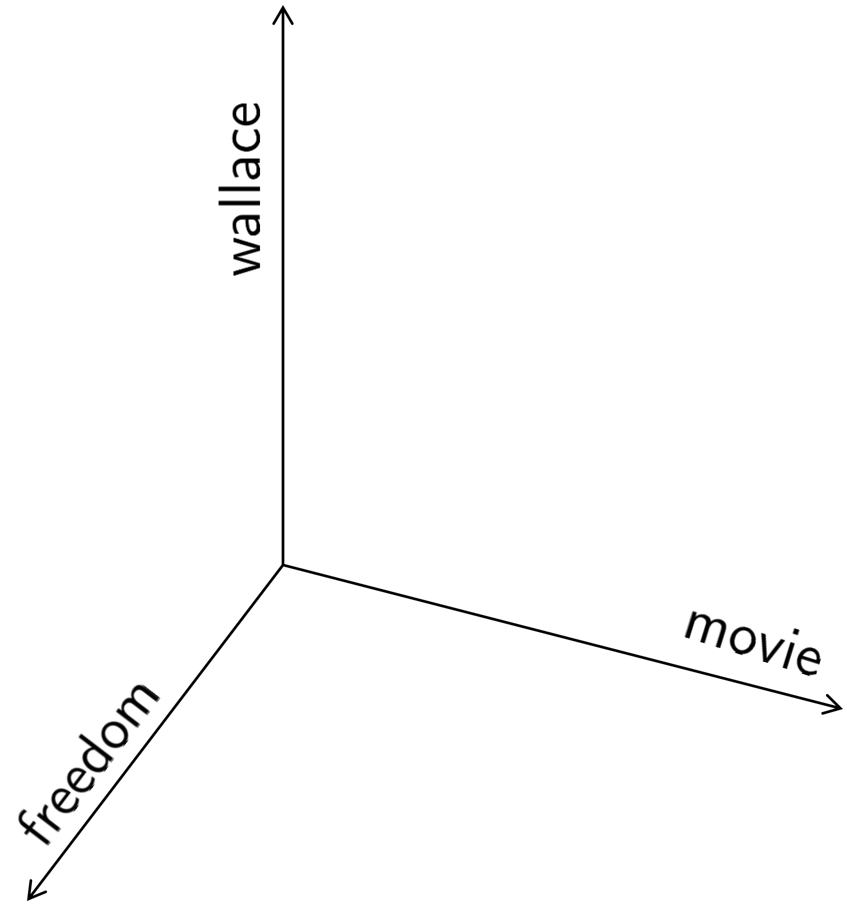
$$\mathrm{score}(q,d) = \textstyle\sum_{t \in q} \mathrm{tf\text{-}idf}(t,d)$$

$$\mathrm{score}\big((\text{movie}, \text{freedom}, \text{wallace}), \text{http://en.wikipedia.org/Braveheart}\big) \approx 439.17$$

Vector Space Model (a mention)

t	$\mathrm{tf}(t,d)$
movie	16
freedom	7
wallace	43

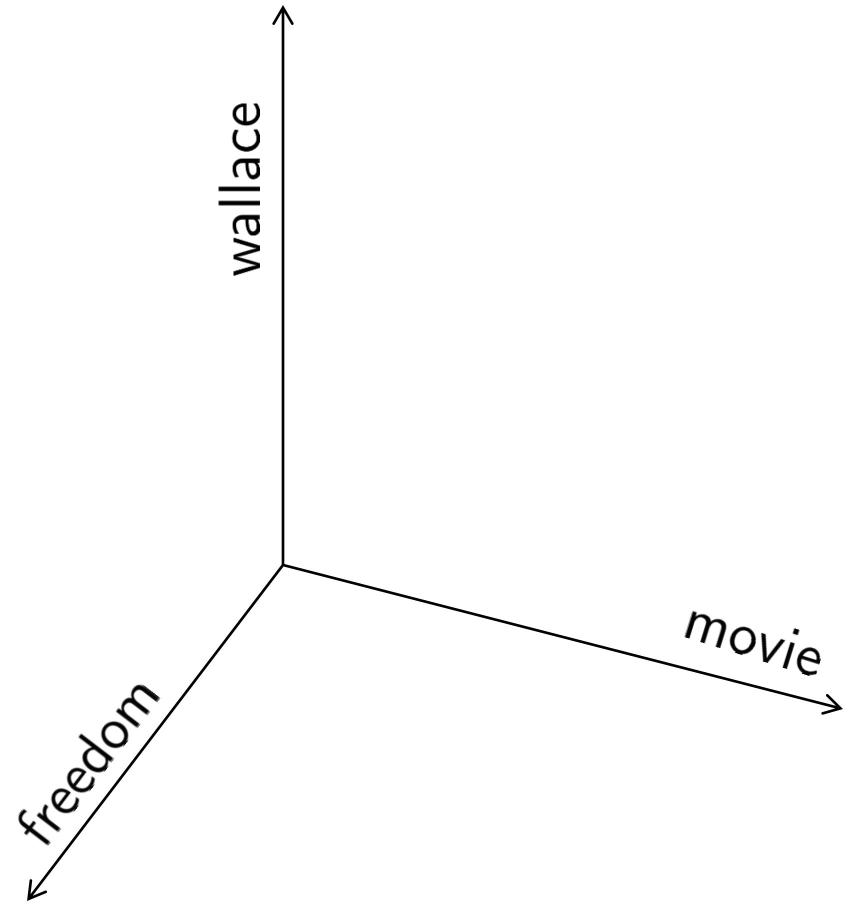
$$l = \sqrt{\sum_{t \in q} \mathrm{tf}(t,d)^2}$$



Vector Space Model (a mention)

t	$\mathrm{tf}(t,d)$	$\mathrm{tf}(t,d)^2$
movie	16	256
freedom	7	49
wallace	43	1,894

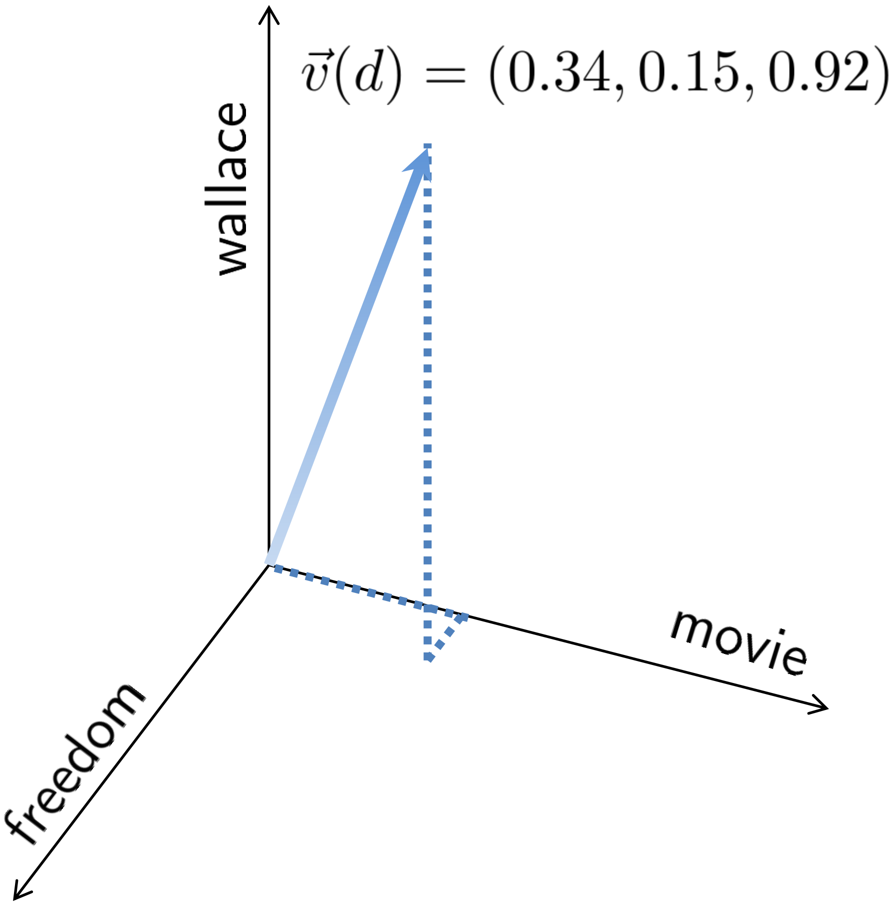
$$l = \sqrt{\sum_{t \in q} \mathrm{tf}(t,d)^2}$$



Vector Space Model (a mention)

t	$\mathrm{tf}(t,d)$	$\mathrm{tf}(t,d)^2$	$\frac{\mathrm{tf}(t,d)}{l}$
movie	16	256	0.34
freedom	7	49	0.15
wallace	43	1,894	0.92

$$l = \sqrt{\sum_{t \in q} \mathrm{tf}(t,d)^2}$$



Dividing by l normalises the length of vector to 1

Vector Space Model (a mention)

- Cosine Similarity

$$\mathrm{sim}(d, d') = \vec{v}(d) \cdot \vec{v}(d')$$

t	$\vec{v}(d)$	$\vec{v}(d')$	$\times$
movie	0.34	0.49	0.17
freedom	0.15	0.82	0.12
wallace	0.93	0.30	0.28

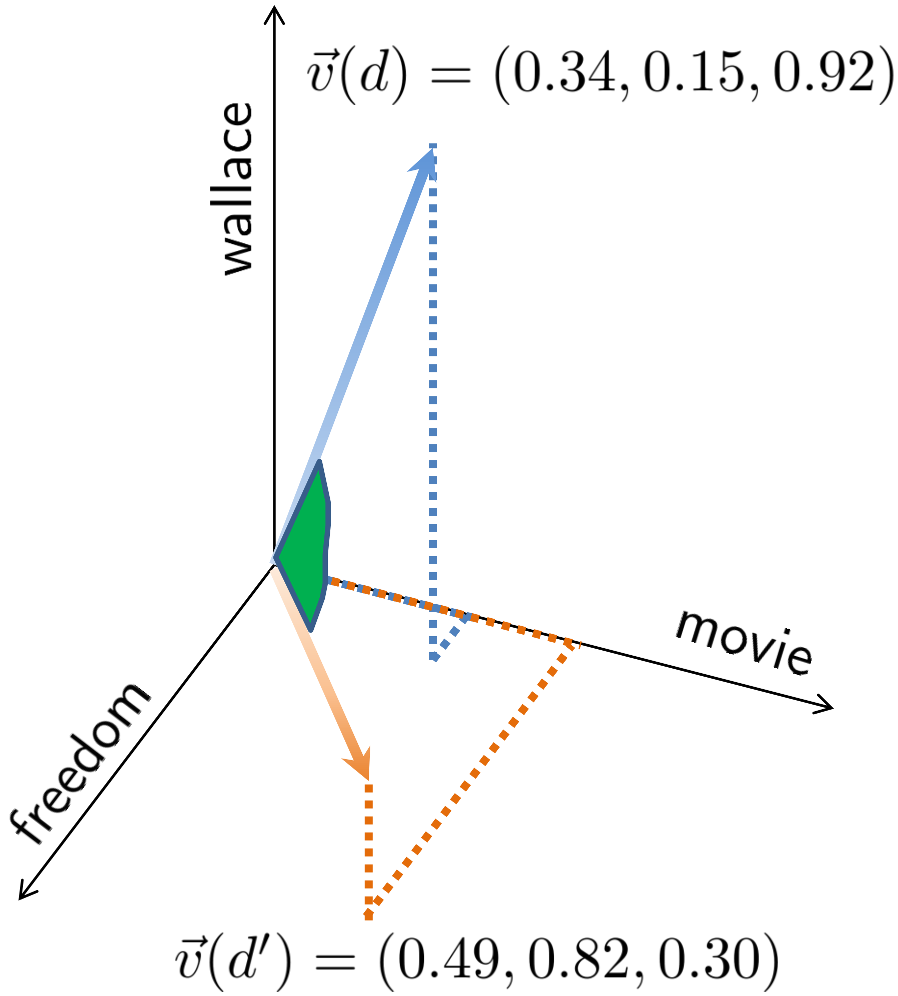
$\sum$

$$\mathrm{sim}(d, d') \approx \boxed{0.57}$$

- Note:

$$\mathbf{a} \cdot \mathbf{b} = |\mathbf{a}|\,|\mathbf{b}| \cos(\angle(\mathbf{a}, \mathbf{b}))$$

$$|\vec{v}(d)| = |\vec{v}(d')| = 1$$



Hence the similarity is the cosine of the **angle** between the vectors

Relevance Measure: TF–IDF

- TF–IDF: Combine Term Frequency and Inverse Document Frequency:

$$\text{tf-idf}(t,d) = \text{tf}(t,d) \times \text{idf}(t,D)$$

- Score for a query
  - Let query $q = (t_1, \ldots, t_n)$
  - Score for a query: $score(q,d) = \sum_{t \in q} \text{tf-idf}(t,d)$

  (There are other possibilities)

... we could also use cosine similarity between query and document using TF–IDF weights

Two Sides to Ranking: Relevance

Field-Based Boosting

- Not all text is equal: titles, headers, etc.

Anchor Text

- See how the Web views/tags a page

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
<head>
  <title>What I watched last night ...</title>
</head>
<body>
<p>Last night I was pretty bored so I made some popcorn and watched
<a href="http://en.wikipedia.org/Braveheart">a movie about William Wallace called Braveheart</a>.
Set in Scotland it has lots of blood and gore.
</p>
</body>
</html>
```

Anchor Text

- See how the Web views/tags a page

Lucene uses relevance scoring

```
Tasks   Console
SearchWikiIndex [Java Application] C:\Program Files\Java\jre1.8.0_65\bin\javaw.exe (03-05-2017 12:45:22 a. m.)
Opening directory at  lucene
Enter a keyword search phrase:
obama
Running query: obama
Parsed query: TITLE:obam^5.0 ABSTRACT:obam
Matching documents: 255
Showing top 10 results
1       http://es.wikipedia.org/wiki/Obama_Republican   Obama Republican
2       http://es.wikipedia.org/wiki/Obama_(Fukui)      Obama (Fukui)
3       http://es.wikipedia.org/wiki/Republicanos_por_Obama     Republicanos por Obama
4       http://es.wikipedia.org/wiki/Engonga_Obame      Engonga Obame
5       http://es.wikipedia.org/wiki/Barack_Obama       Barack Obama
6       http://es.wikipedia.org/wiki/Michelle_Obama     Michelle Obama
7       http://es.wikipedia.org/wiki/Cartel_%22Hope%22_de_Obama Cartel "Hope" de Obama
8       http://es.wikipedia.org/wiki/Transición_presidencial_de_Barack_Obama    Transición presidencial de Barack Obama
9       http://es.wikipedia.org/wiki/Por_qué_Obama_ganará_en_2008_y_en_2012     Por qué Obama ganará en 2008 y en 2012
10      http://es.wikipedia.org/wiki/Ricardo_Mangue_Obama_Nfubea        Ricardo Mangue Obama Nfubea
```

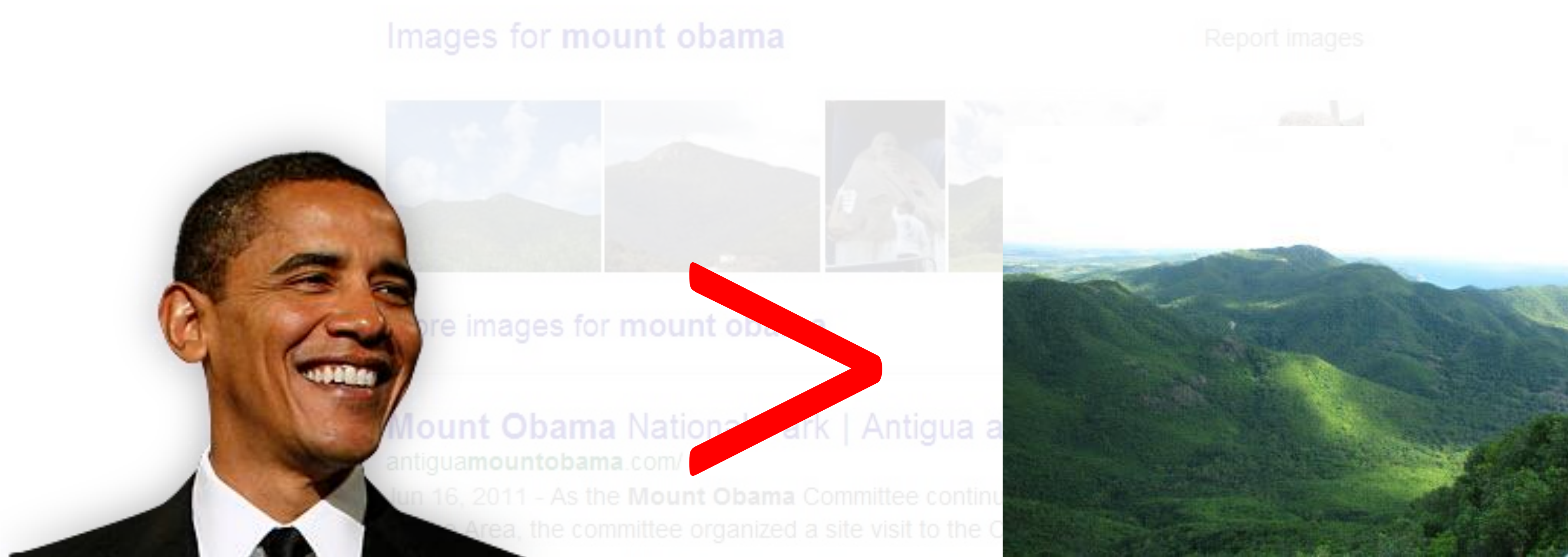
... and Elasticsearch uses Lucene

RANKING:

IMPORTANCE

Two Sides to Ranking: Importance

How could we determine that Barack Obama is more important than Mount Obama as a search result for "**obama**" on the Web?

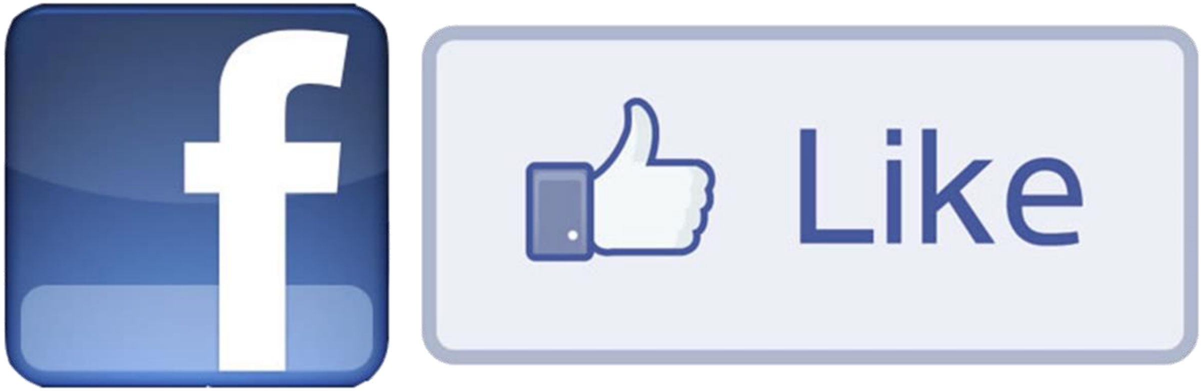


Link Analysis

Which will have more links from other pages?
The Wikipedia article for Mount Obama?
The Wikipedia article for Barack Obama?

Link Analysis

- Consider links as votes of confidence in a page
- A hyperlink is the open Web’s version of …

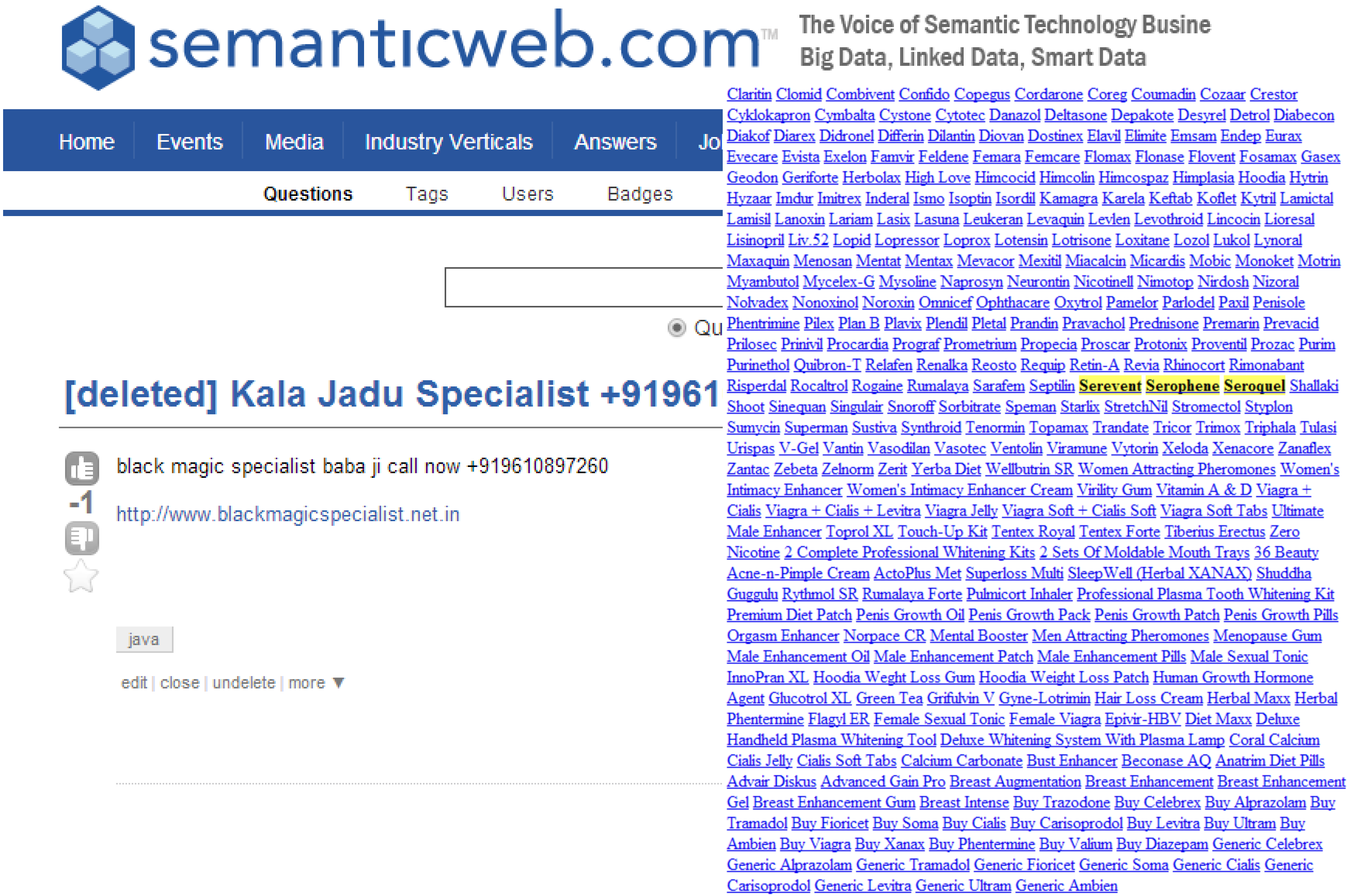


(… even if the page is linked in a negative way.)

Link Analysis

So if we just count links to a page we can determine its importance and we are done?

Link Spamming



Link Importance

So which should count for more?
A link from http://en.wikipedia.org/wiki/Barack_Obama?
Or a link from http://blackmagicspecialist.net.in?

PageRank

PageRank: Central Assumption

A page with lots of inlinks from important pages with few outlinks is more important

PageRank: Recursive Definition

A page with lots of inlinks from <u>important</u> pages with few outlinks is more <u>important</u>

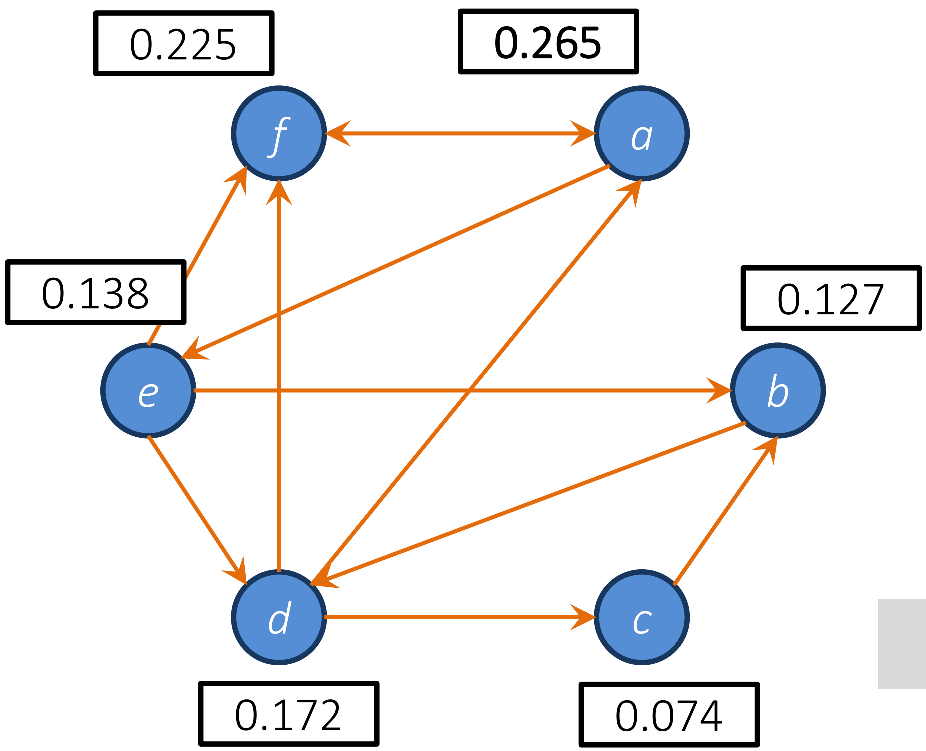
PageRank Model

- The Web: a directed graph

$$G = (V, E)$$

Vertices (*pages*)

Edges (*links*)



Which vertex is most important?

$$V = \{a, b, c, d, e, f\}$$

$$E = \{(a,e), (a,f), (b,d), (c,b), (d,a), (d,c), (d,f), (e,b), (e,d), (e,f), (f,a)\}$$

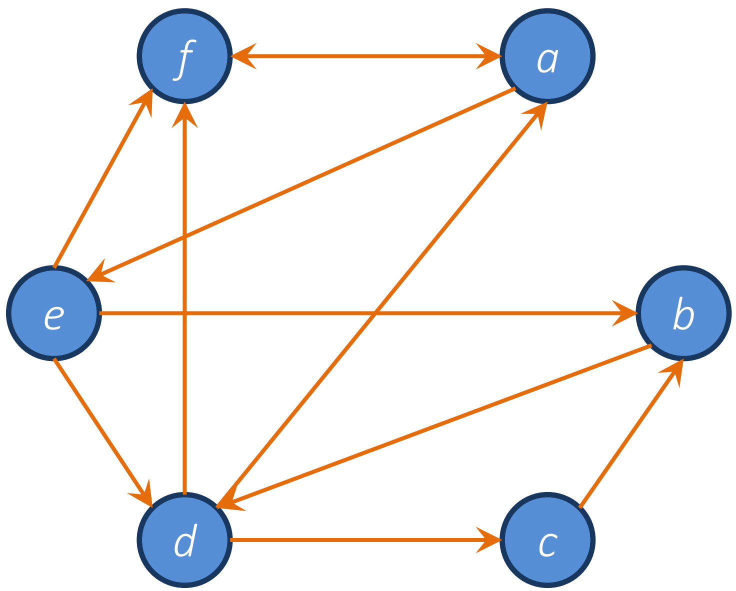
PageRank Model

- The Web: a directed graph

$$G = (\boxed{V}, \boxed{E})$$

Vertices (*pages*)

Edges (*links*)



$$\mathrm{out}(v) \coloneqq \{v' \in V \mid (v, v') \in E\}$$

$$\mathrm{in}(v) \coloneqq \{v' \in V \mid (v', v) \in E\}$$

$$\mathrm{rank}_0(v) \coloneqq \frac{1}{|V|}$$

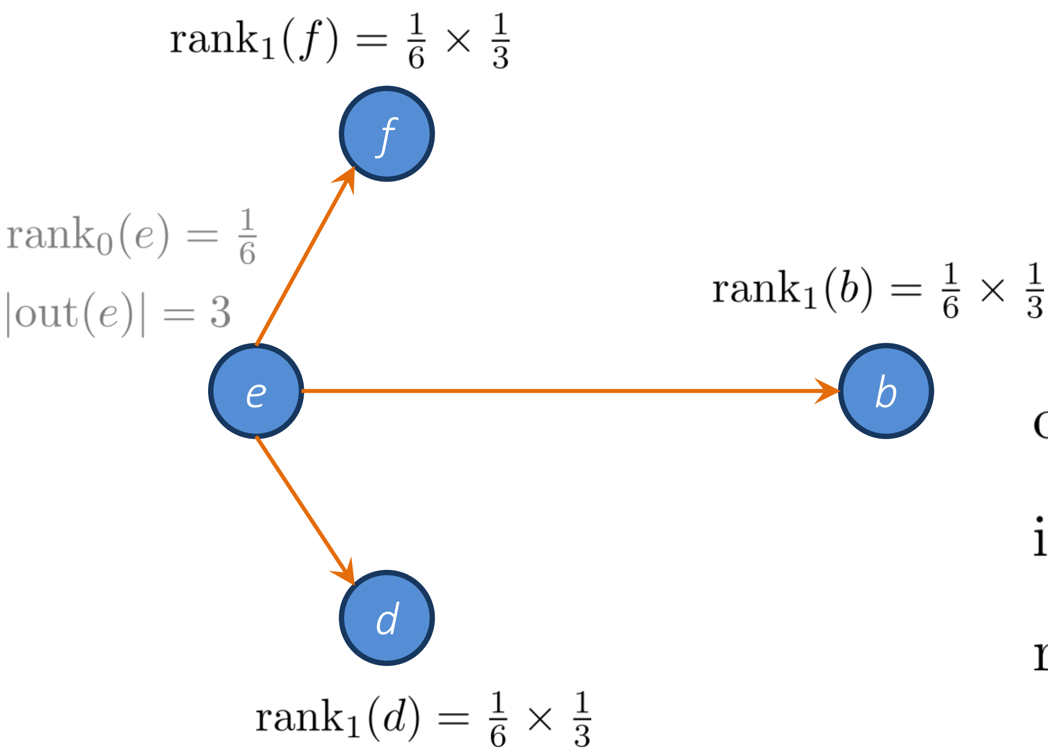
$$\mathrm{rank}_i(v) \coloneqq \sum_{v' \in \mathrm{in}(v)} \frac{\mathrm{rank}_{i-1}(v')}{|\mathrm{out}(v')|}$$

PageRank Model

$$G = (\boxed{V}, \boxed{E})$$

Vertices (*pages*)

Edges (*links*)



$$\mathrm{out}(v) \coloneqq \{v' \in V \mid (v, v') \in E\}$$

$$\mathrm{in}(v) \coloneqq \{v' \in V \mid (v', v) \in E\}$$

$$\mathrm{rank}_0(v) \coloneqq \frac{1}{|V|}$$

$$\mathrm{rank}_i(v) \coloneqq \sum_{v' \in \mathrm{in}(v)} \frac{\mathrm{rank}_{i-1}(v')}{|\mathrm{out}(v')|}$$

PageRank Model

$$G = (V, E)$$

Vertices (*pages*)

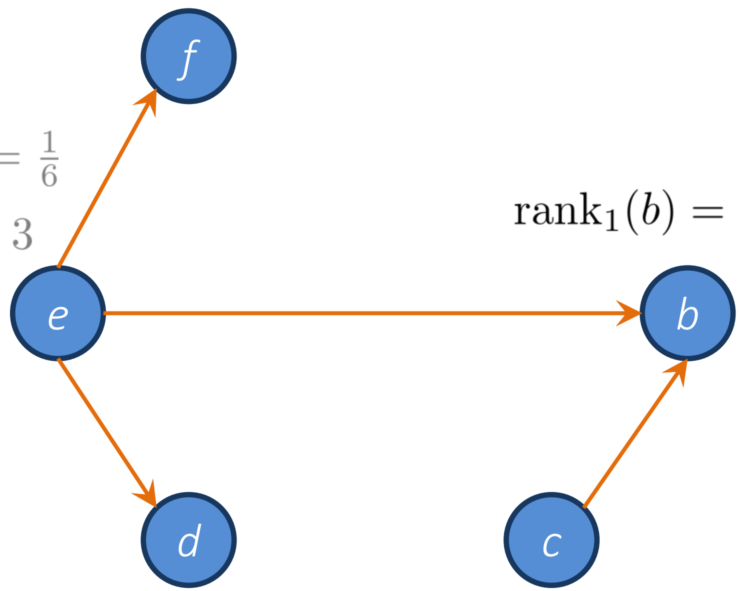
Edges (*links*)

$\mathrm{rank}_1(f) = \frac{1}{6} \times \frac{1}{3}$

$\mathrm{rank}_0(e) = \frac{1}{6}$

$|\mathrm{out}(e)| = 3$

$\mathrm{rank}_1(b) = \frac{1}{6} \times \frac{1}{3} + 1 \times \frac{1}{6}$



$\mathrm{rank}_1(d) = \frac{1}{6} \times \frac{1}{3}$

$\mathrm{rank}_0(c) = \frac{1}{6}$

$|\mathrm{out}(c)| = 1$

$$\mathrm{out}(v) \coloneqq \{v' \in V \mid (v, v') \in E\}$$

$$\mathrm{in}(v) \coloneqq \{v' \in V \mid (v', v) \in E\}$$

$$\mathrm{rank}_0(v) \coloneqq \frac{1}{|V|}$$

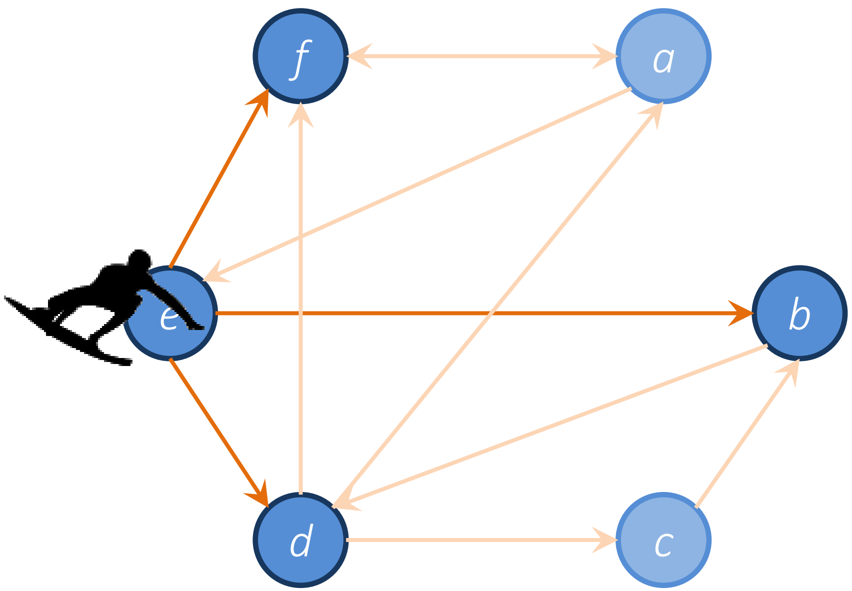
$$\mathrm{rank}_i(v) \coloneqq \sum_{v' \in \mathrm{in}(v)} \frac{\mathrm{rank}_{i-1}(v')}{|\mathrm{out}(v')|}$$

• • •

PageRank: Random Surfer Model

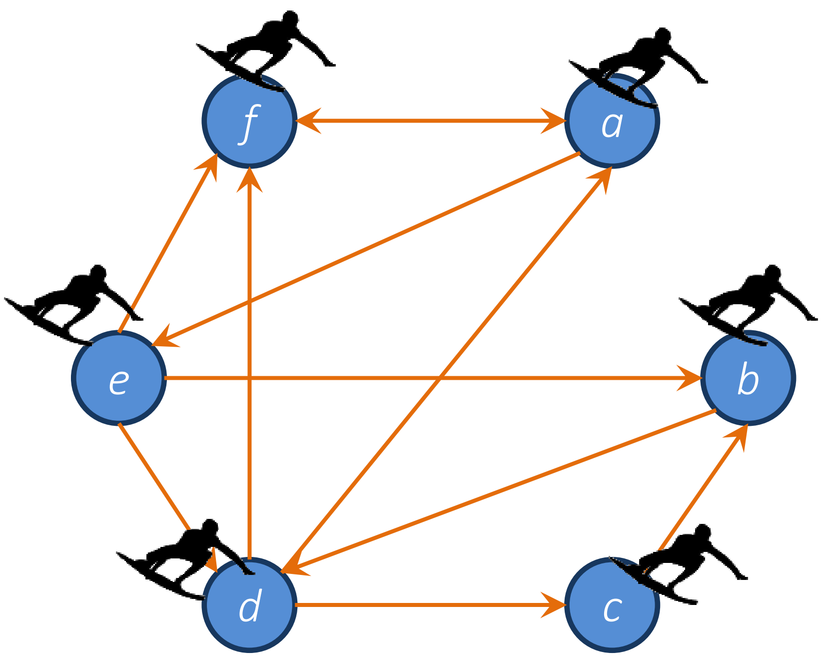
= someone surfing the web, clicking links randomly

- What is the probability of being at page *x* after *n* hops?



PageRank: Random Surfer Model

= someone surfing the web, clicking links randomly

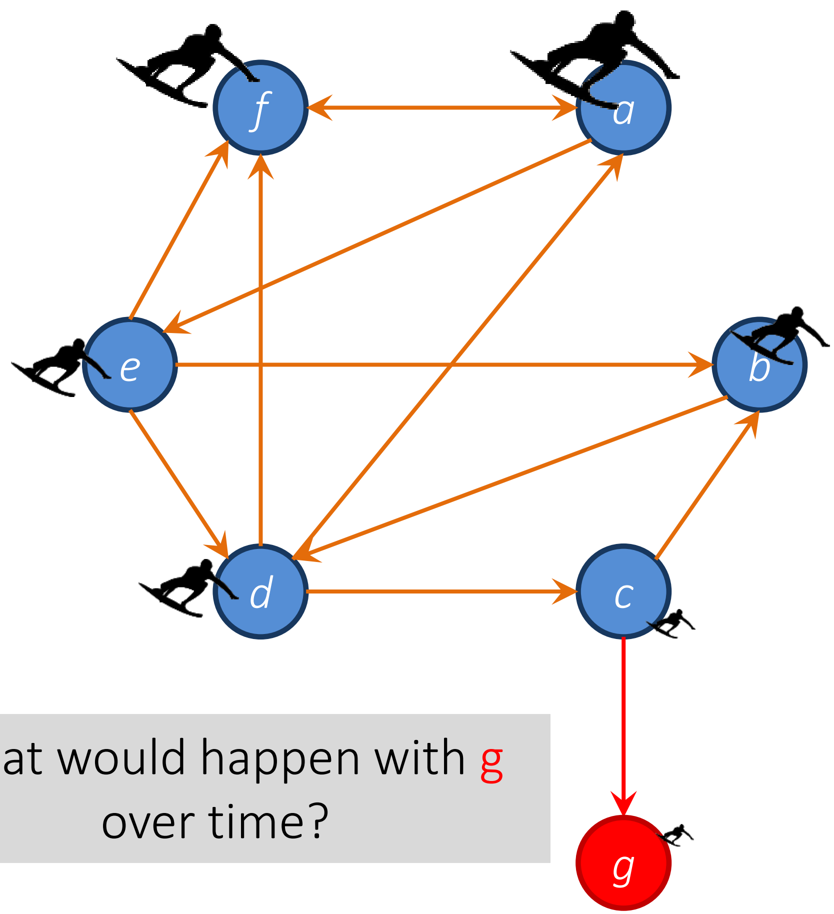


- What is the probability of being at page *x* after *n* hops?
- *Initial state:* surfer equally likely to start at any node

PageRank: Random Surfer Model

= someone surfing the web, clicking links randomly

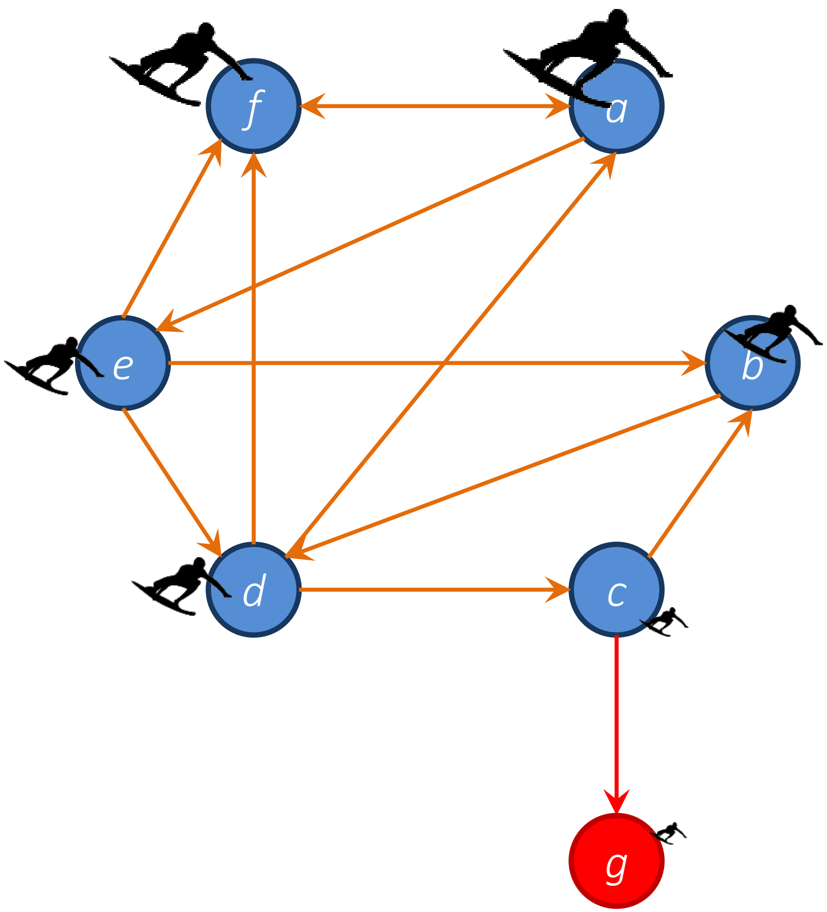
- What is the probability of being at page *x* after *n* hops?
- *Initial state:* surfer equally likely to start at any node
- PageRank applied iteratively for each hop: score indicates probability of being at that page after that many hops



What would happen with g over time?

PageRank: Random Surfer Model

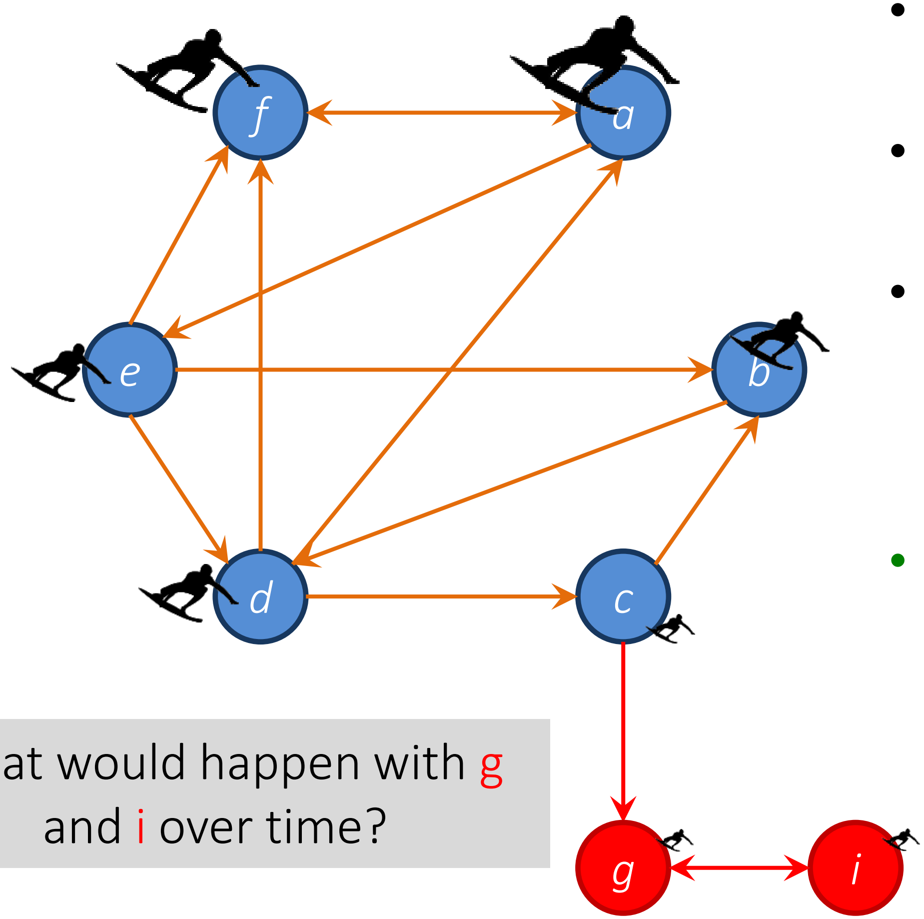
= someone surfing the web, clicking links randomly



- What is the probability of being at page *x* after *n* hops?
- *Initial state:* surfer equally likely to start at any node
- PageRank applied iteratively for each hop: score indicates probability of being at that page after that many hops
- If the surfer reaches a page without links, the surfer randomly jumps to another page

PageRank: Random Surfer Model

= someone surfing the web, clicking links randomly



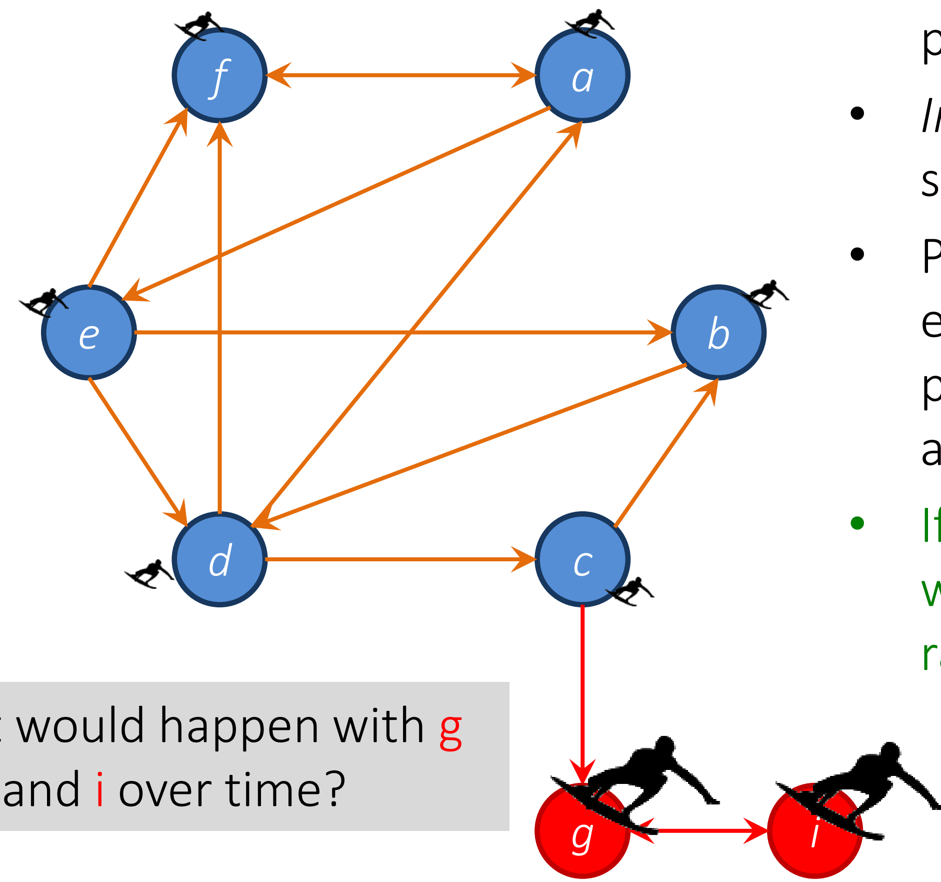
What would happen with g and i over time?

- What is the probability of being at page *x* after *n* hops?
- *Initial state:* surfer equally likely to start at any node
- PageRank applied iteratively for each hop: score indicates probability of being at that page after that many hops
- If the surfer reaches a page without links, the surfer randomly jumps to another page

PageRank: Random Surfer Model

 = someone surfing the web, clicking links randomly

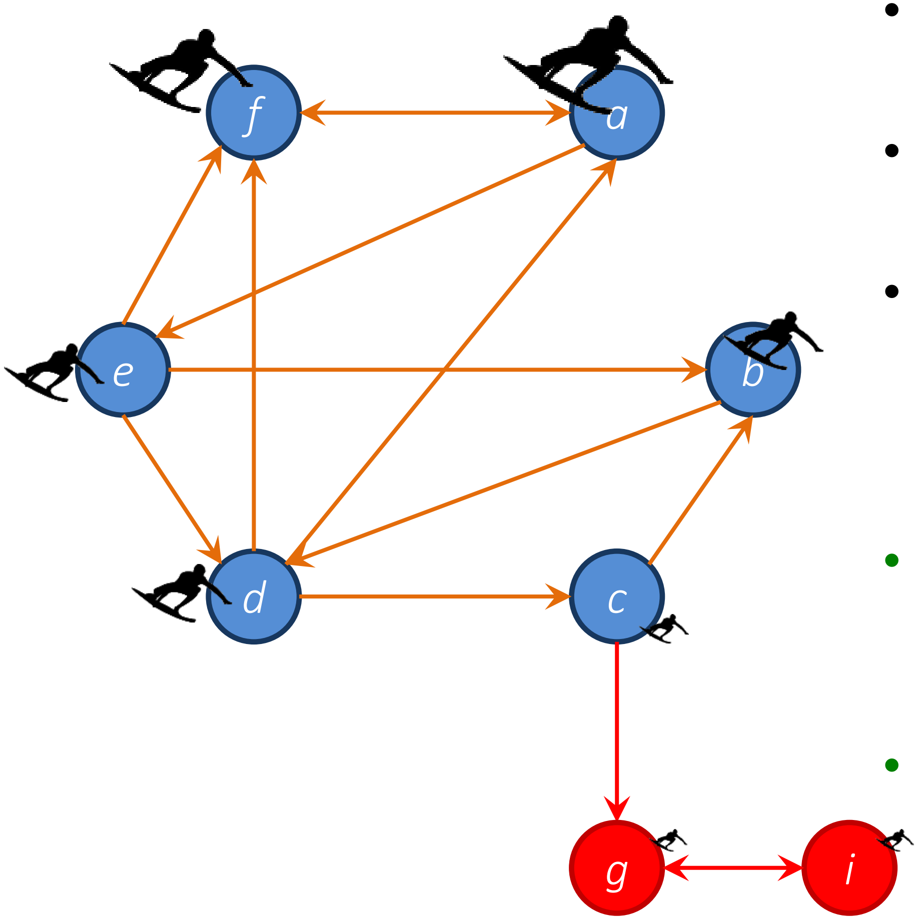
- What is the probability of being at page *x* after *n* hops?
- *Initial state:* surfer equally likely to start at any node
- PageRank applied iteratively for each hop: score indicates probability of being at that page after than many hops
- If the surfer reaches a page without out-links, the surfer randomly jumps to another page



What would happen with g and i over time?

PageRank: Random Surfer Model

= someone surfing the web, clicking links randomly



- What is the probability of being at page *x* after *n* hops?
- *Initial state:* surfer equally likely to start at any node
- PageRank applied iteratively for each hop: score indicates probability of being at that page after than many hops
- If the surfer reaches a page without out-links, the surfer randomly jumps to another page
- The surfer will jump to a random page at any time with a probability 1 – *d* ... *this avoids traps and ensures convergence!*

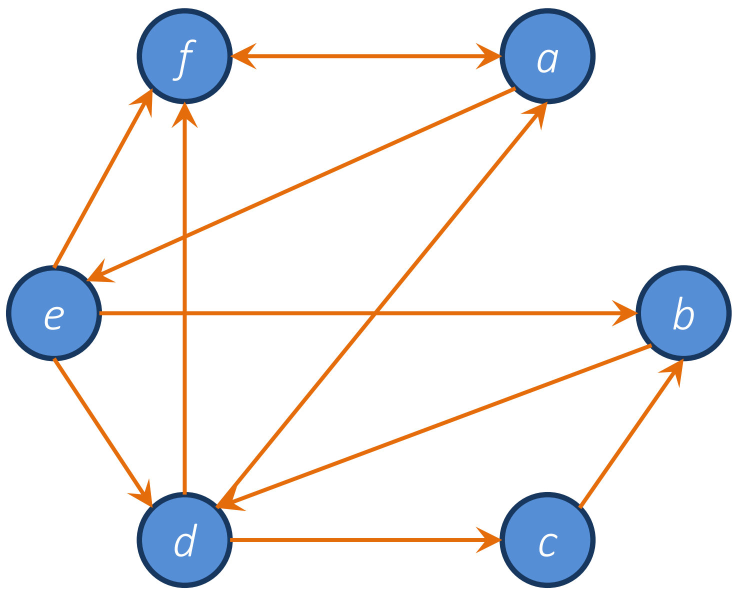
PageRank Model: Final Version

- The Web: a directed graph

$$G = (\boxed{V}, \boxed{E})$$

Vertices (*pages*)

Edges (*links*)



$$\mathrm{out}(v) \coloneqq \{v' \in V \mid (v, v') \in E\}$$

$$\mathrm{in}(v) \coloneqq \{v' \in V \mid (v', v) \in E\}$$

$$\mathrm{rank}_0(v) \coloneqq \frac{1}{|V|}$$

$$V' \coloneqq \{v \in V : |\mathrm{out}(v)| = 0\}$$

$$V'' \coloneqq \{v \in V : |\mathrm{out}(v)| \neq 0\}$$

d is the follow-a-link probability

typically $(d = 0.85)$

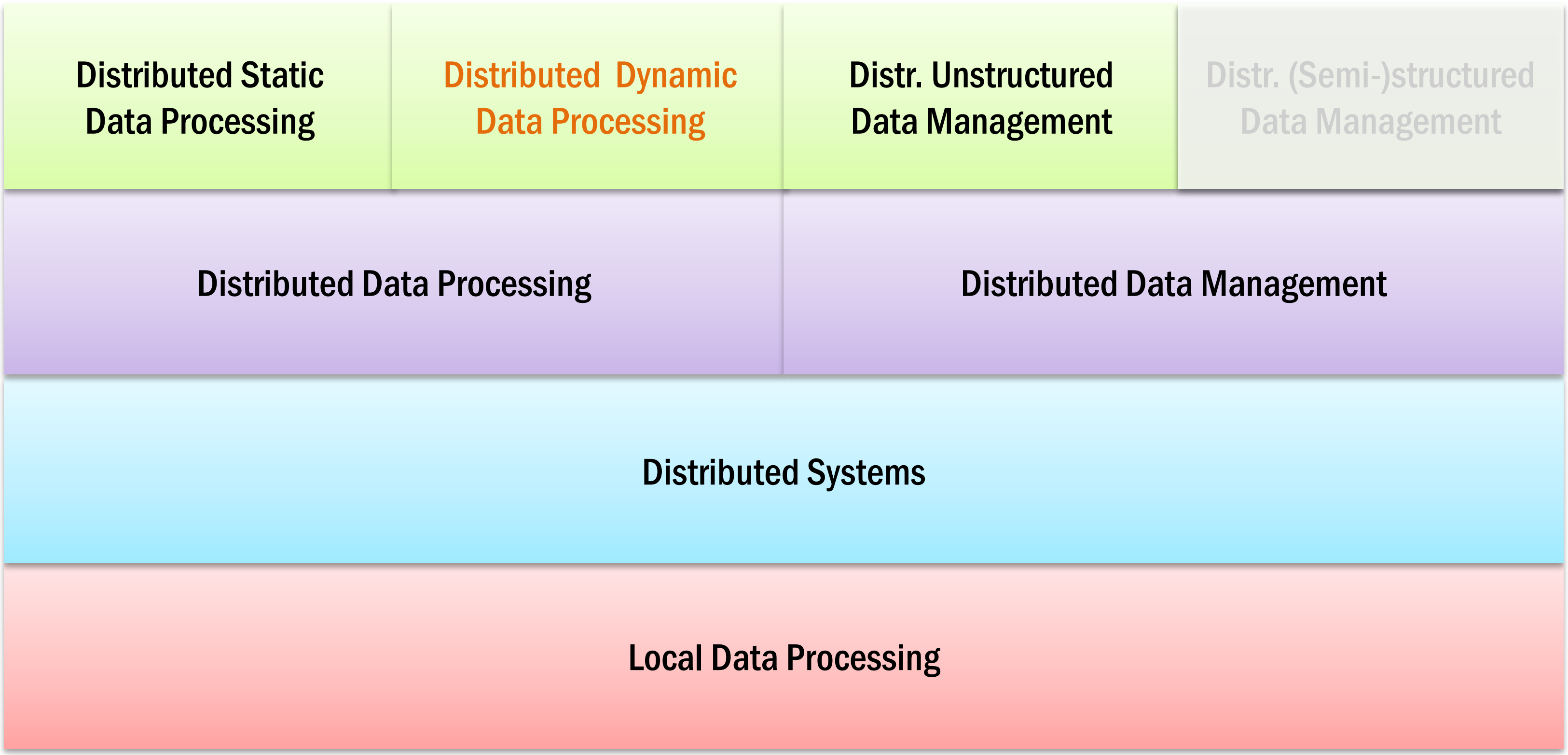
$$\mathrm{rank}_i(v) \coloneqq \boxed{d \times \sum_{u \in \mathrm{in}(v)} \frac{\mathrm{rank}_{i-1}(u)}{|\mathrm{out}(u)|}} + \boxed{\sum_{v' \in V'} \frac{\mathrm{rank}_{i-1}(v')}{|V|}} + \boxed{(1-d) \times \sum_{v'' \in V''} \frac{\mathrm{rank}_{i-1}(v'')}{|V|}}$$

PageRank: Benefits

- ✓ More robust than a simple link count
- ✓ Fewer ties than link counting
- ✓ Scalable to approximate (for sparse graphs)
- ✓ Convergence guaranteed

Two Sides to Ranking: Importance

COMPUTING PAGERANK AT SCALE


Distributed Static Data Processing
Distributed Dynamic Data Processing
Distr. Unstructured Data Management
Distr. (Semi-)structured Data Management
Distributed Data Processing
Distributed Data Management
Distributed Systems
Local Data Processing

Graph Parallel Frameworks: Pregel

Pregel: A System for Large-Scale Graph Processing

Grzegorz Malewicz, Matthew H. Austern, Aart J. C. Bik, James C. Dehnert, Ilan Horn,
Naty Leiser, and Grzegorz Czajkowski
Google, Inc.
{malewicz,austern,ajcbik,dehnert,ilan,naty,gczaj}@google.com

ABSTRACT

Many practical computing problems concern large graphs. Standard examples include the Web graph and various social networks. The scale of these graphs—in some cases billions of vertices, trillions of edges—poses challenges to their efficient processing. In this paper we present a computational model suitable for this task. Programs are expressed as a sequence of iterations, in each of which a vertex can receive messages sent in the previous iteration, send messages to other vertices, and modify its own state and that of its outgoing edges or mutate graph topology. This vertex-centric approach is flexible enough to express a broad set of algorithms. The model has been designed for efficient, scalable and fault-tolerant implementation on clusters of thousands of commodity computers, and its implied synchronicity makes reasoning about programs easier. Distribution-related details are hidden behind an abstract API. The result is a framework for processing large graphs that is expressive and easy to program.

disease outbreaks, or citation relationships among published scientific work—have been processed for decades. Frequently applied algorithms include shortest paths computations, different flavors of clustering, and variations on the page rank theme. There are many other graph computing problems of practical value, *e.g.*, minimum cut and connected components.

Efficient processing of large graphs is challenging. Graph algorithms often exhibit poor locality of memory access, very little work per vertex, and a changing degree of parallelism over the course of execution [31, 39]. Distribution over many machines exacerbates the locality issue, and increases the probability that a machine will fail during computation. Despite the ubiquity of large graphs and their commercial importance, we know of no scalable general-purpose system for implementing arbitrary graph algorithms over arbitrary graph representations in a large-scale distributed environment.

Implementing an algorithm to process a large graph typically means choosing among the following options:

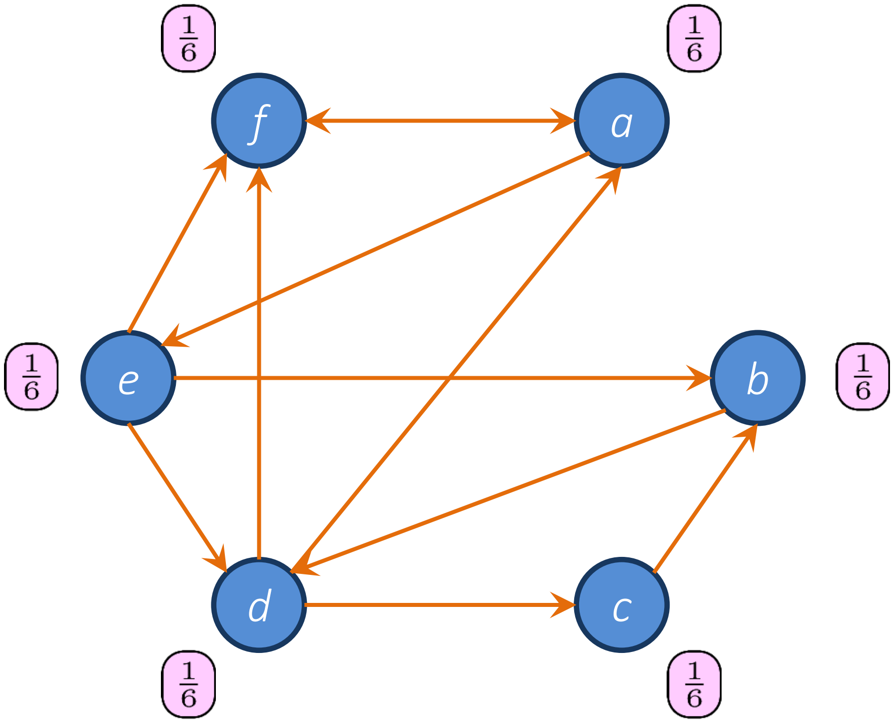
Graph Parallel Frameworks: Open Source

APACHE GIRAPH

GraphX

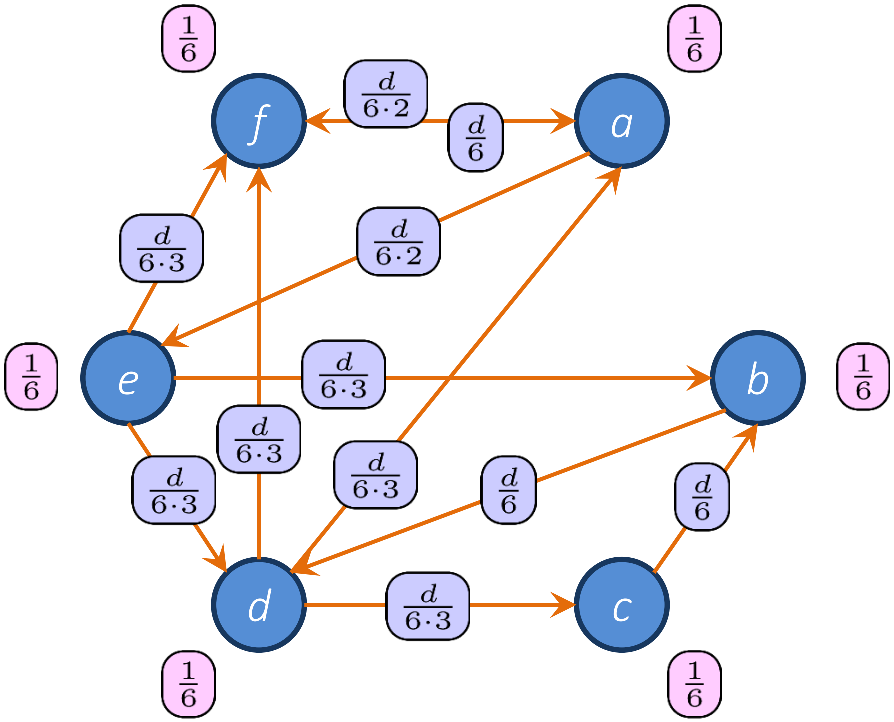
Flink Gelly

Vertex-Centric Computation



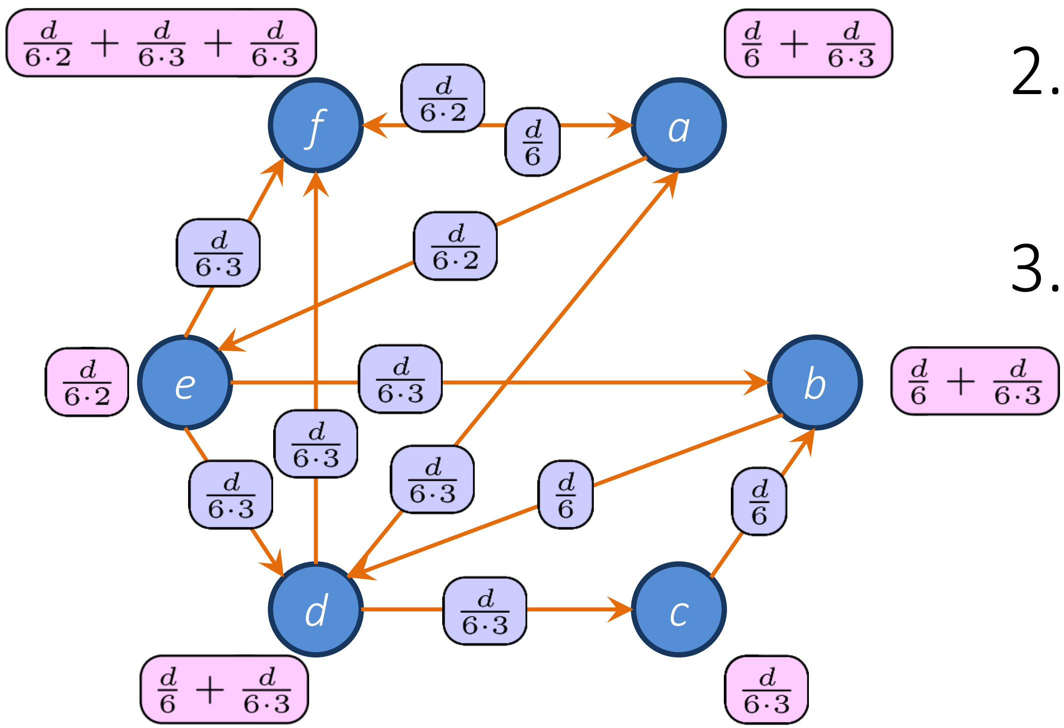
1. Nodes assigned state

Vertex-Centric Computation



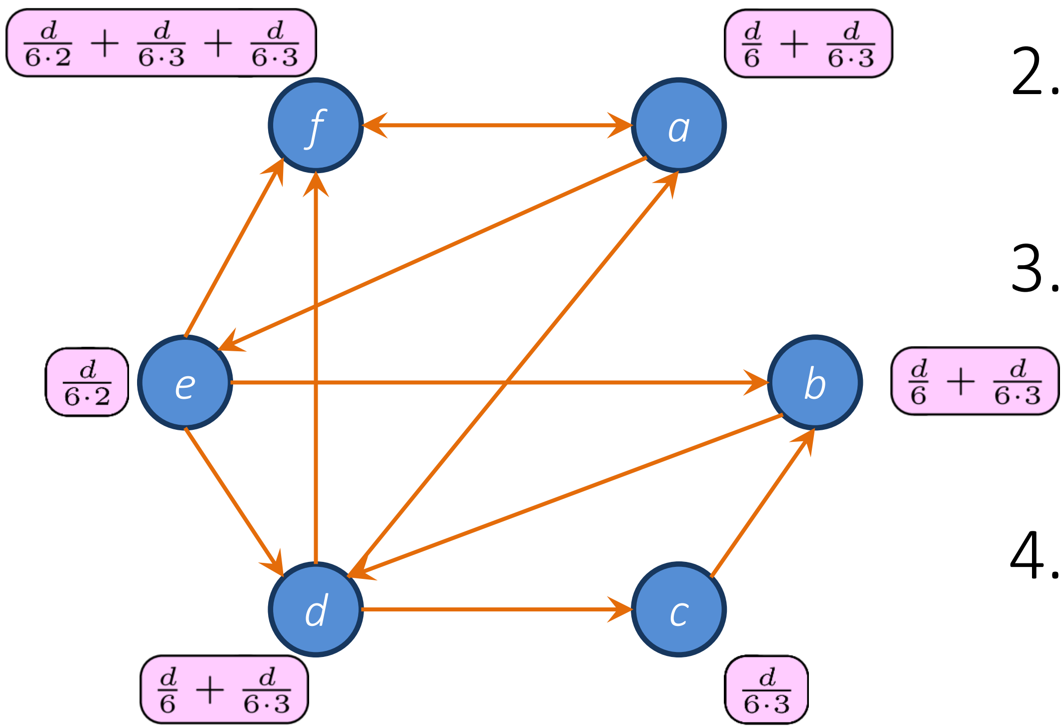
1. Nodes assigned state
2. Nodes pass messages (typically along edges)

Vertex-Centric Computation



1. Nodes assigned state
2. Nodes pass messages (typically along edges)
3. Nodes aggregate messages received and update state

Vertex-Centric Computation



1. Nodes assigned state
2. Nodes pass messages (typically along edges)
3. Nodes aggregate messages received and update state
4. GOTO 2. until some termination criteria are reached

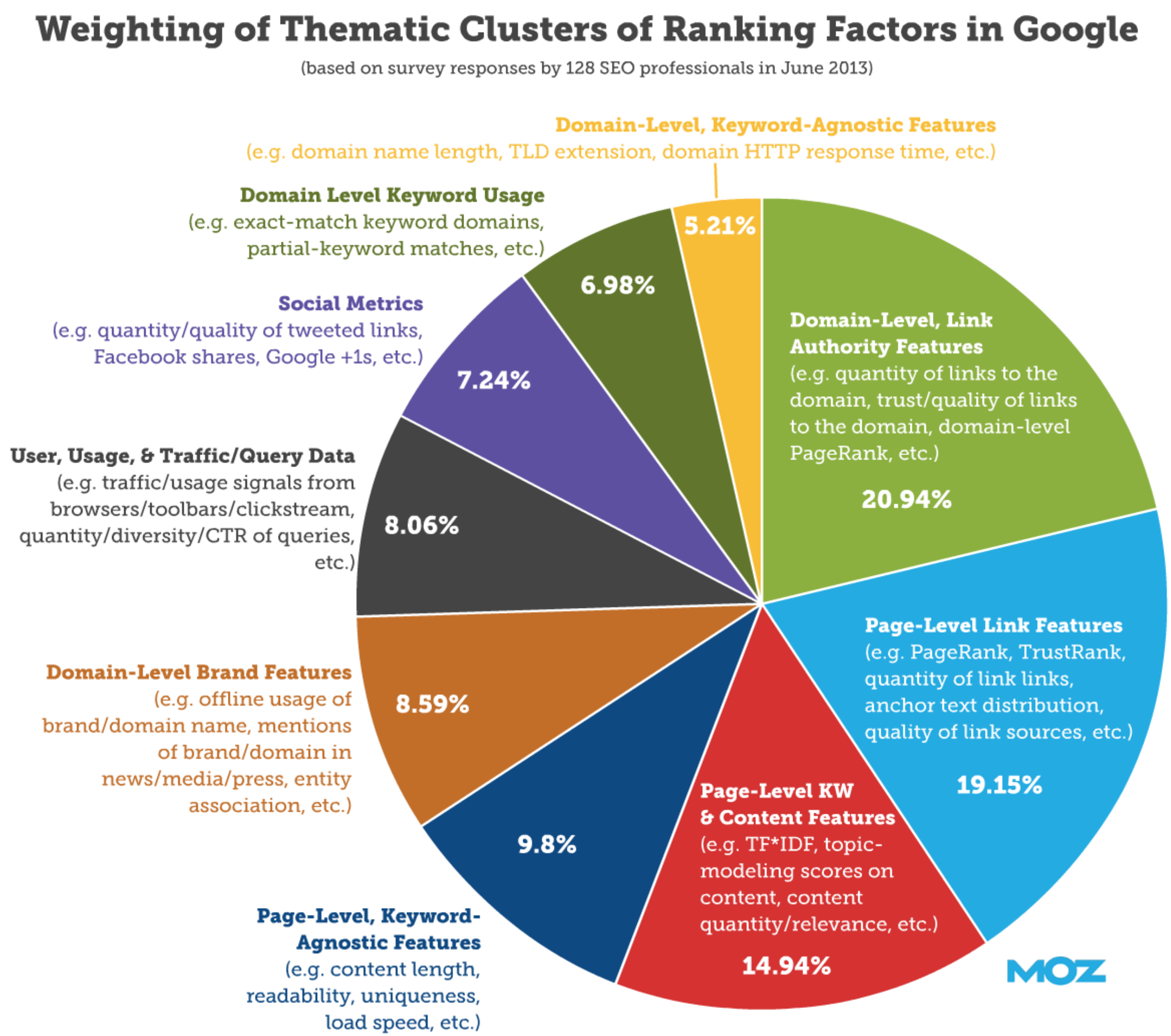
Vertex-Centric Computation: Other Features

- Message passing and aggregation done in parallel
- Optional message passing to non-neighbours
- Optional global “aggregation” phase
- Optional changes to the graph topology

How does google really rank?

An educated guess

How Modern Google ranks results (maybe)



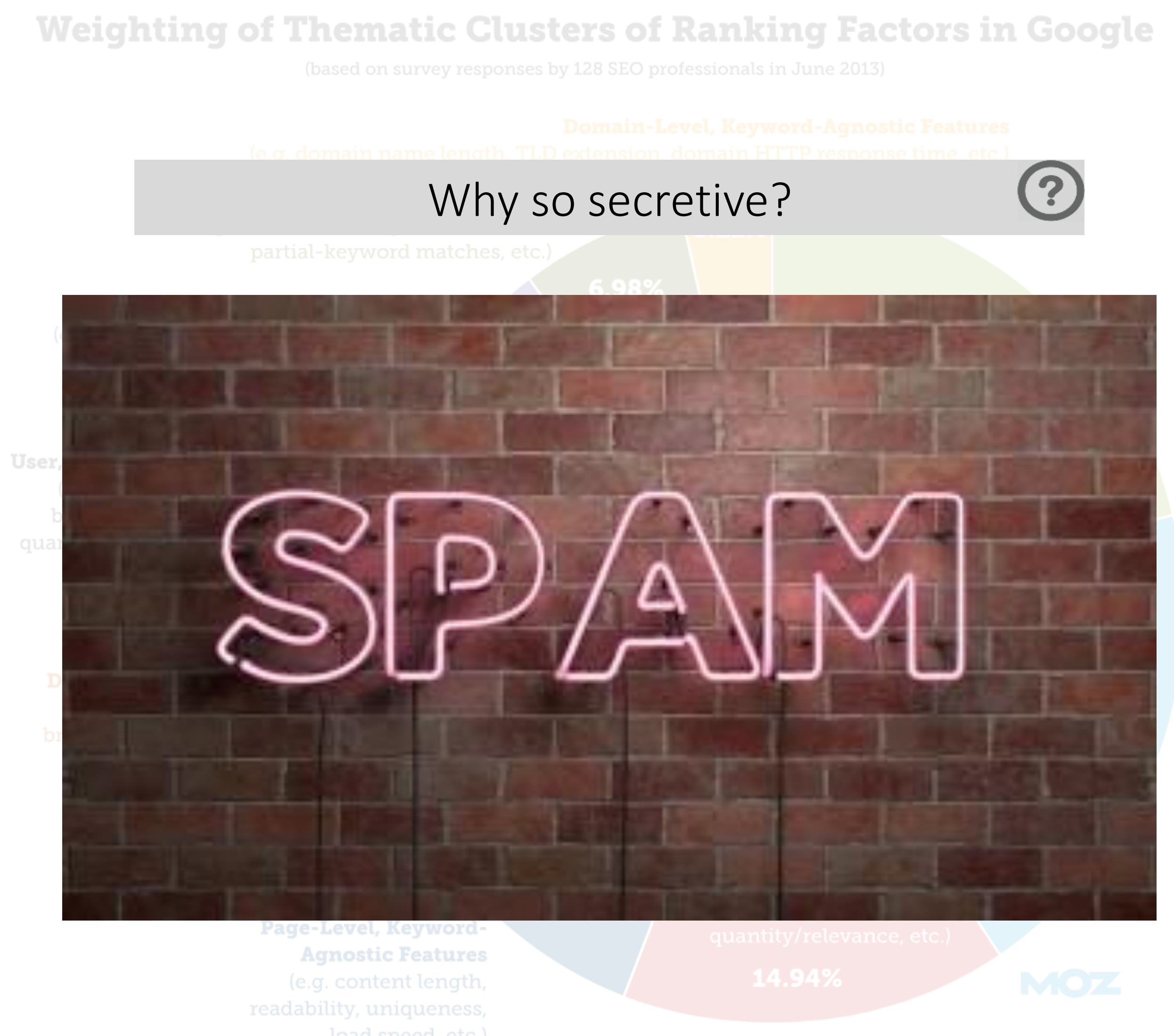
According to survey of SEO experts, not people in Google

How Modern Google ranks results (maybe)

According to survey of SEO experts, not people in Google

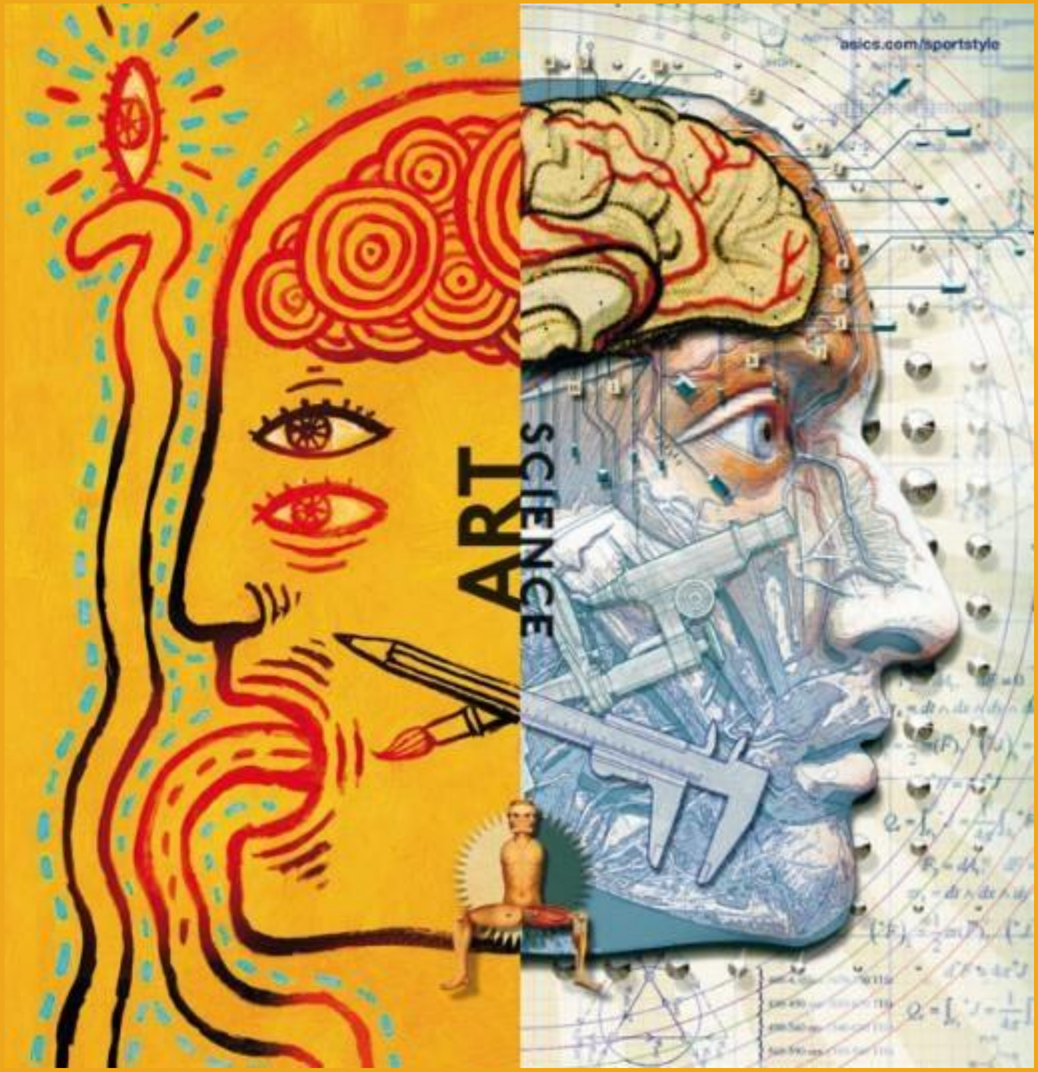
How Modern Google ranks results (maybe)

Why so secretive?



According to survey of SEO experts, not people in Google

Ranking: Science or Art?



Questions?