INFO-H-509 XML Technologies
Searching and Ranking

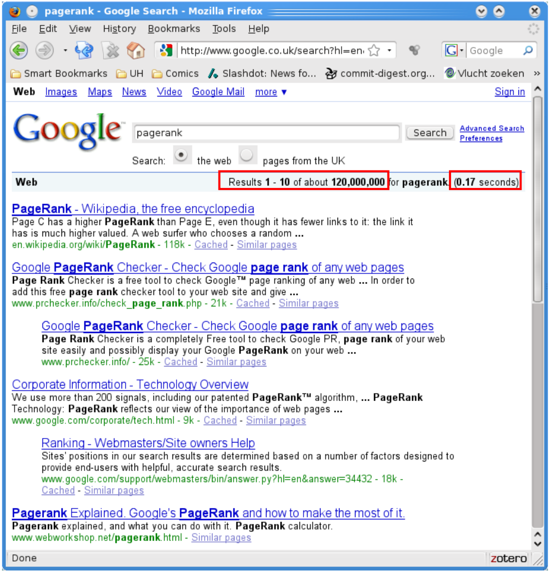
Stijn Vansummeren

May 6, 2010

Navigating the Web

- Directly via a known URL (e.g., `http://www.apple.com/ipad/`)
- By following links from a known web page
- By using a search engine

Searching the web is like searching a needle in a haystack ...

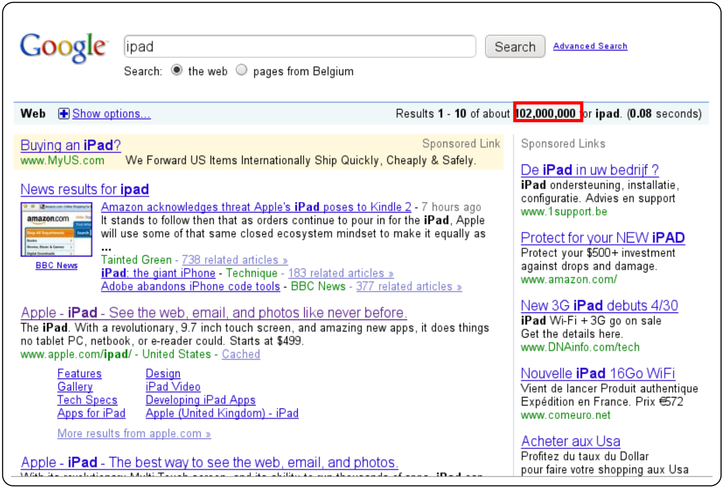


How do they do this?

(show Google video)

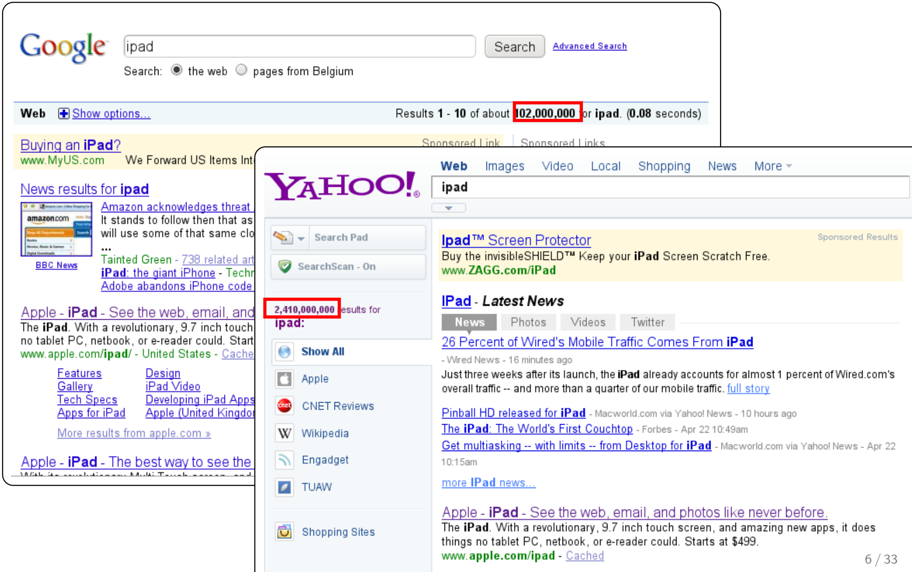
A Remark

Search Engines actually only search part of the web!



A Remark

Search Engines actually only search part of the web!



The Internet in some Numbers

How many websites are there?

- April 2010: 205 **million** sites (according to `netcraft.com`)

To compare:

- 205 million seconds = 6.5 years

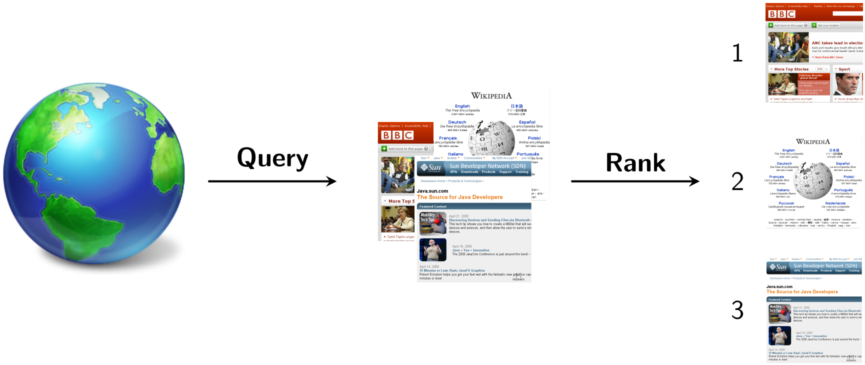
The Internet in some Numbers

Each website has multiple web pages

- wikipedia (more than 3 million)
- La libre belgique
- The BBC website
- ...

- Google stopped counting in 2004: **8 billion pages**
- Yahoo (2005): **19.2 billion** $\simeq$ **600 years**

Ranking: the general idea



Goal:

- Return a **list** of web pages matching the search terms ...
- ... such that the **relevant** pages occur within the first 10 results.

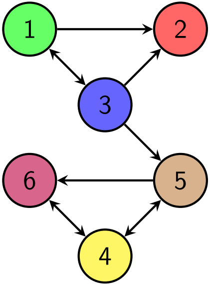
The pre-Google way of ranking

A web page is more relevant w.r.t. a search query if:

- the frequency of a search term in a web page is high;
- a search term occurs in the page's title;
- a search term occurs in bold font;
- ...

By this measure, a web page is important if it says that its important.

Google's idea: Exploit the link structure of the Web



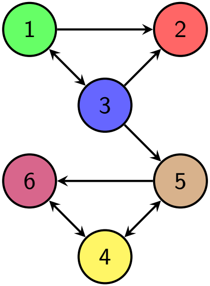
- A link from page A to page B is a vote by A that B is "important"
- So a page with many incoming links is more important than a page with few incoming links.
- But votes from "important" pages are more important than votes from "insignificant" pages
- And votes from pages with many outgoing links are less important than votes from pages with few outgoing links.

By this measure, "importance" is a democratic concept, independent of the search term!

PageRank version 1

Definition

The PageRank x_i of a page i is given by $x_i = \sum_{j \in B_i} \frac{x_j}{N_j}$ where B_i is the set of pages linking to i and N_j is the number of outgoing links on page j.



$$x_1 = \frac{1}{3}x_3$$
$$x_2 = \frac{1}{2}x_1 + \frac{1}{3}x_3$$
$$x_3 = \frac{1}{2}x_1$$
$$x_4 = \frac{1}{2}x_5 + \frac{1}{2}x_6$$
$$x_5 = \frac{1}{3}x_3 + \frac{1}{2}x_4$$
$$x_6 = \frac{1}{2}x_4 + \frac{1}{2}x_5$$

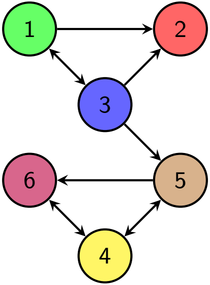
? Does such a system always have a **positive solution**?

PageRank version 1 - matrix notation

Definition

Let n be the number of pages in the web graph. The **hyperlink matrix H** is the $n \times n$ matrix such that

$$\mathbf{H}_{i,j} = \begin{cases} 1/N_j & \text{if } j \text{ links to } i \\ 0 & \text{otherwise} \end{cases}$$



$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} = \begin{bmatrix} 0 & 0 & \frac{1}{3} & 0 & 0 & 0 \\ \frac{1}{2} & 0 & \frac{1}{3} & 0 & 0 & 0 \\ \frac{1}{2} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & \frac{1}{3} & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix}$$

$$\mathbf{x} = \mathbf{H} \cdot \mathbf{x}$$

Some linear algebra terminology

Definition

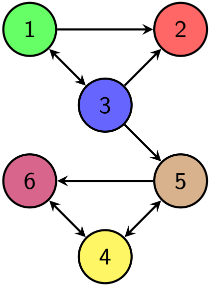
Let $\mathbf{A}$ be a matrix. A number λ and non-zero vector $\mathbf{v}$ satisfying

$$\lambda \mathbf{v} = \mathbf{A} \cdot \mathbf{v}$$

is called an **eigenvalue** and **eigenvector** of $\mathbf{A}$, respectively.

So searching for a **positive solution** to the equation $\mathbf{x} = \mathbf{H} \cdot \mathbf{x}$ actually means that we're searching for an **eigenvector** of $\mathbf{H}$ with **eigenvalue** 1.

A solution does not always exist

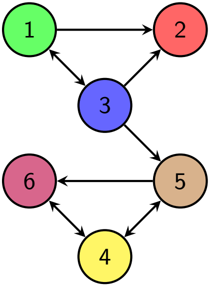


$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} = \begin{bmatrix} 0 & 0 & \frac{1}{3} & 0 & 0 & 0 \\ \frac{1}{2} & 0 & \frac{1}{3} & 0 & 0 & 0 \\ \frac{1}{2} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & \frac{1}{3} & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix}$$

A standard calculation reveals that H's only eigenvalues are:

$$-\frac{1}{2}, -0.3090169943749474, -0.4082482904638630,$$
$$0, 0.4082482904638630, 0.8090169943749474$$

An alternative view on PageRank — Random Surfer



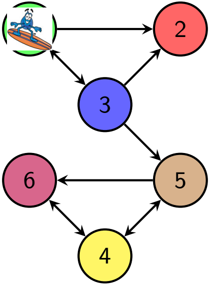
- A user starts surfing by entering a valid URL uniformly at random.
- At a each page, the user moves to a neighboring page by clicking a link on the page uniformly at random.
- The user keeps surfing forever.
- The **probability** p_i that the user visits page i is then given by

$$p_i = \sum_{j \in B_i} \frac{p_j}{N_j}$$

So in essence:

- The equation $\mathbf{x} = H\mathbf{x}$ then asks, for each page, the probability x_i that our random surfer visits page i.
- The higher the probability, the more important the page.

An alternative view on PageRank — Random Surfer



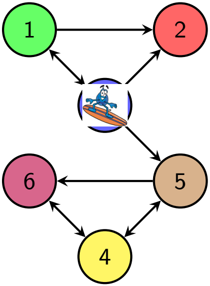
- A user starts surfing by entering a valid URL uniformly at random.
- At a each page, the user moves to a neighboring page by clicking a link on the page uniformly at random.
- The user keeps surfing forever.
- The **probability** p_i that the user visits page i is then given by

$$p_i = \sum_{j \in B_i} \frac{p_j}{N_j}$$

So in essence:

- The equation $\mathbf{x} = H\mathbf{x}$ then asks, for each page, the probability x_i that our random surfer visits page i.
- The higher the probability, the more important the page.

An alternative view on PageRank — Random Surfer



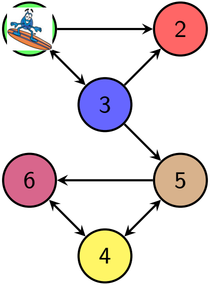
- A user starts surfing by entering a valid URL uniformly at random.
- At a each page, the user moves to a neighboring page by clicking a link on the page uniformly at random.
- The user keeps surfing forever.
- The **probability** p_i that the user visits page i is then given by

$$p_i = \sum_{j \in B_i} \frac{p_j}{N_j}$$

So in essence:

- The equation $\mathbf{x} = H\mathbf{x}$ then asks, for each page, the probability x_i that our random surfer visits page i.
- The higher the probability, the more important the page.

An alternative view on PageRank — Random Surfer



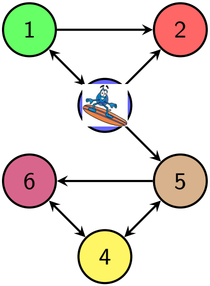
- A user starts surfing by entering a valid URL uniformly at random.
- At a each page, the user moves to a neighboring page by clicking a link on the page uniformly at random.
- The user keeps surfing forever.
- The **probability** p_i that the user visits page i is then given by

$$p_i = \sum_{j \in B_i} \frac{p_j}{N_j}$$

So in essence:

- The equation $\mathbf{x} = H\mathbf{x}$ then asks, for each page, the probability x_i that our random surfer visits page i.
- The higher the probability, the more important the page.

An alternative view on PageRank — Random Surfer



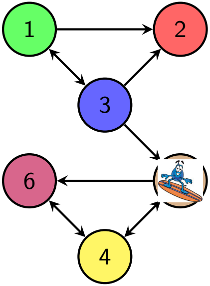
- A user starts surfing by entering a valid URL uniformly at random.
- At a each page, the user moves to a neighboring page by clicking a link on the page uniformly at random.
- The user keeps surfing forever.
- The **probability** p_i that the user visits page i is then given by

$$p_i = \sum_{j \in B_i} \frac{p_j}{N_j}$$

So in essence:

- The equation $\mathbf{x} = H\mathbf{x}$ then asks, for each page, the probability x_i that our random surfer visits page i.
- The higher the probability, the more important the page.

An alternative view on PageRank — Random Surfer



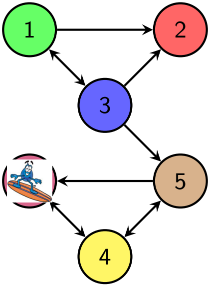
- A user starts surfing by entering a valid URL uniformly at random.
- At a each page, the user moves to a neighboring page by clicking a link on the page uniformly at random.
- The user keeps surfing forever.
- The **probability** p_i that the user visits page i is then given by

$$p_i = \sum_{j \in B_i} \frac{p_j}{N_j}$$

So in essence:

- The equation $\mathbf{x} = H\mathbf{x}$ then asks, for each page, the probability x_i that our random surfer visits page i.
- The higher the probability, the more important the page.

An alternative view on PageRank — Random Surfer



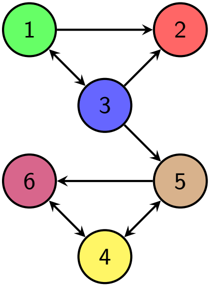
- A user starts surfing by entering a valid URL uniformly at random.
- At a each page, the user moves to a neighboring page by clicking a link on the page uniformly at random.
- The user keeps surfing forever.
- The **probability** p_i that the user visits page i is then given by

$$p_i = \sum_{j \in B_i} \frac{p_j}{N_j}$$

So in essence:

- The equation $\mathbf{x} = H\mathbf{x}$ then asks, for each page, the probability x_i that our random surfer visits page i.
- The higher the probability, the more important the page.

An alternative view on PageRank — Random Surfer



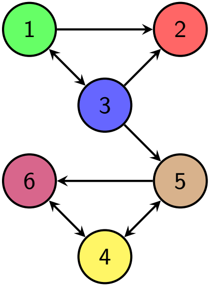
- A user starts surfing by entering a valid URL uniformly at random.
- At a each page, the user moves to a neighboring page by clicking a link on the page uniformly at random.
- The user keeps surfing forever.
- The **probability** p_i that the user visits page i is then given by

$$p_i = \sum_{j \in B_i} \frac{p_j}{N_j}$$

So in essence:

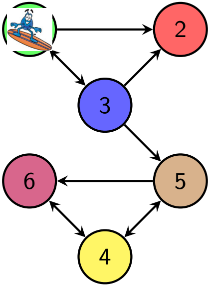
- The equation $\mathbf{x} = H\mathbf{x}$ then asks, for each page, the probability x_i that our random surfer visits page i.
- The higher the probability, the more important the page.

So why is there no solution?



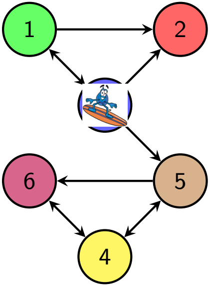
- A user starts surfing by entering a valid URL uniformly at random.
- At a each page, the user moves to a neighboring page by clicking a link on the page uniformly at random.
- **But the user gets stuck at "dangling" nodes**

So why is there no solution?



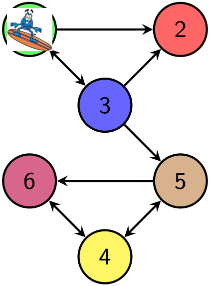
- A user starts surfing by entering a valid URL uniformly at random.
- At a each page, the user moves to a neighboring page by clicking a link on the page uniformly at random.
- **But the user gets stuck at "dangling" nodes**

So why is there no solution?



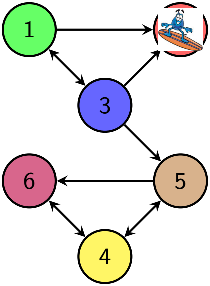
- A user starts surfing by entering a valid URL uniformly at random.
- At a each page, the user moves to a neighboring page by clicking a link on the page uniformly at random.
- **But the user gets stuck at "dangling" nodes**

So why is there no solution?



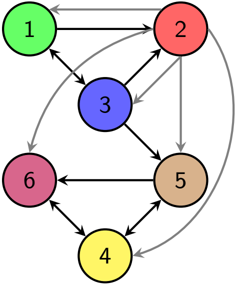
- A user starts surfing by entering a valid URL uniformly at random.
- At a each page, the user moves to a neighboring page by clicking a link on the page uniformly at random.
- **But the user gets stuck at "dangling" nodes**

So why is there no solution?



- A user starts surfing by entering a valid URL uniformly at random.
- At a each page, the user moves to a neighboring page by clicking a link on the page uniformly at random.
- **But the user gets stuck at "dangling" nodes**

PageRank version 2: correct for dangling nodes



$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & \frac{1}{6} & \frac{1}{3} & 0 & 0 & 0 \\ \frac{1}{2} & \frac{1}{6} & \frac{1}{3} & 0 & 0 & 0 \\ \frac{1}{2} & \frac{1}{6} & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{6} & 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & \frac{1}{6} & \frac{1}{3} & \frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{6} & 0 & \frac{1}{2} & \frac{1}{2} & 0 \end{bmatrix}}_{\text{corrected hyperlink matrix } \mathbf{S}} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix}$$

Definition

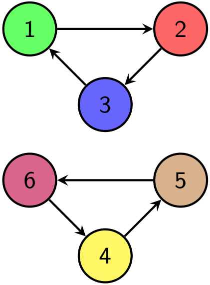
A matrix is **stochastic** if it contains only positive coefficients and all of its columns add to 1

Theorem

Every stochastic square matrix has 1 as eigenvalue

So, the equation $\mathbf{x} = \mathbf{S} \cdot \mathbf{x}$ always has a solution.

Is there only one solution?



$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix}$$

S has multiple eigenvectors with eigenvalue 1**, for instance :**

$$\begin{bmatrix} \frac{1}{3} \\ \frac{1}{3} \\ \frac{1}{3} \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad \begin{bmatrix} 0 \\ 0 \\ 0 \\ \frac{1}{3} \\ \frac{1}{3} \\ \frac{1}{3} \end{bmatrix} \quad \begin{bmatrix} \frac{1}{6} \\ \frac{1}{6} \\ \frac{1}{6} \\ \frac{1}{6} \\ \frac{1}{6} \\ \frac{1}{6} \end{bmatrix} \quad \begin{bmatrix} \frac{1}{p} \\ \frac{1}{p} \\ \frac{1}{p} \\ \frac{1}{q} \\ \frac{1}{q} \\ \frac{1}{q} \end{bmatrix}$$

PageRank - final version

Modify the random surfer model. At each site the surfer:

- visits a random neighbouring site with probability α;
- or "teleports" to an arbitrary site with probability $1-\alpha$.

Google takes $\alpha = 85\%$

- **Probability that surfer visits site i:** $x_i = \alpha \sum_{j \in B_i} \frac{x_j}{N_j} + (1-\alpha) \times \frac{1}{n}$
- In matrix notation:

$$\mathbf{x} = \underbrace{\left(\alpha S + \frac{1-\alpha}{n} E_{n \times n} \right)}_{\text{Google matrix G}} \cdot \mathbf{x}$$

Theorem

The Google matrix G has a **single** eigenvector with eigenvalue 1. Moreover, this eigenvector is **stochastic**.

Computing the PageRank vector

Theorem

Let $\mathbf{v^0}$ be an arbitrary column vector that adds to 1. Let $\mathbf{v^k} := \mathbf{G} \cdot \mathbf{v^{k-1}}$. Then the pagerank vector equals $\lim_{k \to \infty} \mathbf{v^k}$.

Hence, to approximate the pagerank vector with tolerance ε:

- Let n be the total number of web pages
- Initialize $\mathbf{v}$ to $(\frac{1}{n}, \ldots, \frac{1}{n})^T$
- Initialize $\mathbf{v}'$ to $(1, \ldots, 1)^T$
- Repeat until $\|\mathbf{v}' - \mathbf{v}\| < \varepsilon$:
  - set $\mathbf{v}' := \mathbf{v}$
  - set $\mathbf{v} := \mathbf{G} \cdot \mathbf{v}$

According to Google, this algorithm converges to the PageRank vector within a reasonable tolerance for web graphs of 322 million links after roughly 52 iterations.

Computing the PageRank vector (2)

- Each multiplication $\mathbf{G} \cdot \mathbf{v}$ takes $O(n^2)$ time.
- However, n is HUGE: in 2004 Google reported that it had indexed 8×10^9 web pages.
- Using the fact that on average every page has only 10 outgoing links, one can use results from linear algebra to perform this multiplication in $O(n)$ time.
- Still, Google is reported to calculate the PageRank vector only once a month!