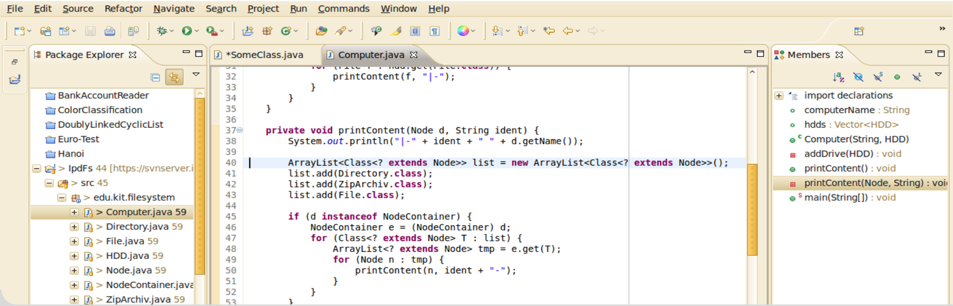
Karlsruher Institut für Technologie

How Google searches work

History, Algorithm

Martin Thoma, Benjamin | 24. Januar 2013

FAKULTÄT FÜR INFORMATIK



Inhaltsverzeichnis

The early days

In the beginning, there were only web catalogues (maintained by hand)

The early days

Web crawler

Basics

Karlsruher Institut für Technologie

- Humans know what is good for them
- Humans create Websites
- Humans will only link to Websites they like

⇒ Hyperlinks are a quality indicator

Basics

- Humans know what is good for them
- Humans create Websites
- Humans will only link to Websites they like

$\Rightarrow$ Hyperlinks are a quality indicator

Basics

- Humans know what is good for them
- Humans create Websites
- Humans will only link to Websites they like

⇒ Hyperlinks are a quality indicator

Basics

- Humans know what is good for them
- Humans create Websites
- Humans will only link to Websites they like

$\Rightarrow$ Hyperlinks are a quality indicator

How could we use that?

- Simply count number of links to a Website

✗ 10,000 links from only one page

- Count numbers of Websites that link to a Website

✗ Quality of the page matters

✗ Total number of links on the source page matters

How could we use that?

Karlsruher Institut für Technologie

- Simply count number of links to a Website
- ✗ 10,000 links from only one page
- Count numbers of Websites that link to a Website
- ✗ Quality of the page matters
- ✗ Total number of links on the source page matters

How could we use that?

- Simply count number of links to a Website
- ✗ 10,000 links from only one page
- Count numbers of Websites that link to a Website
- ✗ Quality of the page matters
- ✗ Total number of links on the source page matters

How could we use that?

Karlsruher Institut für Technologie

- Simply count number of links to a Website
- ✗ 10,000 links from only one page
- Count numbers of Websites that link to a Website
- ✗ Quality of the page matters
- ✗ Total number of links on the source page matters

How could we use that?

- Simply count number of links to a Website
- ✗ 10,000 links from only one page
- Count numbers of Websites that link to a Website
- ✗ Quality of the page matters
- ✗ Total number of links on the source page matters

A brilliant idea

Ideas of PageRank

- Decisions of humans are complicated
- A lot of webpages get visited

$\Rightarrow$ modellize clicks on links as random behaviour

- Links are important
- Links of page A get less important, if A has many links
- Links of page A get more important, if many link to A

$\Rightarrow$ if B has a link from A, the rank of B increases by $\frac{Rank(A)}{Links(A)}$

if A links to B **then**
$$Rank(B) \mathrel{+}= \frac{Rank(A)}{Links(A)}$$

Ideas of PageRank

- Decisions of humans are complicated
- A lot of webpages get visited

$\Rightarrow$ modellize clicks on links as random behaviour

- Links are important
- Links of page A get less important, if A has many links
- Links of page A get more important, if many link to A

$\Rightarrow$ if B has a link from A, the rank of B increases by $\frac{Rank(A)}{Links(A)}$

if A links to B **then**

$$Rank(B) \mathrel{+}= \frac{Rank(A)}{Links(A)}$$

Ideas of PageRank

Karlsruher Institut für Technologie

- Decisions of humans are complicated
- A lot of webpages get visited

$\Rightarrow$ modellize clicks on links as random behaviour

- Links are important
- Links of page A get less important, if A has many links
- Links of page A get more important, if many link to A

$\Rightarrow$ if B has a link from A, the rank of B increases by $\frac{Rank(A)}{Links(A)}$

if A links to B **then**

$$Rank(B) \mathrel{+}= \frac{Rank(A)}{Links(A)}$$

Ideas of PageRank

- Decisions of humans are complicated
- A lot of webpages get visited

⇒ modellize clicks on links as random behaviour

- Links are important
- Links of page A get less important, if A has many links
- Links of page A get more important, if many link to A

⇒ if B has a link from A, the rank of B increases by $\frac{Rank(A)}{Links(A)}$

if A links to B **then**

$$Rank(B) \mathrel{+}= \frac{Rank(A)}{Links(A)}$$

Ideas of PageRank

- Decisions of humans are complicated
- A lot of webpages get visited

$\Rightarrow$ modellize clicks on links as random behaviour

- Links are important
- Links of page A get less important, if A has many links
- Links of page A get more important, if many link to A

$\Rightarrow$ if B has a link from A, the rank of B increases by $\frac{Rank(A)}{Links(A)}$

if A links to B **then**

$$Rank(B) \mathrel{+}= \frac{Rank(A)}{Links(A)}$$

Ideas of PageRank

- Decisions of humans are complicated
- A lot of webpages get visited

⇒ modellize clicks on links as random behaviour

- Links are important
- Links of page A get less important, if A has many links
- Links of page A get more important, if many link to A

⇒ if B has a link from A, the rank of B increases by $\frac{Rank(A)}{Links(A)}$

if A links to B **then**

$Rank(B) \mathrel{+}= \frac{Rank(A)}{Links(A)}$

Ideas of PageRank

- Decisions of humans are complicated
- A lot of webpages get visited

$\Rightarrow$ modellize clicks on links as random behaviour

- Links are important
- Links of page A get less important, if A has many links
- Links of page A get more important, if many link to A

$\Rightarrow$ if B has a link from A, the rank of B increases by $\frac{Rank(A)}{Links(A)}$

if A links to B **then**

$$Rank(B) \mathrel{+}= \frac{Rank(A)}{Links(A)}$$

Ideas of PageRank

- Decisions of humans are complicated
- A lot of webpages get visited

$\Rightarrow$ modellize clicks on links as random behaviour

- Links are important
- Links of page A get less important, if A has many links
- Links of page A get more important, if many link to A

$\Rightarrow$ if B has a link from A, the rank of B increases by $\frac{Rank(A)}{Links(A)}$

if A links to B **then**

$$Rank(B) \mathrel{+}= \frac{Rank(A)}{Links(A)}$$

Ants

- Websites = nodes = anthill
- Links = edges = paths
- You place ants on each node
- They walk over the paths
  (at random, they are ants!)
- After some time, some anthills will have more ants than others
- Those hills are more attractive than others
- # ants is probability that a random user would end on a website

Ants

- Websites = nodes = anthill
- Links = edges = paths
- You place ants on each node
- They walk over the paths
  (at random, they are ants!)
- After some time, some anthills will have more ants than others
- Those hills are more attractive than others
- # ants is probability that a random user would end on a website

Ants

- Websites = nodes = anthill
- Links = edges = paths
- You place ants on each node
- They walk over the paths
  (at random, they are ants!)
- After some time, some anthills will have more ants than others
- Those hills are more attractive than others
- # ants is probability that a random user would end on a website

Ants

Karlsruher Institut für Technologie

- Websites = nodes = anthill
- Links = edges = paths
- You place ants on each node
- They walk over the paths

  (at random, they are ants!)
- After some time, some anthills will have more ants than others
- Those hills are more attractive than others
- # ants is probability that a random user would end on a website

Ants

- Websites = nodes = anthill
- Links = edges = paths
- You place ants on each node
- They walk over the paths
  (at random, they are ants!)
- After some time, some anthills will have more ants than others
- Those hills are more attractive than others
- # ants is probability that a random user would end on a website

Ants

- Websites = nodes = anthill
- Links = edges = paths
- You place ants on each node
- They walk over the paths
  (at random, they are ants!)
- After some time, some anthills will have more ants than others
- Those hills are more attractive than others
- # ants is probability that a random user would end on a website

Ants

Karlsruher Institut für Technologie

- Websites = nodes = anthill
- Links = edges = paths
- You place ants on each node
- They walk over the paths
  (at random, they are ants!)
- After some time, some anthills will have more ants than others
- Those hills are more attractive than others
- # ants is probability that a random user would end on a website

Ants

- Websites = nodes = anthill
- Links = edges = paths
- You place ants on each node
- They walk over the paths
  (at random, they are ants!)
- After some time, some anthills will have more ants than others
- Those hills are more attractive than others
- # ants is probability that a random user would end on a website

Mathematics

Let x be a web page. Then

- $L(x)$ is the set of Websites that link to x
- $C(y)$ is the out-degree of page y
- α is probability of random jump
- N is the total number of websites

$$PR(x) := \alpha \left(\frac{1}{N} \right) + (1 - \alpha) \sum_{y \in L(x)} \frac{PR(y)}{C_y}$$

Pseudocode

Karlsruher Institut für Technologie

```
function PageRank(Graph web, double q = 0.15, int iterations)
    for all page ∈ G do
        page.pageRank = 1/|G|                          ▷ intial probability
    end for
    while iterations > 0 do
        for all page ∈ G do                            ▷ calculate pageRank of page
            page.pageRank = q
            for all y ∈ L(page) do
                page.pageRank += y.pageRank / C(y)
            end for
        end for
        iterations -= 1
```

Pseudocode

function $\textsc{PageRank}$(Graph web, double $q = 0.15$, int $iterations$)
    for all $page \in G$ **do**
        $page.pageRank = \frac{1}{|G|}$ ▷ intial probability
    end for
    while $iterations > 0$ **do**
        for all $page \in G$ **do** ▷ calculate pageRank of $page$
            $page.pageRank = q$
            for all $y \in L(page)$ **do**
                $page.pageRank \mathrel{+}= \frac{y.pageRank}{C(y)}$
            end for
        end for
        $iterations \mathrel{-}= 1$

Pseudocode

function PageRank(Graph web, double $q = 0.15$, int $iterations$)
  for all $page \in G$ **do**
    $page.pageRank = \frac{1}{|G|}$ ▷ intial probability
  end for
  while $iterations > 0$ **do**
    for all $page \in G$ **do** ▷ calculate pageRank of $page$
      $page.pageRank = q$
      for all $y \in L(page)$ **do**
        $page.pageRank \mathrel{+}= \frac{y.pageRank}{C(y)}$
      end for
    end for
    $iterations$ -= 1

Pseudocode

Karlsruher Institut für Technologie

```
function PageRank(Graph web, double q = 0.15, int iterations)
    for all page ∈ G do
        page.pageRank = 1/|G|                        ▷ intial probability
    end for
    while iterations > 0 do
        for all page ∈ G do                          ▷ calculate pageRank of page
            page.pageRank = q
            for all y ∈ L(page) do
                page.pageRank += y.pageRank / C(y)
            end for
        end for
        iterations -= 1
```

Pseudocode

Karlsruher Institut für Technologie

function PageRank(Graph web, double $q = 0.15$, int $iterations$)
  for all $page \in G$ **do**
    $page.pageRank = \frac{1}{|G|}$ ▷ intial probability
  end for
  while $iterations > 0$ **do**
    for all $page \in G$ **do** ▷ calculate pageRank of $page$
      $page.pageRank = q$
      for all $y \in L(page)$ **do**
        $page.pageRank \mathrel{+}= \frac{y.pageRank}{C(y)}$
      end for
    end for
    $iterations \mathrel{-}= 1$

Pseudocode

function PageRank(Graph web, double $q = 0.15$, int $iterations$)
    for all $page \in G$ **do**
        $page.pageRank = \frac{1}{|G|}$ ▷ intial probability
    end for
    while $iterations > 0$ **do**
        for all $page \in G$ **do** ▷ calculate pageRank of $page$
            $page.pageRank = q$
            for all $y \in L(page)$ **do**
                $page.pageRank \mathrel{+}= \frac{y.pageRank}{C(y)}$
            end for
        end for
        $iterations$ -= 1

Pseudocode

Karlsruher Institut für Technologie

function $\textsc{PageRank}$(Graph web, double $q = 0.15$, int $iterations$)
    for all $page \in G$ **do**
        $page.pageRank = \frac{1}{|G|}$ ▷ intial probability
    end for
    while $iterations > 0$ **do**
        for all $page \in G$ **do** ▷ calculate pageRank of $page$
            $page.pageRank = q$
            for all $y \in L(page)$ **do**
                $page.pageRank \mathrel{+}= \frac{y.pageRank}{C(y)}$
            end for
        end for
        $iterations \mathrel{-}= 1$

Pseudocode

```
function PageRank(Graph web, double q = 0.15, int iterations)
    for all page ∈ G do
        page.pageRank = 1/|G|                          ▷ intial probability
    end for
    while iterations > 0 do
        for all page ∈ G do                ▷ calculate pageRank of page
            page.pageRank = q
            for all y ∈ L(page) do
                page.pageRank += y.pageRank / C(y)
            end for
        end for
        iterations -= 1
```

What you've learned

- Web catalogues
- Webcrawler
- Graph (nodes, eges)
- Random walk (ants)
- PageRank
- Read Pseudocode

Image sources

- Sergey Brin by enlewof
- Larry Page by aweigend

Thank you for your attention!