Introduction to Machine Learning

Page Ranking and the Web

Andres Mendez-Vazquez

August 21, 2020

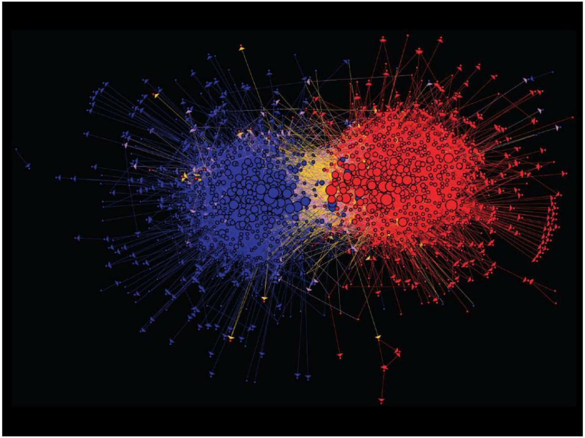
Outline

Graph Data: Social Networks

Facebook social graph

4-degrees of separation [Backstrom-Boldi-Rosa-Ugander-Vigna, 2011].

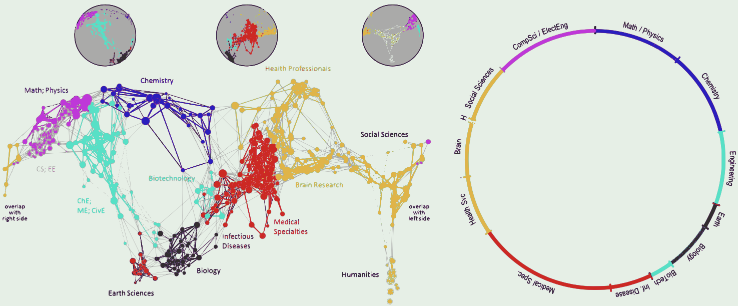
Graph Data: Media Networks



Connections between political blogs

Polarization of the network [Adamic-Glance, 2005].

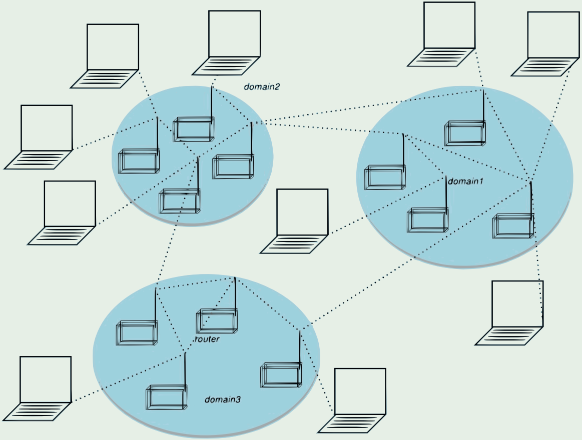
Graph Data: Information Nets



Citation networks and Maps of science

[Börner et al., 2012]

Graph Data: Communication Nets



Internet

Web as Graph

Web as a directed graph

- Nodes: Web-pages
- Edges: Hyperlinks

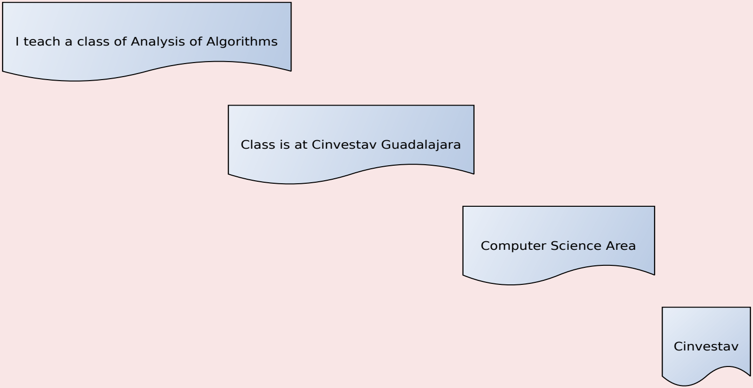
We can have the following Pages

Web as Graph

Web as a directed graph

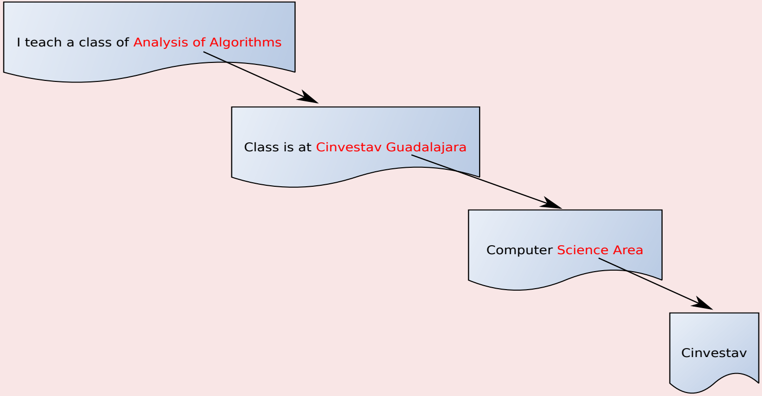
- Nodes: Web-pages
- Edges: Hyperlinks

We can have the following Pages

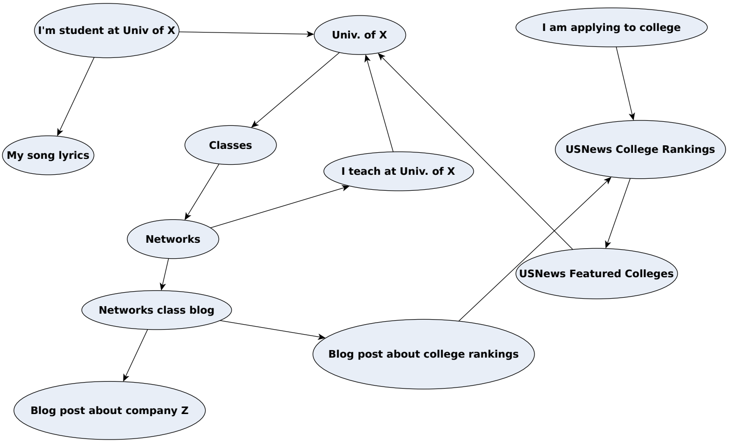


Web as Graph

Now Add the Edges



Another Example, A Semantic Web



Outline

Broad Question

- How to organize the Web?

Broad Question

- How to organize the Web?

First try

- Human curated Web directories
  - Yahoo, DMOZ, LookSmart

- Information Retrieval investigates: Find relevant docs in a small and trusted set
  - Newspaper articles, Patents, etc.
- But: the Web is huge, full of non-trustable documents, random things, web spam, etc.

Broad Question

- How to organize the Web?

First try

- Human curated Web directories
  - Yahoo, DMOZ, LookSmart

Second try

- Web Search
  - Information Retrieval investigates: Find relevant docs in a small and trusted set
    - Newspaper articles, Patents, etc.
  - But the Web is huge, full of non-trustable documents, random things, web spam, etc.

Broad Question

- How to organize the Web?

First try

- Human curated Web directories
  - Yahoo, DMOZ, LookSmart

Second try

- Web Search
  - Information Retrieval investigates: Find relevant docs in a small and trusted set
    - Newspaper articles, Patents, etc.
  - But the Web is huge, full of non-trustable documents, random things, web spam, etc.

Broad Question

- How to organize the Web?

First try

- Human curated Web directories
  - Yahoo, DMOZ, LookSmart

Second try

- Web Search
  - Information Retrieval investigates: Find relevant docs in a small and trusted set
    - ★ Newspaper articles, Patents, etc.
  - But the Web is huge, full of non-trustable documents, random things, web spam, etc.

Broad Question

- How to organize the Web?

First try

- Human curated Web directories
  - Yahoo, DMOZ, LookSmart

Second try

- Web Search
  - Information Retrieval investigates: Find relevant docs in a small and trusted set
    - Newspaper articles, Patents, etc.
  - But the Web is huge, full of non-trustable documents, random things, web spam, etc.

Outline

Web Search: Two Challenges

Thus

We have two main challenges on web search.

Web Search: Two Challenges

Thus

We have two main challenges on web search.

First

- Web contains many sources of information Who do you "trust"?
  - Trick: Trustworthy pages may point to each other!

- No single right answer
- Trick: Pages that actually know about newspapers might all be pointing to many newspapers

Web Search: Two Challenges

Thus

We have two main challenges on web search.

First

- Web contains many sources of information Who do you “trust”?
  - Trick: Trustworthy pages may point to each other!

Second

- What is the “best” answer to query “newspaper”?
  - No single right answer
  - Trick: Pages that actually know about newspapers might all be pointing to many newspapers

Web Search: Two Challenges

Thus

We have two main challenges on web search.

First

- Web contains many sources of information Who do you "trust"?
  - Trick: Trustworthy pages may point to each other!

Second

- What is the "best" answer to query "newspaper"?
  - No single right answer
  - Trick: Pages that actually know about newspapers might all be pointing to many newspapers

Web Search: Two Challenges

Thus

We have two main challenges on web search.

First

- Web contains many sources of information Who do you "trust"?
  - Trick: Trustworthy pages may point to each other!

Second

- What is the "best" answer to query "newspaper"?
  - No single right answer
  - Trick: Pages that actually know about newspapers might all be pointing to many newspapers

Web Search: Two Challenges

Thus

We have two main challenges on web search.

First

- Web contains many sources of information Who do you "trust"?
  - Trick: Trustworthy pages may point to each other!

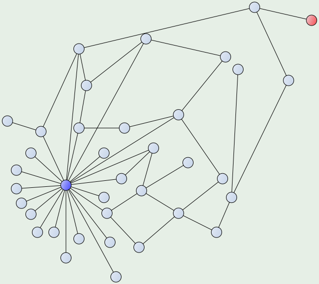
Second

- What is the "best" answer to query "newspaper"?
  - No single right answer
  - Trick: Pages that actually know about newspapers might all be pointing to many newspapers

Outline

Ranking Nodes on the Graph

All web pages are not equally "important" : www.joe-schmoe.com vs. www.stanford.edu

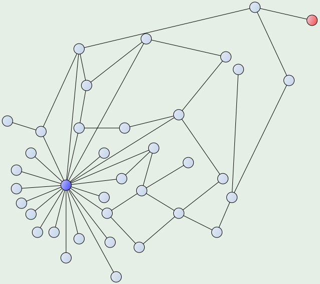


Web-graph node connectivity

- There is large diversity in the web-graph node connectivity. Let's rank the pages by the link structure!

Ranking Nodes on the Graph

All web pages are not equally "important" : www.joe-schmoe.com vs. www.stanford.edu



Web-graph node connectivity

- There is large diversity in the web-graph node connectivity. Let's rank the pages by the link structure!

Outline

Link Analysis Algorithms

We will cover the following Link Analysis approaches for computing importance of nodes in a graph

- Page Rank
- Hubs and Authorities (HITS)
- Topic-Specific (Personalized) Page Rank
- Web Spam Detection Algorithms

Link Analysis Algorithms

We will cover the following Link Analysis approaches for computing importance of nodes in a graph

- Page Rank
- Hubs and Authorities (HITS)
- Topic-Specific (Personalized) Page Rank
- Web Spam Detection Algorithms

Link Analysis Algorithms

We will cover the following Link Analysis approaches for computing importance of nodes in a graph

- Page Rank
- Hubs and Authorities (HITS)
- Topic-Specific (Personalized) Page Rank
- Web Spam Detection Algorithms

Link Analysis Algorithms

We will cover the following Link Analysis approaches for computing importance of nodes in a graph

- Page Rank
- Hubs and Authorities (HITS)
- Topic-Specific (Personalized) Page Rank
- Web Spam Detection Algorithms

Outline

Link as Votes

Idea

- Links as votes:
  - A Page is more important if it has more incoming links
  - In-coming links? Out-going links?

Link as Votes

Idea

- Links as votes:
  - A Page is more important if it has more incoming links
  - In-coming links? Out-going links?

Think of in-links as votes

- www.stanford.edu has 23,400 in-links
- www.joe-schmoe.com has 1 in-link

- Recursive question!

Link as Votes

Idea

- Links as votes:
  - A Page is more important if it has more incoming links
  - In-coming links? Out-going links?

Think of in-links as votes

- www.stanford.edu has 23, 400 in-links
- www.joe-schmoe.com has 1 in-link

Are all in-links are equal?

- Links from important pages count more
- Recursive question!

Link as Votes

Idea

- Links as votes:
  - A Page is more important if it has more incoming links
  - In-coming links? Out-going links?

Think of in-links as votes

- www.stanford.edu has $23,400$ in-links
- www.joe-schmoe.com has 1 in-link

Are all in-links are equal?

- Links from important pages count more
- Recursive question!

Link as Votes

Idea

- Links as votes:
  - A Page is more important if it has more incoming links
  - In-coming links? Out-going links?

Think of in-links as votes

- www.stanford.edu has $23,400$ in-links
- www.joe-schmoe.com has 1 in-link

Are all in-links are equal?

- Links from important pages count more
- Recursive question!

Link as Votes

Idea

- Links as votes:
  - A Page is more important if it has more incoming links
  - In-coming links? Out-going links?

Think of in-links as votes

- www.stanford.edu has $23,400$ in-links
- www.joe-schmoe.com has 1 in-link

Are all in-links are equal?

- Links from important pages count more
- Recursive question!

Link as Votes

Idea

- Links as votes:
  - A Page is more important if it has more incoming links
  - In-coming links? Out-going links?

Think of in-links as votes

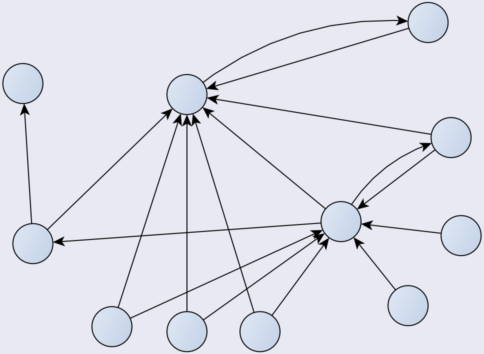
- www.stanford.edu has $23,400$ in-links
- www.joe-schmoe.com has 1 in-link

Are all in-links are equal?

- Links from important pages count more
- Recursive question!

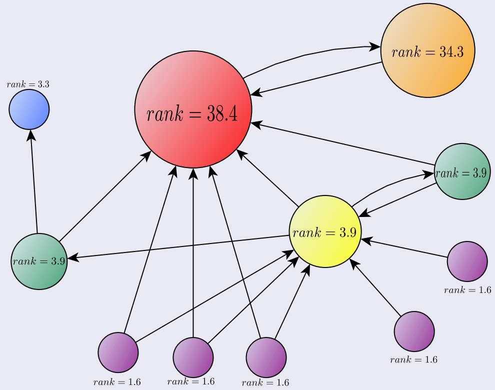
What do we want?

We want to generate Page Rank Scores from the structure



What do we want?

We want to generate Page Rank Scores from the structure



Simple Recursive Formulation

Link's vote

Each link's vote is proportional to the importance of its source page.

Out-links

If page j with importance r_j has n out-links, each link gets $\frac{r_j}{n}$ votes.

Simple Recursive Formulation

Link's vote

Each link's vote is proportional to the importance of its source page.

Out-links

If page j with importance r_j has n out-links, each link gets $\frac{r_j}{n}$ votes.

Simple Recursive Formulation

In-links

Page j's own importance is the sum of the votes on its in-links

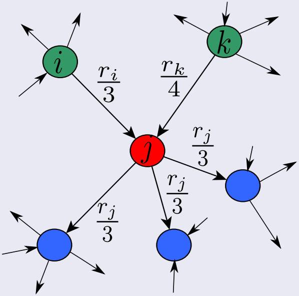
$$r_j = \frac{r_i}{3} + \frac{r_k}{4} \tag{1}$$

Simple Recursive Formulation

In-links

Page j's own importance is the sum of the votes on its in-links

$$r_j = \frac{r_i}{3} + \frac{r_k}{4} \tag{1}$$



Outline

Page Rank: The "Flow" Model

Voting

A "vote" from an important page is worth more

Importance

A page is important if it is pointed to by other important pages

Define a "rank" r_j for page j

$$r_j = \sum_{i \to j} \frac{r_i}{d_i}$$

d_i... out-degree of node i

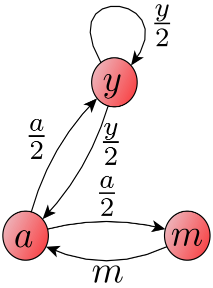
Flow Equations

$$r_y = \frac{r_y}{2} + \frac{r_a}{2}$$

$$r_a = \frac{r_y}{2} + r_m$$

$$r_m = \frac{r_a}{2}$$

The Web in 1839



Page Rank: The "Flow" Model

Voting

A "vote" from an important page is worth more

Importance

A page is important if it is pointed to by other important pages

Define a "rank" r_j for page j

$$r_j = \sum_{i \to j} \frac{r_i}{d_i}$$

$d_i \ldots$ out-degree of node i

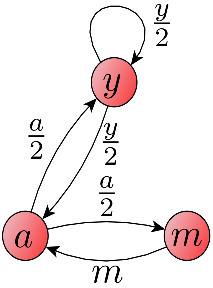
Flow Equations

$$r_y = \frac{r_y}{2} + \frac{r_a}{2}$$

$$r_a = \frac{r_y}{2} + r_m$$

$$r_m = \frac{r_a}{2}$$

The Web in 1839



Page Rank: The "Flow" Model

Voting

A "vote" from an important page is worth more

Importance

A page is important if it is pointed to by other important pages

Define a "rank" r_j for page j

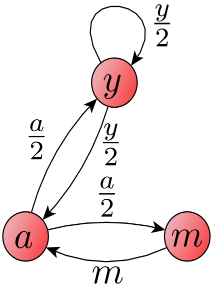
$$r_j = \sum_{i \to j} \frac{r_i}{d_i}$$

d_i... out-degree of node i

Flow Equations

$$r_y = \frac{r_y}{2} + \frac{r_a}{2}$$
$$r_a = \frac{r_y}{2} + r_m$$
$$r_m = \frac{r_a}{2}$$

The Web in 1839



Solving the Flow Equations

Something Notable

- Flow equations: 3 equations, 3 unknowns, no constants:
  - No unique solution.
  - All solutions equivalent modulo the scale factor.
- Additional constraint forces uniqueness:
  - $r_y + r_a + r_m = 1$
    - Solution: $r_y = \frac{2}{5}$, $r_a = \frac{2}{5}$, $r_m = \frac{1}{5}$

Flow equations:

$$r_y = r_y/2 + r_a/2$$
$$r_a = r_y/2 + r_m$$
$$r_m = r_a/2$$

Solving the Flow Equations

Something Notable

- Flow equations: 3 equations, 3 unknowns, no constants:
  - No unique solution.
  - All solutions equivalent modulo the scale factor.
- Additional constraint forces uniqueness:
  - $r_y + r_a + r_m = 1$
    - Solution: $r_y = \frac{2}{5}$, $r_a = \frac{2}{5}$, $r_m = \frac{1}{5}$

Flow equations:

$$r_y = r_y/2 + r_a/2$$
$$r_a = r_y/2 + r_m$$
$$r_m = r_a/2$$

Something Notable

- Gaussian elimination method works for small examples, but we need a better method for large web-size graphs

Solving the Flow Equations

Something Notable

- Flow equations: 3 equations, 3 unknowns, no constants:
  - No unique solution.
  - All solutions equivalent modulo the scale factor.
- Additional constraint forces uniqueness:
  - $r_y + r_a + r_m = 1$
    - Solution: $r_y = \frac{2}{5}$, $r_a = \frac{2}{5}$, $r_m = \frac{1}{5}$

Flow equations:

$$r_y = r_y/2 + r_a/2$$
$$r_a = r_y/2 + r_m$$
$$r_m = r_a/2$$

Something Notable

- Gaussian elimination method works for small examples, but we need a better method for large web-size graphs

Therefore

We need a new formulation!

Solving the Flow Equations

Something Notable

- Flow equations: 3 equations, 3 unknowns, no constants:
  - No unique solution.
  - All solutions equivalent modulo the scale factor.
- Additional constraint forces uniqueness:
  - $r_y + r_a + r_m = 1$
    - ★ Solution: $r_y = \frac{2}{5}$, $r_a = \frac{2}{5}$, $r_m = \frac{1}{5}$

Flow equations:

$$r_y = r_y/2 + r_a/2$$
$$r_a = r_y/2 + r_m$$
$$r_m = r_a/2$$

Something Notable

- Gaussian elimination method works for small examples, but we need a better method for large web-size graphs

Therefore

We need a new formulation!

Solving the Flow Equations

Something Notable

- Flow equations: 3 equations, 3 unknowns, no constants:
  - No unique solution.
  - All solutions equivalent modulo the scale factor.
- Additional constraint forces uniqueness:
  - $r_y + r_a + r_m = 1$
    - ★ Solution: $r_y = \frac{2}{5}$, $r_a = \frac{2}{5}$, $r_m = \frac{1}{5}$

Flow equations:

$$r_y = r_y/2 + r_a/2$$
$$r_a = r_y/2 + r_m$$
$$r_m = r_a/2$$

Something Notable

- Gaussian elimination method works for small examples, but we need a better method for large web-size graphs

Therefore

We need a new formulation!

Solving the Flow Equations

Something Notable

- Flow equations: 3 equations, 3 unknowns, no constants:
  - No unique solution.
  - All solutions equivalent modulo the scale factor.
- Additional constraint forces uniqueness:
  - $r_y + r_a + r_m = 1$
    - ★ Solution: $r_y = \frac{2}{5}$, $r_a = \frac{2}{5}$, $r_m = \frac{1}{5}$

Flow equations:

$$r_y = r_y/2 + r_a/2$$
$$r_a = r_y/2 + r_m$$
$$r_m = r_a/2$$

Something Notable

- Gaussian elimination method works for small examples, but we need a better method for large web-size graphs

Therefore

We need a new formulation!

Outline

Page Rank - Google and Company

Page Rank

Invented by Larry Page and Sergei Brin (1998) during his Ph.d studies at Stanford University

They quit the Ph.d program

However, you need to have the knowledge!!!

Not only that they authored the following paper

"The Anatomy of a Large-Scale Hypertextual Web Search Engine"

Page Rank - Google and Company

Page Rank

Invented by Larry Page and Sergei Brin (1998) during his Ph.d studies at Stanford University

They quit the Ph.d program

However, you need to have the knowledge!!!

Not only that they authored the following paper

"The Anatomy of a Large-Scale Hypertextual Web Search Engine"

Page Rank - Google and Company

Page Rank

Invented by Larry Page and Sergei Brin (1998) during his Ph.d studies at Stanford University

They quit the Ph.d program

However, you need to have the knowledge!!!

Not only that they authored the following paper

"The Anatomy of a Large-Scale Hypertextual Web Search Engine"

Page Rank: Matrix Formulation

Stochastic adjacency matrix M

- Let page i has d_i out-links
- If $i \to j$, then $M_{ji} = \frac{1}{d_i}$ else $M_{ji} = 0$
  - M is a column stochastic matrix
    - Columns sum to 1

Page Rank: Matrix Formulation

Stochastic adjacency matrix M

- Let page i has d_i out-links
- If $i \to j$, then $M_{ji} = \frac{1}{d_i}$ else $M_{ji} = 0$
  - M is a column stochastic matrix
    - Columns sum to 1

Rank vector r

- vector with an entry per page
  - r_i is the importance score of page i
  - $\sum_i r_i = 1$

Page Rank: Matrix Formulation

Stochastic adjacency matrix M

- Let page i has d_i out-links
- If $i \to j$, then $M_{ji} = \frac{1}{d_i}$ else $M_{ji} = 0$
  - M is a column stochastic matrix
    - Columns sum to 1

Rank vector r

- vector with an entry per page
  - r_i is the importance score of page i
  - $\sum_i r_i = 1$

The flow equations can be written

$$r_j = \sum_{i \to j} \frac{r_i}{d_i}$$

$$\boldsymbol{r} = M \cdot \boldsymbol{r}$$

Page Rank: Matrix Formulation

Stochastic adjacency matrix M

- Let page i has d_i out-links
- If $i \rightarrow j$, then $M_{ji} = \frac{1}{d_i}$ else $M_{ji} = 0$
  - M is a column stochastic matrix
    - Columns sum to 1

Rank vector r

- vector with an entry per page
  - r_i is the importance score of page i
  - $\sum_i r_i = 1$

The flow equations can be written

$$r_j = \sum_{i \rightarrow j} \frac{r_i}{d_i}$$

$$r = M \cdot r$$

Page Rank: Matrix Formulation

Stochastic adjacency matrix M

- Let page i has d_i out-links
- If $i \to j$, then $M_{ji} = \frac{1}{d_i}$ else $M_{ji} = 0$
  - M is a column stochastic matrix
    - Columns sum to 1

Rank vector r

- vector with an entry per page
  - r_i is the importance score of page i
  - $\sum_i r_i = 1$

The flow equations can be written

$$r_j = \sum_{i \to j} \frac{r_i}{d_i}$$

$$r = M \cdot r$$

Page Rank: Matrix Formulation

Stochastic adjacency matrix M

- Let page i has d_i out-links
- If $i \to j$, then $M_{ji} = \frac{1}{d_i}$ else $M_{ji} = 0$
  - M is a column stochastic matrix
    - ★ Columns sum to 1

Rank vector r

- vector with an entry per page
  - r_i is the importance score of page i
  - $\sum_i r_i = 1$

The flow equations can be written

$$r_j = \sum_{i \to j} \frac{r_i}{d_i}$$

$$r = M \cdot r$$

Page Rank: Matrix Formulation

Stochastic adjacency matrix M

- Let page i has d_i out-links
- If $i \to j$, then $M_{ji} = \frac{1}{d_i}$ else $M_{ji} = 0$
  - M is a column stochastic matrix
    - Columns sum to 1

Rank vector r

- vector with an entry per page
  - r_i is the importance score of page i
  - $\sum_i r_i = 1$

The flow equations can be written

$$r_j = \sum_{i \to j} \frac{r_i}{d_i}$$

$$r = M \cdot r$$

Page Rank: Matrix Formulation

Stochastic adjacency matrix M

- Let page i has d_i out-links
- If $i \to j$, then $M_{ji} = \frac{1}{d_i}$ else $M_{ji} = 0$
  - M is a column stochastic matrix
    - Columns sum to 1

Rank vector r

- vector with an entry per page
  - r_i is the importance score of page i
  - $\sum_i r_i = 1$

The flow equations can be written

$$r_j = \sum_{i \to j} \frac{r_i}{d_i}$$

$$\boldsymbol{r} = M \cdot \boldsymbol{r}$$

Example

- Remember the flow equation

$$r_j = \sum_{i \to j} \frac{r_i}{d_i}$$

- Flow equation in the matrix form

$$\boldsymbol{r} = M \cdot \boldsymbol{r}$$

Example of i links to 3 pages, including j

Example

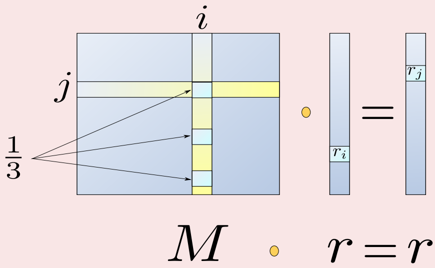
- Remember the flow equation

$$r_j = \sum_{i \to j} \frac{r_i}{d_i}$$

- Flow equation in the matrix form

$$\boldsymbol{r} = M \cdot \boldsymbol{r}$$

Example of i links to 3 pages, including j



Outline

Stochastic Matrices

Markov process

A stochastic process is called a Markov process when it has the Markov property:

$$P\left(X_{t_n}|X_{t_{n-1}}=x_{n-1},...X_{t_1}=x_1\right)=P\left(X_{t_n}|X_{t_{n-1}}=x_{n-1}\right) \quad (2)$$

Simply

The future path of a Markov process, given its current state and the past history before, depends only on the current state (not on how this state has been reached).

Thus (Quite an Oversimplification!!!)

A Markov process is characterized by the (one-step) transition probabilities:

$$p_{i,j}=P\left(X_{t+1}=i|X_t=j\right) \quad (3)$$

Stochastic Matrices

Markov process

A stochastic process is called a Markov process when it has the Markov property:

$$P\left(X_{t_n} | X_{t_{n-1}} = x_{n-1}, \ldots X_{t_1} = x_1\right) = P\left(X_{t_n} | X_{t_{n-1}} = x_{n-1}\right) \tag{2}$$

Simply

The future path of a Markov process, given its current state and the past history before, depends only on the current state (not on how this state has been reached).

Thus (Quite an Oversimplification!!!)

A Markov process is characterized by the (one-step) transition probabilities:

$$p_{i,j} = P\left(X_{t+1} = i | X_t = j\right) \tag{3}$$

Stochastic Matrices

Markov process

A stochastic process is called a Markov process when it has the Markov property:

$$P\left(X_{t_n}|X_{t_{n-1}}=x_{n-1},...X_{t_1}=x_1\right)=P\left(X_{t_n}|X_{t_{n-1}}=x_{n-1}\right) \quad (2)$$

Simply

The future path of a Markov process, given its current state and the past history before, depends only on the current state (not on how this state has been reached).

Thus (Quite an Oversimplification!!!)

A Markov process is characterized by the (one-step) transition probabilities:

$$p_{i,j}=P\left(X_{t+1}=i|X_t=j\right) \quad (3)$$

Markov Chains

Definition (Oversimplified)

A Markov chain is a process X_t indexed by integers $t = 0, 1, \ldots$ such that the states X_t describe the chain at time t.

From here

The probability of a path $i_0, i_1, \ldots, i_n$ is

$$P(X_0 = i_0, X_1 = i_1, \ldots, X_n = i_n) = P(X_0 = i_0)\, p_{i_0 i_1} p_{i_1 i_2} \cdots p_{i_{n-1} i_n} \quad (4)$$

Markov Chains

Definition (Oversimplified)

A Markov chain is a process X_t indexed by integers $t = 0, 1, ...$ such that the states X_t describe the chain at time t.

From here

The probability of a path $i_0, i_1, ..., i_n$ is

$$P(X_0 = i_0, X_1 = i_1, ..., X_n = i_n) = P(X_0 = i_0)\, p_{i_0 i_1} p_{i_1 i_2} \cdots p_{i_{n-1} i_n} \quad (4)$$

The transition probability matrix of a Markov chain

The transition probabilities can be arranged as transition probability matrix $\boldsymbol{P} = (p_{i,j})$

$$
\begin{array}{cc}
 & \text{Final State} \longrightarrow \\
\text{Initial State} \downarrow & \begin{pmatrix} p_{1,1} & p_{1,2} & p_{1,3} & \cdots \\ p_{2,1} & p_{2,2} & p_{2,3} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix} = \boldsymbol{P}
\end{array}
$$

The column j contains the transition probabilities from state j to other states.

Since the system always goes to some state, the sum of the column probabilities is 1:

$$\mathbf{1}^T P = \mathbf{1}^T \tag{5}$$

The transition probability matrix of a Markov chain

The transition probabilities can be arranged as transition probability matrix $\boldsymbol{P} = (p_{i,j})$

$$
\begin{array}{cc}
 & \text{Final State } \longrightarrow \\
\text{Initial State} \downarrow & \begin{pmatrix} p_{1,1} & p_{1,2} & p_{1,3} & \cdots \\ p_{2,1} & p_{2,2} & p_{2,3} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix} = \boldsymbol{P}
\end{array}
$$

The column j contains the transition probabilities from state j to other states.

Since the system always goes to some state, the sum of the column probabilities is 1:

$$\mathbf{1}^T P = \mathbf{1}^T \tag{5}$$

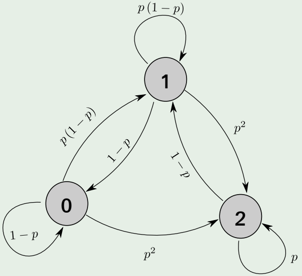
Stochastic Matrix

Definition

- A matrix with non-negative elements such that the sum of each column equals "ONE" is called a stochastic matrix.

Example

We have the following state machine



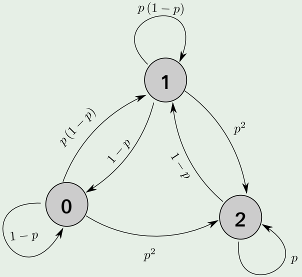
We have the following Stochastic Matrix

$$P = \begin{pmatrix} 1-p & 1-p & 0 \\ p\left(1-p\right) & p\left(1-p\right) & 1-p \\ p^2 & p^2 & p \end{pmatrix} \quad (6)$$

With $p = \frac{1}{3}$

Example

We have the following state machine



We have the following Stochastic Matrix

$$\boldsymbol{P} = \begin{pmatrix} 1-p & 1-p & 0 \\ p\,(1-p) & p\,(1-p) & 1-p \\ p^2 & p^2 & p \end{pmatrix} \quad (6)$$

With $p = \frac{1}{3}$

Therefore, we can use that

To describe the transition in the Markov Chain

Let $\boldsymbol{p}_t \in \mathbb{R}^n$ is the distribution matrix of X_t at time t

$$(\boldsymbol{p}_t)_i = P(X_t = i) \tag{7}$$

Then moving from a distribution to another one we have

$$\boldsymbol{p}_{t+1} = \boldsymbol{P}\boldsymbol{p}_t \tag{8}$$

Therefore, we can use that

To describe the transition in the Markov Chain

Let $\boldsymbol{p}_t \in \mathbb{R}^n$ is the distribution matrix of X_t at time t

$$(\boldsymbol{p}_t)_i = P\left(X_t = i\right) \tag{7}$$

Then moving from a distribution to another one we have

$$\boldsymbol{p}_{t+1} = \boldsymbol{P}\boldsymbol{p}_t \tag{8}$$

Outline

Here, we have Perron-Frobenius

Basic Definition

- A matrix is called positive if all its entries are positive.
- Non-negative, if all its entries are non-negative.

- Matrix Multiplication preserves non-negativity!!!

Here, we have Perron-Frobenius

Basic Definition

- A matrix is called positive if all its entries are positive.
- Non-negative, if all its entries are non-negative.

Basic Facts

If $A \geq 0$ (Element wise) with $A \in \mathbb{R}^{n \times n}$ and $z \geq 0$ with $z \in \mathbb{R}^n$, then $Az \geq 0$

- Matrix Multiplication preserves non-negativity!!!

Here, we have Perron-Frobenius

Basic Definition

- A matrix is called positive if all its entries are positive.
- Non-negative, if all its entries are non-negative.

Basic Facts

If $A \geq 0$ (Element wise) with $A \in \mathbb{R}^{n \times n}$ and $z \geq 0$ with $z \in \mathbb{R}^{n}$, then $Az \geq 0$

- Matrix Multiplication preserves non-negativity!!!

Regularity

Given $A \in \mathbb{R}^{n \times n}$ with $A \geq 0$, then A is called regular if some $k \geq 1$, $A^k > 0$.

Here, we have Perron-Frobenius

Basic Definition

- A matrix is called positive if all its entries are positive.
- Non-negative, if all its entries are non-negative.

Basic Facts

If $A \geq 0$ (Element wise) with $A \in \mathbb{R}^{n \times n}$ and $z \geq 0$ with $z \in \mathbb{R}^{n}$, then $Az \geq 0$

- Matrix Multiplication preserves non-negativity!!!

Regularity

Given $A \in \mathbb{R}^{n \times n}$ with $A \geq 0$, then A is called regular if some $k \geq 1$, $A^k > 0$.

Here, we have Perron-Frobenius

Basic Definition

- A matrix is called positive if all its entries are positive.
- Non-negative, if all its entries are non-negative.

Basic Facts

If $A \geq 0$ (Element wise) with $A \in \mathbb{R}^{n \times n}$ and $z \geq 0$ with $z \in \mathbb{R}^n$, then $Az \geq 0$

- Matrix Multiplication preserves non-negativity!!!

Regularity

Given $A \in \mathbb{R}^{n \times n}$ with $A \geq 0$, then A is called regular if some $k \geq 1$, $A^k > 0$.

Path Property

Meaning of the Previous Definition

From a directed graph on nodes $1, ..., n$ with an arc from i to j whenever $A_{ij} \geq 0$ then

- $\left(A^k\right)_{ij} > 0$ if and only if there is a path of length k from i to j.

Path Property

Meaning of the Previous Definition

From a directed graph on nodes $1, ..., n$ with an arc from i to j whenever $A_{ij} \geq 0$ then

- $\left(A^k\right)_{ij}$ >0 if and only if there is a path of length k from i to j.

Something Notable

A is regular if for some k there is a path of length k from every node to every other node.

Path Property

Meaning of the Previous Definition

From a directed graph on nodes $1, ..., n$ with an arc from i to j whenever $A_{ij} \geq 0$ then

- $\left(A^k\right)_{ij}$ >0 if and only if there is a path of length k from i to j.

Something Notable

A is regular if for some k there is a path of length k from every node to every other node.

Perron-Frobenius Theorem for Regular Matrices

Theorem

Suppose $A \in \mathbb{R}^{n \times n}$ is non-negative and regular, i.e., $A^k > 0$ for some k.

Perron-Frobenius Theorem for Regular Matrices

Theorem

Suppose $A \in \mathbb{R}^{n \times n}$ is non-negative and regular, i.e., $A^k > 0$ for some k.

Then

1. There is an eigenvalue λ_{pf} of A that is real and positive, with positive left and right eigenvectors.
2. For any other eigenvalue λ, we have $|\lambda| < \lambda_{pf}$.
3. The eigenvalue λ_{pf} is simple, i.e., has multiplicity one, and corresponds to a 1×1 Jordan block.

Perron-Frobenius Theorem for Regular Matrices

Theorem

Suppose $A \in \mathbb{R}^{n \times n}$ is non-negative and regular, i.e., $A^k > 0$ for some k.

Then

1. There is an eigenvalue λ_{pf} of A that is real and positive, with positive left and right eigenvectors.
2. For any other eigenvalue λ, we have $|\lambda| < \lambda_{pf}$.
3. The eigenvalue λ_{pf} is simple, i.e., has multiplicity one, and corresponds to a 1×1 Jordan block.

Perron-Frobenius Theorem for Regular Matrices

Theorem

Suppose $A \in \mathbb{R}^{n \times n}$ is non-negative and regular, i.e., $A^k > 0$ for some k.

Then

1. There is an eigenvalue λ_{pf} of A that is real and positive, with positive left and right eigenvectors.
2. For any other eigenvalue λ, we have $|\lambda| < \lambda_{pf}$.
3. The eigenvalue λ_{pf} is simple, i.e., has multiplicity one, and corresponds to a 1×1 Jordan block.

Now, given our matrix $\boldsymbol{P}$

Given $\boldsymbol{P}$ a stochastic matrix

Let π a Perron-Frobenius right eigenvector of P with $\pi \geq 0$ and $1^T \pi = 1$.

Such that $P\pi = \pi$

Then π corresponds to an invariant distribution or equilibrium distribution of the Markov chain for the eigenvalue 1.

Assume that

That $\boldsymbol{P}$ is regular then i.e. that for some k $P^k > 0$.

Now, given our matrix P

Given P a stochastic matrix

Let π a Perron-Frobenius right eigenvector of P with $\pi \geq 0$ and $1^T \pi = 1$.

Such that $P\pi = \pi$

Then π corresponds to an invariant distribution or equilibrium distribution of the Markov chain for the eigenvalue 1.

Assume that

That P is regular then i.e. that for some k $P^k > 0$.

Now, given our matrix $\boldsymbol{P}$

Given $\boldsymbol{P}$ a stochastic matrix

Let π a Perron-Frobenius right eigenvector of P with $\pi \geq 0$ and $1^T \pi = 1$.

Such that $P\pi = \pi$

Then π corresponds to an invariant distribution or equilibrium distribution of the Markov chain for the eigenvalue 1.

Assume that

That $\boldsymbol{P}$ is regular then i.e. that for some k $\boldsymbol{P}^k > 0$.

Thus

If we can force P to be regular

There is unique distribution π such that $\pi > 0$.

Something Notable

- The eigenvalue 1 is simple and dominant.
- Thus, we have $p_t \to \pi$ no matter what the initial distribution p_0

Thus

If we can force P to be regular

There is unique distribution π such that $\pi > 0$.

Something Notable

- The eigenvalue 1 is simple and dominant.
- Thus, we have $\boldsymbol{p}_t \to \pi$ no matter what the initial distribution p_0

Thus, a Simple Algorithms

We have a simple method

Repeatedly apply $\boldsymbol{p}_{t+1} = \boldsymbol{P}\boldsymbol{p}_t$ until convergence to π.

This is a method called

The Power Method

Naive and expensive but

Stable!!!

Thus, a Simple Algorithms

We have a simple method

Repeatedly apply $\boldsymbol{p}_{t+1} = \boldsymbol{P}\boldsymbol{p}_t$ until convergence to π.

This is a method called

The Power Method

Naive and expensive but

Stable!!!

Thus, a Simple Algorithms

We have a simple method

Repeatedly apply $\boldsymbol{p}_{t+1} = \boldsymbol{P}\boldsymbol{p}_t$ until convergence to π.

This is a method called

The Power Method

Naive and expensive but

Stable!!!

Outline

Eigenvector Formulation

The flow equations is written

$$\boldsymbol{r} = M \cdot \boldsymbol{r}$$

Eigenvectors

So the rank vector r is an eigenvector of the stochastic web matrix M

Eigenvector Formulation

The flow equations is written

$$\boldsymbol{r} = M \cdot \boldsymbol{r}$$

Eigenvectors

So the rank vector r is an eigenvector of the stochastic web matrix M

- In fact, its first or principal eigenvector, with corresponding eigenvalue 1!!!

Eigenvector Formulation

Largest eigenvalue of M is 1 since M is column stochastic

- We know r is unit length and each column of M sums to one, so $Mr \leq \mathbf{1}$

- Using the Power Method!!!

Eigenvector Formulation

Largest eigenvalue of M is 1 since M is column stochastic

- We know r is unit length and each column of M sums to one, so $Mr \leq \mathbf{1}$

Thus

- We can obtain r!!!

Eigenvector Formulation

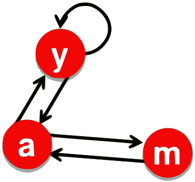
Largest eigenvalue of M is 1 since M is column stochastic

- We know r is unit length and each column of M sums to one, so $Mr \leq \mathbf{1}$

Thus

- We can obtain r!!!
- Using the Power Method!!!

Now going back to the Flow Equation & M



	y	a	m
y	½	½	0
a	½	0	1
m	0	½	0

$$r = M \cdot r$$

$$r_y = r_y/2 + r_a/2$$
$$r_a = r_y/2 + r_m$$
$$r_m = r_a/2$$

$$\begin{bmatrix} y \\ a \\ m \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & 1 \\ 0 & \frac{1}{2} & 0 \end{bmatrix} \begin{bmatrix} y \\ a \\ m \end{bmatrix}$$

Outline

Power Iteration Method

We have that

- Given a web graph with n nodes, where the nodes are pages and edges are hyperlinks.

Power Iteration Method

We have that

- Given a web graph with n nodes, where the nodes are pages and edges are hyperlinks.

Power iteration: a simple iterative scheme

- Suppose there are N web pages
- Initialize: $r^{(0)} = [1/N, ..., 1/N]^T$
- Iterate: $r^{(t+1)} = M \cdot r^{(t)}$
- Stop when $|r^{(t+1)} - r^{(t)}|_1 < \epsilon$
    - $|x|_1 = \sum_{1 \leq i \leq N} |x_i|$ is the L_1 norm

d_i.... out-degree of node i

Power Iteration Method

We have that

- Given a web graph with n nodes, where the nodes are pages and edges are hyperlinks.

Power iteration: a simple iterative scheme

- Suppose there are N web pages
- Initialize: $r^{(0)} = [1/N, ..., 1/N]^T$
- Iterate: $r^{(t+1)} = M \cdot r^{(t)}$
- Stop when $|r^{(t+1)} - r^{(t)}|_1 < \epsilon$
  - $|x|_1 = \sum_{1 \leq i \leq N} |x_i|$ is the L_1 norm

$$r_j^{(t+1)} = \sum_{i \to j} \frac{r_i^t}{d_i}$$

d_i.... out-degree of node i

Power Iteration Method

We have that

- Given a web graph with n nodes, where the nodes are pages and edges are hyperlinks.

Power iteration: a simple iterative scheme

- Suppose there are N web pages
- Initialize: $r^{(0)} = [1/N, ..., 1/N]^T$
- Iterate: $r^{(t+1)} = M \cdot r^{(t)}$
- Stop when $|r^{(t+1)} - r^{(t)}|_1 < \epsilon$
  - $|x|_1 = \sum_{1 \leq i \leq N} |x_i|$ is the L_1 norm

$$r_j^{(t+1)} = \sum_{i \to j} \frac{r_i^t}{d_i}$$

d_i.... out-degree of node i

Power Iteration Method

We have that

- Given a web graph with n nodes, where the nodes are pages and edges are hyperlinks.

Power iteration: a simple iterative scheme

- Suppose there are N web pages
- Initialize: $r^{(0)} = [1/N, ..., 1/N]^T$
- Iterate: $r^{(t+1)} = M \cdot r^{(t)}$
- Stop when $|r^{(t+1)} - r^{(t)}|_1 < \epsilon$
  - $|x|_1 = \sum_{1 \leq i \leq N} |x_i|$ is the L_1 norm

$$r_j^{(t+1)} = \sum_{i \to j} \frac{r_i^t}{d_i}$$

d_i.... out-degree of node i

Power Iteration Method

We have that

- Given a web graph with n nodes, where the nodes are pages and edges are hyperlinks.

Power iteration: a simple iterative scheme

- Suppose there are N web pages
- Initialize: $r^{(0)} = [1/N, ..., 1/N]^T$
- Iterate: $r^{(t+1)} = M \cdot r^{(t)}$
- Stop when $|r^{(t+1)} - r^{(t)}|_1 < \epsilon$
  - $|x|_1 = \sum_{1 \leq i \leq N} |x_i|$ is the L_1 norm

$$r_j^{(t+1)} = \sum_{i \to j} \frac{r_i^t}{d_i}$$

d_i.... out-degree of node i

Power Iteration Method

We have that

- Given a web graph with n nodes, where the nodes are pages and edges are hyperlinks.

Power iteration: a simple iterative scheme

- Suppose there are N web pages
- Initialize: $r^{(0)} = [1/N, ..., 1/N]^T$
- Iterate: $r^{(t+1)} = M \cdot r^{(t)}$
- Stop when $|r^{(t+1)} - r^{(t)}|_1 < \epsilon$
  - $|x|_1 = \sum_{1 \leq i \leq N} |x_i|$ is the L_1 norm

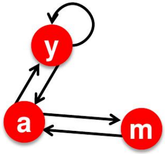
$$r_j^{(t+1)} = \sum_{i \to j} \frac{r_i^t}{d_i}$$

d_i.... out-degree of node i

Page Rank: How to solve?

Power Iteration

- Set $r_j = 1/N$
- 1: $r'_j = \sum_{i \to j} \frac{r_i}{d_i}$
- 2: $r = r'$
- Go to 1



	y	a	m
y	½	½	0
a	½	0	1
m	0	½	0

$$r_y = r_y/2 + r_a/2$$
$$r_a = r_y/2 + r_m$$
$$r_m = r_a/2$$

Example

$$\begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix} = \begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix}$$

Iteration 0, 1, 2, ...

Page Rank: How to solve?

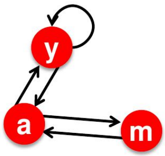
Power Iteration

- Set $r_j = 1/N$
- 1: $r'_j = \sum_{i \to j} \frac{r_i}{d_i}$
- 2: $r = r'$
- Go to 1

Example

$$\begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix} = \begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix}$$

Iteration $0, 1, 2, \ldots$



	y	a	m
y	½	½	0
a	½	0	1
m	0	½	0

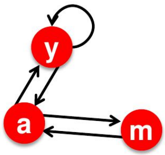
$$r_y = r_y/2 + r_a/2$$
$$r_a = r_y/2 + r_m$$
$$r_m = r_a/2$$

Page Rank: How to solve?

Power Iteration

- Set $r_j = 1/N$
- 1: $r'_j = \sum_{i \to j} \frac{r_i}{d_i}$
- 2: $r = r'$
- Go to 1

Iteration 0, 1, 2, ...



	y	a	m
y	½	½	0
a	½	0	1
m	0	½	0

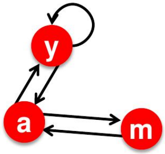
$$r_y = r_y/2 + r_a/2$$
$$r_a = r_y/2 + r_m$$
$$r_m = r_a/2$$

Page Rank: How to solve?

Power Iteration

- Set $r_j = 1/N$
- 1: $r'_j = \sum_{i \to j} \frac{r_i}{d_i}$
- 2: $r = r'$
- Go to 1

Iteration 0, 1, 2, ...



	y	a	m
y	½	½	0
a	½	0	1
m	0	½	0

$$r_y = r_y/2 + r_a/2$$
$$r_a = r_y/2 + r_m$$
$$r_m = r_a/2$$

Example

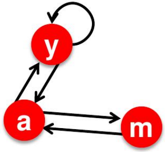
$$\begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix} = \begin{bmatrix} 1/3 & 1/3 & 5/12 & 9/24 & & 6/15 \\ 1/3 & 3/6 & 1/3 & 11/24 & \cdots & 6/15 \\ 1/3 & 1/6 & 3/12 & 1/6 & & 6/15 \end{bmatrix}$$

Page Rank: How to solve?

Power Iteration

- Set $r_j = 1/N$
- 1: $r'_j = \sum_{i \to j} \frac{r_i}{d_i}$
- 2: $r = r'$
- Go to 1

Iteration 0, 1, 2, …



	y	a	m
y	½	½	0
a	½	0	1
m	0	½	0

$$r_y = r_y/2 + r_a/2$$
$$r_a = r_y/2 + r_m$$
$$r_m = r_a/2$$

Example

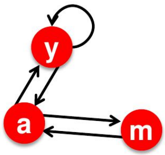
$$\begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix} = \begin{bmatrix} 1/3 & 1/3 & 5/12 & 9/24 & & 6/15 \\ 1/3 & 3/6 & 1/3 & 11/24 & \cdots & 6/15 \\ 1/3 & 1/6 & 3/12 & 1/6 & & 6/15 \end{bmatrix}$$

Page Rank: How to solve?

Power Iteration

- Set $r_j = 1/N$
- 1: $r'_j = \sum_{i \to j} \frac{r_i}{d_i}$
- 2: $r = r'$
- Go to 1

Iteration $0, 1, 2, \ldots$



	y	a	m
y	½	½	0
a	½	0	1
m	0	½	0

$$r_y = r_y/2 + r_a/2$$
$$r_a = r_y/2 + r_m$$
$$r_m = r_a/2$$

Example

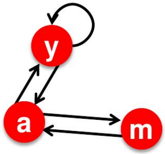
$$\begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix} = \begin{bmatrix} 1/3 & 1/3 & 5/12 & 9/24 & & 6/15 \\ 1/3 & 3/6 & 1/3 & 11/24 & \cdots & 6/15 \\ 1/3 & 1/6 & 3/12 & 1/6 & & 6/15 \end{bmatrix}$$

Page Rank: How to solve?

Power Iteration

- Set $r_j = 1/N$
- 1: $r'_j = \sum_{i \to j} \frac{r_i}{d_i}$
- 2: $r = r'$
- Go to 1

Iteration $0, 1, 2, \ldots$



	y	a	m
y	½	½	0
a	½	0	1
m	0	½	0

$$r_y = r_y/2 + r_a/2$$
$$r_a = r_y/2 + r_m$$
$$r_m = r_a/2$$

Example

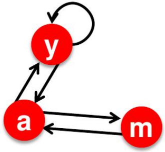
$$\begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix} = \begin{bmatrix} 1/3 & 1/3 & 5/12 & 9/24 & & 6/15 \\ 1/3 & 3/6 & 1/3 & 11/24 & \cdots & 6/15 \\ 1/3 & 1/6 & 3/12 & 1/6 & & 6/15 \end{bmatrix}$$

Page Rank: How to solve?

Power Iteration

- Set $r_j = 1/N$
- 1: $r'_j = \sum_{i \to j} \frac{r_i}{d_i}$
- 2: $r = r'$
- Go to 1

Iteration $0, 1, 2, \ldots$



	y	a	m
y	½	½	0
a	½	0	1
m	0	½	0

$$r_y = r_y/2 + r_a/2$$
$$r_a = r_y/2 + r_m$$
$$r_m = r_a/2$$

Example

$$\begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix} = \begin{bmatrix} 1/3 & 1/3 & 5/12 & 9/24 & & 6/15 \\ 1/3 & 3/6 & 1/3 & 11/24 & \cdots & 6/15 \\ 1/3 & 1/6 & 3/12 & 1/6 & & 6/15 \end{bmatrix}$$

Why Power Iteration works? (1)

Power iteration

- A method for finding dominant eigenvector (the vector corresponding to the largest eigenvalue)
  - $r^{(1)} = M \cdot r^{(0)}$
  - $r^{(2)} = M \cdot r^{(1)} = M(M \cdot r^{(0)}) = M^2 \cdot r^{(0)}$
  - $r^{(3)} = M \cdot r^{(2)} = M(M^2 \cdot r^{(0)}) = M^3 \cdot r^{(0)}$

Why Power Iteration works? (1)

Power iteration

- A method for finding dominant eigenvector (the vector corresponding to the largest eigenvalue)
  - $r^{(1)} = M \cdot r^{(0)}$
  - $r^{(2)} = M \cdot r^{(1)} = M(M \cdot r^{(0)}) = M^2 \cdot r^{(0)}$
  - $r^{(3)} = M \cdot r^{(2)} = M(M^2 \cdot r^{(0)}) = M^3 \cdot r^{(0)}$

Claim

- Sequence $M \cdot r^{(0)}, M^2 \cdot r^{(0)}, \ldots M^k \cdot r^{(0)}, \ldots$ approaches the dominant eigenvector of M

Why Power Iteration works? (1)

Power iteration

- A method for finding dominant eigenvector (the vector corresponding to the largest eigenvalue)
  - $r^{(1)} = M \cdot r^{(0)}$
  - $r^{(2)} = M \cdot r^{(1)} = M(M \cdot r^{(0)}) = M^2 \cdot r^{(0)}$
  - $r^{(3)} = M \cdot r^{(2)} = M(M^2 \cdot r^{(0)}) = M^3 \cdot r^{(0)}$

Claim

- Sequence $M \cdot r^{(0)}, M^2 \cdot r^{(0)}, \ldots M^k \cdot r^{(0)}, \ldots$ approaches the dominant eigenvector of M

Why Power Iteration works? (1)

Power iteration

- A method for finding dominant eigenvector (the vector corresponding to the largest eigenvalue)
  - $r^{(1)} = M \cdot r^{(0)}$
  - $r^{(2)} = M \cdot r^{(1)} = M(M \cdot r^{(0)}) = M^2 \cdot r^{(0)}$
  - $r^{(3)} = M \cdot r^{(2)} = M(M^2 \cdot r^{(0)}) = M^3 \cdot r^{(0)}$

Claim

- Sequence $M \cdot r^{(0)}, M^2 \cdot r^{(0)}, \ldots M^k \cdot r^{(0)}, \ldots$ approaches the dominant eigenvector of M

Why Power Iteration works? (1)

Power iteration

- A method for finding dominant eigenvector (the vector corresponding to the largest eigenvalue)
  - $r^{(1)} = M \cdot r^{(0)}$
  - $r^{(2)} = M \cdot r^{(1)} = M(M \cdot r^{(0)}) = M^2 \cdot r^{(0)}$
  - $r^{(3)} = M \cdot r^{(2)} = M(M^2 \cdot r^{(0)}) = M^3 \cdot r^{(0)}$

Claim

- Sequence $M \cdot r^{(0)}, M^2 \cdot r^{(0)}, ...M^k \cdot r^{(0)}, ...$ approaches the dominant eigenvector of M

Why Power Iteration works? (2)

Claim

- Sequence $M \cdot r^{(0)}, M^2 \cdot r^{(0)}, ... M^k \cdot r^{(0)}, ...$ approaches the dominant eigenvector of M

Why Power Iteration works? (2)

Claim

- Sequence $M \cdot r^{(0)}, M^2 \cdot r^{(0)}, \ldots M^k \cdot r^{(0)}, \ldots$ approaches the dominant eigenvector of M

Proof

- Assume M has n linearly independent eigenvectors, $x_1, x_2, \ldots, x_n$ with corresponding eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_n$, where $\lambda_1 > \lambda_2 > \ldots > \lambda_n$
- Vectors $x_1, x_2, \ldots, x_n$ form a basis and thus we can write: $r^{(0)} = c_1 x_1 + c_2 x_2 + \cdots + c_n x_n$:

$$
\begin{aligned}
M r^{(0)} &= M(c_1 x_1 + c_2 x_2 + \cdots + c_n x_n) \\
&= c_1(M x_1) + c_2(M x_2) + \cdots + c_n(M x_n) \\
&= c_1(\lambda_1 x_1) + c_2(\lambda_2 x_2) + \cdots + c_n(\lambda_n x_n)
\end{aligned}
$$

Why Power Iteration works? (2)

Claim

- Sequence $M \cdot r^{(0)}, M^2 \cdot r^{(0)}, ...M^k \cdot r^{(0)}, ...$ approaches the dominant eigenvector of M

Proof

- Assume M has n linearly independent eigenvectors, $x_1, x_2, ..., x_n$ with corresponding eigenvalues $\lambda_1, \lambda_2, ..., \lambda_n$, where $\lambda_1 > \lambda_2 > ... > \lambda_n$
- Vectors $x_1, x_2, ..., x_n$ form a basis and thus we can write: $r^{(0)} = c_1 x_1 + c_2 x_2 + \cdots + c_n x_n$:

$$
\begin{aligned}
M r^{(0)} &= M(c_1 x_1 + c_2 x_2 + \cdots + c_n x_n) \\
&= c_1(M x_1) + c_2(M x_2) + \cdots + c_n(M x_n) \\
&= c_1(\lambda_1 x_1) + c_2(\lambda_2 x_2) + \cdots + c_n(\lambda_n x_n)
\end{aligned}
$$

cinvestav

Why Power Iteration works? (2)

Claim

- Sequence $M \cdot r^{(0)}, M^2 \cdot r^{(0)}, ...M^k \cdot r^{(0)}, ...$ approaches the dominant eigenvector of M

Proof

- Assume M has n linearly independent eigenvectors, $x_1, x_2, ..., x_n$ with corresponding eigenvalues $\lambda_1, \lambda_2, ..., \lambda_n$, where $\lambda_1 > \lambda_2 > ... > \lambda_n$
- Vectors $x_1, x_2, ..., x_n$ form a basis and thus we can write: $r^{(0)} = c_1 x_1 + c_2 x_2 + \cdots + c_n x_n$:

$$\begin{aligned} M r^{(0)} &= M(c_1 x_1 + c_2 x_2 + \cdots + c_n x_n) \\ &= c_1 (M x_1) + c_2 (M x_2) + \cdots + c_n (M x_n) \\ &= c_1 (\lambda_1 x_1) + c_2 (\lambda_2 x_2) + \cdots + c_n (\lambda_n x_n) \end{aligned}$$

Why Power Iteration works? (3)

Proof (Continued)

- Repeated multiplication on both sides produces

$$M^k r^{(0)} \quad = \quad c_1(\lambda_1^k x_1) + c_2(\lambda_2^k x_2) + \cdots + c_n(\lambda_n^k x_n)$$

-

$$M^k r^{(0)} \quad = \quad \lambda_1^k \left[c_1 x_1 + c_2 \left(\frac{\lambda_2}{\lambda_1} \right)^k x_2 + \cdots + c_n \left(\frac{\lambda_n}{\lambda_1} \right)^k x_n \right]$$

- Since $\lambda_1 > \lambda_2$ then fractions $\frac{\lambda_2}{\lambda_1}, \frac{\lambda_3}{\lambda_1}, \ldots < 1$ and so $\frac{\lambda_i}{\lambda_1} = 0$ as $k \to \infty$ (for all $i = 2 \ldots n$).
- Thus: $M^k r^{(0)} \approx C_1 \lambda_1^k x_1$
  - Note if $c_1 = 0$ then the method will not converge .

Why Power Iteration works? (3)

Proof (Continued)

- Repeated multiplication on both sides produces

$$M^k r^{(0)} = c_1(\lambda_1^k x_1) + c_2(\lambda_2^k x_2) + \cdots + c_n(\lambda_n^k x_n)$$

-

$$M^k r^{(0)} = \lambda_1^k \left[c_1 x_1 + c_2 \left(\frac{\lambda_2}{\lambda_1}\right)^k x_2 + \cdots + c_n \left(\frac{\lambda_n}{\lambda_1}\right)^k x_n \right]$$

- Since $\lambda_1 > \lambda_2$ then fractions $\frac{\lambda_2}{\lambda_1}, \frac{\lambda_3}{\lambda_1}, \ldots < 1$ and so $\frac{\lambda_i}{\lambda_1} = 0$ as $k \to \infty$ (for all $i = 2 \ldots n$).
- Thus: $M^k r^{(0)} \approx C_1 \lambda_1^k x_1$
  - Note if $c_1 = 0$ then the method will not converge .

Why Power Iteration works? (3)

Proof (Continued)

- Repeated multiplication on both sides produces

$$M^k r^{(0)} = c_1(\lambda_1^k x_1) + c_2(\lambda_2^k x_2) + \cdots + c_n(\lambda_n^k x_n)$$

-

$$M^k r^{(0)} = \lambda_1^k \left[c_1 x_1 + c_2 \left(\frac{\lambda_2}{\lambda_1} \right)^k x_2 + \cdots + c_n \left(\frac{\lambda_n}{\lambda_1} \right)^k x_n \right]$$

- Since $\lambda_1 > \lambda_2$ then fractions $\frac{\lambda_2}{\lambda_1}, \frac{\lambda_3}{\lambda_1}, \ldots < 1$ and so $\frac{\lambda_i}{\lambda_1} = 0$ as $k \to \infty$ (for all $i = 2 \ldots n$).
- Thus: $M^k r^{(0)} \approx C_1 \lambda_1^k x_1$
  - Note if $c_1 = 0$ then the method will not converge .

Why Power Iteration works? (3)

Proof (Continued)

- Repeated multiplication on both sides produces

$$M^k r^{(0)} = c_1(\lambda_1^k x_1) + c_2(\lambda_2^k x_2) + \cdots + c_n(\lambda_n^k x_n)$$

- $$M^k r^{(0)} = \lambda_1^k \left[c_1 x_1 + c_2 \left(\frac{\lambda_2}{\lambda_1} \right)^k x_2 + \cdots + c_n \left(\frac{\lambda_n}{\lambda_1} \right)^k x_n \right]$$

- Since $\lambda_1 > \lambda_2$ then fractions $\frac{\lambda_2}{\lambda_1}, \frac{\lambda_3}{\lambda_1}, \ldots < 1$ and so $\frac{\lambda_i}{\lambda_1} = 0$ as $k \to \infty$ (for all $i = 2 \ldots n$).
- Thus: $M^k r^{(0)} \approx C_1 \lambda_1^k x_1$
  - Note if $c_1 = 0$ then the method will not converge .

Why Power Iteration works? (3)

Proof (Continued)

- Repeated multiplication on both sides produces

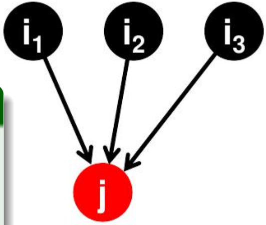
$$M^k r^{(0)} = c_1(\lambda_1^k x_1) + c_2(\lambda_2^k x_2) + \cdots + c_n(\lambda_n^k x_n)$$

-

$$M^k r^{(0)} = \lambda_1^k \left[c_1 x_1 + c_2 \left(\frac{\lambda_2}{\lambda_1}\right)^k x_2 + \cdots + c_n \left(\frac{\lambda_n}{\lambda_1}\right)^k x_n \right]$$

- Since $\lambda_1 > \lambda_2$ then fractions $\frac{\lambda_2}{\lambda_1}, \frac{\lambda_3}{\lambda_1}, \ldots < 1$ and so $\frac{\lambda_i}{\lambda_1} = 0$ as $k \to \infty$ (for all $i = 2 \ldots n$).
- Thus: $M^k r^{(0)} \approx C_1 \lambda_1^k x_1$
  - Note if $c_1 = 0$ then the method will not converge .

Random Walk Interpretation

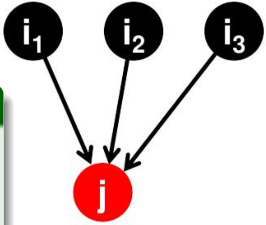


$$r_j = \sum_{i \to j} \frac{r_i}{d_{out}(i)}$$

Imagine a random web surfer

- At any time t, surfer is on some page i
- At time $t+1$, the surfer follows an out-link from i uniformly at random
- Ends up on some page j linked from i
- Process repeats indefinitely

Random Walk Interpretation



$$r_j = \sum_{i \to j} \frac{r_i}{d_{out}(i)}$$

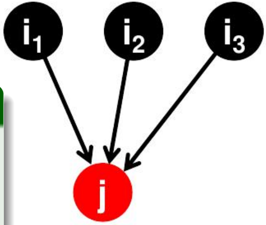
Imagine a random web surfer

- At any time t, surfer is on some page i
- At time $t+1$, the surfer follows an out-link from i uniformly at random
- Ends up on some page j linked from i
- Process repeats indefinitely

Let

- $p(t)$ is the vector whose i^{th} coordinate is the probability that the surfer is at page i at time t.
- So, $p(t)$ is a probability distribution over pages

Random Walk Interpretation



$$r_j = \sum_{i \to j} \frac{r_i}{d_{out}(i)}$$

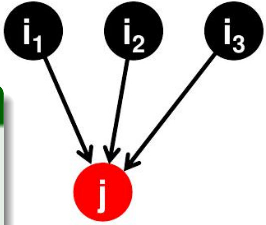
Imagine a random web surfer

- At any time t, surfer is on some page i
- At time $t+1$, the surfer follows an out-link from i uniformly at random
- Ends up on some page j linked from i
- Process repeats indefinitely

Let

- $p(t)$ is the vector whose i^{th} coordinate is the probability that the surfer is at page i at time t.
- So, $p(t)$ is a probability distribution over pages

Random Walk Interpretation



$$r_j = \sum_{i \to j} \frac{r_i}{d_{out}(i)}$$

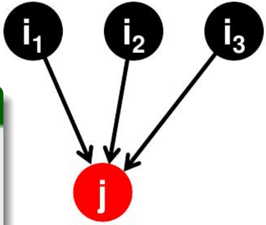
Imagine a random web surfer

- At any time t, surfer is on some page i
- At time $t+1$, the surfer follows an out-link from i uniformly at random
- Ends up on some page j linked from i
- Process repeats indefinitely

Let

- $p(t)$ is the vector whose i^{th} coordinate is the probability that the surfer is at page i at time t.
- So, $p(t)$ is a probability distribution over pages

Random Walk Interpretation



$$r_j = \sum_{i \rightarrow j} \frac{r_i}{d_{out}(i)}$$

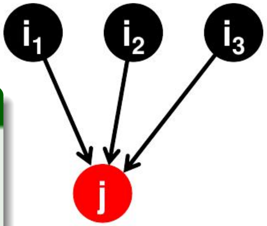
Imagine a random web surfer

- At any time t, surfer is on some page i
- At time $t+1$, the surfer follows an out-link from i uniformly at random
- Ends up on some page j linked from i
- Process repeats indefinitely

Let

- $p(t)$ is the vector whose i^{th} coordinate is the probability that the surfer is at page i at time t.
- So, $p(t)$ is a probability distribution over pages

Random Walk Interpretation



$$r_j = \sum_{i \to j} \frac{r_i}{d_{out}(i)}$$

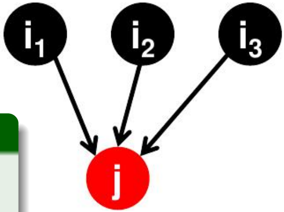
Imagine a random web surfer

- At any time t, surfer is on some page i
- At time $t+1$, the surfer follows an out-link from i uniformly at random
- Ends up on some page j linked from i
- Process repeats indefinitely

Let

- $p(t)$ is the vector whose i^{th} coordinate is the probability that the surfer is at page i at time t.
- So, $p(t)$ is a probability distribution over pages

The Stationary Distribution



$$p(t+1) = M \cdot p(t)$$

Where is the surfer at time $t+1$?

- Follows a link uniformly at random $p(t+1) = M \cdot p(t)$.

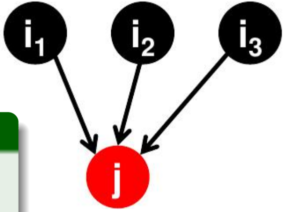
Suppose

- Suppose the random walk reaches a state $p(t+1) = M \cdot p(t) = p(t)$ then $p(t)$ is stationary distribution of a random walk.

Our original rank vector

- Our original rank vector r satisfies $r = M \cdot r$
  - So, r is a stationary distribution for the random walk

The Stationary Distribution



$p(t+1) = M \cdot p(t)$

Where is the surfer at time $t+1$?

- Follows a link uniformly at random
  $p(t+1) = M \cdot p(t)$.

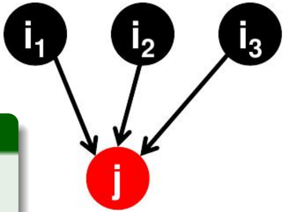
Suppose

- Suppose the random walk reaches a state $p(t+1) = M \cdot p(t) = p(t)$ then $p(t)$ is stationary distribution of a random walk.

Our original rank vector

- Our original rank vector r satisfies $r = M \cdot r$
  - So, r is a stationary distribution for the random walk

The Stationary Distribution



$p(t+1) = M \cdot p(t)$

Where is the surfer at time $t+1$?

- Follows a link uniformly at random
  $p(t+1) = M \cdot p(t)$.

Suppose

- Suppose the random walk reaches a state $p(t+1) = M \cdot p(t) = p(t)$ then $p(t)$ is stationary distribution of a random walk.

Our original rank vector

- Our original rank vector r satisfies $r = M \cdot r$
  - So, r is a stationary distribution for the random walk

Outline

Given the following formulation

Page Rank

$r_j^{(t+1)} = \sum_{i \to j} \frac{r^{(t)}}{d_i}$ or equivalent $r = Mr$

Given the following formulation

Page Rank

$r_j^{(t+1)} = \sum_{i \to j} \frac{r^{(t)}}{d_i}$ or equivalent $r = Mr$

We have the following questions

- Does this converge?
- Does it converge to what we want?
- Are results reasonable?

Given the following formulation

Page Rank

$r_j^{(t+1)} = \sum_{i \to j} \frac{r^{(t)}}{d_i}$ or equivalent $r = Mr$

We have the following questions

- Does this converge?
- Does it converge to what we want?
- Are results reasonable?

Given the following formulation

Page Rank

$r_j^{(t+1)} = \sum_{i \to j} \frac{r^{(t)}}{d_i}$ or equivalent $r = Mr$

We have the following questions

- Does this converge?
- Does it converge to what we want?
- Are results reasonable?

Example

Example: Does this converge?

$$r_j^{(t+1)} = \sum_{i \to j} \frac{r_i^{(t)}}{d_i}$$

Example

$$\begin{array}{c} r_a \\ r_b \end{array} = \begin{array}{cccc} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{array}$$

Iteration 0, 1, 2, ...

Example

Example: Does this converge?

$$r_j^{(t+1)} = \sum_{i \to j} \frac{r_i^{(t)}}{d_i}$$

Example

$$\begin{array}{c} r_a \\ r_b \end{array} = \begin{array}{cccc} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{array}$$

Iteration $0, 1, 2, \ldots$

Example

Example: Does this converge to what we want?

$$r_j^{(t+1)} = \sum_{i \to j} \frac{r_i^{(t)}}{d_i}$$

Example

$$\begin{array}{c} r_a \\ r_b \end{array} = \begin{array}{cccc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{array}$$

Iteration 0, 1, 2, ...

Example

Example: Does this converge to what we want?

$$r_j^{(t+1)} = \sum_{i \to j} \frac{r_i^{(t)}}{d_i}$$

Example

$$\begin{array}{c} r_a \\ r_b \end{array} = \begin{array}{cccc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{array}$$

Iteration $0, 1, 2, \ldots$

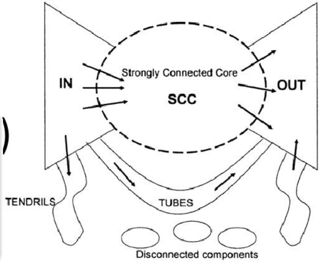
Outline

Page Rank: More Problems

Two problems:

First One

- Some pages are dead ends (have no out-links)
  - Such pages cause importance to "leak out"

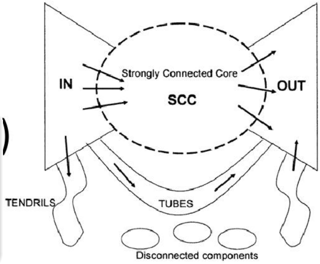


Page Rank: More Problems

Two problems:

First One

- Some pages are dead ends (have no out-links)
  - Such pages cause importance to "leak out"

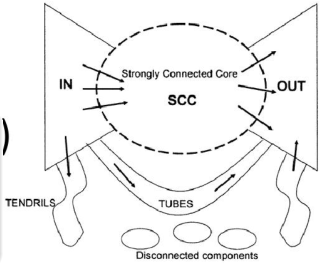


Second One

- Spider traps (all out-links are within the group)
  - Eventually spider traps absorb all importance.

Page Rank: More Problems

Two problems:



First One

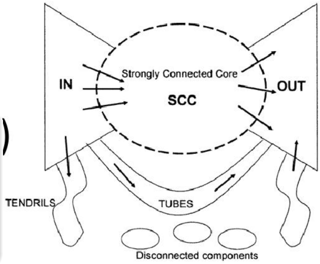
- Some pages are dead ends (have no out-links)
  - Such pages cause importance to "leak out"

Second One

- Spider traps (all out-links are within the group)
  - Eventually spider traps absorb all importance.

Page Rank: More Problems

Two problems:



First One

- Some pages are dead ends (have no out-links)
  - Such pages cause importance to "leak out"

Second One

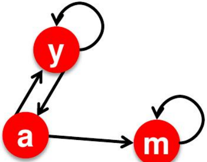
- Spider traps (all out-links are within the group)
  - Eventually spider traps absorb all importance.

.

Problems: Spider Traps

Power Iteration

- Set $r_j = 1$
- $r_j = \sum_{i \to j} \frac{r_i}{d_i}$
  - And iterate



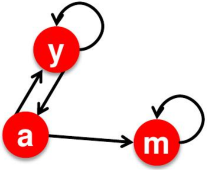
	y	a	m
y	½	½	0
a	½	0	0
m	0	½	1

$$r_y = r_y/2 + r_a/2$$
$$r_a = r_y/2$$
$$r_m = r_a/2 + r_m$$

Example

$$\begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix} = \begin{bmatrix} 1/3 & 2/6 & 3/12 & 5/24 & & 0 \\ 1/3 & 1/6 & 2/12 & 3/24 & \cdots & 0 \\ 1/3 & 3/6 & 7/12 & 16/24 & & 1 \end{bmatrix}$$

Problems: Spider Traps



	y	a	m
y	½	½	0
a	½	0	0
m	0	½	1

$$r_y = r_y/2 + r_a/2$$
$$r_a = r_y/2$$
$$r_m = r_a/2 + r_m$$

Power Iteration

- Set $r_j = 1$
- $r_j = \sum_{i \to j} \frac{r_i}{d_i}$
  - And iterate

Example

$$\begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix} = \begin{bmatrix} 1/3 & 2/6 & 3/12 & 5/24 & & 0 \\ 1/3 & 1/6 & 2/12 & 3/24 & \cdots & 0 \\ 1/3 & 3/6 & 7/12 & 16/24 & & 1 \end{bmatrix}$$

Solution: Random Teleport

The Google solution for spider traps

- At each time step, the random surfer has two options:
  - With prob. β, follow a link at random.
  - With prob. $1 - \beta$, jump to some random page.
  - Common values for β are in the range 0.8 to 0.9

Solution: Random Teleport

The Google solution for spider traps

- At each time step, the random surfer has two options:
  - With prob. β, follow a link at random.
  - With prob. $1-\beta$, jump to some random page.
  - Common values for β are in the range 0.8 to 0.9

Surfer will teleport out of spider trap within a few time steps

Solution: Random Teleport

The Google solution for spider traps

- At each time step, the random surfer has two options:
  - With prob. β, follow a link at random.
  - With prob. $1-\beta$, jump to some random page.
  - Common values for β are in the range 0.8 to 0.9

Surfer will teleport out of spider trap within a few time steps

Solution: Random Teleport

The Google solution for spider traps

- At each time step, the random surfer has two options:
  - With prob. β, follow a link at random.
  - With prob. $1-\beta$, jump to some random page.
  - Common values for β are in the range 0.8 to 0.9

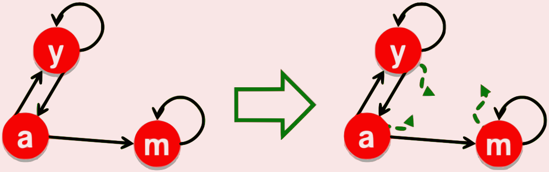
Surfer will teleport out of spider trap within a few time steps

Solution: Random Teleport

The Google solution for spider traps

- At each time step, the random surfer has two options:
  - With prob. β, follow a link at random.
  - With prob. $1-\beta$, jump to some random page.
  - Common values for β are in the range 0.8 to 0.9

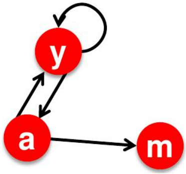
Surfer will teleport out of spider trap within a few time steps



Problem: Dead Ends

Power Iteration on the previous graph

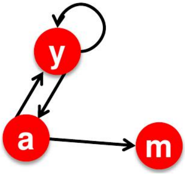
- Set $r_j = 1$
- $r_j = \sum_{i \to j} \frac{r_i}{d_i}$
  - And iterate



	y	a	m
y	½	½	0
a	½	0	0
m	0	½	0

$$r_y = r_y/2 + r_a/2$$
$$r_a = r_y/2$$
$$r_m = r_a/2$$

Problem: Dead Ends



	y	a	m
y	½	½	0
a	½	0	0
m	0	½	0

$$r_y = r_y/2 + r_a/2$$
$$r_a = r_y/2$$
$$r_m = r_a/2$$

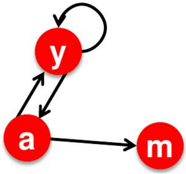
Power Iteration on the previous graph

- Set $r_j = 1$
- $r_j = \sum_{i \to j} \frac{r_i}{d_i}$
  - And iterate

Example

$$\begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix} = \begin{bmatrix} 1/3 & 2/6 & 3/12 & 5/24 & & 0 \\ 1/3 & 1/6 & 2/12 & 3/24 & \cdots & 0 \\ 1/3 & 1/6 & 1/12 & 2/24 & & 0 \end{bmatrix}$$

Problem: Dead Ends



	y	a	m
y	½	½	0
a	½	0	0
m	0	½	0

$r_y = r_y/2 + r_a/2$

$r_a = r_y/2$

$r_m = r_a/2$

Power Iteration on the previous graph

- Set $r_j = 1$
- $r_j = \sum_{i \to j} \frac{r_i}{d_i}$
  - And iterate

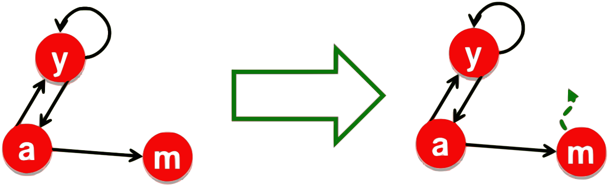
Example

$$\begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix} = \begin{bmatrix} 1/3 & 2/6 & 3/12 & 5/24 & & 0 \\ 1/3 & 1/6 & 2/12 & 3/24 & \cdots & 0 \\ 1/3 & 1/6 & 1/12 & 2/24 & & 0 \end{bmatrix}$$

Solution: Always Teleport

Teleport

- Follow random teleport links with probability 1.0 from dead-ends
  - Adjust matrix accordingly



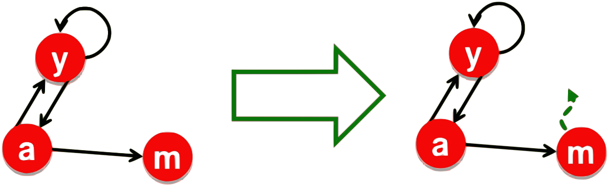
	y	a	m
y	½	½	0
a	½	0	0
m	0	½	0

	y	a	m
y	½	½	⅓
a	½	0	⅓
m	0	½	⅓

Solution: Always Teleport

Teleport

- Follow random teleport links with probability 1.0 from dead-ends
  - Adjust matrix accordingly



	y	a	m
y	½	½	0
a	½	0	0
m	0	½	0

	y	a	m
y	½	½	⅓
a	½	0	⅓
m	0	½	⅓

Why Teleport Solve the Problems?

We know the following

$$r^{(t+1)} = Mr^{(t)}$$

Why Teleport Solve the Problems?

We know the following

$$r^{(t+1)} = Mr^{(t)}$$

Markov chains

- Set of states X
- Transition matrix P where $P_{ij} = P(X_t = i|X_{t-1} = j)$
- π specifying the stationary probability of being at each state $x \in X$
- Goal is to find π such that $\pi = P\pi$

Why Teleport Solve the Problems?

We know the following

$$r^{(t+1)} = Mr^{(t)}$$

Markov chains

- Set of states X
- Transition matrix P where $P_{ij} = P(X_t = i | X_{t-1} = j)$
- π specifying the stationary probability of being at each state $x \in X$
- Goal is to find π such that $\pi = P\pi$

Why Teleport Solve the Problems?

We know the following

$$r^{(t+1)} = Mr^{(t)}$$

Markov chains

- Set of states X
- Transition matrix P where $P_{ij} = P(X_t = i | X_{t-1} = j)$
- π specifying the stationary probability of being at each state $x \in X$
- Goal is to find π such that $\pi = P\pi$

Why Teleport Solve the Problems?

We know the following

$$r^{(t+1)} = Mr^{(t)}$$

Markov chains

- Set of states X
- Transition matrix P where $P_{ij} = P(X_t = i | X_{t-1} = j)$
- π specifying the stationary probability of being at each state $x \in X$
- Goal is to find π such that $\pi = P\pi$

Why is This Analogy Useful?

From

Theory of Markov chains

Why is This Analogy Useful?

From

Theory of Markov chains

We get the following fact

- For any start vector,
  - The power method applied to a Markov transition matrix P will converge to a unique positive stationary vector
  - If P is stochastic, irreducible and aperiodic.

Why is This Analogy Useful?

From

Theory of Markov chains

We get the following fact

- For any start vector,
  - The power method applied to a Markov transition matrix P will converge to a unique positive stationary vector
  - If P is stochastic, irreducible and aperiodic.

Why is This Analogy Useful?

From

Theory of Markov chains

We get the following fact

- For any start vector,
  - The power method applied to a Markov transition matrix P will converge to a unique positive stationary vector
  - If P is stochastic, irreducible and aperiodic.

Outline

Make M Stochastic

Stochastic:

- Every column sums to 1

$$r_y = r_y/2 + r_a/2 + r_m/3$$
$$r_a = r_y/2 + r_m/3$$
$$r_m = r_a/2 + r_m/3$$

Make M Stochastic

Stochastic:

- Every column sums to 1

A possible solution

- Add green links

$$A = M + a\left(\frac{1}{n}e\right)^T$$

- $a_i = \begin{cases} 1 & \text{if node } i \text{ has out deg } 0 \\ 0 & \text{else} \end{cases}$
- e= vector of all 1's

$$r_y = r_y/2 + r_a/2 + r_m/3$$
$$r_a = r_y/2 + r_m/3$$
$$r_m = r_a/2 + r_m/3$$

Make M Stochastic

Stochastic:

- Every column sums to 1

A possible solution

- Add green links

$$A = M + a\left(\frac{1}{n}e\right)^T$$

- $a_i = \begin{cases} 1 & \text{if node } i \text{has out deg } 0 \\ 0 & \text{else} \end{cases}$
- e= vector of all 1's

$$r_y = r_y/2 + r_a/2 + r_m/3$$
$$r_a = r_y/2 + r_m/3$$
$$r_m = r_a/2 + r_m/3$$

Make M Stochastic

Stochastic:

- Every column sums to 1

A possible solution

- Add green links

$$A = M + a\left(\frac{1}{n}e\right)^T$$

- $a_i = \begin{cases} 1 & \text{if node } i \text{has out deg 0} \\ 0 & \text{else} \end{cases}$
- e= vector of all 1's

$$r_y = r_y/2 + r_a/2 + r_m/3$$
$$r_a = r_y/2 + r_m/3$$
$$r_m = r_a/2 + r_m/3$$

Make M Stochastic

Stochastic:

- Every column sums to 1

A possible solution

- Add green links

$$A = M + a\left(\frac{1}{n}e\right)^T$$

- $a_i = \begin{cases} 1 & \text{if node } i \text{has out deg 0} \\ 0 & \text{else} \end{cases}$
- $\mathbf{e}=$ vector of all 1's

$$r_y = r_y/2 + r_a/2 + r_m/3$$
$$r_a = r_y/2 + r_m/3$$
$$r_m = r_a/2 + r_m/3$$

Make M Stochastic

Stochastic:

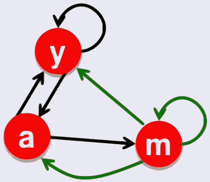
- Every column sums to 1

A possible solution

- Add green links

$$A = M + a\left(\frac{1}{n}e\right)^T$$

- $a_i = \begin{cases} 1 & \text{if node } i \text{has out deg 0} \\ 0 & \text{else} \end{cases}$
- $\mathbf{e}=$ vector of all 1's



	y	a	m
y	½	½	1/3
a	½	0	1/3
m	0	½	1/3

$$r_y = r_y/2 + r_a/2 + r_m/3$$
$$r_a = r_y/2 + r_m/3$$
$$r_m = r_a/2 + r_m/3$$

Make M Aperiodic

Periodic

- A chain is periodic if there is $k > 1$ such that the interval between two visits to some state $\boldsymbol{s}$ is always a multiple of k.

A possible solution

- Add green links

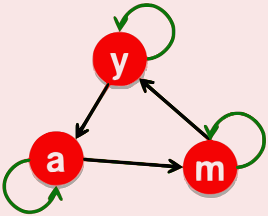
Make M Aperiodic

Periodic

- A chain is periodic if there is $k > 1$ such that the interval between two visits to some state $\boldsymbol{s}$ is always a multiple of k.

A possible solution

- Add green links



Make M Irreducible

Definition

- From any state, there is a non-zero probability of going from any one state to any another

A possible solution for a graph

- Add green links

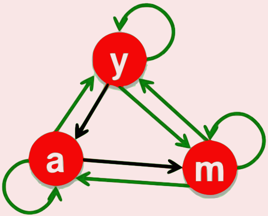
Make M Irreducible

Definition

- From any state, there is a non-zero probability of going from any one state to any another

A possible solution for a graph

- Add green links



Final Solution

Google's solution that does it all

- Makes M stochastic, aperiodic, irreducible.

Final Solution

Google's solution that does it all

- Makes M stochastic, aperiodic, irreducible.

At each step, random surfer has two options

- With probability β, follow a link at random.
- With probability $1-\beta$, jump to some random page.

- This formulation assumes that M has no dead ends.
- We can either preprocess matrix M to remove all dead ends or explicitly follow random teleport links with probability 1.0 from dead-ends.

Final Solution

Google's solution that does it all

- Makes M stochastic, aperiodic, irreducible.

At each step, random surfer has two options

- With probability β, follow a link at random.
- With probability $1 - \beta$, jump to some random page.

Page Rank equation [Brin-Page, 98]

$$r_j = \sum_{i \to j} \beta \frac{r_i}{d_i} + (1 - \beta)\frac{1}{n}$$

- This formulation assumes that M has no dead ends.
- We can either preprocess matrix M to remove all dead ends or explicitly follow random teleport links with probability 1.0 from dead-ends.

Final Solution

Google's solution that does it all

- Makes M stochastic, aperiodic, irreducible.

At each step, random surfer has two options

- With probability β, follow a link at random.
- With probability $1 - \beta$, jump to some random page.

Page Rank equation [Brin-Page, 98]

$$r_j = \sum_{i \to j} \beta \frac{r_i}{d_i} + (1 - \beta) \frac{1}{n}$$

- This formulation assumes that M has no dead ends.
- We can either preprocess matrix M to remove all dead ends or explicitly follow random teleport links with probability 1.0 from dead-ends.

Final Solution

Google's solution that does it all

- Makes M stochastic, aperiodic, irreducible.

At each step, random surfer has two options

- With probability β, follow a link at random.
- With probability $1-\beta$, jump to some random page.

Page Rank equation [Brin-Page, 98]

$$r_j = \sum_{i \to j} \beta \frac{r_i}{d_i} + (1-\beta)\frac{1}{n}$$

- This formulation assumes that M has no dead ends.
- We can either preprocess matrix M to remove all dead ends or explicitly follow random teleport links with probability 1.0 from dead-ends.

Final Solution

Google's solution that does it all

- Makes M stochastic, aperiodic, irreducible.

At each step, random surfer has two options

- With probability β, follow a link at random.
- With probability $1-\beta$, jump to some random page.

Page Rank equation [Brin-Page, 98]

$$r_j = \sum_{i \to j} \beta \frac{r_i}{d_i} + (1-\beta)\frac{1}{n}$$

- This formulation assumes that M has no dead ends.
- We can either preprocess matrix M to remove all dead ends or explicitly follow random teleport links with probability 1.0 from dead-ends.

The Google Matrix

Page Rank equation [Brin-Page, 98]

$$r_j = \sum_{i \to j} \beta \frac{r_i}{d_i} + (1 - \beta) \frac{1}{n}$$

The Google Matrix

Page Rank equation [Brin-Page, 98]

$$r_j = \sum_{i \to j} \beta \frac{r_i}{d_i} + (1-\beta)\frac{1}{n}$$

The Google Matrix A

$$A = \beta M + (1-\beta)\frac{1}{n}\mathbf{e}\cdot\mathbf{e}^{\mathbf{T}}$$

- $\mathbf{e}$... vector of all 1s

Thus

Using the $S = M + a\left(\frac{1}{n}e^T\right)$ to handle nodes with out-degree 0

We can re-write the google matrix

$$A = \beta S + (1-\beta)\frac{1}{n}\mathbf{e}\cdot\mathbf{e^T} \tag{9}$$

Something Notable

The teleporting is random because the teleportation matrix $E = \frac{1}{n}\mathbf{e}\cdot\mathbf{e}^{\mathbf{T}}$ is uniform

Meaning

The surfer is equally likely, when teleporting, to jump to any page.

Thus

Using the $S = M + a\left(\frac{1}{n}e^{T}\right)$ to handle nodes with out-degree 0

We can re-write the google matrix

$$A = \beta S + (1-\beta)\frac{1}{n}\mathbf{e}\cdot\mathbf{e}^{\mathbf{T}} \tag{9}$$

Something Notable

The teleporting is random because the teleportation matrix $E = \frac{1}{n}\mathbf{e}\cdot\mathbf{e}^{\mathbf{T}}$ is uniform

Meaning

The surfer is equally likely, when teleporting, to jump to any page.

Thus

Using the $S = M + a\left(\frac{1}{n}e^T\right)$ to handle nodes with out-degree 0

We can re-write the google matrix

$$A = \beta S + (1-\beta)\frac{1}{n}\mathbf{e}\cdot\mathbf{e^T} \tag{9}$$

Something Notable

The teleporting is random because the teleportation matrix $E = \frac{1}{n}\mathbf{e}\cdot\mathbf{e^T}$ is uniform

Meaning

The surfer is equally likely, when teleporting, to jump to any page.

Thus

There are several consequences of the primitivity adjustment

1. A is stochastic. It is the convex combination of the two stochastic matrices M and E.
2. A is irreducible. Every page is directly connected to every other page, so irreducibility is trivially enforced.
3. A is aperiodic. The self-loops ($A_{ii} > 0$ for all i) create aperiodicity.
4. A is primitive because $A^k > 0$ for some k. Implying that a a unique positive vector π exists, and the power method applied to A is guaranteed to converge to this vector.
5. A is completely dense, which is a very bad thing, computationally.

Thus

There are several consequences of the primitivity adjustment

1. A is stochastic. It is the convex combination of the two stochastic matrices M and E.
2. A is irreducible. Every page is directly connected to every other page, so irreducibility is trivially enforced.
3. A is aperiodic. The self-loops ($A_{ii} > 0$ for all i) create aperiodicity.
4. A is primitive because $A^k > 0$ for some k. Implying that a a unique positive vector π exists, and the power method applied to A is guaranteed to converge to this vector.
5. A is completely dense, which is a very bad thing, computationally.

Thus

There are several consequences of the primitivity adjustment

1. A is stochastic. It is the convex combination of the two stochastic matrices M and E.
2. A is irreducible. Every page is directly connected to every other page, so irreducibility is trivially enforced.
3. A is aperiodic. The self-loops ($A_{ii} > 0$ for all i) create aperiodicity.
4. A is primitive because $A^k > 0$ for some k. Implying that a a unique positive vector π exists, and the power method applied to A is guaranteed to converge to this vector.
5. A is completely dense, which is a very bad thing, computationally.

Thus

There are several consequences of the primitivity adjustment

1. A is stochastic. It is the convex combination of the two stochastic matrices M and E.
2. A is irreducible. Every page is directly connected to every other page, so irreducibility is trivially enforced.
3. A is aperiodic. The self-loops ($A_{ii} > 0$ for all i) create aperiodicity.
4. A is primitive because $A^k > 0$ for some k. Implying that a a unique positive vector π exists, and the power method applied to A is guaranteed to converge to this vector.
5. A is completely dense, which is a very bad thing, computationally.

Thus

There are several consequences of the primitivity adjustment

1. A is stochastic. It is the convex combination of the two stochastic matrices M and E.
2. A is irreducible. Every page is directly connected to every other page, so irreducibility is trivially enforced.
3. A is aperiodic. The self-loops ($A_{ii} > 0$ for all i) create aperiodicity.
4. A is primitive because $A^k > 0$ for some k. Implying that a a unique positive vector π exists, and the power method applied to A is guaranteed to converge to this vector.
5. A is completely dense, which is a very bad thing, computationally.

Given this little adjustment

Thus

- A is stochastic, aperiodic and irreducible, so
$$r^{(t+1)} = A \cdot r^{(t)}$$
- What is β?In practice $\beta = 0.8, 0.9$ (make 5 steps and jump)

Given this little adjustment

Thus

- A is stochastic, aperiodic and irreducible, so

$$r^{(t+1)} = A \cdot r^{(t)}$$

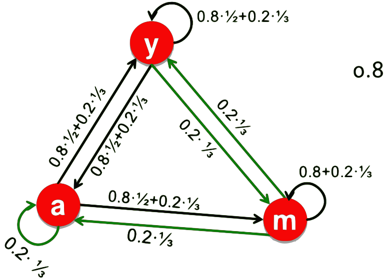
- What is β? In practice $\beta = 0.8, 0.9$ (make 5 steps and jump)

Given this little adjustment

Thus

- A is stochastic, aperiodic and irreducible, so
$$r^{(t+1)} = A \cdot r^{(t)}$$
- What is β?In practice $\beta = 0.8,\ 0.9$ (make 5 steps and jump)

Example: Random Teleport ($\beta = 0.8$)



$$0.8 \underbrace{\begin{array}{|ccc|} \hline 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 \\ 0 & 1/2 & 1 \\ \hline \end{array}}_{\textbf{M}} + 0.2 \underbrace{\begin{array}{|ccc|} \hline 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \\ \hline \end{array}}_{\textbf{1/n}\cdot\textbf{1}\cdot\textbf{1}^{\textbf{T}}}$$

$$\begin{array}{c} y \\ a \\ m \end{array} \begin{array}{|ccc|} \hline 7/15 & 7/15 & 1/15 \\ 7/15 & 1/15 & 1/15 \\ 1/15 & 7/15 & 13/15 \\ \hline \end{array}$$

A

$$\begin{array}{c} y \\ a \\ m \end{array} = \begin{array}{cccccc} 1/3 & 0.33 & 0.24 & 0.26 & & 7/33 \\ 1/3 & 0.20 & 0.20 & 0.18 & \ldots & 5/33 \\ 1/3 & 0.46 & 0.52 & 0.56 & & 21/33 \end{array}$$

Outline

Computing Page Rank

Key step is matrix-vector multiplication

$$r^{new} = A \cdot r^{old}$$

Easy

- Easy if we have enough main memory to hold A, r^{old}, r^{new}

However, if you have $N = 1$ billion pages

- We need 4 bytes for each entry (say)
- 2 billion entries for vectors, approx 8GB
- Matrix A has N^2 entries
  - 10^{18} is a large number!

$$\mathbf{A} = \beta \cdot \mathbf{M} + (1-\beta)\,[1/N]_{N\times N}$$

$$\boldsymbol{A} = 0.8 \begin{bmatrix} ½ & ½ & 0 \\ ½ & 0 & 0 \\ 0 & ½ & 1 \end{bmatrix} + 0.2 \begin{bmatrix} 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \end{bmatrix}$$

$$= \begin{bmatrix} 7/15 & 7/15 & 1/15 \\ 7/15 & 1/15 & 1/15 \\ 1/15 & 7/15 & 13/15 \end{bmatrix}$$

Computing Page Rank

Key step is matrix-vector multiplication

$$r^{new} = A \cdot r^{old}$$

Easy

- Easy if we have enough main memory to hold A, r^{old}, r^{new}

However, if you have $N = 1$ billion pages

- We need 4 bytes for each entry (say)
- 2 billion entries for vectors, approx 8GB
- Matrix A has N^2 entries
  - 10^{18} is a large number!

$$\mathbf{A} = \beta \cdot \mathbf{M} + (1-\beta)\left[1/N\right]_{N\times N}$$

$$\boldsymbol{A} = 0.8 \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{2} & 1 \end{bmatrix} + 0.2 \begin{bmatrix} 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \end{bmatrix}$$

$$= \begin{bmatrix} 7/15 & 7/15 & 1/15 \\ 7/15 & 1/15 & 1/15 \\ 1/15 & 7/15 & 13/15 \end{bmatrix}$$

Computing Page Rank

Key step is matrix-vector multiplication

$$r^{new} = A \cdot r^{old}$$

Easy

- Easy if we have enough main memory to hold A, r^{old}, r^{new}

However, if you have $N = 1$ billion pages

- We need 4 bytes for each entry (say)
- 2 billion entries for vectors, approx 8GB
- Matrix A has N^2 entries
  - 10^{18} is a large number!

$$\mathbf{A} = \beta \cdot \mathbf{M} + (1-\beta)\,[1/N]_{N \times N}$$

$$\mathbf{A} = 0.8 \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 \\ 0 & 1/2 & 1 \end{bmatrix} + 0.2 \begin{bmatrix} 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \end{bmatrix}$$

$$= \begin{bmatrix} 7/15 & 7/15 & 1/15 \\ 7/15 & 1/15 & 1/15 \\ 1/15 & 7/15 & 13/15 \end{bmatrix}$$

Matrix Formulation

Suppose

- Suppose there are N pages.
- Consider page j, with d_j out-links.
- We have $M_{ij} = 1/|d_j|$ when $j \to i$ and $M_{ij} = 0$ otherwise.

Matrix Formulation

Suppose

- Suppose there are N pages.
- Consider page j, with d_j out-links.
- We have $M_{ij} = 1/|d_j|$ when $j \to i$ and $M_{ij} = 0$ otherwise.

The random teleport is equivalent to

- Adding a teleport link from j to every other page and setting transition probability to $(1-\beta)/N$.
- Reducing the probability of following each out-link from $1/|d_j|$ to $\beta/|d_j|$.
- Equivalent: Tax each page a fraction $(1-\beta)$ of its score and redistribute evenly

Matrix Formulation

Suppose

- Suppose there are N pages.
- Consider page j, with d_j out-links.
- We have $M_{ij} = 1/|d_j|$ when $j \to i$ and $M_{ij} = 0$ otherwise.

The random teleport is equivalent to

- Adding a teleport link from j to every other page and setting transition probability to $(1-\beta)/N$.
- Reducing the probability of following each out-link from $1/|d_j|$ to $\beta/|d_j|$.
- Equivalent: Tax each page a fraction $(1-\beta)$ of its score and redistribute evenly

Matrix Formulation

Suppose

- Suppose there are N pages.
- Consider page j, with d_j out-links.
- We have $M_{ij} = 1/|d_j|$ when $j \to i$ and $M_{ij} = 0$ otherwise.

The random teleport is equivalent to

- Adding a teleport link from j to every other page and setting transition probability to $(1-\beta)/N$.
- Reducing the probability of following each out-link from $1/|d_j|$ to $\beta/|d_j|$.
- Equivalent: Tax each page a fraction $(1-\beta)$ of its score and redistribute evenly

Matrix Formulation

Suppose

- Suppose there are N pages.
- Consider page j, with d_j out-links.
- We have $M_{ij} = 1/|d_j|$ when $j \to i$ and $M_{ij} = 0$ otherwise.

The random teleport is equivalent to

- Adding a teleport link from j to every other page and setting transition probability to $(1-\beta)/N$.
- Reducing the probability of following each out-link from $1/|d_j|$ to $\beta/|d_j|$.
- Equivalent: Tax each page a fraction $(1-\beta)$ of its score and redistribute evenly

Matrix Formulation

Suppose

- Suppose there are N pages.
- Consider page j, with d_j out-links.
- We have $M_{ij} = 1/|d_j|$ when $j \to i$ and $M_{ij} = 0$ otherwise.

The random teleport is equivalent to

- Adding a teleport link from j to every other page and setting transition probability to $(1-\beta)/N$.
- Reducing the probability of following each out-link from $1/|d_j|$ to $\beta/|d_j|$.
- Equivalent: Tax each page a fraction $(1-\beta)$ of its score and redistribute evenly

Outline

Rearranging the Equation (1)

$A_{ij} = \beta M_{ij} + \frac{1-\beta}{N}$

$$\begin{aligned}
r &= A \cdot r \\
r_i &= \sum_{j=1}^{N} A_{ij} \cdot r_j \\
r_i &= \sum_{j=1}^{N} \left[\beta M_{ij} + \frac{1-\beta}{N} \right] \cdot r_j \\
&= \sum_{j=1}^{N} \beta M_{ij} \cdot r_j + \frac{1-\beta}{N} \sum_{j=1}^{N} r_j \\
&= \sum_{j=1}^{N} \beta M_{ij} \cdot r_j + \frac{1-\beta}{N} 1
\end{aligned}$$

since $\sum_{j=1}^{N} r_j = 1$

Rearranging the Equation (1)

$A_{ij} = \beta M_{ij} + \frac{1-\beta}{N}$

$$
\begin{aligned}
r &= A \cdot r \\
r_i &= \sum_{j=1}^{N} A_{ij} \cdot r_j \\
r_i &= \sum_{j=1}^{N} \left[\beta M_{ij} + \frac{1-\beta}{N} \right] \cdot r_j \\
&= \sum_{j=1}^{N} \beta M_{ij} \cdot r_j + \frac{1-\beta}{N} \sum_{j=1}^{N} r_j \\
&= \sum_{j=1}^{N} \beta M_{ij} \cdot r_j + \frac{1-\beta}{N} 1
\end{aligned}
$$

since $\sum_{j=1}^{N} r_j = 1$

Rearranging the Equation (1)

$A_{ij} = \beta M_{ij} + \frac{1-\beta}{N}$

$$
\begin{aligned}
r &= A \cdot r \\
r_i &= \sum_{j=1}^{N} A_{ij} \cdot r_j \\
r_i &= \sum_{j=1}^{N} \left[\beta M_{ij} + \frac{1-\beta}{N} \right] \cdot r_j \\
&= \sum_{j=1}^{N} \beta M_{ij} \cdot r_j + \frac{1-\beta}{N} \sum_{j=1}^{N} r_j \\
&= \sum_{j=1}^{N} \beta M_{ij} \cdot r_j + \frac{1-\beta}{N} 1
\end{aligned}
$$

since $\sum_{j=1}^{N} r_j = 1$

Rearranging the Equation (1)

$A_{ij} = \beta M_{ij} + \frac{1-\beta}{N}$

$$
\begin{aligned}
r &= A \cdot r \\
r_i &= \sum_{j=1}^{N} A_{ij} \cdot r_j \\
r_i &= \sum_{j=1}^{N} \left[\beta M_{ij} + \frac{1-\beta}{N} \right] \cdot r_j \\
&= \sum_{j=1}^{N} \beta M_{ij} \cdot r_j + \frac{1-\beta}{N} \sum_{j=1}^{N} r_j \\
&= \sum_{j=1}^{N} \beta M_{ij} \cdot r_j + \frac{1-\beta}{N} 1
\end{aligned}
$$

since $\sum_{j=1}^{N} r_j = 1$

Rearranging the Equation (1)

$A_{ij} = \beta M_{ij} + \frac{1-\beta}{N}$

$$
\begin{aligned}
r &= A \cdot r \\
r_i &= \sum_{j=1}^{N} A_{ij} \cdot r_j \\
r_i &= \sum_{j=1}^{N} \left[\beta M_{ij} + \frac{1-\beta}{N} \right] \cdot r_j \\
&= \sum_{j=1}^{N} \beta M_{ij} \cdot r_j + \frac{1-\beta}{N} \sum_{j=1}^{N} r_j \\
&= \sum_{j=1}^{N} \beta M_{ij} \cdot r_j + \frac{1-\beta}{N} \mathbf{1}
\end{aligned}
$$

since $\sum_{j=1}^{N} r_j = 1$

Rearranging the Equation (1)

$A_{ij} = \beta M_{ij} + \frac{1-\beta}{N}$

$$
\begin{aligned}
r &= A \cdot r \\
r_i &= \sum_{j=1}^{N} A_{ij} \cdot r_j \\
r_i &= \sum_{j=1}^{N} \left[\beta M_{ij} + \frac{1-\beta}{N} \right] \cdot r_j \\
&= \sum_{j=1}^{N} \beta M_{ij} \cdot r_j + \frac{1-\beta}{N} \sum_{j=1}^{N} r_j \\
&= \sum_{j=1}^{N} \beta M_{ij} \cdot r_j + \frac{1-\beta}{N} \mathbf{1}
\end{aligned}
$$

since $\sum_{j=1}^{N} r_j = \mathbf{1}$

Rearranging the Equation (2)

So we get

$$r = \beta M \cdot r + \left[\frac{1-\beta}{N}\right]_N$$

Rearranging the Equation (2)

So we get

$$r = \beta M \cdot r + \left[\frac{1-\beta}{N}\right]_N$$

Note

- Here we assumed M has no dead-ends.
- $[x]_N$ ...a vector of length N with all entries x

Rearranging the Equation (2)

So we get

$$r = \beta M \cdot r + \left[\frac{1-\beta}{N}\right]_N$$

Note

- Here we assumed M has no dead-ends.
- $[x]_N$...a vector of length N with all entries x

Sparse Matrix Formulation

We just rearranged the Page Rank equation

$$r = \beta M \cdot r + \left[\frac{1-\beta}{N}\right]_N$$

- where $[(1-\beta)/N]_N$ is a vector with all N entries $(1-\beta)/N$

Sparse Matrix Formulation

We just rearranged the Page Rank equation

$$r = \beta M \cdot r + \left[\frac{1-\beta}{N}\right]_N$$

- where $[(1-\beta)/N]_N$ is a vector with all N entries $(1-\beta)/N$

M is a sparse matrix! (with no dead-ends)

- 10 links per node, approx $10N$ entries

- Add a constant value $(1-\beta)/N$ to each entry in r^{new}
  - Note if M contains dead-ends then $\sum_j r_j^{new} < 1$ and we also have to re-normalize r^{new} so that it sums to 1

Sparse Matrix Formulation

We just rearranged the Page Rank equation

$$r = \beta M \cdot r + \left[\frac{1-\beta}{N}\right]_N$$

- where $[(1-\beta)/N]_N$ is a vector with all N entries $(1-\beta)/N$

M is a sparse matrix! (with no dead-ends)

- 10 links per node, approx $10N$ entries

So in each iteration, we need to

- Compute $r^{new} = \beta M \cdot r^{old}$
- Add a constant value $(1-\beta)/N$ to each entry in r^{new}
  - Note if M contains dead-ends then $\sum_i r_i^{new} < 1$ and we also have to re-normalize r^{new} so that it sums to 1.

Sparse Matrix Formulation

We just rearranged the Page Rank equation

$$r = \beta M \cdot r + \left[\frac{1-\beta}{N}\right]_N$$

- where $[(1-\beta)/N]_N$ is a vector with all N entries $(1-\beta)/N$

M is a sparse matrix! (with no dead-ends)

- 10 links per node, approx $10N$ entries

So in each iteration, we need to

- Compute $r^{new} = \beta M \cdot r^{old}$
- Add a constant value $(1-\beta)/N$ to each entry in r^{new}
  - Note if M contains dead-ends then $\sum_i r_i^{new} < 1$ and we also have to re-normalize r^{new} so that it sums to 1.

Sparse Matrix Formulation

We just rearranged the Page Rank equation

$$r = \beta M \cdot r + \left[\frac{1-\beta}{N}\right]_N$$

- where $[(1-\beta)/N]_N$ is a vector with all N entries $(1-\beta)/N$

M is a sparse matrix! (with no dead-ends)

- 10 links per node, approx $10N$ entries

So in each iteration, we need to

- Compute $r^{new} = \beta M \cdot r^{old}$
- Add a constant value $(1-\beta)/N$ to each entry in r^{new}
  - Note if M contains dead-ends then $\sum_i r_i^{new} < 1$ and we also have to re-normalize r^{new} so that it sums to 1.

Sparse Matrix Formulation

We just rearranged the Page Rank equation

$$r = \beta M \cdot r + \left[\frac{1-\beta}{N}\right]_N$$

- where $[(1-\beta)/N]_N$ is a vector with all N entries $(1-\beta)/N$

M is a sparse matrix! (with no dead-ends)

- 10 links per node, approx $10N$ entries

So in each iteration, we need to

- Compute $r^{new} = \beta M \cdot r^{old}$
- Add a constant value $(1-\beta)/N$ to each entry in r^{new}
  - Note if M contains dead-ends then $\sum_i r_i^{new} < 1$ and we also have to re-normalize r^{new} so that it sums to 1.

Sparse Matrix Formulation

We just rearranged the Page Rank equation

$$r = \beta M \cdot r + \left[\frac{1-\beta}{N}\right]_N$$

- where $[(1-\beta)/N]_N$ is a vector with all N entries $(1-\beta)/N$

M is a sparse matrix! (with no dead-ends)

- 10 links per node, approx $10N$ entries

So in each iteration, we need to

- Compute $r^{new} = \beta M \cdot r^{old}$
- Add a constant value $(1-\beta)/N$ to each entry in r^{new}
  - Note if M contains dead-ends then $\sum_i r_i^{new} < 1$ and we also have to re-normalize r^{new} so that it sums to 1.

Page Rank: The Complete Algorithm

Input: Graph G and parameter β

- Directed graph G with spider traps and dead ends
- Parameter β

Page Rank: The Complete Algorithm

Input: Graph G and parameter β

- Directed graph G with spider traps and dead ends
- Parameter β

Output: Page Rank vector r

- Set: $r_j^{(0)} = \frac{1}{N}$, $t = 1$
- do:
  - $\forall j$: $r_j'^{(t)} = \sum_{i \to j} \beta \frac{r_i^{t-1}}{d_i}$
    - $r_j'^{(t)} = 0$ if in-deg. of j is 0
  - Now re-insert the leaked Page Rank:
    - $\forall j$: $r_j^{(t)} = r_j'^{(t)} + \frac{1-S}{N}$ where $S = \sum_j r_j'^{(t)}$
  - $t = t + 1$
- while $\sum_j |r_j^{(t)} - r_j^{(t+1)}| > \epsilon$

Page Rank: The Complete Algorithm

Input: Graph G and parameter β

- Directed graph G with spider traps and dead ends
- Parameter β

Output: Page Rank vector r

- Set: $r_j^{(0)} = \frac{1}{N}$, $t = 1$
- do:
  - $\forall j$: $r_j'^{(t)} = \sum_{i \to j} \beta \frac{r_i^{t-1}}{d_i}$
    - $r_j'^{(t)} = 0$ if in-deg. of j is 0
  - Now re-insert the leaked Page Rank:
    - $\forall j$: $r_j^{(t)} = r_j'^{(t)} + \frac{1-S}{N}$ where $S = \sum_j r_j'^{(t)}$
  - $t = t + 1$
- while $\sum_j |r_j^{(t)} - r_j^{(t+1)}| > \epsilon$

Page Rank: The Complete Algorithm

Input: Graph G and parameter β

- Directed graph G with spider traps and dead ends
- Parameter β

Output: Page Rank vector r

- Set: $r_j^{(0)} = \frac{1}{N}$, $t = 1$
- do:
  - $\forall j$: $r_j'^{(t)} = \sum_{i \to j} \beta \frac{r_i^{t-1}}{d_i}$
    - $r_j'^{(t)} = 0$ if in-deg. of j is 0
  - Now re-insert the leaked Page Rank:
    - $\forall j$: $r_j^{(t)} = r_j'^{(t)} + \frac{1-S}{N}$ where $S = \sum_j r_j'^{(t)}$
  - $t = t + 1$
- while $\sum_j \left| r_j^{(t)} - r_j^{(t+1)} \right| > \epsilon$

Page Rank: The Complete Algorithm

Input: Graph G and parameter β

- Directed graph G with spider traps and dead ends
- Parameter β

Output: Page Rank vector r

- Set: $r_j^{(0)} = \frac{1}{N}$, $t = 1$
- do:
  - $\forall j$: $r_j'^{(t)} = \sum_{i \to j} \beta \frac{r_i^{t-1}}{d_i}$
    - $r_j'^{(t)} = 0$ if in-deg. of j is 0
  - Now re-insert the leaked Page Rank:
    - $\forall j$: $r_j^{(t)} = r_j'^{(t)} + \frac{1-S}{N}$ where $S = \sum_j r_j'^{(t)}$
  - $t = t + 1$
- while $\sum_j |r_j^{(t)} - r_j^{(t+1)}| > \epsilon$

Page Rank: The Complete Algorithm

Input: Graph G and parameter β

- Directed graph G with spider traps and dead ends
- Parameter β

Output: Page Rank vector r

- Set: $r_j^{(0)} = \frac{1}{N}$, $t = 1$
- do:
  - $\forall j$: $r_j'^{(t)} = \sum_{i \to j} \beta \frac{r_i^{t-1}}{d_i}$
    - ★ $r_j'^{(t)} = 0$ if in-deg. of j is 0
  - Now re-insert the leaked Page Rank:
    - ★ $\forall j$: $r_j^{(t)} = r_j'^{(t)} + \frac{1-S}{N}$ where $S = \sum_j r_j'^{(t)}$
  - $t = t + 1$
- while $\sum_j |r_j^{(t)} - r_j^{(t+1)}| > \epsilon$

Page Rank: The Complete Algorithm

Input: Graph G and parameter β

- Directed graph G with spider traps and dead ends
- Parameter β

Output: Page Rank vector r

- Set: $r_j^{(0)} = \frac{1}{N}$, $t = 1$
- do:
  - $\forall j$: $r_j'^{(t)} = \sum_{i \to j} \beta \frac{r_i^{t-1}}{d_i}$
    - $r_j'^{(t)} = 0$ if in-deg. of j is 0
  - Now re-insert the leaked Page Rank:
    - $\forall j$: $r_j^{(t)} = r_j'^{(t)} + \frac{1-S}{N}$ where $S = \sum_j r_j'^{(t)}$
  - $t = t + 1$
- while $\sum_j |r_j^{(t)} - r_j^{(t+1)}| > \epsilon$

Page Rank: The Complete Algorithm

Input: Graph G and parameter β

- Directed graph G with spider traps and dead ends
- Parameter β

Output: Page Rank vector r

- Set: $r_j^{(0)} = \frac{1}{N}$, $t = 1$
- do:
  - $\forall j$: $r_j'^{(t)} = \sum_{i \to j} \beta \frac{r_i^{t-1}}{d_i}$
    - $r_j'^{(t)} = 0$ if in-deg. of j is 0
  - Now re-insert the leaked Page Rank:
    - $\forall j$: $r_j^{(t)} = r_j'^{(t)} + \frac{1-S}{N}$ where $S = \sum_j r_j'^{(t)}$
  - $t = t + 1$
- while $\sum_j |r_j^{(t)} - r_j^{(t+1)}| > \epsilon$

Page Rank: The Complete Algorithm

Input: Graph G and parameter β

- Directed graph G with spider traps and dead ends
- Parameter β

Output: Page Rank vector r

- Set: $r_j^{(0)} = \frac{1}{N}$, $t = 1$
- do:
  - $\forall j$: $r_j'^{(t)} = \sum_{i \to j} \beta \frac{r_i^{t-1}}{d_i}$
    - $r_j'^{(t)} = 0$ if in-deg. of j is 0
  - Now re-insert the leaked Page Rank:
    - $\forall j$: $r_j^{(t)} = r_j'^{(t)} + \frac{1-S}{N}$ where $S = \sum_j r_j'^{(t)}$
  - $t = t + 1$
- while $\sum_j |r_j^{(t)} - r_j^{(t+1)}| > \epsilon$

Page Rank: The Complete Algorithm

Input: Graph G and parameter β

- Directed graph G with spider traps and dead ends
- Parameter β

Output: Page Rank vector r

- Set: $r_j^{(0)} = \frac{1}{N}$, $t = 1$
- do:
  - $\forall j$: $r_j'^{(t)} = \sum_{i \to j} \beta \frac{r_i^{t-1}}{d_i}$
    - $r_j'^{(t)} = 0$ if in-deg. of j is 0
  - Now re-insert the leaked Page Rank:
    - $\forall j$: $r_j^{(t)} = r_j'^{(t)} + \frac{1-S}{N}$ where $S = \sum_j r_j'^{(t)}$
  - $t = t + 1$
- while $\sum_j |r_j^{(t)} - r_j^{(t+1)}| > \epsilon$

Outline

Sparse Matrix Encoding

Encode sparse matrix using only non-zero entries

- Space proportional roughly to number of links
- Say $10N$, or $4 * 10 * 1$ billion $= 40$GB
- Still will not fit in memory, but will fit on disk

Source Node	Degree	Destination Node
0	3	1,5,6
1	4	17,64,113,117
2	2	12,23

Sparse Matrix Encoding

Encode sparse matrix using only non-zero entries

- Space proportional roughly to number of links
- Say $10N$, or $4 * 10 * 1$ billion $= 40$GB
- Still will not fit in memory, but will fit on disk

Source Node	Degree	Destination Node
0	3	1,5,6
1	4	17,64,113,117
2	2	12,23

Sparse Matrix Encoding

Encode sparse matrix using only non-zero entries

- Space proportional roughly to number of links
- Say $10N$, or $4 * 10 * 1$ billion $= 40$GB
- Still will not fit in memory, but will fit on disk

Source Node	Degree	Destination Node
0	3	1,5,6
1	4	17,64,113,117
2	2	12,23

Sparse Matrix Encoding

Encode sparse matrix using only non-zero entries

- Space proportional roughly to number of links
- Say $10N$, or $4 * 10 * 1$ billion $= 40$GB
- Still will not fit in memory, but will fit on disk

Source Node	Degree	Destination Node
0	3	1,5,6
1	4	17,64,113,117
2	2	12,23

Basic Algorithm: Update Step

Assume enough RAM to fit r^{new} into memory

- Store r^{old} and matrix M on disk

Basic Algorithm: Update Step

Assume enough RAM to fit r^{new} into memory

- Store r^{old} and matrix M on disk

Then, the first step of power-iteration is

- Initialize all entries of r^{new} to $(1-\beta)/N$
- For each page p (of out-degree n):
  - Read into memory: $p, n, dest_1, \ldots, dest_n, r^{old}(p)$ for $j = 1 \ldots n \Rightarrow r^{new}(dest_j) += \beta r^{old}(p)/n$

r^{new}: 0, 1, 2, 3, 4, 5, 6

Source	Degree	Destination
0	3	1,5,6
1	4	17,64,113,117
2	2	12,23

r^{old}: 0, 1, 2, 3, 4, 5, 6

Basic Algorithm: Update Step

Assume enough RAM to fit r^{new} into memory

- Store r^{old} and matrix M on disk

Then, the first step of power-iteration is

- Initialize all entries of r^{new} to $(1-\beta)/N$
- For each page p (of out-degree n):
  - Read into memory: $p, n, dest_1, ..., dest_n, r^{old}(p)$ for $j = 1...n \Rightarrow r^{new}(dest_j) += \beta r^{old}(p)/n$

r^{new}

0	
1	
2	
3	
4	
5	
6	

Source	Degree	Destination
0	3	1,5,6
1	4	17,64,113,117
2	2	12,23

r^{old}

0	
1	
2	
3	
4	
5	
6	

Basic Algorithm: Update Step

Assume enough RAM to fit r^{new} into memory

- Store r^{old} and matrix M on disk

Then, the first step of power-iteration is

- Initialize all entries of r^{new} to $(1-\beta)/N$
- For each page p (of out-degree n):
  - Read into memory: $p, n, dest_1, ..., dest_n, r^{old}(p)$ for $j = 1...n \Rightarrow r^{new}(dest_j) += \beta r^{old}(p)/n$

$\boldsymbol{r}^{new}$

	$\boldsymbol{r}^{new}$
0	
1	
2	
3	
4	
5	
6	

Source	Degree	Destination
0	3	1,5,6
1	4	17,64,113,117
2	2	12,23

$\boldsymbol{r}^{old}$

	$\boldsymbol{r}^{old}$
0	
1	
2	
3	
4	
5	
6	

Analysis

Assume enough RAM to fit r^{new} into memory

- Store r^{old} and matrix M on disk

Analysis

Assume enough RAM to fit r^{new} into memory

- Store r^{old} and matrix M on disk

In each iteration, we have to

- Read r^{old} and M
- Write r^{new} back to disk
- IO cost $= 2|r| + |M|$

Analysis

Assume enough RAM to fit r^{new} into memory

- Store r^{old} and matrix M on disk

In each iteration, we have to

- Read r^{old} and M
- Write r^{new} back to disk
- IO cost $= 2|r| + |M|$

Question

What if we could not even fit r^{new} in memory?

Analysis

Assume enough RAM to fit r^{new} into memory

- Store r^{old} and matrix M on disk

In each iteration, we have to

- Read r^{old} and M
- Write r^{new} back to disk
- IO cost $= 2|r| + |M|$

Question

What if we could not even fit r^{new} in memory?

Analysis

Assume enough RAM to fit r^{new} into memory

- Store r^{old} and matrix M on disk

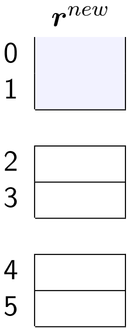
In each iteration, we have to

- Read r^{old} and M
- Write r^{new} back to disk
- IO cost $= 2|r| + |M|$

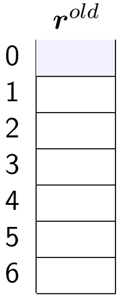
Question

What if we could not even fit r^{new} in memory?

Block-based Update Algorithm



Source	Degree	Destination
0	4	0,1,3,5
1	2	0,5
2	2	3,4



Analysis Block Update

Similar to nested-loop join in databases

- Break r^{new} into k blocks that fit in memory.
- Scan M and r^{old} once for each block.

Analysis Block Update

Similar to nested-loop join in databases

- Break r^{new} into k blocks that fit in memory.
- Scan M and r^{old} once for each block.

k scans of M and r^{old}

$k(|M| + |r|) + |r| = k|M| + (k+1)|r|$.

Analysis Block Update

Similar to nested-loop join in databases

- Break r^{new} into k blocks that fit in memory.
- Scan M and r^{old} once for each block.

k scans of M and r^{old}

$k(|M| + |r|) + |r| = k|M| + (k+1)|r|$.

Can we do better?

- Hint: M is much bigger than r (approx $10-20r$), so we must avoid reading it k times per iteration.

Analysis Block Update

Similar to nested-loop join in databases

- Break r^{new} into k blocks that fit in memory.
- Scan M and r^{old} once for each block.

k scans of M and r^{old}

$k(|M| + |r|) + |r| = k|M| + (k+1)|r|$.

Can we do better?

- Hint: M is much bigger than r (approx $10 - 20x$), so we must avoid reading it k times per iteration

Block-Stripe Update Algorithm

$\mathbf{r}^{new}$

0
1

2
3

4
5

Source	Degree	Destination
0	4	0,1
1	3	0
2	2	1

0	4	3
2	2	3

0	4	5
1	3	5
2	2	4

$\mathbf{r}^{old}$

0
1
2
3
4
5
6

Block-Stripe Analysis

Break M into stripes

- Each stripe contains only destination nodes in the corresponding block of r^{new}

Some additional overhead per stripe

- But it is usually worth it

Cost per iteration

$|M|(1+\epsilon)+(k+1)|r|$

Block-Stripe Analysis

Break M into stripes

- Each stripe contains only destination nodes in the corresponding block of r^{new}

Some additional overhead per stripe

- But it is usually worth it

Cost per iteration

$|M|(1+\epsilon)+(k+1)|r|$

Block-Stripe Analysis

Break M into stripes

- Each stripe contains only destination nodes in the corresponding block of r^{new}

Some additional overhead per stripe

- But it is usually worth it

Cost per iteration

$|M|(1+\epsilon)+(k+1)|r|$

Some Problems with Page Rank

Measures generic popularity of a page

- Biased against topic-specific authorities
  - Solution: Topic-Specific Page Rank (next)

Uses a single measure of importance

- Other models e.g., hubs-and-authorities
  - Solution: Hubs-and-Authorities (next)

Susceptible to Link spam

- Artificial link topographies created in order to boost page rank
  - Solution a more advanced way of page rank: Trust Rank

Some Problems with Page Rank

Measures generic popularity of a page

- Biased against topic-specific authorities
  - Solution: Topic-Specific Page Rank (next)

Uses a single measure of importance

- Other models e.g., hubs-and-authorities
  - Solution: Hubs-and-Authorities (next)

Susceptible to Link spam

- Artificial link topographies created in order to boost page rank
  - Solution a more advanced way of page rank: Trust Rank

cinvestav

Unidad Guadalajara

Some Problems with Page Rank

Measures generic popularity of a page

- Biased against topic-specific authorities
  - Solution: Topic-Specific Page Rank (next)

Uses a single measure of importance

- Other models e.g., hubs-and-authorities
  - Solution: Hubs-and-Authorities (next)

Susceptible to Link spam

- Artificial link topographies created in order to boost page rank
  - Solution a more advanced way of page rank: Trust Rank