Agenda

1. Google PageRank algorithm
2. Developing a formula for ranking web pages
3. Interpretation
4. Computing the score of each page
Google: background

- Mid nineties:
  - many search engines
  - often times not that effective
- Late nineties:
  - Google goes online
    - very effective search engine
- Seems to get what we are looking for
- At the heart of the engine: PageRank
Search engines

Three basic tasks

1. Locate all the web pages with public access
2. Index all the web pages so that they can be searched efficiently (by key words or phrases)
3. Rate the importance of each page;
   
   query $\rightarrow$ returns most important pages first

Many search engines & many ranking algorithms (until Google)
PageRank

- Determined entirely by the link structure of the Web
- Does not involve any of the actual content of webpages or of any individual query
- Given a query, finds the pages on the web that match that query and lists those pages in the order of their PageRank
Importance of PageRank

- Understanding PageRank influences web page design
  
  *how do we get listed first?*

- Had a profound influence on the structure of the Internet
PageRank: basic idea

Internet is a directed graph with nodes and edges
- nodes are pages; \( n \) pages indexed by \( i = 1, 2, \ldots, n \)
- edges are hyperlinks; \( G \) is the \( n \times n \) connectivity matrix

\[
G_{i,j} = \begin{cases} 
1 & \text{if there is a link from page } j \text{ to page } i \\
0 & \text{otherwise}
\end{cases}
\]

Importance score of page \( i \) is \( x_i \)
- \( x_i \) is nonnegative
- \( x_i > x_j \) means that page \( i \) is “more important” than page \( j \)
First ideas...

Why not take as $x_i$ the number of backlinks for page $i$?

First objection: a link to page $i$ should carry much more weight if it comes from an "important page." E.g. a link from CNN or Yahoo! should count more than a link from my webpage.

Modification: let $L_i$, set of webpages with a link to page $i$, be

$x_i = \sum_{j \in L_i} x_j$

Second objection: democracy! We do not want to have a page gaining overwhelming influence by simply linking to many pages.
First ideas...

Why not take as $x_i$ the number of backlinks for page $i$?

First objection: a link to page $i$ should carry much more weight if it comes from an “important page.” E.g. a link from CNN or Yahoo! should count more than a link from my webpage.
First ideas...

Why not take as $x_i$ the number of backlinks for page $i$?

First objection: a link to page $i$ should carry much more weight if it comes from an "important page." E.g. a link from CNN or Yahoo! should count more than a link from my webpage.

Modification: $L_i$, set of webpages with a link to page $i$

$$x_i = \sum_{j \in L_i} x_j$$
First ideas...

Why not take as $x_i$ the number of backlinks for page $i$?

First objection: a link to page $i$ should carry much more weight if it comes from an “important page.” E.g. a link from CNN or Yahoo! should count more than a link from my webpage.

Modification: $L_i$, set of webpages with a link to page $i$

$$x_i = \sum_{j \in L_i} x_j$$

Second objection: democracy! We do not want to have a page gaining overwhelming influence by simply linking to many pages.
Better idea

Define the self-referential scores as

\[ x_i = \sum_{j \in L_i} x_j / n_j, \]

where \( n_j \) is the number of outgoing links from page \( j \). A page has high rank if it has links to and from other pages with high rank.
Define the self-referential scores as

\[ x_i = \sum_{j \in L_i} \frac{x_j}{n_j}, \]

where \( n_j \) is the number of outgoing links from page \( j \). A page has high rank if it has links to and from other pages with high rank.

Finding \( x \) is some sort of eigenvalue problem since

\[ x = Ax \quad A_{i,j} = \frac{G_{i,j}}{n_j} \]

that is, \( x \) is an eigenvector of \( A \) with eigenvalue 1.
Define the self-referential scores as

\[ x_i = \sum_{j \in L_i} \frac{x_j}{n_j}, \]

where \( n_j \) is the number of outgoing links from page \( j \). A page has high rank if it has links to and from other pages with high rank.

Finding \( x \) is some sort of eigenvalue problem since

\[ x = Ax \quad A_{i,j} = G_{i,j}/n_j \]

that is, \( x \) is an eigenvector of \( A \) with eigenvalue 1.

But \( A \) may not have 1 as an eigenvalue...
Interpretation: Markov chain

- Surfing the web, going from page to page by randomly choosing an outgoing link from one page to get to the next
- There can be problems:
  - lead to dead ends at pages with no outgoing links (dangling nodes)
  - cycles around cliques of interconnected pages
- Ignoring this, random walk on the web is a Markov chain
- Matrix $A$ is the transition probability matrix of the chain

\[ A_{ij} \geq 0, \quad \sum_i A_{ij} = 1 \]
Interpretation: Markov chain

- Surfing the web, going from page to page by randomly choosing an outgoing link from one page to get to the next
- There can be problems:
  - lead to dead ends at pages with no outgoing links (dangling nodes)
  - cycles around cliques of interconnected pages
- Ignoring this, random walk on the web is a Markov chain
- Matrix $A$ is the transition probability matrix of the chain

$$A_{ij} \geq 0, \quad \sum_i A_{ij} = 1$$

The score $x_i$ is the limiting probability that the surfer visits any particular page — the fraction of time spent, in the long run, on page $i$

$x$ is the eigenvector of $A$ with eigenvalue 1
Nonunique rankings

What if there are no dangling nodes (so that $A$ is column stochastic) but the web is such that there are two sets of pages which are disconnected from one another?

E.g. Starting from page $i$, and following hyperlinks, there are pages you will never see; i.e. the graph is disconnected

Then the eigenspace with eigenvalue 1 is at least of dimension 2. The score is ill-defined
The last idea

Define the transition probability matrix $Q$

$$Q_{i,j} = (1 - \delta)A_{i,j} + \delta/n, \quad Q = (1 - \delta)A + (\delta/n)\mathbf{1}\mathbf{1}^T$$

In some implementation, Google sets $\delta = .15$

**Interpretation**

- With probability $1 - \delta$, surfer chooses a link at random
- With probability $\delta$, surfer chooses a random page from anywhere on the web (uniformly at random)

- If $\delta = 0$, this is our previous idea
- If $\delta = 1$, then all the webpages have the same score
Perron Frobenius Theorem

Assume no dangling node so that $A$ is stochastic, then $Q$ is stochastic and

$$Q_{ij} = (1 - \delta)A_{ij} + \delta/n > 0$$

- There is a unique (up to scaling) eigenvector with eigenvalue 1. Its components are all positive
- Any other eigenvalue obeys $|\lambda| < 1$

With $\sum_i x_i = 1$, this is the limiting probability distribution and the $x_i$’s are Google’s PageRanks
How to compute the largest eigenvector?

- Big problem: $n$ is above 1 trillion (in 2008, over 1 trillion unique URLs)
How to compute the largest eigenvector?

- Big problem: $n$ is above 1 trillion (in 2008, over 1 trillion unique URLs)
- Only real hope is the power method
How to compute the largest eigenvector?

- Big problem: $n$ is above 1 trillion (in 2008, over 1 trillion unique URLs)
- Only real hope is the power method
- Power method along with modification for speedup (shifts etc.):
  - Pick $x^{(0)}$ and set $i = 0$
  - Repeat
    - $x^{(i+1)} = Qx^{(i)}/\|Qx^{(i)}\|$
  - until convergence

Rate of convergence depends on the eigenvalue gap, expected decrease is proportional to $\|x^{(i)} - x\| \leq O(|\lambda|)\|x^{(0)} - x\|$ where $|\lambda|$ is largest eigenvalue smaller than 1 (in absolute value)

Computed frequently: can use yesterday’s eigenvector as today’s $x^{(0)}$

Requires applying $A$ (sparse) and $1^T$ (cheap) many times. Still, this is an enormous computation (requires many computers, shared memory etc.)
How to compute the largest eigenvector?

- Big problem: $n$ is above 1 trillion (in 2008, over 1 trillion unique URLs)

- Only real hope is the power method

- Power method along with modification for speedup (shifts etc.):
  - Pick $x^{(0)}$ and set $i = 0$
  - Repeat
    - $x^{(i+1)} = Qx^{(i)}/\|Qx^{(i)}\|$
  - until convergence

- Rate of convergence depends on the eigenvalue gap, expected decrease is proportional to
  $$\|x^{(i)} - x\| \leq O(|\lambda|^i) \|x^{(0)} - x\|$$
  where $|\lambda|$ is largest eigenvalue smaller than 1 (in absolute value)
How to compute the largest eigenvector?

- Big problem: \( n \) is above 1 trillion (in 2008, over 1 trillion unique URLs)

- Only real hope is the power method

- Power method along with modification for speedup (shifts etc.):
  - Pick \( x^{(0)} \) and set \( i = 0 \)
  - Repeat
    - \( x^{(i+1)} = Qx^{(i)}/\|Qx^{(i)}\| \)
  - until convergence

- Rate of convergence depends on the eigenvalue gap, expected decrease is proportional to
  \[
  \|x^{(i)} - x\| \leq O(|\lambda|^i) \|x^{(0)} - x\|
  \]
  where \( |\lambda| \) is largest eigenvalue smaller than 1 (in absolute value)

- Computed frequently: can use yesterday’s eigenvector as today’s \( x^{(0)} \)
How to compute the largest eigenvector?

- Big problem: $n$ is above 1 trillion (in 2008, over 1 trillion unique URLs)
- Only real hope is the power method
- Power method along with modification for speedup (shifts etc.):
  - Pick $x^{(0)}$ and set $i = 0$
  - Repeat
    - $x^{(i+1)} = Qx^{(i)}/\|Qx^{(i)}\|$  
    - until convergence
  - Rate of convergence depends on the eigenvalue gap, expected decrease is proportional to 
    $\|x^{(i)} - x\| \leq O(|\lambda|^i) \|x^{(0)} - x\|$ 
    where $|\lambda|$ is largest eigenvalue smaller than 1 (in absolute value)
- Computed frequently: can use yesterday’s eigenvector as today’s $x^{(0)}$
- Requires applying $A$ (sparse) and $11^T$ (cheap) many times. Still, this is an enormous computation (requires many computers, shared memory etc.)
References
