Motivation

The World-Wide Web was invented by Tim Berners-Lee circa 1991.

By the late 1990s, the amount of content on the Web had exploded, but searching this content was becoming an increasingly pressing problem.

Human-generated indices (e.g. Yahoo!) could not keep up, and automatically generated indices (e.g. AltaVista) were sometimes of low quality.

The PageRank algorithm essentially solved the web search problem and was the basis for Google's success.

When a search query is received, satisfying that query can be thought of as a three-stage process:

1. Determine which pages can be returned (e.g. which ones contain the query string);
2. Rank these pages in order of importance;
3. Format the ranked list and display it to the user.

PageRank solves the second stage above. The others are also interesting algorithmic challenges (and you will hear more about the first later in this course).
“Importance”

▶ What does this even mean?
▶ Intuitively, we might think that a web page is important if many other pages link to it; each link provides a "recommendation" that the page is worth visiting.
▶ However, not all links are created equal...
▶ A link from an important web page is more significant than a link from an unimportant web page, so should be “worth” more, e.g.

Nobel Prize official website vs. Ashley’s list of people who should win the Nobel prize

Also, being linked from a page which has many outgoing links is less significant than being linked from a page with only a few outgoing links, e.g.

The comprehensive list of cheeses vs. Winners of the 2013 cheese competition

Brie Cheddar ... Wensleydale Farleigh Wallop Stinking Bishop

Intuitively, the importance of a page is “diluted” by having too many outgoing links.

Simplified PageRank

The following quantity encapsulates these two ideas:

Definition (Simplified PageRank)
The simplified PageRank of a vertex \( v \) is the real number \( R(v) \) satisfying the equation

\[
R(v) = \sum_{u \in B(v)} \frac{R(u)}{\deg(u)}.
\]

In the above definition:
▶ \( B(v) \) is the set of backlinks from \( v \), i.e. the set of vertices \( u \) such that there is an edge \( u \rightarrow v \).
▶ \( \deg(u) \) is the degree of \( u \), i.e. the number of outgoing edges.

(Does such a function \( R \) actually exist, and is it unique? See later...)

The random surfer model

We now discuss an alternative approach one could follow for ranking pages on the Web, which is known as the random surfer model. This will turn out to be basically equivalent.

▶ Imagine a user browsing the Web. The user starts at an arbitrary page, and follows links at random forever, or until he gets stuck (i.e. comes to a page with no outgoing links).
▶ This can be modelled as a random walk on the web graph. At each step, the user picks an outgoing edge at random. For vertices \( u \) with no outgoing edges, we can add a single edge \( u \rightarrow u \), so the user stays at the current page (or just ignore all such vertices).
▶ We then define the (variant) simplified PageRank \( R'(v) \) as the probability of being at vertex \( v \) after \( k \) steps, starting at an arbitrary vertex, as \( k \rightarrow \infty \).

(Is this well-defined? See later...)

The comprehensive list of cheeses vs. Winners of the 2013 cheese competition

Brie Cheddar ... Wensleydale Farleigh Wallop Stinking Bishop

Intuitively, the importance of a page is “diluted” by having too many outgoing links.
Connecting the two notions of simplified PageRank

These two ideas for how to define PageRank are essentially equivalent, which we can show using ideas from linear algebra.

- Let $A$ be the matrix whose rows and columns are indexed by vertices, and
  
  \[ A_{uv} = \begin{cases} \frac{1}{\deg(v)} & \text{if there is an edge } u \to v \\ 0 & \text{otherwise.} \end{cases} \]

- Assume that the user starts at vertex $w$. If the probability that the user is at vertex $u$ after $k$ steps is $p_u^{(k)}$, then the probability that the user is at vertex $v$ after $k + 1$ steps is precisely
  
  \[ \sum_u \Pr[\text{at vertex } u \text{ after } k \text{ steps}] \times \Pr[\text{move from } u \text{ to } v] = \sum_u p_u^{(k)} A_{uv}. \]

- In vector form, we can write
  
  \[ p^{(k+1)} = p^{(k)} A. \]

Connecting the two notions of simplified PageRank

These two ideas for how to define PageRank are essentially equivalent, which we can show using ideas from linear algebra.

- Let $A$ be the matrix whose rows and columns are indexed by vertices, and
  
  \[ A_{uv} = \begin{cases} \frac{1}{\deg(v)} & \text{if there is an edge } u \to v \\ 0 & \text{otherwise.} \end{cases} \]

- Assume that the user starts at vertex $w$. If the probability that the user is at vertex $u$ after $k$ steps is $p_u^{(k)}$, then the probability that the user is at vertex $v$ after $k + 1$ steps is precisely
  
  \[ \sum_u \Pr[\text{at vertex } u \text{ after } k \text{ steps}] \times \Pr[\text{move from } u \text{ to } v] = \sum_u p_u^{(k)} A_{uv}. \]

- In vector form, we can write
  
  \[ p^{(k+1)} = p^{(k)} A. \]

Connecting the two notions of simplified PageRank

For any vertex $v$, let $e_v$ denote the vector whose components are indexed by vertices, and which is 1 at position $v$, and 0 elsewhere. Then $p^{(0)} = e_v$, so

\[ p^{(k)} = e_v A^k. \]

For example:

\[ A = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ \frac{1}{4} & \frac{1}{4} & 0 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \]

\[ p^{(0)} = (1, 0, 0, 0) \quad p^{(1)} = (0, 1/2, 0, 1/2) \quad p^{(2)} = (0, 0, 1/2, 1/2) \quad p^{(3)} = (1/6, 1/6, 0, 2/3) \quad p^{(4)} = (0, 1/12, 1/6, 3/4) \]

Connecting the two notions of simplified PageRank

By our second definition of simplified PageRank $R'$, if there exists a vector $\pi$ such that $\lim_{k \to \infty} e_v A^k = \pi$ for any starting vertex $v$, then $R'(v) = \pi_v$.

- If such a probability distribution $\pi$ exists, we must have $\pi A = \pi$, so $\pi$ is an eigenvector of the matrix $A$ with eigenvalue 1. $\pi$ is called a stationary distribution of the random walk.

- If such a distribution $\pi$ exists, $R'$ satisfies the first definition of simplified PageRank too:
  
  \[ \pi A = \pi \iff \sum_{u \in B(v)} \frac{R'(u)}{\deg(u)} = R'(v) \text{ for all } v. \]

- For the graph on the previous slide, $\pi = (0, 0, 0, 1)$ works.

Problems with the simplified PageRank

There are two problems with this simplified model of PageRank. First, such a stationary distribution $\pi$ might not exist (or be unique).

- Second, there is a conceptual problem highlighted by the random surfer model: if there is a page which links into some subset of webpages which don’t connect to anything else, the surfer will never leave this subset of webpages.

\[ \begin{array}{c} A \\ B \\ C & D \end{array} \]

- For example, taking the above graph the simplified PageRank of $A$ will be 0.

- We can fix this by including some probability $p$ for the surfer to get bored and go to a random web page.
**PageRank**

**Definition (PageRank)**

The PageRank of a vertex $v$ is the real number $PR(v)$ satisfying the equation

$$PR(v) = \frac{p}{N} + (1 - p) \sum_{u \in B(v)} \frac{PR(u)}{\deg(u)}.$$

Here $N$ is the total number of vertices and $p$ is some constant between 0 and 1 giving the “probability of boredom”. $p \approx 0.15$ is often used.

- This is equivalent to modifying $A$ to be of the form

  $$A'_{uv} = \begin{cases} \frac{p}{N} + \frac{1 - p}{\deg(u)} & \text{if there is an edge } u \rightarrow v \\ \frac{p}{N} & \text{otherwise.} \end{cases}$$

**Example**

Using the same graph as before, and taking $p = 0.15$:

$$A = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 3 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$A' \approx \begin{pmatrix} 0.0375 & 0.463 & 0.0375 & 0.463 \\ 0.0375 & 0.463 & 0.0375 & 0.463 \\ 0.321 & 0.321 & 0.0375 & 0.321 \\ 0.0375 & 0.0375 & 0.0375 & 0.888 \end{pmatrix}$$

- It turns out that $\pi \approx (0.073, 0.104, 0.126, 0.696)$ satisfies $\pi A' = A'$.
- So $PR(1) \approx 0.073$, $PR(2) \approx 0.104$ etc.

**PageRank**

We have seen that $PR$, if it exists, corresponds to an eigenvector of $A'$ with eigenvalue $1$.

**Theorem (19th century)**

Any matrix describing a random walk on a graph has all eigenvalues in the range $[-1, 1]$, and has an eigenvector with eigenvalue $1$. Further, the entries of this eigenvector are non-negative.

So $PR$ exists. To show it is the unique such eigenvector requires a bit more work. In fact, the following stronger result is known.

**Theorem (Haveliwala and Kamvar ’03)**

The second-largest eigenvalue $\lambda_2$ of $A'$ satisfies $|\lambda_2| \leq 1 - p$.

**Computing the PageRank**

- The random surfer model allows us to compute PageRank efficiently.
- As $PR$ is the stationary distribution of the random walk, we simply apply $A'$ repeatedly, until the resulting vector does not change much. We must then have approximated the stationary distribution of $A'$.

**PageRank()**

1. for all $i$, $v^{(0)}_i \leftarrow 1/N$
2. $k \leftarrow 0$
3. repeat forever
4. $v^{(k+1)} \leftarrow v^{(k)} A'$
5. if $\sum |v^{(k+1)}_i - v^{(k)}_i| \leq \epsilon$
6. return $v^{(k+1)}$
7. $k \leftarrow k + 1$

Here $\epsilon$ is an arbitrary small parameter specifying the desired accuracy.
Example
At the start of the algorithm:

\[
A' \approx \begin{pmatrix}
0.0375 & 0.463 & 0.0375 & 0.463 \\
0.0375 & 0.0375 & 0.888 & 0.0375 \\
0.321 & 0.321 & 0.0375 & 0.321 \\
0.0375 & 0.0375 & 0.0375 & 0.888
\end{pmatrix}
\]

▶ Blue labels: the updating PageRanks of the vertices.

Example
After one iteration:

\[
A' \approx \begin{pmatrix}
0.0375 & 0.463 & 0.0375 & 0.463 \\
0.0375 & 0.0375 & 0.888 & 0.0375 \\
0.321 & 0.321 & 0.0375 & 0.321 \\
0.0375 & 0.0375 & 0.0375 & 0.888
\end{pmatrix}
\]

▶ Blue labels: the updating PageRanks of the vertices.

Example
After two iterations:

\[
A' \approx \begin{pmatrix}
0.0375 & 0.463 & 0.0375 & 0.463 \\
0.0375 & 0.0375 & 0.888 & 0.0375 \\
0.321 & 0.321 & 0.0375 & 0.321 \\
0.0375 & 0.0375 & 0.0375 & 0.888
\end{pmatrix}
\]

Example
After three iterations:

\[
A' \approx \begin{pmatrix}
0.0375 & 0.463 & 0.0375 & 0.463 \\
0.0375 & 0.0375 & 0.888 & 0.0375 \\
0.321 & 0.321 & 0.0375 & 0.321 \\
0.0375 & 0.0375 & 0.0375 & 0.888
\end{pmatrix}
\]

▶ Blue labels: the updating PageRanks of the vertices.
Runtime

- The result that $|\lambda_2| \leq 1 - p$, i.e. is a constant strictly less than 1, turns out to imply that it suffices to repeat the above loop $O(\log N)$ times to achieve a value of $\epsilon$ which is a small constant (e.g. 0.001).

- The algorithm is very efficient in practice; in their 1998 paper, Brin and Page report that the PageRank of all pages in a 322 million page database can be approximately computed in 52 iterations.

- The multiplication by $A'$ can also be performed very efficiently, as the “web graph” is sparse. One iteration takes time $O(N + L)$, where $L$ is the number of links on the web. (Multiplication by a general matrix would take time $\Theta(N^2)$.)

Summary

- The PageRank algorithm has enabled web search to keep pace with the hugely increasing quantities of data on the Internet. It also has a number of other applications (e.g. ranking academic research using the graph of citations).

- Developing an index of importance of web pages can be done quite accurately by modelling humans as clicking on links at random, occasionally getting bored and going to a completely random page.

- Google, a $300$ billion company, ultimately stems from an efficient algorithm and some Victorian-era linear algebra.

Does this make sense?

Is PageRank actually a sensible way to rank pages?

- Ultimately, the metric that measures PageRank’s success is its usefulness to its users.

- Nowadays Google uses a number of other methods to rank pages (most of them secret), including AI / machine learning techniques. As well as improving search quality in general, this is helpful to avoid spam and other undesirable pages.

- Companies like Google perform extensive user testing and validation to determine whether their algorithms actually work.

Further reading

- “The PageRank Citation Ranking: Bringing Order to the Web”
  L. Page and S. Brin and R. Motwani and T. Winograd
  http://ilpubs.stanford.edu:8090/422/

- “The Anatomy of a Large-Scale Hypertextual Web Search Engine”
  S. Brin and L. Page
  http://ilpubs.stanford.edu:8090/361/

- A short lecture course on PageRank and other Google technology:
  http://michaelnielsen.org/blog/lecture-course-the-google-technology-stack/
Historical notes

- PageRank was developed by graduate students Sergey Brin and Larry Page, who went on to drop out of their PhDs and found Google.

- Similar ideas had been developed by some other people previously and concurrently (e.g. Robin Li, Jon Kleinberg).

- Although PageRank is a method for ranking pages, it was in fact named after Larry Page.