FAQs
• Duplicated printout for PA1
• PA2 has been posted

Topics
• Large-scale Analytics 1. Web-Scale Link and Social Network Analysis

This material is built based on,
  • Chapter 5
• http://infolab.stanford.edu/~ullman/mmds.html
What are these?

• JumpStation
• Go
• Infoseek
• Snap
• Direct Hit
• Lycos
• AltaVista
• Excite
• Yahoo
• Google

Early Search Engines

• They worked by crawling the Web and listing the terms
• Words or other strings of characters other than white space
• In an inverted index

• An inverted index is a data structure that makes it easy, given a term, to find (pointer to) all the places where that term occurs

Inverted index (1/2)

• Inverted index
  • For given texts,

    S[0] = "Colorado State University"
    S[1] = "Colorado water source"
    S[2] = "University of Colorado"

• We have the following inverted file index

  "Colorado": {0,1,2}
  "of":{2}
  "source":{1}
  "State": {0}
  "University": {0,2}
  "water":{1}

Inverted index (2/2)

• A term search for the terms, "Colorado", "State", and "University" would give the set

  \{0,1,2\} ∩ \{0\} ∩ \{0,2\} = \{0\}

Term spam

• If you were selling toaster on the Web
  • All you care about was that people would see your page
  • You could add a term like “movie” to your page
    • Add thousands of times
    • It does not even need to show
    • Same color as background to the letters
    • A search engine would think this page is very important one about "movie"
  • You could go to the search engine and search “movie” and see the first listed page
    • Copy that page with the same color as background

Part 1. Large Scale Data Analytics

1. Web-Scale Link Analysis and Social Network Analysis

Web-Scale Link Analysis: PageRank Algorithm
Current total number of Web pages: More than 1.4 B indexed pages

PageRank

• Goals
  • Providing effective summaries for the search results
  • Ordering/Ranking results

• Simulate random Web surfers
  • Pages that would have a large number of surfers were considered more "important" than pages that would rarely be visited

• The content of a page was judged not only by the terms appearing on that page
  • But by the terms used in or near the links to that page

Definition of PageRank

• A function that assigns a real number to each page in the Web
• The higher the PageRank of a page, the more "important" it is
• There is NOT one fixed algorithm for assignment of PageRank
Example

• Page A has links to B, C and D
• Page B has links to A and D
• Page C has a link to A
• Page D has links to B and C

Example

• Suppose that a random surfer starts at page A
• Page B, C and D will be the next with probability 1/3
• 0 probability of being at A

Example

• Now suppose the random surfer at B
  • B has probability of ½ of being at A, ½ of being at D and 0 of being at B or C

Example

• Transition matrix $M$
  • What happens to random surfers after one step
  • $M$ has $n$ rows and columns ($n$ pages)
  • What is the transition matrix for this example?

Example

$M = \begin{bmatrix} 0 & 1/2 & 1 & 0 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 0 & 0 & 0 \end{bmatrix}$

What does this matrix mean?

• The probability distribution for the location of a random surfer
  • A column vector whose $j$th component is the probability that the surfer is at page $j$
What does this matrix mean? [2/6]

- If we surf at any of the $n$ pages of the Web with equal probability
  - The initial vector $v_0$ will have $1/n$ for each component
  - After the first step, the distribution of the surfer will be $Mv_0$
  - After two steps, $M(Mv_0) = M^2v_0$ and so on

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1/2</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1/3</td>
<td>0</td>
<td>0</td>
<td>1/2</td>
<td></td>
</tr>
<tr>
<td>1/3</td>
<td>0</td>
<td>0</td>
<td>1/2</td>
<td></td>
</tr>
<tr>
<td>1/3</td>
<td>1/2</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

What does this matrix mean? [3/6]

- Multiplying the initial vector $v_0$ by $M$ a total of $i$ times
  - The distribution of the surfer after $i$ steps
  - The probability for being in the next possible location
  - The probability for the next step from the current location
  - The probability for being in the current location

\[
M = \begin{bmatrix}
0 & 1/2 & 1 & 0 \\
1/3 & 0 & 0 & 1/2 \\
1/3 & 1/2 & 0 & 0 \\
0 & 1/3 & 0 & 0 \\
\end{bmatrix}
\]

What does this matrix mean? [4/6]

- The probability $x_i$ that a random surfer will be at node $i$ at the next step
  - $x_i = \sum_j m_{ij} v_j$
  - $m_{ij}$ is the probability that a surfer at node $j$ will move to node $i$ at the next step
  - $v_j$ is the probability that the surfer was at node $j$ at the previous step

What does this matrix mean? [5/6]

- The distribution of the surfer approaches a limiting distribution $\mathbf{v}$ that satisfies $\mathbf{v} = M\mathbf{v}$ provided two conditions are met:
  1. The graph is strongly connected
     - It is possible to get from any node to any other node
  2. There are no dead ends
     - Nodes that have no arcs out

What does this matrix mean? [6/6]

- The limit is reached when multiplying the distribution by $M$ another time does not change the distribution
  - The limiting $\mathbf{v}$ is an eigenvector of $M$
  - Since $M$ is stochastic (its columns each add up to 1), $\mathbf{v}$ is the principle eigenvector
  - Its associated eigenvalue is the largest of all eigenvalues
- The principle eigenvector of $M$
  - Where the surfer is most likely to be after a long time
- For the Web, 50-75 iterations are sufficient to converge to within the error limits of double-precision arithmetic

Example

\[
M = \begin{bmatrix}
0 & 1/2 & 1 & 0 \\
1/3 & 0 & 0 & 1/2 \\
1/3 & 1/2 & 0 & 0 \\
1/3 & 0 & 0 & 1/2 \\
\end{bmatrix}
\]

- Suppose we apply this process to the matrix $M$
  - The initial vector $v_0$ and $v_1$ after multiplying $M$
Example

\[
M = \begin{bmatrix}
0 & 1/2 & 0 & 1/2 \\
1/3 & 0 & 0 & 1/2 \\
1/3 & 0 & 0 & 1/2 \\
1/3 & 0 & 0 & 0
\end{bmatrix}
\]

• Suppose we apply this process to the matrix \( M \)

• The initial vector \( v_0 \) and \( v_1 \) after multiplying \( M \)

\[
v_0 = \begin{bmatrix} 0 \ 1/2 \ 1 \ 0 \end{bmatrix}, \quad v_1 = \begin{bmatrix} 1/3 \ 0 \ 0 \ 1/2 \end{bmatrix}
\]

What is the \( v_2 \)?

\[
M = \begin{bmatrix}
0 & 1/2 & 0 & 1/2 \\
1/3 & 0 & 0 & 1/2 \\
1/3 & 1/2 & 0 & 0
\end{bmatrix}
\]

• Suppose we apply this process to the matrix \( M \)

• The initial vector \( v_0 \) and \( v_1 \) after multiplying \( M \)

\[
v_0 = \begin{bmatrix} 0 \ 1/2 \ 1 \ 0 \end{bmatrix}, \quad v_1 = \begin{bmatrix} 1/3 \ 0 \ 0 \ 1/2 \end{bmatrix}
\]

Example continued

• The sequence of approximations to the limit

• We get by multiplying at each step by \( M \)

\[
\begin{bmatrix}
0/3 \ 5/24 \ 11/48 \ 7/32 \\
5/24 \ 0/3 \ 11/48 \ 7/32 \\
5/24 \ 11/48 \ 0/3 \ 7/32 \\
7/32 \ 7/32 \ 7/32 \ 0/3
\end{bmatrix}
\]

• This difference in probability is not noticeable

• In the real Web, there are billions of nodes of greatly varying importance

• The probability of being at a node like www.amazon.com is orders of magnitude greater than others

Matrix-vector Multiplication by MapReduce [1/3]

• Suppose we have an \( n \times n \) matrix \( M \), whose element in row \( i \) and column \( j \) will be denoted \( M_{ij} \)

• Then the matrix-vector product is the vector \( x \) of length \( n \), whose \( i^{th} \) element \( x_i \) is given by,

\[
x_i = \sum_j M_{ij} v_j
\]

Matrix-vector Multiplication by MapReduce [2/3]

• If \( n = 100 \), we do NOT need DFS or MapReduce

• However, if this calculation is a part of ranking Web pages (\( n \) is 10M) that goes on at search engine? The vector \( v \) cannot fit in main memory

• More than 1.4B pages
Matrix-vector Multiplication by MapReduce [3/3]

- The matrix $M$ and the vector $v$ each will be stored in a file of the DFS/HDFS.
- Assume that row-column coordinates of each matrix element will be discoverable:
  - Either from its position in the file or explicit coordinates.

The Map function

- The Map function is written to apply to one element of $M$.
- Each Map task will operate on a chunk of the matrix $M$.
- From each matrix element $m_{ij}$, it produces the key-value pair $(i, m_{ij}v_j)$.

- All terms of the sum that make up the component $x_k$ of the matrix-vector product will get the same key, $k$.

The Reduce function

- Sums all the values associated with a given key $k$.
- The result will be a pair $(k, x_k)$.
  
  $$x_k = M_{k,0}x_0 + M_{k,1}x_1 + M_{k,2}x_2 + \ldots + M_{k,(n-1)}x_{n-1}$$

If the vector $v$ cannot fit in main memory?

- It is possible that the vector $v$ is so large that it will not fit in its main memory entirely.
- We can divide the matrix into vertical stripes of equal width and divide the vector into an equal number of horizontal stripes of the same height.
- The goal is to use enough stripes so that the portion of the vector in one stripe can fit into main memory.
Link Analysis
PageRank Algorithm for the real Web

Structure of Web (1/3)

• Is the Web strongly connected?

Structure of Web (2/3)

<table>
<thead>
<tr>
<th>In Component</th>
<th>Out Component</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strongly Connected Component</td>
<td>Consisting of pages reachable from the In Component but not able to reach the SCC</td>
</tr>
<tr>
<td>Tendrils Out</td>
<td>Consisting of pages reachable from the SCC but unable to reach the SCC</td>
</tr>
<tr>
<td>Tendrils In</td>
<td>Consisting of pages that could reach via following links but not reachable from the SCC</td>
</tr>
<tr>
<td>Disconnected Components</td>
<td>Consisting of pages reachable from the In Component but not reachable from the Out Component</td>
</tr>
</tbody>
</table>

Structure of Web (3/3)

• Tubes
  • Pages reachable from the In component and able to reach the Out component, but unable to reach the SCC or be reached from the SCC

• Isolated Components
  • Unreachable from the large components

Anomalies from the Web Structure

• These structures violate the assumptions needed for the Markov process iteration to converge to a limit
  • When a random surfer enters the out component, they can never leave
  • Surfers starting in either the SCC or In component are going to wind up in either the out component or a tendril off the In component
  • No page in the SCC or In component winds up with any probability of a surfer being there

• Nothing in the SCC or In component will be of any importance
Problems we need to avoid

- **Dead end**
  - A page that has no links out
  - Surfers reaching such a page will disappear
  - In the limit, no page that can reach a dead end can have any PageRank at all

- **Spider traps**
  - Groups of pages that all have outlinks but they never link to any other pages

Example

- Suppose that both the PageRank and TrustRank were computed
- Teleport set was page B and D
- Which nodes are not the link spams?
- Is there any link spam?

<table>
<thead>
<tr>
<th>Web Page</th>
<th>PageRank</th>
<th>TrustRank</th>
<th>SpamMess</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>3/9</td>
<td>5/6/10</td>
<td>0.228</td>
</tr>
<tr>
<td>B</td>
<td>2/9</td>
<td>1/6/10</td>
<td>0.264</td>
</tr>
<tr>
<td>C</td>
<td>2/9</td>
<td>1/6/10</td>
<td>0.186</td>
</tr>
<tr>
<td>D</td>
<td>2/9</td>
<td>1/6/10</td>
<td>0.264</td>
</tr>
</tbody>
</table>

Questions?