Agenda

• PageRank – The Algorithm
• Applications
• Sequential Implementation
• Parallel Implementation
• Results
• Observation
• Convergence of PageRank
• References
• Questions?
What is PageRank?

PageRank is an iterative algorithm used by Google Search to rank web pages in their search engine results. A page is considered more important if it is pointed to by other important pages.

How does PageRank work?

The algorithm takes into consideration the number of links to a page and also the quality of these links in order to determine a rough estimate of how important the page is. It is designed with the underlying assumption that more important websites are likely to receive more links.
Source: https://medium.com/analytics-vidhya/google-page-rank-and-markov-chains-d65717b98f9c
Applications

When PageRank is used within applications, it tends to acquire a new name:

- PageRank in Biology and Bioinformatics: GeneRank, ProteinRank, IsoRank
- PageRank in Complex Engineered Systems: MonitorRank
- PageRank of the Linux Kernel
- Roads and Urban Spaces: to predict both traffic flow and human movement.
- PageRank in Literature: BookRank
Sequential Implementation

Here, there are 4 pages: A, B, C and D with links between them as shown.

Initially, (for iteration 0) the pagerank of each page is taken as $\frac{1}{n}$

Thus, $\text{PR}(A) = \text{PR}(B) = \text{PR}(C) = \text{PR}(D) = \frac{1}{4}$

In every successive iteration, the pagerank of each page is calculated as:

$$\text{PR}_n(u) = \frac{1 - d}{n} + d \sum_{v \in B_u} \frac{\text{PR}_{n-1}(v)}{L(v)}$$

where $\text{PR}_n(u) \Rightarrow \text{PageRank of u in nth iteration where } n > 0$;

$B_u \Rightarrow \text{pages pointing to u}$;

$L(v) \Rightarrow \text{number of outbound links from page v}$

d $\Rightarrow \text{damping factor or click-through probability of the surfer (usually 0.85)}$
PageRank – With Example Continued

Damping factor is taken as 1.

<table>
<thead>
<tr>
<th></th>
<th>Iteration 0</th>
<th>Iteration 1</th>
<th>Iteration 2</th>
<th>PageRank at iter 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1/4</td>
<td>1/4</td>
<td>3/8 = 0.375</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>1/4</td>
<td>3/8</td>
<td>5/16 = 0.3125</td>
<td>2</td>
</tr>
<tr>
<td>C</td>
<td>1/4</td>
<td>1/8</td>
<td>1/8 = 0.125</td>
<td>4</td>
</tr>
<tr>
<td>D</td>
<td>1/4</td>
<td>1/4</td>
<td>3/16 = 0.1875</td>
<td>3</td>
</tr>
</tbody>
</table>

Running Time: $O(n + m)$

$n$: number of nodes, $m$: number of edges
Parallel Implementation

- Consider N pages (or nodes) and P processors.
- Each processor selects its own portion of the adjacency matrix (of N/P nodes) that it will work on.
- For iteration 0, a pagerank vector for N pages with each page having 1/N as value, is computed for all the nodes across all the P processors.
- Each processor then calculates an array of the number of connections for each of its set of nodes.
- The pagerank values of each of these nodes are divided by the number of incoming connections to get the weights of each node.
- The weights array is send to all the others nodes in its neighborhood.
- By summing up the received weights, the tentative page ranks are calculated for each node.
- This process is repeated for 40 iterations to get the pagerank of all the pages.
- At the end, each processor will hold the tentative page rank value for its set of pages.
RESULTS
32415 nodes (~200K edges)

<table>
<thead>
<tr>
<th>P</th>
<th>Time (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>648.446</td>
</tr>
<tr>
<td>4</td>
<td>478.173</td>
</tr>
<tr>
<td>8</td>
<td>319.305</td>
</tr>
<tr>
<td>16</td>
<td>295.85</td>
</tr>
<tr>
<td>32</td>
<td>287.182</td>
</tr>
<tr>
<td>64</td>
<td>300.62</td>
</tr>
</tbody>
</table>

Time vs P
74035 nodes (~1M edges)

<table>
<thead>
<tr>
<th>P</th>
<th>Time (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>701.371</td>
</tr>
<tr>
<td>4</td>
<td>519.447</td>
</tr>
<tr>
<td>8</td>
<td>420.014</td>
</tr>
<tr>
<td>16</td>
<td>357.727</td>
</tr>
<tr>
<td>32</td>
<td>300.99</td>
</tr>
<tr>
<td>64</td>
<td>380.658</td>
</tr>
</tbody>
</table>

Time vs P
Observations

• The run time decreases with an increase in the number of processing units.
• When the number of processors is increased beyond 40-50, the runtime starts increasing.
• Thus, the decrease in runtime or increase in speedup is determined by both the computations and the communications across the processors.
• For lower number of processors, computations triumph over communication.
• For higher number of processors, communication plays the major role and thus, the performance starts decreasing.
Convergence of PageRank

• Random Surfer Model
• Ergodic Markov Chains converge to a stationary distribution.

What is a Markov Chain?
• Stochastic Model
• Probability of an event depends only on the state attained in the previous event.
• Real world example: Weather forecast

Ergodic Markov Chain: Irreducible and Aperiodic Markov Chain

Source: https://en.wikipedia.org/wiki/Markov_chain
Convergence of PageRank - Continued

Ergodic Markov Chain:

• Irreducible – able to get from any state to any other state eventually.

• Aperiodic – not cycling back and forth between states at regular intervals.

Such Ergodic Markov Chains eventually converge to a steady-state equilibrium (stationary distribution).

Example: User distribution with 90% iPhone users and 10% Android users.

iPhone: 80% stay with iPhone(72), 20% switch to Android(18)
Android: 70% stay with Android(7), 30% switch to iPhone(3)

Source: https://towardsdatascience.com/the-intuition-behind-markov-chains-713e6ec6ce92
Why PageRank is an Ergodic Markov Chain?

PageRank is both Irreducible and Aperiodic.

**Irreducible** because we can reach any page from any other page following a series of state transitions. (A row filled with zeros (or a sink) in the state transition matrix is replaced with 1/n probability, i.e., random website is chosen)

**Aperiodic** because every diagonal element in the transition matrix $T$ is positive because of including the damping factor.

$T \Rightarrow$ transition matrix  
$\beta \Rightarrow$ damping factor  
$N \Rightarrow$ total number of pages

Thus, PageRank following an Ergodic Markov Chain always converges.

Source: https://towardsdatascience.com/the-intuition-behind-markov-chains-713e6ec6ce92
REFERENCES:


• https://blog.majestic.com/company/understanding-googles-algorithm-how-pagerank-works/

• https://www.shoutmeloud.com/how-to-calculate-pagerank-google-seo.html

• https://cklixx.people.wm.edu/teaching/math410/google-pagerank.pdf

• https://snap.stanford.edu/data/

• https://towardsdatascience.com/the-intuition-behind-markov-chains-713e6ec6ce92

• https://en.wikipedia.org/wiki/Markov_chain
Questions?