PARALLELIZING PAGE RANK

CSE 708 – Venkata Hemanth Athota

University at Buffalo
Graduate School of Education

Index

- History
- Applications
- Algorithm
- Sequential Implementation
- Parallel Implementation
- Results

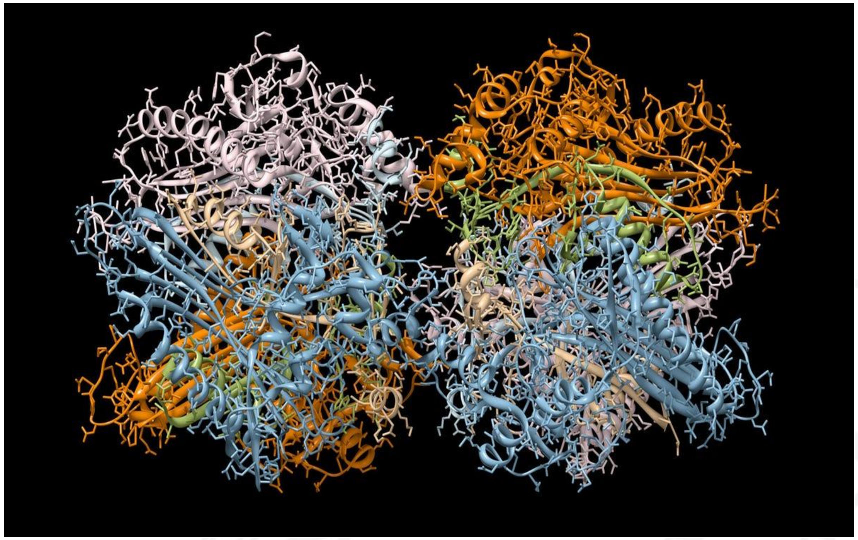
History

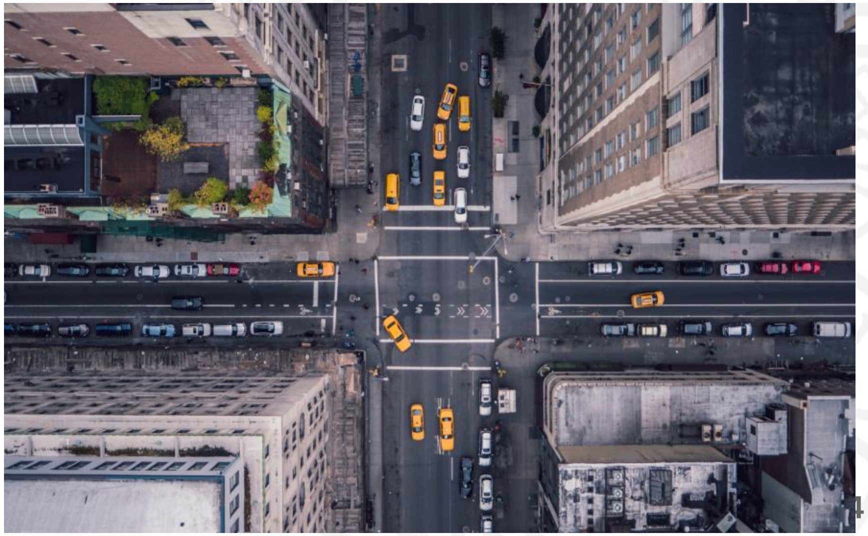
The heart of the Google search engine is the PageRank algorithm, which was described in the paper titled *The PageRank Citation Ranking: Bringing Order to the Web.*

Sergie Brin & Larry Page

Web Graph: The World Wide Web is a graph where web pages are nodes, and hyperlinks between pages are edges. This graph considers a web page with many incoming links more important or authoritative.

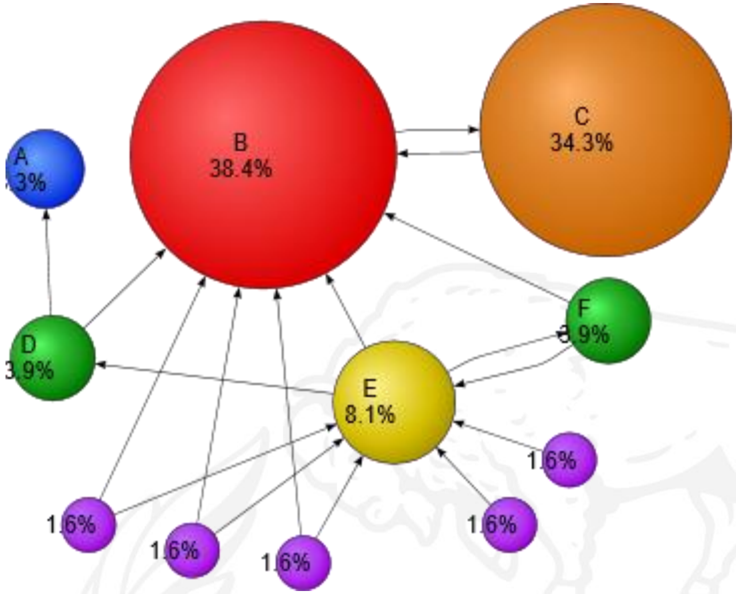
Applications of Page Rank





Algorithm

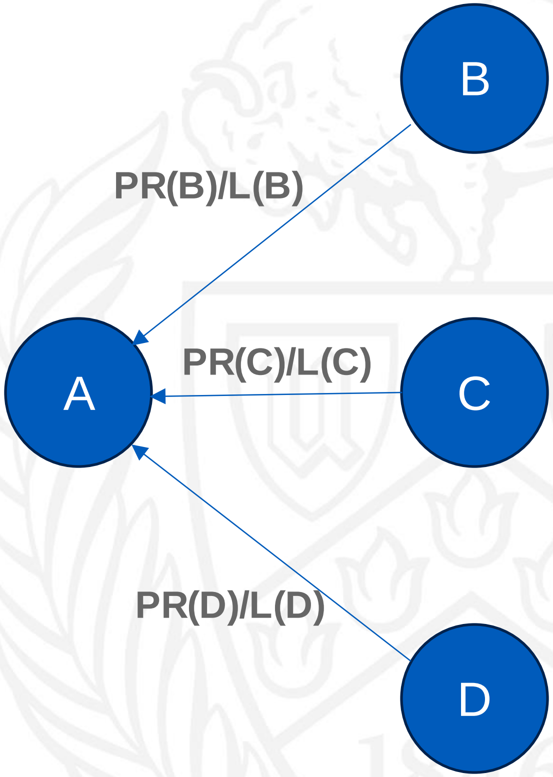
- The PageRank algorithm gives each page a rating of its importance, which is a recursively defined measure whereby a page becomes important if important pages link to it. This definition is recursive because the importance of a page refers back to the importance of other pages that link to it.



- One way to think about PageRank is to imagine a random surfer on the web, following links from page to page. The page rank of any page is roughly the probability that the random surfer will land on a particular page. Since more links go to the important pages, the surfer is more likely to end up there.

Simplified Sequential implementation

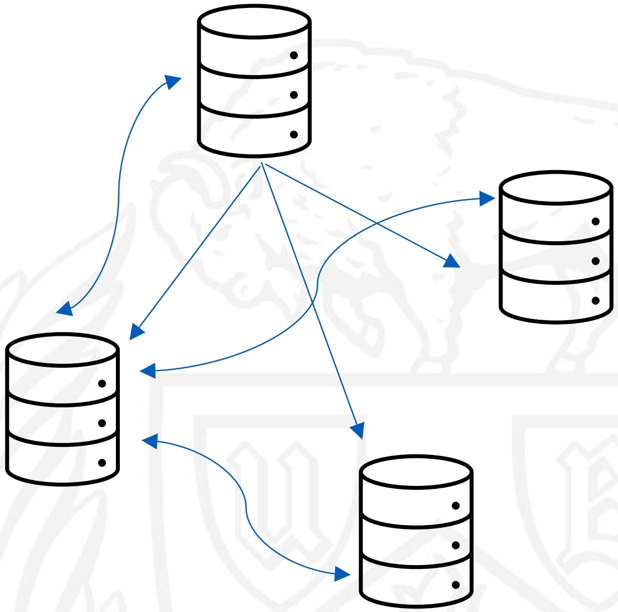
- Iterate over each of the vertices and update individual ranks based on incoming edges.
- All these fractions of votes are added together but, to stop the other pages having too much influence, this total vote is damped down by multiplying it with d.



$$PR(A) = \frac{1-d}{N} + d\left(\frac{PR(B)}{L(B)} + \frac{PR(C)}{L(C)} + \frac{PR(D)}{L(D)} + \cdots\right).$$

Parallelized implementation

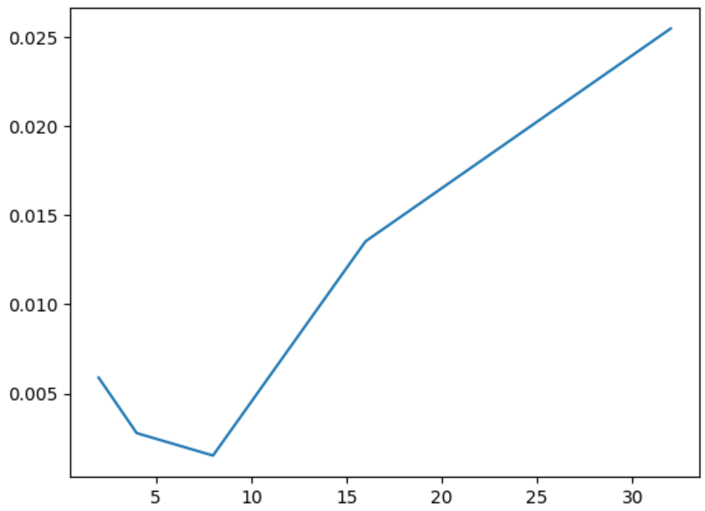
- Root processor reads from a file.
- Divide the NxN graph into (p) partitions and broadcast it to each of the nodes. So [N/p] rows for each.
- Iteratively(*timer starts here*)
  - Calculate the latest PR value
  - Share (the local start to end row in page rank array) with rest of the nodes and update each of its values.
- Perform above 2 steps for k times.
- End the timer & save the rank values



Results

Time taken for different number of processors for a graph with 100 vertices and 2000 edges.
Lowest: 8 nodes.

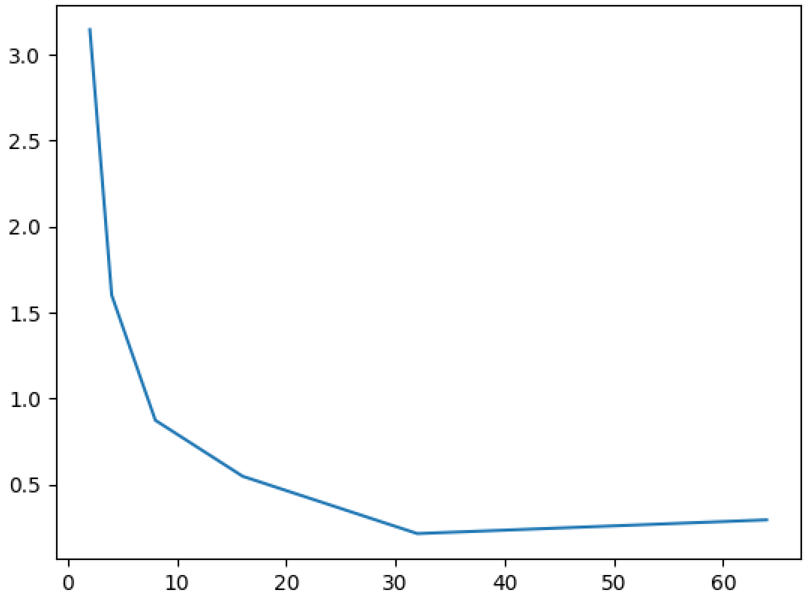
Processors	Time
2	0.005897
4	0.002780
8	0.001521
16	0.013540
32	0.025477



Results

Time taken for different number of processors for a graph with 1000 vertices and 20000 edges.
Lowest: 32 nodes.

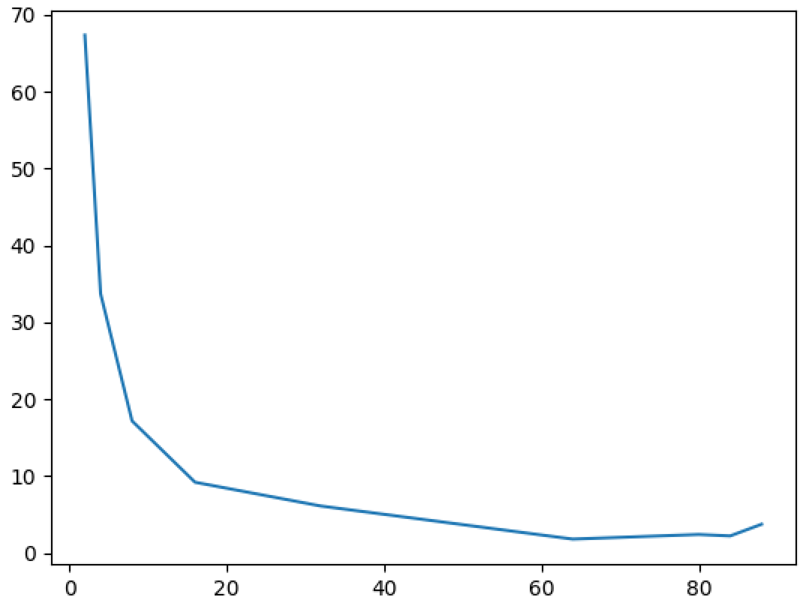
Processors	Time
2	3.145767
4	1.601854
8	0.874434
16	0.548852
32	0.215619
64	0.295514



Results

Time taken for different number of processors for a graph with 10000 vertices and 200,000 edges.
Lowest: 64 nodes.

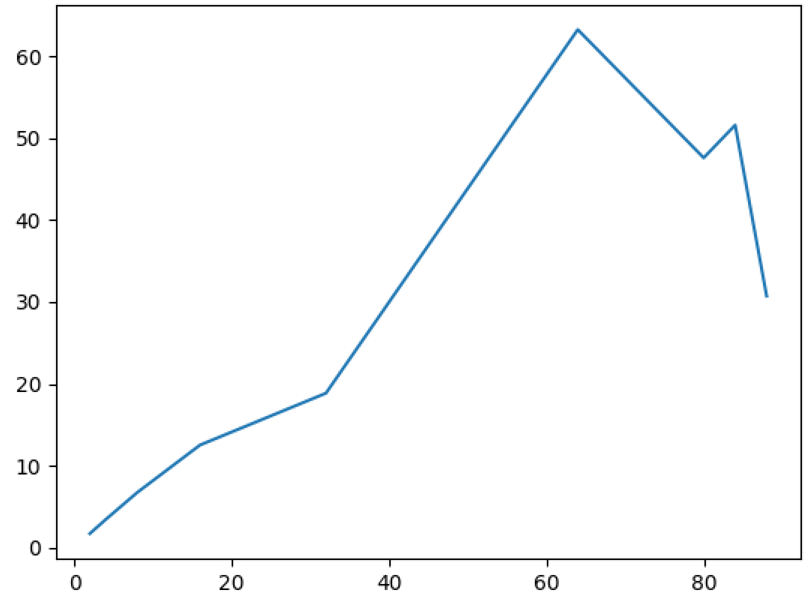
Processors	Time
1	115.460798
2	67.363267
4	33.729089
8	17.190452
16	9.219321
32	6.125911
64	1.826219
80	2.42698
82	2.23899
84	3.76102



Speedup

The speedup is the ratio of time taken sequentially vs parallel for the same program.

Speedup = $T_{sequential} / T_{parallel}$



References

1. https://www.cis.upenn.edu/~mkearns/teaching/NetworkedLife/pagerank.pdf
2. https://en.wikipedia.org/wiki/PageRank
3. https://www.cs.purdue.edu/homes/dgleich/publications/Gleich%202015%20-%20prbeyond.pdf
4. https://docs.ccr.buffalo.edu/en/latest/
5. https://www2.math.upenn.edu/~kazdan/312F12/JJ/MarkovChains/markov_google.pdf
6. https://web.stanford.edu/class/cme194/cgi-bin/
7. https://blog.kagi.com/age-pagerank-over
8. https://www.micsymposium.org/mics_2007/Osmani.pdf
9. https://www.ese.wustl.edu/~nehorai/research/genomic/gene_ranking.html

QUESTIONS?

University at Buffalo
Graduate School of Education