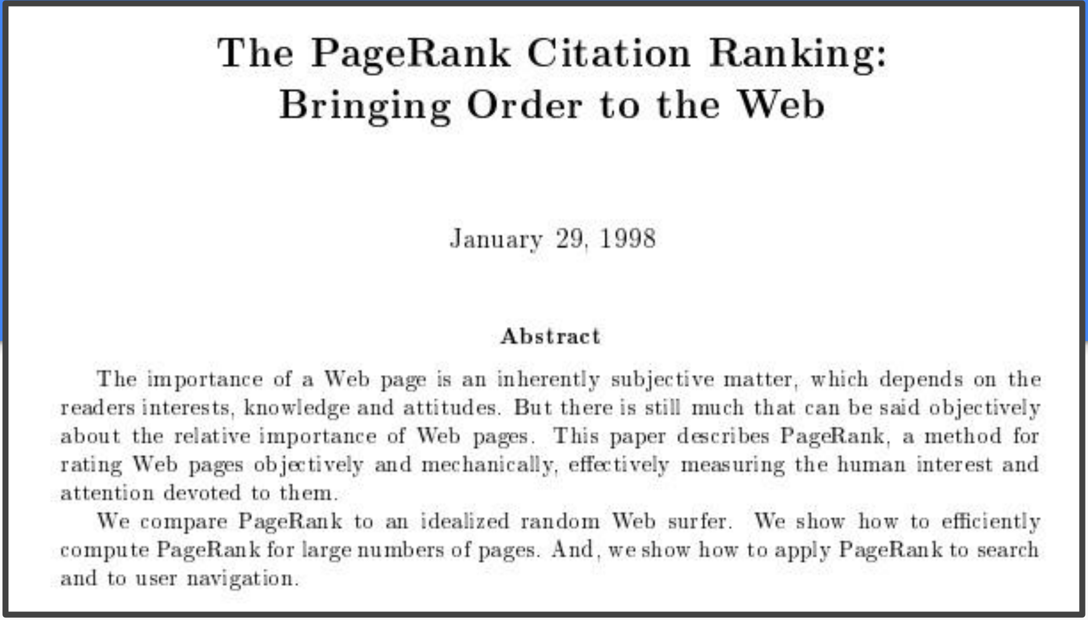
The Google PageRank Algorithm

Prakriti Shrestha & Amna Shahid Majeed
COMSC-312 **Algorithms**
Spring 2021
Mount Holyoke College

Introduction

- PageRank algorithm: what made Google such a successful search engine.



The PageRank Citation Ranking: Bringing Order to the Web

January 29, 1998

Abstract

The importance of a Web page is an inherently subjective matter, which depends on the readers interests, knowledge and attitudes. But there is still much that can be said objectively about the relative importance of Web pages. This paper describes PageRank, a method for rating Web pages objectively and mechanically, effectively measuring the human interest and attention devoted to them.

We compare PageRank to an idealized random Web surfer. We show how to efficiently compute PageRank for large numbers of pages. And, we show how to apply PageRank to search and to user navigation.

Paper: http://ilpubs.stanford.edu:8090/422/1/1999-66.pdf

- Originally described in this paper by Larry Page, Sergey Brin, Rajeev Motwani and Terry Winograd.
- PageRank is a ranking algorithm; its an iterative algorithm that provides a PR score to each webpage and updates their score in each iteration
- After a certain # of iterations, the scores provide a ranking of the pages (i.e. relevance/importance of a webpage for the given search)

Motivating problem

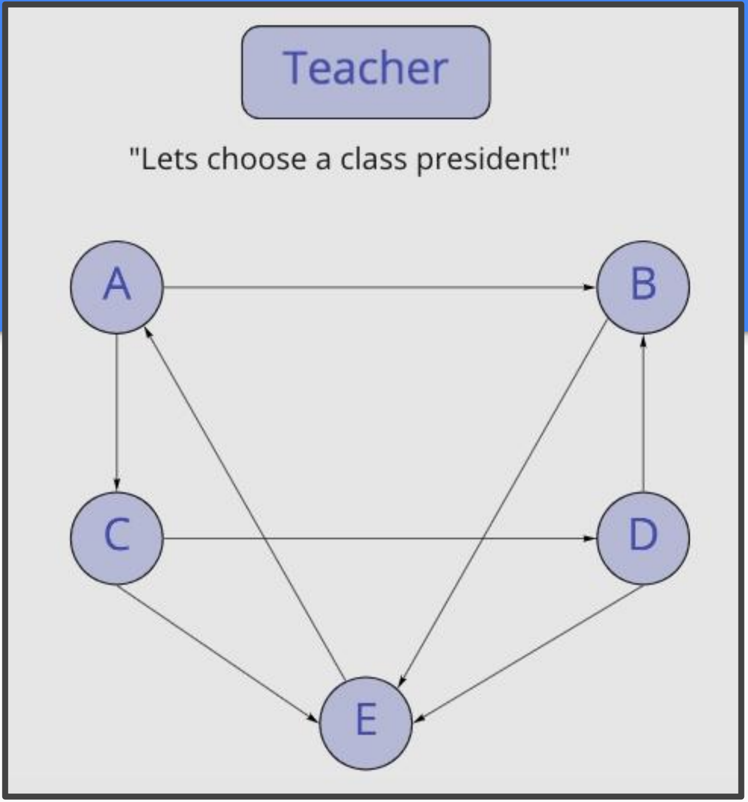
- We have a class of 5 students
- **Problem**: how do we create a ranking to know who the students would prefer most as their *class president*? Votes.

- **Assumptions**:

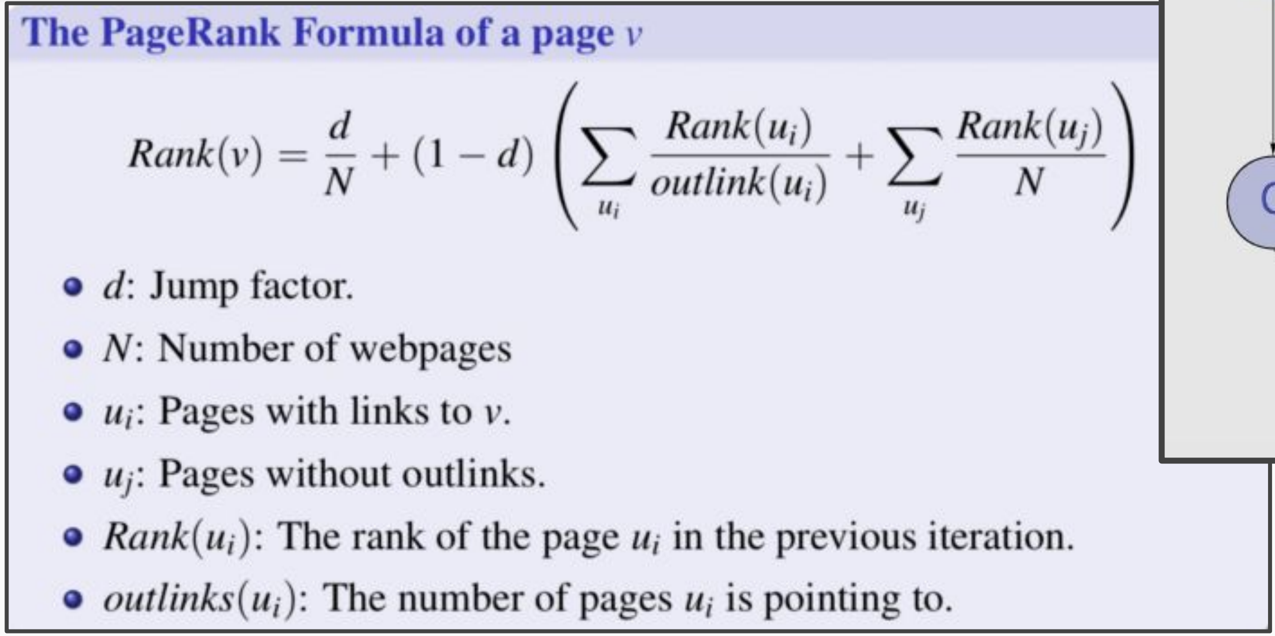
I. Students will not vote for themselves

II. Each student can vote for more than one student

- Here, we've represented this problem as a **directed graph.**
- Each **student** is a **node** and each of their **votes** is an **outgoing edge**



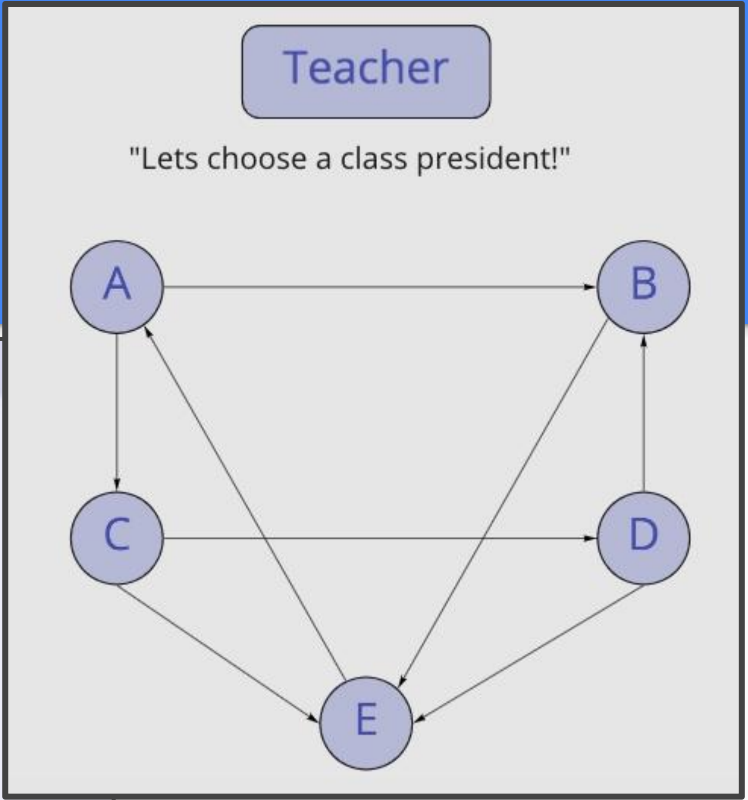
Proposed solution



The PageRank Formula of a page v

$$Rank(v) = \frac{d}{N} + (1-d)\left(\sum_{u_i} \frac{Rank(u_i)}{outlink(u_i)} + \sum_{u_j} \frac{Rank(u_j)}{N}\right)$$

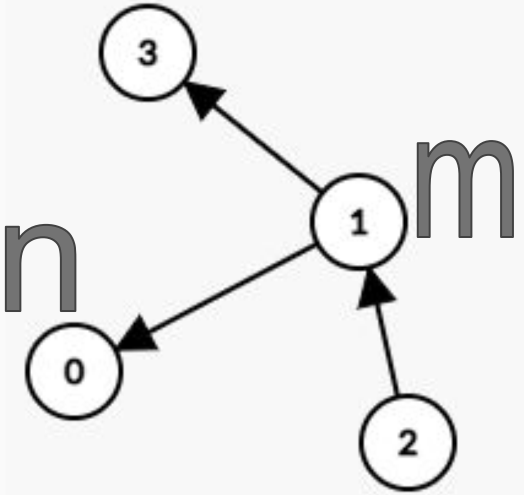
- d: Jump factor.
- N: Number of webpages
- u_i: Pages with links to v.
- u_j: Pages without outlinks.
- $Rank(u_i)$: The rank of the page u_i in the previous iteration.
- $outlinks(u_i)$: The number of pages u_i is pointing to.



Breaking down the algorithm

$$Rank(v) = \frac{d}{N} + (1-d)\left(\sum_{u_i} \frac{Rank(u_i)}{outlink(u_i)} + \sum_{u_j} \frac{Rank(u_j)}{N}\right)$$

$$\sum \frac{PageRank\ of\ inbound\ link}{Number\ of\ links\ on\ that\ page}$$



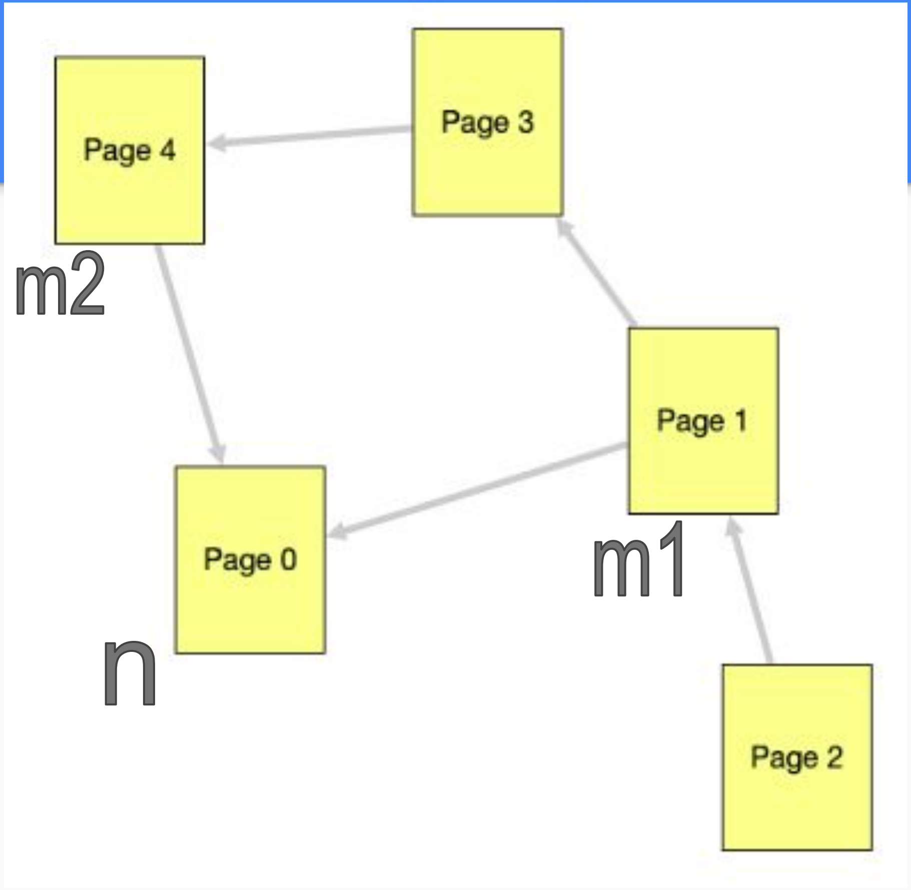
- Two sums are being calculated
- The nodes represent web pages
- The edges represent web links
- Lets understand the first sum for a single node **n**
- For each incoming edge, **m**, to **n,** we make the calculation**:**
- Page rank of **m** / number of outgoing edges from **m**
- We sum these calculations

Calculating $\sum_{u_i} \frac{Rank(u_i)}{outlink(u_i)}$

- Node n → page 0
- Node m1 → page 1
- Node m2 → page 4

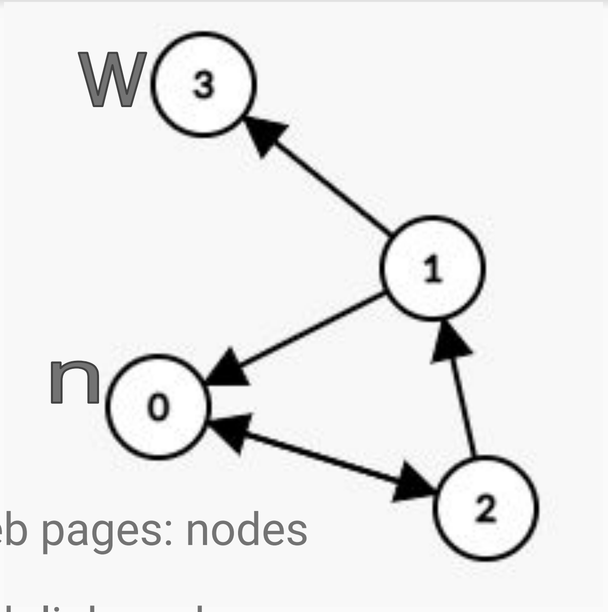
For Rank (Page 0):

(Rank(Page 1)/2) + (Rank(Page 4)/1)



The second sum

$$Rank(v) = \frac{d}{N} + (1-d)\left(\sum_{u_i} \frac{Rank(u_i)}{outlink(u_i)} + \sum_{u_j} \frac{Rank(u_j)}{N}\right)$$

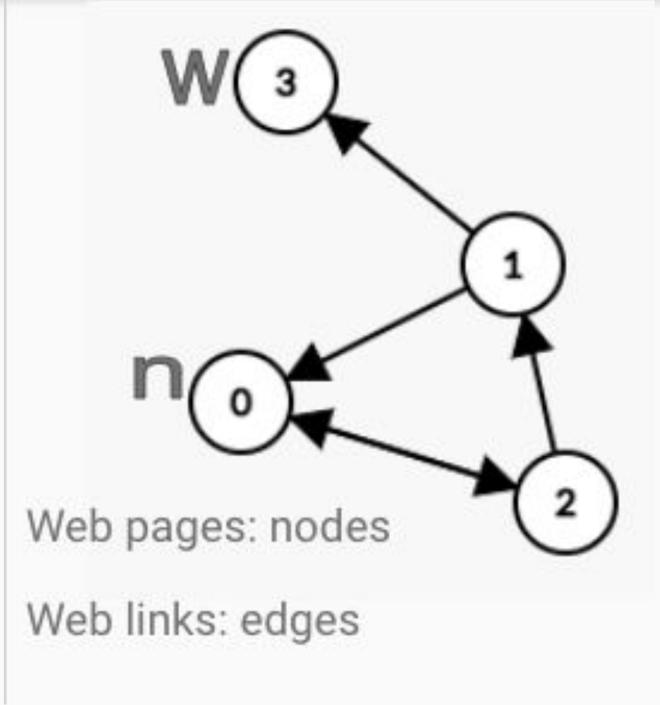


Web pages: nodes

Web links: edges

- While using the web, we don’t necessarily go to another web page by clicking on a link
- There are pages not linked to any of the other pages by an outgoing link
- In that case we would go to another page by typing in its web address
- For example node w
- w has one incoming edge (from node 1)
- w doesn’t have outgoing edge(s) to any node
- To go from w to another node, we will choose a random node out of the rest
- But how do we choose the node?

Calculating $\sum_{u_j} \frac{Rank(u_j)}{N}$



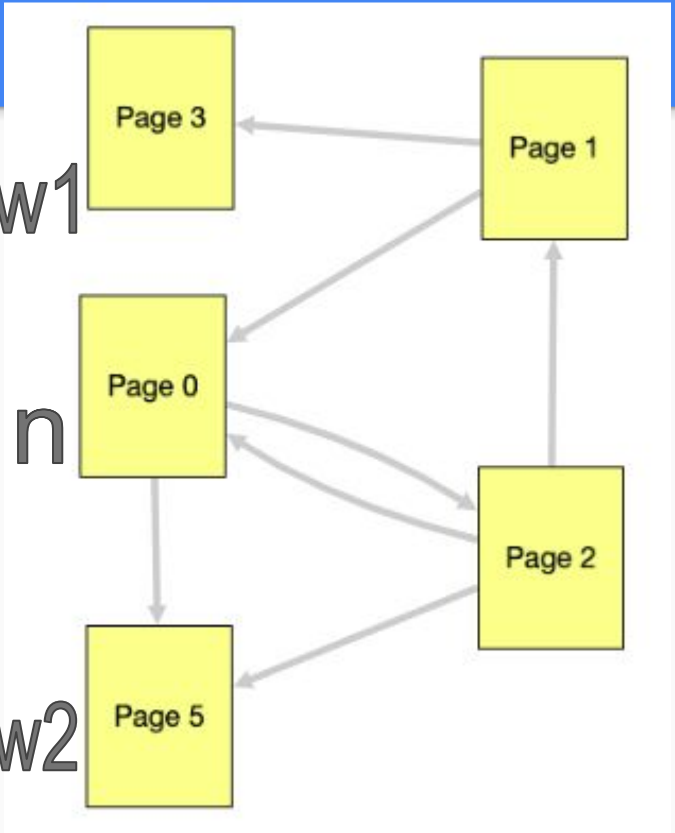
Web pages: nodes

Web links: edges

- By knowing the probability of going from w to each of the other nodes, including n
- Therefore while calculating the page rank for n, we need to account for this probability
- The ranking of n is affected by the probability of jumping to n from w
- This probability is calculated by:
- rank(w) / N (N → total number of nodes)
- We are relocating the probability of w going to another node
- While calculating the rank of each node n we make this calculation for every w
- Sum the calculations and add to the first sum $\sum_{u_i} \frac{Rank(u_i)}{outlink(u_i)} + \sum_{u_j} \frac{Rank(u_j)}{N}$

Calculating

$$\sum_{u_j} \frac{Rank(u_j)}{N}$$



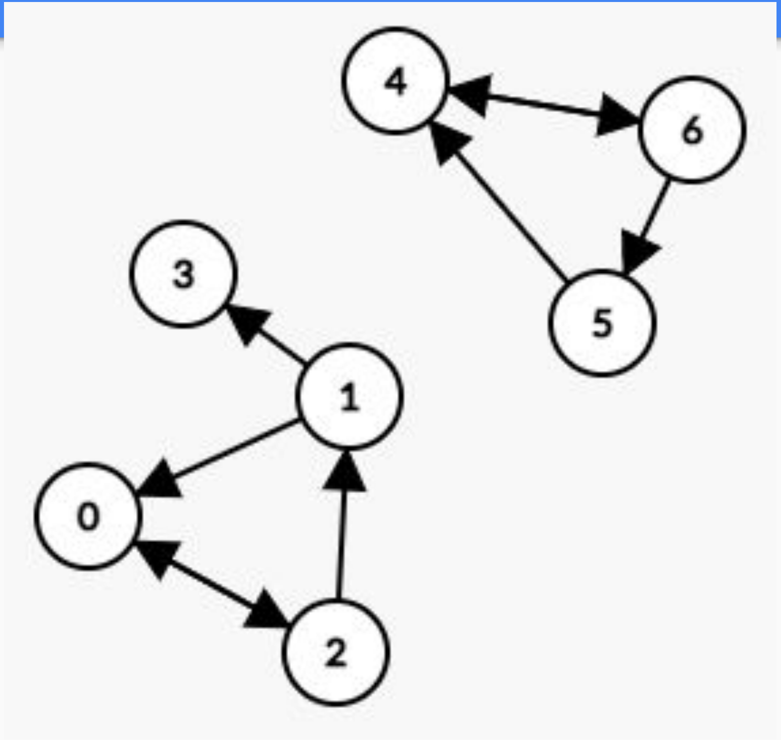
- Node n $\rightarrow$ page 0
- Node w1 $\rightarrow$ page 3
- Node w2 $\rightarrow$ page 5
- N = 5

For Rank(Page 0)

(Rank(Page 3)/5) + (Rank(Page 5)/5)

The Damping Factor (d)

$$\frac{d}{N} + (1 - d)$$



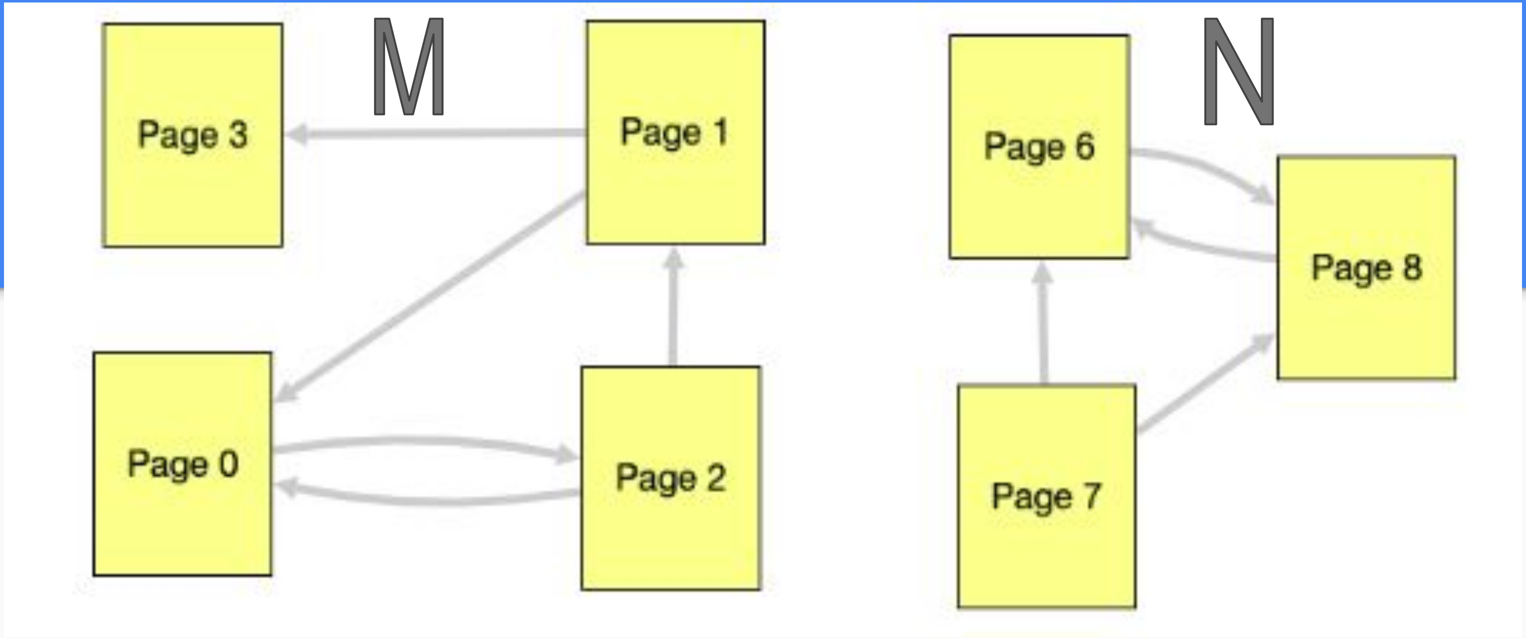
- In our web we will not have links between all the pages
- There might be more than one set of web pages
- We will be ignoring the other set of pages if we only go to web-pages through web-links
- We solve this problem by using the damping

  factor/jump factor

- It is modeling user behaviour of quitting a web page by typing in the web address instead of following a web-link
- The damping factor makes sure we explore the whole network

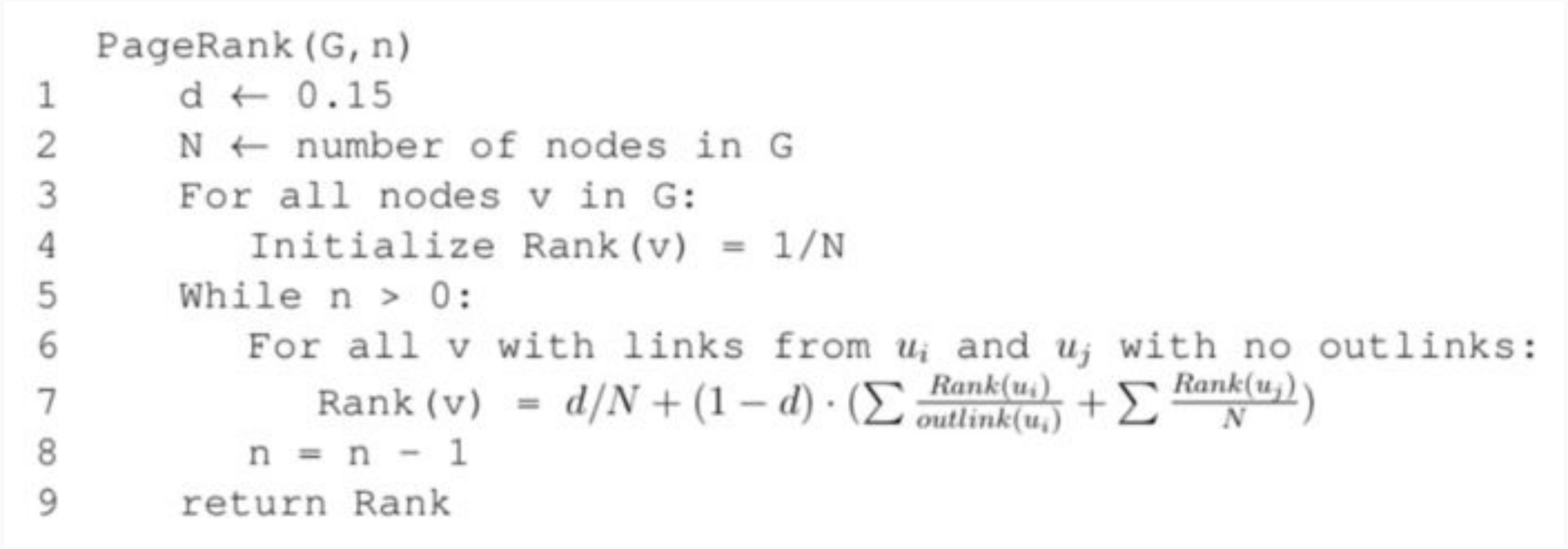
The Damping Factor (d)

- For example, if we have a damping factor of 0.80
- We will be exploring the web pages through web links 80% of the time
- The other 20% of the time the web server will be reset to a random web page
- This will insure that pages not connected by web-links are also explored



- Without the damping factor, we would end up on pages of only set M, or set N
- And the pages of the unexplored set will always have their page ranking at 0
- The damping factor ensures all web pages on a network can be accessed, and ranked accordingly

Pseudocode



```
PageRank(G,n)
    d ← 0.15
    N ← number of nodes in G
    For all nodes v in G:
        Initialize Rank(v) = 1/N
    While n > 0:
        For all v with links from $u_i$ and $u_j$ with no outlinks:
            Rank(v) = $d/N + (1-d)\cdot(\sum \frac{Rank(u_i)}{outlink(u_i)} + \sum \frac{Rank(u_j)}{N})$
        n = n - 1
    return Rank
```

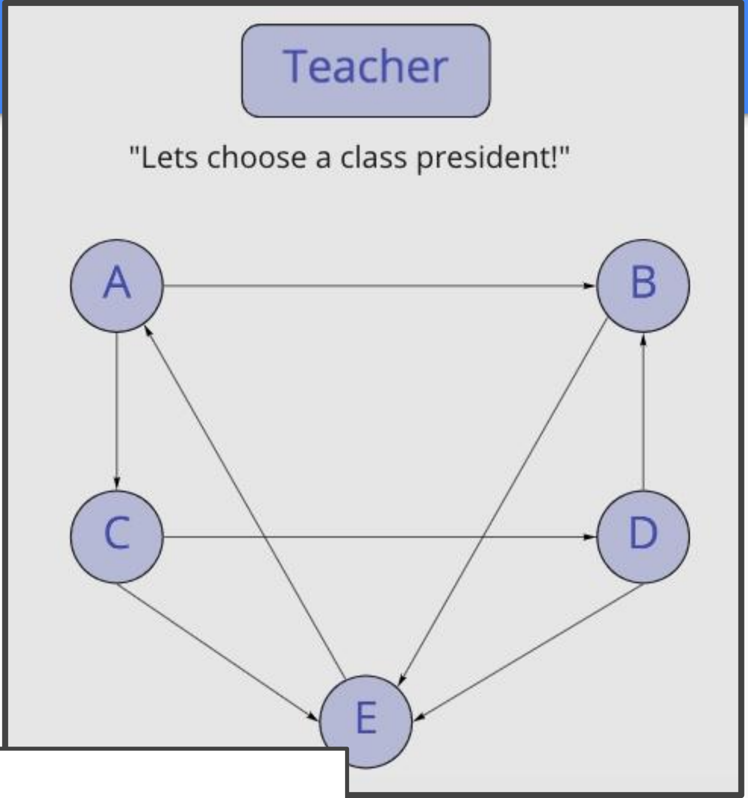
https://swang21.medium.com/what-is-googles-pagerank-algorithm-d0ac17cbc167

Votes casted:

- $A \rightarrow B, C$
- $B \rightarrow E$
- $C \rightarrow D, E$
- $D \rightarrow B, E$
- $E \rightarrow A$

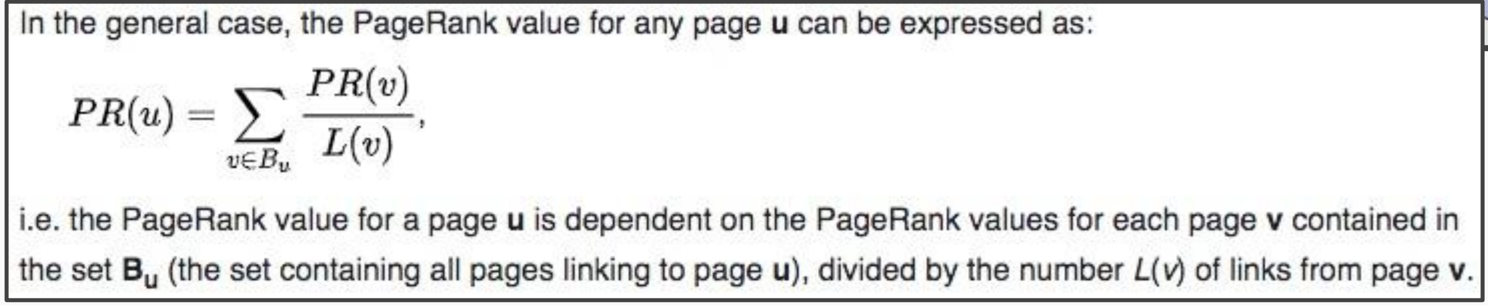
Votes received:

- $A \rightarrow E$
- $B \rightarrow A, D$
- $C \rightarrow A$
- $D \rightarrow C$
- $E \rightarrow B, C, D$



PageRank - important points to take note of:

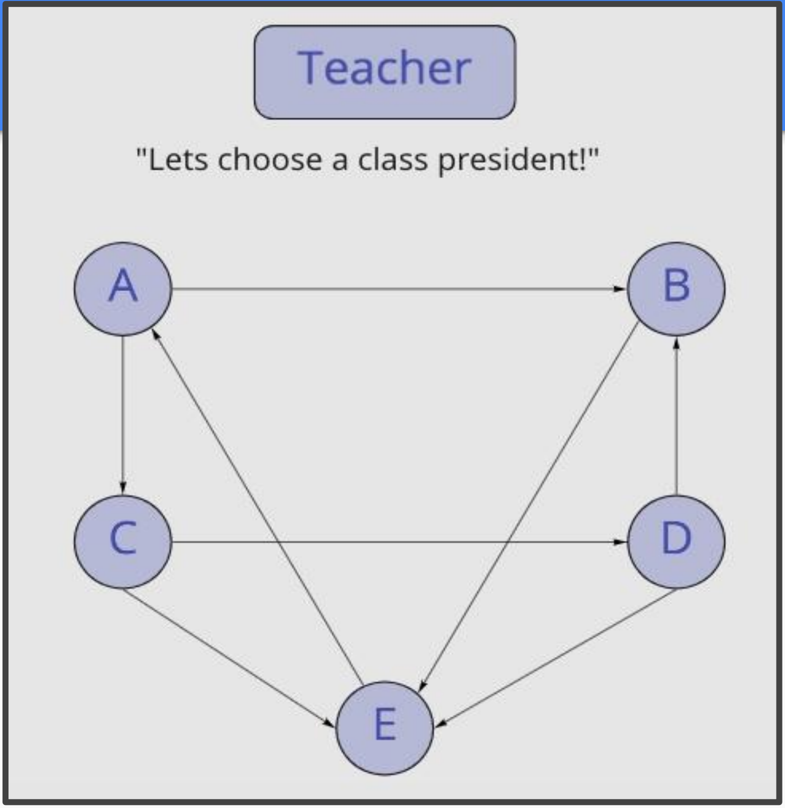
- Ranking increases by both **quantity** and **quality** of votes
- Quantity: how many votes were received?
- Quality: whose votes were received? A vote from a more well-connected node will be more valuable
- The more votes each student casts, the less important each of their votes will be


In the general case, the PageRank value for any page **u** can be expressed as:

$$PR(u) = \sum_{v \in B_u} \frac{PR(v)}{L(v)},$$

i.e. the PageRank value for a page **u** is dependent on the PageRank values for each page **v** contained in the set $\mathbf{B_u}$ (the set containing all pages linking to page **u**), divided by the number $L(v)$ of links from page **v**.

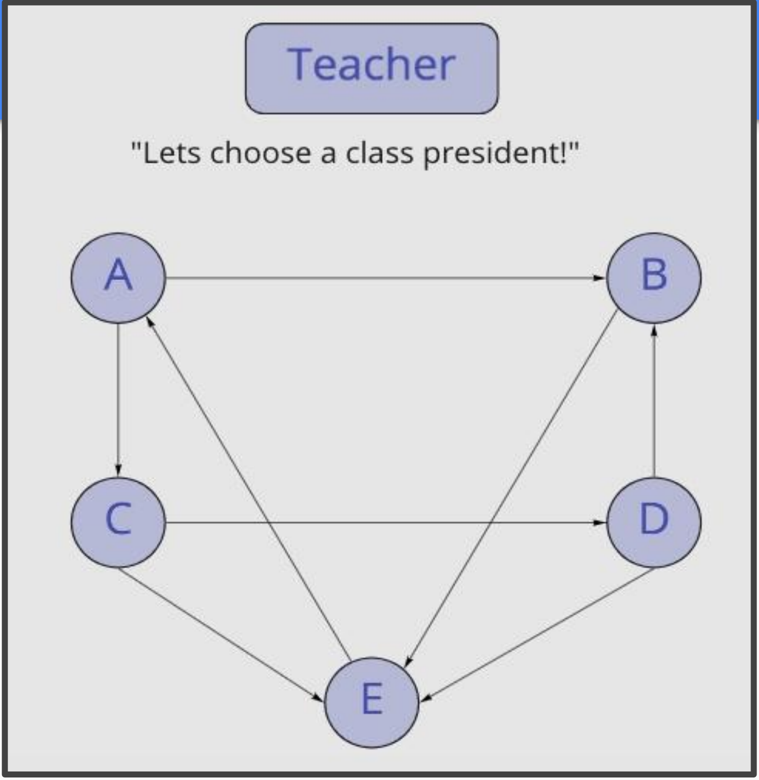
Iterations/ Nodes:	0th	1st	2nd	Final Ranks
A				
B				
C				
D				
E				
Total				



0th iteration:

- We'll start with the same score for all nodes
- 1/n for each node
- Since in this case, n=5, so:
  - $PR(A)_{i=0} = PR(B)_{i=0} = PR(C)_{i=0} = PR(D)_{i=0} = PR(E)_{i=0} = \frac{1}{5}$

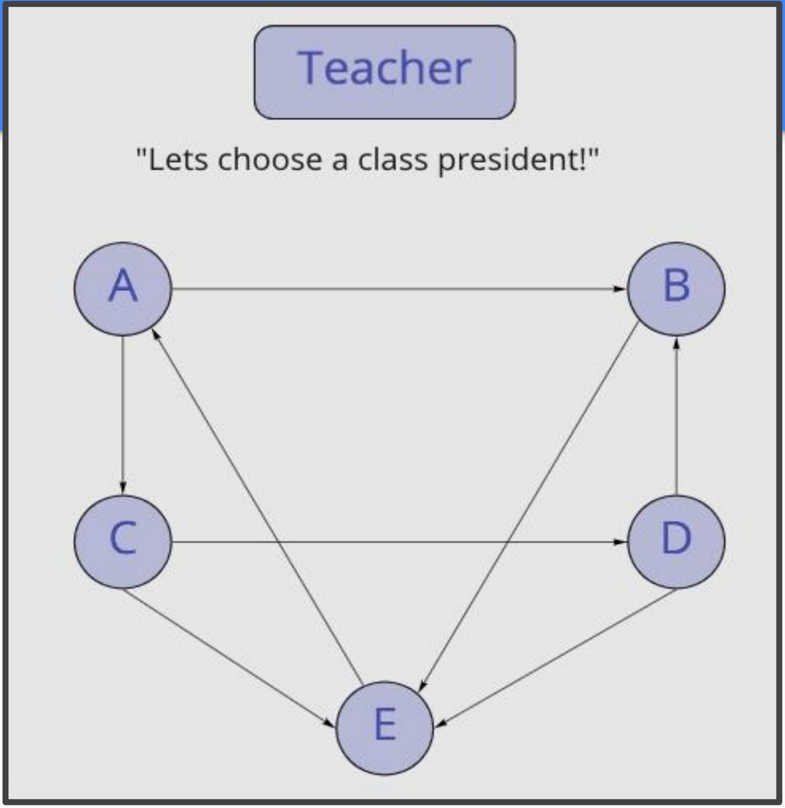
Iterations/ Nodes:	0th	1st	2nd	Final Ranks
A	1/5			
B	1/5			
C	1/5			
D	1/5			
E	1/5			
Total	1			



1st iteration:

- $PR(A)_{i=1} = PR(E)_{i=0} / E_{out} = (⅕)/1 =$ **1/5**
- $PR(B)_{i=1} = PR(A)_{i=0} / A_{out} + PR(D)_{i=0} / D_{out} = (⅕)/2 + (⅕)/2 =$ **1/5**
- $PR(C)_{i=1} = PR(A)_{i=0} / A_{out} = (⅕)/2 =$ **1/10**
- $PR(D)_{i=1} = PR(C)_{i=0} / C_{out} = (⅕)/2 =$ **1/10**
- $PR(E)_{i=1} = PR(B)_{i=0} / B_{out} + PR(C)_{i=0} / C_{out} + PR(D)_{i=0} / D_{out} = (⅕)/1 + (⅕)/2 + (⅕)/2 =$ **2/5**

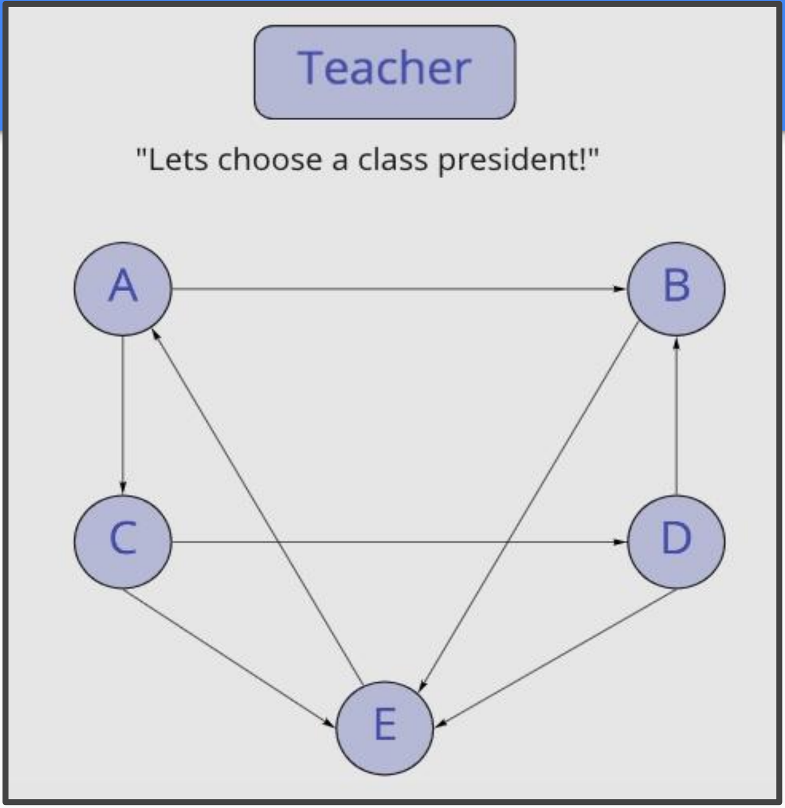
Iterations/ Nodes:	0th	1st	2nd	Final Ranks
A	1/5	1/5		
B	1/5	1/5		
C	1/5	1/10		
D	1/5	1/10		
E	1/5	2/5		
Total	1	1		



2nd iteration:

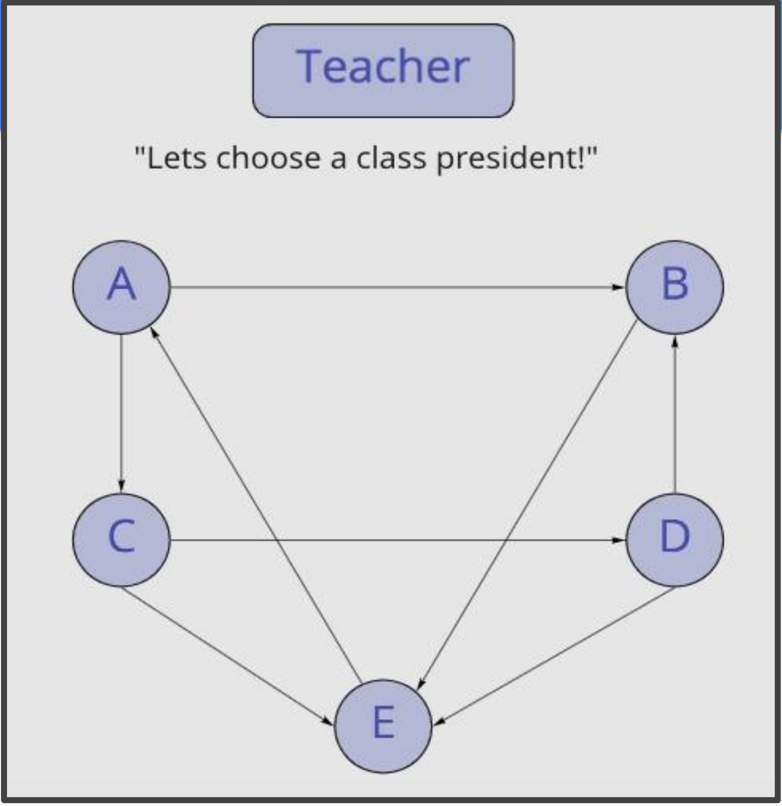
- $PR(A)_{i=2} = PR(E)_{i=1} / E_{out} = (⅖) / 1 = \mathbf{2/5}$
- $PR(B)_{i=2} = PR(A)_{i=1} / A_{out} + PR(D)_{i=1} / D_{out} = (⅕) / 2 + (1/10) / 2 = \mathbf{3/20}$
- $PR(C)_{i=2} = PR(A)_{i=1} / A_{out} = (⅕) / 2 = \mathbf{1/10}$
- $PR(D)_{i=2} = PR(C)_{i=1} / C_{out} = (1/10) / 2 = \mathbf{1/20}$
- $PR(E)_{i=2} = PR(B)_{i=1} / B_{out} + PR(C)_{i=1} / C_{out} + PR(D)_{i=1} / D_{out} = (⅕) / 1 + (1/10)/2 + (1/10) / 2 = \mathbf{3/10}$

Iterations/ Nodes:	0th	1st	2nd	Final Ranks
A	1/5	1/5	2/5 = <u>0.40</u>	
B	1/5	1/5	3/20 = <u>0.15</u>	
C	1/5	1/10	1/10 = <u>0.10</u>	
D	1/5	1/10	1/20 = <u>0.05</u>	
E	1/5	2/5	3/10 = <u>0.30</u>	
Total	1	1	1	

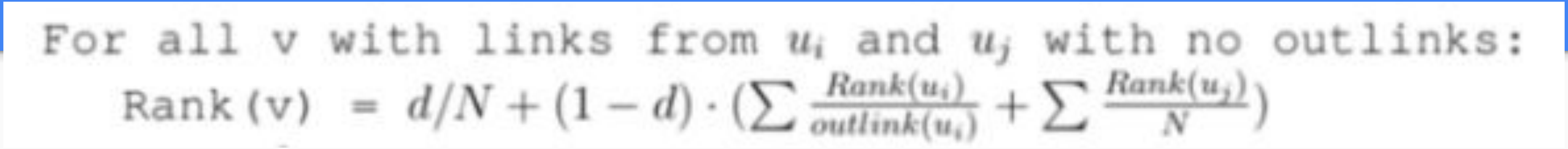


If an actual implementation, the algorithm would probably be allowed to iterate 100s (if not 1000s) of times, but for this problem, we will stop at 2 iterations.

Iterations/ Nodes:	0th	1st	2nd	Final Ranks
A	1/5	1/5	2/5 = 0.40	1st
B	1/5	1/5	3/20 = 0.15	3rd
C	1/5	1/10	1/10 = 0.10	4th
D	1/5	1/10	1/20 = 0.05	5th
E	1/5	2/5	3/10 = 0.30	2nd
Total	1	1	1	



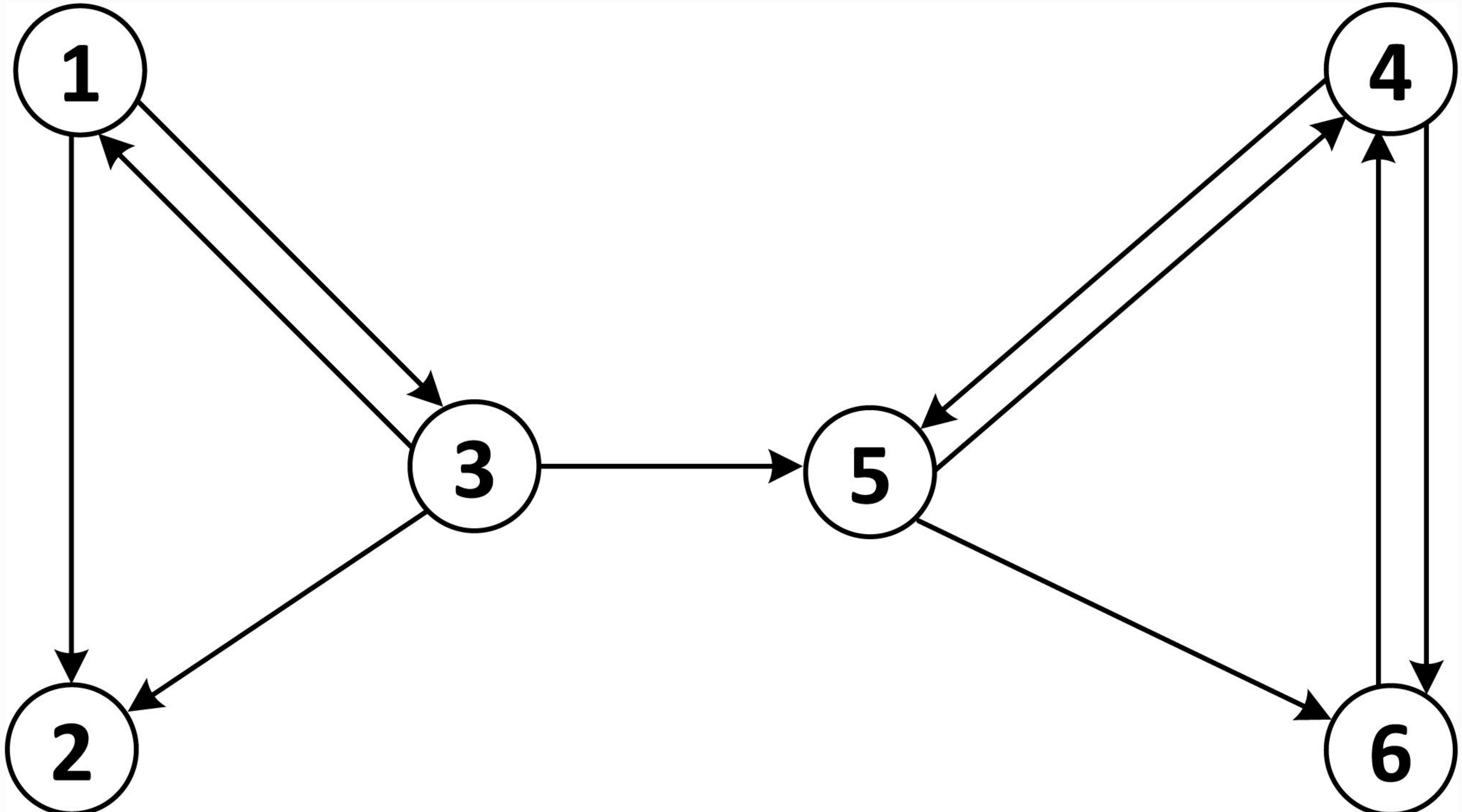
Runtime Analysis (single iteration)



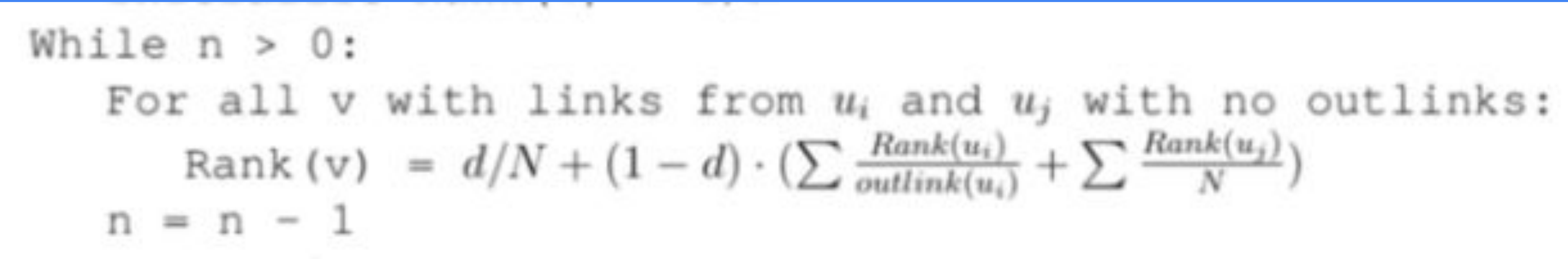
For all v with links from u_i and u_j with no outlinks:

$$\text{Rank(v)} = d/N + (1-d)\cdot\left(\sum \frac{Rank(u_i)}{outlink(u_i)} + \sum \frac{Rank(u_j)}{N}\right)$$

- There are N nodes and M edges
- Go through each node (v) to calculate its page rank
- There are a total of N nodes
- Go through all outcoming edges for each v
- Total outcoming edges = M
- End up going through all edges
- Runtime for a *single iteration*: **N + M**


1
4
3
5
2
6

Final Runtime



```
While n > 0:
    For all v with links from $u_i$ and $u_j$ with no outlinks:
        Rank(v) = $d/N + (1-d) \cdot (\sum \frac{Rank(u_i)}{outlink(u_i)} + \sum \frac{Rank(u_j)}{N})$
    n = n - 1
```

- We carry out n iterations of this algorithm
- Giving us the final runtime:
- **O(n*(N+M))**

THANK YOU!

References

- https://swang21.medium.com/what-is-googles-pagerank-algorithm-d0ac17cbc167
- http://computerscience.chemeketa.edu/cs160Reader/_static/pageRankApp/index.html
- https://www.youtube.com/watch?v=P8Kt6Abq_rM
- https://web.stanford.edu/class/cs54n/handouts/24-GooglePageRankAlgorithm.pdf
- https://www.youtube.com/watch?v=meonLcN7LD4
- https://csacademy.com/app/graph_editor/
- https://en.wikipedia.org/wiki/PageRank#