Software Project Lab - 1

Google’s PageRank Algorithm

Presented by
Shah Alam Abir
Roll: 1439
Session: 2021-2022

Supervised by
Dr. Ahmedul Kabir
Associate Professor
IIT, University of Dhaka

Table of contents

CONTENTS

1

Introduction

Introduction

- Google's PageRank Algorithm is an algorithm used by Google Search Engine to rank web pages in search engine results.

- It was developed by **Larry Page** and **Sergey Brin**, the co-founders of Google, while they were graduate students at Stanford University.

Introduction

- PageRank is designed to assess the importance and relevance of web pages in the context of search engine results.

- The underlying idea is that valuable and authoritative web pages are more likely to be linked to by other web pages.

- Therefore, a web page with many high-quality incoming links is considered more valuable and is likely to rank higher in search results.

2

Project Motivation

Project Motivation

- Foundation of Modern Search Engines
- Refining Search Precision
- Impact on Accessibility
- Real-World Applications
- Academic and Research Value

3

Features

Features

Main Feature:

- Implementation of PageRank Algorithm (that was developed by Larry Page & Sergey Brin) by solving 2 core page ranking problems.

Sub Feature:

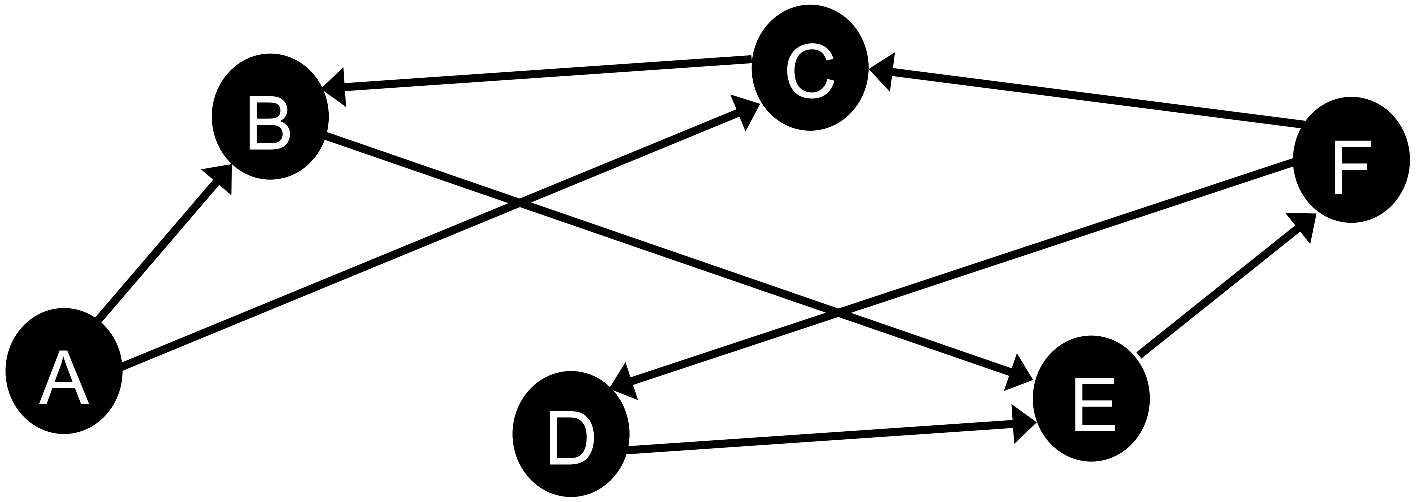
- Can add new pages
- Can find neighbours of a particular page
- Show List of the dangling nodes
- Visualizing Pages ranking after every iterations of iterative model
- Showing Initial Transition Matrix (made from the web graph)
- Showing Gooogle Matrix (made of PageRank formula)

4

The Web as a Graph

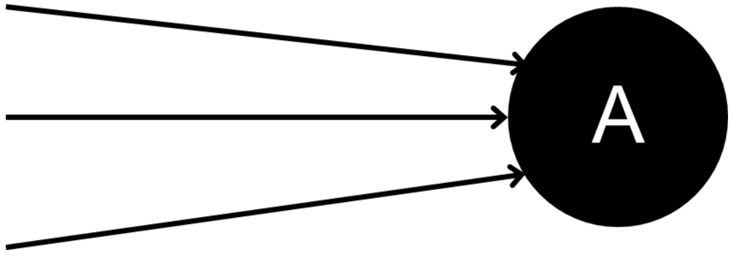
The web as a Graph

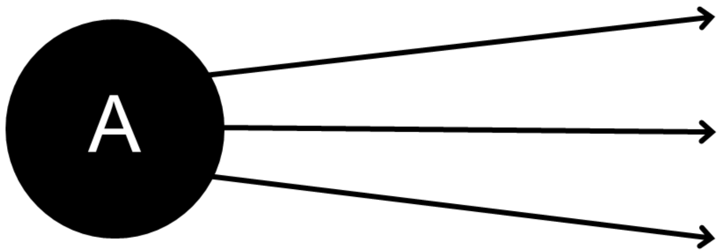
- We can represent WWW’s structure as a huge directed graph, where node’s of that graph are the webpages and edges are the links between webpages.



The web as a Graph

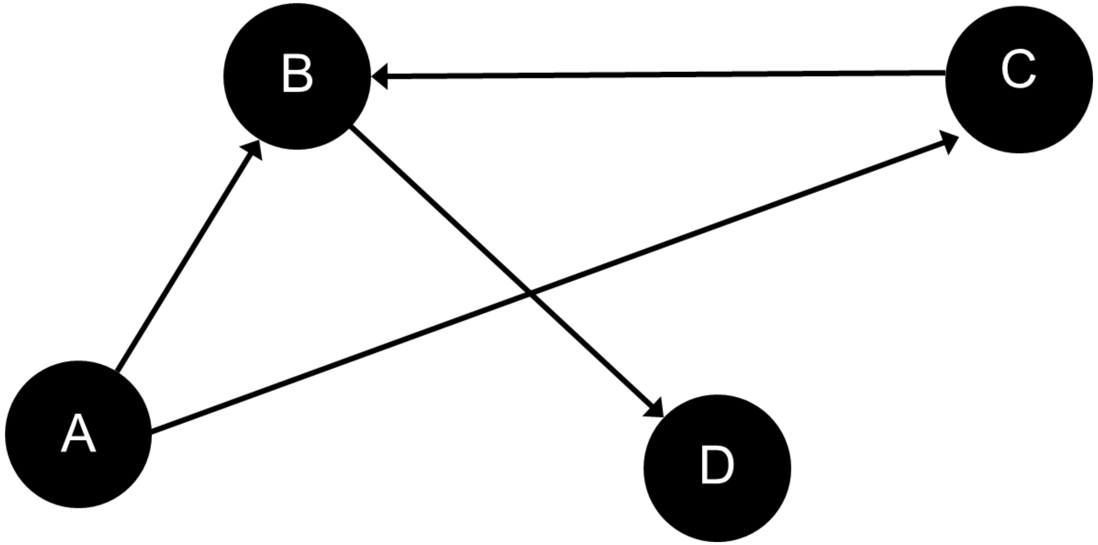
- **Inbound links** are links that point from another website to the target website
- **Outbound links** are links that point to another websites from target website





The web as a Graph

- **Dangling Nodes** are the nodes with no outbound links
- **Dangling Links** are links that point to the dangling nodes



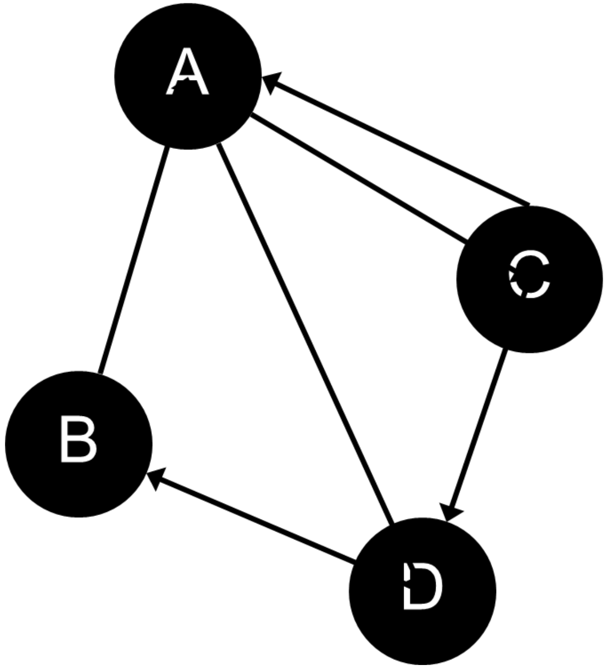
- Here **D** is a Dangling Node
- **BD** is a dangling Link

5

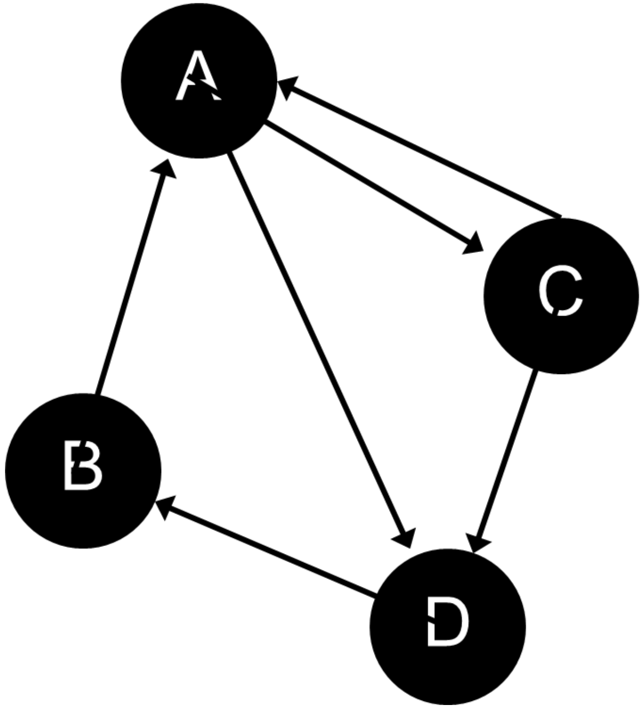
Random Surfer Model

Random Surfer Model

- The Random Surfer Model is a conceptual framework used in the context of web page ranking algorithms, particularly in understanding Google's PageRank algorithm. It is a simplified representation of how a hypothetical internet user behaves when navigating the web and is used to explain the concept of PageRank.



Random Surfer Model



Matrix Presentation

$$r_{(k+1)}(P_i) = \sum_{P\,j \in B_{P\,j}} \frac{r_k\,(P_j)}{|P_j|}$$

The PageRank of a page P_i, denoted $r(P_i)$, is the sum of the Page-Ranks of all pages pointing into P_i.

where B_{P_i} is the set of pages pointing into P_i and $|P_j|$ is the number of out-links from page P_j .

	A	B	C	D
A	0	0	0.5	0.5
B	1	0	0	0
C	0.5	0	0	0.5
D	0	1	0	0

Transition matrix **H**

Random Surfer Model (Iterative Calculation)

Initializing page rank for every page with 1/n. Where, n is the total number of webpages of the graph. Thus, we will have another matrix **X** with initialized page-rank value.

From previous slide we got Transition matrix **H**

We will perform matrix multiplication **X = HX** iteratively until it converge. Then we will get a modified page-rank matrix **X**. Now, if we sort them according to probability, then we will get webpages in descending order.

A	¼
B	¼
C	¼
D	¼

Matrix **X**

Problems with Random Surfer Model

But, there is two major problem with this Random Surfer Model. These are –

1. If there is any dangling node [webpage], then all the elements of the final matrix will be 0.

2. If some nodes of the graph is highly connected, then probability of these particular group of nodes becomes significantly higher then other nodes. Because of this, a random surfer iterates through a group of nodes again and again.

6

PageRank Formula

PageRank Formula

To solve first problems-

We have to convert **H** matrix to a *Stochastic* matrix. This *Stochastic* matrix solves the problem of dangling node by initializing nodes probability with a value. If **S** is a *Stochastic* matrix, then-

$$\mathbf{S = H + a(1/n}\boldsymbol{e^T}\mathbf{)}$$

Here,
S, Stochastic Matrix
H, Transition Matrix
a, is a (nx1) matrix. Value of a_i is 1 if a dangling node else 0
e^T, is a (1xn) matrix. Every element is 1.

PageRank Formula

To solve second problems-

We have to make ***GOOGLE (G)*** matrix from ***S*** matrix & ***H*** matrix. This **G** matrix ensures teleportation from the dense part to the sparse part of the graph. If **G** is a ***GOOGLE*** matrix, then-

$$\mathbf{G} = \boldsymbol{\alpha S} + (\mathbf{1} - \boldsymbol{\alpha})\frac{\mathbf{1}}{\boldsymbol{\alpha}}\boldsymbol{e}\boldsymbol{e}^{\boldsymbol{T}} = \boldsymbol{\alpha H} + (\boldsymbol{\alpha a} + (\mathbf{1} - \boldsymbol{\alpha})\boldsymbol{e})\frac{\mathbf{1}}{\boldsymbol{n}}\boldsymbol{e}^{\boldsymbol{T}}$$

Here,
G, GOOGLE Matrix
S, Scholastic Matrix
H, Transition Matrix
a, is a (nx1) matrix. Value of a_i is 1 if a dangling node else 0
e, is a (nx1) matrix. Every element is 1
e^T, is a (1xn) matrix. Every element is 1.
α, is the damping factor. It ensures the teleportation.

Damping Factor

The damping factor in the PageRank algorithm is a parameter that represents the probability that a random surfer, navigating the web by clicking on links, will continue to browse and click on links within the current web page rather than jumping to a completely random page.

PageRank Formula

Now, we got our final formula to make a ***GOOGLE matrix*** by doing *Stochastic adjustment* & *Primitive adjustment* of the initial *Transition matrix*, **H**.

$$\mathbf{G} = \boldsymbol{\alpha S} + (\mathbf{1} - \boldsymbol{\alpha})\frac{\mathbf{1}}{\boldsymbol{\alpha}}\boldsymbol{e}\boldsymbol{e}^{\boldsymbol{T}} = \boldsymbol{\alpha H} + (\boldsymbol{\alpha a} + (\mathbf{1} - \boldsymbol{\alpha})\boldsymbol{e})\frac{\mathbf{1}}{\boldsymbol{n}}\boldsymbol{e}^{\boldsymbol{T}}$$

Now, We will perform Iterative matrix multiplication **X = HX** until it converges. Then we will get a modified page-rank matrix, **X**.

Now, if we sort matrix **X** according to probabilistic value, then we will get webpages in descending order.

Thank you