Assignment No: 5

Title: Implement Page Rank Algorithm.

Problem Definition:

Implement Page Rank Algorithm. (Use python or beautiful soup for implementation).

Outcome:

Students will be able to,

1. Apply various tools and techniques for information retrieval and web mining.
2. Evaluate and analyze retrieved information.

Theory:

Introduction to PageRank Algorithm:

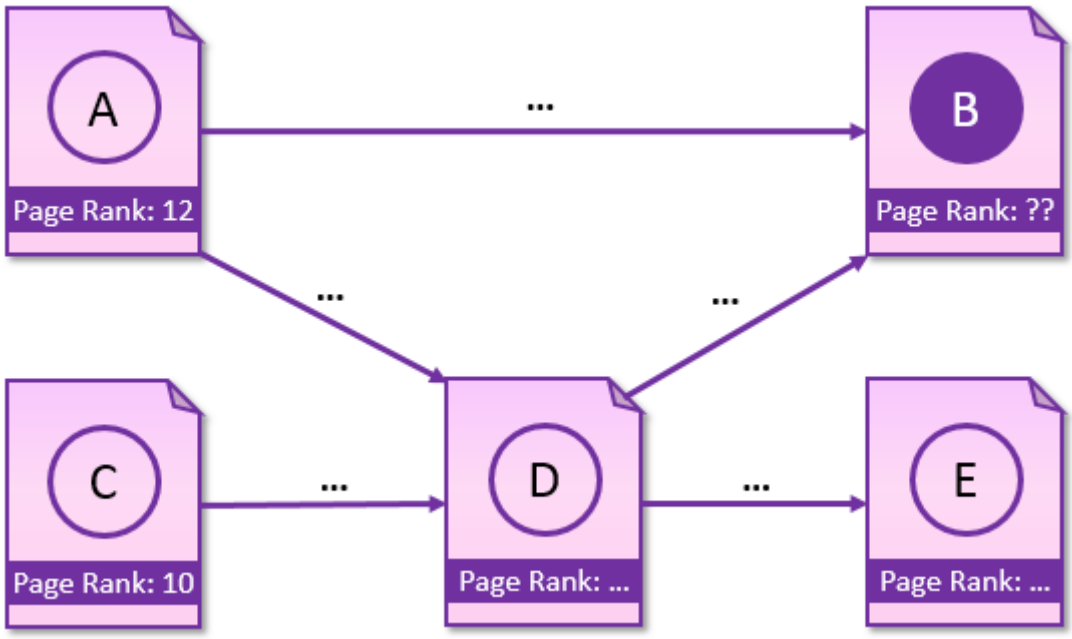
The PageRank algorithm is applicable in web pages. Web page is a directed graph, we know that the two components of Directed graph -nodes and connections. The pages are nodes and hyperlinks are the connections, the connection between two nodes.

We can find out the importance of each page by the PageRank and it is accurate. The value of the PageRank is the probability will be between 0 and 1.

The PageRank value of individual node in a graph depends on the PageRank value of all the nodes which connect to it and those nodes are cyclically connected to the nodes whose ranking we want, we use converging iterative method for assigning values to PageRank.

Different Page Rank Algorithms:

1. PageRank Algorithm
2. Weighted PageRank Agorithm
3. HITS Algorithm
4. Distance Rank Algorithm
5. Eigen Rumor Algorithm



Algorithm:

The PageRank algorithm outputs a probability distribution used to represent the likelihood that a person randomly clicking on links will arrive at any particular page. PageRank can be calculated for collections of documents of any size. It is assumed in several research papers that the distribution is evenly divided among all documents in the collection at the beginning of the computational process. The PageRank computations require several passes, called “iterations”, through the collection to adjust approximate PageRank values to more closely reflect the theoretical true value.

Simplified Algorithm:

Assume a small universe of four web pages: A, B, C, and D. Links from a page to itself, or multiple outbound links from one single page to another single page, are ignored. PageRank is initialized to the same value for all pages. In the original form of PageRank, the sum of PageRank over all pages was the total number of pages on the web at that time, so each page in this example would have an initial value of 1. However, later versions of PageRank, and the remainder of this section, assume a probability distribution between 0 and 1. Hence the initial value for each page in this example is 0.25.

The PageRank transferred from a given page to the targets of its outbound links upon the next iteration is divided equally among all outbound links.

If the only links in the system were from pages B, C, and D to A, each link would transfer 0.25 PageRank to A upon the next iteration, for a total of 0.75.

Suppose instead that page B had a link to pages C and A, page C had a link to page A, and page D had links to all three pages. Thus, upon the first iteration, page B would transfer half of its existing value, or 0.125, to page A and the other half, or 0.125, to page C. Page C would transfer all of its existing value, 0.25, to the only page it links to, A. Since D had three outbound links, it would transfer one-third of its existing value, or approximately 0.083, to A. At the completion of this iteration, page A will have a PageRank of approximately 0.458.

In other words, the PageRank conferred by an outbound link is equal to the document's own PageRank score divided by the number of outbound links L( ).

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

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

page v. The algorithm involves a damping factor for the calculation of the PageRank. It is like the income tax which the govt extracts from one despite paying him itself.

Implementing the PageRank algorithm for information retrieval involves simulating how web pages or documents in a collection are ranked based on their importance. In an information retrieval context, PageRank can be adapted to rank documents rather than web pages. Below is a simplified Python implementation of PageRank for document ranking:

```
import numpy as np


# Define the document collection (replace this with your own document collection)
documents = {
    'doc1': 'This is the first document',
    'doc2': 'This document is the second document',
    'doc3': 'And this is the third one',
    'doc4': 'Is this the first document',
}


# Tokenize and preprocess the documents
document_tokens = {doc_id: document.lower().split() for doc_id, document in documents.items()}


# Create a vocabulary of unique terms
```

```
all_terms = set(term for terms in document_tokens.values() for term
in terms)
term_index = {term: idx for idx, term in enumerate(all_terms)}
num_terms = len(all_terms)

# Create the document-term matrix
document_term_matrix = np.zeros((len(documents), num_terms))
for i, doc_id in enumerate(documents.keys()):
    for term in document_tokens[doc_id]:
        term_id = term_index[term]
        document_term_matrix[i, term_id] += 1

# Normalize the document-term matrix
row_sums = document_term_matrix.sum(axis=1)
document_term_matrix /= row_sums[:, np.newaxis]

# Initialize PageRank scores
num_documents = len(documents)
pagerank_scores = np.ones(num_documents) / num_documents

# PageRank iterations
damping_factor = 0.85
num_iterations = 20
for _ in range(num_iterations):
```

```
    new_scores = np.zeros(num_documents)
    for i in range(num_documents):
        for j in range(num_documents):
            if i != j:
                new_scores[i] += damping_factor *
(document_term_matrix[j, i] / row_sums[j]) * pagerank_scores[j]
        new_scores[i] += (1 - damping_factor) / num_documents
    pagerank_scores = new_scores

# Display PageRank scores for documents
for doc_id, score in zip(documents.keys(), pagerank_scores):
    print(f"{doc_id}: PageRank Score = {score:.4f}")
```

Conclusion:-

we have to implemented the PageRank algorithm for information retrieval involves a complex process of ranking documents based on their importance and relevance to a given query or context.

Performance & Understanding	Innovation	Timely Completion	Total	Sign & Date
3	1	1	5	