Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

Department of Computer Engineering

Name of the Student: SHASHWAT SHAH **Roll Number: O201**

SAP ID: 60004220126 **Class: C2** **Division:C2** **Batch:C22**

Subject: Web Intelligence

DATE OF PERFORMANCE: 06/03/2025 **DATE OF SUBMISION:_____**

EXPERIMENT NO: 05

AIM: Implementation of HITS (Hyperlink Induced Topic Search) algorithm (CO2)

SOFTWARE/IDE USED: Google Colab/Jupyter Notebook

THEORY:

1. What is the main purpose of the HITS algorithm, and how does it differ from PageRank?

➔ The HITS (Hyperlink-Induced Topic Search) algorithm identifies two types of web pages: hubs (pages linking to many others) and authorities (pages that are linked to by many hubs). It focuses on local, topic-specific networks and computes two scores: hub score and authority score for each page.

PageRank, on the other hand, evaluates web pages based on the number and quality of incoming links, treating them as endorsements. It operates on a global scale, assigning a single score to each page.

HITS: Focuses on hubs and authorities, works on local networks, computes two scores.

PageRank: Focuses on overall link quality and quantity, works globally, computes a single score.

2. What are hub and authority scores, and how are they calculated in the HITS algorithm?

➔ In the HITS algorithm, hub and authority scores are used to rank web pages based on their relationships with other pages.

Hub score: Measures how good a page is at linking to other authoritative pages. High hub scores mean the page is a good source of links to important content.

Authority score: Measures how good a page is at being linked to by many hubs. High authority scores mean the page is an important destination for relevant information.

Calculation:

Hub score: A page's hub score is the sum of the authority scores of the pages it links to.

Authority score: A page's authority score is the sum of the hub scores of the pages that link to it.

These scores are calculated iteratively by updating the hub and authority values until they converge.

3. Why does the HITS algorithm require an initial set of seed pages before execution?

➔ The HITS algorithm requires an initial set of seed pages to focus the analysis on a specific topic or subgraph of the web. These seed pages act as the starting point for the algorithm to identify relevant hubs and authorities within that specific topic. Without these seeds, the algorithm

would analyze the entire web, making it less efficient and less focused on a particular area of interest. The seed pages help guide the algorithm toward a relevant subset of pages, improving the quality and relevance of the hub and authority scores.

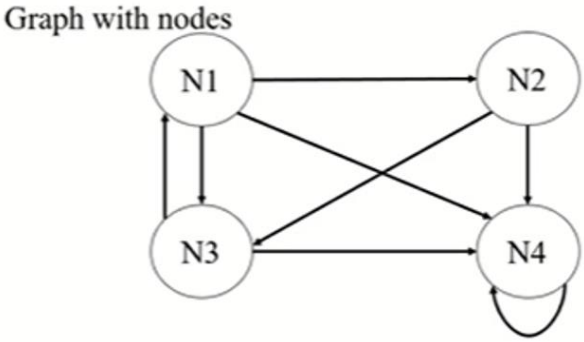
4. How does the algorithm update hub and authority scores during each iteration?

➔ In the HITS algorithm, hub and authority scores are updated iteratively as follows:
Update authority scores: Each page's authority score is the sum of the hub scores of all pages that link to it.
Update hub scores: Each page's hub score is the sum of the authority scores of all the pages it links to.
The process is repeated, with scores being recalculated in each iteration, until the scores converge (i.e., they stabilize and stop changing significantly).

5. What are the major limitations of the HITS algorithm, and how can they be mitigated?

➔ The major limitations of the HITS algorithm are:
Sensitive to noise and irrelevant links: HITS can assign high scores to pages with many links but low quality.
Mitigation: Use link pruning or incorporate quality-based filtering to focus on high-value links.
Initialization bias: The algorithm relies on the initial seed pages, which can influence results.
Mitigation: Use a diverse and representative set of seed pages to minimize bias.
Computationally expensive: Iterative calculations for large datasets can be time-consuming.
Mitigation: Use scalable algorithms or approximation techniques to reduce computation time.
Topic-specific: HITS is designed for local, topic-based analysis, which may not generalize well across the entire web.
Mitigation: Combine HITS with other algorithms, like PageRank, for broader analysis.

6. Identify the best hub and authority score for the given graph. Calculate the hub and authority score using HITS algorithm for *k* = 3 iterations.



IMPLEMENTATION:

Perform implementation for the same using Python programming.

Code:

```
import numpy as np


class HITS:
    def __init__(self, graph, max_iter=3, tol=1e-6):
        # graph is represented as an adjacency matrix where graph[i][j]
= 1 if there is a link from i to j, 0 otherwise
```

```
        self.graph = np.array(graph)
        self.max_iter = max_iter
        self.tol = tol
        self.n = self.graph.shape[0]  # Number of nodes

        # Initialize hub and authority scores to 1
        self.hub_scores = np.ones(self.n)
        self.authority_scores = np.ones(self.n)

    def update_scores(self):
        # Update authority scores: A = G^T * H
        new_authority_scores = np.dot(self.graph.T, self.hub_scores)

        # Update hub scores: H = G * A
        new_hub_scores = np.dot(self.graph, self.authority_scores)

        return new_hub_scores, new_authority_scores

    def normalize_scores(self, hub_scores, authority_scores):
        # Normalize the scores to prevent overflow
        hub_norm = np.linalg.norm(hub_scores, 2)
        authority_norm = np.linalg.norm(authority_scores, 2)

        hub_scores /= hub_norm
        authority_scores /= authority_norm

        return hub_scores, authority_scores

    def run(self):
        for iteration in range(self.max_iter):
            new_hub_scores, new_authority_scores = self.update_scores()
            new_hub_scores, new_authority_scores =
self.normalize_scores(new_hub_scores, new_authority_scores)

            # Check for convergence
            hub_diff = np.linalg.norm(new_hub_scores - self.hub_scores,
2)
            authority_diff = np.linalg.norm(new_authority_scores -
self.authority_scores, 2)

            if hub_diff < self.tol and authority_diff < self.tol:
                print(f"Converged after {iteration + 1} iterations")
                break

            # Update scores for next iteration
            self.hub_scores = new_hub_scores
            self.authority_scores = new_authority_scores
        else:
            print(f"Reached max iterations ({self.max_iter}) without
convergence")

    def get_hub_scores(self):
        return self.hub_scores
```

```
    def get_authority_scores(self):
        return self.authority_scores

# Adjacency matrix representation of a graph (1 means there is a link)
graph = [
    [0, 1, 1, 1],
    [0, 0, 1, 1],
    [1, 0, 0, 1],
    [0, 0, 0, 1]
]

hits = HITS(graph)
hits.run()

print("Hub Scores:", hits.get_hub_scores())
print("Authority Scores:", hits.get_authority_scores())
```

Output:

```
Reached max iterations (3) without convergence
Hub Scores: [0.65926851 0.53565567 0.41204282 0.32963426]
Authority Scores: [0.18543974 0.25961563 0.48214331 0.81593484]
```

CONCLUSION: Your inferences and learnings from this experiment to be summarized in 3-4 to lines.

POST-EXPERIMENTAL EXERCISE:

1. How did the hub and authority scores evolve over multiple iterations, and did they converge?

➔ Over multiple iterations in the HITS algorithm, the hub and authority scores evolve as follows:
Hub scores increase for pages that link to important authoritative pages, and decrease for those linking to less important ones.
Authority scores increase for pages that are linked to by high-scoring hubs and decrease for those with fewer or lower-scoring links.
The scores converge when the changes between iterations become very small, meaning the hub and authority scores stabilize and no longer significantly change. This convergence indicates that the algorithm has identified a stable ranking of hubs and authorities.

2. How sensitive is the HITS algorithm to changes in the link structure of the web graph?

➔ The HITS algorithm is sensitive to changes in the web graph's link structure. Small changes in the links, such as adding or removing links between pages, can significantly alter the hub and authority scores. This sensitivity occurs because the scores are directly dependent on the relationships between pages, and even minor changes in those relationships can cause the scores to shift, especially in the early iterations before convergence.

3. How did the choice of the initial seed set impact the final ranking of pages?

➔ The choice of the initial seed set in the HITS algorithm can significantly impact the final ranking of pages. If the seed set is not representative or focused on a specific topic, it can lead to biased or irrelevant hub and authority scores. A well-chosen seed set helps the algorithm focus on the relevant subset of pages, while a poor seed set can skew the results, making the final ranking less accurate or meaningful.

4. What were the computational costs of running HITS on a large dataset, and how could efficiency be improved?

Faculty In-charge: Mr. Vivian Lobo

➔ The computational costs of running HITS on a large dataset are high due to the need for iterative updates of hub and authority scores for all pages in the web graph, which can be time-consuming for large networks.

To improve efficiency, the following methods can be used:

Parallel processing to speed up calculations.

Link pruning to remove less relevant links, reducing the graph size.

Approximation techniques to simplify computations.

Using sparse matrices to store link structures more efficiently.

5. What improvements or modifications could be made to enhance the accuracy or efficiency of the HITS algorithm?

➔ To enhance the accuracy and efficiency of the HITS algorithm, the following improvements can be made:

Incorporate quality-based link filtering to reduce the influence of irrelevant or low-quality links, improving accuracy.

Use better initialization of the seed set to ensure more relevant and diverse starting points.

Implement parallel or distributed computing to speed up the iterative process on large datasets.

Apply link pruning or dimensionality reduction to remove unnecessary or redundant links, improving efficiency.

Combine HITS with other algorithms (e.g., PageRank) to increase robustness and accuracy in ranking.