LAB EXPERIMENT NO. 09

NAME: PARTH PAREKH SAP ID:60004200006

BATCH:A1 BRANCH:COMPUTER ENGG.

Aim:

To Implement HITS algorithm.

Theory:

HITS Algorithm

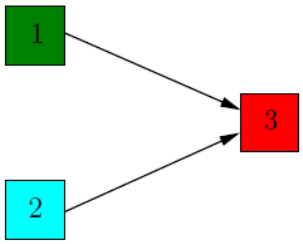
Hyperlink Induced Topic Search (HITS) is an algorithm used in link analysis. It could discover and rank the webpages relevant for a particular search. The idea of this algorithm originated from the fact that an ideal website should link to other relevant sites and also being linked by other important sites. HITS uses hubs and authorities to define a recursive relationship between webpages.

- Given a query to a Search Engine, the set of highly relevant web pages are called Roots. They are potential Authorities.
- Pages that are not very relevant but point to pages in the Root are called Hubs. Thus, an Authority is a page that many hubs link to whereas a Hub is a page that links to many authorities.

HITS algorithm is in the same spirit as PageRank. They both make use of the link structure of the Web graph in order to decide the relevance of the pages. The difference is that unlike the PageRank algorithm, HITS only operates on a small subgraph (the seed SQ) from the web graph. This subgraph is query dependent; whenever we search with a different query phrase, the seed changes as well. HITS ranks the seed nodes according to their authority and hub weights. The highest-ranking pages are displayed to the user by the query engine.

Algorithm

1. Initialize the hub and authority of each node with a value of 1
2. For each iteration, update the hub and authority of every node in the graph
3. The new authority is the sum of the hub of its parents
4. The new hub is the sum of the authority of its children
5. Normalize the new authority and hub



authority depends on how many nodes link to it, hub depends on how many nodes it links to.

The adjacency matrix of the graph is $A = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$, with transpose $A^t = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 0 \end{bmatrix}$. Assume the initial hub weight vector is: $u = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$.

We compute the authority weight vector by:

$$v = A^t \cdot u = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 2 \end{bmatrix}$$

Then, the updated hub weight is:

$$u = A \cdot v = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 0 \\ 2 \end{bmatrix} = \begin{bmatrix} 2 \\ 2 \\ 0 \end{bmatrix}$$

Code:

```
import math
import numpy as np
from tabulate import tabulate

count = 0
myTable = []

def input():
    outgoing_matrix = [[0, 1, 1, 1], [0, 0, 1, 1], [1, 0, 0, 1], [0, 0, 0,
 1]]
    k = 2
    return outgoing_matrix, k

def k_degree_0(matrix, u, v):
    out_degree = []
    in_degree = []

    out_count = 0
    in_count = 0
    for row in matrix:
        for elem in row:
            if(elem == 1):
                out_count += 1
        out_degree.append(out_count)
        out_count = 0

    for i in range(len(matrix)):
        for j in range(len(matrix)):
            if(matrix[j][i] == 1):
                in_count += 1
        in_degree.append(in_count)
        in_count = 0

    matrix_transpose = np.copy(matrix)
    matrix_transpose = np.transpose(matrix_transpose)
    v = np.matmul(matrix_transpose, u)
    u = np.matmul(matrix, v)
    myTable.append([count, v, u])
    return u, v

def sum_of_sq(arr):
    sum = 0
    for elem in arr:
        sum += elem ** 2
```

```
    return math.sqrt(sum)

def k_degree_n(matrix, u, v, k, count):
    if(k == 0):
        return u, v

    ssq_v = sum_of_sq(v)
    v_ = []
    for i in range(len(v)):
        v_.append(round(v[i] / ssq_v, 3))

    ssq_u = sum_of_sq(u)
    u_ = []
    for i in range(len(u)):
        u_.append(round(u[i] / ssq_u, 3))
    myTable.append([count, v_, u_])
    return k_degree_n(matrix, u_, v_, k-1, count+1)

def u_v_init(matrix, u, v):
    for row in matrix:
        u.append(1)
        v.append(0)
    return u, v

def main():
    u = []
    v = []

    matrix, k = input()
    u, v = u_v_init(matrix, u, v)
    u, v = k_degree_0(matrix, u, v)
    u, v = k_degree_n(matrix, u, v, k, count + 1)
    print(tabulate(myTable, headers=['k', 'v', 'u']))

if __name__ == "__main__":
    main()
```

Output

k	v	u
0	[1 1 2 4]	[7 6 5 4]
1	[0.213, 0.213, 0.426, 0.853]	[0.624, 0.535, 0.445, 0.356]
2	[0.213, 0.213, 0.426, 0.853]	[0.624, 0.535, 0.445, 0.356]

Conclusion:

Thus, we have successfully implemented HITS algorithm. It could discover and rank the webpages relevant for a particular search. The idea of this algorithm originated from the fact that an ideal website should link to other relevant sites and also being linked by other important sites. It uses hubs and authorities to define a recursive relationship between webpages.