Name: Jigar Siddhpura **SAPID:** 60004210155

DIV: C/C2 **Branch:** Computer Engineering

DMW - Experiment 9

Jigar Siddhpura
60004210155

DMW Experiment 9

Aim: To implement HITS algorithm

Theory:

1. Hyperlink induced Topic Search (HITS) Algorithm is a link analysis algo that rates web pages.
2. It is used for web links structures to discover & rank the web pages relevant to a particular search.
3. HITS uses hubs & authorities to define a recursive relationship btw web pages.
4. Before understanding HITS algo, we need to know about hubs & authorities.
5. Given a query to a search engine, the set of highly relevant pages are called Roots. They are potential authorities.
6. Pages that are relevant but point to pages in roots are called hubs.
7. Thus, authority is a page that many hubs links to whereas Hub is a page that links to many authorities.

Algorithm:

1. Let the no. of iteration be 'k'.
2. Each node is assigned a hub score = 1 & an authority score = 1
3. Repeat k times: Hub update:
   Each node's hub score = Σ (Authority score of each node it points to)

Authority update : Each node's authority score
= $\sum$ (Hub score of each node pointing it)

Normalize the hub score by dividing it by square root of sum of squares of all hub scores & similarly for authority score.

Conclusion :

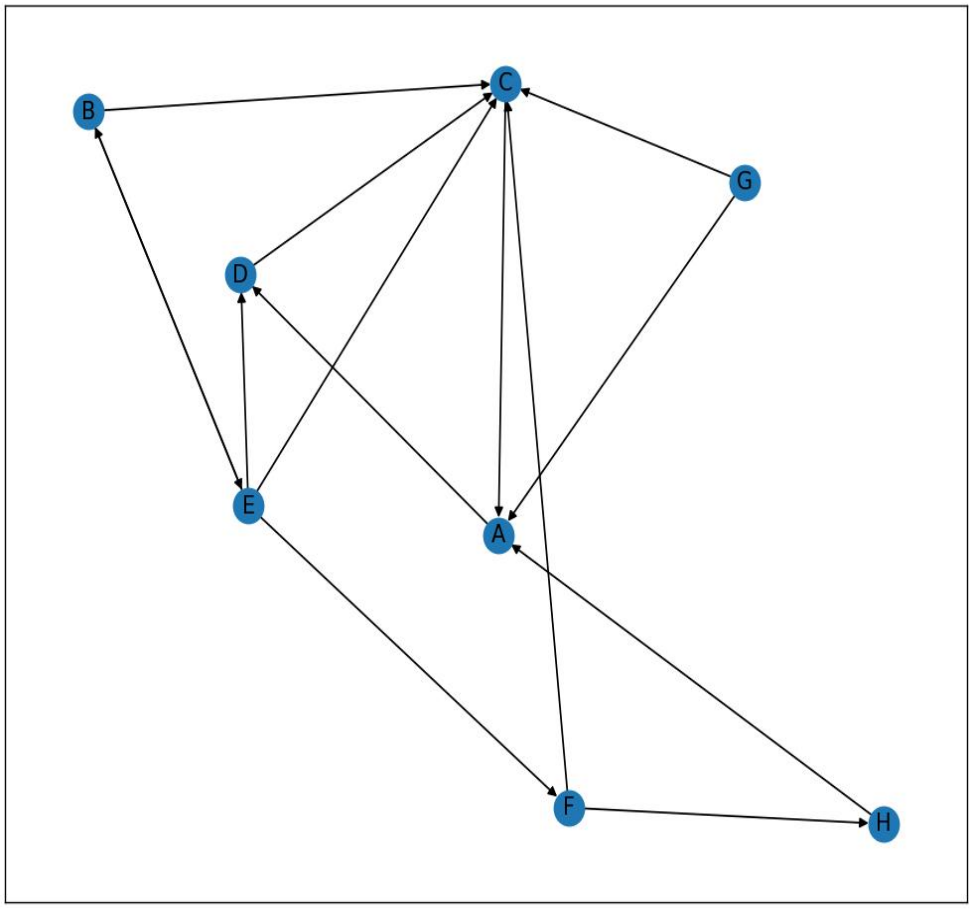
Hence, we implemented this algo in python with the help of networkx library & implemented HITS algo on graph given.

After implementation, we get highest hub score (0.2588) to node E indicating good Hub ; Node B & F also have relatively high hub scores ; Node A, C, D, G, H have lower hub score. Now, node C has highest authority score (0.3884) means a good authority. Node D, B, F have relatively high authority score & Node G has low score. So we can tell Node E as central Hub & Node C as central authority where as B & F as both hubs & authority.

Code:

```
import networkx as nx
import matplotlib.pyplot as plt
import numpy as np

graph_matrix = np.array([
[0, 0, 0, 1, 0, 0, 0, 0], # A -> D
[0, 0, 1, 0, 1, 0, 0, 0], # B -> E, C
[1, 0, 0, 0, 0, 0, 0, 0], # C -> A
[0, 0, 1, 0, 0, 0, 0, 0], # D -> C
[0, 1, 1, 1, 0, 1, 0, 0], # E -> B, C, D, F
[0, 0, 1, 0, 0, 0, 0, 1], # F -> C, H
[1, 0, 1, 0, 0, 0, 0, 0], # G -> A, C
[1, 0, 0, 0, 0, 0, 0, 0], # H -> A
])

G = nx.DiGraph()
labels = {}

for i in range(len(graph_matrix)):
    node_label = chr(ord('A') + i)
    labels[i] = node_label
    G.add_node(i, label=node_label)
    for j in range(len(graph_matrix[i])):
        if graph_matrix[i][j] == 1:
            G.add_edge(i, j)

plt.figure(figsize=(10, 10))
pos = nx.spring_layout(G)
nx.draw_networkx(G, pos=pos, with_labels=True, labels=labels)
hubs, authorities = nx.hits(G, max_iter=50, normalized=True)

print("Hub Scores:")
for key, value in hubs.items():
    print(f'{labels[key]}: {value}')
print()

print("Authority Scores:"

for key, value in authorities.items():
    print(f'{labels[key]}: {value}')
plt.show()
```

Graph:



Output:

```
Hub Scores:
A: 0.04642540403219996
D: 0.13366037526115382
B: 0.15763599442967324
C: 0.0373891322464265
E: 0.2588144598468665
F: 0.1576359944296732
H: 0.0373891322464265
G: 0.1710495075075803

Authority Scores:
A: 0.10864044011724333
D: 0.13489685434358004
B: 0.114379740733364446
C: 0.38837280038761807
E: 0.06966521184241486
F: 0.11437974073336447
H: 0.06966521184241474
G: -0.0
PS D:\SEM-5\DMW\EXPERIMENTS>
```