

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Programme | : | **B.Tech** | Semester | : | **Win Sem 21-22** |
| Course | : | **Web Mining** | Code | : | **CSE3024** |
| Faculty | : | **Dr.Bhuvaneswari A** | Slot | : | **L7+L8** |
| Date | : | **01-03-2022** | Marks | : | **10 Points** |

**Name: Dimish Pamnani**

**Register No: 19BCE1407**

**Submission Date:4 Mar. 22**

**Exercise 7: HUBS AND AUTHORITY USING HITS ALGORITHM**

**GOOGLE COLAB LINK:**

[**https://colab.research.google.com/drive/1O666lIXu1ZlnxHRlcMrlh7MKT4sxaF3p?usp=sharing**](https://colab.research.google.com/drive/1O666lIXu1ZlnxHRlcMrlh7MKT4sxaF3p?usp=sharing)

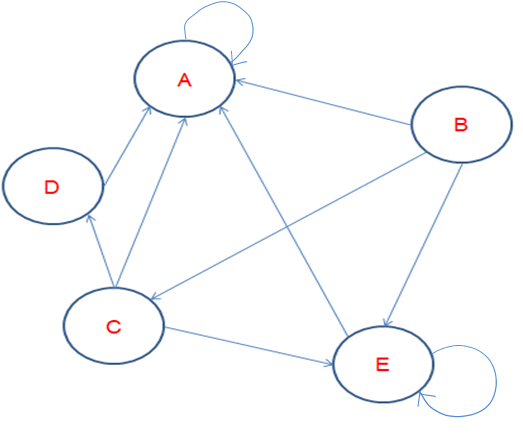
**PROCEDURE:**

1. Import numpy as np and create function which returns hub score and authorities.
2. Input the size of the matrix (adjacency matrix) or the number of nodes present in the graph
3. Input the elements of adjacency matrix using for loop
4. Set array of value 1 of size as length(outlink\_shape)
5. Compute the authorities and hub scores values using the formula *of all Authority scores. (optional)*

* Hub update : Each node’s Hub score = \Sigma (Authority score of each node it points to).
* Authority update : Each node’s Authority score = \Sigma (Hub score of each node pointing to it).

Normalize the scores by dividing each Hub score by square root of the sum of the squares of all Hub scores, and dividing each Authority score by square root of the sum of the squares of all Authority scores. (optional)

Compute the Hubs and authority weights for the following graph. Verify the consistency of results obtained with the Custom Implementation and inbuilt HITS Algorithm function using k=4.



**INBUILT HITS ALGORITHM FUNCTION:**

**CODE:**

# importing modules

import networkx as nx

import matplotlib.pyplot as plt

G = nx.DiGraph()

G.add\_edges\_from([('A', 'A'), ('B', 'A'), ('B', 'E'), ('B','C'), ('E', 'E'), ('E', 'A'), ('C', 'A'), ('C','D'),('C','E'),('D', 'A')])

plt.figure(figsize =(10, 10))

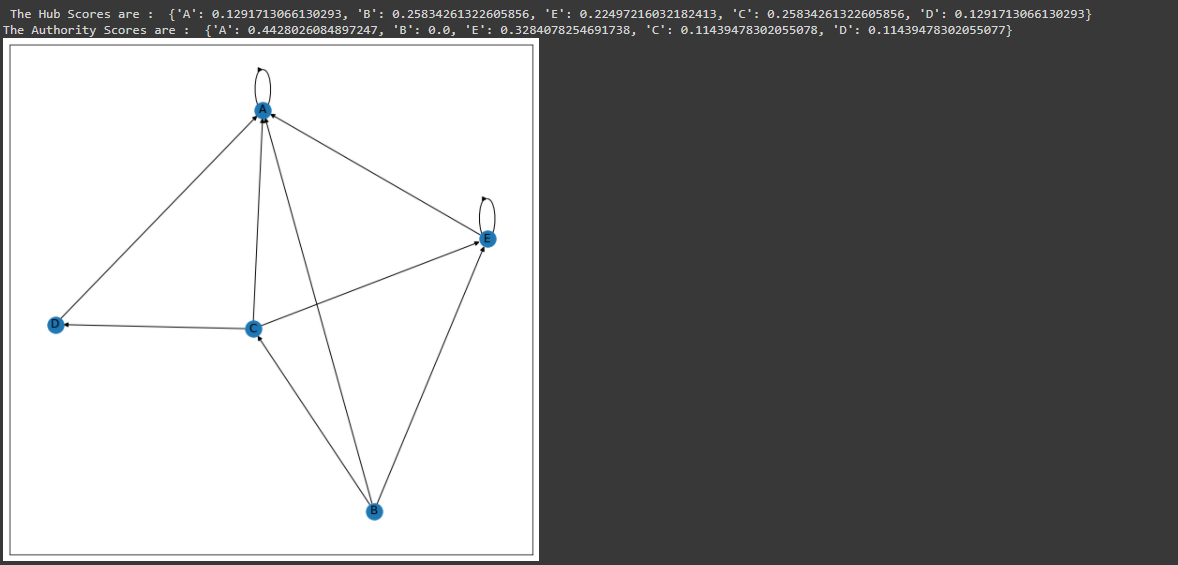
nx.draw\_networkx(G, with\_labels = True)

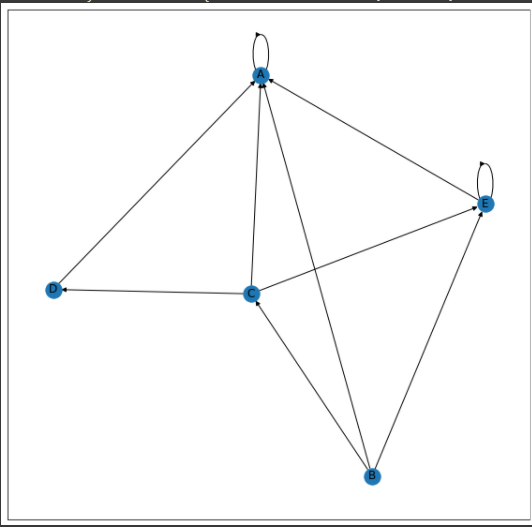
hubs, authorities = nx.hits(G, max\_iter = 50, normalized = True)

print(" The Hub Scores are : ", hubs)

print("The Authority Scores are : ", authorities)

**OUTPUT:**





**CUSTOM HITS ALGORITHM FUNCTION:**

**CODE:**

import numpy as np

# Function to calculate the authority and hub score of the nodes

def authority\_hub\_score(outlinks):

  # size of the matrix

  size = outlinks.shape[0]

  # Initializing the lists

  hub\_scores = [1.0 for i in range(size)]

  authority\_scores = [1.0 for i in range(size)]

  # Printing initial Hub scores

  print(hub\_scores)

  for \_ in range(100):

    # Calculating the authority scores of the nodes

    for j in range(size):

      temp\_auth = 0.0

      for i in range(size):

        if outlinks[i][j] == 1:

          temp\_auth += hub\_scores[i]

      authority\_scores[j] = temp\_auth

    # Normalizing the authority scores

    auth\_sum = sum(authority\_scores)

    # print(auth\_sum)

    for i in range(len(authority\_scores)):

      authority\_scores[i] /= auth\_sum

    # Calculating the hub scores of the nodes

    for i in range(size):

      temp\_hub = 0.0

      for j in range(size):

        if outlinks[i][j] == 1:

          temp\_hub += authority\_scores[j]

      hub\_scores[i] = temp\_hub

    # Normalizing the hub scores

    hub\_sum = sum(hub\_scores)

    # print(hub\_sum)

    for i in range(len(hub\_scores)):

      hub\_scores[i] /= hub\_sum

  return authority\_scores, hub\_scores

n = int(input('Enter the size of the matrix:\t'))

outlinks = []

for i in range(n\*n):

  temp = int(input('Enter the element:\t'))

  outlinks.append(temp)

outlinks = np.reshape(outlinks, (n, n))

authority\_scores, hub\_scores = authority\_hub\_score(outlinks)

print("Authority Scores:")

for i in (authority\_scores):

  print(round(i, 4))

print("Hub Scores:")

for i in (hub\_scores):

  print(round(i, 4))

**OUTPUT:**

s