| | |
|---|---|
| Name: | Prerna Sunil Jadhav |
| Sap Id: | 60004220127 |
| Class: | T. Y. B.Tech (Computer Engineering) |
| Course: | Data Mining and Warehouse Laboratory |
| Course Code: | DJ19CEL501 |
| Experiment No.: | 09 |

**AIM:** Implementation of HITS Algorithm

CODE:

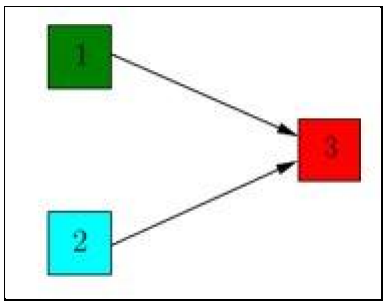
```
from math import sqrt
def hits_algorithm(num_nodes, graph, iterations):
    authority_scores = dict()
    hub_scores = dict()
for i in
range(len(graph)):
authority_scores[i] =
1
        hub_scores[i] =
    1 incoming_nodes
    = dict() for i in
    range(len(graph))
    :
        temp=[]
        for node in
            graph: if
            node[i]:
                temp.append(node
        ) incoming_nodes[i] =
        temp
    outgoing_nodes = dict()
    for i,node in
        enumerate(graph):
        temp = []
        for j,edge in
            enumerate(node): if
            edge:
                temp.append(
        graph[j])
        outgoing_nodes[i] =
        temp
    print()
    for k in range(iterations):
        print('Iteration : ',k+1)
```

```
        print('Authority Score')
        normalization_value = 0
        for i,node in enumerate(graph):
            authority_scores[i]=0
            for j,other_node in
                enumerate(graph): if
                other_node in
                incoming_nodes[i]:
                    authority_scores[i] += hub_scores[j]
            normalization_value +=
            (authority_scores[i]**2)
        normalization_value =
        sqrt(normalization_value) for i in
        range(num_nodes):
            authority_scores[i] /= normalization_value
            print('{} :{:.2f}'.format(chr(65+i),authority_scores[i]),end=' | ')
        print()
        print('Hub Score')
        normalization_value = 0
        for i,node in enumerate(graph):
            hub_scores[i]=0
            for j,other_node in
                enumerate(graph): if
                other_node in
                outgoing_nodes[i]:
                    hub_scores[i] += authority_scores[j]
            normalization_value += (hub_scores[i]**2)
    normalization_value = sqrt(normalization_value)
    for i in range(num_nodes):
        hub_scores[i] /= normalization_value
        print('{} :{:.2f}'.format(chr(65+i),hub_scores[i]),end=' | ')
    print("\n\n")

def main():
    n = int(input('Enter the no of nodes : '))
    graph = []
    print('Enter Adjacency Matrix :
    ') for i in range(n):
        temp = input()
        temp_list = temp.split(' ')
        graph.append(list(map(int,temp_list)
        ))
    k = int(input('Enter No of Iterations to be performed : '))
    hits_algorithm(n, graph, k)
```

main()

GRAPH:




OUTPUT:

Enter the no of nodes :
3 Enter Adjacency
Matrix : 0 0 1
0 0 1
0 0 0
Enter No of Iterations to be performed : 3

Iteration : 1
Authority
Score
A :0.00 | B :0.00 | C :1.00 |
Hub Score
A :0.71 | B :0.71 | C :0.00 |

Iteration : 2
Authority
Score
A :0.00 | B :0.00 | C :1.00 |
Hub Score
A :0.71 | B :0.71 | C :0.00 |

Iteration : 3
Authority
Score
A :0.00 | B :0.00 | C :1.00 |
Hub Score
A :0.71 | B :0.71 | C :0.00 |