**Implementation of Page Rank/HITS algorithm**

import numpy as np

def page\_rank(links, damping=0.85, max\_iter=100, tol=1e-6):

n = len(links)

M = np.zeros((n, n))

for i in range(n):

if len(links[i]) > 0:

for j in links[i]:

M[j][i] = 1 / len(links[i])

rank = np.ones(n) / n

for \_ in range(max\_iter):

new\_rank = (1 - damping) / n + damping \* M.dot(rank)

if np.linalg.norm(new\_rank - rank) < tol:

break

rank = new\_rank

return rank

links = [

[1, 2],

[2],

[0, 3],

[2]

]

page\_ranks = page\_rank(links)

print("PageRank Results:")

for i, pr in enumerate(page\_ranks):

print(f"Page {i}: {pr:.4f}")

def hits\_algorithm(links, max\_iter=100, tol=1e-6):

n = len(links)

hub = np.ones(n)

authority = np.ones(n)

adj\_matrix = np.zeros((n, n))

for i in range(n):

for j in links[i]:

adj\_matrix[i][j] = 1

for \_ in range(max\_iter):

new\_authority = adj\_matrix.T.dot(hub)

new\_hub = adj\_matrix.dot(new\_authority)

new\_authority = new\_authority / np.linalg.norm(new\_authority)

new\_hub = new\_hub / np.linalg.norm(new\_hub)

if (np.linalg.norm(new\_authority - authority) < tol and

np.linalg.norm(new\_hub - hub) < tol):

break

authority = new\_authority

hub = new\_hub

return hub, authority

hub\_scores, authority\_scores = hits\_algorithm(links)

print("\nHITS Algorithm Results:")

print("Hub Scores:")

for i, score in enumerate(hub\_scores):

print(f"Page {i}: {score:.4f}")

print("\nAuthority Scores:")

for i, score in enumerate(authority\_scores):

print(f"Page {i}: {score:.4f}")

print("\nPageRank vs HITS Comparison:")

print("Page | PageRank | Hub Score | Authority Score")

for i in range(len(links)):

print(f"{i:4} | {page\_ranks[i]:8.4f} | {hub\_scores[i]:9.4f} | {authority\_scores[i]:13.4f}")