Bhuvi Ghosh
60009210191

SNA Experiment-6

Step 1: The given network of pages:

- **Page A** links to **Page B** and **Page C**.
- **Page B** does not link to any page.
- **Page C** links back to **Page A**.

Step 2: Initial Conditions

- **PageRank (PR)** for all pages is initially 1.
- **Damping factor (d)** is 0.85, meaning that the surfer will follow links with probability 0.85 and will randomly jump to a page with probability 0.15 (this is known as the "random jump").

We will use the **PageRank formula** to update the rank of each page.

Step 3: PageRank Formula

The formula for updating the PageRank for a given page PPP is:

PR(P)=1−dN+d∑i∈In(P)PR(i)L(i)PR(P) = \frac{1 - d}{N} + d \sum_{i \in \text{In}(P)} \frac{PR(i)}{L(i)}PR(P)=N1−d+di∈In(P)∑L(i)PR(i)

Where:

- NNN is the total number of pages in the network (3 in our case: A, B, and C).
- ddd is the damping factor (0.85).
- In(P)\text{In}(P)In(P) represents the set of pages linking to page PPP.
- L(i)L(i)L(i) is the number of outgoing links from page iii.

Step 4: Setting Up for Iteration

The random surfer model will converge iteratively, so we will perform multiple iterations until the PageRank values stabilize. We will iterate for a couple of steps to show the process.

Step 5: First Iteration (Using the Initial PageRank)

Initial PageRank:

- PR(A) = 1
- PR(B) = 1
- PR(C) = 1

PageRank Calculation for Each Page:

1. **For Page A**:
   - **Incoming Links to A**: Page C links to A.
   - The formula becomes:

PR(A)=1−0.853+0.85×PR(C)1=0.153+0.85×1=0.05+0.85=0.9PR(A) = \frac{1 - 0.85}{3} + 0.85 \times \frac{PR(C)}{1} = \frac{0.15}{3} + 0.85 \times 1 = 0.05 + 0.85 = 0.9PR(A)=31−0.85+0.85×1PR(C)=30.15+0.85×1=0.05+0.85=0.9

2. **For Page B**:
   - **Incoming Links to B**: Page A links to B.
   - The formula becomes:

     PR(B)=1−0.853+0.85×PR(A)2=0.153+0.85×12=0.05+0.425=0.475PR(B) = \frac{1 - 0.85}{3} + 0.85 \times \frac{PR(A)}{2} = \frac{0.15}{3} + 0.85 \times \frac{1}{2} = 0.05 + 0.425 = 0.475PR(B)=31−0.85+0.85×2PR(A)=30.15+0.85×21=0.05+0.425=0.475

3. **For Page C**:
   - **Incoming Links to C**: Page A links to C.
   - The formula becomes:

     PR(C)=1−0.853+0.85×PR(A)2=0.153+0.85×12=0.05+0.425=0.475PR(C) = \frac{1 - 0.85}{3} + 0.85 \times \frac{PR(A)}{2} = \frac{0.15}{3} + 0.85 \times \frac{1}{2} = 0.05 + 0.425 = 0.475PR(C)=31−0.85+0.85×2PR(A)=30.15+0.85×21=0.05+0.425=0.475

So after the first iteration, the PageRank values are:

- PR(A) = 0.9
- PR(B) = 0.475
- PR(C) = 0.475

Step 6: Second Iteration (Using Updated PageRank)

1. **For Page A**:
   - **Incoming Links to A**: Page C links to A.
   - The formula becomes:

     PR(A)=1−0.853+0.85×PR(C)1=0.153+0.85×0.475=0.05+0.40375=0.45375PR(A) = \frac{1 - 0.85}{3} + 0.85 \times \frac{PR(C)}{1} = \frac{0.15}{3} + 0.85 \times 0.475 = 0.05 + 0.40375 = 0.45375PR(A)=31−0.85+0.85×1PR(C)=30.15+0.85×0.475=0.05+0.40375=0.45375

2. **For Page B**:
   - **Incoming Links to B**: Page A links to B.
   - The formula becomes:

     PR(B)=1−0.853+0.85×PR(A)2=0.153+0.85×0.92=0.05+0.3825=0.4325PR(B) = \frac{1 - 0.85}{3} + 0.85 \times \frac{PR(A)}{2} = \frac{0.15}{3} + 0.85 \times \frac{0.9}{2} = 0.05 + 0.3825 = 0.4325PR(B)=31−0.85+0.85×2PR(A)=30.15+0.85×20.9=0.05+0.3825=0.4325

3. **For Page C**:
   - **Incoming Links to C**: Page A links to C.
   - The formula becomes:

     PR(C)=1−0.853+0.85×PR(A)2=0.153+0.85×0.92=0.05+0.3825=0.4325PR(C) = \frac{1 - 0.85}{3} + 0.85 \times \frac{PR(A)}{2} = \frac{0.15}{3} + 0.85 \times \frac{0.9}{2} = 0.05 + 0.3825 = 0.4325PR(C)=31−0.85+0.85×2PR(A)=30.15+0.85×20.9=0.05+0.3825=0.4325

After the second iteration, the PageRank values are:

- PR(A) = 0.45375
- PR(B) = 0.4325
- PR(C) = 0.4325

Step 7: Convergence Check

If the values are close to each other (as they are in this case), we can stop iterating. At this point, we can conclude the PageRank values have converged.

Final PageRank:

After a few iterations, the final converged PageRank values are approximately:

- **PR(A)** ≈ 0.45375
- **PR(B)** ≈ 0.4325
- **PR(C)** ≈ 0.4325

0s

```python
import numpy as np

def pagerank(links, damping_factor=0.85, max_iterations=5, tol=1.0e-6):
    pages = list(links.keys())
    N = len(pages)

    # Initialize PageRank values
    PR = {page: 1.0 for page in pages}

    # Construct adjacency matrix
    M = np.zeros((N, N))

    for i, page in enumerate(pages):
        if links[page]:
            for linked_page in links[page]:
                j = pages.index(linked_page)
                M[j, i] = 1 / len(links[page])

    # Iteratively calculate PageRank
    for iteration in range(max_iterations):
        new_PR = {}
        for i, page in enumerate(pages):
            sum_rank = sum(M[i, j] * PR[pages[j]] for j in range(N))
            new_PR[page] = (1 - damping_factor) / N + damping_factor * sum_rank

        # Print PageRank for this iteration
        print(f'Iteration {iteration + 1}:')
        for page, rank in new_PR.items():
            print(f'  PR({page}) = {rank:.5f}')
        print()

        # Check for convergence
        if all(abs(new_PR[page] - PR[page]) < tol for page in pages):
            break

        PR = new_PR

    return PR
```

0s

```python
# Define the network
links = {
    'A': ['B', 'C'],
    'B': [],
    'C': ['A']
}

# Compute PageRank
page_ranks = pagerank(links)

# Print final PageRank values
print("Final PageRank values:")
for page, rank in page_ranks.items():
    print(f'PR({page}) = {rank:.5f}')
```

```
Iteration 1:
  PR(A) = 0.90000
  PR(B) = 0.47500
  PR(C) = 0.47500

Iteration 2:
  PR(A) = 0.45375
  PR(B) = 0.43250
  PR(C) = 0.43250

Iteration 3:
  PR(A) = 0.41762
  PR(B) = 0.24284
  PR(C) = 0.24284

Iteration 4:
  PR(A) = 0.25642
  PR(B) = 0.22749
  PR(C) = 0.22749

Iteration 5:
  PR(A) = 0.24337
  PR(B) = 0.15898
  PR(C) = 0.15898

Final PageRank values:
PR(A) = 0.24337
PR(B) = 0.15898
PR(C) = 0.15898
```

Conclusion: After running the PageRank algorithm for five iterations, node A achieves the highest PageRank value compared to nodes B and C. This indicates that node A holds the most influence within the given network structure, as it receives direct or indirect importance from other pages.