PageRank & HITS

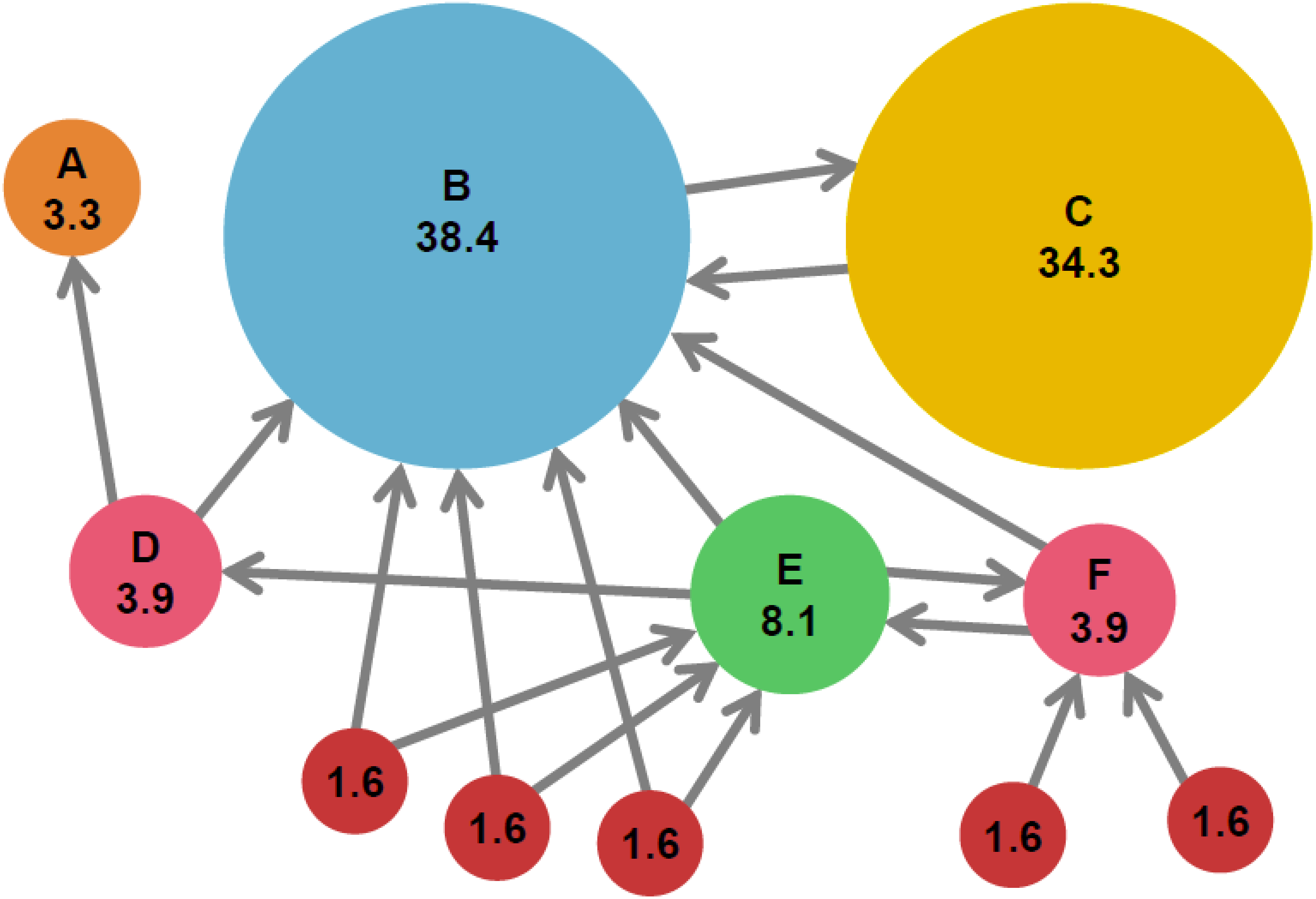
doc. dr. Matej Guid

Fakulteta za računalništvo in informatiko
Univerza v Ljubljani

februar 2022

© Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, 2021

PageRank Scores



© Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, 2021

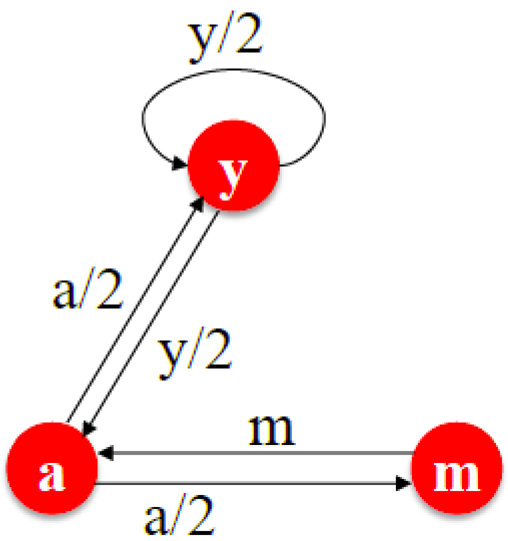
THE „FLOW“ MODEL

A page is important if it is pointed to by other important pages

Define a "rank" r_j for page j

$$r_j = \sum_{i \to j} \frac{r_i}{\mathrm{d}_i}$$

d_i … out-degree of node i



"Flow" equations:

$$r_y = r_y/2 + r_a/2$$
$$r_a = r_y/2 + r_m$$
$$r_m = r_a/2$$

Additional constraint forces uniqueness:

- $r_y + r_a + r_m = 1$
- Solution: $r_y = \frac{2}{5}$, $r_a = \frac{2}{5}$, $r_m = \frac{1}{5}$

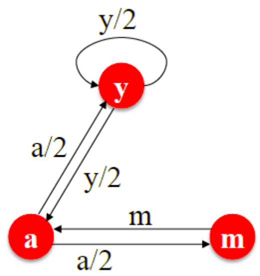
© Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, 2021

PageRank: Matrix Formulation

Rank vector r: vector with an entry per page
- r_i is the importance score of page i
- $\sum_i r_i = 1$

The flow equations can be written

$$r = M \cdot r$$



"Flow" equations:

$$r_y = r_y/2 + r_a/2$$
$$r_a = r_y/2 + r_m$$
$$r_m = r_a/2$$

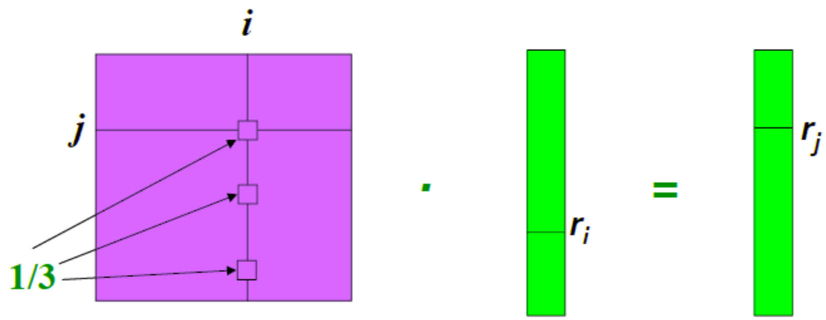
$$r_y + r_a + r_m = 1$$

Solution: $r_y = \frac{2}{5},\ r_a = \frac{2}{5},\ r_m = \frac{1}{5}$

- **Flow equation in the matrix form**

$$M \cdot r = r$$

  - **Suppose page *i* links to 3 pages, including *j***



$M \quad \cdot \quad r \quad = \quad r$

$$r_j = \sum_{i \to j} \frac{r_i}{d_i}$$

d_i … out-degree of node i

M is a **column stochastic matrix**
- Columns sum to 1

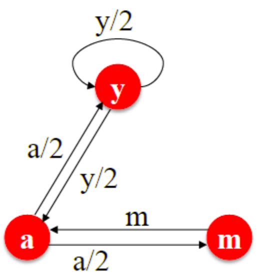
So the **rank vector r** is an **eigenvector** of the stochastic web matrix M

NOTE: x is an eigenvector with the corresponding eigenvalue λ if:

$$Ax = \lambda x$$

© Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, 2021

Power Iteration Method



"Flow" equations:

$$r_y = r_y/2 + r_a/2$$
$$r_a = r_y/2 + r_m$$
$$r_m = r_a/2$$

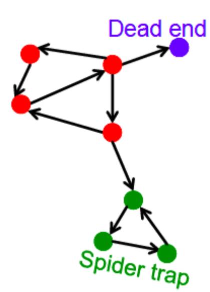
	y	a	m
y	½	½	0
a	½	0	1
m	0	½	0

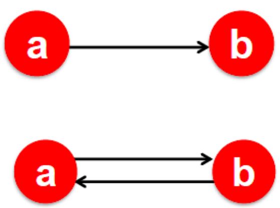
- **Power Iteration:**
  - Set $r_j = 1/N$
  - **1:** $r'_j = \sum_{i \to j} \frac{r_i}{d_i}$
  - **2:** $r = r'$
  - Goto **1**

$$\begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix} = \begin{array}{cccccc} 1/3 & 1/3 & 5/12 & 9/24 & & 6/15 \\ 1/3 & 3/6 & 1/3 & 11/24 & \ldots & 6/15 \\ 1/3 & 1/6 & 3/12 & 1/6 & & 3/15 \end{array}$$

Iteration 0, 1, 2, …

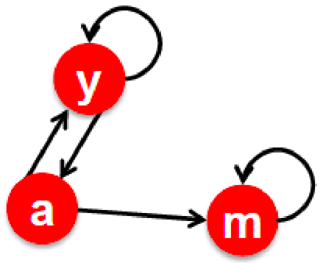
2 problems:





© Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, 2021

SPIDER TRAPS AND TELEPORTS



m is a spider trap

	y	a	m
y	½	½	0
a	½	0	0
m	0	½	1

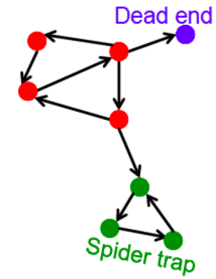
$$r_y = r_y/2 + r_a/2$$
$$r_a = r_y/2$$
$$r_m = r_a/2 + r_m$$

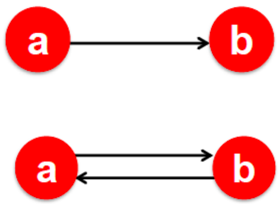
$$\begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix} = \begin{matrix} 1/3 & 2/6 & 3/12 & 5/24 & & 0 \\ 1/3 & 1/6 & 2/12 & 3/24 & \ldots & 0 \\ 1/3 & 3/6 & 7/12 & 16/24 & & 1 \end{matrix}$$

Iteration 0, 1, 2, …

All the PageRank score gets "trapped" in node m.

2 problems:





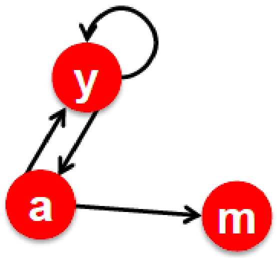
Surfer will teleport out of spider trap

With prob. β, follow a link at random

With prob. **1-**β, jump to some random page

© Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, 2021

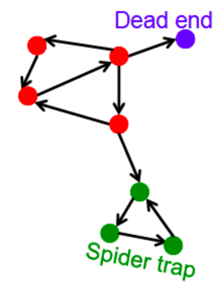
Dead Ends: Always Teleport!

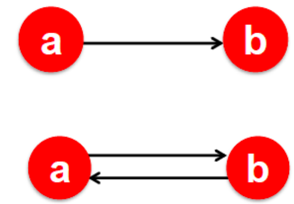


	y	a	m
y	½	½	0
a	½	0	0
m	0	½	0

$$r_y = r_y/2 + r_a/2$$
$$r_a = r_y/2$$
$$r_m = r_a/2$$

2 problems:

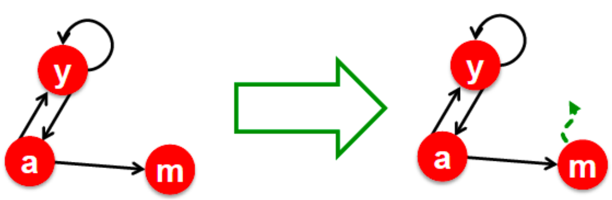




$$\begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix} = \begin{array}{cccccc} 1/3 & 2/6 & 3/12 & 5/24 & & 0 \\ 1/3 & 1/6 & 2/12 & 3/24 & \ldots & 0 \\ 1/3 & 1/6 & 1/12 & 2/24 & & 0 \end{array}$$

Iteration 0, 1, 2, …

Teleports: Follow random teleport links with probability 1.0 from dead-ends



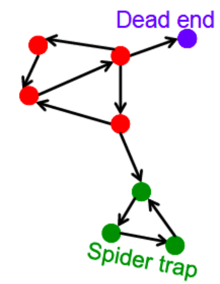
	y	a	m
y	½	½	⅓
a	½	0	⅓
m	0	½	⅓

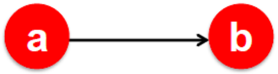
© Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, 2021

Random Teleports

Dead-ends are a problem

- The matrix is not column stochastic so our initial assumptions are not met





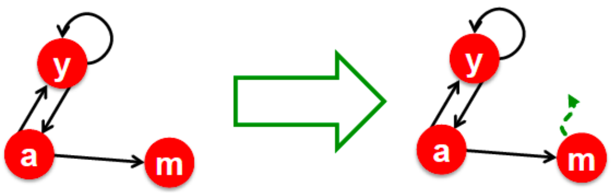
At each step, random surfer has two options:

- With probability β, follow a link at random
- With probability 1-β, jump to some random page

- **PageRank equation** [Brin-Page, 98]

$$r_j = \sum_{i \to j} \beta \frac{r_i}{d_i} + (1-\beta)\frac{1}{N}$$

d_i … out-degree of node i



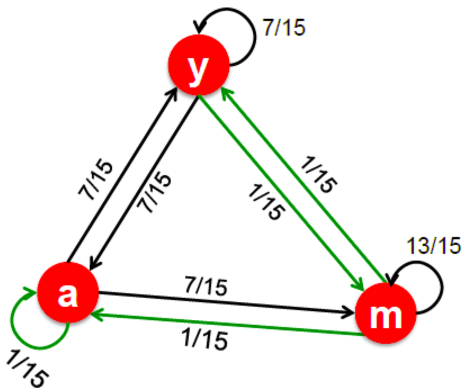
The Google Matrix *A*:

$[1/N]_{NxN}$…N by N matrix where all entries are 1/N

$$A = \beta\, M + (1-\beta)\left[\frac{1}{N}\right]_{N\times N}$$

© Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, 2021

Computing PageRank & Rearraging the Equation



$$
0.8 \underbrace{\begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 \\ 0 & 1/2 & 1 \end{bmatrix}}_{\mathbf{M}} + 0.2 \underbrace{\begin{bmatrix} 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \end{bmatrix}}_{[1/N]_{N\times N}}
$$

$$
\mathbf{A} = \begin{array}{c} y \\ a \\ m \end{array} \begin{bmatrix} 7/15 & 7/15 & 1/15 \\ 7/15 & 1/15 & 1/15 \\ 1/15 & 7/15 & 13/15 \end{bmatrix}
$$

A = $\beta\cdot$**M** + $(1-\beta)\ [1/N]_{N\times N}$

$$
\mathbf{A} = 0.8 \begin{bmatrix} ½ & ½ & 0 \\ ½ & 0 & 0 \\ 0 & ½ & 1 \end{bmatrix} + 0.2 \begin{bmatrix} 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \end{bmatrix}
$$

$$
= \begin{bmatrix} 7/15 & 7/15 & 1/15 \\ 7/15 & 1/15 & 1/15 \\ 1/15 & 7/15 & 13/15 \end{bmatrix}
$$

y		1/3	0.33	0.24	0.26		7/33
a	=	1/3	0.20	0.20	0.18	. . .	5/33
m		1/3	0.46	0.52	0.56		21/33

Matrix A has N^2 entries

We just rearranged the **PageRank equation**

$$
\boldsymbol{r} = \boldsymbol{\beta M} \cdot \boldsymbol{r} + \left[\frac{\mathbf{1}-\boldsymbol{\beta}}{\boldsymbol{N}}\right]_{\boldsymbol{N}}
$$

- where $[(1\text{-}\beta)/N]_N$ is a vector with all ***N*** entries **(1-β)/N**

M is a **sparse matrix!** (with no dead-ends)

© Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, 2021

PageRank: The Complete Algorithm

- **Input: Graph G and parameter β**
  - Directed graph $\boldsymbol{G}$ (can have **spider traps** and **dead ends**)
  - Parameter $\boldsymbol{\beta}$
- **Output: PageRank vector $\boldsymbol{r}^{new}$**
  - **Set:** $r_j^{old} = \frac{1}{N}$
  - **repeat until convergence:** $\sum_j \left| r_j^{new} - r_j^{old} \right| > \varepsilon$
    - $\forall j\colon \boldsymbol{r'}_j^{new} = \sum_{i \to j} \boldsymbol{\beta} \frac{\boldsymbol{r}_i^{old}}{\boldsymbol{d}_i}$

      $\boldsymbol{r'}_j^{new} = \mathbf{0}$ if in-degree of $\boldsymbol{j}$ is **0**
    - **Now re-insert the leaked PageRank:**

      $\forall \boldsymbol{j}\colon \boldsymbol{r}_j^{new} = \boldsymbol{r'}_j^{new} + \frac{\mathbf{1-S}}{\boldsymbol{N}}$ where: $S = \sum_j r'^{new}_j$
    - $\boldsymbol{r}^{old} = \boldsymbol{r}^{new}$

If the graph has no dead-ends then the amount of leaked PageRank is **1-β**. But since we have dead-ends the amount of leaked PageRank may be larger. We have to explicitly account for it by computing **S**.

© Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, 2021

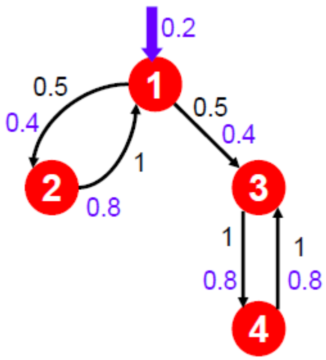
To make this work all we need is to update the teleportation part of the PageRank formulation:

$$A_{ij} = \begin{cases} \beta M_{ij} + (1-\beta)/|S| & \text{if } i \in S \\ \beta M_{ij} + 0 & \text{otherwise} \end{cases}$$

- A is a stochastic matrix!

We weighted all pages in the teleport set S equally

- **Could also assign different weights to pages!**



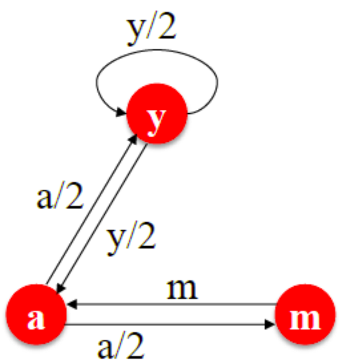
Suppose $S = \{1\}$, $\beta = 0.8$

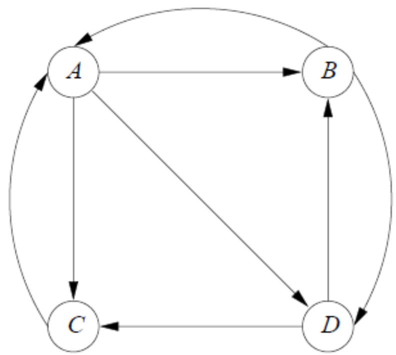
Node	Iteration 0	1	2	…	stable
1	0.25	0.4	0.28		0.294
2	0.25	0.1	0.16		0.118
3	0.25	0.3	0.32		0.327
4	0.25	0.2	0.24		0.261

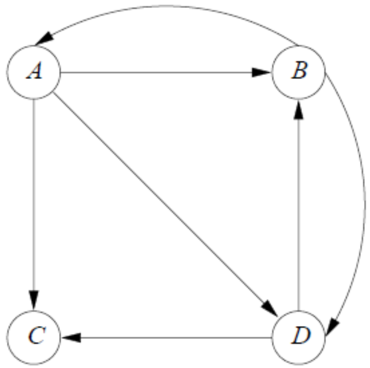
S={1,2,3,4}, β=0.8:
r=[0.13, 0.10, 0.39, 0.36]
S={1,2,3} , β=0.8:
r=[0.17, 0.13, 0.38, 0.30]
S={1,2} , β=0.8:
r=[0.26, 0.20, 0.29, 0.23]
S={1} , β=0.8:
r=[0.29, 0.11, 0.32, 0.26]

S={1}, β=0.9:
r=[0.17, 0.07, 0.40, 0.36]
S={1} , β=0.8:
r=[0.29, 0.11, 0.32, 0.26]
S={1}, β=0.7:
r=[0.39, 0.14, 0.27, 0.19]

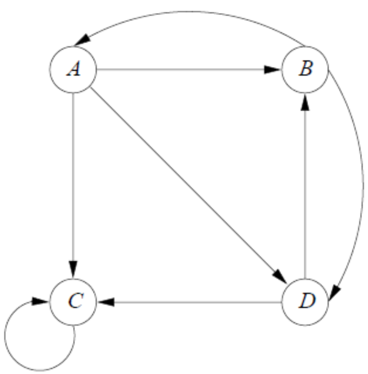
© Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, 2021

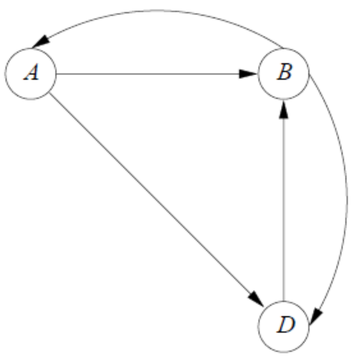


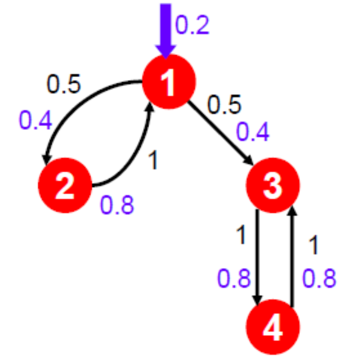




Exercises

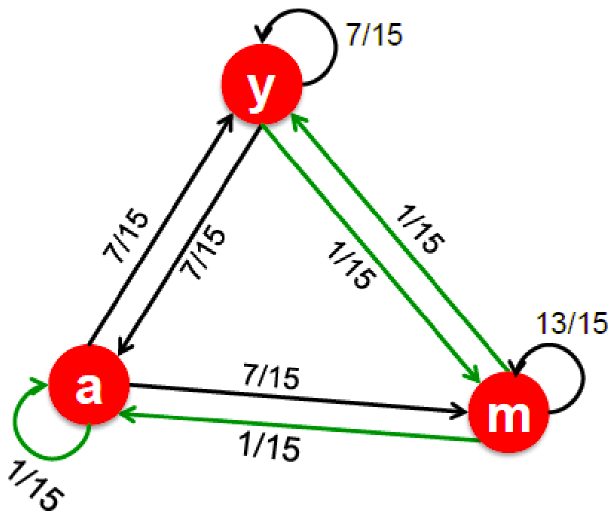






© Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, 2021

Univerza v Ljubljani
Fakulteta za računalništvo in informatiko



$$\begin{array}{c} \mathbf{M} \\ 0.8 \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 \\ 0 & 1/2 & 1 \end{bmatrix} \end{array} \begin{array}{c} \mathbf{[1/N]_{NxN}} \\ + 0.2 \begin{bmatrix} 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \end{bmatrix} \end{array}$$

$$\begin{array}{c} \begin{array}{c} y \\ a \\ m \end{array} \begin{bmatrix} 7/15 & 7/15 & 1/15 \\ 7/15 & 1/15 & 1/15 \\ 1/15 & 7/15 & 13/15 \end{bmatrix} \\ \mathbf{A} \end{array}$$

$$\begin{array}{l} y \\ a \\ m \end{array} = \begin{array}{c} 1/3 \\ 1/3 \\ 1/3 \end{array} \quad \begin{array}{c} 0.33 \\ 0.20 \\ 0.46 \end{array} \quad \begin{array}{c} 0.24 \\ 0.20 \\ 0.52 \end{array} \quad \begin{array}{c} 0.26 \\ 0.18 \\ 0.56 \end{array} \quad \ldots \quad \begin{array}{c} 7/33 \\ 5/33 \\ 21/33 \end{array}$$

$$A = \beta\, M + (1-\beta)\left[\frac{1}{N}\right]_{N\times N}$$

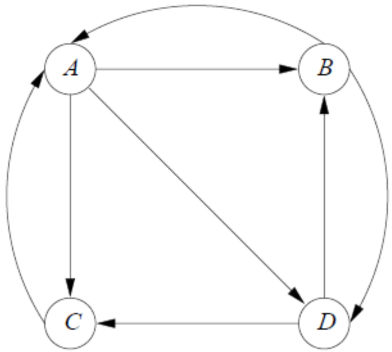
$[1/N]_{NxN}$…N by N matrix where all entries are 1/N

$$\boldsymbol{r} = \boldsymbol{\beta}\, \boldsymbol{M} \cdot \boldsymbol{r} + \left[\frac{\boldsymbol{1-\beta}}{\boldsymbol{N}}\right]_{\boldsymbol{N}}$$

since $\sum r_i = 1$

© Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, 2021

Exercise 2



$$M = \begin{bmatrix} 0 & 1/2 & 1 & 0 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{bmatrix}, \begin{bmatrix} 9/24 \\ 5/24 \\ 5/24 \\ 5/24 \end{bmatrix}, \begin{bmatrix} 15/48 \\ 11/48 \\ 11/48 \\ 11/48 \end{bmatrix}, \begin{bmatrix} 11/32 \\ 7/32 \\ 7/32 \\ 7/32 \end{bmatrix}, \dots, \begin{bmatrix} 3/9 \\ 2/9 \\ 2/9 \\ 2/9 \end{bmatrix}$$

$$A = \beta\, M + (1-\beta)\left[\frac{1}{N}\right]_{N\times N}$$

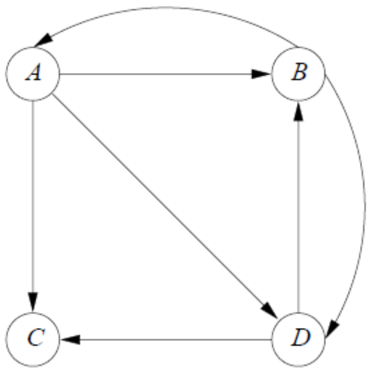
$[1/N]_{NxN}$…N by N matrix
where all entries are 1/N

$$\boldsymbol{r} = \boldsymbol{\beta}\, \boldsymbol{M} \cdot \boldsymbol{r} + \left[\frac{1-\boldsymbol{\beta}}{\boldsymbol{N}}\right]_{\boldsymbol{N}}$$

since $\sum r_i = 1$

© Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, 2021

EXERCISE 3



$$M = \begin{bmatrix} 0 & 1/2 & 0 & 0 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{bmatrix}, \begin{bmatrix} 3/24 \\ 5/24 \\ 5/24 \\ 5/24 \end{bmatrix}, \begin{bmatrix} 5/48 \\ 7/48 \\ 7/48 \\ 7/48 \end{bmatrix}, \begin{bmatrix} 21/288 \\ 31/288 \\ 31/288 \\ 31/288 \end{bmatrix}, \ldots, \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$A = \beta\, M + (1-\beta)\left[\frac{1}{N}\right]_{N\times N}$$

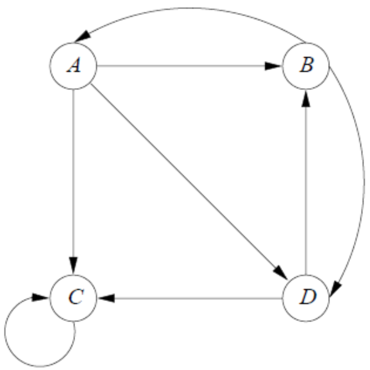
$[1/N]_{NxN}$…N by N matrix
where all entries are 1/N

$$\boldsymbol{r} = \boldsymbol{\beta}\, \boldsymbol{M} \cdot \boldsymbol{r} + \left[\frac{\mathbf{1-\beta}}{\boldsymbol{N}}\right]_{\boldsymbol{N}}$$

since $\sum r_i = 1$

© Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, 2021

Exercise 4



$$M = \begin{bmatrix} 0 & 1/2 & 0 & 0 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 0 & 1 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{bmatrix}, \begin{bmatrix} 3/24 \\ 5/24 \\ 11/24 \\ 5/24 \end{bmatrix}, \begin{bmatrix} 5/48 \\ 7/48 \\ 29/48 \\ 7/48 \end{bmatrix}, \begin{bmatrix} 21/288 \\ 31/288 \\ 205/288 \\ 31/288 \end{bmatrix}, \dots, \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

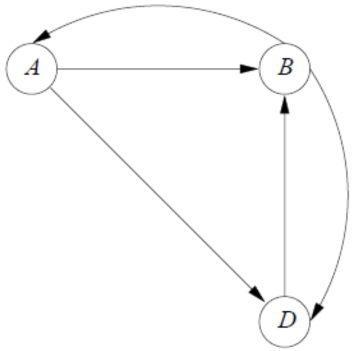
$$A = \beta\, M + (1-\beta)\left[\frac{1}{N}\right]_{N\times N}$$

$[1/N]_{NxN}$…N by N matrix
where all entries are 1/N

$$\boldsymbol{r} = \boldsymbol{\beta}\, \boldsymbol{M} \cdot \boldsymbol{r} + \left[\frac{\mathbf{1-\beta}}{\boldsymbol{N}}\right]_{\boldsymbol{N}}$$

since $\sum r_i = 1$

© Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, 2021



$$M = \begin{bmatrix} 0 & 1/2 & 0 \\ 1/2 & 0 & 1 \\ 1/2 & 1/2 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix}, \begin{bmatrix} 1/6 \\ 3/6 \\ 2/6 \end{bmatrix}, \begin{bmatrix} 3/12 \\ 5/12 \\ 4/12 \end{bmatrix}, \begin{bmatrix} 5/24 \\ 11/24 \\ 8/24 \end{bmatrix}, \ldots, \begin{bmatrix} 2/9 \\ 4/9 \\ 3/9 \end{bmatrix}$$

$$A = \beta\, M + (1-\beta)\left[\frac{1}{N}\right]_{N\times N}$$

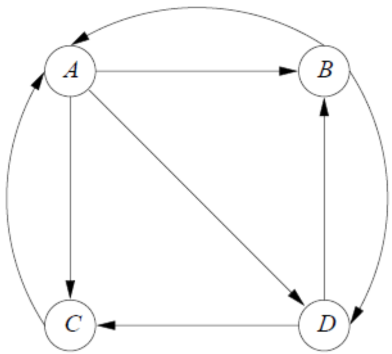
$[1/N]_{NxN}$…N by N matrix
where all entries are 1/N

$$\boldsymbol{r} = \boldsymbol{\beta}\, \boldsymbol{M} \cdot \boldsymbol{r} + \left[\frac{\mathbf{1}-\boldsymbol{\beta}}{\boldsymbol{N}}\right]_{\boldsymbol{N}}$$

since $\sum r_i = 1$

© Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, 2021

Exercise 6



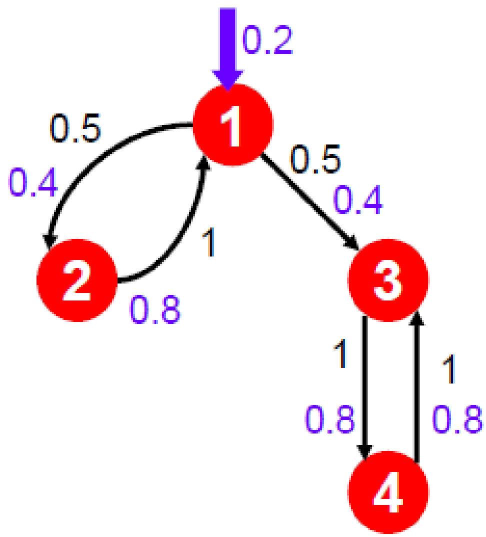
$$\beta M = \begin{bmatrix} 0 & 2/5 & 4/5 & 0 \\ 4/15 & 0 & 0 & 2/5 \\ 4/15 & 0 & 0 & 2/5 \\ 4/15 & 2/5 & 0 & 0 \end{bmatrix}$$

$$\mathbf{v}' = \begin{bmatrix} 0 & 2/5 & 4/5 & 0 \\ 4/15 & 0 & 0 & 2/5 \\ 4/15 & 0 & 0 & 2/5 \\ 4/15 & 2/5 & 0 & 0 \end{bmatrix} \mathbf{v} + \begin{bmatrix} 0 \\ 1/10 \\ 0 \\ 1/10 \end{bmatrix}$$

$$\begin{bmatrix} 0/2 \\ 1/2 \\ 0/2 \\ 1/2 \end{bmatrix}, \begin{bmatrix} 2/10 \\ 3/10 \\ 2/10 \\ 3/10 \end{bmatrix}, \begin{bmatrix} 42/150 \\ 41/150 \\ 26/150 \\ 41/150 \end{bmatrix}, \begin{bmatrix} 62/250 \\ 71/250 \\ 46/250 \\ 71/250 \end{bmatrix}, \ldots, \begin{bmatrix} 54/210 \\ 59/210 \\ 38/210 \\ 59/210 \end{bmatrix}$$

© Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, 2021

Topic-Specific Page Rank



Suppose $S = \{1\}$, $\beta = 0.8$

Node	Iteration 0	1	2	…	stable
1	0.25	0.4	0.28		0.294
2	0.25	0.1	0.16		0.118
3	0.25	0.3	0.32		0.327
4	0.25	0.2	0.24		0.261

beta=0.8				0.9	0.7
S={1,2,3,4}	S={1,2,3}	S={1,2}	S={1}	S={1}	S={1}
0.13	0.17	0.26	0.29	0.17	0.39
0.1	0.13	0.2	0.11	0.07	0.14
0.39	0.38	0.29	0.32	0.4	0.27
0.36	0.3	0.23	0.26	0.36	0.19

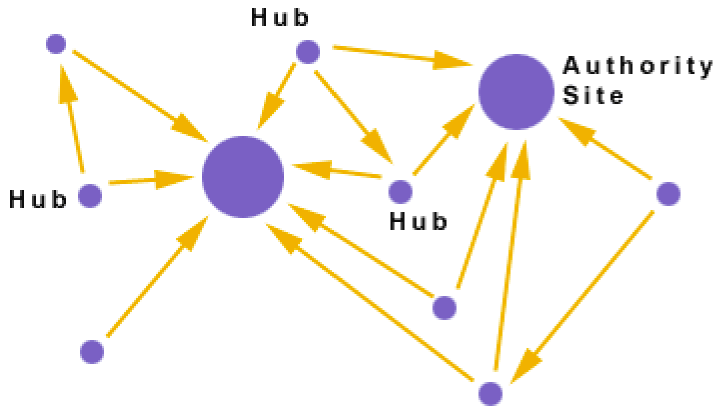
S={1}, β=0.9:
r=[0.17, 0.07, 0.40, 0.36]
S={1} , β=0.8:
r=[0.29, 0.11, 0.32, 0.26]
S={1}, β=0.7:
r=[0.39, 0.14, 0.27, 0.19]

S={1,2,3,4}, β=0.8:
r=[0.13, 0.10, 0.39, 0.36]
S={1,2,3} , β=0.8:
r=[0.17, 0.13, 0.38, 0.30]
S={1,2} , β=0.8:
r=[0.26, 0.20, 0.29, 0.23]
S={1} , β=0.8:
r=[0.29, 0.11, 0.32, 0.26]

© Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, 2021

Avtoritete

- ugledne spletne strani (npr. spletne strani univerz in vladnih organov)
- vsebujejo koristne informacije



Vozlišča

- spletne strani s povezavami do avtoritet

  oz. spletnih strani s koristnimi informacijami

© Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, 2021

HITS

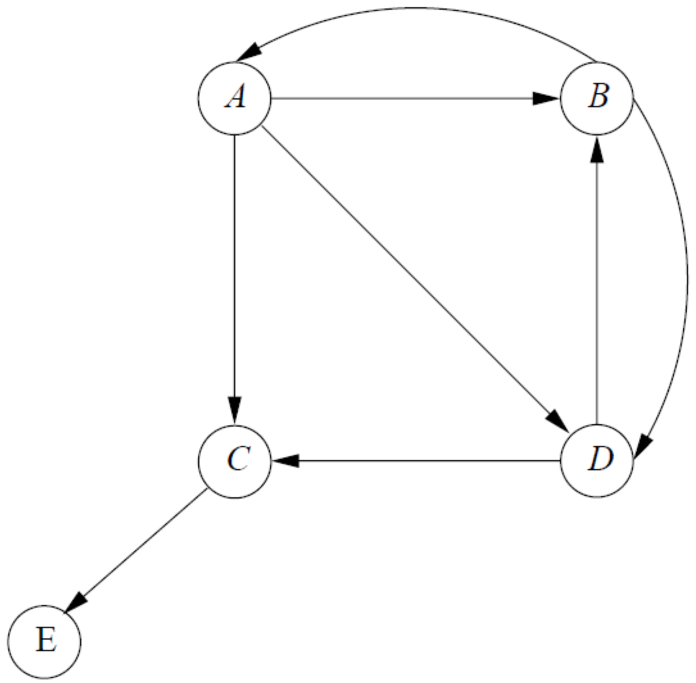


Figure 5.18: Sample data used for HITS examples

$$L = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \qquad L^{\mathrm{T}} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

© Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, 2021

HITS ALGORITHM

Repeat until convergence (or use fixed number of iterations):

1. Compute **a** = L^T**h**.
2. Scale so the largest component is 1.
3. Compute **h** = L**a**.
4. Scale again so the largest component is 1.

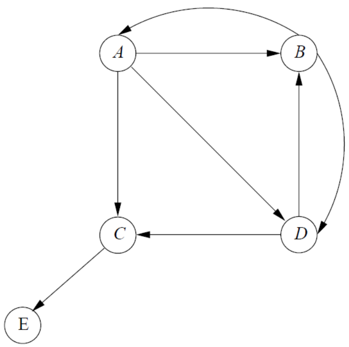


Figure 5.18: Sample data used for HITS examples

$$L = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \qquad L^{\mathrm{T}} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

$$\underset{\mathbf{h}}{\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}} \quad \underset{L^{\mathrm{T}}\mathbf{h}}{\begin{bmatrix} 1 \\ 2 \\ 2 \\ 2 \\ 1 \end{bmatrix}} \quad \underset{\mathbf{a}}{\begin{bmatrix} 1/2 \\ 1 \\ 1 \\ 1 \\ 1/2 \end{bmatrix}} \quad \underset{L\mathbf{a}}{\begin{bmatrix} 3 \\ 3/2 \\ 1/2 \\ 2 \\ 0 \end{bmatrix}} \quad \underset{\mathbf{h}}{\begin{bmatrix} 1 \\ 1/2 \\ 1/6 \\ 2/3 \\ 0 \end{bmatrix}}$$

$$\underset{L^{\mathrm{T}}\mathbf{h}}{\begin{bmatrix} 1/2 \\ 5/3 \\ 5/3 \\ 3/2 \\ 1/6 \end{bmatrix}} \quad \underset{\mathbf{a}}{\begin{bmatrix} 3/10 \\ 1 \\ 1 \\ 9/10 \\ 1/10 \end{bmatrix}} \quad \underset{L\mathbf{a}}{\begin{bmatrix} 29/10 \\ 6/5 \\ 1/10 \\ 2 \\ 0 \end{bmatrix}} \quad \underset{\mathbf{h}}{\begin{bmatrix} 1 \\ 12/29 \\ 1/29 \\ 20/29 \\ 0 \end{bmatrix}}$$

$$\sum_i \left(h_i^{(t)} - h_i^{(t-1)}\right)^2 < \varepsilon$$

$$\sum_i \left(a_i^{(t)} - a_i^{(t-1)}\right)^2 < \varepsilon$$

© Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, 2021

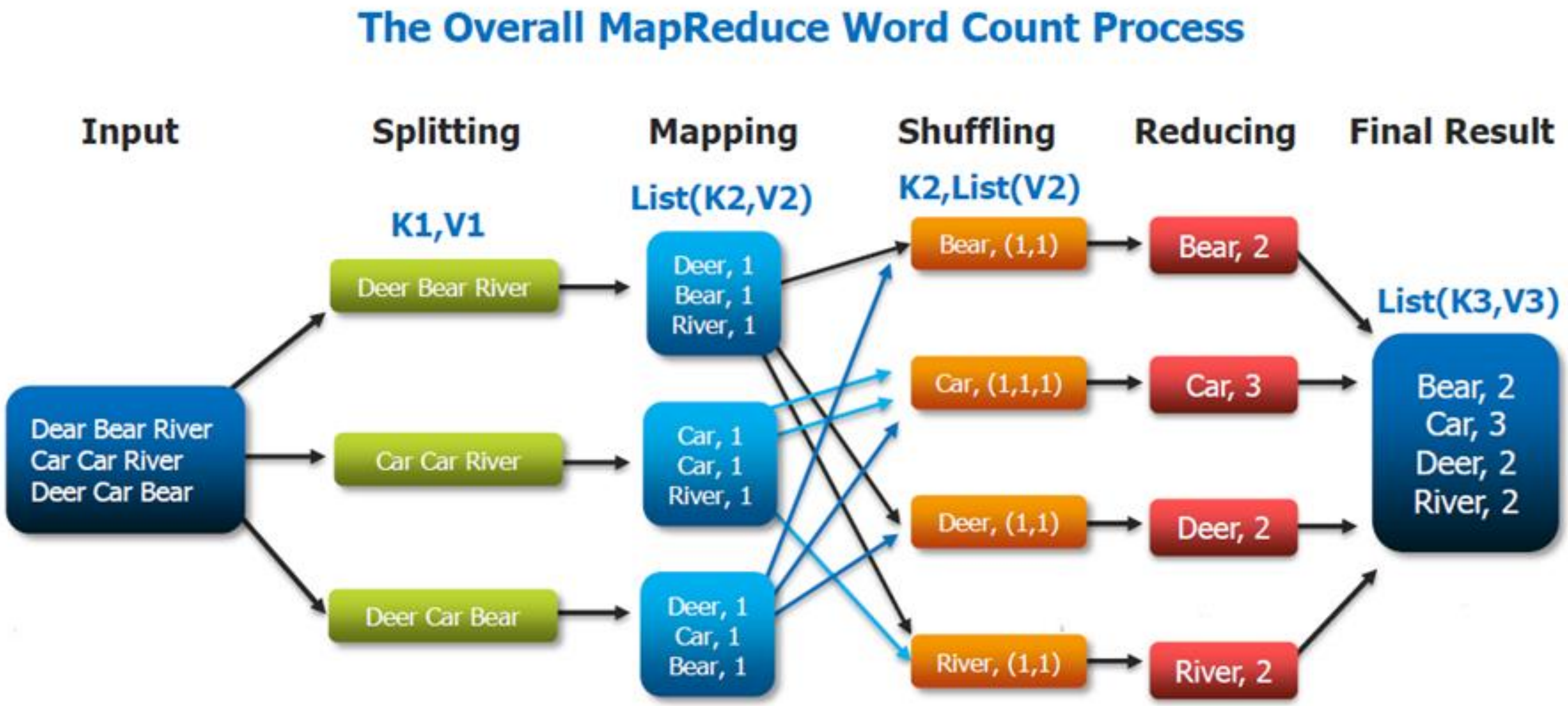
HITS Simulation

SUM =$H2*A$9+$I2*A$10+$J2*A$11+$K2*A$12+$L2*A$13

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1		L						L^T									
2		0	1	1	1	0		0	1	0	0	0					
3		1	0	0	1	0		1	0	0	1	0					
4		0	0	0	0	1		1	0	0	1	0					
5		0	1	1	0	0		1	1	0	0	0					
6		0	0	0	0	0		0	0	1	0	0					
7																	
8	h	LTh	a	La	h	LTh	a	La	h	LTh	a	La	h	LTh	a	La	h
9	1.000	=$H2*A$9	0.500	3.000	1.000	0.500	0.300	2.900	1.000	0.414	0.245	2.837	1.000	0.381	0.224	2.810	1.000
10	1.000	2.000	1.000	1.500	0.500	1.667	1.000	1.200	0.414	1.690	1.000	1.082	0.381	1.705	1.000	1.034	0.368
11	1.000	2.000	1.000	0.500	0.167	1.667	1.000	0.100	0.034	1.690	1.000	0.020	0.007	1.705	1.000	0.004	0.002
12	1.000	2.000	1.000	2.000	0.667	1.500	0.900	2.000	0.690	1.414	0.837	2.000	0.705	1.381	0.810	2.000	0.712
13	1.000	1.000	0.500	0.000	0.000	0.167	0.100	0.000	0.000	0.034	0.020	0.000	0.000	0.007	0.004	0.000	0.000
14																	
15					0.00000		0.20000		0.00000		0.05510		0.00000		0.02127		0.00000
16					0.50000		0.00000		0.08621		0.00000		0.03250		0.00000		0.01343
17					0.83333		0.00000		0.13218		0.00000		0.02729		0.00000		0.00569
18					0.33333		0.10000		-0.02299		0.06327		-0.01538		0.02661		-0.00668
19					1.00000		0.40000		0.00000		0.07959		0.00000		0.01619		0.00000

© Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, 2021

MapReduce



source: https://wikis.nyu.edu/display/NYUHPC/Tutorials

© Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, 2021

HITS & MAPREDUCE

TIMTOWTDI - There Is More Than One Way To Do It

STEP 1: obtain key-value pairs for L and L^T

```
L =             # key-value pairs for L-matrix
LT =            # key-value pairs for transpose of L-matrix
```

STEP 2: start with *h* (hubbiness) vector of all 1's

```
h =             # initial hubbiness vector
```

STEP 3: compute vectors *h* (hubbiness) and *a* (authority) iteratively in mutual recursion

```
for _ in range(NUM_ITERATIONS):
    a =            # compute a = LTh
    a_max =        # obtain maximum value in a
    a =            # scale a so the largest component is 1

    h =            # compute h = La
    h_max =        # obtain maximum value in h
    h =            # scale h so the largest component is 1
```

STEP 4: List the nodes with the highest/lowest hubbiness/authority score

© Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, 2021

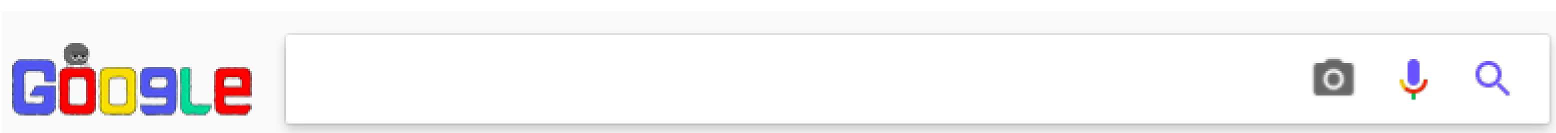
References

Stanford Computer Science course CS246: Mining Massive Datasets: Book

- Mining of Massive Datasets book by Jure Leskovec, Anand Rajaraman, and Jeff Ullman. http://www.mmds.org/

Stanford Computer Science course CS246: Mining Massive Datasets: HANDOUTS

- Page, Lawrence, Sergey Brin, Rajeev Motwani, and Terry Winograd. "The pagerank citation ranking: Bring order to the web." In Stanford Digital Libraries Working Paper. 1998.



© Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, 2021

Univerza v Ljubljani Fakulteta za računalništvo in informatiko