We would like to prove that $\omega(r') = \omega(r)$, when we propose that Web has no dead ends. We calculate:

$$\begin{aligned}
\omega(r') &= \sum_{i=1}^{n} r'_i \\
&= \sum_{i=1}^{n} \left( \sum_{j=1}^{n} M_{ij} r_j \right) \\
&= \sum_{j=1}^{n} \left( \sum_{i=1}^{n} M_{ij} r_j \right) \\
&= \sum_{j=1}^{n} r_j \left( \sum_{i=1}^{n} M_{ij} \right) \\
&= \sum_{j=1}^{n} r_j \\
&= \omega(r)
\end{aligned}$$

In fifth equality we used $\sum_{i=1}^{n} M_{ij} = 1$. This holds true, because there is a sum of $j$-th column of matrix $M$ on the left side. Since the Web has no dead ends there are $k_j$ values equal to $\frac{1}{k_j}$ in the $j$-th column for all $j$. The sum of each column is therefore equal to 1.

In this part we teleport to a random node with probability $1-\beta$, where $0<\beta<1$. Next estimate of $r_i$ is

$$r_i' = \beta\left(\sum_{j=1}^{n} M_{ij} r_j\right) + \frac{1-\beta}{n}.$$

We would like to see when $\omega(r') = \omega(r)$ is true. Therefore we need to determine when

$$\begin{aligned}
\sum_{i=1}^{n} r_i &= \sum_{i=1}^{n} r_i' \\
\sum_{i=1}^{n} r_i &= \sum_{i=1}^{n}\left(\beta\left(\sum_{j=1}^{n} M_{ij} r_j\right) + \frac{1-\beta}{n}\right) \\
\sum_{i=1}^{n} r_i &= \beta\sum_{i=1}^{n}\left(\sum_{j=1}^{n} M_{ij} r_j\right) + \sum_{i=1}^{n}\frac{1-\beta}{n} \\
\sum_{i=1}^{n} r_i &= \beta\sum_{j=1}^{n} r_j\left(\sum_{i=1}^{n} M_{ij}\right) + 1-\beta \\
\sum_{i=1}^{n} r_i &= \beta\sum_{j=1}^{n} r_j + 1-\beta \\
\omega(r) &= \beta\omega(r) + 1-\beta \\
\omega(r) - \beta\omega(r) &= 1-\beta \\
\omega(r)\left(1-\beta\right) &= 1-\beta \\
\omega(r) &= 1
\end{aligned}$$

We come to the conclusion that $\omega(r') = \omega(r)$ holds true, if $\omega(r) = 1$.

At each iteration, we teleport from live nodes with probability $1 - \beta$ and from dead nodes with probability 1. In both cases, we choose a random node uniformly to teleport to. Let $D$ be a set of all dead nodes. Equation for $r'_i$ can be written as

$$r'_i = \beta \left( \sum_{j \notin D} M_{ij} r_j \right) + \frac{1-\beta}{n} \sum_{j \notin D} r_j + \frac{1}{n} \sum_{j \in D} r_j,$$

where the first summand represents that with probability $\beta$ a web surfer chooses an out-link on their current page and the second summand $\frac{1-\beta}{n}$ represents that a web surfer opens a new page, in other words teleports form a live node. The last summand represents teleportation from a dead node

Now we would like to calculate the sum of all components or $r'$,

$$\omega(r') = \sum_{i=1}^{n} \left( \beta \sum_{j \notin D} M_{ij} r_j + \frac{1-\beta}{n} \sum_{j \notin D} r_j + \frac{1}{n} \sum_{j \in D} r_j \right).$$

Given the assumption $\omega(r) = \sum_{j=1}^{n} r_j = 1$ and knowing that $\sum_{i=1}^{n} M_{ij} = 1$, we calculate

$$\begin{aligned}
\omega(r') &= \sum_{i=1}^{n} \left( \beta \sum_{j \notin D} M_{ij} r_j + \frac{1-\beta}{n} \sum_{j \notin D} r_j + \frac{1}{n} \sum_{j \in D} r_j \right) \\
&= \beta \sum_{j \notin D} \left( \sum_{i=1}^{n} M_{ij} r_j \right) + \frac{1-\beta}{n} \sum_{j \notin D} r_j \left( \sum_{i=1}^{n} 1 \right) + \frac{1}{n} \sum_{j \in D} r_j \left( \sum_{i=1}^{n} 1 \right) \\
&= \beta \sum_{j \notin D} r_j + (1-\beta) \sum_{j \notin D} r_j + \sum_{j \in D} r_j \\
&= \sum_{j \notin D} r_j + \sum_{j \in D} r_j \\
&= \sum_{j=1}^{n} r_j \\
&= 1
\end{aligned}$$

Top 5 node ids with the highest PageRank scores, from highest to lowest:

- 263,
- 537,
- 965,
- 243,
- 285

Top 5 node ids with the lowest PageRank scores, from lowest to highest:

- 558,
- 93,
- 62,
- 424,
- 408

Top 5 node ids with the highest hubbines scores, from highest to lowest:

- 840,
- 155,
- 234,
- 389,
- 472

Top 5 node ids with the lowest hubbines scores, from lowest to highest:

- 23,
- 835,
- 141,
- 539,
- 889

Top 5 node ids with the highest authority scores, from highest to lowest:

- 893,
- 16,
- 799,
- 146,
- 473

Top 5 node ids with the lowest authority scores, from lowest to highest:

- 19,
- 135,
- 462,
- 24,
- 910

$C_i$ is defined as a set of nodes of $G$ that are divisible by $i$, $i > 0$. By this definition every pair of nodes has a common factor $i$ and is therefore connected. Hence there is an edge between every two nodes in $C_i$. We conclude that $C_i$ is a clique.

$C_i$ is a maximal clique if and only if $i$ is a prime number.

If $i$ is prime, then all the numbers between 2 and 1000000 that are divisible by $i$ are in $C_i$. If we add another node $v$ to $C_i$, the added node $v$ is not divisible by $i$, since all numbers divisible by $i$ are already in $C_i$. Therefore $v$ will not connect to all the nodes already in $C_i$ and $C_i \cup \{v\}$ will not have the property of a clique. Then $C_i$ really is a maximal clique and that the condition that $i$ is prime is sufficient.

If $i$ is not prime, we can write it as $i = p_1^{k_1} \cdots p_n^{k_n}$. A node $p_1$ is not in $C_i$ since it is not divisible by $i$. However if we add it in $C_i$ it will connect to all the nodes already in $C_i$, because $i$ and $p_1$ have a common factor $p_1$, forming a clique. Therefore for $i$ not prime, $C_i$ is not a maximal clique. This is proof that condition that $i$ is prime is necessary.

$C_2$ is the largest clique among all the cliques of form $C_i$, since cardinality of a clique in this form is

$$|C_i| = \lfloor \frac{1000000}{i} \rfloor.$$

Because $i = 2$ is the smallest number, $|C_2| = 500000$ is the highest of all $C_i$, $i > 1$.

Cliques that are not of form $C_i$ are sets of nodes that all have some common factor, but not all nodes that are divisible by this common factor are necessarly in this clique. Therefore cliques of this form are subsets of $C_i$ for some $i$ and therefore have smaller cardinality than $C_2$.

We conclude that $C_2$ is the largest clique.

i. We want to prove $|A(S)| \geq \frac{\epsilon}{1+\epsilon}|S|$. Let $\overline{A(S)} = \{i \in S | \deg_S(i) > 2(1+\epsilon)\rho(S)\}$ be complement of $A(S)$. Then it's cardinality is

$$|\overline{A(S)}| = |S| - |A(S)|. \tag{1}$$

We know that the sum of all degrees in a graph is equal to $2|E(S)|$, since every edge is counted twice. We also know that $\overline{A(S)}$ is a subgraph of $S$, therefore sum of all degrees of vertices in $\overline{A(S)}$ is at most sum of all degrees of vertices in $S$. We obtain inequality:

$$2|E(S)| \leq \sum_{i \in S} \deg_S(i) \leq \sum_{i \in \overline{A(S)}} \deg_S(i). \tag{2}$$

Now sum all the degrees of vertices in $\overline{A(S)}$. We know that for every node $i$ in $\overline{A(S)}$ holds: $\deg_S(i) > 2(1+\epsilon)\rho(S)$, therefore:

$$\sum_{i \in \overline{A(S)}} \deg_S(i) > \sum_{i \in \overline{A(S)}} 2(1+\epsilon)\rho(S) = |\overline{A(S)}| \cdot 2(1+\epsilon)\rho(S).$$

Using inequality (2), we get

$$|\overline{A(S)}| \cdot 2(1+\epsilon)\rho(S) < 2|E(S)|.$$

By using (1) and the definition of $\rho(S)$, we obtain the following set of inequalites

$$\begin{aligned}(|S| - |A(S)|) \cdot 2(1+\epsilon)\rho(S) &< 2\rho(S)|S| \\ |S| - |A(S)| &< \frac{|S|}{1+\epsilon} \\ |S| \cdot \left(1 - \frac{1}{1+\epsilon}\right) &< |A(S)| \\ |S| \cdot \left(\frac{\epsilon}{1+\epsilon}\right) &< |A(S)|.\end{aligned}$$

This concludes our proof.

ii. Now we would like to prove that the algorithm terminates in $O(\log_{1+\epsilon}(n))$ iterations, where $|S| = n$ is initial number of nodes. To see this let's denote $S_i$ as a subgraph obtained in $i$-it iteration. Then cardinality of $S_i$ is

$$|S_i| = |S_{i-1}| - |A(S_{i-1})| \leq |S_{i-1}| - \frac{\epsilon}{1+\epsilon}|S_{i-1}| \leq |S_{i-1}| \cdot \left(\frac{1}{1+\epsilon}\right).$$

We would like to find number of iterations $k$ after which algorithm terminates. Cardinality of $S$ at the beginning is $|S_0| = n$. After $k$ iterations $S_k$ has cardinality $S_k = n \cdot \left(\frac{1}{1+\epsilon}\right)^k$. We are searching for highest $k$, for which $|S_k|$ is still nonzero. Let's say it's cardinality has to be at least 1.

.

Obtaining the following set of inequalites:

$$
\begin{aligned}
0 < |S_k| &\leq n \cdot \left(\frac{1}{1+\epsilon}\right)^k \\
1 &\leq n \cdot \left(\frac{1}{1+\epsilon}\right)^k \\
\frac{1}{n} &\leq (1+\epsilon)^{-k} \\
\log_{1+\epsilon}\left(\frac{1}{n}\right) &\leq -k \\
-\log_{1+\epsilon}(n) &\leq -k \\
\log_{1+\epsilon}(n) &\geq k
\end{aligned}
$$

We conclude that algorithm takes at most $\log_{1+\epsilon}(n)$ steps.

i. For the densest subgraph $S^*$ of $G$ we would like to prove that for any $v \in S^*$, we have $\deg_{S^*}(v) \geq \rho^*(G)$. $\rho(S)$ is defined as $\rho(S) = \frac{|E(S)|}{|S|}$. Since $\rho(S^*)$ is the highest among all densities of subgraphs, it has to include all possible edges between nodes in $S^*$ that are in $G$. Therefore $S^*$ is the induced subgraph. Because it is also the densest of all induced subgraph it holds:

$$\rho^*(G) = \rho(S^*). \tag{3}$$

Now consider that there exists a vertex $v \in S^*$ such that

$$\deg_{S^*}(v) < \rho^*(G). \tag{4}$$

Define $\tilde{S} = S^* \backslash \{v\}$. Then

$$\begin{aligned}
\rho(S^*) &= \frac{E[S^*]}{|S^*|} \\
&= \frac{E[\tilde{S}] + \deg_{S^*}(v)}{|S^*|} \\
&= \frac{|S^*| - 1}{|S^*|} \frac{E[\tilde{S}]}{|S^*| - 1} + \frac{\deg_{S^*}(v)}{|S^*|} \\
\rho(S^*) &= \left(1 - \frac{1}{|S^*|}\right) \rho(\tilde{S}) + \frac{1}{|S^*|} \deg_{S^*}(v).
\end{aligned}$$

Last equality tells us that $\rho(S^*)$ is a weighted sum of $\rho(\tilde{S})$ and $\deg_{S^*}(v)$. From (3) and (4) it follows that $\deg_{S^*}(v) < \rho^*(S) = \rho(S^*)$. Therefore $\rho(\tilde{S}) > \rho(S^*)$. But this is in contradiction with the fact that $S^*$ is the densest subgraph.

ii. Assume that there exists a node $v \in S^* \cap A(S)$. We would like to prove that $2(1+\epsilon)\rho(S) \geq \rho^*(G)$.

Because $v \in A(S)$, from (a) part follows that $\deg_S(v) \leq 2\,(1+\epsilon)\,\rho(S)$.

Because $v \in S^*$, from $(i)$ part follows that $\rho^*(G) \geq \deg_{S^*}(v)$.

From construction of $S^*$, we know that $S^* \subset S$, therefore each node in $S^*$ has smaller degree than the same node in $S$: $\deg_{S^*}(v) \leq \deg_S(v)$. Combining all elements, we obtain:

$$2(1+\epsilon)\rho(S) \geq \deg_S(v) \geq \deg_{S^*}(v) \geq \rho^*(G).$$

iii. Now we would like to prove that $\rho(\tilde{S}) \geq \frac{1}{2(1+\epsilon)}\rho^*(G)$.

In every iteration we remove all the nodes from $A(S)$ if $\rho(S) > \rho(\tilde{S})$. From some step forward $\rho(S) \leq \rho(\tilde{S})$ will hold true. Therefore $\tilde{S} \longleftarrow S$ will never again be executed. While $\rho(S)$ will become smaller with every iteration $\rho(\tilde{S})$ will stay the same.

At the final iteration we get $\rho(S) \leq \rho(\tilde{S})$.

Using $(ii)$ it follows:

$$\rho^*(G) \leq 2\,(1+\epsilon)\,\rho(S) \leq 2\,(1+\epsilon)\,\rho(\tilde{S}).$$

We proved that $\rho(\tilde{S}) \geq \frac{1}{2(1+\epsilon)}\rho^*(G)$.

- doc 1 : $[-201.49, 35.3, -38.34, 19.77, 8.59, 18.24, 18.29, 22.89, 19.29]$,
- doc 2 : $[-147.26, -70.19, -26.35, 5.38, 54.18, -2.28, 2.21, 10.97, -8.43]$,
- doc 3 : $[-145.09, 26.81, 38.77, 33.8, 3.94, -36.21, 22.54, -44.29, -4.61]$,
- doc 4 : $[-190.2, -43.32, 13.14, -15.7, -39.66, 1.77, 14.61, 24.71, -5.47]$,
- doc 5 : $[-132.83, -19.84, 57.46, -16.56, 0.53, 5.76, 29.67, 11.23, -2.67]$,
- doc 6 : $[-122.42, 87.1, -0.45, -64.67, 6.6, -33.02, -25.73, 17.98, -19.92]$,
- doc 7 : $[-176.57, -24.33, 26.83, -34.24, 30.45, 24.33, 2.54, -13.11, -4.95]$,
- doc 8 : $[-111.91, 10.42, -52.93, -1.56, -5.22, 57.57, -19.43, -36.29, -13.46]$,
- doc 9 : $[-181.56, -71.17, -15.71, -27.35, -43.39, -3.15, -3.69, -6.77, -5.0]$,
- doc 10 : $[-206.53, -10.93, 9.4, -33.49, 3.95, -30.78, -17.87, -25.97, 27.13]$,
- doc 11 : $[-169.18, 31.94, 19.64, 27.51, 42.18, 8.29, 5.42, 8.96, -6.43]$,
- doc 12 : $[-124.33, -37.22, -60.29, 61.51, -2.53, -43.56, -21.63, 7.94, -8.48]$,
- doc 13 : $[-105.75, 28.98, -54.54, -22.75, 4.85, 6.57, -4.21, 6.63, 18.69]$,
- doc 14 : $[-185.75, 71.43, -19.03, 29.17, -37.05, 6.43, 22.3, -1.2, -6.0]$,
- doc 15 : $[-134.84, 5.53, 85.12, 46.08, -13.35, 23.5, -49.63, 13.29, 6.1]$.

- word 1: $[-198.48, -72.47, 0.17, -54.0, 2.16, 42.5, 4.41, 10.14, 0.93]$,
- word 2: $[-229.25, -36.06, 4.4, 61.75, 52.02, -24.49, 5.65, 22.33, 22.12]$,
- word 3: $[-169.56, 25.05, 98.28, -32.86, 19.21, -24.56, -48.48, -5.29, -7.1]$,
- word 4: $[-184.76, 15.79, 25.26, -39.47, -48.4, 1.3, 12.88, 40.16, 15.81]$,
- word 5: $[-169.09, 4.15, -32.55, 67.41, -59.41, 3.88, -38.9, 6.32, -4.37]$,
- word 6: $[-173.17, -72.72, 4.26, -9.66, -31.15, -61.04, 26.8, -32.39, -6.43]$,
- word 7: $[-169.51, 110.36, -45.99, -28.28, 0.71, -10.0, 4.62, -29.09, 21.04]$,
- word 8: $[-240.77, -30.04, -89.27, -25.31, 26.59, 13.63, -23.31, -9.46, -9.86]$,
- word 9: $[-224.5, 18.01, 62.67, 43.26, -1.27, 54.23, 24.96, -31.9, -4.28]$,
- word 10: $[-169.95, 69.79, -14.04, 5.15, 14.9, -16.15, 28.44, 30.92, -31.49]$.

# Information sheet
# CS246: Mining Massive Data Sets

**Assignment Submission** Fill in and include this information sheet with each of your assignments. This page should be the last page of your submission. Assignments are due at 11:59pm and are always due on a Thursday. All students (SCPD and non-SCPD) must submit their homework via Gradescope (http://www.gradescope.com). Students can typeset or scan their homework. Make sure that you answer each (sub-)question on a separate page. That is, one answer per page regardless of the answer length. Students also need to upload their code on Gradescope. Put all the code for a single question into a single file and upload it.

**Late Homework Policy** Each student will have a total of *two* late periods. *Homework are due on Thursdays at 11:59pm PT and one late period expires on the following Monday at 11:59pm PT*. Only one late period may be used for an assignment. Any homework received after 11:59pm PT on the Monday following the homework due date will receive no credit. Once these late periods are exhausted, any assignments turned in late will receive no credit.

**Honor Code** We strongly encourage students to form study groups. Students may discuss and work on homework problems in groups. However, each student must write down their solutions independently, i.e., each student must understand the solution well enough in order to reconstruct it by him/herself. Students should clearly mention the names of all the other students who were part of their discussion group. Using code or solutions obtained from the web (GitHub/Google/previous year's solutions etc.) is considered an honor code violation. We check all the submissions for plagiarism. We take the honor code very seriously and expect students to do the same.

**Your name:** Lucija Fekonja

**Email:** lf90992@student.uni-lj.si **SUID:** 27232071

Discussion Group: Nik Mrhar

I acknowledge and accept the Honor Code.

*(Signed)* LF