Homework 7: computing PageRank

Fundamentals of Data Science 2015

Write a Python script named `ID.py`, where `ID` is you student's ID, that

1. Takes as command-line arguments a file name and a floating-point value α.
2. Reads the file, which stores a directed graph as text lines; each line is in the format `u,v` meaning that the graph contains the arc (u, v). Skip lines starting with "#".
3. Computes the PageRank vector of the graph, using α as damping factor. Use the iterative approach and stop as soon as the distribution vector changes by less than 10^{-10} in 1-norm between consecutive iterations.

The script should print:

1. the number of nodes in the graph, the number of nonzero entries in the adjacency matrix, and its density
2. the number of iterations performed and the overall time they took (in seconds)
3. the minimum, maximum, average, and sum of the PageRank vector
4. on screen, a log-log plot of the rank function

   $$r(x) = 1 + (\text{number of nodes having PageRank score} > x)$$

   with x ranging over the distinct values in $\mathbf{p}$.

Figure 1 and Figure 2 give rough guidelines (of course for a specific graph). Your code must use sparse matrices – it will be tested on medium-large graphs ($n \approx 100k$), making the computation impossible if you use standard NumPy matrices/arrays.

PageRank

Recall that, given a graph $G = (V, A)$, the PageRank column vector can be computed by picking an arbitrary initial probability distribution vector $\mathbf{p}$ (i.e. $\mathbf{p} \geq 0$, $|\mathbf{p}| = 1$) and repeatedly performing the update operation:

$$\mathbf{p} := \alpha \cdot \mathbf{M} \cdot \mathbf{p} + \frac{1-\alpha}{n} \cdot \mathbf{1}$$

where $\mathbf{M}$ is the *transition matrix* having

$$M_{i,j} = \begin{cases} \frac{1}{outdeg(j)} & \text{if } (j,i) \in A \\ 0 & \text{if } (j,i) \notin A \end{cases}$$

Remember to take care of dangling nodes, that is, nodes with outdegree 0.

Sample Output

Figure 1: sample score/rank plot.

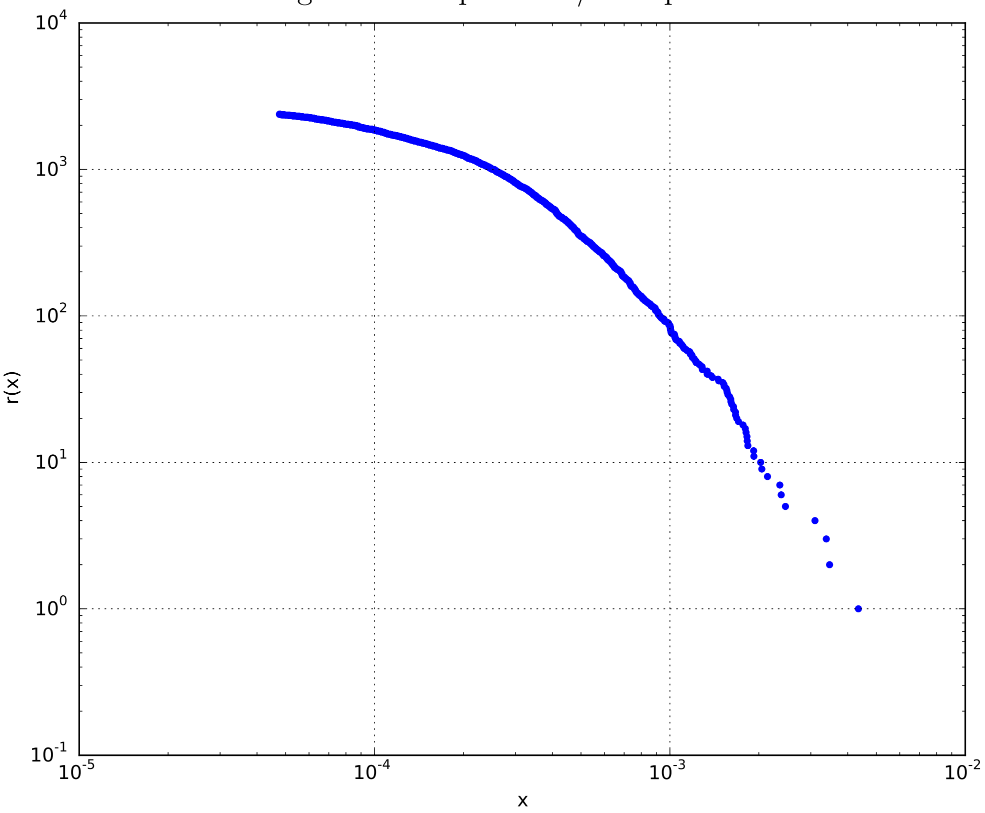

Figure 2: sample console output.

```
$ python ID.py mygraph.txt 0.85
n = 8298, M.nnz = 103689, density = 1.51e-03
iterations = 29, elapsed = 7.27e-03
min(p), max(p), avg(p), sum(p) = 4.76e-05, 4.35e-03, 1.21e-04, 1.00e+00
```