# AI607: GRAPH MINING AND SOCIAL NETWORK ANALYSIS (FALL 2020)

## Homework 2: Citation Network Analysis using HITS

Release: September 25, 2020,
Due: October 9, 2020, 11:59pm

In this assignment, your tasks are to implement HITS, which is an algorithm for measuring the importance of web pages, and to analyze a citation network using HITS. More information about HITS can be found in the lecture note and Chapter 5 of MMDS. (`http://infolab.stanford.edu/~ullman/mmds/ch5.pdf`).

## 1 Introduction

Some web pages called hubs are important not because they provide information but because they give links to informative web pages called authorities. In other words, a good hub is a page with links to many good authorities, and a good authority is a page linked to by many hubs. Thus, web pages need to be evaluated based on two scores, one for the quality as a hub and the other for the quality as an authority. For each node, its Hub score is obtained by summing the Authority scores of its neighbors. Next, the Authority score of each node is obtained by summing all of its neighbors' Hub scores. Repeating these two steps with normalization sufficiently leads to the final scores.

## 2 Definition

Let the Authority score and the Hub score of the node $i$ at epoch $t$ as $a_i^{(t)}$ and $h_i^{(t)}$. Then, the Hub and Authority score of the node $i$ at epoch $t+1$ is computed as follows:

$$h_i^{(t+1)} = \sum_{i \to j} a_j^{(t)}, \quad a_i^{(t+1)} = \sum_{j \to i} h_j^{(t+1)} \tag{1}$$

Then, $a^{(t+1)}$ and $h^{(t+1)}$ are normalized so that their L2 norms become 1. In this assignment, you should implement the HITS algorithm, which repeats this until $a^{(t)}$ and $h^{(t)}$ converge. Then, you should apply the HITS algorithm to a citation network, and report the top-10 Hub scores and Authority scores with the corresponding node IDs. More details are provided in the following section.

# 3 Implementation

## 3.1 Preprocessing

Your first task is to complete the `CitationNetwork` class in `graph.py`. To do this, you have to parse the file which contains the citation data. We will provide you `graph.txt`[1], which contains $\sim 600,000$ blocks, each of which is for a paper. Each block is in the following format:

```
#* --- Paper title
#@ --- Authors
#t --- Year
#c --- Publication venue
#index --- Index id of this paper
#% --- The id of references of this paper
        (there are multiple lines, with each indicating a reference)
#! --- Abstract
```

When converting the file into a directed graph, consider each paper as a node, and each (paper, reference) pair as a directed edge.

## 3.2 HITS

Next, fill out the `hits` function in `graph.py`, which is for computing the Hub and Authority scores using the power iteration method. In this homework, the Hub and Authority scores should be initialized uniformly. In addition, the iteration should stop when one of the following conditions is satisfied:

- The number of iterations exceeds the `max_iter`
- The L2-norm of the difference between previous Hub scores and new Hub scores is lower than the tolerance, i.e., $\left\|h^{(t+1)} - h^{(t)}\right\|_2 < tol$.

## 3.3 Top-$k$ Selection

Finally, implement the `print_top_k` function in `graph.py`, which is for printing the top-k scores and their corresponding paper ids in descending order in the following format:

```
<SCORE 1><TAB><PAPER ID 1>
<SCORE 2><TAB><PAPER ID 2>
...
<SCORE K><TAB><PAPER ID K>
```

More information can be found in the skeleton codes. Your implementation should be compatible with Python 3.7 or 3.8. Furthermore, you are allowed to use `numpy` and `scipy`, but other external libraries are not allowed. Feel free to use any python's default library.

[1]The provided dataset can be obtained from the ACM citation network dataset (`https://www.aminer.org/citation`).

# 4 Test

You can test your implementation by executing the `main.py` with some options:

```
usage: main.py [-h] [-f FILE]

Get the top 10 Hubs and Authority nodes for the given graph

optional arguments:
  -h, --help            show this help message and exit
  -f FILE, --file FILE  A file path for an initial matrix
```

- If you add the argument '-f [file]', then the program will load the input file as a graph, and run HITS algorithm.

# 5 Analysis

- Using `main.py`, report the list of papers with the top-10 hub scores and the papers with the top-10 authority scores in `graph.txt`. Do not change the default values of maximum iterations and tolerance.

# 6 Notes

- Your implementation should run on TA's desktop within 10 minutes.
- You may encounter some subtleties when it comes to implementation. Come up with your own design and/or contact Taehyung Kwon (taehyung.kwon@kaist.ac.kr) and Inkyu Park (inkyuhak@kaist.ac.kr) for discussion. Any ideas can be taken into consideration when grading if they are written in the *readme* file.
- Different implementations may give slightly different Hub-Authority scores due to the randomness, floating point errors, and so on. In that reason, your implementation will be evaluated in a robust way. Specifically, the Hub and Authority scores that your implementation gives and the ground-truth scores will be compared after we round both scores to the nearest ten thousandth.

# 7 How to submit your assignment

1. Create hw2-[your student id].tar.gz, which should contain the following files:
   - **main.py, graph.py**: these should contain your implementation.
   - **report.pdf**: this should contain your answers in Section 5 (Analysis).
   - **readme.txt**: this file should contain the names of any individuals from whom you received help, and the nature of the help that you received. That includes help from friends, classmates, lab TAs, course staff members, etc. In this file, you are also welcome to write any comments that can help us grade your assignment better, your evaluation of this assignment, and your ideas.

2. Make sure that no other files are included in the tar.gz file.
3. Submit the tar.gz file at KLMS (`http://klms.kaist.ac.kr`).