Project 2 – Influence Maximization Problem(IMP)

1. Overview

In this project, you have two computational tasks for Influence Maximization Problems (IMPs) in social networks. The first is to implement an estimation algorithm for the influence spread and the second is to design and implement a search algorithm for IMPs. An introduction to IMP (including problem formulations and influence spread definition) can be found in the package we provide (IMP.pdf). The IMP is NP-hard and the influence spread computation is #P-hard under the definitions shown in the introduction. Overall, your estimation algorithm needs to give a good estimation of the influence spread and your search algorithm needs to find a high-quality solution to IMPs as fast as possible within a limited time. The scores you get in this project will be given according to the performance of your algorithms in our test.

This project is divided into two phases. In each phase, you will have at least one week.

	Scoring rules	DeadLine
Lab4	Influence spread computation: We will select different seed sets with different sizes and test the estimation error of your estimator on the instances with these seed set. The score you get in this test depends on how many test instances you can pass. Whether your code passes one instance depends on whether the estimation error is in a given range.	Date: 14th Nov. Time: 23:55

Lab5	Influence maximization: we will specify different sizes of the seed set and test your solver on the instances. The solution quality of all participant solvers is compared in terms of the influence spread and consumed time. The scores you get in this test depend on the rankings your solver obtain.	Date: 21th Nov. Time: 23:55

After Lab5, a carefully-written experiment report need to be submitted.

Program_Score =(Lab4+Lab5)/2

Project2_score = Program_Score *0.7+Report_Score*0.3

2. Task Description

After your submission, we will test your influence spread estimator and IMP solver on different IMP problem instances. To make this process as smooth as possible, the program you submit must satisfy the following requirements.

a) Task 1: influence spread computation

The executable estimator must be named by **ISE.py.**

i. Input: the format of the estimator call should be as follows:

python ISE.py –i <social network> -s <seed set> -m <diffusion model> -t <time budget>

<social network> is the absolute path of the social network file

<seed set> is the absolute path of the seed set file

<diffusion model> can only be IC or LT

<time budget> is a positive number which indicates how many seconds (in Wall clock time, range: [60s, 120s]) your algorithm can spend on this instance.

ii. Output: the value of the estimated influence spread.

b) Task 2: influence maximization

The executable solver must be named by **IMP.py.**

i. Input: the format of the solver call should be as follows:

python IMP.py –i <social network> -k <predefined size of the seed set> -m <diffusion model> -t <time budget>

<social network> is the absolute path of the social network file

<predefined size of the seed set> is a positive integer

<diffusion model> can only be IC or LT

<time budget> is a positive number which indicates how many seconds (in Wall clock time, range: [60s, 120s]) your algorithm can spend on this instance.

ii. Output: the seed set found by your algorithm.

The format of the seed set output should be as follows: each line contains a node index. An example is also included in the package.

c) Supplementary instruction

i. The social networks are given in a uniform format and an example is provided in the package. The first line contains the number of nodes n and number of edges m, each of the next m lines describes an edge following the format: <src> <dest> <weight>. Node index starts from 1.

ii. The seed set are given in a uniform format and an example is provided in the package. Each line contains a node index. Your algorithm should give the output in this format as well.

3. Test Environment

1) Python version: 3.6
2) You should submit you ISE.py and IMP.py to the url: **************
3) Read the NOTICE on the homepage carefully before you submit your code. Please notice that the LeaderBoard is only a reference about your performance. After DDL, we will test your code with other test data. So, the ranking on the test platform does not decide your final grade.

Pay attention that all graphs given are sparse graphs. Using adjacency matrix is a waste of memory.

Even if your program exited normally now, it does not mean that you can pass all final tests. Some of you are at the border of OOM (Out of memory). NetHEPT only has 15,233 nodes. But final datasets contain up to 50,000 nodes. Bonus datasets may contain even more nodes.

If your program exited with exit code 137, it usually means your program was killed by docker due to exceeding memory usage.

For further information please refer to docker document: https://success.docker.com/article/what-causes-a-container-to-exit-with-code-137

Note that condition 1 in the above article does not apply to OJ system. Your program will receive SIGKILL only after time limit is reached. In this condition exit code 137 will not be collected. -1 is returned instead.

From now on, if your program exits with code 137, OJ system will treat it as OOM.

Also, if you have used multiprocessing, it might happen that only child processes were killed due to OOM. If parent process did not handle properly, it may hang forever with zero CPU usage, causing timed out result.

Sum up by Yi Xie