AI607: Graph Mining and Social Network Analysis (Fall 2021)

Homework 3: Disease Spreading with the SIR Model

Release: Oct 29, 2021,
Due: Nov 12, 2021, 11:59pm

In this assignment, we will simulate disease spreading using the SIR model, a widely-used epidemic model, and investigate the effects of the infection rate β of the model and the number of initial active nodes on the cascading behavior of networks.

1 The SIR Model

The SIR model is an epidemic spreading model where each individual is at any time in one of the three states, susceptible (healthy), infected (able to transmit the disease to others), or recovered (immunized). At the beginning, initially chosen nodes are in the infected state, while the others start are the susceptible state. The possible transition patterns at each timestamp are (1) individuals in the susceptible state getting infected from individual(s) independently with the infection rate β upon interaction, (2) individuals in the infected state getting cured with the recovery rate δ, transitioning to a recovered state (see Fig. 1). Note that individual(s) in the recovered state cannot transition to the other states. Individuals are nodes in a given graph, and the interactions between individuals, which are assumed to be static over all timestamps, are represented as edges.

Figure 1: The transition patterns between the three states (**S**usceptible, **I**nfected, or **R**ecovered).

2 Implementation [50 points]

In this section, you will implement the SIR model. You are allowed to use NumPy for the implementation. You are **NOT** allowed to use any other external libraries (Snap.py, NetworkX, etc.). In this assignment, we

assume that all the node indexes are between 0 to $N-1$, where N is the number of nodes in the input graph.

2.1 Sampling Initial Nodes [5 points]

To simulate the SIR model, we need to select initial active nodes. Your first task is to implement the `sampleInitialNodes` function in the `SIR.py`, which uniformly and randomly samples initial active nodes. Given the number of nodes N and the target number of initial active nodes N_0, your task is to return a list containing exactly N_0 different node indexes. Complete the function without changing its interface provided in `SIR.py`.

```
def sampleInitialNodes(num_nodes, num_initial_nodes):
    """
    Implement the function that samples the initial active nodes.

    Inputs:
        num_nodes: the number of nodes N in the given graph.
        num_initial_nodes: the target number of initial active nodes N_0.

    Output: a list of N_0 initial active nodes.
    """
```

2.2 Simulating the SIR model [45 points]

Your next task is to implement the `runSimulation` function in the `SIR.py` which simulates the SIR model. When the number of nodes, a list of edges, a list of the initial nodes, and the parameters of the SIR model are given, your implementation should return the number of the recovered nodes. Complete the function without changing its interface provided in `SIR.py`.

```
def runSimulation(num_nodes, edges, initial_nodes, beta, delta):
    """
    Implement the function that counts the ultimate number of recovered nodes.

    Inputs:
        num_nodes: the number of nodes in the input graph.
        edges: a list of tuples of the form (u, v),
          which indicates an edge from u to v.
          You can assume that there is no isolated node.
        initial_nodes: a list of initial active nodes.
        beta: the infection rate of the SIR model.
        delta: the recovery rate of the SIR model.

    Output: the number of recovered nodes after the diffusion process ends.
    """
```

3 Analysis [50 points]

In this section, you will evaluate the SIR model with various combinations of the parameters. You can evaluate your implementation by executing `simulate.py` with the following options:

```
usage: simulate.py [-h] [--beta BETA] [--delta DELTA]
                   [--initial-nodes INITIAL_NODES] [-seed SEED] INPUT

positional arguments:
  INPUT                 a path of the input graph.

optional arguments:
  --beta BETA           the infection rate of the SIR model.
  --delta DELTA         the recovery rate of the SIR model.
  --initial-nodes INITIAL_NODES
                        a target number of the initial nodes.
  --seed SEED           a random seed.
```

For example, you can run `simulate.py` as:

```
python simulate.py --beta 0.1 --delta 0.8
                   --initial-nodes 50 --seed 0 graph.txt
```

Your task is to investigate the effects of the infection rate of the SIR model and the number of the initial active nodes using `simulate.py`. To reduce randomness, `simulate.py` performs 100 trials then outputs the expected number of recovered nodes.

3.1 Effect of the infection rate β [25 points]

Your task is to visualize the relationship between the final number of recovered nodes and the infection rate β. The x-axis and y-axis should represent the infection rate β and the number of recovered nodes at the end of the diffusion, respectively. Plot the results when $\beta = \{0.001, 0.002, 0.005, 0.01, 0.02, 0.05, 0.1\}$, while fixing the recovery rate δ and the number of initial active nodes as 0.8 and 50, respectively. Briefly describe your observation.

3.2 Effect of the number of the initial active nodes N_0 [25 points]

Your task is to visualize the relationship between the final number of recovered nodes and the number of the initial active nodes N_0. The x-axis and y-axis should represent the number of the initial active nodes N_0 and the number of recovered nodes at the end of the diffusion, respectively. Plot the results when $N_0 = \{10, 20, 50, 100, 200, 500\}$ and $(\beta, \delta) = \{(0.005, 0.8), (0.02, 0.6)\}$. Briefly describe your observation.

4 Notes

- Your implementation should run on TA's desktop within **5 minutes** for a single combination of the parameters.

- You may encounter some subtleties when it comes to implementation. Please come up with your own design and/or contact Kyuhan Lee (kyuhan.lee@kaist.ac.kr) and Jihoon Ko (jihoonko@kaist.ac.kr) for discussion. Any ideas can be taken into consideration when grading if they are written in the *readme* file.
- In `simulate.py`, do not modify the code outside of the "TODO" regions. For example, do not import additional python libraries outside "TODO" regions.
- For your convenience, we provide `plot.py` to draw the figures for the report, and you don't need to submit this file.

5 How to submit your assignment

1. Create hw3-[your student id].tar.gz, which should contain the following files:
   - **SIR.py**: this file should contain your implmentation.
   - **report.pdf**: this file should contain your answers in Section 3 (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. All **3 files** should be included in a **single folder**.
3. Make sure that no other files are included in the tar.gz file.
4. Submit the tar.gz file at KLMS (`http://klms.kaist.ac.kr`).