Course project (52 pts)

- The deadline for handing in your solutions is December 20th 2019 23:55.
- Return your solutions (one `.pdf` file and one `.zip` file containing Python code) in MyCourses (Assignments tab). Additionally, submit your pdf file also to the Turnitin plagiarism checker in MyCourses.
- Check also the course practicalities page in MyCourses for more details on writing your report.

- Note that you need to get at least 25 points from the project in order to pass the course!
- There is no code template similar to the normal exercises in this course.
- Some of the tasks are marked as bonus tasks. These are more open ended tasks which might be more difficult and time consuming. They can have multiple valid solutions and allows for some creativity. It is not expected that all students complete the bonus task. Further, we recommend that you complete the normal tasks before attempting the bonus tasks.
- Discussing project solutions with other students is allowed when properly indicated in the report. Each student should nevertheless return his/her own report.

Introduction

In the project work, your task is to implement a Susceptible-Infected (SI) disease spreading model and run it on top of a temporal network from air transport data, containing information on the departure and arrival times of flights. You will study the dynamics of spreading and how it depends on where the process starts as well as the infectivity of the disease, and use static-network centrality measures to understand the roles that specific nodes play.

Model specifications

In the SI model, each node is either Susceptible or Infected. When an Infected node is in contact with a Susceptible node, the Susceptible node may become infected with some probability $p \in [0, 1]$, reflecting the infectivity of the disease. Infected nodes remain Infected forever.

In our model that mimics the spreading of disease through the air transport network, nodes are airports and time-stamped connections are flights between them. Initially, only one airport node (called *the seed* node) is set to the Infected state, and all other airports are susceptible. Now, following the SI process, a Susceptible node may become Infected when an airplane originating from an Infected node arrives to the airport. Note that the source airport needs to be Infected already at the time of the flight's departure! If this condition is met, the destination airport may become Infected with probability $p \in [0, 1]$, and if so, it remains in this state for the rest of the simulation.

Data description

The data that are used in the project are available at the course MyCourses page in one `.zip` file. The flight data that you will use to perform your simulation are located in the file `events_US_air_traffic_GMT.txt`, where each row contains the following fields:

```
1st column -> Source [0-278]
2nd column -> Destination [0-278]
3rd column -> Start Time (GMT) [seconds after Unix epoch time]
4th column -> End Time (GMT)
5th column -> Duration [Same as (EndTime-StartTime)]
```

The aggregated weighted network `aggregated_US_air_traffic_network_undir.edg` is constructed based on the event data, so that the weight of each link corresponds to the number of flights between the nodes. This network is static and undirected. Additionally, you can find the information about the airports in file `US_airport_id_info.csv`.

Provided code

In this project, you will write your own code from scratch. Start by importing the usual packages, creating useful lists and dictionaries and loading the data with functions given in general hints.

To help you in verifying that your code actually works, we provide a Python module `si_animator.py`. The use of this module is **optional**. In the module, you can find a function called `plot_network_USA`, which is used in Task 7. This function can be useful.

As an extra visualization, you can find a function called `visualize_si` which animates the air transport using `matplotlib`. For the animations to work, the background image `US_air_bg.png`, the airport id-info file `US_airport_id_info.csv` and the events file `events_US_air_traffic_GMT.txt` need to be located in the directory where you are running Python. See the documentation within the module for further usage info. Note that this visualization is NOT needed for completion, it's just for your benefit.

General hints on implementing the model:

- Please familiarize yourself with all tasks before you begin coding—this helps you to write your code so that the basic implementation works for all tasks, rather than writing code for Task 1 and only later noticing that Task 2 needs something else.
- What you want to have is a) the times of infection of each node, and b) the timeline of the number of infected, for each run.
- The fact that flights have duration complicates things a bit. It helps if you use a dictionary whose keys are nodes and values are their times of infection. Initialize the dictionary as empty, *e.g.* `infection_times={}`, except for the seed node (you can set the data set's first flight's departure time as the infection time for the seed). Then, whenever a node becomes infected, set its time of infection to the arrival time of the flight carrying the infection.
- The most transparent and straightforward way to implement the model is to first sort all flights in increasing order of departure time, and then go through them in sequence. Is the source node infected at the current time (look this up in your dictionary)? If it is,

there are two options: a) the target node is susceptible (if `node in infection_times` is `false`), in which case you should infect the target node with probability p and add it to the dictionary, setting its infection time to the flight's arrival time, and b) the target node is already infected—if so, check if the current flight's arrival time is earlier than the node's infection time in the dictionary, and if, update it to the arrival tie. The latter is because there may well be a shorter flight that carries the infection and arrives earlier than some longer flight that is responsible for the time of infection.

- The above dictionary contains all necessary information, but you will also need the timeline of the number of infected. The easiest way to get this is to first run the simulation to get the infection times dictionary, and then extract its values (the infection times) to a list (`infection_list=list(infection_times.values())` and sort the list in increasing order. Now, the i'th element of the list is the time when i nodes have been infected.
- Pro tip: write the whole thing as one function that takes in the seed node id and p and returns the number of infected nodes as function of time (see above) as well as a dictionary of the infection times of with nodes as keys. Reuse this function for all tasks.
- Even more pro tip: have this function also read in the desired time intervals (see Task 2) for the prevalence curves, so that you can within the function count the fraction of infected at those time points from your sorted infection time list, and return these fractions as a list.
- For reading in the csv data, there are many options, such as `numpy.genfromtxt`, the `read_csv` function from the `pandas` package, or the built-in `csv` library in the Python standard library.
  For example, with `numpy.genfromtxt` use something like
  `event_data = np.genfromtxt('events_US_air_traffic_GMT.txt', names=True, dtype=int)`
  `sorted_data = event_data.sort(order=['StartTime'])` for loading and sorting the data.
- You may freely use any functions from `networkx`, `matplotlib`, `numpy` and `scipy` that you find useful.

Tasks:

Task 1: Basic implementation (5 pts)

Implement the SI model using the temporal air traffic data in `events_US_air_traffic_GMT.txt`. Use the provided visualization module to check that your implementation works reasonably. Assume first that $p = 1$, *i.e.,* the disease is always transmitted.

a) (5 pts) If Allentown (node-id=0) is infected at the beginning of the data set, at which time does Anchorage (ANC, node-id=41) become infected?
   Hint: The time point should fall within the range `1229290000-1229291000`.

Task 2: Effect of infection probability p on spreading speed (5 pts)

Run the SI model 10 times with each of the infection probabilities [0.01, 0.05, 0.1, 0.5, 1.0]. Again, let Allentown (node-id=0) be the initially infected node. Record all infection times of the nodes, and answer the following questions:

a) (4 pts) Plot the averaged prevalence $\rho(t)$ of the disease (fraction of infected nodes) as a function of time for each of the infection probabilities. Plot the 5 curves in one graph.

Hints:

- For averaging, things are easier if you divide the whole time span (from the first departure to the last arrival) into equal-sized steps using *e.g.* `np.linspace`)
- For each step, for each of your runs, count the number of nodes infected before this step and normalize by the total number of nodes (see the pro tip above in general hints). Then, average over these values over the 10 iterations.

b) (1 pt) For which infection probabilities does the whole network become fully infected? What are the stepwise, nearly periodic "steps" in the curves due to?

Task 3: Effect of seed node selection on spreading speed (5 pts)

Next, we will investigate how the selection of the initially infected seed node affects the spreading speed.

a) (3 pts) Use nodes with node-ids [0, 4, 41, 100, 200] (ABE, ATL, ACN, HSV, DBQ) as seeds and $p = 0.1$, and run the simulation 10 times for each seed node. Then, plot the average prevalence of the disease separately for each seed node as a function of time (recycling your code for Task 2).

b) (1 pt) You should be able to see differences in spreading speed. Are these differences visible in the beginning of the epidemic or only later on? Why?

c) (1 pts) In the next tasks, we will, amongst other things, inspect the vulnerability of a node for becoming infected with respect to various network centrality measures. Why will it be important to average the results over different seed nodes?

Task 4: Where to hide? (7 pts)

Now, consider that you'd like to be as safe from the epidemic as possible. How should you select your refuge? To answer this question, run your SI model 50 times with $p = 0.5$ using different random nodes as seeds and record the *median* infection times for each node.

In this task, you can use the pre-built static network (`aggregated_US_air_traffic_network_undir.edg`) to compute the various centrality measures using the ready-made networkxx functions.

a) (4 pts) Run the 50 simulations, and create scatter plots showing the median infection time of each node as a function of the following nodal network measures:

i) *unweighted* clustering coefficient c
ii) degree k
iii) strength s
iv) *unweighted* betweenness centrality

b) (1 pt) Use the Spearman rank-correlation coefficient [1] for finding out which of the measures is the best predictor for the infection times[1].

Hint: `scipy.stats.spearmanr`

c) (2 pts) Discuss your results for each network centrality metric. In particular, discuss the ranking of the network measures as measured by the median infection time.

Task 5: Shutting down airports (10 pts)

Now take the role of a government official considering shutting down airports to prevent the disease from spreading to the whole country. In our simulations, the shutting down of airports corresponds to immunization: an airport that has been shut down can not become infected at any point of the simulation.

One immunization strategy suggested for use in social networks is to pick a random node from the network and immunize a random neighbour of this node. Your task is now to compare this strategy against seven other immunization strategies: the immunization of random nodes and the immunization of nodes that possess the largest values of the six measures of centrality/importance that we used in task 4. In this exercise, use $p = 0.5$ and average your results over 20 runs of the model for each immunization strategy (160 simulations in total).

To reduce the variance due to the selection of seed nodes, always use same seed nodes when investigating each immunization strategy: first select your immunized nodes, and then select 20 random seed nodes such that none of them belongs to the group of immunized nodes in any of the 8 different strategies.

a) (5 pts) Adapt your code to enable immunization of nodes, and plot the prevalence of the disease as a function of time for the 8 different immunization strategies (social net., random node, and 6 nodal network measures) when 10 nodes are always immunized.

Hint: If you use a function to run one simulation, add a list (or set) of nodes as an optional input, with an empty list (or set) as the default value. Then, when about to infect a node, check if the node is in the list/set, and if, do not infect.

b) (2 pts) Discuss the ranking of the immunization strategies. In particular, compare your immunization results with the results you obtained in the previous task (Task 4). Are there some measures that are bad at predicting the infection time but important with regards to immunization? Or vice versa? Why?

c) (2 pts) The pick-a-neighbour immunization strategy probably worked better than the random node immunization[2]. Let us try to understand why.

- First, if the degree distribution of the network is $P(k)$, what is the probability of picking a random node of degree k?
- What is the expected outcome if you then pick a random neighbour of the random node (hint: see lectures 3 and 4)?
- Consequently, which of the strategies is expected to be more effective and why?

[1] We use Spearman rank-correlation coefficient instead of the linear Pearson's coefficient, as we can not assume the dependency between the average infection time and different nodal network measures to be linear.

[2] However, due to the randomness of the selection process there is a small probability that this was not the case for you.

d) (1 pt) Although the social network immunization strategy outperforms the random immunization, it is not necessarily as effective as some other immunization strategies (and there is random variation). Nevertheless, **explain** shortly, why it still makes sense to use this strategy in the context of social networks?

Task 6: Disease transmitting links (8 pts)

So far we have only analyzed the importance of network nodes—next, we will discuss the role of links. We will do this by recording the number of times that each link transmits the disease to another node. So adapt your code to recording the (undirected) static links which are used to transmit the disease. You can do this by storing for each node from which other node it obtained the infection, or alternatively, by using a dictionary where the keys are tuples corresponding to a link, `airport_a,airport_b` with id's ordered so that a<b, or by using a list of links, and the values are incremented whenever the infection passes through the link. Run 20 simulations using random nodes as seeds and $p = 0.5$. For each simulation, record which links are used to infect yet uninfected airports (either susceptible airports or the infecting flight arrives before the already recorded infection time).

a) (4 pts) Run the simulations, and compute the fraction of times that each link is used for infecting the disease (f_{ij}). Then use the provided function `plot_network_USA` which can be found in `si_animator.py` to visualize the network on top of the US map (see the example code given in the function). Adjust the width of the links according to the fractions f_{ij} to better see the overall structure. Compare your visualization with the maximal spanning tree of the network.

b) (1 pt) What do you notice? How would you explain your finding?

c) (2 pts) Create scatter plots showing f_{ij} as a function of the following link properties:

   i) link weight w_{ij}

   ii) *unweighted* link betweenness centrality eb_{ij} (`edge_betweenness_centrality` in networkx)

   Compute also the Spearman correlation coefficients between f_{ij} and the two link-wise measures.

d) (1 pt) Explain the performance of the two link properties for predicting f_{ij}.

BONUS task (5 pts)

None of the above measures were extremely good for predicting f_{ij}. As a bonus task, you can come up with a measure of your own, or find out an appropriate measure from the literature that you think is better for predicting f_{ij} than the two measures listed above. Motivate selection of method, and perform similar investigations as with the other link properties. Especially evaluate the Spearman correlation between the measure you developed and f_{ij}.

Note:
Remember that your measure should be based only on the information contained in the weighted network *i.e.* do not use geographical or time-stamped air traffic data for devising your measure.

Grading guideline for the bonus:
Good effort, 1 pt; a solution that is better than the three other measures min. 2 pts; for more

sophisticated/elaborated/principled attempts and efforts: max. 5 pts. Especially novel, well motivated approaches will be valued (even if the end results would not be outstanding). If you end up trying multiple measures, reporting more than one measure can gain you more points.

Task 7: Discussion (2 pts)

Even though extremely simplistic, our SI model could readily give some insights on the spreading of epidemics. Nevertheless, the model is far from an accurate real-world estimate for epidemic spreading.

Discuss the deficiencies of the current epidemic model by listing at least four (4) ways how it could be improved to be more realistic.

Hint: Check *e.g.* Ref. [2] for ideas.

BONUS task: (max. 5 pts)

You are now free hand to extend the analysis of the SI model in any way you wish. You may use the SI model and compare its results to some more detailed network analyses and/or extend it to be more realistic in any way you want. For the latter, Refs. [2] and [3] can be helpful. If you decide to extend your model, provide simulation results that illustrate the new behaviour of your model (show how your improved model differs from the simple SI model). Remember to discuss your findings.

Feedback (1 pt)

To earn one bonus point, give feedback on this exercise set and the corresponding lecture latest two day after the report's submission deadline.
Link to the feedback form: `https://forms.gle/FcUKPa4vdSS3W9286`.

References

[1] [Online]. Available: http://en.wikipedia.org/wiki/Spearman%27s_rank_correlation_coefficient

[2] R. Pastor-Satorras, C. Castellano, P. Van Mieghem, and A. Vespignani, "Epidemic processes in complex networks," *arXiv preprint arXiv:1408.2701*, 2014.

[3] A. A. Alemi, M. Bierbaum, C. R. Myers, and J. P. Sethna, "You Can Run, You Can Hide: The Epidemiology and Statistical Mechanics of Zombies," p. 12, 2015.