Written Examination
M1 CompuPhys: Introduction to python language
24th of November 2022

What if COVID19 epidemic happened in GAME OF THRONES

- What is Game of Thrones about - Game of Thrones (GOT) is an American television series published by HBO which is an adaptation of George R. R. Martin's fantasy novels. The story takes place on two fictional continents, Westeros and Essos and the different episodes picture several story arcs. The show depicts, first, political conflicts between noble families claiming the Iron throne to reign over the seven Kingdoms of Westeros, or fighting for their independence. Second, the story of the last descendant of the realm, who has been exiled to Essos, claiming the throne. The third arc is related to a military order, the Night's Watch, whose objective is to defend the realm against threats from beyond Westeros's northern border. This show contains 8 seasons.

- Dataset and Network Construction - This examination is based on a dataset coming from fan-generated scripts from `https://genius.com/artists/Game-of-thrones`. For each of the 8 seasons of GOT, the dataset provides the list of characters present (or mentioned) in at least one episode (file location: ./data/got-sX-nodes.csv containing information on their Sovereign House affiliation, where X denotes the season ID in $[1, 2, \ldots, 8]$), as well as the list of interactions between pairs of characters (file location: ./data/got-sX-edges.csv, where X denotes the season ID).

These interactions can be:

1. Character A speaks directly after Character B
2. Character A speaks about Character B
3. Character C speaks about Character A and Character B
4. Character A and Character B are mentioned in the same stage direction
5. Character A and Character B appear in a scene together

All these interactions have been aggregated and there is no way to distinguish them. For instance the line *ROS,TYRION,17,1* in the file got-s1-edges.csv, related to season 1, tells that there were 17 interactions between Tyrion Lannister and Ros, however there is no mention about the interaction type.

- Exam Topic - We call GOT interaction network (GOT-NET) the graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ consisting in a set of nodes $\mathcal{V}$ representing the GOT's characters and a set of links $\mathcal{E}$ based on the previous interaction types. We can consider $\mathcal{G}$ as a weighted graph. Indeed, the interaction data provides the number of times a pair of characters has interacted during a specific season. The exam is constructed in two parts, I^{st} a topological analysis of the network, analysis of the links, node's centrality measures and network resilience against attacks. In the II^{nd} part of this exam, a contagion through the network will be modeled and analyzed.

Tab. 1: List of needed python packages.

networkx	graph structure and graph analysis tools
pandas	data analysis
numpy	linear algebra
matplotlib.pyplot	figure generation
random	(pseudo-)random number generator

Important: Files have to be compressed in a .zip file and sent by email at celestin.coquide@utinam.cnrs.fr by the 1st of December 2022. These files must contain: the report (PDF format) including answers, figures and tables, the generated files (lists and figures) and all python scripts, including ones that generate figures. Note that not properly annotated scripts could count for two negative points

I Topological analysis (12 points)

Exercise 1: From interactions to a network

1.1 Based on the interaction types provided in the header of this document, group these interactions into directed and undirected ones. Give some arguments pointing toward the fact that GOT-NET has to be considered as an undirected graph.

1.2 Give the definition of the adjacency matrix A of a network.

Exercise 2: Centrality measures in the unweighted GOT-NET

2.1 What is a node centrality ?

2.2 Give the definition of the degree $D(i)$ of any node i and its meaning in the context of GOT-NET.

2.3 Give the definition of the closeness $C(i)$ of any node i and its meaning in the context of GOT-NET.

2.4 Give the definition of the betweenness $B(i)$ of any node i and its meaning in the context of GOT-NET.

2.5 For each season, give the 5 most important GOT's characters in terms of degree, closeness and betweenness centrality measures? (represent them in a table.)

2.6 For each season, create a file that contains the list of GOT's character with their associated centrality scores.

Exercise 3: Centrality measure in the weighted GOT-NET
In the context of a weighted network, one can use the weighted degree $\tilde{D}$ instead of the degree. It is defined as follows:

$$\tilde{D}(i) = \sum_{j \in \mathcal{V}} w(ij) \tag{1}$$

where $w(ij)$ is the weight associated to the link between nodes i and j.

3.1 Give the meaning of the weighted degree for GOT-NET.

3.2 For each season, give the 5 most important GOT's characters in terms of weighted degree centrality?

3.3 For each season, create a file consisting in the list of GOT's characters with their associated weighted degree.

3.4 Generate a figure representing the distribution of the links' weight. Note that each season-related distribution has to be in the same figure (you can choose different color.)

Exercise 4: Unweighted GOT-NET resilience to random node attacks
A random node attack is the fact that we remove a randomly chosen node from the network, as well as its links. A network is said resilient if a several portion of its nodes and links need to be removed to disconnect it. It is possible that some nodes composing a network are only connecting with each other, we call such a sub part of the network a connected component, denoted by c. The network connectivity can be measured with its average connected component size $|\bar{c}|$. The lower is this number, the more fragmented is the network. The networkx Python's package has a function permitting to construct the set of connected components $\mathcal{C}$ composing a network. This function is networkx.connected_components whose documentation is available at `https://networkx.org/documentation/stable/reference/`, section Algorithms>Component>Connectivity.

4.1 Based on the package documentation, tell what are the input and output of this function.

4.2 For each season give the number of connected components and their relative size $|c|/N$ in a table. The notation $|c|$ denotes for the number of nodes composing the connected component c, and N is the number of nodes in the network.

4.3 When there is more than one connected component, draw the network corresponding to the smallest one. (Node colors represents House affiliation, node size the number of connections, and label the character name.)

The nodes failure simulation consists in selecting randomly a fraction f of nodes and removing corresponding nodes and their connections. In order to select a random node from graph's nodes use the random package and the function random.choice. This function takes as input a list. The list of nodes can be obtained from the command list(G.nodes()), where G is a networkx graph structure. By using the command G.remove_node("i"), we remove node "i" from G. Note that the associated links are automatically removed from G.edges().

4.4 In the context of the first season, plot the evolution of the average size of connected components $|\bar{c}|$ with the fraction of attacked nodes f. f evolves in the range $[0, 1]$ with a step $df = 0.05$.

4.5 Is there a transition observed from a connected phase to a disconnected phase? If so give f_c, the critical fraction of nodes where the transition arrives.

Due to the non-deterministic nature of this algorithm, we need to run several random simulations in order to have a more realistic analysis.

4.6 In the context of the first season, plot the averaged evolution of $|\bar{c}|/N$ over $N_{trial} = 100$ random simulations with f.

4.7 Is there a transition observed from a connected phase to a disconnected phase? If so give f_c, the critical fraction of nodes where the transition arrives.

4.8 Is this network resilient against random attack?

II Susceptible-Infectious-Susceptible (SIS) Model (8 points)

We are interesting in modeling a pandemic, such that COVID19, in the GOT's universe. SIS model is one of the easiest contagion model. In this model two types of individual coexist in a population, susceptible s and infected i people. s are not infected yet whereas i have contracted the disease. The epidemic propagation process is the following.

1 A random set of initial infected individuals I is chosen

2 During the contagion, any infected individual can propagate the disease to one of its neighbors (ex: physical contact) with a probability β

3 During the contagion, an infected individual can become susceptible again with a probability μ

4 The contagion stops after an arbitrary number of steps

The epidemic rate of a virus, $\lambda = \beta/\mu$, is a biological observable quantifying the propagation rate of a virus.

This model can be applied on a network of social interaction such as our GOT-NET. In this case any infected neighbors consists in the list of nodes it has connections with. The algorithm is the following:

I Construct the set of initial infected nodes I

II Select randomly a node and check its infectious state (1=infected, 0=susceptible)

- a If it is infected:
  - a.1 For each of its neighbor, pick a random float in $[0, 1]$, if it is $\leq \beta$, then the neighbor can be infected.
  - a.2 After having tested all of its neighbors, pick a random float in $[0, 1]$, if it is $\leq \mu$, the node becomes susceptible
- b If it is not infected:
  - b.1 Select another node

III The current time step ends when all nodes have been visited

IV If the number of steps $\tau \geq \tau_{max}$ stop the process

Algorithm 1

Exercise 1: SIS applied to the unweighted GOT-NET of the first season
We consider three different pairs of parameters (β, μ): (0.5,0.5), (0.3,0.7) and (0.7, 0.3) and a maximal number of step $\tau_{max} = 100$.

1.1 For each set of parameters, generate the list of initial infected characters and the list of final infected characters in two different files.

1.2 For each set of parameters, plot the evolution of the ratio of infected characters with time step τ.

1.3 Comment these figures and tell if the epidemic is fast spreading or not.

Exercise 2: SIS applied to the weighted GOT-NET of the first season
We consider three different pairs of parameters (β, μ): (0.5,0.5), (0.3,0.7) and (0.7, 0.3) and a maximal number of step $\tau_{max} = 100$.

2.1 Modify algorithm 1 in order to take into account the number of contacts between characters.

2.2 For each set of parameters, generate the list of initial infected characters and the list of final infected characters in two different files.

2.3 For each set of parameters, plot the evolution of the ratio of infected characters with time step τ.