Course Project

Complex Systems and Network Science

Corso di Laurea Magistrale in Informatica

A.Y. 2023-2024

Dipartimento di Informatica — Scienza e Ingegneria
Università di Bologna

Ozalp Babaoglu Francesco Alfieri

November 16, 2023

1 General Information about the Project

As part of your course requirement, you are to complete the project described below, which must be carried out individually. Submission of the project for evaluation must be done via email to the address: `francesco.alfieri5@studio.unibo.it`.

The deadline for submission is **23:59:59 hours on 04 January, 2024**.The email must have the subject field as *CSNS Project 2024* and must be sent from your University address (`name.surname@studio.unibo.it`).
You will receive a confirmation message within a few days of your submission. The email should contain an archive (in .zip or .tar.gz format) containing the following:

1. The source code that was developed (either in NetLogo, MatLab or PeerSim);
2. A short paper (up to 12 pages), in PDF format, describing the model that was implemented, the experiments that were carried out using it, and a discussion explaining the results that were obtained.

Your full name, email address and student ID number (matricola) must be included in all of the source files, in the paper, and in the submission email that you send. The source code should be well documented and formatted, following good programming practices. The paper must be written in English, and should be structured like a technical paper, thus containing a title, abstract and bibliography. It is strongly suggested that you follow the Springer format for *Lecture Notes in Computer Science* (LNCS). Templates are available for both Word[1] and LaTeX[2]. You can use any text processing system you prefer (even though LaTeX is suggested) to write the paper as long as you submit the result as a PDF file.

The project must be done *individually*: no sharing of papers or source code is permitted. You are, of course, encouraged to discuss issues and solutions with fellow students or with the instructors.

2 Grading Policy

For your project to be satisfactory, it must satisfy the following requirements:

- The project must implement the specifications that follow. You are allowed (and encouraged) to apply modifications and extensions to the project, but they must be proposed to the instructors beforehand and approved by them.
- The model's implementation, and all of the related simulations and experiments, must be carried out using either the NetLogo, MatLab or PeerSim

[1] Link to .doc template
[2] Link to .tex template

software systems. If PeerSim is chosen, the cycle-driven simulation engine should be used, and the simulator must be configurable the standard PeerSim configuration file.

- Your paper must thoroughly describe the model that was implemented and justify all significant design decisions and extensions that were applied to it. You should also discuss the expected behavior of the model, by making previsions. Most importantly, you have to explain the experiments that you performed in terms of methodology and the results that you obtained. Significant implementation details can be inserted, if important in the context of the model, but should otherwise be kept as comments in the code itself.

You are encouraged to focus on a simple model and to apply extensions to it only if you completely understand the behavior of the base model. This can be achieved by working in modular fashion, thus incrementally (and carefully) adding new features, enriching your model. Ending up with a complex, unpredictable and difficult to understand model is very easy. On the contrary, you should prove through your experiments that you fully understand the behavior of your model and that you can interpret the results you obtained, and are able to relate them with real-world phenomena.

3 Introduction to the Project

The purpose of this year's project is to study, implement and analyze the evolution of an epidemic throughout a network with discrete timesteps. In the CSNS course, you have studied contagion and the evolution of epidemics by means of differential equations, for example with SIR, SEIR models and variants. In this project, your goal is to build a simulation of an epidemic with discrete timesteps by means of adjacency matrices.

3.1 Adjacency matrices and network theory

Definition 1. *Suppose $G = (V, E)$ is a network where $V = \{1, 2, \ldots, n\}$. For $1 \leq i, j \leq n$ we define*

$$a_{ij} = \begin{cases} 1, & (i,j) \in E \\ 0, & (i,j) \notin E. \end{cases}$$

Then the square matrix $A = (a_{ij})$ is called the adjacency matrix *of G.*

Adjacency matrices are a powerful tool to represent network topologies and processes that take place on a given graph/network. For example, the out-degree of node i is the sum of values in the i-th row of A. Moreover several measures of nodes' centralities can be computed starting from A (the most important ones can be obtained by means of spectral properties of matrices). More useful informations about adjacency matrices can be found in [1].

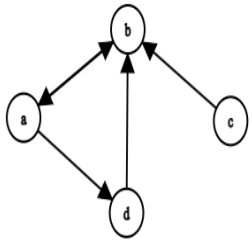

Figure 1: graphical representation of the graph from the example below

A remarkably important property for our purposes is that the adjacency matrix of a graph allows to represent the flow of *scores/values* along the graph itself by means of matrix multiplications.

Consider for example the directed graph in Figure 1 with adjacency matrix

$$A = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}.$$

Assume moreover that the nodes $1, 2, 3, 4$ have respectively scores $\begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix}$. Then, if we *propagate* these scores along the edges of G, we expect at the next time step to obtain the following scores: $\begin{pmatrix} b \\ a+c+d \\ 0 \\ a \end{pmatrix}$.

This fact holds in general: indeed, if we want to propagate a vector of scores v_0 along the edges of a graph with adjacency matrix A, we just need to compute the new vector of values $v_1 = A^T v_0$.

Of course, the edges of a network can be weighted (for example in order to represent the probability of values flowing from one node to another). This is a generalization of the previous case, and in this situation, the entry A_{ij} of the adjacency matrix would assume as a value the weight of the edge (if present) going from node i to node j.

3.2 A simple example of epidemic

Let $G = (V, E)$ be a graph with $V = \{1, 2, \dots, n\}$ and adjacency matrix A, and let $p(0) = \begin{pmatrix} p_1(0) \\ p_2(0) \\ \vdots \\ p_n(0) \end{pmatrix}$ be the vector which describes the *levels of infection* for each node at the time step $t = 0$. Let $\beta \geq 0$ and $0 \leq \delta \leq 1$ be two parameters which represent respectively an *infection rate* and a *recovery rate*. With these premises, a node i can be considered healthy at time t if the respective value $p_i(t)$ is low (a threshold can be set), and the *global* infection level can be represented by $s(t) := \sum_i p_i(t)$.

In order to obtain the levels of infection at time $t + 1$ we need to compute the vector $p(t + 1)$: this would be an appropriate combination of the previous infection level and the flow of the previous infection level through edges:

$$p(t + 1) = (1 - \delta)p(t) + \beta A^T p(t).$$

Thus, if we let $M := (1 - \delta)I + \beta A^T$, we simply obtain $p(t + 1) = Mp(t)$.

This gives complete information about the infection levels for all nodes at each time step k, given $p(0)$: in fact we get $p(k) = M^k p(0)$. Moreover, it can be shown under some general hypotheses (or it can be experimentally observed) that the ratio $\frac{s(t+1)}{s(t)}$ converges to a finite value $\rho(M) > 0$, which is the *spectral radius* of M. This value depends on the parameters α, β, and on the topology of the network given by A. If $\rho(M) < 1$, then the epidemic will tend to extinguish, if $\rho(M) = 1$, the global level of infection will stabilize, and otherwise it will grow indefinitely.

Observe that this model is completely deterministic and, in principle, would not require any simulations.

4 Project requirements

The example above is too simplistic: the behavior of the model is completely deterministic and equilibria can be found analitically. For your project you are required to build a simulation of a more complex model which should inlcude several sources of randomness, in order to make its evolution less predictable.

The following list of items is a checklist of what should be done in your project. Note that you have to adhere to this general structure, but you are allowed and encouraged to make slight changes (you might be forced to, depending on the actual tool you decide to use) in order to enrich you model. You can refer to the next section for some suggestions.

1. Select the topology of the network and instantiate the corresponding graph.
   (a) You are required to implement at least one of the Erdos-Renyi, Watts-Strogatz and preferential attachment models (possibly more than

one). In addition, study at least one network from the real world. You can find datasets for example on *https://snap.stanford.edu/data/*, on *https://icon.colorado.edu/#!/networks* or any other reliable source you prefer. For instance, using a graph from social networks friendships might prove an (overestimated) acceptable approximation of contact tracing networks.

Make sure your network has a reasonable size: it should be large enough to let complex behaviours emerge, but if it becomes *too* large the simulation might require too much time.

(b) Edges should represent contacts that are able to transmit the disease. The values of the adjacency matrix should be weighted in order to represent the frequency of such contacts or the probability with which the disease can spread along the edges, depending on how you decide to model you simulation. Note than not all contacts have the same frequency or probability of making diseases spread. For this reason, you should not assign fixed, predetermined weights to all edges but you should sample randomly from an interval around the intended value. Moreover, the strength of these ties should evolve over time: make sure to change the weights during the simulation.

2. Decide how to represent infected individuals: in the previous example we showed a model with a vector of "levels of infection" , where infected nodes could be identified for example by thresholding. If your edges are weighted with the probability spreading of the disease at each time step, another reasonable option would be to mantain a vector $I \in \{0,1\}^N$, where $I_i = 1$ if and only if node i is infected.

3. Select the method by which the disease should spread at each time step.

   (a) If you rely on a vector of scores as per the example above, then multiplication by the transposed and weighted adjacency matrix and an infection rate parameter is a reasonable idea,.

   (b) If you are using a boolean representation of infected individuals, and weights of entry a_{ij} in the adjacency matrix represent the probability of the disease spreading from node i to node j, then you must instantiate your stochastic matrix $A \in [0,1]^{N \times N}$ into a *transmission matrix* $T = (t_{ij})$, where

$$t_{ij} = \begin{cases} 1, \text{ with probability } a_{ij} \\ 0, \text{ with probability } 1 - a_{ij} \end{cases}$$

   and use T instead of A in order to propagate the infection. In this case the vector of *newly infected* individuals will be obtained simply from $T^T I$, where each entry larger than 1 (if any) is set to 1.

4. Choose how to implement *recoveries* and *recovery rates*. Can infected people become susceptible again (useful in order to model recurring diseases),

or do they gain immunity from future illness? If you implement the latter, be sure to set the corresponding column values in the adjacency matrix to 0 when spreading the disease at subsequent time steps. In the previous example the recovery rate was implemented by subtracting, at each time step, a vector $\delta p(t)$ from the new vector of scores. If you rely on a boolean representation of infected individuals instead, you might want to consider using a parameter (either fixed or dynamic) encoding the probability of recovery at each time step.

5. Initialize the starting setting of the model. You should perform experiments by changing the number and the positions of initially infected individuals. For example, you can start with a random subset of infected individuals or investigate what happens if the initially infected individuals are the ones with larger degrees and the most *central* ones.

6. Decide how many iterations you want to perform. You can either use a predefined number of steps or stop the simulation when certain conditions are met (for example when the changes in some given metrics of the network become small enough).

7. In your report, compare the evolution of your model to SIS/SIR models and to simulations from *https://gabgoh.github.io/COVID/index.html* (where you can find other modeling ideas). Evaluate what are the differences, what your model captures well and what are its limitations. Note that since at each time step each individual typically has a large number of contacts, time steps should represent intervals longer than a single day. Be sure to include meaningful observations about how every parameter you set and/or change across multiple experiments affects the outcome. Observe if there meaningful behaviours appear in the long-term: are there asymptotic equilibria or oscillatory behaviors?

   Keep in mind that this is not a research project. Thus, its scope is inevitably somewhat limited. However, you must make sure to indicate the limitations of your model and how it could be improved further.

5 Hints and suggestions

When building your model try to inlcude more realistic elements and make it more complex. Below there is a non-exhaustive list of ideas you can try to implement in your model. You are not required to model all of them and you can either add more or make substantial changes to them.

- Monitor the evolution of the pandemic throughout the simulation: for example you can keep track with a plot of the number of infected, susceptible, recovered, exposed (...) individuals at each time step.

- The requirements above already make the evolution of the epidemic not deterministic: you can introduce more sources of randomness that may alter the simulation given the same initial configuration.
- Contrary to SIR model (and variants), in this case you have perfect information about the spread of the disease over the entire network. You should make observations by exploiting this fact. For example, you may want to monitor the evolution of the disease over small and/or isolated communities.
- Try varying the parameters for infection and recovery rates. This can be done for instance by repeating experiments with different fixed values, or the values can be modified dynamically during the simulation. You can also include more parameters depending on what phenomenon you want to include in your model. Nevertheless, before adding more and more parameters, it is important that you understand how each parameter already present affects the simulation, so try to repeat experiments varying the parameters (or the schedules of the parameteres if you decide to use dynamic ones) and include a description of their effects in the report.
- Think about practical containment measures that could be applied in order to limit the spread of the disease (e.g. social distancing, vaccines providing either complete or partial immunity, ...) and how they can be included in your model.
- Nodes can have different statuses (e.g. Susceptible, Infected, Recovered, Dead, Vaccinated, Treated, Asymptomatic, Latent, Quarantined, ...). You can keep track of these statuses, and they can in turn affect for example both the topology of the network and the strength of the ties (i.e. the weights of the edges) in the network.
- During the evaluation of your model, you can compare it to real-world epidemics and variants of SIR/SEIR models.

Note that these suggestions are not an exhaustive list of project requirements. In fact, you may exclude some items, and implementing others is encouraged.

However, keep in mind that it is very easy to make the model too intricate by introducing too many features, and as a consequence results may become difficult to interpret. In fact, it is more important to provide precise motivations for your design choices, to carefully analyze the effects of all of the already implemented features on the spreading of the disease and to critically evaluate what you model successfully captures and what could be added to make it more realistic.

References

[1] Estrada E., Knight P. (2015) *A First Course in Network Theory.* Oxford University Press.