UNIVERSITÀ DEGLI STUDI
DI GENOVA

Department of Computer Science

GA - Graph Analytics A.Y. 2018/2019

Social Contagion

Andrea Canepa (S 4185248)

Ninety-nine percent of the people in the world are fools and the rest of us are in great danger of contagion.

Thornton Wilder

4 May 2020

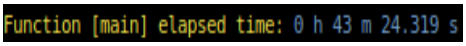
Contents

1 Introduction

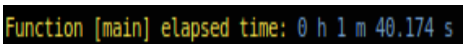
In this last experiment, I try to study how an infection spreads through the network. Here I would like to briefly explain how I decided to organize this report:

- **Structure**: first of all, I explain the *model* and the *algorithms* implemented, paying attention to *optimization techniques* and *boundaries* to aid the algorithm to *converge*;
- **Small Graph**: in this section, I show the results obtained using a crafted *Barabàsi-Albert* graph with $N = 300$ and $M = 4$, where N is the number of nodes and M is the number of edges *to attach* from a *new* node to *existing* nodes;
- **Real Graph**: finally, I show how a *real* network could be influenced by a spreading phenomenon.

Once again, I developed my code in *Python3* language, exploiting the functionalities offered by the ***networkx*** library. Since this time the overall time was reasonable, as we can see in figure 1, I decided to take an *iterative*, *single-threaded* approach.



(a) Real graph experiment



(b) Small graph experiment

Figure 1: Execution timing

A small off-topic note:
In this difficult period, in which coronavirus is the main concern worldwide, this experiment has been very useful to understand how the scientific community is working on the problem. All the countermeasures adopted, such as ***social distancing*** *and the* ***research of patient zero*** *were in line with the theory investigated. Moreover, I found very interesting to align my results with those reported by the media about the actual situation and found them in agreement. I hope that these tough times have taught something to the human race, and all the efforts made to cope with the disease are not forgotten in the next future.*

2 Structure

In this section, I explain the core of the *simulation*. I applied two different ***types of contagion*** on the network, choosing the ***patients zero*** according to the different measures mentioned in the previous works to understand if the most important nodes play a central role during a pandemic. In the following, I enucleate better the key points.

2.1 Epidemic Model

Among different models, I have decided to render the simplest one: the **Susceptible-Infected** (SI). In this simple model, all subject at the beginning are *susceptible* to the infection but yet *sane*. At time $t = 0$, a bunch of nodes become infected and the contagion starts spreading. Once a node becomes *infected*, it cannot heal. This division of subjects it is known as **compartmentalization**, to be more clear:

- ***susceptible*** (S): *healthy* individuals who have *not yet* contacted the pathogen;
- ***infected*** (I): *contagious* individuals who have contacted the pathogen and hence can *infect* others.

2.2 Algorithm

The algorithm that gradually tries to infect the network follows the scheme of a ***breadth-first visit***. But there are some important differences:

- In the beginning, all nodes are marked as *not visited* and *not infected*;
- then some nodes are chosen as *seed* for the pandemic according to some criterion: the visit starts from the *most important one* w.r.t. that criterion;
- a node could be visited *more than once* if it does not get infected;
- two parameters control the termination of the algorithm:
  - ***timestamp***: I put an upper bound on the number of epochs allowed, 75000;
  - ***same_sz***: if the situation does not change after 5000 iterations, it is unlikely that in the future there will be some unpredictable events.

2.3 Contagion Types

I performed my experiments considering two different types of outbreak:

1. **simple**: each individual can be infected with some *probability* (15% if the graph is small, 40% otherwise) each unit of time;
2. **complex**: a node becomes infected if at least a *significant* fraction of its neighbours already have the disease.

In the complex case, it is convenient to imagine the situation as a *game in a strategic form with two players* and then define the ***payoff matrix***. I considered three different states, shown below:

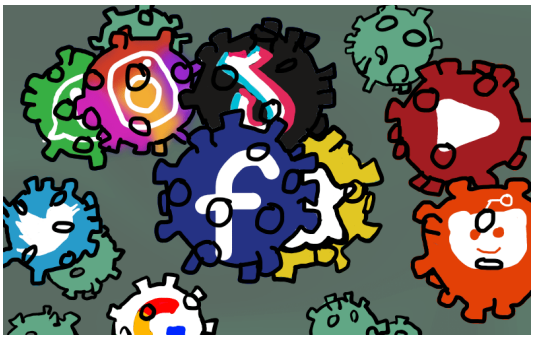
$$P_1^S = \begin{bmatrix} 5 & 0 \\ 0 & 1 \end{bmatrix} P_2^S = \begin{bmatrix} 7 & 0 \\ 0 & 1 \end{bmatrix} P_3^S = \begin{bmatrix} 35 & 0 \\ 0 & 1 \end{bmatrix}$$

with the small graph, and these others with the big one:

$$P_1^B = \begin{bmatrix} 8 & 0 \\ 0 & 1 \end{bmatrix} P_2^B = \begin{bmatrix} 11 & 0 \\ 0 & 1 \end{bmatrix} P_3^B = \begin{bmatrix} 22 & 0 \\ 0 & 1 \end{bmatrix}$$

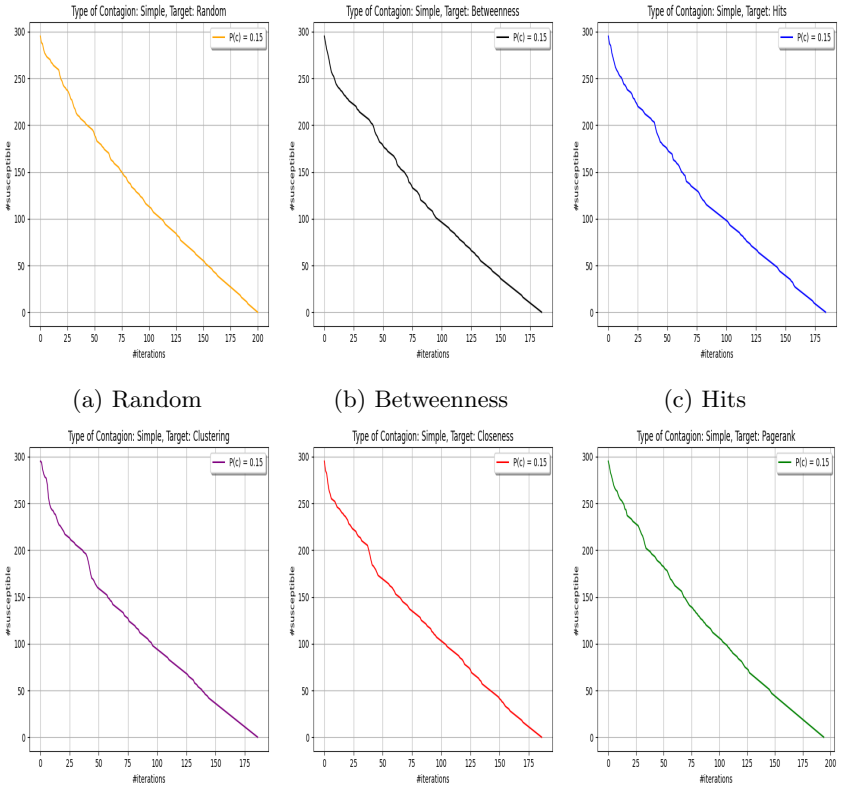
The *threshold* that determines the infection is computed as:

$$\boxed{t_{nk} \geq \frac{P_{n11}^k}{P_{n11}^k + P_{n00}^k}}$$



3 Small Graph

3.1 Simple Contagion



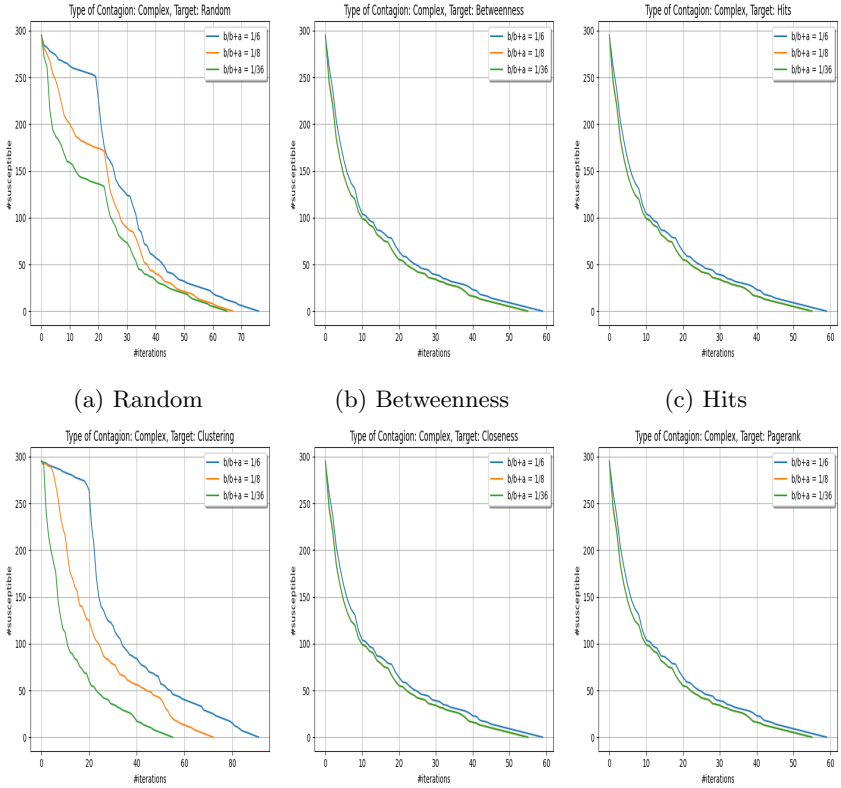
(a) Random (b) Betweenness (c) Hits

(d) Clustering (e) Closeness (f) Pagerank

Figure 2: Different choice for patient zeros in simple contagion

It is clear, especially from the output shown in figure 4, that the best *performance* was obtained thanks to a selection of the first infected based on *PageRank*, while the worst performance was regarding *closeness*. In terms of effectiveness all choices are equivalent since the algorithm have always converged infecting all the nodes in the network.

3.2 Complex Contagion



(a) Random (b) Betweenness (c) Hits

(d) Clustering (e) Closeness (f) Pagerank

Figure 3: Different choice for patient zeros in complex contagion

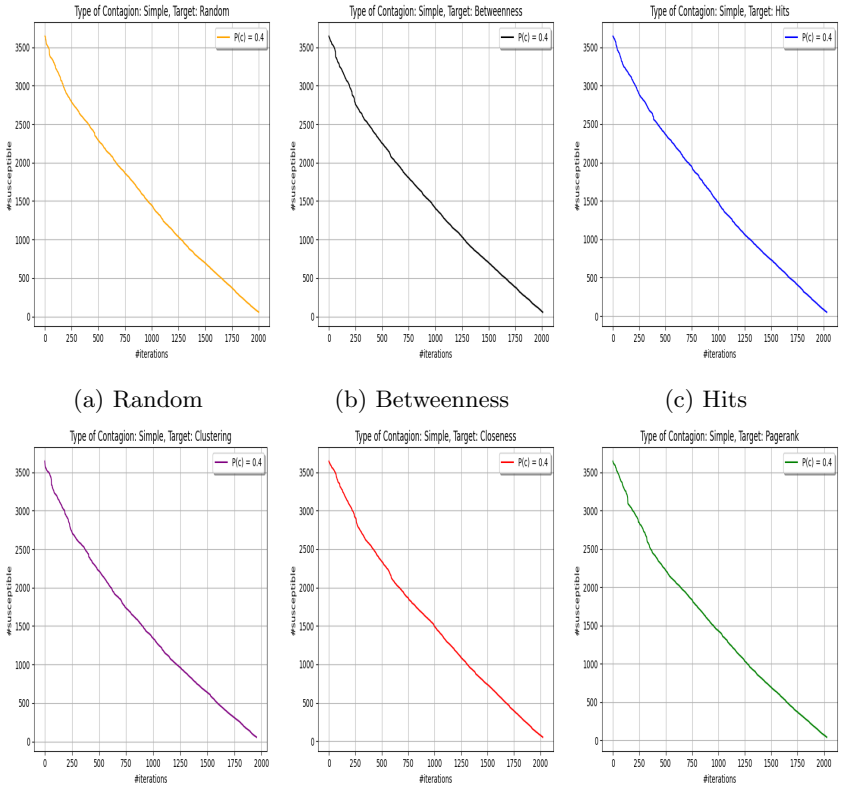
In this case, all trends are very similar, with the exception of *random* and *clustering* ones. The former, in general, obtained the best performance with nearly all values of t_{nk}, whilst the latter was the weakest. Again all trials ended with the totality of the nodes infected. This is probably due to the conformation of the *Barabàsi-Albert* graph.

3.3 Output

Figure 4: Output of execution, small graph

4 Big Graph

4.1 Simple Contagion



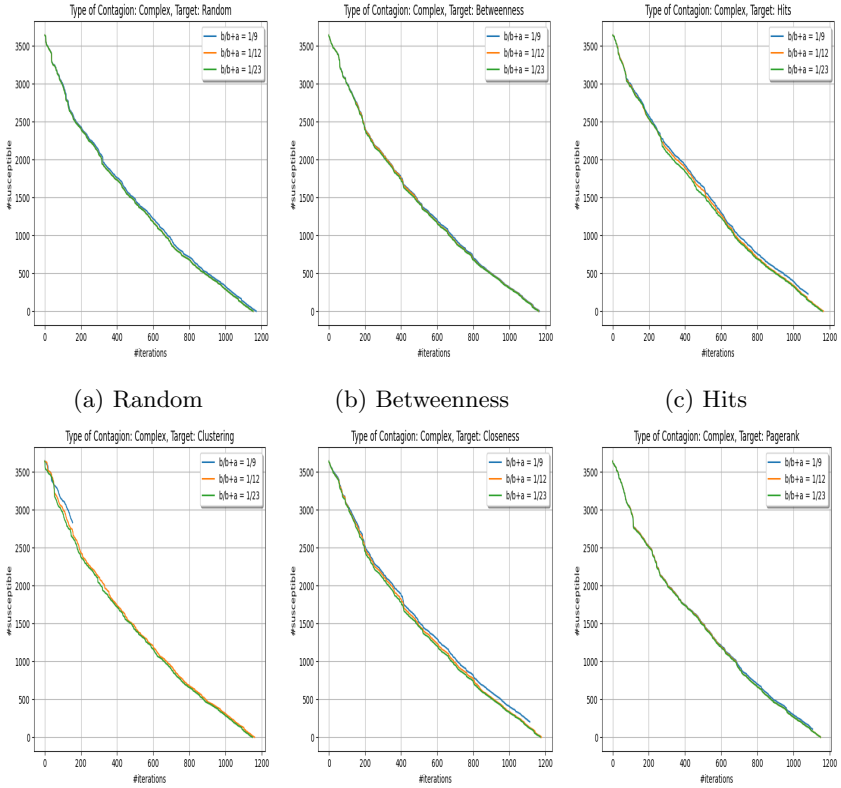
(a) Random (b) Betweenness (c) Hits

(d) Clustering (e) Closeness (f) Pagerank

Figure 5: Different choice for patient zeros in simple contagion

With the real graph, the game changes. In all trials, although the vast majority of the graph was infected, it was not possible to reach the totality of the nodes. Furthermore, the *probability* of infection was quite high. Moreover, I can tell that all the times the algorithm ended because for too many iterations did not happen anything, thanks to the *heuristics*.

4.2 Complex Contagion



(a) Random (b) Betweenness (c) Hits

(d) Clustering (e) Closeness (f) Pagerank

Figure 6: Different choice for patient zeros in complex contagion

Those are the most interesting results. A first interesting fact to notice is that the choice of P_n^B is crucial to achieve the best results in terms of coverage. The most satisfying outcome was obtained with a pandemic based on ***betweenness*** since it was able to contaminate all the nodes for each n in a restrained number of iterations. On the other side, very poor performances are obtained with a ***clustering***-based contagion: after a lot of time units, it was able only to plague only 1/3 of the entire graph. With the lowest value of t, that is due to P_1^B, some nodes remained isolated, this is a clear signal of the presence of some strong *communities* among the network.

4.3 Output

Figure 7: Output of execution, real graph

References

[1] Albert-László Barabási et al. *Network science*, chapter 10, Spreading Phenomena. Cambridge university press, 2016.

[2] Ryan A. Rossi and Nesreen K. Ahmed. The network data repository with interactive graph analytics and visualization. In *AAAI*, 2015.

[3] Benedek Rozemberczki, Ryan Davies, Rik Sarkar, and Charles Sutton. Gemsec: Graph embedding with self clustering. In *Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2019*, pages 65–72. ACM, 2019.