Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Lecture with Computer Exercises:
Modelling and Simulating Social Systems with MATLAB

Project Report

Simulation of Information Spreading in a Facebook Network

Urs Lustenberger , Nino Wili , Patrick Zöchbauer

Zurich
December 2013

Agreement for free-download

We hereby agree to make our source code for this project freely available for download from the web pages of the SOMS chair. Furthermore, we assure that all source code is written by ourselves and is not violating any copyright restrictions.

Urs Lustenberger Nino Wili Patric Zöchbauer

Contents

1 Abstract

2 Individual contributions

3 Introduction and Motivations

3.1 Introduction and fundamental questions

Social Networks like facebook and twitter are growing very fast. Most young people like students do have an account in one ore more of them. Many articles, pictures and videos on the internet have a direct "share" button. It is a very interesting question how information spreads in such networks. First, the internet may have become the fastest and sometimes most important source of "news" and information beside the "traditional" media and second, companies seem to be very interested in "informing" the right people with their commercials.

The simplest way to describe the information spreading uses similarities to epidemiology, which means that the flow of information of one person to another is looked at as an "infection". In this work, we implemented a simple model in a homogeneous way with a set of differential equations and their numeric solutions as well as an agent-based, heterogeneous model, using a real facebook network.
The fundamental questions of this work are:

- Are there relevant differences in the time evolution of the homogeneous and the agent-based network?
- Are there indiviuals which are more important to the spreading of information? Can they be recognized in the sense of position and connectivity in the network?

3.2 Motivation

As the team of authors consists of two chemists and a mathematician, the personal motivations also have a broad range. We were glad to recognize the parallels between our simulations with statistical physics and chemical reaction kinetics...

4 Description of the Model

The process of information spreading has many similarities with epidemiology, so the well known SIR-model was adapted[1]. The "susceptibles", are not aware of the information and are called "ignorants" within this work. "Infected" individuals know about the information and are willing to share it with other people, in other words they spread it and are therefore called "spreaders". The adaptation of the epidemiological term "recovered" is not that straightforward because it is not entirely clear, what that means in this context but they can be looked at persons that are aware of the information but don't want to tell it others. They are called "stiflers".

In the SIR-model as well as in the Agent-based model, the following set of "reaction equations" was used(I: Ignorant, S: Spreader, R: Stifler):

$$\begin{cases} I + S \xrightarrow{\lambda} 2S & (1) \\ S + R \xrightarrow{\alpha} 2R & (2) \\ S + S \xrightarrow{\alpha} S + R & (3) \end{cases}$$

The main difference to the standard SIR-model is that spreaders don't become stiflers spontaneously but this this change is only induced by meeting an other spreader or stifler. Also it can be shown that for $\frac{\lambda}{\alpha} > 0$, i.e. for any positive rate, the number of stiflers is non-zero for large times. That means that there is no epidemic threshold as in the standard SIR-model.

4.1 Homogeneous SIR model

In the mathematical treatment of the model in a homogeneous system, the set of differential equations is expressed in terms of the densities $i(t) = I(t)/N$, $s(t) = S(t)/N$ and $r(t) = R(t)/N$.

$$\begin{cases} \frac{\mathrm{d}i(t)}{\mathrm{d}t} = -\lambda \cdot s(t)i(t) & (4) \\ \frac{\mathrm{d}s(t)}{\mathrm{d}t} = \lambda \cdot s(t)i(t) - \alpha \cdot s(t)[s(t) + r(t)] & (5) \\ \frac{\mathrm{d}r(t)}{\mathrm{d}t} = \alpha \cdot s(t)[s(t) + r(t)] & (6) \end{cases}$$

4.2 Agent-based model

In the inhomogeneous, agent-based model, the individuals are connected in a certain manner (facebook friends in this particular work). Only connected agents are able to meet. If a meeting occurs, transitions are induced at a certain probability depending on the relation between the agents (details are discussed later). Additional to the different connectivity of the agents, they also have a different activity, i.e. different probability to meet somebody.

To convert the reaction equations into an agent-based model, time was discretized and in each time step a series of two steps is performed. First, the agents randomly meet another agent they know or nobody. Only two-agent meetings are possible. In the second step, the status of the agents change corresponding to the situation. There are six possible combinations of ignorants, spreaders and stiflers. Three of them correspond to the "reactions" 1-3, the other ones (I+I, I+R and R+R) have no other influence than "occupying" the agents. The probability λ was chosen to be dependent on the number of common friends. For simplicity, α was a constant.

5 Implementation

5.1 Homogeneous SIR-model

The system of ordinary differential equations described in section ..... was solved in MatLab using ode45. The simulation was carried four times using α to λ ratios between 0.1 and 10. The starting conditions were always S = 383, I = 1 and R = 0.

5.2 Agent-based model

Initial condition At the beginning of each simulation, all agents are ignorant but one, which is a spreader. In the biggest performed simulation, every one of the 384 nodes was the first spreader 10 times.

Determine the meetings (See `talkstep.m` for details.)

In order to determine who meets who, the program goes through the vector of agents (1:N) randomly . With a probability corresponding to the activity of that agent, he may meet somebody . The person he meets is determined randomly and must be one he knows and one which is not already meeting another agent in this time step. Agents with more friends also have more meetings(If you don't take the activity into account).

Status changes in meetings After the meetings are determined, the status of each agent has to change according to the model. In order to be able to determine who infected whom, and how many where infected, a list `infecpath` was created in each round containing the directed edges of infections.

Exit condition The was terminated if further spreading was not possible. This is the case when all current spreaders and all their friends have an activity of zero, including the obvious case that the number of spreaders is zero. For stability, not more than 5000 steps of meeting and status updates were performed in each initiated round.

Running the simulation 3840 rounds were initiated. For every round the first infected person, the number of meetings, the number of direct infections and the number of cumulative infections were saved for further analysis.

5.3 Defining relevant output variables

5.3.1 Importance of an individual

One of our main questions was, if there are persons which are more important to the spreading of the information. The definition of importance in this context isn't trivial. The nearest quantitative variable which might indicate the importance is the number of *direct infections*, which means the sum of all meetings of an infected person which resulted in the infection of his meeting partner. But in our opinion, a better variable are the *cumulative infections*, which means the number of direct infections of a person plus the number of direct infections of all other agents infected by that person and so on. Watts [2] defines a similar value, namely the size of a *cascade*. A cascade is just the sequence of infections initiated by an individual. The size of a cascade is just the number of infections in this cascade. So the value of cumulative infections and the size of the cascade are the same if you look at the person which was infected first (our initial condition). But cumulative infections are in a sense more general.

Cumulative infections With a given `infecpath`, with the directed edges of infections, it is obviously easy to obtain the number of direct infections of every node and also relatively easy to get the number of "cumulative infections", which is the whole subtree of infections of each node (see Details later). The ladder was obtained in a recursive manner in the following way:

```
L=length(infectpath(:,1));
for i=L:-1:1

    p1=infectpath(i,1);
    p2=infectpath(i,2);
    cum_infections(p1)=1+cum_infections(p1)+cum_infections(p2);
end
```

5.4 Getting the network

To get the data for our simulation, we used the so-called Facebook Graph API, a programming tool designed to support better access to conventions on the Facebook social media platform.[1]
Unfortunately, there is no implementation for Matlab, so we programmed the data-fechting in python. For the coding in python we used the facebook sdk[2].

Get access First, one needs to get access to the social graph. Therefore, you need to enter an access token[3].

```
token = raw_input('Enter your Access Token: ')
anonym = input('Do you want to anonymize your Data? Yes (1) or No (0)? ')
graph = facebook.GraphAPI(token)
```

Get graph Having enterd a valid token, we can now access all information this token provide us. For simplicty, we'll only discuss the non-anonymized case in details. [For more details on the anonymization, look at subsection anonymization]

```
profile = graph.get_object("me")
friends = graph.get_connections("me", "friends")
```

In our case, we're interested in data about our own profile, as well as all available information we get from our friends. The object friends is nothing else but an array that contains all id's of our friends. For each id there is another array, named data, that contains all (available) information about this specific friend.

```
friend_list = [friend['id'] for friend in friends['data']]
```

With this simple for-loop, one can now get a list of all friends. Since we're not primarly interesseted in our friends, but the connection in between them, we have to access at least the mutualfriends.

```
mutualfriends = graph.get_connections(tfriend, "mutualfriends")
```

[1]http://www.techopedia.com/definition/28984/facebook-graph-api
[2]https://github.com/pythonforfacebook/facebook-sdk
[3]https://developers.facebook.com/docs/facebook-login/access-tokens/

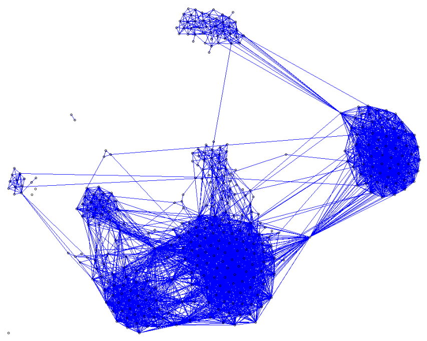


Figure 1: sdlhfa

The object mutualfriends provides us a list with all mutualfriends of "me" and friend "tfriend" (which is nothing else but one of the id's we have in the saved in "friend_list"

Repeat this proceedure for all friends, and we'll get exatly what we're looking for - a graph of all connection between our friends.

Get activity How did we define the parameter activity? We simply determine for each friend how many posts he makes on average per day. This gives us an indicator, how active a certain person is on facebook.

Again, this is done fairly quickly (using graph api).

```
statuses = graph.get_connections(tfriend , "statuses", fields="updated_time",
    limit="100")
```

This objecct provides us, all information we need (a list of all posts incl. a time stamp when it's been posted) to calculate the above mentioned quotient.

Get coordinates To visualise the network in MatLab, the software "Gephi" was used to determine the x- and y- coordinates of each individual. Therefore, the file "gephi.csv", an edge list of the network, was imported. The nodes were then arranged in communities using the layout method "ForceAtlas2". Finally, normalised node coordinates where exported to "coordinates.gdf". After deleting all headers and all edge data from this file, the coordinates could be easily imported to MatLab. The Network is visualized in Figure 12.

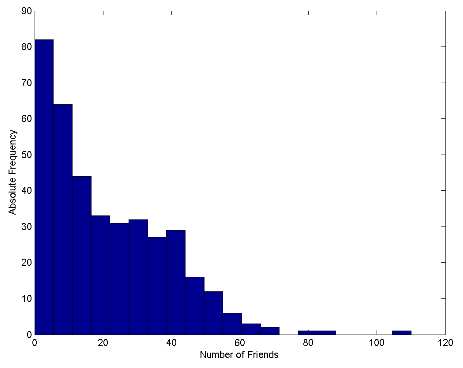
Anonymization The anonymization is acutally only a pseudo-anonymization. First all id's get arranged in order of size. Secondly, we create a list from 1 to n (number of friends). We then define a bijection between the two sets, such that the i-th element of the first set, maps to the i-th element of the second set.

Eventhough, this is not really anonymizing our data. It is more then sufficient for our purpose.

6 Simulation Results and Discussion

6.1 Characteristics of the network

The network we used consists of 384 nodes. The distribution of the number of friends and the local clustering coefficients are shown in Figure 2.



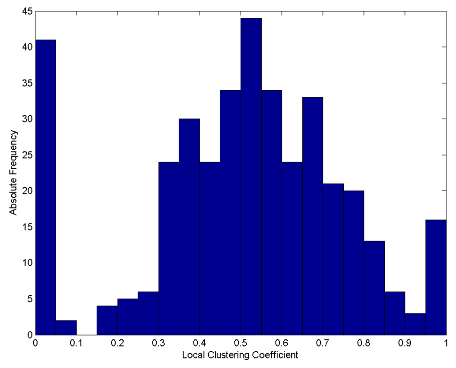


Figure 2: Distribution of the number of friends (left) and the local clustering coefficient (right).

be ba buzzemann

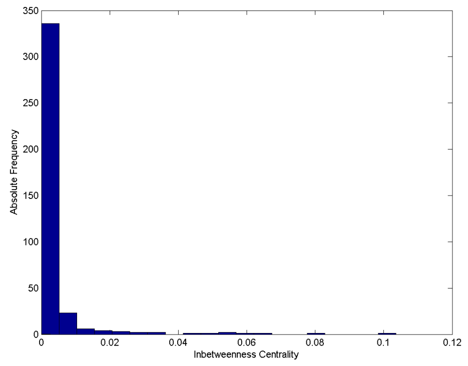
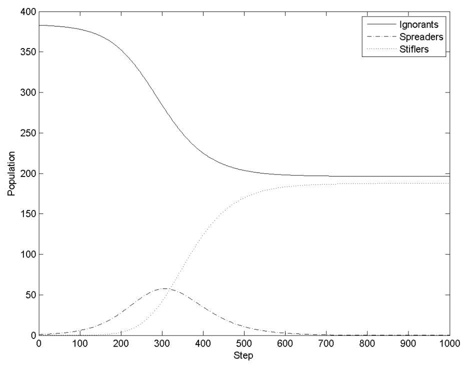


Figure 3: Distribution of betweenness centrality.

6.2 Differences in the time evolution

Most of the times, the form of the evolution graph of the system looks very similar in the case of the homogeneous and the inhomogeneous, agent-based model (Figure 4).



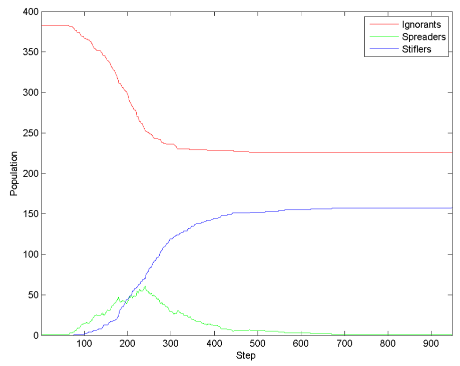


Figure 4: Left: Typical evolution of the system in the homogeneous SIR-model. Right: The Evolution in the agent-based model. In most cases, the evolution looks somewhat like this.

But there were also interesting cases where two local maxima occurred, as in Figure 5. This can never be the case in a homogeneous model. This requires several neighbourhoods in the network (and some luck, since the simulation uses random numbers). This might be a rather obvious insight, but it proofs that a homogeneous model might fail totally. If you would base any decision on a homogeneous model and you recognize that the number of spreaders decreases, you would expect that the spreading comes to an end. But it might be that it hardly started if you have inhomogeneities.

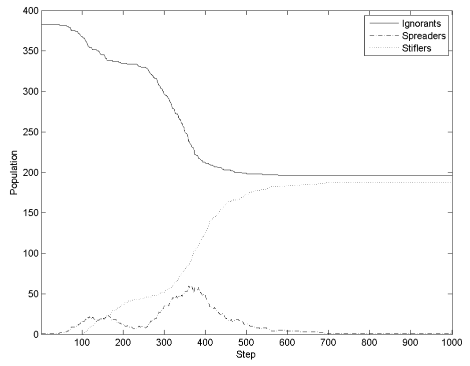


Figure 5: two local maxima, not continuous

6.3 Influence of p$_{forget}$

As shown if figures 7 and 6 the number of ignorants at the end of the simulation increases for higher p$_{forget}$ (α) if p$_{inform}$ (λ) is kept constant.

SIR: max of spreaders and the number of ignorants at the end are lower for lower pfoerget (lamda always the same).

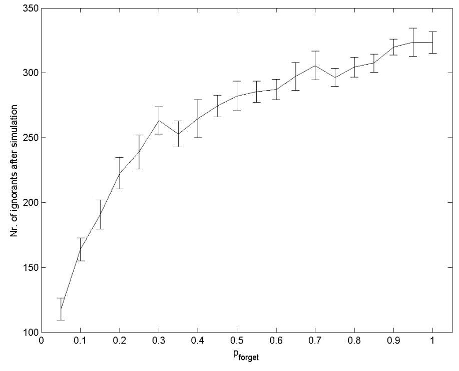


Figure 6: sdlhfa

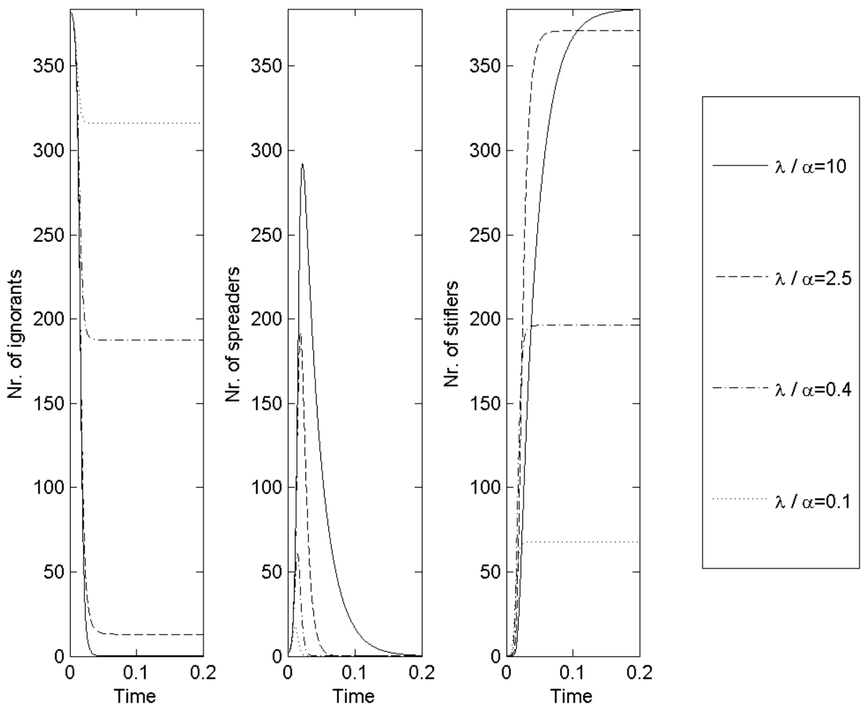


Figure 7: sdlhfa

6.4 Influentials

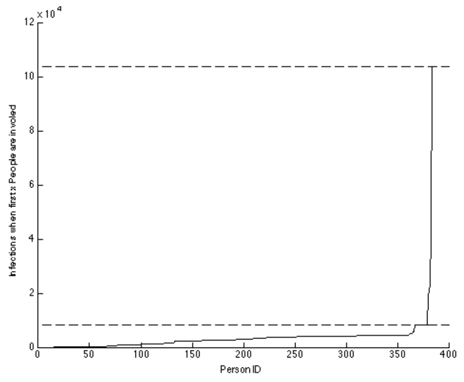
6.4.1 Existance and importance of influentials

To analyise the existence of influentials, we first need to define its meaning. We will use de definiton given by Watts [2], which simply says that a person's importance is determined by the number of its cumulative infections (called casced).

Figure 8 is a simple histogramm, that shows that the vast majority of people has only a small cumulative infection value. Whereas we see in Figure **??** (which again plots the cumultative infection, sorted from small to large), that there are some (eventough only a few) with a very large cumultative infection value.

We can even go one step further and try to estimate the importance of those influentials on all infections that occur in total. In order to do so, we sort the people after their importance (defined as above). We then calculate the amout of infections conditioned that only the m least important person have been involved in the spreading process (1 $\leq$ m $\leq$ 384).
For instance, $m = 384$ implies all infections that occured, when only the last (therefore most important) person has been excluded.

Given these results, we get that 94% of all infections, happen in the runs where the 1%-quantile of the most important people (in our case 4) have contributed (ie. were not excluced). This again indicates their importance.



According to Goldenberg [**?**], the importance of influential is often over-estimated. Our simulation strongly contradicts this statement.

Reasons might be, that our networt is a) of relative small size and b) due to its construction we might overwighted some individuals (in particular, those with high number of mutualfriends)

6.4.2 Determine influentials

Obviously the next question that comes up, is the following. If we have influetials, how can we determine these in our network. Do they have properties that differs them from the other individuals?

To find these mentioned, we have two possible aproaches:

1) Since our network is indeed a real world example of a Facebook Graph (the one of Patrick's Facebook Account). We can actually identify our, say top two, influetials. Then using this knowledge, we tried to narrow down the properties that might be of interest. It acutally turned out, that the two above mentioned induviduals indeed share very interessting propertiy. Namely, they are linkers of two or more cluster of our graph.

2) Taking the results of 1) into consideration, it lead us to a more graph theroretical analysis of our data. Since we were interessted connectivity properties, we calculated the *betweenness coefficent* for every node in our graph. It turns out that our top two influenctials indeed have the highest betweenness values.

Figure 10 shows, that high cumulative infection value implies high betweenness. But as we can also see in figure 3, it doesn't really imply the other direction.

7 Summary and Outlook

blub

8 Appendix

References

[1] A. Barrat, M. Barthlemy, A. Vespignani. Dynamical Processes on Complex Networks. *Cambridge University Press* Chapter 10, pp. 216-241

[2] D.J.Watts, P.S.Dodds. Influentials, Networks, and Public Opinion Formation. *Journal of Consumer Research* Vol 34, No. 4 (December 2007), pp. 441-458

9 Additional figures

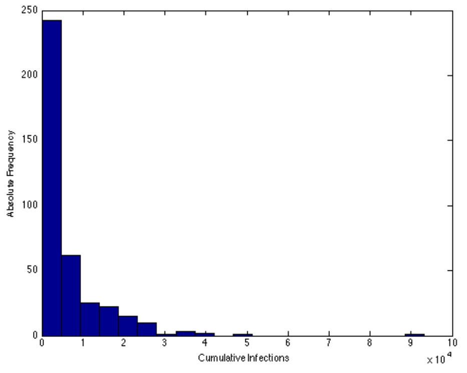


Figure 8: sdlhfa

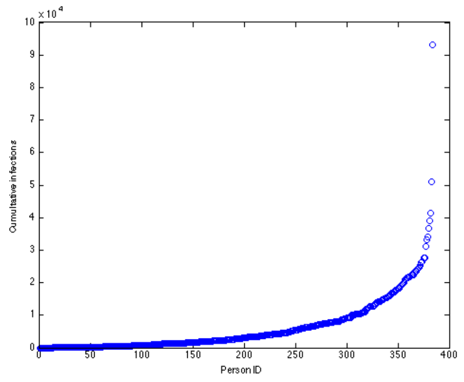


Figure 9: sdlhfa

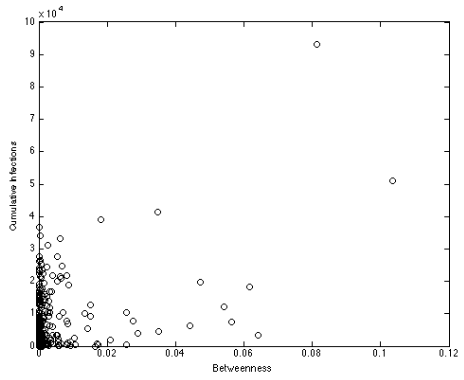


Figure 10: sdlhfa

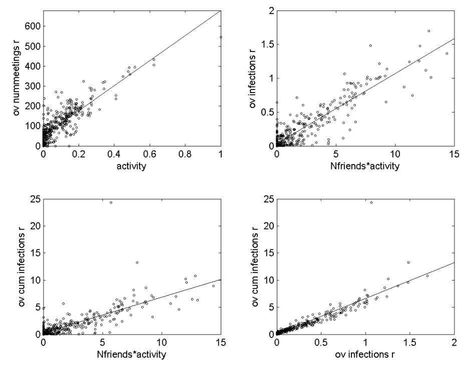


Figure 11: sdlhfa

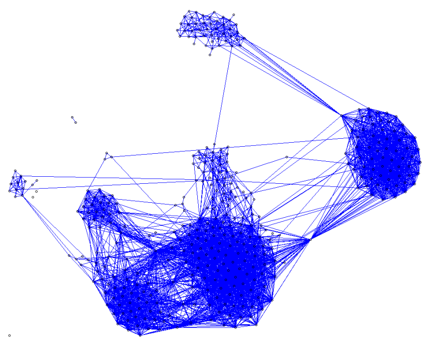


Figure 12: sdlhfa