# Network Dynamics and Learning: Homework 3

Lorenzo Melchionna
*Politecnico di Torino*
Torino, Italia
s304651@studenti.polito.it

***Abstract*—In this report has been explained how I solved the Homework 3 tasks and the results I obtained. In this homework I have exchanged ideas and compared the results with my colleague Jasmine Guglielmi.**

## I. Exercise

The first exercise consists in simulating the 2009 H1N1-virus (known as swine-flu) pandemic in Sweden.
This virus infected about 1.5 million people and in order to reduce its impact the Swedish government carried on a vaccination program which involved more than 60% of the population.

The main goal of this task is to infer the characteristics and disease-dynamics parameters of the pandemic through its simulation.

The simulations have been carried on following the upcoming 4 steps.

### A. Preliminary Step

The first step is a preliminary one, and it is made up by 2 parts.

*1) Epidemic on a known graph:* In the first part an epidemic on a symmetric k-regular undirected graph has been simulated using a discrete-time simplified version of the SIR epidemic as propagation model.

The graph is made up by $n = 500$ nodes where every node is directly connected with $k = 4$ closest nodes.
Given that we are using a SIR-like model, each node $i$ can have 3 different states $X_i \in \{S, I, R\}$, respectively Susceptible, Infected and Recovered.

At the initial state, there are 10 randomly choosen nodes in state $I$ and all the others are in state $S$.
The state transactions are managed using 2 different parameters:

- $\beta \in [0, 1]$ which is the probabilty that the infection is spread from a node in state $I$ and a node in state $S$ in 1 time step. If a given node has $m$ infected neighbors, its probability to be infected is $1 - (1 - \beta)^m$. In this case $\beta = 0.7$.
- $\rho \in [0, 1]$ which is the probability that a node in state $I$ switches to state $R$ during one time step. In this case $\rho = 0.3$.

In this case the time step is a week and simulations are performed on 15 weeks.
In order to have less aleatory results, the experiment is repeated $N = 100$ times, and results are shown in Fig. 1 and Fig. 2.

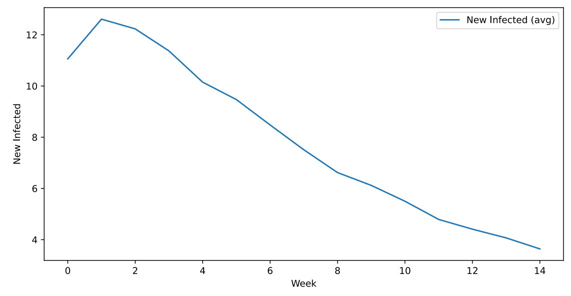

Fig. 1. New infected on the known graph

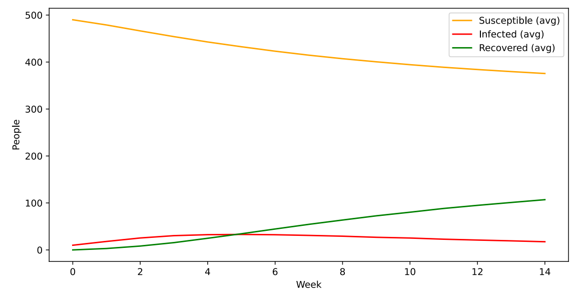

Fig. 2. Pandemic behaviour on the known graph

*2) Generate a Random Graph:* In the second part a random graph has been generated basing on the Preferential Attachment model, in order to have an average degree close to a given $k$.
This is done starting from a complete graph with $k + 1$ nodes and at each step a new node is added with $c = k/2$ direct connection, and the nodes to connect to the new one are sampled according to the degree distribution of the graph

before applying the current step.
The generation ends when a specified number of nodes is reached.

## B. Simulate a pandemic without vaccination

The goal of this step is to simulate a pandemic without vaccination.
To do that, a random graph with 500 nodes and average degree 6 has been generated as explained in the previous step.
The propagation has been simulated in the same way as in the previous step:

- discrete-time SIR propagation model
- $\beta = 0.7,\ \rho = 0.3$
- 1 week as unit of time, simulation performed for 15 weeks
- 10 initial infected randomly sampled

The simulation has been repeated $N = 100$ times, and results are showed in Fig. 3 and Fig. 4.

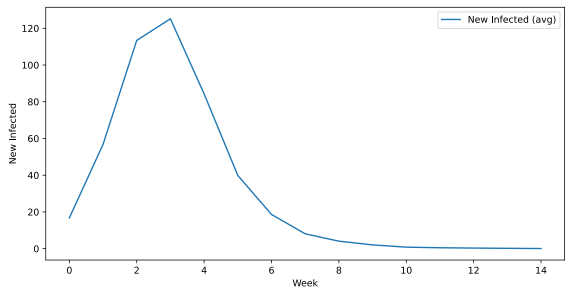

Fig. 3. New infected simulating pandemic without vaccination

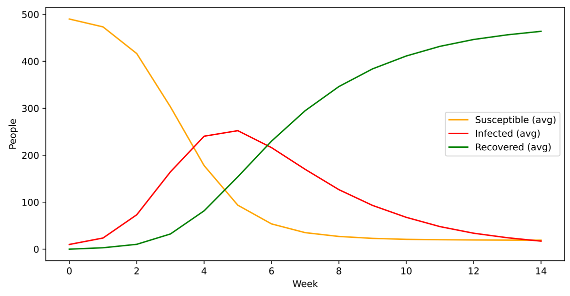

Fig. 4. Pandemic behaviour without vaccination

Comparing the results with the ones obtained performing the simulation on the known graph, it can be observed that there are much more infected, and this is given by the fact that the average degree in this case is higher so the graph is more connected.

## C. Simulate a pandemic with vaccination

The goal of this step is to simulate a pandemic without vaccination.
To do that, it has been basically done the same thing as the previous point, but in addition taking into account the vaccination and its impact on the evolution of the pandemic, assuming that a vaccinated person cannot be infected anymore and that the vaccine has immediately effect.
The random graph and the simulation parameters are the same as in the previous step, and it's given that the total fraction of the population that has received the vaccination by week is according to:

$$Vacc(t) = [0, 5, 15, 25, 35, 45, 55, 60, 60, 60, 60, 60, 60, 60, 60]$$

so it can be deduced that the fraction of the population that received vaccination during each week is according to:

$$Vacc\_weekly(t) = [0, 5, 10, 10, 10, 10, 10, 5, 0, 0, 0, 0, 0, 0, 0]$$

The simulation has been repeated $N = 100$ times, and results are showed in Fig. 5 and Fig. 6.

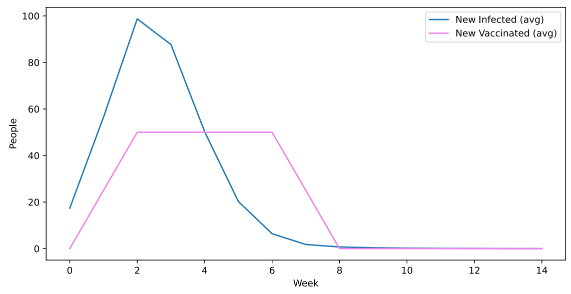

Fig. 5. New infected and new vaccinated simulating pandemic with vaccination

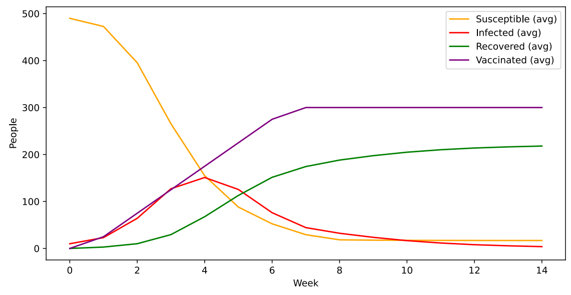

Fig. 6. Pandemic behaviour with vaccination

With respect to the previous simulation, it can be observed that trend of infected has been influenced by the vaccination, and being a correlated with the number of infected, also the trend of the recovered is influenced.
In conclusion, the vaccination has an impact also on the susceptible curve because we assume that when people are vaccinated they can not be infected anymore.

## *D. The H1N1 pandemic in Sweden 2009*

In this final step the previous methods are combined in order to simulate a real-world case of pandemic.
For execution time motivations, a sample of the the population $n = 934$ has been used, and for this sample we have the number of newly infected per week according to:

$I_0(t) = [1, 1, 3, 5, 9, 17, 32, 32, 17, 5, 2, 1, 0, 0, 0, 0]$

The goal of this step is to do a gradient-based search over the parameters $k$, $\beta$ and $\rho$ in order to find the combination that best approximate the real pandemic.
This had been done setting up a range for each parameter and performing an analysis on different combination that can be obtained selecting the one with the lowest Root-Mean-Square-Error, defined as:

$$RMSE = \sqrt{\frac{1}{weeks} \sum_{t=1}^{weeks} (I(t) - I_0(t))^2} \tag{1}$$

where $I(t)$ is the average number of weekly infected obtained by the simulation, and $I_0(t)$ is the true number of weekly infected.
The research starts, as suggested, performing a grid search over:

- $[k - \Delta\_k, k, k + \Delta\_k]$
- $[\beta - \Delta\_\beta, \beta, \beta + \Delta\_\beta]$
- $[\rho - \Delta\_\rho, \rho, \rho + \Delta\_\rho]$

several times, updating at each iteration the value of $k_0$, $\beta_0$ and $\rho_0$ to the combination with the lowest $RMSE$. The algorithm stops when the best parameters obtained are already equals to $k_0$, $\beta_0$ and $\rho_0$.
The best selected parameters are: $k = 8$, $\beta = 0.2$ and $\rho = 0.5$ with which a $RMSE = 3.496$ is obtained. Results are showed in Fig. 7 and Fig. 8.

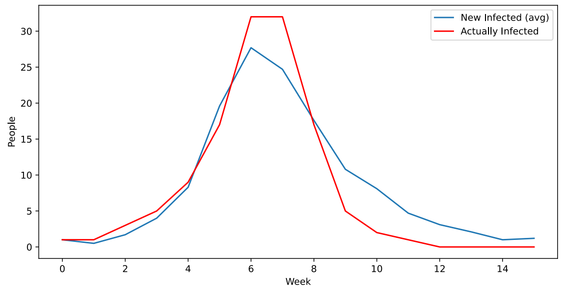

Fig. 7. Comparison between simulated infected and real ones in H1N1 pandemic

## *E. Challenge*

Two modifications were also tried out with the aim to perform a better simulation.

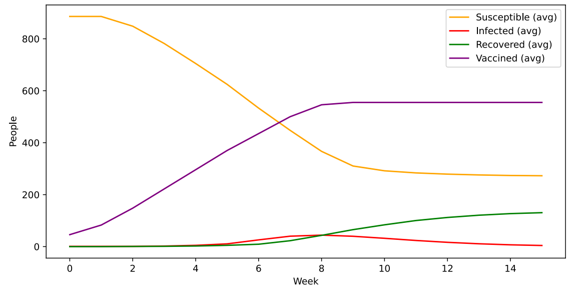

Fig. 8. H1N1 pandemic behaviour

*1) Small World:* The Small World random graph has been implemented to model the Swedish population, because as in many real-world network, is reasonable to assume that it has small diameter and high clustering.

To generate this random graph, a predefined $networkx$ function has been exploited.

The best selected parameters are: $k = 12$, $\beta = 0.5$ and $\rho = 0.4$ with which a $RMSE = 3.261$ is obtained.
Results are showed in Fig. 9 and Fig. 10.

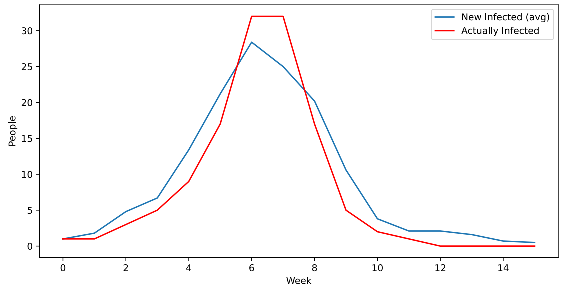

Fig. 9. Comparison between simulated infected and real ones in H1N1 pandemic

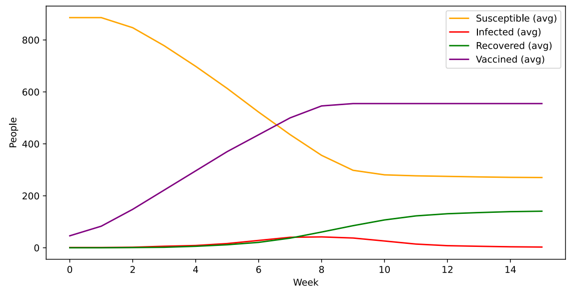

Fig. 10. H1N1 pandemic behaviour

This approach made some improvement to the initial algorithm, and it could happen because the approximation of the real connection between people in Sweden is modeled better by small world model.

*2) Bayesian Optimization:* It has been tried Bayesian optimization in combination with small world model to find the best parameter configuration for H1N1 pandemic, which is a method for global optimization of black-box functions.
It is particularly useful for problems with expensive function evaluations. The method is based on the Bayesian statistical framework, in which a probability distribution is used to model the function being optimized. The optimization procedure iteratively updates this distribution, based on the function evaluations, to converge to the global optimum.

The Bayesian optimization has been applied to minimize the RMSE over the simulations.
The parameter $k$ is considered as continuous in the optimizer, but it is discretized to perform the simulation.
The best selected parameters are: $k = 8$, $\beta = 1$ and $\rho = 0.15$ with which a $RMSE = 2.335$ is obtained.
Results are showed in Fig. 11 and Fig. 12.

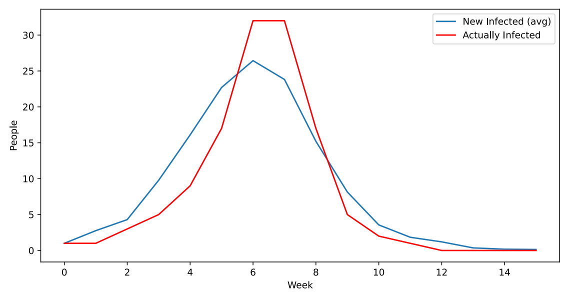

Fig. 11. Comparison between simulated infected and real ones in H1N1 pandemic

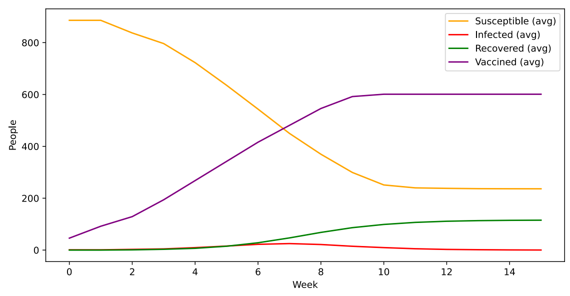

Fig. 12. H1N1 pandemic behaviour

This approach make important improvement with respect to the initial approach.
However there are some problems:

- The implementation used is implemented to deal with continuous parameters, and $k$ is a discrete one. To manage this problem in the simulation $k$ has been rounded, but this may not be a correct solution.
- The selected range of $k$ is too small and could not contain the optimal value
- $\beta = 1$ is an unexpected value

## II. Exercise

The second exercise consists in deploying graph coloring algorithms in order to minimize a given potential function. The exercise is divided in 2 parts.

### A. Simple Line Graph

In this part it has been studied a line graph with 10 nodes represented in Fig. 13.

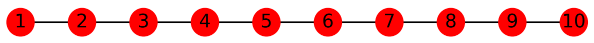

Fig. 13. Line graph at initialization

Each node $i$ in each time-step $t$ has a state $X_i(t)$, which can assume 2 different values: $C = \{red, green\}$. At the beginning, all nodes are initialized to $red$.
At each time-step $t$ a random node is sampled and its state is updated according to the following probability distribution:

$$\mathbb{P}(X_i(t+1) = a | X(t), I(t) = i) = \frac{e^{-\eta(t) \sum_j W_{ij} c(s, X_j(t))}}{\sum_{s \in C} e^{-\eta(t) \sum_j W_{ij} c(s, X_j(t))}} \quad (2)$$

where $W$ is the adjacency matrix, $c$ is the cost function, $\eta$ is the inverse of the noise and they are defined as:

$$c(s, X_j(t)) = \begin{cases} 1 \text{ if } X_j(t) = s \\ 0 \text{ otherwise} \end{cases} \qquad \eta(t) = \frac{t}{100}$$

To quantify how the algorithm is close to the solution, the potential function:

$$U(t) = \frac{1}{2} \sum_{i,j} W_{ij} c(X_i(t), X_j(t)) \quad (3)$$

The simulations is stopped when the potential function reaches 0, and it can be observed that it happens always, but in a different number of iteration depending on random factors.

The behaviour of the potential function $U(t)$ with respect to 10 different simulation is showed in Fig 14 and one of the graph coloring solutions is showed in Fig 15.

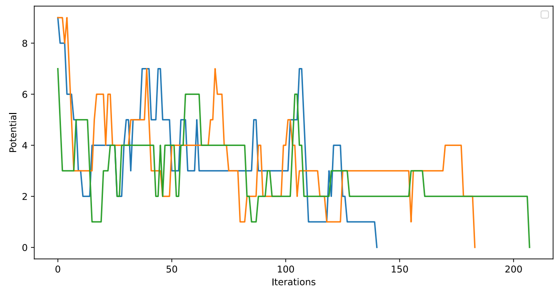

Fig. 14. Behaviour of the potential function in 3 different simulations

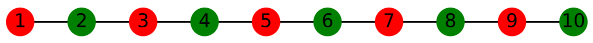

Fig. 15. Solution for the coloring problem

### B. Wifi-channels to routers

In this part the previous developed algorithm ha been generalize to a much complex real-world case: the assignment of wifi-channels to routers.
The graph representing the router network is showed in Fig. 16.
There are 100 nodes which can assume 8 different states: $C = \{red, green, blue, yellow, magenta, cyan, white, black\}$.
At the beginning nodes are initialized randomly.
The probability distribution used to update colors and the noise are the same as previous.
An important different is about the cost function, that in this case is defined as:

$$c(s, X_j(t)) = \begin{cases} 2 \text{ if } X_j(t) = s \\ 1 \text{ if } |X_j(t) - s| = 1 \\ 0 \text{ otherwise} \end{cases}$$

In this case the potential function never reaches 0, because

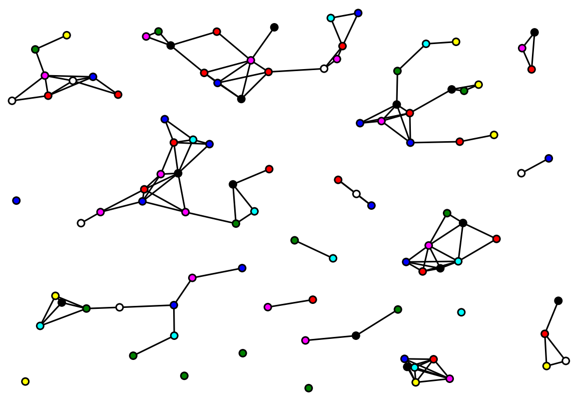

Fig. 16. Router network at initialization

there are not enough different color to totally avoid conflicts. Given this fact the simulation is stopped when the potential function reaches 5, over 10 simulations it happens in the previous case in different number of iteration depending on random factors. The behaviour of the potential function $U(t)$ with respect to 10 different simulation is showed in Fig 17 and one of the graph coloring solutions is showed in Fig 18.

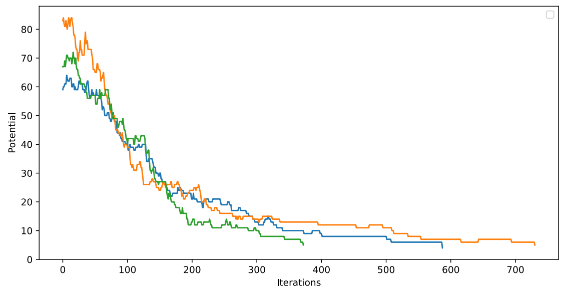

Fig. 17. Behaviour of the potential function in 3 different simulations

### C. Optional

In this point different values of $\eta$ are used in order to observe how the behaviour of the potential function is influenced.
To better compare the situations, the initial condition has been set differently, initializing each node to state *red*.

Choosing constant values, for example $10^{-5}$, 1 and $10^{2}$, results are showed in Fig. 19.
It can be observed that with a very low $\eta$ (high noise) the stop condition is never reached and the potential function remains very high, increasing $\eta$ the potential has a good behaviour in the initial iterations, but then is not able to go down a certain threshold, and finally taking an high value of

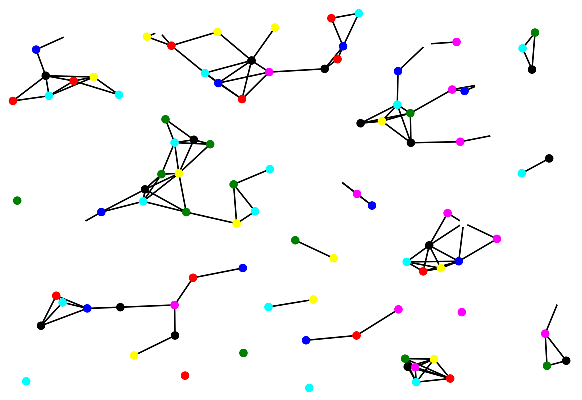

Fig. 18. Solution (not optimal) to the coloring problem

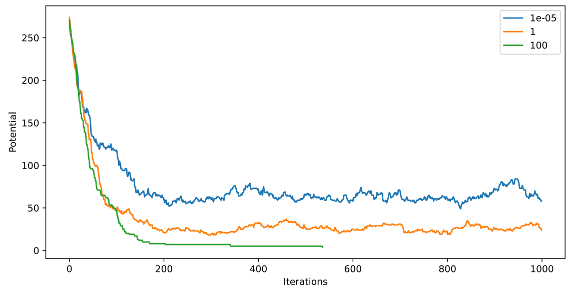

Fig. 19. Behaviour of the potential function with 3 different fixed values of $\eta$

$\eta$ (low noise) the stop condition is reached very quickly.

Choosing other increasing functions, for example:

$\eta(t) = \frac{\log(t)}{100} \qquad \eta(t) = \frac{t^4}{100}$

results are showed in Fig. 20.

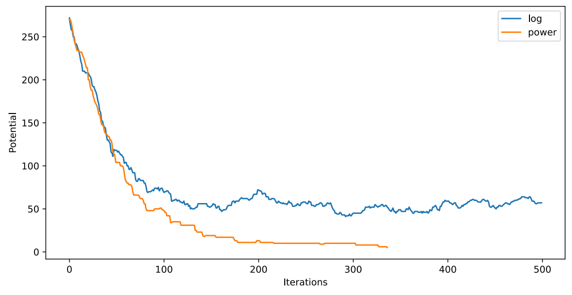

Fig. 20. Behaviour of the potential function with 2 different functions $\eta(t)$

It can be observed that the behaviour of the potential is similar in the first iterations (where the values of the 2 function are close), but it becomes much different later: using a logarithmic function, the noise decreases very slowly, so no stop condition is reached performing 500 iteration, using a power function instead, the noise decreases much faster and the stop condition is reached after few hundred iterations.