Networks Dynamics and Learning: Homework 2

Valerio Zingarelli
Politecnico di Torino
Student id: s281586
s281586@studenti.polito.it

***Abstract*—In this report I will explain and explore the methods I adopted to solve proposed exercises and the results I got. The essay will be divided into sections, each one corresponding to one of the three assigned problems. During the resolution of the exercises, Vito Palmisano and I have compared our results and provided advice to each other, especially in second and third exercises.**

I. Problem 1

The first task of the assigned homework is divided into 2 different parts. More specifically, given the graph in Fig.1, and the matrix Λ, several tasks required to be solved.

$$\Lambda = \begin{bmatrix} 0 & 2/5 & 1/5 & 0 & 0 \\ 0 & 0 & 3/4 & 1/4 & 0 \\ 1/2 & 0 & 0 & 1/2 & 0 \\ 0 & 0 & 1/3 & 0 & 2/3 \\ 0 & 1/3 & 0 & 1/3 & 0 \end{bmatrix} \tag{1}$$

Problem 1, part 1

The first part required to consider Λ as Transition Rate Matrix of a continuous Markov Chain represented by the graph in Fig. 1. In the following sections the solutions and the approaches used to solve this problem will be shown.

A. Problem. 1a

In the first point of exercise 1, we were required to perform some simulations to find what is the average time it takes a particle that starts in node *"a"* to leave the node and then return to it. In other words, we were required to find the return time of the node *"a"*. Let's have a look at the proposed solution.

First of all, vector ω has been computed:

$$\omega = \Lambda 1 \tag{2}$$

$$\omega* = max(\omega) \tag{3}$$

After that, matrix P has been constructed:

$$D = diag(\omega) \tag{4}$$

$$P = D^{-1}\Lambda \tag{5}$$

Concerning time simulation, a rate-r Poisson process has been used. More specifically, to simulate the time between two next ticks of the Poisson-clock, the following has been used:

$$\tau_{next} = -\frac{log(u)}{r} \tag{6}$$

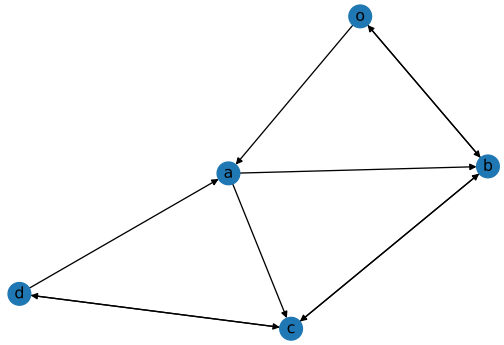

Fig. 1. Graph of exercise 1

where u is a uniformly distributed random variable $u \sim U(0,1)$. In order to determine where the particle should go, the cumulative sum of the rows of matrix P has been computed and compared to the result a uniformly distributed random variable.

100 simulations have been performed and, in the end, the average return time is:

$$T_a^+ \sim 6.77 \tag{7}$$

B. Problem 1b

In the second point of the first task, we were asked to compute, theoretically, the return time of node "a" and to compare it with the result of the simulations. According to the theory, the expected return time of a node i can be computed as:

$$\mathbb{E}_i[T_i^+] = \frac{1}{\bar{\pi}_i \omega_i} \tag{8}$$

So, now we need to estimate and , to do that, we compute the matrix Q defined as:

$$Q_{ij} = \frac{\Lambda_{ij}}{\omega*}, \;\; Q_{ii} = 1 - \sum_{i \neq j} Q_{ij} \tag{9}$$

Suddenly, using eigenvalues and eigen-vectors of Q, $\bar{\pi}$ is computed. Obtained value of $\mathbb{E}_i[T_i^+]$ is:

$$\mathbb{E}_a[T_a^+] = 6.75 \tag{10}$$

which is very very close with the one obtained with simulations. Of course, if the number of simulations is increased, the simulations' value will converge to the theoretical one.

C. Problem 1c

Similarly to what was required before, in this point we are asked to find the average time it takes to move a particle from node "o" to node "d". The reasoning was exactly the same of point $1.a$ but,this time the simulation is stopped when the particles reaches node "d". Once again, several simulations have been performed and the obtained result is:

$$\mathbb{E}_o[T_d] \sim 8.70 \tag{11}$$

D. Problem 1d

Same as before, in this point we were required to compute the hitting-time determined during the previous point in a theoretical way. According to the theory:

$$\mathbb{E}_i[T_S] = \frac{1}{\omega_i} + \sum_j P_{ij}\mathbb{E}_j[T_S] \tag{12}$$

But there is another way of computing this quantity. The expected hitting times $\hat{x} = (\mathbb{E}_i[T_S])_{i\in R}$ for a set S and for all the nodes $i \in R = \mathbb{V}/S$ can be computed solving:

$$\hat{x} = \Omega + \hat{P}\hat{x} \tag{13}$$

where $\Omega_i = \frac{1}{\omega_i}$ and $\hat{P}$ is obtained by removing rows and columns corresponding to the nodes in S from P. More specifically, $\hat{x}$ can be expressed as:

$$\hat{x} = (I - \hat{P})^{-1}\Omega \tag{14}$$

It's important to specify that $(I-\hat{P})$ is invertible because $V\backslash S$ has a link pointing to S. In fact, in this case, there is the link (c ; d) pointing to $S = \{d\}$.

The obtained value, according to this calculations is $\mathbb{E}_o[T_d] = 8.78$, which is quite similar to the one obtained by simulations. Also in this case, increasing the number of simulations will lead to the convergence of the value obtained by them and the theoretical one.

Problem 1, part 2

In the second part of the first problem, we were required to perform some analysis on consensus dynamics, using as a reference the graph in Fig.1. Of course, since we are not talking about Markov Chains anymore, matrix Λ will be considered as the weight- matrix of the graph $G = (V, E, \Lambda)$.

E. Problem 1.e

The first task of the second part of Problem 1 required to simulate French-DeGroot dynamics on the graph in Fig. 1, according to Λ and starting from an arbitrary initial condition $x(0)$.

In Fig.2 an example of some evolution of consensus is provided. More in details, five different initial configurations have been evolved: in the i-th initial configuration, a 1 is put in position i of $x(0)$. In all the cases, consensus is reached. This is because G is *strongly connected* and *aperiodic* and, as a consequence, consensus is always reached. In addition to

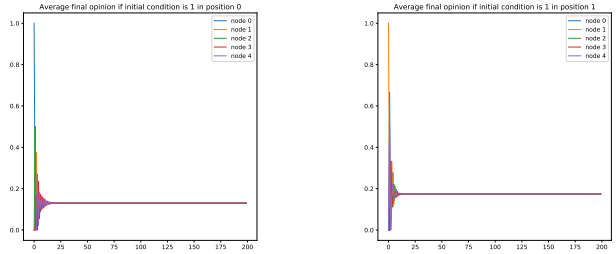

Fig. 2. An example of the dynamics for 1.e. Only two initial configurations have been inserted in the essay for space-reasons.

that, an estimation of π has been computed. In fact, we know that:

$$\alpha = \sum_k \pi_k x(0)_k \tag{15}$$

where α is the consensus value. Since a 1 has been put in position i of x(0) during the i-th simulation, the final value of consensus will be equal to the i-th element of π. Simulations confirmed that.

F. Problem 1.f

In this point, the initial state of the system is not fixed anymore. In fact, each element of the initial state vector is now a random variable with variance σ^2. I choose to adopt a uniformly distributed random variable xi such that:

$$x(0)_i = \xi_i,\ \xi_i \sim U(0,1) \tag{16}$$

Specifically, $Var(\xi_i) = \frac{1}{12}$. The request of the exercise wanted us to compute the variance of the final value of consensus α starting from the variance of the random variable adopted as a starting condition. Once again, the variance has been computed using simulations and using theory. Concerning the simulations part, several experiments have been performed and the variance of the final consensus α has been computed using variance definition:

$$Var(\alpha) = \mathbb{E}[(\alpha - \mathbb{E}[\alpha])^2] \tag{17}$$

Regarding theoretical computation of $Var(\alpha)$, we have to keep in mind a simple rule:

$$Var(aX + bY) = a^2Var(X) + b^2Var(y) + 2Cov(X,Y) \tag{18}$$

Recalling (15) and the fact that all the initial random variables are i.i.d. we can state:

$$Var(\alpha) = Var(\sum_k \pi_k x(0)_k) = \sum_k \frac{\pi_k^2}{12} \tag{19}$$

Regarding numerical values, obtained results are: $Var(\alpha)_{th} = 0.0178$ and $Var(\alpha)_{sim} \sim 0.018$. As it's possibile to notice, values are really close.

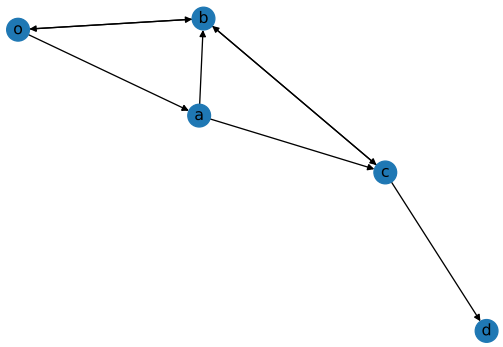

Fig. 3. The graph obtained for point 1.g

G. Problem 1.g

In this exercise, the task was to remove edges (d;a) and (d;c) and then to evolve the system and describe the asymptotic behaviour. After simulations, it's possible to notice that the system converges to "d"'s initial opinion(i.e. "d" is a sink and its opinion is not influenced by anyone). In addition to that, it's interesting to notice that, computing pi, we get:

$$\pi = [0, 0, 0, 0, 1] \tag{20}$$

Suddenly, the variance of the consensus has been computed using the same reasonings and methods of point 1.f and obtained results are $Var(\alpha)_{th} = 0.083$ and $Var(\alpha)_{sim} \sim 0.084$. Applying (19), we get:

$$Var(\alpha) = \sum_k \frac{\pi_k}{12} = \frac{1}{12} \tag{21}$$

Inf Fig.4 an example of a system evolution is provided. Notice that the first node to reach consensus is node "c" which is the closest to the sink.

H. Point 1.h

Starting from $G = (V, E, \Lambda)$, in this point it was asked to eliminate links $(c; b)$ and $(d; a)$ and analyse the French-DeGroot dynamics of the new obtained graph that is provided in Fig.6. It's quite easy to notice that, in this situations, nodes "c" and "d" represent a trapping component and they will behave, as a whole, just like node "d" did in previous point. In fact, consensus is reached only if "c" and "d" have the same opinion, otherwise consensus is not reached. An explanatory trajectory is provided in Fig.6. Starting from an initial condition $x(0) = [1.0.0.1.1.]$, consensus has been reached since "c" and "d" have the same opinion. Notice that, also in this case, first node to reach consensus are "a" and "b", the closest to the trapping component, and the last is "o" which is the farthest.

II. Problem 2

In this exercise, we have to deal with the graph of Fig.1 and Λ is the transition matrix of a Continuous Markov Chain again. However, now we will simulate many particles moving around in the network in continuous time. Each of the particles in the network will move around just as the single particle moved around in Problem 1: the time it will stay in a node is exponentially distributed and the next move of a particle is computed basing on matrix P computed in (5). Two different perspectives will be analysed: *particle perspective* and *node perspective*.

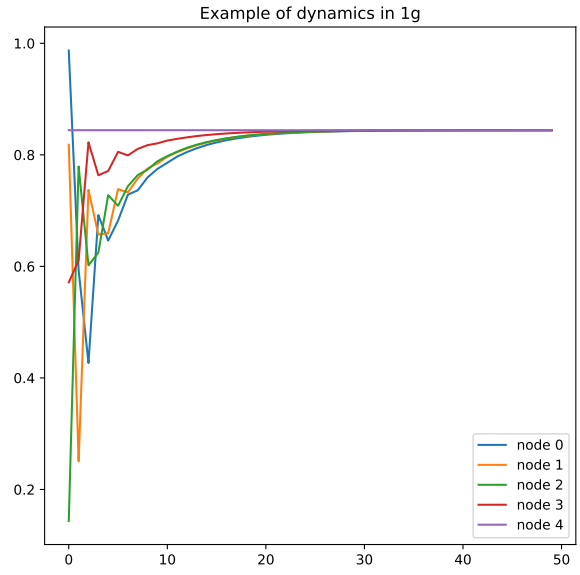

Fig. 4. An example of asymptotic behaviour in point 1.g. In this simulation "d"'s original value was 0.835. Even if 200 steps have been run, x-axis has been reduced to 50 for clarity reasions.

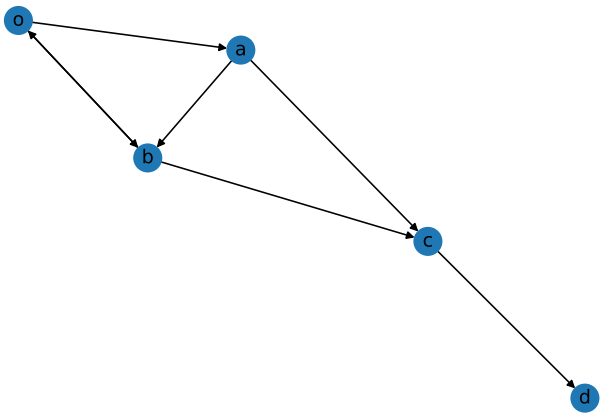

Fig. 5. Obtained graph after modifications required by Problem 1.h

A. Problem 2a, particle perspective

In this point a Poisson-clock is installed on each of the one hundred particles. Since the movement of a particle will not influence the behaviour of the others, we can consider this

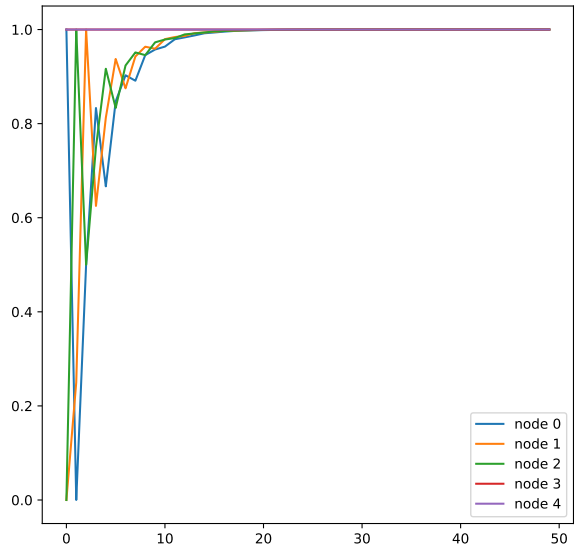

Fig. 6. Dynamics of the graph of point 1.h. In this case, initial array was: $x(0) = [1.0.0.1.1.]$. x-axis has been reduced to only 50 steps for clarity reasons

point as the same of exercise 1.a. In fact, if we simulate one particle at a time for 100 times or we simulate 100 particles simultaneously one time, we will get the same results. So, code for point 1.a has been runned $100 * 100 = 10000$ times and the results are the same of 1.a. As I specified before, this could easily explained if we think that all the particles are independent.

B. Problem 2b, node perspective

In this point, we were required to simulate the same system as previous point but, this time, from nodes' perspective. So, now, we don't care about particles' movements but only about the number of particles inside each node over time. In addition to that, the rate of each node Poisson-clock is proportional to number of particle for each node.

$$\tau_{nexti} = -\frac{log(u)}{\omega_i * n_{particlesi}} \tag{22}$$

Then, in the end, the node to change and update is the one with the minimum $\tau_n ext$. The system has been evolved for 60 time units as required; in Fig7 dynamics, in terms of number of particles, have been shown for node "o" and "a". I refer to the code for the number of particles of the other nodes over time. Anyway, the average particles distribution after several simulations is:

$$n_{particles} = [18.9, 15.88, 21.64, 21.56, 22.02] \tag{23}$$

Where position 0 corresponds to node "o", position 1 to node "a" and so on.

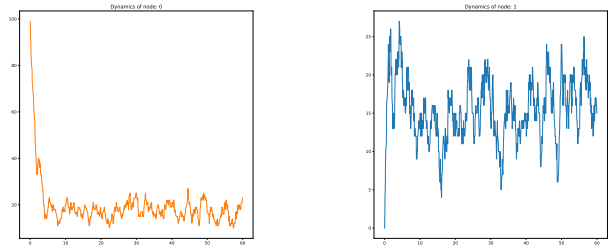

Fig. 7. Some example of the evolution of the system for point 2.b. On y-axis we have the number of particles for each node while, on the x-axis we have time.

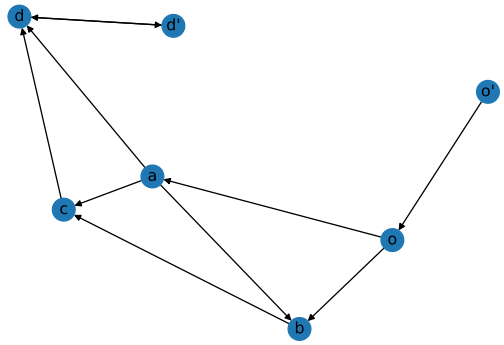

Fig. 8. Resulting graph for problem 3.

III. Problem 3

In this exercise, an open graph has been provided according to the following transition-rate matrix Λ_{open}.

$$\Lambda_{open} = \begin{bmatrix} 0 & 2/3 & 1/3 & 0 & 0 \\ 0 & 0 & 1/4 & 1/4 & 2/4 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{24}$$

To deal with such a graph proposed solutionsa are basically two:

- To solve the problem of the all-zeros row of Λ_{open}, a self loop on d has been added with a very very low corresponding value in Λ_{open}
- Two nodes, "o'" and "d'", have been added to the node in order to simulate the "entrance" and the "exit" of particles in the graph, respectively. Of course, during the execution of algorithm, number of particles in such nodes do not never modify. Resulting graph has been shown in Fig.8

A. Problem 3a

For this system, particles will enter the system at node "o" according to a Poisson process with rate = 1. Each node will then pass along a particle according to a given rate. Two kind of $rate$ will be used: for point a the rate will be proportional to the number of particles in the node, for point b, this rate

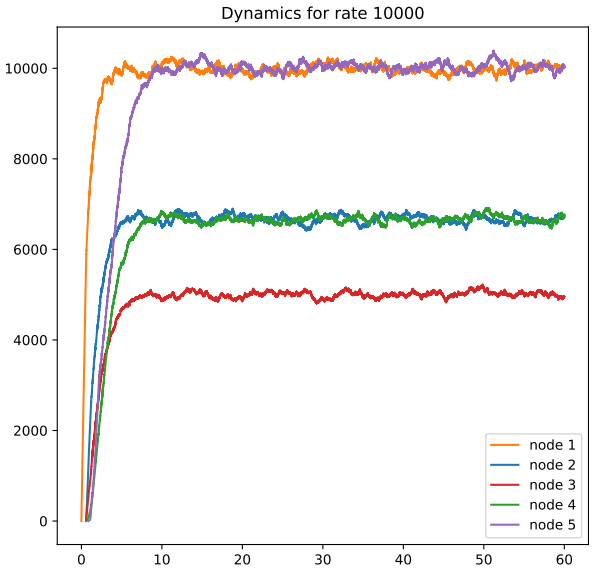

Fig. 9. An example of dynamics in problem 3.a. In this case rate has been se equal to $r = 10000$

will be fixed and equal to 1. tha requirement was to find, using simulations, the rate r that makes the system blow-up. The task has been solved using a code similar to the one of Problem 2 but, this time:

$$\tau_{next}[i] = -\frac{log(u)}{n_{particles}[i]} \tag{25}$$

Different rates r have been tested(from 1 to 1000000) and an example of dynamics with $r = 10000$ has been provided in fig.9. All the rates that have been tested didn't make the system to blow-up. That's because, in my opinion, the nodes adapt to the velocity of the source, since the time of updating $\tau_{next} \sim \frac{1}{n_{particles}}$. This fact makes them send particles faster and faster.

B. Problem 3b

In the last request of the second homework we were required to fulfill the same task of problem 3a but, in this case, the rate will not be dependent on the number of particles but it's fixed and, more specifically, it's equal to 1. In figures 10 and 11 a sample of the number of particles for each node has been shown. It's easy to notice that with rate $r = 1$ the system manages to handle the particles but, a rate equal to 2 is enough to make to system blow-up. That's because, in my opinion, the nodes are too slow. In fact, since their τ_{next} is fixed, they don't adapt anymore to the speed of the source.

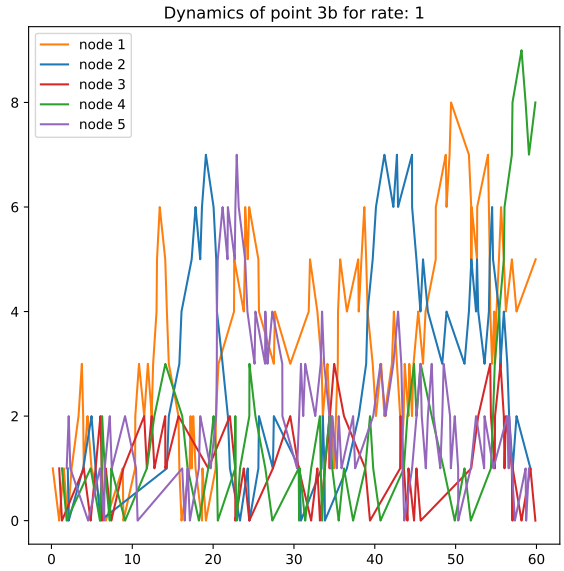

Fig. 10. The number of particles per node with r=1

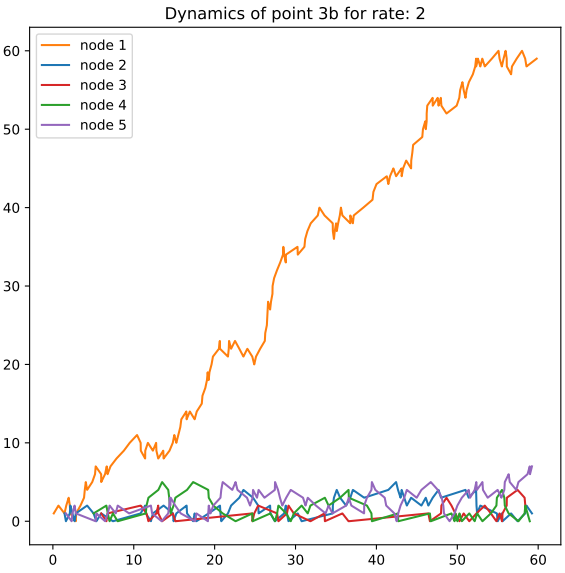

Fig. 11. The number of particles per node with r=2