Homework assignment 2

Jiahao Zhang

December 11, 2022

Abstract

This is a proposed solution for homework assignment 2 of the course Network Dynamics and Learning 2022/23. For the result, some numbers have been truncated for a better visualization quality

Exercise 1

The first part of this assignment consists in studying a single particle performing a continuous-time random walk in the network described by the graph in Fig. 1 and with the following transition rate matrix:

$$\mathbf{\Lambda} = \begin{array}{c} \\ o \\ a \\ b \\ c \\ d \end{array} \begin{array}{c} \begin{array}{ccccc} \bar{o} & \bar{a} & \bar{b} & \bar{c} & \bar{d} \end{array} \\ \begin{bmatrix} 0 & 2/5 & 1/5 & 0 & 0 \\ 0 & 0 & 3/4 & 1/4 & 0 \\ 1/2 & 0 & 0 & 1/2 & 0 \\ 0 & 0 & 1/3 & 0 & 2/3 \\ 0 & 1/3 & 0 & 1/3 & 0 \end{bmatrix} \end{array} \tag{1}$$

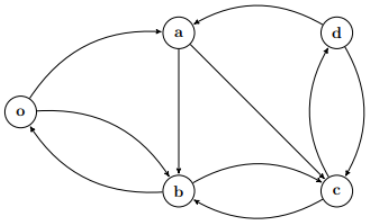

Figure 1: Closed network in which particles move according to the transition rate matrix (1)

Your task is to simulate the particle moving around in the network in continuous time according to the transition rate matrix (1)

Before presenting the solution, I would like firstly to introduce some theoretical concepts. The simulations done for the exercises can be considered as Continuous Time Markov Chain, where the time is not discrete but it flows in a continuum. The random process still describes the evolution of a state variable x inside a discrete state space $\mathcal{X}$ with a graph structure. We are given a graph $G = (\mathcal{X}, \Lambda)$ with nodes $\mathcal{X}$ and weight matrix Λ describing possible transitions between nodes/states.

Transitions now happen at random time instants that are decided by the tick of a so-called Poisson clock. A Poisson clock is characterized by the property that the time elapsed between any two of its consecutive ticks are an independent random variable with exponential distribution with a specified rate.

Remark: to simulate Continuous Time Markov chains the following fact will be useful. To simulate a Poisson clock with rate r, one must simulate the time between two consecutive ticks, which we denote by t_{next}. We can compute t_{next} as

$$t_{next} = -\frac{\ln(u)}{r}$$

where u is a random variable with uniform distribution, $u \in \mathcal{U}(0,1)$.

For simplicity, in this exercise, we will use a global clock where the time between two consecutive ticks is computed as :

$$t_{next} = -\frac{\ln(u)}{\omega^*}$$

where ω^* is the maximum value of the row sums of the matrix Λ. In particular

$$\omega^* = \max_i(\omega_i) = \max_i(\sum_j \Lambda_{ij})$$

(a) *What is, according to the simulations, the average time it takes a particle that starts in node a to leave the node and then return to it*

- Here we are simulating the so-called return time. Since we are simulating CTMC (Continuous Time Markov Chain) with a global clock, the simulation's steps can be defined as follows:

  1) you define a unique global Poisson clock with an appropriate rate as described in the intro of the exercise
  2) when you are at node i and the global clock ticks, either you jump to a neighbor j with probability

  $$Q_{ij} = \frac{\Lambda_{ij}}{\omega^*},\ i \neq j$$

  or you stay in the same node (no transition) with probability

  $$Q_{ii} = 1 - \sum_{i \neq j} Q_{ij}$$

  .

  In this approach, the continuous time is "discretized" using a global clock, while the matrix Q describes the jumps. For this reason, the matrix Q is called the jump chain of the CTMC.

  In order to simulate the average time a node stays in a specific node, we can consider the interval of time from which this particle enters the node and the time it leaves from there. The estimate of the invariant measure for that state then is equal to the total time spent on the node divided by the total time of the process.

  For this simulation, we performed 20000 iterations, where each iteration corresponds to a tick of the Poisson clock.

  The estimate of the invariant distribution is :

  $$\pi_{est} = \begin{bmatrix} 0.1834 & 0.1497 & 0.2200 & 0.2219 & 0.2251 \end{bmatrix}$$

  For the general return time, we could derive it simply from the formula (using the same notation as done in the lecture slides)

  $$E_i[T_i^+] = \frac{1}{\pi_i \omega_i} \tag{2}$$

  So, in our case, by substituting the values, we obtain the estimated value of the return time of node a

  $$E_a[T_a^+]_{estimate} = \frac{1}{\pi_{a\_est}\omega_a} = 6.6779$$

(b) *How does the result in a) compare to the theoretical return-time $E_a[T_a^+]$? (Include a description of how this is computed.)*

- The formula for computing the return time is already provided. In this case, we only need to compute the invariant distribution of the system given the matrix Λ (1). Given the normalized weighted matrix Q defined in a),we compute its eigenvalues and extract the eigenvector associated with the maximum eigenvalue (that in this case should correspond to 1 and unique because the matrix Q gives a graph that's strongly connected and aperiodic by construction). In the end, we just extract the real value of each element of the eigenvector and normalize it. The theoretical value of the invariant distribution obtained here is

  $$\pi = \begin{bmatrix} 0.1852 & 0.1481 & 0.2222 & 0.2222 & 0.2222 \end{bmatrix} \tag{3}$$

It is easy to observe that this vector is very similar to the estimated value obtained in a), so we would expect similar behavior in computing the exact return time $E_a[T_a^+]$. In fact:

$$E_a[T_a^+] = \frac{1}{\pi_a \omega_a} = 6.7500$$

This also proves that our simulation was good enough, obtaining a relative error of 1%

(c) *What is, according to the simulations, the average time it takes to move from node o to node d?*

- The request of this exercise corresponds to finding the hitting time $E_o[T_d]$ from node o to node d. To respond to this problem, we create a new setup in which we iterate over 20000 times the time spent by a particle to move from node o to node d, we save this time interval for each particle and at the end, we do the average. The simulated result obtained is

$$E_o[T_d]_{estimate} = 8.8124$$

(d) *How does the result in c) compare to the theoretical hitting-time $E_o[T_d]$? (Describe also how this is computed.)*

- Given the set S the set of globally reachable nodes in the graph, the theoretical hitting time from any node i to S can be computed using the recursive formula:

$$\begin{aligned} E_i[T_S] &= 0 && if i \in S \\ E_i[I_S] &= \frac{1}{\omega_i} + \sum_j Q_{ij} E_j[T_S] && if i \notin S \end{aligned}$$

which in the matrix format could be written as

$$\hat{x} = \frac{1}{\hat{\omega}} + \hat{Q}\hat{x}$$

where $\hat{Q}$ and $\hat{w}$ are obtained respectively from Q (the normalized weight matrix of the graph) and ω by removing the rows and/or columns corresponding to the nodes in the set S. moving $\hat{x}$ on the left-hand side, the hitting time can be expressed as

$$\hat{x} = (I - \hat{Q})^{-1} \frac{1}{\hat{\omega}}$$

By setting the set $S = \{d\}$, we obtain as result the following vector

$$E_i[T_d] = \begin{bmatrix} \mathbf{8.7857} & 7.1429 & 7.0714 & 3.3571 & 0 \end{bmatrix}$$

and the result we are looking for is the first element of the array (marked as bold). We notice that the obtained value from the theoretical computation is very similar to the simulated one, which indicates that the steps we did for the simulation are reasonable and acceptable.

(e) *Interpret the matrix Λ as the weight matrix of a graph $G = (V, E, \Lambda)$, and simulate the French-DeGroot dynamics on G with an arbitrary initial condition $x(0)$. Do the dynamics converge to a consensus state for every initial condition $x(0)$? Motivate your answer.*

- Before simulating the French-De Groot model, we first do some preliminary checks on the graph. In fact, if we can verify that our analyzed graph is strongly connected and aperiodic, by the French-De Groot theorem we can surely say that all opinions converge to a common value defined as **consensus**, and it is a convex combination of the original opinions weighted by the invariant distribution π (this last is different from the π obtained before because this one is derived from the dominant eigenvalues of $P = D^{-1}\Lambda$). To check the strongly connectedness and aperiodicity we can apply some library functions present on networkx, in particular, *networkx.is_strongly_connected* and *networkx.is_aperiodic*, and the result is that our graph 1 is both strongly connected and aperiodic. Since we satisfy the hypothesis of the theorem, we can state that in this exercise, the French-De Groot model always converges to a consensus value. In particular, the linear averaging dynamics:

$$x(t) = P^t x(0)$$

will converge to a value as $t \to \infty$:

$$\lim_{t\to\infty} x(t) = \alpha \mathbf{1}$$

where $\alpha = \pi' x(0)$ with π the invariant distribution computed in (b). For the simulation, we first need to define $P = D^{-1}\Lambda$, and starting from an initial state $x(0)$, the next state is obtained as $x(1) = Px(0)$, and so on. From the simulation, we can notice that when $t = 50$ we already have a convergence to a consensus value, and it corresponds exactly to the value $\alpha = \pi' x(0)$. An example is shown in the code, where

$$x(0) = \begin{bmatrix} 0 & 1 & 1 & 0 & 1 \end{bmatrix}$$

and after simulating the averaging dynamics 50 times, we get

$$x(50) = \begin{bmatrix} 0.6087 & 0.6087 & 0.6087 & 0.6087 & 0.6087 \end{bmatrix}$$

To check if we have convergence, we compute the eigenvectors of the dominant eigenvalue of $P = D^{-1}D$ and we obtain

$$\pi = \begin{bmatrix} 0.1304 & 0.1739 & 0.2609 & 0.2609 & 0.1739 \end{bmatrix}$$

and we see that the consensus value $\alpha = \pi' x(0) = \mathbf{0.6087}$ is exactly the one obtained with the French-De Groot dynamics.

(f) *Assume that the initial state of the dynamics for each node $i \in V$ is given by $x_i(0) = \xi_i$, where $\xi_i, i \in V$ are i.i.d random variables with variance σ^2. Compute the variance of the consensus value, and compare your results with numerical simulations.*

- For simplicity, we assume that the initial state $x(0)$ is taken from i.i.d uniform distribution $U(0, 1)$. In this case we already know that the variance is by definition $\sigma^2 = 1/12$. In the previous point, we have already proved that when $t \to \infty$, we have convergence towards a consensus value and this implies that the variance will converge to zero as well. This effect can be explained as **wisdom of crowds**, where we can say that the crowd is wiser than a single.

(g) *Remove the edges (d, a) and (d, c). Describe and motivate the asymptotic behavior of the dynamics. If the dynamics converge to a consensus state, how is the consensus value related to the initial condition $x(0)$? Assume that the initial state of the dynamics for each node $i \in V$ is given by $x_i(0) = \xi_i$, where $\xi_i, i \in V$ are i.i.d random variables with variance σ^2. Compute the variance of the consensus value. Motivate your answer.*

- Here is shown the new graph without the nodes mentioned in the text, which we will call G_g

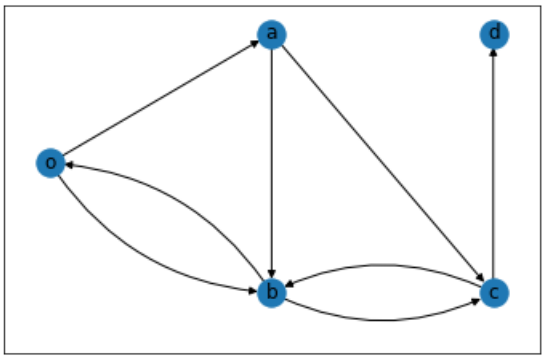

Figure 2: New graph without the nodes (d,a) and (d,c)

We need to do the same analysis done in points (e) and (f), so before starting some simulation, let's check if we can again use the French-De Groot theorem. From the result of the two

functions in networkx library *networkx.is_strongly_connected* and *networkx.is_aperiodic*, the graph is still aperiodic but no more connected. In fact, we could also plot the condensation graph for G_g

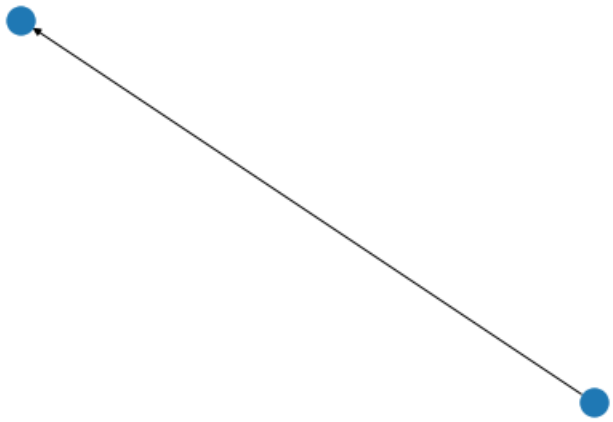

Figure 3: Condensation graph of G_g

By doing some processing on the info related to the condensation graph, we can say that G_g has a sink component composed only by node d. In this case, we cannot provide a general result as before, but we can still conclude something related to the behavior of the linear averaging dynamics as $t \to \infty$. In fact, we still have convergence, meaning that

$$\lim_{t\to\infty} x(t) = \pi' x(0)$$

but in this case, π will only have support on the globally reachable node (node d). In fact, by computing the new invariant distribution, we obtain

$$\pi = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

As consequence, we can say that the opinions of agents not belonging to the globally reachable component have no influence on the final consensus value: the global consensus value is already decided by the value imposed on node d in the initial condition $x(0)$. In the code we present 2 simulations :

$$x^*(0) = \begin{bmatrix} 0 & 1 & 1 & 0 & 1 \end{bmatrix} \tag{4}$$

and

$$x^{**}(0) = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 \end{bmatrix} \tag{5}$$

After performing the average dynamics $x(t) = P^t x(0)$ for $t = 100$, we notice that for (4), the consensus value obtained is 1, and in the second case (for (5)), the consensus value obtained is 0. Those simulation results correspond exactly to the theoretical observations and theorems. Related to the experiment for the variance, we still use the same setting as done in point (f), which is taking as sample i.i.d uniform distribution $U(0, 1)$. As observed for the variance, since we still have convergence towards a consensus value (even if it is only decided by the starting condition of node d), so as consequence the final state's variance is still smaller than the expected variance.

(h) Consider the graph (V, E, Λ), and remove the edges (c, b) and (d, a). Analyze the French-DeGroot dynamics on the new graph. In particular, describe and motivate the asymptotic behavior of the dynamics in terms of the initial condition $x(0)$.

- The new graph without the nodes (c, b) and (d, a) are shown as follows, and for simplicity, we will call it as G_h

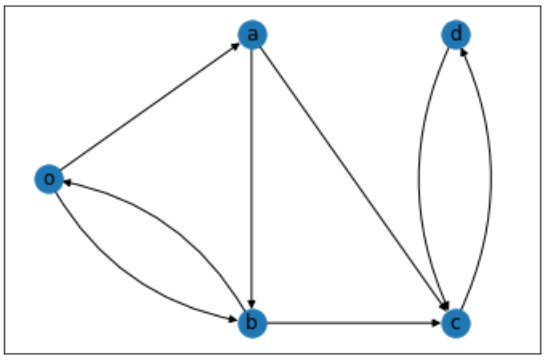

Figure 4: New graph without the nodes (c,b) and (d,a)

Let's check if we can use the French-De Groot theorem by observing the periodicity and connectedness of G_h. We notice that the graph is still aperiodic but is not strongly connected. In particular, let's draw the condensation graph:

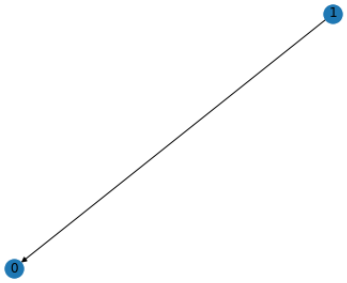

Figure 5: Condensation graph of G_h

We may say that the condensation graph in 3 of point (g) is the same as the one shown above, but if we look into some info related to the condensation graph, we will notice that in this case, the sink component have two nodes inside, which are node c and node d. From the plot 4, we could see that the sink component is periodic of $period = 2$, and this by definition does not lead towards a convergence. But since there's only one sink in the condensation graph, we cannot exclude the case of having a global consensus. Since we have 2 nodes in the condensation graph, here we will simulate 4 French-De Groot dynamics.
Remark: It is important to notice that we have only one sink in the graph, so this means that the invariant distribution π has only support on the globally reachable component. In fact,

$$\pi = \begin{bmatrix} 0 & 0 & 0 & 0.5 & 0.5 \end{bmatrix}$$

This leads to the conclusion that the initial opinions of agents outside this component have no influence on the final consensus (this peculiarity will be shown also during the simulations). The different initial conditions for our simulation is the following:

$$x^{(1)}(0) = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 \end{bmatrix} \tag{6}$$

$$x^{(2)}(0) = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{7}$$

$$x^{(3)}(0) = \begin{bmatrix} 0 & 1 & 1 & 0 & 1 \end{bmatrix} \tag{8}$$

$$x^{(4)}(0) = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 \end{bmatrix} \tag{9}$$

By simulating the French-De Groot dynamics, we notice that the result of the first 2 cases((6) and (7)) returns the same result as $t \to \infty$ (for the experiment we will set t = 100:

$$x^{(1)}(t) = x^{(2)}(t) = \begin{bmatrix} 0.5385 & 0.5769 & 0.2308 & 1 & 0 \end{bmatrix}$$

With this result, we see that the initial condition of the nodes not belonging to the sink has no effect on the final value, and we do not have convergence because by the formula introduced in point (e),

$$\alpha = \pi' x(0)$$

in this case, corresponds to $\alpha = 0.5$.

For the 3rd case (8), the obtained value is also different, because we changed the opinions in the sink.

$$x^{(3)}(t) = \begin{bmatrix} 0.4615 & 0.4231 & 0.7692 & 0 & 1 \end{bmatrix}$$

An interesting case is shown in the (9) case, where in the sink we have the same opinion. In this case

$$x^{(4)}(t) = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

We observe that in the special case when the sink has the same opinion, we will have both consensus over the graph, and also convergence towards the opinion of the sink.

Exercise 2

In this part, we will again consider the network of 1, with weights according to (1). 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 on average it will stay $1/\omega_i$ time-units in a node i before moving to one of its out neighbors. The next node it will visit is based on the probability matrix $P = diag(\omega)^{-1}\Lambda$, where $\omega = \Lambda \mathbf{1}$.

Your task is to simulate this system from two different perspectives: the particle perspective, i.e. "follow the particle", and the node perspective, i.e. "observe from the node"

Remark: For simplicity, this exercise is done again using a global Poisson clock with rate $r = 100$ as suggested by the hint, because the total number of particles is 100 as shown later in the request of the exercises.

For the particle prospective case, at every tick of the system-wide clock, we randomly select which particle to move. This particle is then moved to a neighbor node (which can be itself) based on the transition probability matrix Q (using the same notation of Exercise 1). This means, that if a particle in node o is selected, there is a probability of $1 - \omega_o = 1 - 3/5 = 2/5$ that it will stay in node o, and a probability of $\omega_o = 3/5$ that it will leave the node.

For the node perspective, at each tick of the global clock we randomly, and proportionally to the number of particles in the different nodes, select a node from which we should move a particle. Then a particle from the selected node will move according to the transition probability matrix Q.

a) Particle prospective

1) *If 100 particles all start in node a, what is the average time for a particle to return to node a?*

- In this case we basically run the simulation using the same algorithm that has been used for point (a) of Exercise 1, but when the global clock ticks, we firstly choose randomly a particle over the 100 present in the graph, and then we compute it's next step with the transition probability matrix Q as done before. At the end of the simulation, we simply compute for each particle its simulated invariant distribution, and we do the average over all the particles. In the end, we use the obtained "*average invariant distribution*" to compute the return time with the formula (2). By setting the number of ticks to 20000, the simulated averaged invariant distribution results :

$$\pi_{estimated} = \begin{bmatrix} 0.1931 & 0.1515 & 0.2307 & 0.2162 & 0.2090 \end{bmatrix}$$

and the return time of node a :

$$E_a[T_a^+]_{estimate} = \frac{1}{\pi_{a\_est}\omega_a} = 6.6027$$

2) *How does this compare to the answer in Problem 1, why?*

- It is easy to notice that when we run the algorithm multiple times, since we cannot control the probability, sometimes it shows reasonable results, while in other cases the error is more than 10%. This is caused by the fact that despite running the algorithm with the same number of steps, we have different global times (total time passed from the start to the end of the simulation, not the real-time spent to run the simulation code). For Exercise 1, the global time was more than 10^5, while with 100 particles, despite setting a greater rate, we only arrive at the order of 10^2. After those assumptions, a better way to compare the results would be to not use the number of steps, but instead to set a threshold on the global time.

b) Node prospective

1) *If 100 particles start in node o, and the system is simulated for 60-time units, what is the average number of particles in the different nodes at the end of the simulation?*

- To simulate the moving of different particles in the system, we could reuse the code of the return time, but we save for each clock tick the number of particles in the various nodes. In order to estimate the movement from one node to another, we first check which nodes are not empty, and then we do a random choice over those non-empty nodes. Then, according to the transition probability matrix, we move a particle from the chosen node to its neighbor, thus updating the node's status. We run 100 simulations of the 60-time units experiment, and after averaging the number of nodes over the 100 simulations, and normalizing the result, we obtain

$$\pi_{estimated} = \begin{bmatrix} 0.1803 & 0.1455 & 0.2221 & 0.2233 & 0.2288 \end{bmatrix}$$

which is actually very close to the invariant distribution (3).

2) *Illustrate the simulation above with a plot showing the number of particles in each node during the simulation time.*

- One simulation of the 60-time units experiment is shown in the following graph :

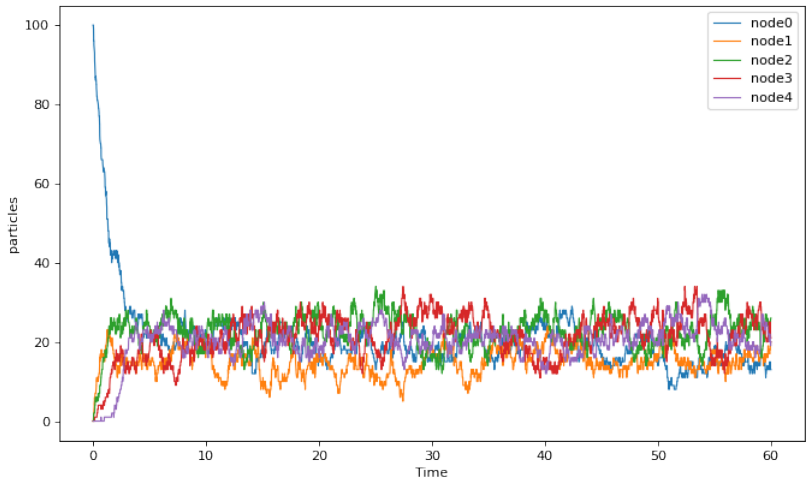

Figure 6: Simulation of a 60-unit times experiment

The final state of the nodes before the threshold of 60-time units is :

$$node\_state = \begin{bmatrix} 13 & 20 & 26 & 21 & 20 \end{bmatrix}$$

Actually, we observe that the convergence is reached already after 5-time units. After the 5-time units, we have just some oscillation of the node state, but overall we can say that the 'equilibrium' is already reached without going further to 60-time units.

3) *Compare the simulation result in the first point above with the stationary distribution of the continuous-time random walk followed by the single particles.*

- We already introduced this result in part 1 of the node perspective. Actually from the result of the simulation, we could generally say that our simulation is done correctly because our scope is to have a similar result with respect to the invariant distribution (3). Despite one single simulation may have a large variance, this variance can be reduced if we do the simulation multiple times and average all the results (This approach is also known as the law of large numbers)

Exercise 3

In this part, we study how different particles affect each other when moving around in a network in continuous time. We consider the open network of 7, with transition rate matrix Λ_{open} according to (10).

$$\mathbf{\Lambda_{open}} = \begin{array}{c} \\ o \\ a \\ b \\ c \\ d \end{array} \begin{array}{c} \begin{array}{ccccc} \bar{o} & \bar{a} & \bar{b} & \bar{c} & \bar{d} \end{array} \\ \begin{bmatrix} 0 & 3/4 & 3/8 & 0 & 0 \\ 0 & 0 & 1/4 & 1/4 & 2/4 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array} \tag{10}$$

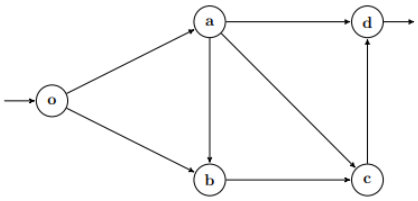

Figure 7: Open network with weights according to (10)

For this system, particles will enter the system at node o according to a Poisson process with input rate $\lambda = 1$. Each node will then pass along a particle according to a given rate, similar to what is done in Problem 2 with the "node perspective". Let $\omega = \Lambda\mathbf{1}$ and let N(t) denote the vector of numbers of particles in each node at time t.

Simulate two different scenarios that differ by what rate the nodes will pass along particles: i) proportional rate, and ii) fixed rate. In scenario i), each node i will pass along particles according to a Poisson process with the rate equal to the number of particles in the node times the rate of the local Poisson clock of node i, i.e., the node i will pass along particles rate with the rate $\omega_i \mathbf{N_i}(t)$. In scenario ii), each node i will instead pass along particles with a fixed rate ω_i. Since node d does not have a node to send its particles to, we assume that $\omega_d = 2$. When the Poisson clock ticks for this node you could simply decrease the number of particles in the node by one (if there are any particles in the node).

Remark: Different from Exercise 2, in this exercise, we will associate a clock for each of the nodes in the graph. In the following plots, a different notation has been used which corresponds to the following mapping:

$$\{node\_o : 0\}, \{node\_a : 1\}, \{node\_b : 2\}, \{node\_c : 3\}, \{node\_d : 4\}$$

a) Proportional rate

1) *Simulate the system for 60-time units and plot the evolution of the number of particles in each node over time.*

- As mentioned in the remark, in this case, the clock associated with each node has a rate that directly depends on the number of particles in that node. Since at the beginning the graph starts with zero particles, we will assume that at the start, only the input and output clocks are running, while the others are set as infinite. When a node is not empty anymore, it starts

its clock according to the number of particles in that instant and its own local Poisson clock rate. To avoid conflicts, when a node that has already started a Poisson clock receives a new particle, the original clock will be replaced with an updated version that has a new rate proportional to the new number of particles. The graph showing the number of particles in each node over time is shown below:

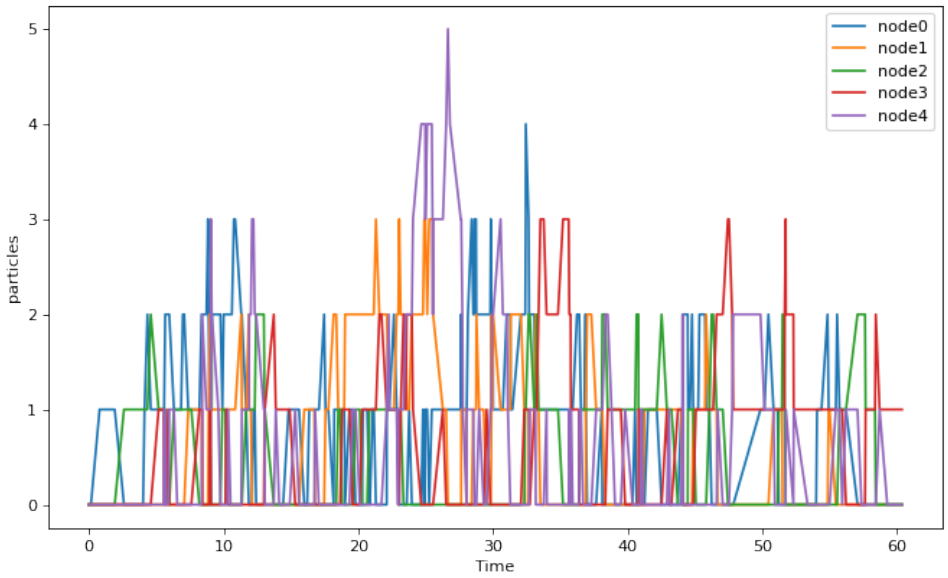

Figure 8: Proportional rate experiment simulation

2) *What is the largest input rate that the system can handle without blowing up?*

- By simulating for different times, we notice that when the set input rate $\lambda = 2$, the system is still able to clean the queue without any problems, but when input rate $\lambda = 3$, we have a blow-up of particles in the node d, as shown in the figure 9

Figure 9: Blow-up simulation with $\lambda = 3$

This behavior may be explained by the fact that since the output rate on node d $\omega_d = 2$ is constant, it has no problem on handling input with the same or lower rate. But when the input rate λ goes beyond, the output node cannot digest all the incoming particles, creating

congestion on the final node. We also point out that since the intermediate nodes have a Poisson clock rate proportional to the number of nodes in their node, it is easy for them to digest the traffic, and since the graph by construction has only one sink node, all the particles will go to the sink node d.

b) Fixed rate

1) *Simulate the system for 60-time units and plot the evolution of the number of particles in each node over time.*

- We use the same approach done in part a), but we point out that in this case the rate is fixed, meaning that we do not need to set some clock to infinite in the beginning, and we also do not need to reset the clocks when a node receives a new particle. Since all the clocks start at the same time, when we have a transition of a particle that is not present in that particular node, we do not have actions on the graph and we simply compute the next tick for that node. The graph showing the number of particles in each node over time is shown below:

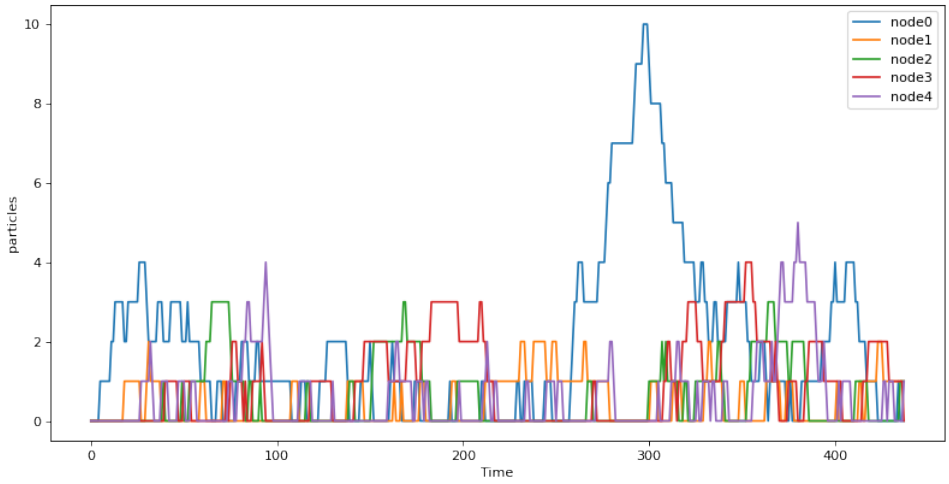

Figure 10: Fixed rate experiment simulation

Concerning the proportional rate, we see that in the starting node we have an increase with respect to the average number of particles per node. This happens because the rates of each node are

$$\omega = \begin{bmatrix} 1.125 & 1 & 1 & 1 & 0 \end{bmatrix} \tag{11}$$

Since those rates are very close to the input rate, we will likely have on average more particles in the transition. This phenomenon would not happen at a proportional rate because when a node has a high concentration of particles, its clock rate increases, and the particles run out faster.

2) *What is the largest input rate that the system can handle without blowing up? Why is this different from the other case?*

- We notice that simulating with input rate $\lambda = 2$, we already have the blow-up situation, as in the figure

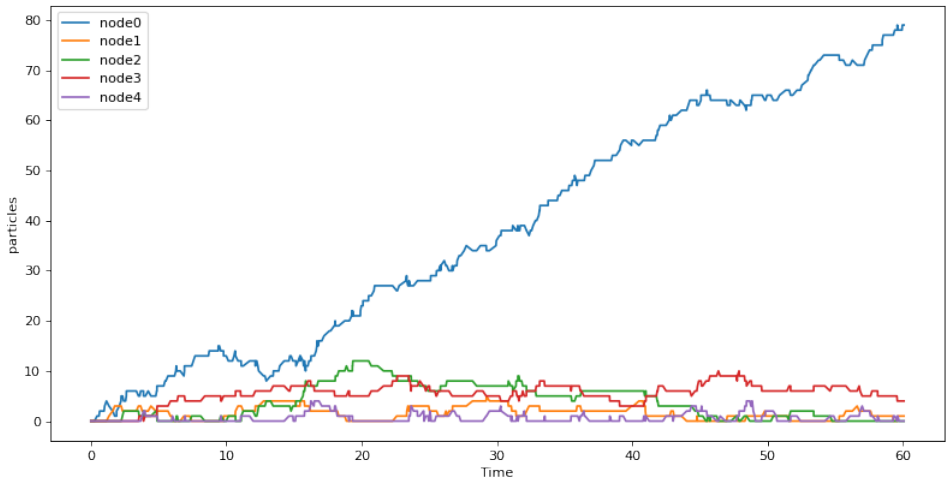

Figure 11: Blow-up simulation with $\lambda = 2$

Notice that in this case, the blow-up situation happens in the input node o, instead of the node d as in the proportional case. The result can be explained by the fact that since the transition rate (11) is fixed, the input node o is not able to digest all the nodes that are coming from outside because its transition rate is lower concerning the input rate (1.25 versus 2).