Second homework Networks Dynamics and Learning

[redacted] [redacted]

Luca Pesce
Politecnico di Torino

[redacted]

***Abstract*—In this first homework assignment, we dealt with topological and algebraic graph theory the assignment is divided into three different exercises :**

- **In the first we are given a network and its link capacities.**
- **In the second there is a list of people with characteristics to link.**
- **In the more realistic third one there is to analyze the lines of highways in the city of Los Angles.**

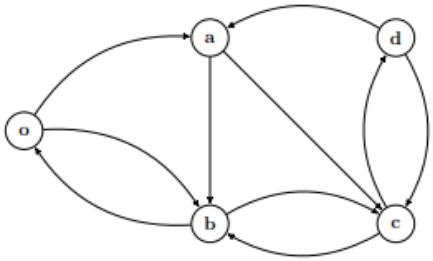

Fig. 1. Closed Network

I. ST EXERCISE

In the following we are reporting an analysis of random walks in continuous time and opinion dynamics within a network controlled by a specific chart. The research combines simulations and calculations to understand particle motion and the dynamics in complex networks. The network is represented in Figure 1.

A. Simulating Random Walks

In the beginning of the study, random walks were simulated on the network which dynamics was described by a given transition rate matrix. The average time for a particle starting from node 'b' to leave and return in this node was estimated doing a simulation of 10000 random walks in the graph. During each walk the choice of traversed nodes was based on probabilities resulting from the normalized transition matrix P and the time it takes for the node to activate is A random variable $T \sim \frac{\ln U}{w_i}$, where w_i is defined as $\sum_j \Lambda_{ij}$.

Average time from b to b is: **4,62 s**

B. Comparison with the theoretical return time

We calculate the thoretical return time with its formula for continous time Markov Chains $E[T_i^+]=\frac{1}{w_i \bar{\pi}}$

Theoretical time from b to b is: **4,60 s**

C. Average time from node o to node d

Continuing on we dealt with the study of node-to-node movement. The average time from node 'o' to node 'd' was computed with the same method as in the point A, but modifying the stopping criteria of the random walk algorithm, we put on it the staring node and the final node as parameters, stopping it when we reach the specified destination. Figure 2 represents a edge sequence of a random walk .

Average time from o to d is: **10,76 s**

D. Comparison with the theoretical hitting time

We compute all the theoretical hitting times from i to d solving this sistem:

$$\begin{cases} \mathbb{E}_i[T_d] = 0 & \text{if } i = d \\ \mathbb{E}_i[T_d] = \frac{1}{w[i]} + \sum_j P_{ij}\mathbb{E}_j[T_d] & \text{if } i \neq d \end{cases}$$

Then we take the index 0 of the vector, corresponding to the theoretical hitting time from o to d.

Theoretical time from o to d is: **10,76 s**

E. French De Groot dynamics

We interpreted the transition matrix as the weight matrix of the graph, then we simulated the French DeGroot dynamics on it starting from various definite initial states for the nodes. The result was that the dynamics converged to consensus in each trial with 15-20 steps. The reason of this is that the graph, in addition to being aperiodic, is also strongly connected, in fact its condensation graph is represented by a single node, therefore the opinion spreads more easily among all nodes.

F. Variance of consensus

The variance of consensus was computed starting by random variables with specific variances as values of the initial condition for the nodes. To compute the consensus variance, 200 trials of French DeGroot were launched, each one with 500 steps. Therefore we computed the dynamics at each run

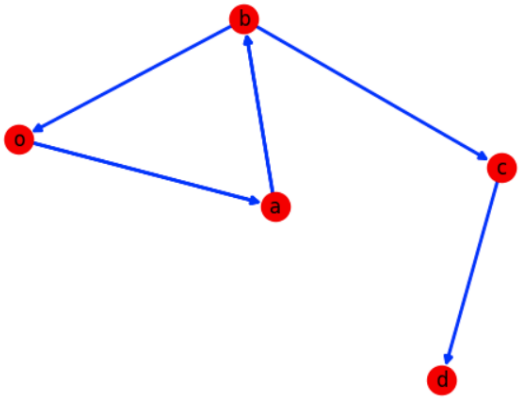

Fig. 2. random walk edge sequence

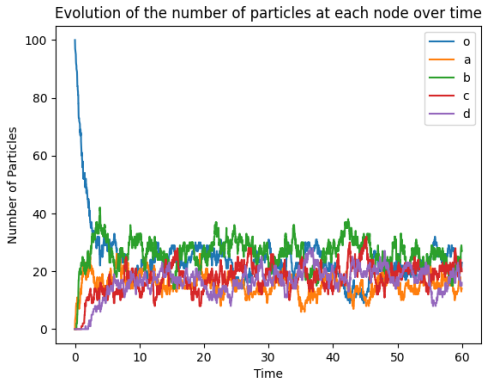

Fig. 3. Particles flow per Node

reaching a specific consensus, then we calculated the squared error from the general consensus value (supposing it equal to 1/2) and the one computed. In the end we averaged the values of every trial to get the variance. *Variance of consensus is*: **0,36**

G. Remove (d,a) and (d,c)

Since the dynamics is governed by the transition matrix, the removal of edges alters the paths that the points can take, affecting the convergence behavior. We removed the edges (d,a) and (d,c) exploring the behaviour, starting from independent random variables with same variance as values of the initial state. We noted that removing the edges leaded to modify the structure of the condensation graph, in fact in this case we have two different connected components (1 source node and 1 sink node). This fact implies that the opinion will be less spread between the nodes and, generally, it flows from the source to the sink. We reached consensus for different initial states, noting that the speed convergence was slower than in the previous case, moreover there was more dependency on the initial conditions.

H. Remove (c,b) and (d,a)

Also in this case we have a condensation graph with 2 nodes, and we always reach consensus.

II. Particle and Node Perspective Simulation

This exercise requires to simulate the movement of different particles within a given network over continuous time. The network is the same represented in the first exercise, a directed graph with weighted edges, and particles move between nodes after spending some time in the node. Two perspectives are explored in the simulation: the particle perspective as in the first exercise, and the node perspective, where the focus is on the flow and number of particles in each node.

A. Particle Perspective

In the particle perspective, 100 particles are simulated to move within the network. The simulation is based on a transition rate matrix P, and the average time for a particle to return to its starting node is calculated and it results in *11s*. Comparing this results with those obtained in the first exercise we saw that they are similar as we expected. In fact for the purpose of calculating average return time moving 100 particles once is like moving one particle 100 times, so it's equal to compute the average return time as in first exercise but for 100 paths.

Average time from b to b for 100 particles: **4,90 s**

B. Node Perspective

In the node perspective, the simulation is carried out by considering each node's particle count over time. The network is initialized with a certain number of particles in the 'o' node, and particles move between nodes based on the clock of each node Characterized by a Poisson random variable with rate $w_i \cdot N_i$, where N_i is the number of particles present in the node at that moment. To simulate the system, we used an approach that controlled the activation of individual clocks on a time-by-time basis. Specifically, each time a clock is activated a particle moves through the system from node to node, enters the system or leaves the system. Once the particle has moved(instantaneously) we control which clock activates next(can also be the clock just activated). We can implement this approach because the Poisson distribution is memoryless, that is, the fact that a clock does not activate for a period of time does not change the probability of when it will activate in the future and same thing if it does. The simulation is run for sixty seconds, and the average number of particles in each node at the end of the simulation is calculated. Our approach showed statistically consistent results. A line plot is generated to visualize the number of particles in each node over time. Represent in Figure 3.
In conclusion, the dual-perspective simulation approach provides a comprehensive understanding of particle dynamics in the given network, offering valuable insights into both individual particle behavior and overall system characteristics.

III. Simulation on a Directed Acyclic Graph

In this exercise we have to simulate a continuous time Markov Chain on a Directed Acyclic Graph where particles

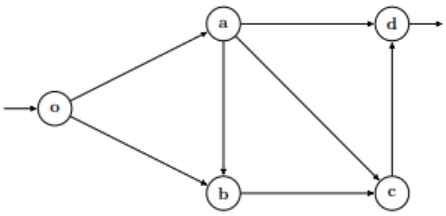

Fig. 4. Particles flow per Node

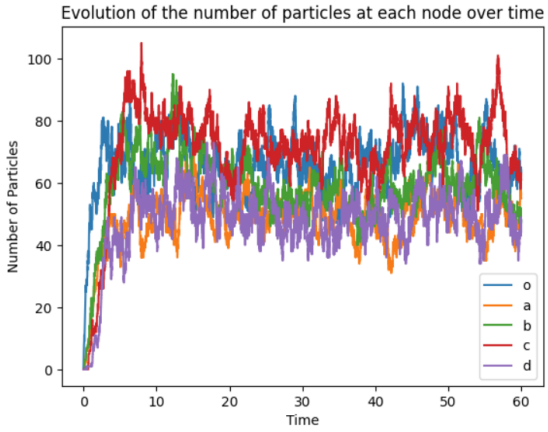

Fig. 5. Particles flow per Node

enter node o and leave node d. We simulate two cases: in the first case the clock rate for each node changes as a function of the number of particles as in exercise 2, in the second case it remains constant. In both scenarios the input rate is constant. We want to find, if it exists, a threshold value of the input rate for which the network blow up, i.e., sending time to infinity causes it to fill infinitely. The network is shown in Figure 4.

A. Variable clock rates

In this section, we simulate the network with variable clock rates for each node. Specifically, the rate of each clock i varies according to the function $w_i \cdot N_i$ as in Exercise 2. In this case, we also have an external clock with a rate of 100 which, each time it activates, introduces a particle into node 'o'. Since node 'd' is a sink, theoretically, its rate should be 0, and its row in the P matrix should consist of only 0's. To simulate the fact that each time it activates, a particle exits the system, we set its rate to 2, and each time it activates, we subtract one particle. Whenever a node $i \neq$d activates, the particle moves from that node to another node j with probability P_{ij} as in the previous two exercises. The simulation is created exactly as in Exercise 2, and in the end, we plot the graph of the number of particles in the nodes over time and the average number of particles per node during the simulation, obtaining the expected results. The number of particles in each node over time fluctuates around an equilibrium position. This provides the foundation to answer the second question, "What is the largest input rate that the system can handle without blowing up?" The network does not blow up for any input rate value because the node rates are proportional to the number of particles. The expected value of T is equal to $1/r$ when T is distributed as $\ln(U)/r$. Thus, the average frequency over long periods will be r. This means that regardless of the average input frequency of a node, there exists a number of particles for which the average output frequency will be equal. This doesn't imply that the particles in the node will remain constant due to statistical variables, but they will stabilize around an equilibrium. All this is consistent with the empirical results of our simulations.

B. Constant clock rates

In this section, we simulate the network with rates for each node w_i and an input rate of 1, similar to the previous section, and the system remains stable. Now, our objective is to determine the maximum input rate for which the system does not diverge. We anticipate that there exists a maximum input rate because, in this scenario, the average frequency with which each node expels particles is constant. Therefore, introducing particles at a higher frequency would cause a node to continuously accumulate particles, leading to an eventual system divergence. After adding a 'source' node and a 'sink' node connected to 'o' and 'd,' respectively, we approach this problem as an optimization task to find the maximum incoming flow into the 'o' node while maintaining system equilibrium. To mathematically formalize the constraints, we use formulations involving the incidence matrix B, an incidence matrix B that considers only outgoing edges from each node, and the maximum flow vector w. To keep the system in equilibrium, the total incoming flow to each node must equal the outgoing flow, and the outgoing flow from a node i cannot exceed w_i, which is its average outgoing flow. We find that the maximum incoming flow is $4/3$, and it can be easily observed that the bottleneck is the 'c' node.