Lecture Notes

Week 3: Random networks and branching processes

1 Networks

Definition 1. *A **Network** (or Graph) is a collection of entities called **Nodes** (or Vertices) and the relationships or connections between them, termed as **Edges** (or Links). Each edge connects two nodes and indicates a relationship between them.*

Example 1. *We define the set of nodes V and the set of edges E as follows:*

$$V = \{1, 2, 3, 4, 5\} \tag{1}$$

$$E = \{(1, 2), (2, 3), (1, 3), (2, 4), (3, 4), (4, 5)\} \tag{2}$$

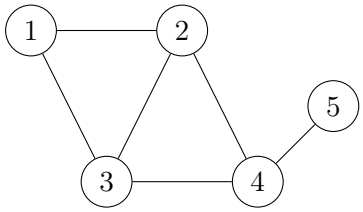

Figure 1: Example network with 5 nodes and 6 edges.

After defining networks in terms of nodes and edges, a mathematical representation is essential for analysis. The adjacency matrix offers a compact way to depict the relationships within a network.

Definition 2. *The **Adjacency Matrix** A is a square matrix of size $N \times N$ where N is the total number of nodes. Entry A_{ij} equals 1 if there's an edge from node i to node j and 0 otherwise.*

Example 2. *The adjacency matrix for the network in the previous example is:*

$$A = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

An adjacency matrix is a powerful tool for representing graphs because it encodes all the information about connections between nodes in a systematic manner. The cell A_{ij} represents the edge from node i to node j. If $A_{ij} = 1$, then an edge exists; otherwise, it's zero. This binary encoding simplifies complex relationships into a format easily analyzed computationally.

Definition 3. *A **Shortest Path** between two nodes u and v in a graph is a path that has the minimum number of edges (in an unweighted graph) or the minimum sum of edge weights (in a weighted graph) among all possible paths between u and v. The length of this path is denoted as $d(u, v)$.*

Definition 4. *The **Diameter** of a graph is the longest shortest path between any two nodes in the graph. Formally, if $d(u, v)$ is the shortest path between nodes u and v, then the diameter D is defined as:*

$$D = \max_{u,v \in V} d(u, v)$$

Example 3. *Let's consider the previous network example, where $V = \{1, 2, 3, 4, 5\}$ and $E = \{(1, 2), (2, 3), (1, 3), (2, 4), (3, 4), (4, 5)\}$.*

The shortest paths between all pairs of nodes are:

- *Shortest path from 1 to 2, 3, 4, 5 are $1, 1, 2, 3$ respectively.*
- *Shortest path from 2 to 1, 3, 4, 5 are $1, 1, 1, 2$ respectively.*
- *Shortest path from 3 to 1, 2, 4, 5 are $1, 1, 1, 2$ respectively.*
- *Shortest path from 4 to 1, 2, 3, 5 are $2, 1, 1, 1$ respectively.*
- *Shortest path from 5 to 1, 2, 3, 4 are $3, 2, 2, 1$ respectively.*

The diameter of this graph is the longest of these shortest paths, which in this case is 3 (from node 1 to node 5).

An interesting property of adjacency matrices is that they can be multiplied to find paths of varying lengths between nodes. Specifically, A^2 (the matrix A multiplied by itself) will give us all possible paths of length 2 between any two nodes i and j.

For example, the element $(A^2)_{ij}$ will represent the number of paths of length 2 between nodes i and j. The logic behind this comes from the nature of matrix multiplication, where each element in the resulting matrix is computed as a sum of products involving elements from the corresponding row and column of the original matrices. In the context of networks, this translates to summing up possible intermediary steps to form paths of the length in question.

Example 4. *Given the adjacency matrix A from our previous example:*

$$A = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

We can calculate A^2 by multiplying A by itself. The resulting matrix A^2 will represent the number of paths of length 2 between any two nodes i and j.

$$A^2 = \begin{pmatrix} 2 & 1 & 1 & 1 & 0 \\ 1 & 3 & 2 & 1 & 1 \\ 1 & 2 & 3 & 1 & 1 \\ 1 & 1 & 1 & 2 & 0 \\ 0 & 1 & 1 & 0 & 1 \end{pmatrix}$$

For instance, the element $(A^2)_{13} = 1$ tells us there is exactly 1 path of length 2 from node 1 to node 3. This path is $1 \to 2 \to 3$.

Similarly, the element $(A^2)_{24} = 1$ indicates there is one path of length 2 from node 2 to node 4. This path is $2 \to 3 \to 4$.

Knowing the shortest paths and diameters in a network has a wide range of practical applications. For instance, in social networks, the diameter can provide insights into how quickly information may spread across the network. In transportation networks, identifying the shortest paths is crucial for optimizing travel routes. Understanding these properties is integral for network resilience, efficiency, and information dissemination.

Before delving into the fascinating properties of random graphs, let's establish what it means for a set of nodes to be connected within a graph. This idea naturally extends from our discussion about paths of varying lengths.

Definition 5 (Connected Component)**.** *A **Connected Component** in a network is a maximal set of nodes C such that for every pair of nodes a, b in C, there exists an undirected path from a to b.*

To identify connected components, one may start at an arbitrary node and find all nodes reachable from it. This set forms one connected component, and the process is repeated until all nodes are included in a connected component.

Example 5. *Consider our earlier example network with adjacency matrix A as follows:*

$$A = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

We already calculated A^2 which provides us with all paths of length 2 between any two nodes i and j:

$$A^2 = \begin{pmatrix} 2 & 1 & 1 & 2 & 0 \\ 1 & 3 & 2 & 1 & 1 \\ 1 & 2 & 3 & 1 & 1 \\ 2 & 1 & 1 & 2 & 0 \\ 0 & 1 & 1 & 0 & 1 \end{pmatrix}$$

Now, let's calculate A^3 to find all paths of length 3 between any two nodes:

$$A^3 = A \times A^2 = \begin{pmatrix} 0 & 3 & 3 & 1 & 1 \\ 3 & 0 & 3 & 3 & 1 \\ 3 & 3 & 0 & 3 & 1 \\ 1 & 3 & 3 & 0 & 2 \\ 1 & 1 & 1 & 2 & 0 \end{pmatrix}$$

For instance, the element $(A^3)_{14}$ is 1, which means there is one path of length 3 between node 1 and node 4. Similarly, $(A^3)_{52}$ is also 1, indicating a single path of length 3 between nodes 5 and 2.

These calculations reinforce the notion that the network consists of a single connected component $C = \{1, 2, 3, 4, 5\}$, as all nodes are reachable from one another via paths of varying lengths.

Having reviewed the basic notations and mathematical properties of networks, in the upcoming sections, we'll delve deeper into various network models. These models will help illuminate the structures and intricacies of networks we encounter daily, from social media connections to vast communication networks. With these tools, we aim to foster a richer understanding of how systems and patterns interconnect in the world around us.

2 Random Graphs

Random graphs are a foundational construct in the mathematical treatment of network theory, offering researchers a framework for understanding the probabilistic interactions within complex networks.

Definition 6 (Random Network)**.** *A **Random Network** is a graph in which the presence or absence of an edge between any two distinct nodes is determined by a random process or probabilistic rule.*

Among various random graph models, the Erdos-Renyi model stands out due to its simplicity and foundational role in network theory.

Definition 7 (Erdos-Renyi Model)**.** *The **Erdos-Renyi Model**, denoted as $G(n, p)$, is defined as a random graph consisting of n nodes, where each potential edge between distinct nodes i and j is included with probability p, independently of the other edges.*

The probability of an edge forming between any two nodes i and j in the Erdos-Renyi model $G(n, p)$ is:

$$P(\text{Edge between } i \text{ and } j) = p \tag{3}$$

Definition 8 (Giant Connected Component)**.** *A **Giant Connected Component** in a graph is a connected component that includes a substantial fraction of the entire set of nodes in the graph. In the Erdos-Renyi model, a giant connected component emerges when the edge probability p surpasses a critical value p_c.*

The critical probability p_c for the emergence of a giant component in an Erdos-Renyi graph $G(n, p)$ is approximately:

$$p_c \approx \frac{\log(n)}{n} \tag{4}$$

Experiment 1 (Empirical Estimation of p_c)**.** *To empirically estimate the critical probability p_c at which a giant connected component forms in an Erdos-Renyi graph $G(n, p)$, we will perform the following experiment:*

1. *Initialize $n = 1000$ nodes.*
2. *Vary p from 0 to 1 in increments of 0.01.*
3. *For each p, generate a random graph $G(n, p)$.*
4. *Identify the largest connected component C in G.*
5. *Record the size $|C|$ of the largest connected component.*
6. *Plot $|C|$ as a function of p.*
7. *Observe the value of p where $|C|$ starts to dramatically increase. This value is an empirical estimate of p_c.*

The Erdos-Renyi model, despite its simplistic assumptions, serves as a key reference model in the realm of network theory. Its clear framework for randomness offers a foundation for the study of more complex networks, making it a cornerstone in disciplines such as computer science, biology, and social sciences.

2.1 Preferential Attachment Model

The Preferential Attachment Model, popularized by Albert-László Barabási and Réka Albert, is grounded in the adage "the rich get richer." Nodes are more likely to link to nodes that already have many connections. This dynamic leads to "hubs" or nodes with significantly higher connectivity than others.

Mathematically, the probability $\Pi(k)$ that a new node connects to a node with k connections is proportional to k. This results in a scale-free network, characterized by a power-law degree distribution.

The Preferential Attachment Model is a foundational concept often used to explain how real-world networks evolve to exhibit a scale-free degree distribution. Proposed by Barabási and Albert in 1999, this model argues that networks grow by the principle of "the rich get richer," whereby new nodes are more likely to connect to already well-connected nodes.

The central equation governing the Preferential Attachment Model is:

$$\Pi(k) = \frac{k}{\sum_j k_j} \tag{5}$$

where $\Pi(k)$ is the probability that a new node will connect to a node with degree k, and the sum runs over all nodes j in the network.

The Preferential Attachment Model provides a basis for understanding how highly connected "hubs" emerge in networks. These hubs play a critical role in the network's overall structure and resilience, often dominating processes like information spread or failure propagation.

The Preferential Attachment Model finds applications in various fields, including the World Wide Web, citation networks, and even biological systems, to explain phenomena like protein-protein interaction networks.

3 Branching processes

Branching processes are stochastic models representing the evolution of populations, where each individual produces offspring according to a specified probability distribution, independently of others. These models are widely applied across various fields such as biology, for modeling population dynamics and disease spread, and physics, for particle decay simulations.

Definition 9 (Branching Process)**.** *Let $\{Z_n\}_{n=0}^{\infty}$ be a sequence of random variables where Z_n represents the number of individuals in the n-th generation. The branching process is defined by the initial condition $Z_0 = 1$ and the recursive relation*

$$Z_{n+1} = \sum_{i=1}^{Z_n} X_{n,i},$$

where $\{X_{n,i}\}$ are independent and identically distributed random variables representing the number of offspring produced by the i-th individual of the n-th generation, following a common probability distribution.

The Galton-Watson process is a classic example of a branching process, primarily used to analyze extinction probabilities, such as the likelihood of a family name dying out. This model and its extensions allow for the exploration of critical population thresholds, growth rates, and long-term survival chances under varying reproductive assumptions.

Definition 10 (Galton-Watson Process)**.** *A* Galton-Watson process *is a type of branching process where each individual in a generation produces a random number of offspring according to a fixed probability distribution, independently of the other individuals. The process starts with a single ancestor, and the population evolves over discrete generations.*

Formally, let $\{Z_n\}_{n=0}^{\infty}$ denote the sequence of random variables representing the number of individuals in the n-th generation. The Galton-Watson process is defined by the initial condition $Z_0 = 1$ and the recurrence relation

$$Z_{n+1} = \sum_{i=1}^{Z_n} X_{n,i},$$

where $\{X_{n,i}\}$ are independent and identically distributed random variables representing the number of offspring produced by the i-th individual of the n-th generation. The common distribution of $\{X_{n,i}\}$ is known as the offspring distribution.

Example 6. *Consider a simple Galton-Watson process with a binary offspring distribution where each individual has either 0 or 2 offspring with equal probability* 0.5*. This process models a population where each individual has a* 50% *chance of having no offspring and a* 50% *chance of having two offspring, independent of the others.*

In this example, the population can either become extinct if all individuals have no offspring or grow exponentially if individuals continue to have two offspring. The probability of eventual extinction or unbounded growth depends on the initial parameters of the process, which in this case, hinges on the binary nature of the offspring distribution.

Consider a population model where individuals produce offspring. Let X_n represent the population size in the n-th generation. Each individual in generation $n-1$ produces offspring according to a probability distribution $\{P_j\}_{j\geq 0}$, where P_j is the probability of producing j offspring.

Definition 11 (Mean number of offspring)**.** *The mean number of offspring produced by an individual, denoted by μ, is given by the expected value:*

$$\mu = \sum_{j=0}^{\infty} jP_j.$$

Definition 12 (Variance of offspring)**.** *The variance in the number of offspring produced by an individual, denoted by σ^2, is:*

$$\sigma^2 = \sum_{j=0}^{\infty} (j-\mu)^2 P_j.$$

For the branching process, the state space is the set of nonnegative integers, with state 0 indicating extinction. If $\mu \leq 1$, the population tends to die out, whereas if $\mu > 1$, there's a potential for indefinite growth.

Example 7 (Calculation of μ and σ^2)**.** *If $P_0 = 0.5$, $P_2 = 0.5$, and $P_j = 0$ for $j \neq 0, 2$, then $\mu = 0 \times 0.5 + 2 \times 0.5 = 1$ and $\sigma^2 = (2-1)^2 \times 0.5 = 0.5$.*

Definition 13 (Extinction Probability)**.** *The probability that the population eventually dies out, denoted by π_0, is particularly interesting. For $\mu \leq 1$, $\pi_0 = 1$, indicating certain extinction. For $\mu > 1$, determining π_0 requires solving:*

$$\pi_0 = \sum_{j=0}^{\infty} (\pi_0)^j P_j.$$

Example 8 (Determination of π_0)**.** *Given $P_0 = 1/3$, $P_1 = 1/3$, and $P_3 = 1/3$, we find $\mu = 1$ (indicating critical population growth) and must solve the equation above to find π_0.*

This approach leverages the Markov chain properties of branching processes to analyze population dynamics over generations, focusing on mean offspring numbers, variance, and extinction probabilities.

Time Reversible Markov Chain

Time Reversible Markov Chains facilitate analysis of Markov processes in reverse time while preserving Markovian characteristics. For a Markov chain with:

- Transition probabilities: P_{ij}, transitioning from state i to state j.
- Stationary distribution: π_i, the long-term state probabilities.

The reversed chain's transition probabilities, Q_{ij}, are calculated as:

$$Q_{ij} = \frac{\pi_j P_{ji}}{\pi_i} \tag{6}$$

For time reversibility, the detailed balance condition must be satisfied:

$$\pi_i P_{ij} = \pi_j P_{ji}, \quad \forall i, j \tag{7}$$

This condition ensures the forward and reverse processes are indistinguishable at equilibrium, reflecting a symmetry in transitions.

Implications of Time Reversibility

Time reversibility imposes a unique equilibrium structure on the Markov chain, leading to:

1. **Symmetric Behavior:** The chain exhibits a balance in transitions, allowing forward and backward analysis.
2. **Ergodicity:** For ergodic chains, time reversibility ensures uniform mixing over time, strengthening the chain's stochastic properties.
3. **Inference and Estimation:** Time reversibility simplifies the estimation of transition probabilities and stationary distributions, facilitating easier model parameterization.

Monte Carlo Markov Chain (MCMC)

Monte Carlo Markov Chain (MCMC) methods are essential for sampling from complex probability distributions by constructing a Markov chain that has the desired distribution as its stationary distribution. The main objective is to utilize these samples for approximating integrals, optimizing functions, and exploring properties of distributions that are difficult to analyze analytically.

An MCMC method constructs a Markov chain such that for every pair of states x and y, the detailed balance condition with respect to π is satisfied:

$$\pi_x P_{xy} = \pi_y P_{yx}, \quad \forall x, y \tag{8}$$

Metropolis-Hastings Algorithm:

1. Begin with an initial state x_0.
2. For each iteration $t = 1, 2, \ldots, T$:
   (a) Propose a new state y from a proposal distribution $q(y|x_{t-1})$.
   (b) Calculate the acceptance ratio $a = \frac{\pi_y q(x_{t-1}|y)}{\pi_{x_{t-1}} q(y|x_{t-1})}$.

(c) Accept the new state with probability $\min(1, a)$, resulting in $x_t = y$; otherwise, retain $x_t = x_{t-1}$.

Gibbs Sampling: This algorithm is particularly useful for sampling from multivariate distributions. It sequentially updates each component of the state vector, sampling from the conditional distribution of each component given all other components, thus facilitating efficient exploration of the state space.

Illustration: Sampling from a Biased Coin Distribution

To demonstrate the application of MCMC, consider sampling from the distribution of a biased coin's outcomes, where the probability of heads (H) is $\pi_H = 0.7$ and tails (T) is $\pi_T = 0.3$.

Procedure:

1. Define the target distribution $\pi(x)$ with $\pi_H = 0.7$ and $\pi_T = 0.3$.
2. Employ a proposal distribution $q(y|x)$ that allows transitions between states with a certain probability.
3. Use the Metropolis-Hastings algorithm to accept or reject proposed transitions, thereby generating a sample from the target distribution.

Proof of Convergence for the Metropolis-Hastings Algorithm

To demonstrate that the Markov chain generated by the Metropolis-Hastings algorithm converges to the desired stationary distribution π, we rely on the detailed balance condition. This condition is essential for ensuring that the Markov chain is time-reversible and reaches equilibrium.

Detailed Balance Condition

The detailed balance condition for a Markov chain with stationary distribution π is given by:

$$\pi_i P_{ij} = \pi_j P_{ji}, \quad \forall i, j \tag{9}$$

Metropolis-Hastings Algorithm Steps

The Metropolis-Hastings algorithm involves two steps: proposal and acceptance-rejection.

1. **Proposal:** Propose a transition from the current state i to a new state j using a proposal distribution q_{ij}.

2. **Acceptance-Rejection:** The acceptance ratio a_{ij} for transitioning from state i to state j is computed as:

$$a_{ij} = \frac{\pi_j q_{ji}}{\pi_i q_{ij}} \tag{10}$$

The transition is accepted with probability $\alpha_{ij} = \min(1, a_{ij})$, resulting in the effective transition probability:

$$P_{ij} = q_{ij}\alpha_{ij} \tag{11}$$

For non-accepted transitions, the chain remains in the current state, ensuring the probabilities sum to 1.

Proof of Convergence Using Detailed Balance

To verify that the detailed balance condition is satisfied and thus prove convergence to the distribution π, we examine the effective transition probabilities:

$$\begin{aligned} \pi_i P_{ij} &= \pi_i q_{ij} \alpha_{ij} \\ &= \pi_i q_{ij} \min\left(1, \frac{\pi_j q_{ji}}{\pi_i q_{ij}}\right) \end{aligned}$$

For the reverse transition from state j to state i, the symmetry in the min function allows us to write:

$$\begin{aligned} \pi_j P_{ji} &= \pi_j q_{ji} \alpha_{ji} \\ &= \pi_j q_{ji} \min\left(1, \frac{\pi_i q_{ij}}{\pi_j q_{ji}}\right) \end{aligned}$$

By design, the Metropolis-Hastings algorithm's acceptance-rejection step ensures that for all state pairs (i, j), the detailed balance condition:

$$\pi_i P_{ij} = \pi_j P_{ji} \tag{12}$$

is satisfied, proving that π is indeed the stationary distribution of the Markov chain generated by the algorithm.

4 Markov Decision Processes (MDP)

A Markov Decision Process (MDP) provides a mathematical framework for modeling decision-making in situations where outcomes are partly random and partly under the control of a decision-maker. MDPs are useful in studying optimization problems solved via dynamic programming and reinforcement learning.

An MDP is defined as a tuple (S, A, P, R) where:

- S is a finite set of states.
- A is a finite set of actions.
- P is the state transition probability function, $P_{ij}(a)$, representing the probability of transitioning from state i to state j under action a.
- R is the reward function, $R(i, a)$ or $R(i, j, a)$, specifying the reward received when transitioning from state i to state j due to action a.

The goal within an MDP framework is to discover a policy π that maximizes the expected cumulative reward from any initial state over a horizon, which can be finite or infinite.

A policy π specifies the action a to be taken in each state s. The optimal policy π^* is the one that maximizes the expected cumulative reward over time, which can be found using dynamic programming techniques, such as value iteration or policy iteration, or through direct policy search methods in reinforcement learning contexts.

The challenge in solving MDPs lies in the trade-off between immediate rewards and long-term gains, requiring careful consideration of future state probabilities and rewards when choosing actions.

Definition 14 (Markov Decision Process)**.** *A Markov Decision Process is defined by a tuple (S, A, P, R), where S is the set of states, A is the set of actions, P is the state transition probability function $P : S \times A \times S \to [0, 1]$, and R is the reward function $R : S \times A \to \mathbb{R}$.*

A policy β specifies the action to be chosen in each state, potentially as a probability distribution over actions. The goal is to identify a policy that maximizes the expected average reward over time. For any policy β, the expected reward when taking action a in state i is given by $R(i, a)$. The steady-state probability of being in state i and taking action a under policy β is denoted by π_{ia}, satisfying:

$$\sum_{a \in A} \pi_{ia} = 1, \quad \forall i \in S \tag{13}$$

$$\pi_{ja} = \sum_{i \in S} \pi_{ia} P_{ij}(a), \quad \forall j \in S, a \in A \tag{14}$$

The expected average reward under policy β is then given by:

$$\text{Expected Average Reward}(\beta) = \sum_{i \in S} \sum_{a \in A} \pi_{ia} R(i, a) \tag{15}$$

The aim is to find a policy β that maximizes this expected average reward.

Example 9. ***Grid Navigation MDP:*** *Consider a grid where an agent aims to move from a start position to a goal position with minimal steps. The states S represent grid cells, and the actions $A = \{up, down, left, right\}$ move the agent between states. Assume all movements have a reward of -1, encouraging the shortest path, and reaching the goal yields a reward of $+10$.*

If the agent attempts to move into a wall or outside the grid, it remains in its current state. Let's consider a simplified 2x2 grid with states $\{1, 2, 3, 4\}$, where state 4 is the goal, and state 3 is an obstacle. The transition probabilities $P_{ij}(a)$ for moving from state i to state j with action a might look like:

$$P_{11}(up) = 0, \qquad P_{11}(right) = 1, \qquad \textit{and so on.}$$

The reward function $R(i, a)$ for moving from state i using action a:

$$R(1, right) = -1, \qquad R(2, right) = +10.$$

To determine the optimal policy, one could use value iteration to calculate the value of each state and derive the policy that maximizes rewards.

Example 10. ***Gambling MDP:*** *A gambler with an initial wealth of W dollars can bet any integer amount $b \leq W$ on a fair coin toss. Winning doubles the bet, while losing forfeits it. Here, states S represent the gambler's current wealth, and actions A are the possible bet sizes. The goal is to reach a wealth of W_{max}.*

The state transition probabilities for betting b dollars are:

$$P_{W,W+b}(b) = 0.5, \qquad P_{W,W-b}(b) = 0.5.$$

The reward function $R(W, b)$ could be defined as:

$$R(W, b) = \begin{cases} +1 & \text{if } W + b = W_{max} \\ -1 & \text{if } W - b = 0 \\ 0 & \text{otherwise} \end{cases}$$