capstone week 5

Salah Omar

April 2025

Contents

1 Introduction

The growing pentagon network is a variant of the preferential attachment model where each new node attaches to an existing node with probability proportional to the degree of the existing node. We impose a degree limit of $k = 4$ to prevent any node from exceeding a degree of 4. This modification provides a realistic growth model and ensures that the network does not become overly connected in the long term.

2 Generating Function for Degree Distribution

We aim to analyze the degree distribution of the network using generating functions. The generating function $G(x)$ is defined as the generating function for the degree distribution, i.e.,

$$G(x) = \sum_{k=0}^{\infty} P(k)x^k$$

where $P(k)$ is the probability that a randomly chosen node has degree k. The generating function encapsulates the degree distribution and can be used to track how the distribution evolves as the network grows.

2.1 Initial Configuration

We start with a pentagon, where each of the 5 nodes has degree 2. The degree distribution at this stage is given by $P(k) = \delta_{k,2}$, meaning that the degree of each node is 2. The corresponding generating function is

$$G(x) = x^2$$

2.2 Growth Process

At each time step, a new node is added to the network. The new node connects to existing nodes according to the preferential attachment rule, with the probability of attaching to node i being proportional to the degree of node i. We also impose a degree limit of 4, meaning no node can have more than 4 edges.

We update the generating function as follows:

$$G(x) \to G(x) \cdot (1 + x + x^2 + x^3 + x^4)$$

This update reflects the degree limitation. The factor $(1 + x + x^2 + x^3 + x^4)$ accounts for the fact that each new edge can increase the degree of the existing nodes by at most 4, ensuring no node exceeds the degree limit.

3 Recursive Update of the Generating Function

After each new node is added, the generating function is updated. The recursive relation for the generating function is:

$$G_{n+1}(x) = G_n(x) \cdot (1 + x + x^2 + x^3 + x^4)$$

where $G_n(x)$ is the generating function after the n-th node has been added. Initially, $G_0(x) = x^2$.

3.1 Degree Distribution at Large N

As the network grows, the generating function converges to a steady-state form that can be used to approximate the degree distribution for large N. By solving the recursion and performing a series expansion, we can recover the asymptotic behavior of the degree distribution.

For large N, the degree distribution $P(k)$ approaches a power-law distribution, where the probability $P(k)$ behaves like:

$$P(k) \sim k^{-\gamma}$$

where γ is the degree exponent. This reflects the "scale-free" nature of the network, where a few nodes (hubs) have very high degrees, and most nodes have low degrees.

3.2 Conclusion

By using the generating function approach, we have derived the degree distribution for the growing pentagon network with a degree limit of $k = 4$. The recursive update of the generating function provides a tractable way to analyze the degree distribution as the network grows. The imposition of the degree limit ensures that no node exceeds a degree of 4, making the model more realistic and controlled. The degree distribution eventually follows a power-law, demonstrating the emergence of hubs in the network.

4 Derivation of the Law of Cosines for Hyperbolic Space

The law of cosines for hyperbolic space is a modification of the Euclidean version, adapted for the non-Euclidean geometry of hyperbolic space. In this model, distances and angles behave differently due to the curvature of the space.

4.1 Hyperbolic Space and the Law of Cosines

In hyperbolic geometry, the law of cosines relates the sides and angles of a triangle in a way similar to Euclidean geometry but adjusted for the hyperbolic metric.

Let a, b, and c be the hyperbolic distances between the vertices of a triangle in hyperbolic space. Let θ be the angle between the sides a and b, and c be the side opposite to this angle. The law of cosines in hyperbolic space is given by:

$$\cosh(c) = \cosh(a)\cosh(b) - \sinh(a)\sinh(b)\cos(\theta)$$

Where: - cosh and sinh are the hyperbolic cosine and sine functions, respectively. - θ is the angle between sides a and b, and c is the side opposite this angle.

4.1.1 Interpretation of the Formula

The formula is similar to the Euclidean law of cosines but incorporates the hyperbolic trigonometric functions cosh and sinh. These terms arise from the curvature of hyperbolic space, which affects how distances and angles relate.

4.1.2 Derivation in the Poincaré Disk Model

In the Poincaré disk model, hyperbolic space is represented as the interior of a unit disk. The distance between two points z_1 and z_2 in this model is given by the hyperbolic distance formula:

$$d(z_1, z_2) = \operatorname{arcosh}\left(1 + \frac{|z_1 - z_2|^2}{2(1 - |z_1|^2)(1 - |z_2|^2)}\right)$$

This formula allows us to compute distances in hyperbolic space and helps derive the law of cosines for triangles in this geometry.

4.2 Conclusion

The law of cosines for hyperbolic space provides a way to relate the angles and distances in hyperbolic triangles. It is a crucial tool for understanding the geometry of hyperbolic space, which has applications in various fields, including general relativity, cosmology, and network theory.

article amsmath

Area of a Hyperbolic Triangle and Angle Deficit

5 Area of a Hyperbolic Triangle

In hyperbolic geometry, the area of a triangle is related to its angle deficit. The area A of a hyperbolic triangle is given by the following formula:

$$A = \pi - (\alpha + \beta + \gamma)$$

Where: - α, β, and γ are the interior angles of the triangle. - π is the total angle sum in Euclidean geometry.

This formula shows that the area of a hyperbolic triangle is the difference between π and the sum of its interior angles, which will always be less than π. This reflects the curvature of the hyperbolic space.

5.1 Calculating the Area of $rqinf$

Let α, β, and γ be the interior angles of the triangle in the region $rqinf$. Using the formula for the area of a hyperbolic triangle, we can calculate the area A as:

$$A = \pi - (\alpha + \beta + \gamma)$$

To verify that the area of the triangle adds up correctly, we will check if the sum of the angles $\alpha + \beta + \gamma$ is less than π. For this verification, we compute the angle deficit.

5.2 Verification of the Area Formula

The formula for the area A of a hyperbolic triangle ensures that the area is directly related to the angle deficit. We can verify this by calculating the sum of the angles and checking the following:

$$A = \pi - (\alpha + \beta + \gamma)$$

If the sum of the angles is less than π, the area will be positive, and the calculation will hold true.

6 Popularity Similarity Model

The Popularity Similarity Model is a network construction model where the likelihood of two nodes forming an edge is based on their degree similarity. This model captures the idea that nodes with similar degrees (popularity) are more likely to be connected, reflecting the nature of many real-world networks, where popular nodes tend to connect with other popular nodes.

6.1 Model Description

In this model:

1. A degree sequence is generated for the network, typically from a Poisson distribution. The degree of each node represents its popularity.
2. The degree similarity between two nodes is computed as the inverse of the absolute difference in their degrees. Nodes with similar degrees have a higher likelihood of being connected.
3. Edges are added between pairs of nodes whose degree similarity exceeds a certain threshold. The higher the degree similarity, the more likely the edge is to be formed.

This model allows us to generate networks where the structure is heavily influenced by the degree similarity between nodes, providing insights into the assortative or disassortative nature of the network.

6.2 Methodology

The steps for generating a network based on the Popularity Similarity Model are as follows:

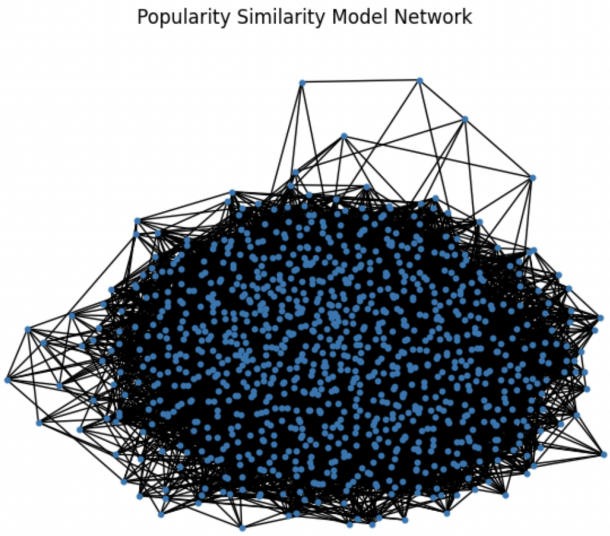
1. Generate a degree sequence based on a Poisson distribution, where the average degree p_{degree} is specified.
2. Check for graphicality of the degree sequence to ensure that it is realizable as a simple graph.
3. Generate a configuration model network based on the degree sequence, which ensures that the resulting network has the specified degree distribution.
4. Compute the similarity between every pair of nodes based on their degrees.
5. Add edges between nodes whose degree similarity exceeds a given threshold, which reflects the popularity similarity between nodes.

6.3 Python Code for Popularity Similarity Model

The following Python code implements the Popularity Similarity Model using the 'NetworkX' library. The code generates a network with a given degree distribution, computes the similarity between nodes, and adds edges based on the similarity threshold.

```
import networkx as nx
import numpy as np
import matplotlib.pyplot as plt


def generate_network(N, p_degree):

    while True:
        degree_sequence = np.random.poisson(p_degree, N)
        if nx.is_graphical(degree_sequence):
            break


    G = nx.configuration_model(degree_sequence)
    G = nx.Graph(G)
    G.remove_edges_from(nx.selfloop_edges(G))

def compute_similarity(node1, node2, G):

    degree1 = G.degree(node1)
    degree2 = G.degree(node2)
    return 1 / (1 + abs(degree1 - degree2))


def popularity_similarity_model(N, p_degree, similarity_threshold=0.5):


    G = generate_network(N, p_degree)


    for node1 in range(N):
        for node2 in range(node1 + 1, N):

            similarity = compute_similarity(node1, node2, G)


            if similarity > similarity_threshold:
                G.add_edge(node1, node2)

    return G
```

```
N = 1000
p_degree = 10
similarity_threshold = 0.5


G = popularity_similarity_model(N, p_degree, similarity_threshold)


nx.draw(G, with_labels=False, node_size=10)
plt.title("Popularity Similarity Model Network")
plt.show()
```



hbt!

6.4 Results and Discussion

The implementation of the Popularity Similarity Model generates a network where edges are formed based on the degree similarity between nodes. The degree sequence is generated using a Poisson distribution, and the network is constructed using the configuration model. The similarity between two nodes is calculated as the inverse of the absolute difference in their degrees. Nodes with high degree similarity are more likely to form an edge.

The degree distribution of the generated network can be visualized as a histogram. The generated network exhibits properties influenced by the degree similarity between nodes. The threshold for the similarity determines how many edges are formed between nodes.

6.5 Conclusion

The Popularity Similarity Model provides a useful framework for constructing networks based on the degree similarity between nodes. By controlling the similarity threshold, we can generate networks with different structural properties, including assortative and disassortative behavior. This model can be used to simulate real-world networks where the connections between nodes depend on their degree similarity, such as in social networks, where similar popularity leads

to increased connectivity.