

Structural Differences Between Random Graphs and Small World, Scale-Free Graphs

Thomas Draycott

08/02/2023

Contents

1	Introduction	3
2	Literature Review	3
2.1	Graph Theory	3
2.2	Random Graphs	3
2.3	Small World Graphs	3
2.4	Scale Free Graphs	4
2.5	Barabasi-Albert Model	4
2.6	SIRD Model	4
3	Mathematical Methods	4
3.1	Graph Theory	4
3.2	Random Graphs	4
3.3	Small World Graphs	4
3.4	Scale-Free Graphs	5
4	Analysis	8
5	Evaluation	8
6	APPENDIX	8
6.1	Proofs	8
6.1.1	Graph Theory	8
6.1.2	Random Graphs	8
6.1.3	Small World Graphs	8
6.1.4	Scale-Free Graphs	9
6.2	Python	10
6.3	Networkx	10
6.4	Creating Graphs	11
6.5	Creating Random Graphs	11
6.6	Creating Barabasi-Albert Graphs	12
6.7	Gaining Information From Graphs	13

1 Introduction

This project seeks to investigate the structural differences between random graphs and small-world, scale-free graphs and in particular Albert-Barabási graphs. Then later we will use a SIRD model as a vehicle to explore whether these structural differences have an effect on random processes, while traditionally a project that contains an SIRD model would focus on the model as its core concept we will only use it as a simple lens to investigate the differences between graph structures therefore little time will be afforded to the mathematics behind the SIRD simulation, only how it computes on our graphs.

Random graphs were chosen as the comparison as by the fact they are random they are free of overarching structures that are present in other types of graph. The motivation for this project stems from my deep interest in graph theory and with the recent COVID-19 pandemic I wanted to explore whether how the way populations internally structured lead to a significant difference in the spread of disease.

2 Literature Review

2.1 Graph Theory

Graph Theory is the study of networks where vertices are connected by edges, see APPENDIX for a further explanation. note add the set theory def

2.2 Random Graphs

TO DO note difference between erod and gilbert models

2.3 Small World Graphs

In May 1967 Professor of Psychology at the Graduate School and University Center of the City University of New York, Stanley Milgram ran an experiment to see if a person living in Omaha, Nebraska could get a parcel to a stockbroker in Boston, Massachusetts (Milgram 1967). In his experiment he found the average path length to reach the stockbroker was 5.5, which created the term six degrees of separation (however Milgram's experiment had flaws which puts the exact number into doubt). This idea of having such a small average path length for such numerous nodes is a hallmark of a small world graph.

A Small world graph is formally defined by the following property: $L \propto \log N$ where L is the average shortest path length of the network and N is the total number of nodes (Watts and Strogatz 1998). Several models exist to generate small world graphs such as the Watts-Strogatz Model.

2.4 Scale Free Graphs

In networks that appear in the real world such as the internet and social groups, there exists nodes known as "hubs" (a node that has a significantly higher degree than the average of the graph). This is an important property encapsulated in Scale-Free Graphs.

Scale-free Graphs are formally defined by the following power law: $P(k) \sim k^{-\gamma}$, k is the degree of a vertex, $P(k)$ is the probability of a node having degree k and γ is a parameter determined by the graph typically $2 < \gamma < 3$ (Onnela et al. 2007).

2.5 Barabasi-Albert Model

In 1999, Albert-László Barabási and Réka Albert developed the Albert-Barabási Model which generates small world, scale free graphs by a process of preferential attachment (Barabási and Albert 1999). The model works as such: define two parameters n (The number nodes the graph at the end of the process will have) and e (the number of edges added for each new node) take a seed graph, add one new node to the graph, using preferential attachment add e edges from the new node to nodes on the seed graph, continue till there are n nodes on the graph.

Preferential attachment describes a 'rich get richer effect' that is the higher the degree of the node the more likely it will gain a new edge, the following formula describes it $\Pi(k_i) = \frac{k_i}{\sum_j k_j}$ where k_i is the degree of node i .

2.6 SIRD Model

SIRD stands for Susceptible, Infected, Recovered and Dead which is the states each individual in the model can take. There are many ways of implementing a model like this such as a virus having specific infection 'power' and mortality rates or having the infection be determined entirely by the individual.

3 Mathematical Methods

3.1 Graph Theory

TODO Chapter 1

3.2 Random Graphs

TODO Chapter 2

3.3 Small World Graphs

TODO Chapter 3

3.4 Scale-Free Graphs

Scale-Free Graphs follow a power law, that is $p_k \sim k^{-\gamma}$ where p_k is the probability of having degree k . We will now show the discrete formalism of the power law (Barabási 2013).

Proof.

Assuming: $k \in \mathbb{N}$

$p_k = Ck^{-\gamma}$ where C is a constant

C is determined by the normalization condition:

$$\sum_{k=1}^{\infty} p_k = 1$$

Using line 2 we obtain

$$C \sum_{k=1}^{\infty} k^{-\gamma} = 1$$

Thus

$$C = \frac{1}{\sum_{k=1}^{\infty} k^{-\gamma}} = \frac{1}{\zeta(\gamma)}$$

Where $\zeta(s)$ is the Riemann Zeta Function

$$\therefore p_k = \frac{k^{-\gamma}}{\zeta(\gamma)} \quad (4.1)$$

Therefore the average degree of a scale-free graph is:

$$\begin{aligned} \bar{k} &= \sum_{k=1}^{\infty} kp_k \\ \therefore \bar{k} &= \sum_{k=1}^{\infty} k \frac{k^{-\gamma}}{\zeta(\gamma)} \end{aligned} \quad (4.2)$$

We can also define a Continuum formalism for the power law distribution because for analytic calculations it is useful to let the degree k be any positive real

number as follows (Barabási 2013):

Assuming: $k \in \mathbb{R}^+$

$$p(k) = Ck^{-\gamma}$$

Using the normalization condition:

$$\begin{aligned} \int_{k_{\min}}^{\infty} p(k) dk &= 1 \\ \therefore C &= \frac{1}{\int_{k_{\min}}^{\infty} k^{-\gamma} dk} = (\gamma - 1)k_{\min}^{\gamma-1} \\ \therefore p(k) &= (\gamma - 1)k_{\min}^{\gamma-1}k^{-\gamma} \end{aligned} \tag{4.3}$$

Where k_{\min} is the smallest degree for which the power law holds.

Note only the integral of $p(k)$ holds any precise interpretation in the continuum formalism

□

We would now like to calculate k_{\max} this represents the expected size of the largest hub in the network.

We assume in a network of N nodes we expect at most one node in the (k_{\max}, ∞) range such that

$$\int_{k_{\max}}^{\infty} p(k) dk = \frac{1}{N}$$

Using result (4.3) we get the result:

$$k_{\max} = k_{\min} N^{\frac{1}{\gamma-1}} \tag{4.4}$$

See APPENDIX Proofs section Scale Free number 1

This means that k_{\max} has a polynomial dependence on N , so there can be a order of magnitude difference between k_{\min} and k_{\max} .

We can extend this to talk about the 'moments' of the degree distribution with the following formula (Papoulis and Unnikrishna Pillai 2002):

$$\langle f(x) \rangle = \sum f(x)P(x)$$

Applying this to degree distribution we gain the following, the n^{th} moment is defined as:

$$\langle k^n \rangle = \sum_{k_{\min}}^{\infty} k^n p_k \approx \int_{k_{\min}}^{\infty} k^n p(k) dk$$

Where k_{\min} is the smallest degree such that the distribution holds

Now the interpretation of these moments are important. $n = 1$ gives us the mean degree. $n = 2$ allows us to calculate the variance of the degree distribution via

the formula: $\sigma^2 = \langle k^2 \rangle - \langle k \rangle^2$. $n = 3$ gives us the skewness of the degree distribution. Now we apply this to our scale-free graph to gain insights to its moments.

$$\langle k^n \rangle = \int_{k_{\min}}^{k_{\max}} k^n p(k) dk = C \frac{k_{\max}^{n-\gamma+1} - k_{\min}^{n-\gamma+1}}{n - \gamma + 1}$$

Where k_{\max} is the largest hub in the network

A proof of the above integral will be in the APPENDIX Proofs section Scale Free number 2

Let us allow k_{\min} to stay fixed and to investigate the behavior of $\langle k^n \rangle$ we will take the limit as $k_{\max} \rightarrow \infty$ doing, so we see that the values of $\langle k^n \rangle$ are dictated by the $n - \gamma - 1$ term. If $n - \gamma - 1 \leq 0$ then $k_{\max}^{n-\gamma+1}$ will tend to 0 as k_{\max} increases, therefore all moments that satisfy $n \leq \gamma - 1$ are finite. Conversely, if $n - \gamma - 1 > 0$ then $\langle k^n \rangle$ goes to infinity as $k_{\max} \rightarrow \infty$ so for all moments larger than $\gamma - 1$ diverge.

For most scale-free graphs $2 < \gamma < 3$ and thus as $N \rightarrow \infty$ the first moment $\langle k \rangle$ is finite but $\langle k^2 \rangle$ and $\langle k^3 \rangle$ will go to infinity. Thus, we can see with the divergence of $\langle k^2 \rangle$ the variance (and standard deviation) can be arbitrarily large, if we picked a node at random we know very little about its degree. "Hence networks with $\gamma < 3$ do not have a meaningful internal scale but are scale free" (Barabási 2013). As an example to concrete this in his 1993 paper Barabási found the degree k of a randomly chosen node in the WWW was 4.6 ± 1546 which shows that despite $\langle k^2 \rangle$ only diverges in the $N \rightarrow \infty$ limit that the standard deviation $\sigma \gg \langle k \rangle$.

The average shortest path length $\langle d \rangle$ depends on the network size N and the degree exponent γ described in the following formula (Bollobás* and Riordan 2004)(Cohen and Havlin 2003):

$$\langle d \rangle \sim \begin{cases} \text{const.} & \gamma = 2 \\ \ln \ln N & 2 < \gamma < 3 \\ \frac{\ln N}{\ln \ln N} & \gamma = 3 \\ \ln N & \gamma > 3 \end{cases}$$

Let us discuss the implication of the behaviors described above (Barabási 2013): For γ the degree of the largest hub k_{\max} grows linearly with network size N . Which forces the network into a 'hub and spoke' configuration where all the nodes are close together because they all connect to a central hub. The average shortest path length does not depend on N . This is known as an Anomalous regime

For $2 < \gamma < 3$ the average shortest path length grows much slower than the $\ln N$ derived for random networks, this is referred to an ultra-small world regime (Cohen and Havlin 2003). The hubs massively reduce the average path length. For $\gamma = 3$ Here the 2nd moment of the degree distribution now longer diverges, we regain the $\ln N$ dependence from the random networks but the $\ln \ln N$ shrinks the average path length under that of a similarly sized random networks.

For $\gamma > 3$ At this point $\langle k^2 \rangle$ is finite and the average shortest path length follows the small world effect from random graphs. Hubs at this point are not sufficiently large to have an impact on the average shortest path length. (Note to expand on later large scale free graphs where $\gamma < 2$ do not exist as simple graphs, this may allow our assumption of $\gamma > 1$)

4 Analysis

TODO

5 Evaluation

TODO

6 APPENDIX

6.1 Proofs

6.1.1 Graph Theory

TODO

6.1.2 Random Graphs

TODO

6.1.3 Small World Graphs

TODO

6.1.4 Scale-Free Graphs

Proof 1: note is this assumption valid

$$\int_{k_{\max}}^{\infty} p(k) dk = \frac{1}{N}$$

$$\text{Result (4.3): } p(k) = (\gamma - 1)k_{\min}^{\gamma-1}k^{-\gamma}$$

Note: $(\gamma - 1)k_{\min}^{\gamma-1}$ is a constant which we will call C

Thus

$$C \int_{k_{\max}}^{\infty} k^{-\gamma} dk = \frac{1}{N}$$

$$C \left[\frac{1}{1-\gamma} k^{1-\gamma} \right]_{k_{\max}}^{\infty} = \frac{1}{N}$$

Assuming $\gamma > 1$

$$\lim_{k \rightarrow \infty} \frac{1}{1-\gamma} k^{1-\gamma} = 0$$

$$\therefore C \frac{k_{\max}^{1-\gamma}}{\gamma-1} = \frac{1}{N}$$

$$(\gamma - 1)k_{\min}^{\gamma-1} \frac{k_{\max}^{1-\gamma}}{\gamma-1} = \frac{1}{N}$$

$$k_{\max}^{1-\gamma} = \frac{1}{Nk_{\min}^{\gamma-1}}$$

$$k_{\max}^{\gamma-1} = Nk_{\min}^{\gamma-1}$$

$$k_{\max} = (Nk_{\min}^{\gamma-1})^{\frac{1}{\gamma-1}}$$

$$\therefore k_{\max} = k_{\min} N^{\frac{1}{\gamma-1}} \quad \square$$

Proof 2:

$$\langle k^n \rangle = \int_{k_{\min}}^{k_{\max}} k^n p(k) dk$$

$$\text{Result (4.3): } p(k) = (\gamma - 1) k_{\min}^{\gamma-1} k^{-\gamma}$$

$$\langle k^n \rangle = \int_{k_{\min}}^{k_{\max}} (\gamma - 1) k_{\min}^{-\gamma} k^n k^{-\gamma} dk$$

Again $(\gamma - 1) k_{\min}^{-\gamma}$ is a constant we will call C

$$\langle k^n \rangle = C \int_{k_{\min}}^{k_{\max}} k^n k^{-\gamma} dk$$

$$\langle k^n \rangle = C \int_{k_{\min}}^{k_{\max}} k^{n-\gamma} dk$$

$$\langle k^n \rangle = C \left[\frac{1}{n - \gamma + 1} k^{n-\gamma+1} \right]_{k_{\min}}^{k_{\max}}$$

$$\langle k^n \rangle = C \frac{k_{\max}^{n-\gamma+1} - k_{\min}^{n-\gamma+1}}{n - \gamma + 1}$$

6.2 Python

This project is written in the Python programming language as it is the language I am most familiar with and has very useful modules for graph analysis. All the code to create these graphs is completely doable without 3rd party libraries however as this project focuses on more complicated topics than just graph creation I will be using a 3rd party library to simplify creating graphs and manipulating them as detailed below.

6.3 Networkx

To begin we should explain the software that this project is based most heavily on. Networkx is a module for the Python programming language that allows for the creation and manipulation of graphs (Hagberg, Schult, and Swart 2008).

6.4 Creating Graphs

First we need to show how to create a graph using the software.

```
1 import networkx
2 import numpy
3 G = networkx.Graph()
4
5 G.add_node('A')
6 G.add_node('B')
7 G.add_node('C')
8 G.add_edge('A', 'B')
9 G.add_edge('C', 'B')
```

The above code does the following: it imports the Networkx package for us to use, creates a 'graph' object called G, creates nodes A, B, C, then adds two edges between A and B and B and C.

However that is too cumbersome for normal use so the Networkx package provides functions such as:

```
networkx.complete_graph(5)
```

to generate a complete graph with 5 nodes in one line, but we still have very fine control of the graph as in the first example.

6.5 Creating Random Graphs

To create a random graph on a piece of paper it is quite simple: Select N nodes to add to the graph and p probability, draw N nodes on the paper, then go through every possible pair of nodes in the graph and draw an edge between them if when picking a random real number from $[0,1]$ it is less than p . Networkx implements a function `networkx.gnp_random_graph` to create a random graph using the following code:

```

1 import networkx
2 import itertools
3 def gnp_random_graph(n, p, seed=None):
4     edges = itertools.combinations(range(n), 2)
5     G = networkx.Graph()
6     G.add_nodes_from(range(n))
7     if p <= 0:
8         return G
9     if p >= 1:
10        return networkx.complete_graph(n, create_using=G)
11
12    for e in edges:
13        if seed.random() < p:
14            G.add_edge(*e)
15    return G

```

Figure 1: The Random Graph Function

Let us explain, the function takes n (The total number of nodes) and p (The probability of drawing an edge between a pair of nodes) as parameters. The `itertools.combinations(range(n), 2)` creates a list of every combination of 2 numbers up to n i.e. (0,1), (0,2), (1,2) and so on, then to graph G we add n nodes, if $p = 0$ then there are 0 edges in the graph, so it stays empty and if $p = 1$ then every node is attached to every other node, so it becomes a complete graph, if neither of those are true then we loop through all the possible combinations of nodes and choose a random number from $[0, 1]$ and if that is less than p we add an edge between those nodes.

6.6 Creating Barabasi-Albert Graphs

First let us describe how to create Barabasi-Albert Graphs on a piece of paper. To create a Barabasi-Albert Graph we must first start with a "seed" graph (An already drawn graph that we will grow using the Barabasi-Albert Model) then we will define two more parameters n for the total number of nodes we want this new graph to have and m the number of edges we will add each iteration of the model. The algorithm works in the following way: take your "seed" graph and assign each node on the graph a probability p according to its degree k using the formula $p(k_i) = \frac{k_i}{\sum_j k_j}$ we then randomly select m unique nodes according to the probabilities calculated (this can be done by label each node 1 to n then write on a piece of paper that label for each node, then fold the paper a number of times proportional to its probability for example folding 0.2 twice but 0.2 five times then randomly choosing the papers), add 1 new node to the graph and draw an edge from this new node to each of the randomly preexisting nodes, repeat this until we have n nodes on the graph.

As Barabasi-Albert graphs are the main focus of this project we will now show how they are created in networkx.

```

615 def barabasi_albert_graph(n, m, seed=None, initial_graph=None):
616     if m < 1 or m >= n:
617         raise nx.NetworkXError(
618             f"Barabási-Albert network must have m >= 1 and m < n, m = {m}, n = {n}"
619         )
620
621     if initial_graph is None:
622         # Default initial graph : star graph on (m + 1) nodes
623         G = star_graph(m)
624     else:
625         if len(initial_graph) < m or len(initial_graph) > n:
626             raise nx.NetworkXError(
627                 f"Barabási-Albert initial graph needs between m={m} and n={n} nodes"
628             )
629         G = initial_graph.copy()
630
631     # List of existing nodes, with nodes repeated once for each adjacent edge
632     repeated_nodes = [n for n, d in G.degree() for _ in range(d)]
633     # Start adding the other n - m0 nodes.
634     source = len(G)
635     while source < n:
636         # Now choose m unique nodes from the existing nodes
637         # Pick uniformly from repeated_nodes (preferential attachment)
638         targets = _random_subset(repeated_nodes, m, seed)
639         # Add edges to m nodes from the source.
640         G.add_edges_from(zip([source] * m, targets))
641         # Add one node to the list for each new edge just created.
642         repeated_nodes.extend(targets)
643         # And the new node "source" has m edges to add to the list.
644         repeated_nodes.extend([source] * m)
645
646         source += 1
647     return G

```

Figure 2: The Barabasi-Albert Graph Function

This is the Barabási-Albert function from Networkx. It takes 4 parameters, n , m , `seed`, and `initial_graph`. n and m are simply the same as explained above, `seed` is for the random functions later so we can control the behavior, `initial_graph` is where we input our 'seed graph' for the model to build on (If no graph is provided then it uses a Star graph with $(m+1)$ nodes). The function then creates a list of nodes: `repeated_nodes` where each node is repeated equal to its degree (A node with degree 5 is repeated 5 times) then takes random samples from this list to create the preferential attachment then finally returns the Barabási-Albert graph.

6.7 Gaining Information From Graphs

Methods in networkx allow us to extract key information from the graph objects such as `G.number_of_edges` which returns the number of edges that are contained in graph G or `networkx.all_neighbors(G,n)` which returns all the neighbors of node n in graph G .

References

- Barabási, Albert-László (2013). “Network science”. In: *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 371.1987, p. 20120375.
- Barabási, Albert-László and Réka Albert (1999). “Emergence of scaling in random networks”. In: *science* 286.5439, pp. 509–512.
- Bollobás*, Béla and Oliver Riordan (2004). “The diameter of a scale-free random graph”. In: *Combinatorica* 24.1, pp. 5–34.
- Cohen, Reuven and Shlomo Havlin (2003). “Scale-free networks are ultrasmall”. In: *Physical review letters* 90.5, p. 058701.
- Hagberg, Aric A., Daniel A. Schult, and Pieter J. Swart (2008). “Exploring Network Structure, Dynamics, and Function using NetworkX”. In: *Proceedings of the 7th Python in Science Conference*. Ed. by Gaël Varoquaux, Travis Vaught, and Jarrod Millman. Pasadena, CA USA, pp. 11–15.
- Milgram, Stanley (1967). “The small world problem”. In: *Psychology today* 2.1, pp. 60–67.
- Onnela, J-P et al. (2007). “Structure and tie strengths in mobile communication networks”. In: *Proceedings of the national academy of sciences* 104.18, pp. 7332–7336.
- Papoulis, Athanasios and S Unnikrishna Pillai (2002). *Probability, random variables and stochastic processes*.
- Watts, Duncan J. and Steven H. Strogatz (June 1998). “Collective dynamics of ‘small-world’ networks”. In: *Nature* 393.6684, pp. 440–442. ISSN: 1476-4687. DOI: 10.1038/30918. URL: <https://doi.org/10.1038/30918>.