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

Thomas Draycott

08/02/2023

Abstract

This paper investigates the structural differences between random graphs and small world, scale-free graphs using the Barabási-Albert model as the example of the latter. Using the underlying mathematics of the graphs as well as a SIRD infection model to interrogate the differences between the two graphs. The analysis shows that for all attributes investigated there is a significant difference between the two types of graph relating to their structure. We conclude that there is a significant difference between the structure of the two types of graph and that and a SIRD model works sufficiently as a tool to investigate the structural differences of a pair of graphs.

Contents

1 Introduction

This project investigates the structural differences between random graphs and small-world, scale-free graphs and in particular, Barabási-Albert graphs.
Later, we will use an SIRD model as a vehicle to explore whether these structural differences can affect a random process. Traditionally a project that contains an SIRD model would focus on the model as its core concept. Alternatively 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, more time will be afforded to how it computes on our graphs. Random graphs were chosen as the comparison because 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. Additionally, with the recent COVID-19 pandemic I wanted to explore if it is how the way populations are internally structured which lead to a significant difference in the spread of disease.
Code and data used can be found at the following link:
https://www.dropbox.com/sh/ilb1vjvloguj33l/AAA2jwv5oi5SeFTtlSgjPn3Fa?dl=0

2 Literature Review

2.1 Graph Theory

Graph Theory is the study of networks where vertices are connected by edges (The words 'node' and 'vertex' will be used interchangeably as will 'network' and graph however graph will be used for more abstract representations). A simple graph contains no self loops and there can only be one edge between each pair of nodes. We will only be focusing on simple undirected graphs in this project and any graph mentioned can be assumed to be simple and undirected. A strict definition of simple graphs is as follows: Let G be a graph, G is an ordered pair $G = (V, E)$ comprising V: a set of vertices and $E \subseteq \{\{x, y\} | x, y \in V \text{ and } x \neq y\}$: a set of edges which are unordered pairs of vertices (Bender and Williamson 2010).

2.2 Random Graphs

In 1959, Paul Erdös and Alfréd Rényi published their paper 'On Random Graphs' (Erdos and Renyi 1959). Independently, Edgar Nelson Gilbert published his paper 'Random Graphs' (Gilbert 1959). Both papers describe how to create a graph using probabilities which became a very useful tool for solving problems in graph theory. In particular, problems in which they aimed to prove the existence of certain graphs held various properties or to provide a strong definition for what it meant for a property to hold for almost all graphs.
However, Erdös' and Gilbert's models differed in how they worked. The Erdös-Réyi model works as follows; to generate a random graph $G(N, M)$ where N is the number of nodes in the graph and M is the total number of edges to be

added, randomly pick M pairs of vertices from the set of all combinations of two vertices, ignoring repeats and draws edges between them. In this model the average degree is easily calculated. However, it is rare that the number of edges will be exact which brings on to the Gilbert model. For the Gilbert Model we define the random graph G as such; $G(N, p)$ where N is the total number of nodes and p is the probability of joining two vertices together and works as follows: draw N nodes for the graph, select a vertex pair, generate a random number between 0 and 1, if the number exceeds p draw mn edge between those two vertices, repeat for all vertex pairs once. For the rest of this paper we will use 'random graph' to refer to Gilbert model random graphs as they better represent real networks because the number of edges is not fixed.

2.3 Small World Graphs

In May 1967, the 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 send a parcel to a stockbroker in Boston, Massachusetts that they had never met personally by asking the subjects to give the parcel to someone they thought may know the recipient (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 a numerous amount of nodes the 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 1999, Albert-László Barabási and Réka Albert mapped a portion of the World Wide Web as a graph in doing so they discovered that some nodes had a far higher degree than others and that the network followed a power law. After further investigations of other networks the pair coined the term 'scale-free' to refer to these graphs. Why power laws appear in these contexts was a matter of dispute, with two main camps: The Preferential Attachment camp and the optimization camp. The preferential treatment camp headed in the 50s by Herbert Simon suggested that power laws appear due to a preferential attachment mechanism existing within these contexts (Simon 1955). The optimization camp during the same time was headed by Benoit Mandelbrot who argued that the answer was in random optimization (Mandelbrot et al. n.d.). However, using the ideas of Mandelbrot we were able to explain why preferential attachment occurs in these contexts, so both camps turned put to be correct.
In networks that appear in the real world such as the internet and social groups, there exist nodes known as "hubs" (nodes that have a significantly higher degrees

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

Continuing thier work on scale free graphs Albert-László Barabási and Réka Albert developed the Barabási-Albert 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 m (the number of edges added for each new node) take a seed graph (A graph we start with originally that the model will grow from), add one new node to the graph, using preferential attachment add m edges from the new node to nodes on the seed graph (With no loops or self loops), continue until there are N nodes on the graph.
'Preferential attachment' describes a 'rich get richer effect' that means the higher the degree of the node the more likely it will gain a new edge. This is described by the following formula $\prod(k_i) = \frac{k_i}{\sum_{\forall j} k_j}$ where k_i is the degree of node i.

2.6 SIRD Model

SIRD stands for Susceptible, Infected, Recovered and Dead which are the four states each individual in the model can take. All nodes begin Susceptible, then they can become Infected, once Infected they can infect their neighbours, eventually after a predetermined time they will either recover or die. 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. SIRD models vary massively in the assumptions they make, we will be assuming a discrete SIRD model with a constant infection rate, constant recovery rate and total immunity after a node has been infected once. The algorithm used for the SIRD model is in the dropbox link mentioned in the introduction.

3 Mathematical Methods

3.1 Graph Theory

3.1.1 Degrees

We will provide some key definitions and results from Graph Theory which will be useful for the further topics (Barabási 2013).
Degree: The number of edges attached to a node (Commonly written as k_i denoting the degree k of node i).

We can calculate the number of edges L of a graph from the sum of the degrees that is:

$$L = \frac{1}{2}\sum_{i=1}^{N} k_i$$

Average degree:

$$\langle k \rangle = \frac{1}{N}\sum_{i=1}^{N} k_i = \frac{2L}{N}$$

($\langle k \rangle$ and $\overline{k}$ will be used interchangeably, but $\langle k \rangle$ will be more common)
Degree distribution: The probability distribution of the degrees of the graph will be very relevant in the later sections, so we will define it here as p_k which is the probability a node has degree k.

As p_k is a probability we must normalize it with the following condition:

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

For a network with N nodes where N_k is the number of nodes with degree k

$$p_k = \frac{N_k}{N}$$

With this we can redefine $\langle k \rangle$ as:

$$\langle k \rangle = \sum_{k=0}^{\infty} k p_k$$

3.1.2 Adjacency Matrix

Graphs can also be represented by adjacency matrices. We define the matrix A, entry $A_{i,j} = 1$ if there is an edge between node i and node j else $A_{i,j} = 0$. For an undirected graph A is symmetrical. This can useful for quick computation that does not require any recursion.

3.1.3 $L_{\max}$

The total number of edges L for a graph is bounded by $L_{\max}$ and $L = 0$

$$L_{\max} = \frac{N(N-1)}{2}$$

We can see this by the most edges a simple graph can have is if it is a complete graph and complete graphs have $\frac{N(N-1)}{2}$ therefore a non-complete graph must have fewer edges.
Most real networks are sparse that is $L \ll L_{\max}$

3.1.4 Paths

A path is an ordered list of nodes describing a walk from each node to another. A Path length is the number of nodes contained in a path. The shortest path is the path between node i and node j with the smallest possible path length and is usually denoted as $d_{i,j}$ or just d. The diameter of a graph is defined as the longest shortest path of a graph and is denoted as $d_{\max}$.
The average shortest path length is defined as the average of the shortest paths between all pairs of nodes and is denoted as $\langle d \rangle$ and is calculated as such:

$$\langle d \rangle = \frac{1}{N(N-1)} \sum_{i,j=1,N|i \neq j} d_{i,j}$$

3.1.5 Clustering Coefficient

The local clustering coefficient, as defined by (Watts and Strogatz 1998), measures the extent to which a node's neighbors connect with one another. (Barabási 2013) explains that this coefficient is calculated for a node i based on its degree, denoted as k_i

$$C_i = \frac{2L_i}{k_i(k_i-1)}$$

Where L_i is the number of links between the k_i neighbours of node i. C_i is between 0 and 1

C_i represents the proportion that the neighbours of node i are connected
We can define the average clustering coefficient of a graph as such:

$$\langle C \rangle = \frac{1}{N} \sum_{i=1}^{N} C_i$$

$\langle C \rangle$ can also be interpreted as the probability two neighbours of a randomly selected node link to each other

3.1.6 Global Cluserting Coefficient

The Global Clustering Coefficient is another measure of a graph's clustering. It is defined as such:

Let $L_\triangle$ be the number of closed triangles in the graph
Let L_t the number of connected triplets

$$C_g = \frac{3L_\triangle}{L_t}$$

Note the average clustering coefficient and global clustering coefficient are not equal. Although it is a useful measure the literature focuses more on the average clustering coefficent, so that will be used for the rest of the project.

3.2 Random Graphs

For a random graph $G(N, p)$ we will define some key behaviours and values.

3.2.1 Number of Edges

We can begin by defining the probability of a random graph having exactly L edges: To begin, there are $\frac{N(N-1)}{2}$ pairs of nodes that can potentially be connected with edges. The probability that L out of these pairs have successfully formed an edge is p^L, while the probability that the remaining $\frac{N(N-1)}{2} - L$ pairs have not formed an edge is $(1-p)^{\frac{N(N-1)}{2}-L}$. Ultimately, there are $\binom{\frac{N(N-1)}{2}}{L}$ ways to connect L edges among the $\frac{N(N-1)}{2}$ pairs of nodes. Taking the product of these factors we gain:

$$p_L = \binom{\frac{N(N-1)}{2}}{L} p^L (1-p)^{\frac{N(N-1)}{2}-L}$$

Which is clearly a binomial distribution, therefore the mean number of edges $\langle L \rangle$ is given by np which for this example is equal to $p\frac{N(N-1)}{2}$. We also obtain the average degree of the random graph:

$$\langle k \rangle = \frac{2\langle L \rangle}{N} = p(N-1)$$

Note that $N-1$ is the maximum degree a node can have in this graph.

3.2.2 Degree distribution

Similarly to the case of edges above, we can construct a binomial distribution for the degrees of the random graph. The probability a randomly chosen node will have degree k will be a product of three terms defined as follows: The probability the node has k edges is p^k, the probability the remaining $N-1-k$ edges are missing is $(1-p)^{N-1-k}$. The number of ways a node can have degree k from degree $N-1$ is $\binom{N-1}{k}$ thus.

$$p_k = \binom{N-1}{k} p^k (1-p)^{N-1-k}$$

Using the properties of the binomial distribution we see that the average degree is $p(N-1)$ and the variance is $p(N-1)(1-p)$

3.2.3 The Poisson Approximation

Most real networks are sparse that is $\langle k \rangle \ll N$ in these cases the degree distribution is well approximated with:

$$p_k = e^{-\langle k \rangle} \frac{\langle k \rangle^k}{k!}$$

Using this approximation can be very useful for a number of reasons, firstly the variance and other key characteristics only depend on a single value, that being $\langle k \rangle$. It also suggests an interesting detail, because it does not rely on N, the degree distributions of graphs of varying sizes, but the small average degree are in fact identical to each other. It also makes further calculations easier. However, it must be firmly stated that the approximation only holds for sparse graphs and typically small graphs (those where N is less than 1000) are not sparse enough for this approximation to hold. This approximation will be used later but the requirements for it to hold must be remembered.

3.3 Small World Graphs

3.3.1 Diameter And Average Shortest Path

Considering a random network with an average degree $\langle k \rangle$, a node will have on average (Barabási 2013):
'$\langle k \rangle$ nodes at distance 1 ($d = 1$)
$\langle k \rangle^2$ nodes at distance 2 ($d = 2$)
$\langle k \rangle^3$ nodes at distance 3 ($d = 3$)
...
$\langle k \rangle^d$ nodes at distance d (d)'
Precisely the expected number of nodes up to distance d from our starting node is $N(d) = 1 + \langle k \rangle + \langle k \rangle^2 + \langle k \rangle^3 + ... + \langle k \rangle^d$ which is equal to $\frac{\langle k \rangle^{d+1}-1}{\langle k \rangle - 1}$. We have to set an upper bound on $N(d)$ as cannot be greater than N, so we set $N(d_{\max}) \approx N$. Assuming $\langle k \rangle \gg 1$ we can ignore the (-1) term in the numerator and denominator to obtain: $\langle k \rangle^{d_{\max}} \approx N$ therefore the diameter of the network follows $d_{\max} \approx \frac{\log N}{\log \langle k \rangle}$. Here we have the definition of the small world effect, the previous equation describes the scaling of the diameter however the average shortest path length $\langle d \rangle$ is better approximated by the previous equation as the diameter is easily increased by rare extremely long shortest path lengths. Generally $\log N \ll N$ which suggest the average shortest path length is orders of magnitude smaller than the size of the network which gives the definition of small: $\langle d \rangle \propto \log N$, with the $\frac{1}{\langle k \rangle}$ term correcting for denser networks having a smaller distance between nodes.

3.3.2 Clustering Coefficient

We define C_i as the local clustering coefficient of node i. To calculate the local clustering coefficient we need the expected number of edges E_i between the nodes k_i neighbours. In a random graph the probability that two of i's neighbours link to each other is p.
There are $\frac{k_i(k_i-1)}{2}$ possible edges between k_i neighbours of node i, the expected

value of E_i is:

$$\langle E_i \rangle = p\frac{k_i(k_i-1)}{2}$$

Therefore the local clustering coefficient of a random network is

$$C_i = \frac{2\langle E_i \rangle}{k_i(k_i-1)} = p = \frac{\langle k \rangle}{N}$$

From this we glean, for a fixed average degree, the larger the network the smaller a nodes clustering is and the local clustering coefficient of a node is independent of the nodes degree.

3.4 Scale-Free Graphs

3.4.1 Degree Distribution

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, page 7 of Network Science

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

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

$$\overline{k} = \sum_{k=1}^{\infty} kp_k$$

$$\therefore \overline{k} = \sum_{k=1}^{\infty} k\frac{k^{-\gamma}}{\zeta(\gamma)} \tag{4.2}$$

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). We can do this as 'the power law is a slowly varying function of k for large k' (Yang 2013):

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

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

Using the normalization condition:

$$\int_{k_{\min}}^{\infty} p(k)\,dk = 1$$

$$\therefore C = \frac{1}{\int_{k_{\min}}^{\infty} k^{-\gamma}\,dk} = (\gamma - 1)k_{\min}^{\gamma-1}$$

$$\therefore p(k) = (\gamma - 1)k_{\min}^{\gamma-1}k^{-\gamma} \tag{4.3}$$

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 $(k_{\max}, \infty)$ 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 an order of magnitude difference between $k_{\min}$ and $k_{\max}$.

3.4.2 Moments of the Distribution

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

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

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 behaviour of $\langle k^n \rangle$ we will take the limit as $k_{\max} \to \infty$ doing, so we see that the values of $\langle k^n \rangle$ are dictated by the $n - \gamma - 1$ term. If $n - \gamma - 1 \leqslant 0$ then $k_{\max}^{n-\gamma-1}$ will tend to 0 as $k_{\max}$ increases, therefore all moments that satisfy $n \leqslant \gamma - 1$ are finite. Conversely, if $n - \gamma - 1 > 0$ then $\langle k^n \rangle$ goes to infinity as $k_{\max} \to \infty$ so for all moments larger than $\gamma - 1$ diverge.
For most scale-free graphs $2 < \gamma < 3$ and thus as $N \to \infty$ the first moment $\langle k \rangle$ is finite but $\langle k^2 \rangle$ and $\langle k^3 \rangle$ will go to infinity (Yang 2013). 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 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 $N \to \infty$ limit that the standard deviation $\sigma \gg \langle k \rangle$.

3.4.3 Average Shortest Path Length

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 behaviours described above (Barabási 2013):
For $\gamma = 2$ the degree of the largest hub $k_{\max}$ grows linearly with network size N. 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.

3.5 Barabási-Albert model

3.5.1 Description of the model

We will describe the Barabási-Albert model inn further detail here including some important equations. First, let us define some key values. Let N be the number of desired nodes for the final graph to have, let m be the number of edges added each iteration of the model and let G be the initial graph the model will grow from. The Number of nodes in G will be denoted as G_N and this will be strictly less than N and the number of edges in G will be denoted as G_E(we will call G the 'seed graph' but note this is not widely accepted terminology). Also, m must be chosen such that $m < G_N$. First we take G and add one unconnected node, then we assign each node in G a probability using the formula below:

$$\prod(k_i) = \frac{k_i}{\sum_{\forall j} k_j}$$

Where k_h is the degree of node h

Using these probabilities we add m edges from our new unconnected node to m unique nodes. We continue this $N - G_N$ times thus we will add $m(N - G_N)$ edges

3.5.2 The Idea of Seed Graphs

There are many ways to construct a Barabási-Albert graph. One of which is to start with no nodes, then add one each iteration adding as many links less than m as you can. We however will be using a 'seed' graph to build the final Barabasi-Albert graph. The idea is to have a pre-existing graph and to apply the Barabási algorithm on that graph, this could be a random graph or a completely disconnected graph (n nodes but 0 edges) or a complete graph This allows us to artificially construct a basis for the algorithm and see how it grows from there.

3.5.3 Degree Distribution

The Barabási-Albert model produces scale free graphs. The degree distribution of it is: $p_k \sim k^{-3}$ (Barabási 2013). As we can see from the section on scale free graphs the Barabási-Albert model exists in the $2 < \gamma < 3$ range. Note that the degree distribution is independent of N and the number of iterations the model takes.

3.5.4 Diameter And Average Shortest Path Length

From (3.4.3) and because $\gamma = 3$ our diameter follows for $m > 1$ and large N:

$$\langle d \rangle \sim \frac{\ln N}{\ln \ln N}$$

The average shortest path length scales similarly.

3.5.5 Clustering Coefficient

The average clustering coefficient of the Barabási-Albert model follows (ibid.)(Klemm and Eguiluz 2002):

$$\langle C \rangle \sim \frac{(\log N)^2}{N}$$

3.6 SIRD Model

We will briefly describe what SIRD model we will be using and the parameters it takes. We will be using a discrete SIRD model assuming constant infection rate (p_i) and recovery rate (p_r). It will be discrete as we are looking at nodes being infected so the numbers of infected or recovered will always be positive integers. A constant infection rate means no matter the state of the model the probability of node i infecting its neighbour j will always be p_i. Our recovery rate works as follows after some defined number of time t the node has been infected it has a boolean decision, recover with probability p_r or die with probability $1 - p_r$. If the node recovers they are permanently immune to the infection and can not spread the infection. The algorithm works as follows: given a graph G we randomly select 1 node to be initially infected, then the infected node will attempt to infect its direct neighbours with probability (p_i) if successful those nodes will be become infected then we check if any node has been infected for t days and if they have been they will attempt to recover or die. This defines a single iteration of the algorithm, note that the algorithm only ends under two conditions all the nodes have died or there are no infected nodes left.

3.7 Two Sample t Tests

Two Sample t tests allow us to compare the means of two samples to find if they are significantly different. Normally the t test used is a Student's t test however

the assumption of equal variance required for that test will not be valid for our samples so instead we will use the Welch's t test (Welch 1947). This requires the following assumptions:
• Independence: Observations from one sample does not affcet the other.
• Normality: Both samples are approximately normally distributed.
• Random samples: Both samples are random samples.
The Independence and random sample assumptions follow from our methods that is, observations from a random graph do not affect those from a Barabási-Albert graph. The normality assumption will follow from the central limit theorem assuming a large enough sample size.
Here is an example of a two sample Welch's t test:

$H_0 : \mu_1 - \mu_2 = 0$
$H_1 : \mu_1 - \mu_2 \neq 0$
Our test statistic is:

$$t = \frac{\bar{x_1} - \bar{x_2}}{\sqrt{\frac{{s^2}_1}{n_1} + \frac{{s^2}_2}{n_2}}}$$

Where ${s^2}_i$ is the sample variance, $\bar{x_i}$ is the sample mean and n_i is the sample size
Our degrees of freedom are:

$$\frac{(\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2})^2}{(\frac{(\frac{s_1^2}{n_1})^2}{n_1 - 1} + \frac{(\frac{s_2^2}{n_2})^2}{n_2 - 1})}$$

From our significance level and degrees of freedom we gain a critical value that the t statistic must be greater than in order to reject H_0. We can also gain a p value from the test and to reject H_0 the p value must be less than our significance level.

4 Analysis

For our analysis we will now compare the properties of Random graphs and the properties of Barabási-Albert graphs. Firstly from a theoretical view using the equations and formulas we have defined prior. Then, we will look at data from random graphs and Barabási-Albert graphs created by a computer to see if our theoretical formulas match to real graphs. Finally, we will use our SIRD model to see if the difference in graph structure that we have shown so far has a significant effect on the output of the SIRD model. All the random graphs and Barabási-Albert graphs we look at will have the same number of nodes and edges so the only difference between them is the underlying structure of their creation.

4.1 Parameters Of Our Analysis

For this analysis we will be looking at random graphs of size N and probaility of joining two nodes 0.5 and Barabási-Albert graphs of size N adding 5 edges with each iteration using a random seed graph of size 10 and $p = 0.5$. We have chosen a p value for the random graphs by way of maximizing entropy (to do so we want to maximise the variances of the random graph which means maximising $Np(1-p)$ which yields $p = 0.5$). Using a random seed graph for the Barabási-Albert model gives us a starting off point for the model to grow from which is free of structural bias. We add 5 edges each time because it is less than the 10 nodes of the seed graph making sure that we do not end up creating a complete graph with each iteration.
We will be looking at different values of N which are
$[30, 50, 70, 90, 110, 130, 150, 170, 190, 200, 300, 400, 500, 1000, 1500, 2000, 2500]$
this will help show if the differences become more apparent as the size of the networks increases. After 2500 the computation time becomes too long for a lot of data to be collected. Sample sizes for the data will be 300 which is enough for the central limit theorem to apply so our t tests later will be valid, see appendix for normality tests to confirm this.

4.2 Theoretical difference

Firstly it can be noted that the average degree of the two graphs will be equal as $\langle k \rangle = \frac{2L}{N}$ with N and L being equal for both graphs then $\langle k \rangle$ will also be equal.

4.2.1 Highest degree

For a graph the largest and smallest degree on average should be bounded by $\langle k \rangle \pm \sigma$ where σ is the standard deviation. Using the Poisson approximation for random graph's degree distribution we find that the upper and lower bounds of the degree distribution would be $\langle k \rangle \pm \sqrt{\langle k \rangle}$. Because the Barabasi-Albert Model has $\gamma = 3$ its variance and by extension standard deviation can be arbitrarily large and certainly much larger than $\sqrt{\langle k \rangle}$, so it can be seen that on average Barabási-Albert graphs will have a larger highest degree to random graphs. In a sample of 300 Random Graphs paired with 300 Barabási-Albert graphs so that each has the same number of edges where $N = 2500$ we found the average highest degree for the random graphs as 22.57 with standard deviation of 1.43 while Barabási-Albert graphs had an average of 180.6 and a standard deviation of 24.4, using a two sampled t-test at a 5% significance level, resulted in a p value of 0 which is enough to suggest a significant difference in the average highest degree in the two graph types. Summarized data for the rest of the two sampled t tests are below.

4.2.2 Average Shortest Path Length

For both our graphs the average shortest path length $\langle d \rangle$ is proportional to $\ln N$ however for the Barabási-Albert model $\langle d \rangle \sim \frac{\ln N}{\ln \ln N}$ this means that $\langle d \rangle$ grows much slower for Barabási-Albert graphs, so they should on average have a smaller $\langle d \rangle$ than an equivalently sized random graph.

4.2.3 Average Clustering

The average clustering coefficient of a random graph is: $\langle C \rangle = \frac{\langle k \rangle}{N}$. For a Barabási-Albert graph $\langle C \rangle \sim \frac{(\ln N)^2}{N}$. We can see that on average Barabási-Albert graphs should have a higher clustering coefficient than random graphs.

4.3 SIRD Model Parameters

For the SIRD model we used the same types of graphs. The parameters for the SIRD model go as follows: An infection rate of 0.5, a recovery rate of 0.6,1 initial infected node and 0 initial immune nodes. From the SIRD model we get two important results '`Days_Taken`' (This is how long it took for the virus to either die out or kill all the nodes) and 'Survivors' (The number of nodes left after the infection has died out).

4.4 Summary Analysis

These are the p values from the 85 two sample t tests for each of the attributes above (Note any value less than 0.05 is considered statistically significant and any value of 0.0 is not necessarily equal to 0 it is just less than 0.00001.) All raw data can be found in the dropbox link in the introduction.

n	Highest Degree	Average Shortest Path	Average Clustering	Days_Taken	survivors
30	0.0	0.003469	0.0	0.377733	0.915393
50	0.0	0.000434	0.0	0.685641	0.860232
70	0.0	0.000046	0.0	0.010473	0.889225
90	0.0	0.000000	0.0	0.000000	0.938562
110	0.0	0.000000	0.0	0.000000	0.520753
130	0.0	0.0	0.0	0.000000	0.251608
150	0.0	0.0	0.0	0.000115	0.811857
170	0.0	0.0	0.0	0.000000	0.978347
190	0.0	0.0	0.0	0.000000	0.705481
200	0.0	0.0	0.0	0.000000	0.106506
300	0.0	0.0	0.0	0.000000	0.907223
400	0.0	0.0	0.0	0.000000	0.454334
500	0.0	0.0	0.0	0.000000	0.518512
1000	0.0	0.0	0.0	0.0	0.834069
1500	0.0	0.0	0.0	0.0	0.993874
2000	0.0	0.0	0.0	0.0	0.603536
2500	0.0	0.0	0.0	0.0	0.648783

From the table we can see that clearly there is a significant difference for all attributes of the graphs except for the number of survivors which is more likely to be dependent on the number of nodes in the graph rather than its underlying structure.

5 Evaluation

To evaluate, there are some criticisms of this project that should be addressed if it was to be taken further. Firstly, the project was written in the python programming language, while this meant extremely fast prototyping and a huge amount of flexibility in its construction its main draw back is that is exponentially slower than other languages that could have been used instead, if I was to undertake this project again I would likely learn a language like C++ to use as it fast much faster computation time then equivalent python code. Secondly the SIRD model used for our analysis is extremely naive and focuses solely on a single potential mechanism that is a constant rate infection, however in my code there does exist room for additional mechanisms to be added, so a further project could focus much more heavily on the SIRD model and investigate the different potential mechanisms of infection. And finally a further investigation into generic scale free or small world models, such as the Watts-Strogatz model, may have proved enlightening to see which attributes rely on which structure i.e. Average Clustering is more affected by the scale free property or the small world property. However, the SIRD model did prove effective in showing a difference between the two graphs and an interesting idea could be whether it is possible to predict what type of graph was used as input from the output of the SIRD model.

6 Conclusion

In this paper we set out to investigate the difference in the structures generated by the Gilbert random graph model and the Barabási-Albert models to get at the heart of what makes the random process of the Gilbert model and the preferential attachment of the Barabási-Albert model so different. In doing so we have looked at the mathematics behind these two models, found what their highest degree, average shortest path length and average clustering should theoretically be, compared and contrasted them and finally used an SIRD model as a novel way to investigate their structures. We conclude as such that there are significant structural differences between random and small world, scale-free graphs at the 5% significance level. A final interesting note with how effective the SIRD model was at showing a difference between the graphs is as follows: is it possible to take the output from the SIRD model and work out what type of graph was used an input?

7 APPENDIX

7.1 Problems With Milgram's Study

Firstly the participants were not chosen at random, they likely self selected themselves on whether they believed they would be able to complete a chain. Secondly Milgram asked participants to hand the parcel off to they thought was most likely to know the stockbroker which could have made the chains longer if the participant did not realize some they knew had a closer connection. And finally longer chains were underrepresented as the longer a chain was the more likely it was someone along it would give up on the experiment entirely thus not have that chain recorded.

7.2 Proofs

7.2.1 Scale-Free Graphs

Proof 1:

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

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

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

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

Assuming $\gamma > 1$

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

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

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

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

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

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

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

Proof 2:

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

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

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

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

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

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

7.3 Normality Tests

Here are Shaprio-Wilk p values for first random graphs then Barabási-Albert graphs:

n	Highest Degree	Average Shortest Path	Average Clustering	Days_Taken	survivors
30	0.000000	0.0	0.168563	0.0	0.000369
50	0.000022	0.0	0.004301	0.0	0.030823
70	0.001205	0.0	0.182920	0.0	0.021531
90	0.000000	0.0	0.617911	0.0	0.019819
110	0.000569	0.0	0.157740	0.0	0.108718
130	0.000021	0.0	0.069936	0.0	0.170925
150	0.000000	0.0	0.125660	0.0	0.107659
170	0.000000	0.0	0.029395	0.0	0.348010
190	0.002460	0.0	0.766267	0.0	0.353831
200	0.000000	0.0	0.003507	0.0	0.675189
300	0.000090	0.0	0.002691	0.0	0.000000
400	0.000218	0.0	0.215111	0.0	0.210440
500	0.000011	0.0	0.040442	0.0	0.010366
1000	0.000176	0.0	0.012026	0.0	0.333258
1500	0.000076	0.0	0.090985	0.0	0.615573
2000	0.000013	0.0	0.000370	0.0	0.134748
2500	0.000000	0.0	0.157920	0.0	0.813079

n	Highest Degree	Average Shortest Path	Average Clustering	Days_Taken	survivors
30	0.0	0.071009	0.812125	0.0	0.000051
50	0.0	0.861007	0.521119	0.0	0.001539
70	0.0	0.652746	0.986989	0.0	0.005444
90	0.0	0.271781	0.268814	0.0	0.124502
110	0.0	0.002702	0.128650	0.0	0.011746
130	0.0	0.105071	0.000100	0.0	0.088875
150	0.0	0.268874	0.818732	0.0	0.103702
170	0.0	0.297079	0.607007	0.0	0.071503
190	0.0	0.829072	0.021850	0.0	0.179250
200	0.0	0.020997	0.810162	0.0	0.217576
300	0.0	0.998746	0.361316	0.0	0.386509
400	0.0	0.669806	0.175246	0.0	0.093892
500	0.0	0.397257	0.026119	0.0	0.928841
1000	0.0	0.188685	0.087219	0.0	0.020617
1500	0.0	0.641171	0.041183	0.0	0.976792
2000	0.0	0.538786	0.784619	0.0	0.180491
2500	0.0	0.902277	0.445436	0.0	0.442948

Note just because the p values are may be greater than 0.05 that does not immediately suggest that the attributes are not normally distributed just that further analysis is needed.

7.4 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.

7.5 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).

7.6 Creating Graphs

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

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

7.7 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
4  def gnp_random_graph(n, p, seed=None):
5      edges = itertools.combinations (range(n), 2)
6      G = networkx.Graph()
7      G.add_nodes_from(range(n))
8      if p <= 0:
9          return G
10     if p >= 1:
11         return networkx.complete_graph (n, create using=G)
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 s 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.

7.8 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 pre-existing 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.

```python
import networkx as nx
def barabasi_albert_graph(n, m, seed=None, initial_graph=None):
    if m < 1 or m >= n:
        raise nx.NetworkXError(f"Barabási Albert network must have m >= 1 and m <n, m = {m}, n = {n}"
    if initial_graph is None:
    # Default initial graph: star graph on (m + 1) nodes
        G = star_graph (m)
    else:
        if len(initial_graph) < m or len(initial_graph) > n:
            raise nx.NetworkXError(
            f"Barabási Albert initial graph needs between m={m} and n={n} nodes"
            )
        G = initial_graph.copy()
        # List of existing nodes, with nodes repeated once for each adjacent edge
    repeated_nodes = [n for n, d in G.degree() for in range(d)]
    # Start adding the other n - me nodes.
    source = len (G)
    while source < n:
        # Now choose m unique nodes from the existing nodes
        # Pick uniformly from repeated_nodes (preferential attachment)
        targets _random_subset (repeated_nodes, m, seed)
        # Add edges to m nodes from the source.
        G.add_edges_from(zip ([source] * m, targets))
        # Add one node to the list for each new edge just created.
        repeated_nodes.extend (targets)
        # And the new node "source" has m edges to add to the list.
        repeated_nodes.extend([source] * m)
        source += 1
    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 behaviour,

`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.

7.9 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_neighbours(G,n)` which returns all the neighbours of node n in graph G.

7.10 SIRD Model Code

The code can also be found in the dropbox link.

7.11 Acknowledgements

I would like to thank the following people for their assistance in this project: Firstly my girlfriend Jessica for ensuring I stay focused throughout and helping me proofread as well as emotional support, then I would like to thank Vince Kwasnica for being extremely helpful as my project supervisor and personal tutor given guidance and reigning in my project when I would nrealy run it off the tracks. And Finally I would like to thank the developers of Python, Networkx, Pandas and Scipy without their software the project would not have run as smoothly as it has and the maintainers of LaTeXwhich proved extremely useful in the writing of this project.

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.

Bender, Edward A and S Gill Williamson (2010). *Lists, decisions and graphs.* S. Gill Williamson.

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.

Erdos, P and A Renyi (1959). "On random graphs". In: *Publ. Math. Debrecen* 6, pp. 290–297.

Gilbert, Edgar N (1959). "Random graphs". In: *The Annals of Mathematical Statistics* 30.4, pp. 1141–1144.

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.

Klemm, Konstantin and Victor M Eguiluz (2002). "Growing scale-free networks with small-world behavior". In: *Physical Review E* 65.5, p. 057102.

Mandelbrot, Benoit et al. (n.d.). "An informational theory of the statistical structure of language". In: ().

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*.

Simon, Herbert A (1955). "On a class of skew distribution functions". In: *Biometrika* 42.3/4, pp. 425–440.

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.

Welch, Bernard L (1947). "The generalization of 'STUDENT'S'problem when several different population varlances are involved". In: *Biometrika* 34.1-2, pp. 28–35.

Yang, Song (2013). *Networks: An Introduction by MEJ Newman: Oxford, UK: Oxford University Press. 720 pp., $85.00.*