Network Project
A Growing Network Model

CID: 01347129

16th March 2020

Abstract: Three models of a growing complex network were investigated for which analytic and numerical results were obtained. Firstly, a preferential attachment model was simulated, followed by a random attachment model and finally the random walk preferential attachment model. The largest degree in the preferential attachment model was found both analytically and numerically to scale with system size, via a power law. The theoretical power was 0.5, with the power numerically determined to be 0.51 ± 0.001. Finite scaling effects mean that these values are likely in agreement. The random walk preferential attachment model was found to have probability distributions corresponding to random and preferential attachment, depending on the value of probability q.

Word Count: 2414 words excluding font page, figure captions, table captions, acknowledgement and bibliography.

1 Introduction

This report aims to study the degree distributions produced when an attachment algorithm is followed for growing a network. One of the networks simulated is the Barabasi-Albert model, which uses preferential attachment. Two other models are a random attachment model and a model where a node is picked randomly then a random walk is done, which has a defined probability of ending on each step. The first two models will have their theoretical degree distribution analysed. Following this, the graph will be simulated using python and the degree distributions will be compared to the expected theory. The third model will be simulated and the results will be investigated to look for similarities to the other two models.

1.1 Definition

The general model that is used is defined:

1. Initial network $\mathcal{G}_0$ set up at time t_0.
2. Time goes from $t \to t+1$
3. New node added, with label $t+1$
4. m edges are added according to the rules:
   - One end connected to new node
   - Other end connected to an existing node chosen with probability Π, where Π is specified for each method
5. Repeat from step 2 until the final number N vertices are in the network.

2 Phase 1: Pure Preferential Attachment Π_{pa}

2.1 Implementation

2.1.1 Numerical Implementation

The code was structured using object-oriented programming. In Phase 1, the simulation was implemented using lists. As is outlined in section 2.2, the probability Π of connecting to an edge to a node is proportional to the degree of that node. Therefore the degree of each node, the time and the name of each node are the only variables that need to be stored. The list of edges between nodes isn't tracked. Note that all nodes are labelled by the time they were added, with the time label starting at $2m+2$, as there are $2m+1$ nodes initially in the system.

Initially, a degree list is created with $2m+1$ elements. The degree of each node is set to $2m$, representing a complete graph. Another list that is created is labelled as the 'preference list'. If the degree of a node is k_i, there will be k_i entries of node label i in the preference list. The aim of doing this is to randomly select from the preference list instead of choosing a node with a probability proportional to its degree as the number of times a node appears in the preference list is the same as the degree of that node.

To implement steps 2-4, a for loop is run for $N-(2m+1)$ iterations. Within this for loop, another for loop represents the steps in the bullet points, which has m iterations.

After each iteration of the first for loop, the preference list and degree list are updated, by appending m of the most recently added node to the preference list and appending a value m to the end of the degree list. The lists are designed such that the index of the degree list corresponds to the label each node.

2.1.2 Initial Graph

The initial graph is a complete graph with $2m + 1$ nodes. In section 2.2 the assumption is made that $E = mN$, where E is the number of edges and N the number of nodes. The number of edges in a complete network is given by:

$$E = \sum_{x=1}^{N-1} x.$$

This can be seen by counting edges in simple networks. For example in a network with q nodes the first node that's counted will have $q - 1$ edges, the next node will have $q - 2$ edges that haven't already been counted and so on. Consequently it can be shown by the sum of an arithmetic sequence that:

$$E = \sum_{x=1}^{N-1} x = \frac{N(N-1)}{2} = mN,$$

$$\frac{N-1}{2} = m,$$

$$N_0 = 2m + 1.$$

Therefore the initial number of nodes $N_0 = 2m + 1$ for the assumption to hold. This also guarantees that $k_i \geq m$ for all i as each new node, has a degree of at least m.

2.1.3 Type of Graph

The graph type is an undirected multigraph. The graph has no self-loops, as they can't be created using the algorithm in section 1.1 and are not present in the initial graph N_0. No direction is needed as the algorithm doesn't specify directions.

The graph does need to allow multiple edges however as there is a non-zero probability that multiple edges will be present between nodes. The probability does decrease over time as new nodes are added, meaning for very large N this should be rare.

The fact that self-loops are not allowed is representative of many real networks however, having multiple edges between two nodes is not. As the probability of this happening will decrease with large N, for large N the network should become more representative of a real network.

2.1.4 Working Code

To test that the code was working correctly, several approaches were taken. The first approach was to run each line of the implementation to ensure that line was doing what it was supposed to do.

Following this, the graph was drawn using the Networkx package in Python. Initially, this was done with $N = 5$ and $m = 1$. After this, all variables were inspected. The degree list was compared to the visual degree on the graph, the preference list and therefore the

probability was then compared to the probability list to check that $\Pi(i) \propto k_i$. After this was verified, the same process was repeated for $m = 2$ and $N = 6$. As this is an iterative model, if it works for a small number of iterations it should work for large iterations too.

2.1.5 Parameters

The parameters that the program needs are N, the number of nodes that should be added and m, the number of initial edges added per new node. In various tasks of the investigation, different N and m had to be chosen. There were some general aims of choosing the parameters though, as N and m were chosen such that $N \gg m$. The reason for this is that when $N \approx m$, finite scaling effects were more prevalent.

The other parameter that was used in the code was the number of repeats. To maximise precision, a high number of repeats should be chosen however, this would mean a very long time to run the code. The number of repeats is based on these two considerations for each test.

2.2 Preferential Attachment Degree Distribution Theory

2.2.1 Theoretical Derivation

The master equation is shown in Eq (1)

$$n(k, t+1) = n(k, t) + m\Pi(k-1, t)n(k-1, t) - m\Pi(k, t)n(k, t) + \delta_{k,t}. \tag{1}$$

In the preferential attachment model, $\Pi(k) \propto k$. Equation (2) then represents the condition for normalisation.

$$\Pi(k) = \frac{k}{\sum_{i=1}^{N} k_i}. \tag{2}$$

As $n(k, t)$ in Eq (1) represents the number of nodes with total degree k and every edge is connected to two nodes, meaning that $\sum_{i=1}^{N} k_i = 2E(t)$. This means that we can substitute $n(k, t) = N(t)p(k, t)$ and use the assumption that $E(t) = mN(t)$ to obtain Eq (3).

$$N(t+1)p(k, t+1) = N(t)p(k, t) + \frac{1}{2}(k-1)p(k-1, t) - \frac{1}{2}kN(t)p(k, t) + \delta_{k,t}. \tag{3}$$

We assume that $\lim_{x\to\infty} p(k, t) \to p_\infty(k)$, where $p_\infty(k)$ is a probability distribution that's constant in time. Using the relation $N(t+1) = N(t) + 1$, Eq (3) becomes Eq (4).

$$p_\infty(k) = \frac{1}{2}[(k-1)p_\infty(k-1) - kp_\infty(k)] + \delta_{k,t}, \tag{4}$$

Now the case where $k > m$ will be considered. In this case $\delta_{k,t} = 0$, so Eq (4) can be rearranged into Eq 5.

$$\frac{p_\infty(k)}{p_\infty(k-1)} = \frac{k-1}{2+k}. \tag{5}$$

Now consider the Gamma function $\Gamma(z)$:

$$\Gamma(n) = (n-1)!, \text{ For all integers n,}$$

By substitution, it can be shown that Eq (6a) has the solution (6b).

$$\frac{f(z)}{f(z-1)} = \frac{z+a}{z+b}, \tag{6a}$$

$$f(z) = A\frac{\Gamma(z+1+a)}{z+1+b}, \tag{6b}$$

$$\frac{f(z)}{f(z-1)} = \frac{(z+a)!}{(z+a-1)!}\frac{(z+b-1)!}{(z+b)!} = \frac{(z+a)}{(z+b)},$$

For $k > m$, Eq (5) can be solved by using the expression in Eq (6a), with $a = -1$ and $b = 2$. Therefore $p_\infty(k)$ can be found as is shown below, with the final result in Eq (7).

$$p_\infty(k) = A\frac{\Gamma(k)}{\Gamma(k+3)} = A\frac{(k-1)!}{(k+2)!,}$$

$$p_\infty(k) = \frac{A}{k(k+1)(k+2).} \tag{7}$$

To find the constant A in Eq (7), consider the case $k = m$. Eq (4) can be substituted into Eq (7) to solve for A. Also used is the fact that due all nodes having a degree $k \geq m$, $p_\infty(m-1) = 0$.

$$\frac{A}{m(m+1)(m+2)} = \frac{1}{2}[0 - m\frac{A}{m(m+1)(m+2)}] + 1,$$

$$A = 2m(m+1). \tag{8}$$

Finally, the complete equation for p_∞ is shown in Eq (9). For $k < m$ $p_\infty(k) = 0$.

$$p_\infty(k) = \frac{2m(m+1)}{k(k+1)(k+2)}, \text{ for } k \geq m \tag{9}$$

For finite N, there will be a scaling function $\mathcal{F}(x)$ that relates $p_N(k)$ to the limit where $N \to \infty$. This is shown in Eq (10).

$$p_N(k) = \mathcal{F}(\frac{k}{k_{max}})p_\infty(k). \tag{10}$$

2.2.2 Theoretical Checks

The theoretical distribution in Eq (9) should be normalised and also have a declining probability of nodes with high degrees. If the distribution is scale-free as expected when on a log-scale, p_∞ should be linear.

In order to check the normalisation, the sum to infinity of the distribution is taken over all the integers from m, as shown in Eq (11).

$$2m(m+1)\sum_{k=m}^{\infty}\frac{1}{k(k+1)(k+2)} = 1, \tag{11}$$

Using partial fractions Eq (11) becomes Eq (12).

$$2m(m+1)\sum_{k=m}^{\infty}(\frac{1}{2k} + \frac{1}{k+1} + \frac{1}{2(k+2)}) = 1, \tag{12}$$

The sum is over consecutive integers, meaning that many terms will cancel out. Consequently, Eq (12) becomes (13), which is true, hence $p_\infty(k)$ is normalised.

$$m(m+1)(\frac{1}{m} - \frac{1}{m+1}) = 1. \tag{13}$$

The second property, that the probability of getting nodes with a larger degree is smaller can be seen to be true from inspecting Eq (9). The numerator contains no variables, whereas the denominator contains terms with the highest order of k^3, with k being the only variable. Consequently, as the degree k grows, $p_\infty(k)$ will shrink, hence the criteria are fulfilled.

To check the scale-free nature of $p_\infty(k)$, take the logarithm. The denominator is approximated as its highest order term k^3.

$$\log p_\infty(k) \approx C - 3 \log k.$$

In the above equation, C is a constant value. So for large k, where k^3 dominates over the other terms $p_\infty(k)$ is scale-free as expected.

2.3 Preferential Attachment Degree Distribution Numerics

2.3.1 Fat-Tail

The distribution expected for the preferential attachment $p_N(k)$, is very small for large values of k, meaning that there will be a lot of statistical noise for large k, known as a fat tail. To deal with the noise, the data was log-binned with scale a. This means that each bin is 45% larger than the previous bin if $a = 1.45$, with bins starting at m and ending at the bin that includes the largest degree. The underlying distribution will be far more clear when log-binning is used.

2.3.2 Numerical Results

To obtain data that represented the actual distribution, the code was run 10 times for a system of $N = 4000$ for $m \in \{1, 2, 4, 8, 16, 32\}$. As outlined in section 2.3.1 the data was log-binned and plotted against k shown in the left of figure 1. If the measured values followed the probability distribution $p_\infty(k)$, all values will lie on the line $y = x$ on the right of figure 1. Instead, it can be seen the measured values seem to deviate more from $p_\infty(k)$ when $p_N(k)$ is small. This is also seen less clearly in the left figure, where the crosses seem to stray from the line for large k. For large values of $p_N(k)$ the distribution tends to agree with $p_\infty(k)$. This suggests that there is indeed a scaling function as shown in Eq (10). This also suggests that there is a cutoff degree $k_{max}(N)$, where scaling effects become important.

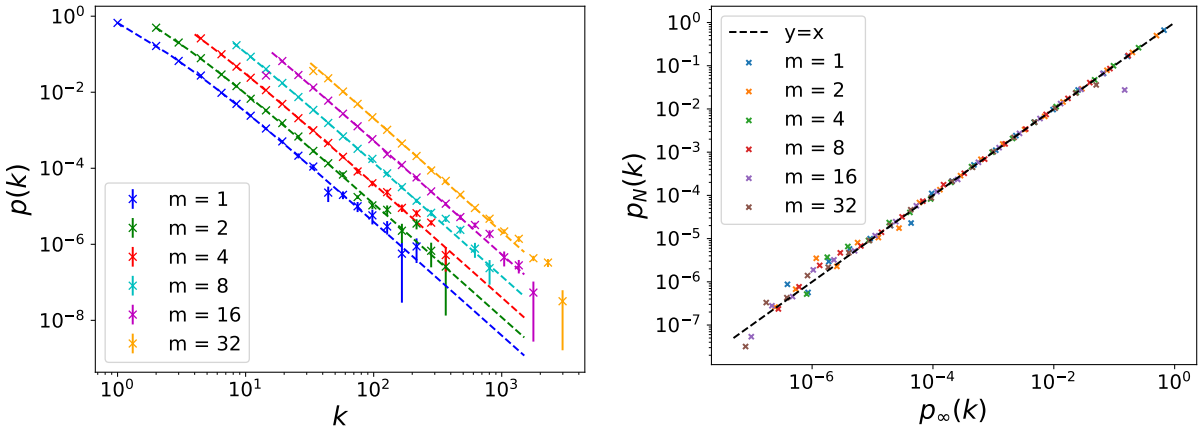

Figure 1: Log-binned degree distribution for $N = 4000$, $m \in \{1, 2, 4, 8, 16, 32\}$. In the left figure, the dotted lines show the theoretical distribution of $p_\infty(k)$, given by Eq (9). In the right figure values of $p_N(k)$ are plotted against the theoretical values $p_\infty(k)$.

m	χ^2	p value
1	2.52×10^{-4}	1
2	2.00×10^{-4}	1
4	3.96×10^{-4}	1
8	9.20×10^{-5}	1
16	9.90×10^{-2}	1
32	4.08×10^{-3}	1

Table 1: χ^2 test for preferential attachment.

2.3.3 Statistics

Because of the cutoff property of the data, it is difficult to use statistical techniques to compare the data to the theoretical distribution. One approach would be to fit the log-binned data up to a cutoff value using linear regression, then use the R^2 value to evaluate the fit. However it has been shown that applying linear regression to a log-log plot is invalid, so it can't be used to evaluate the goodness of fit [1].

A K-S test was considered on the continuous distribution function (CDF). This test requires that the raw data is continuous, which is approximately true. The fat tails create an issue however, as the CDF is far more accurate in the middle than the end, therefore the KS test wasn't used.

The test that was used was the χ^2 goodness of fit test, as χ^2 is designed for binned data. The results are shown in Table 1. All the p values are 1, meaning that the null hypothesis, which is that the data follows the distribution $p_\infty(k)$ can't be rejected.

2.4 Preferential Attachment Largest Degree and Data Collapse

2.4.1 Largest Degree Theory

. The largest expected degree k_1 is defined as the degree for which only one node of degree $k \geq k_1$ will be expected. Equation (14a) is the expected number for a graph with

N nodes. Methods from section 2.2.2, Eq (14b) is formed.

$$\sum_{k=k_1}^{\infty} p_\infty(k) = \sum_{k=k_1}^{\infty} \frac{2m(m+1)}{k(k+1)(k+2)} = \frac{1}{N}, \tag{14a}$$

$$\frac{m(m+1)}{k_1^2 + k_1} = \frac{1}{N}, \tag{14b}$$

By using the quadratic formula, k_1 is solved for in Eq (15), which for large N can be approximated as Eq (16). This shows that for large N, k_1 scales with $\sqrt{N}$.

$$k_1 = \frac{-1 + \sqrt{1 + 4Nm(m+1)}}{2}, \tag{15}$$

$$k_1 \approx \sqrt{mN(m+1)}. \tag{16}$$

2.4.2 Numerical Results for Largest Degree

The system was initialised and run 10 times for each unique value of N, with $m = 3$. The range of N was $N \in \{10, 10^2, 10^3, 10^4\}$. The value, $m = 3$ was chosen as $m \ll N$. The plot on figure 2 is on a log-scale on both axes. The reason for this is because as outlined in section 2.4.1, the largest degree scales with $N^{\frac{1}{2}}$. This is a power law, which consequently means that with a log-scale the line should be linear.

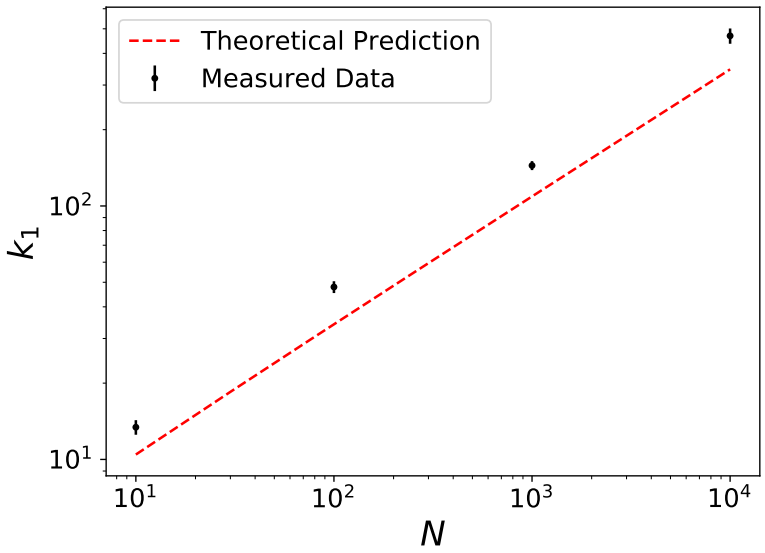

Figure 2: Largest degree versus theoretical k_1 for $m = 3$ and $N \in \{10, 100, 1000, 10000\}$. Error bars are shown however they are very small. The trend is clearly linear and falls just above the expected line for k_1, as would be expected.

The cutoff degree k_1 is below the largest degree as shown in figure 2. This behavior is expected as k_1 represents the value for which only one node in the network has a bigger degree. Consequently, k_1 is the lower bound of where the largest degree is expected to be. A line of best fit was done on the measured values for the largest degree, with a gradient of 0.51 ± 0.001. This value is close to the expected theoretical value of 0.5. It differs slightly because Eq (16) is an approximation of Eq (15).

2.4.3 Data Collapse

The data collapse aims to find the approximate form of $\mathcal{F}(x)$ from Eq (10). As was shown in section 2.4.2, the largest degree scales with $\sqrt{N}$. Therefore, using this fact and the form of Eq (10), the scaling function $\mathcal{F}$ can be found by plotting $\frac{p_N(k)}{p_\infty(k)}$ against $\frac{k}{k_1}$. This was done for $N \in \{10^2, 10^3, 10^4\}$ with $m = 3$, for 10 iterations. The results were then averaged to get figure 3.

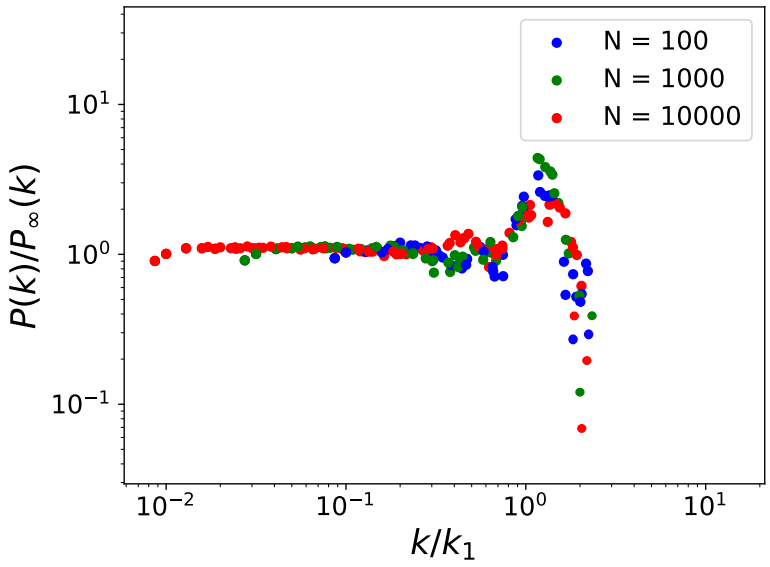

Figure 3: Data collapse of the log binned data for $m = 3$, $N \in \{10^2, 10^3, 10^4\}$, where $a = 1.3 - 1.5$. Figure created by plotting $\frac{p_N(k)}{p_\infty(k)}$ against $\frac{k}{k_1}$.

As is seen in the data collapse, the points do indeed collapse onto one function with some noise. $\mathcal{F}(x) \approx 1$ until $k \approx k_1$, where there is a bump. This bump is caused because the system is finite, as $k_{max} \leq N$. Therefore some nodes would've had larger degrees but don't due to the system being finite. A rapid drop-off is seen when $k > k_1$, as the probability of attaining nodes with these degrees is very small once again due to finite size.

3 Phase 2: Pure Random Attachment Π_{rnd}

3.1 Random Attachment Theoretical Derivations

3.1.1 Degree Distribution Theory

Random attachment states that there is an equal chance of selecting any existing node in step for of section 1.1, shown by Eq (17)

$$\Pi_{rand} = \frac{1}{N(t)}. \tag{17}$$

If this is substituted into the master equation shown in Eq (1), Eq (18) is formed in the limit where $t \to \infty$.

$$p_\infty(k) = mp_\infty(k-1) - mp_\infty(k) + \delta_{k,m}. \tag{18}$$

For $k < m$, $p_\infty(k) = 0$, as every node added initially forms m edges and there is no process for removing edges. Equation 19 shows the case where $k = m$.

$$p_\infty(m) = \frac{1}{1+m}, \tag{19}$$

Using Eq (19), the case where $k = m + 1$ can be calculated:

$$p_\infty(m+1) = (\frac{m}{m+1})p_\infty(m) = (\frac{m}{m+1})(\frac{1}{1+m}),$$

It is clear that this is an iterative process, so for an integer m, $p_\infty(m)$ is multiplied by a factor of $(\frac{m}{m+1})^{k-m}$ if $k \geq m$, shown in Eq (20).

$$p_\infty(k) = (\frac{1}{1+m})(\frac{m}{m+1})^{k-m}, \text{ for } k \geq m. \tag{20}$$

To check this is normalised, sum from $k = m$ to $k = \infty$:

$$1 = \frac{1}{m+1}\sum_{k=m}^{\infty}(\frac{m}{m+1})^{k-m},$$

This is a geometric sequence with a multiplier < 1 hence:

$$1 = \frac{m+1}{1-\frac{m}{m+1}}\frac{1}{m+1}.$$

The expression above is true, therefore it is normalised. Larger degrees are also less probable than smaller degrees in Eq (20), which is expected.

3.1.2 Largest Degree Theory

Using the same method as in section 2.4.1:

$$\sum_{k=k_1}^{\infty} p_\infty(k) = \frac{1}{N}, \tag{21}$$

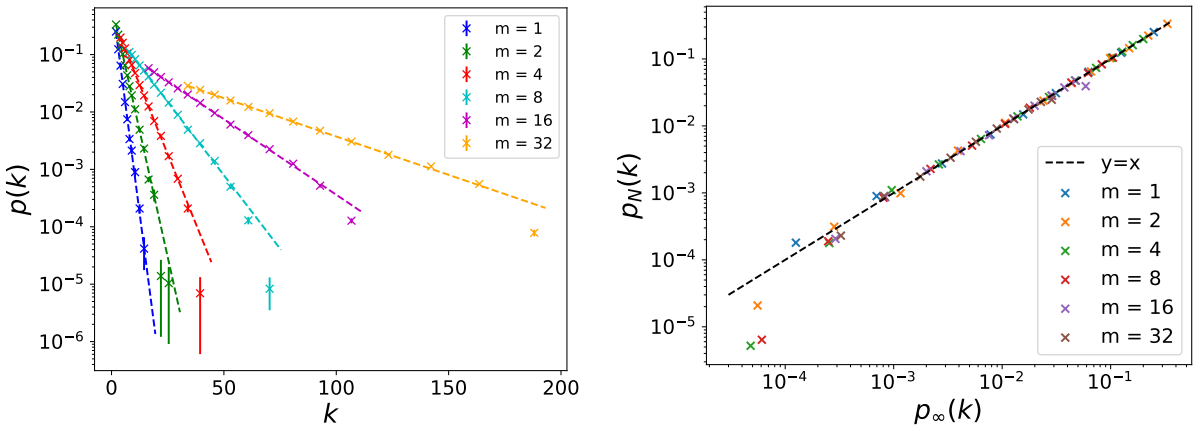

Figure 4: Log-binned degree distribution for $N = 4000$, $m \in \{1, 2, 4, 8, 16, 32\}$. In the left figure, the dotted lines show the theoretical distribution given by Eq (20). In the right figure, values of $p_N(k)$ are plotted against the theoretical values $p_\infty(k)$.

$$\sum_{k=k_1}^{\infty} (\frac{m}{m+1})^k = (\frac{m+1}{N})(\frac{m+1}{m})^m, \tag{22}$$

By changing the summation index to j:

$$\sum_{j=0}^{\infty} (\frac{m}{1+m})^{j+k_1} = (\frac{m}{1+m})^{k_1} \sum_{j=0}^{\infty} (\frac{m}{1+m})^j = \frac{m^m(1+m)^{1-m}}{N}, \tag{23}$$

Using the infinite sum of a geometric sequence, Eq (23) becomes Eq (24).

$$(\frac{m}{1+m})^{k_1} = \frac{m^m(1+m)^{1-m}}{N(1+m)}, \tag{24}$$

Eq (24) can now be rearranged for k_1, the cutoff degree by taking the logarithm of both sides, forming Eq (25). Note that Eq (25) is not a power law in N, so the network is not scale-free.

$$k_1 = m - \frac{\ln N}{\ln m - \ln m + 1}. \tag{25}$$

3.2 Random Attachment Numerical Results

3.2.1 Degree Distribution Numerical Results

The simulation was run for $m \in \{1, 2, 4, 8, 16, 32\}$ and $N = 4000$, with each configuration run 10 times and the log binned results averaged. The log-binned distributions were plotted against the theoretical distribution in Eq (20), as is shown in figure 4. The leftmost half of 4, $p(k)$ is plotted on a log-scale, as Eq (20) should appear linear in this format making deviations from $p_\infty(k)$ easier to identify.

Both half's of figure 4 shows that $p_\infty(k)$ is followed until a cut-off value, which is thought to be due to finite scaling effects.

A Chi-Square test is shown in table 2. Once again the p values were unanimously 1, meaning that the null hypothesis can't be rejected. The distribution for random attachment clearly doesn't lead to a fat tail. There also was no bump before the drop off for the range of m investigated, meaning it unfeasible to attempt a data collapse.

m	χ^2	p value
1	2.45×10^{-4}	1.00
2	4.99×10^{-1}	1.00
4	2.49×10^{-1}	1.00
8	1.32×10^{-1}	1.00
16	3.76×10^{-2}	1.00
32	1.69×10^{-3}	1.00

Table 2: Random Attachment χ^2 test.

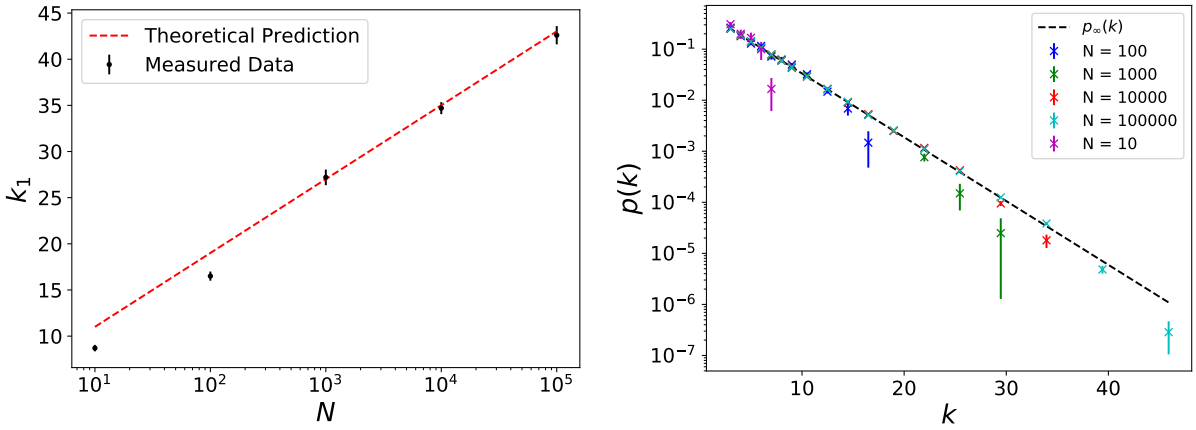

Figure 5: In the left figure, the largest degree on average for each value of N is shown versus the theoretical prediction. In the right figure, the degree distribution for each N is shown versus $p_\infty(k)$. The system has $m = 3$. On the right, it is clear that the systems follow the distribution until a cut-off point, where they deviate from $p_\infty(k)$. The distribution isn't scale-free, as the figure shows a power law isn't followed.

3.2.2 Largest Degree Numerical Results

The simulation was run 10 times for $N \in 10^1, 10^2, 10^3, 10^4, 10^5$, $m = 3$. The mean largest degrees and errors, as well as the theoretical k_1, are shown in figure 5 on the left. On the right, the degree distribution is shown.

The scaling of the largest degree k_1 appears to differ from the theoretical distribution, being closer to the theory for higher N. It may be the case for $N \gg 10^5$, due to finite scaling, the theory is correct, but due to computational restrictions this wasn't tested.

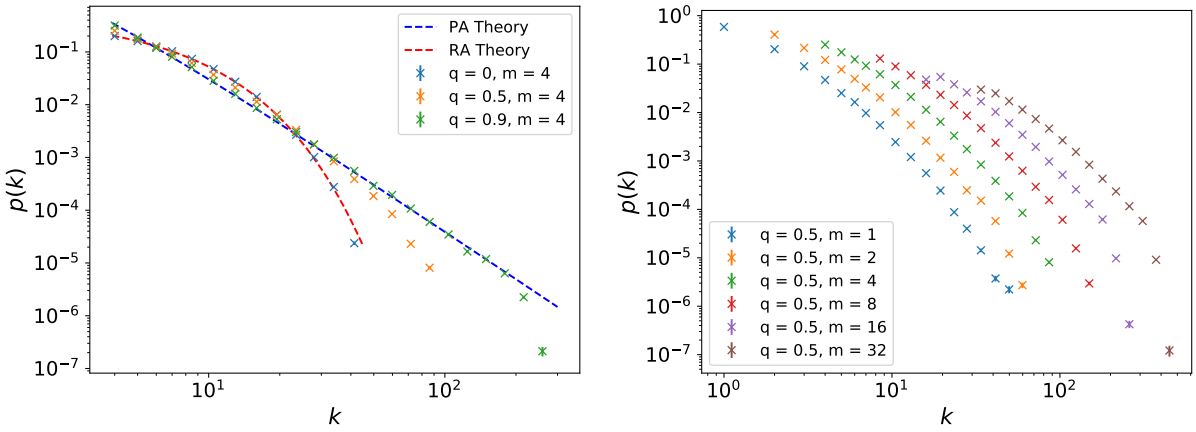

Figure 6: In the left figure, the model is shown for $m = 4$, $q \in \{0, 0.5, 0.9\}$. The random attachment distribution from Eq (20) is plotted, as well as the preferential attachment distribution in Eq 9. It can be seen that for $q = 0$, $p(k)$ follows Eq (20), whereas for $q = 0.9$ Eq (9) is approximately followed. In the right figure, the model is shown for $q = 0.5$, $m \in \{1, 2, 4, 8, 16, 32\}$. Error bars are shown however they are very small.

4 Phase 3: Random Walks and Preferential Attachment

4.1 Implementation

The following steps are repeated:

1. An existing node v_i is picked at random from all the existing nodes.
2. Step 4 is skipped to with probability $1 - q$.
3. Else, choose a neighboring node of v_i at random. This is v_{i+1}. Now go to step 2 again, where v_i now becomes v_{i+1}.
4. Walk stops at the current vertex, $v_i = v_{final}$. This is the target for a new edge between a new node and v_{final}.

The networkx package was used to create a graph and find the nearest neighbors. The graph was simulated for $N = 4000$, $m \in \{1, 2, 4, 8, 16, 32\}$ and $q =\in \{0, 0.5, 0.9\}$. Each configuration was repeated 25 times, then the log-binned results were averaged.

4.2 Numerical results

Figure 6 shows the probability distribution for the random walk and preferential attachment model.

Following the visual inspection of the data, χ^2 tests were applied as shown in tables 3 and 4.

4.3 Discussion of Results

The results show that the model can approximate both the Random Attachment model and the Preferential attachment model, depending on the value of q. This is shown as

m	χ^2	p value
1	6.90×10^{-5}	1.00
4	4.80×10^{-5}	1.00
32	6.97×10^{-4}	1.00

Table 3: χ^2 test for data at $q = 0.0$ against the random attachment model.

m	χ^2	p value
1	1.21×10^{-4}	1.00
4	1.77×10^{-3}	1.00
32	3.12×10^{-3}	1.00

Table 4: χ^2 test for data at $q = 0.9$ against the preferential attachment model.

the χ^2 test gives p values of 1 unanimously, meaning that the null hypothesis can't be rejected. The time that is spent on the random walk is defined by q, as the lower the q the less time spent walking. A fat tail emerges as $q \to 1$, as if a node has many connections, the random walk is more likely to end on that node. However, a model where $q = 1$ can't be simulated, as this would mean an ∞ time would be spent on each walk, as it would never end.

With $q \approx 1$, the emergence of a fat tail emulates what is seen in many real-world networks, where popular nodes tend to gain more popularity and unpopular nodes are far more plenty and less popular. Examples can be seen in the sales of books, the distribution of wealth and financial markets returns [2], [3], [4].

5 Conclusions

Models for preferential and random attachment were investigated both analytically and numerically. The long time distribution $p_\infty(k)$ was compared to distributions created with finite N. Where possible, data collapses were produced, as the cutoff for a finite-size network was investigated. Further investigations would focus on simulating $N \gg 100000$ nodes to investigate further the theoretical distribution for random attachment.

References

[1] A. Clauset, C. R. Shalizi, and M. E. Newman, "Power-law distributions in empirical data," *SIAM review*, vol. 51, no. 4, pp. 661–703, 2009.

[2] X. Gabaix, P. Gopikrishnan, V. Plerou, and H. E. Stanley, "A theory of power-law distributions in financial market fluctuations," *Nature*, vol. 423, no. 6937, pp. 267–270, 2003.

[3] F. Deschatres and D. Sornette, "Dynamics of book sales: Endogenous versus exogenous shocks in complex networks," *Physical Review E*, vol. 72, no. 1, p. 016112, 2005.

[4] M. Patriarca, A. Chakraborti, K. Kaski, and G. Germano, "Kinetic theory models for the distribution of wealth: Power law from overlap of exponentials," in *Econophysics of wealth distributions*. Springer, 2005, pp. 93–110.

Acknowledgements

I would like to thank Tim Evans for delivering the Networks course and Henry Price for his help during the computing sessions.