ECE 232E Spring 2023

Random Graphs and Random Walks

Generating Random Networks

Question 1a

For this question, the task was to generate several undirected Erdos-Renyi graphs with varying probabilities ranging from $2x10^{-3}$ to 10^{-1}, and plot their degree distribution. In an Erdos-Renyi graph, a vertex *i* is randomly connected to another node *j* with probability *p*, or otherwise it can be thought that the edge between node *i* and *j* is cut with probability *1-p*.

The distribution that an undirected Erdos-Renyi graph follows is that of a Binomial distribution which as n goes towards infinity changes from a Binomial distribution to a Laplacian distribution. The degree of a $vertex_i$ is measured as the number of vertices with which there is an edge connecting $vertex_i$ to other vertices in the graph.

For a $vertex_i$ to have degree k, there are combinatorically $\binom{n-1}{k}$ possibilities. There are *n-1* nodes and not *n* nodes since we do not consider self-loops currently, thus we have to deduct 1 node from the calculation. And as we've mentioned before, each node can connect to another node with probability *p,* and not connected to another node with probability *1-p*. Piecing these together we get the following equation (1). The theoretical mean and variance of a binomial distribution are shown in equations (2) and (3), respectively.

$$P(degree(v_i == k)) = \binom{n-1}{k} p^k (1-p)^{n-1-k} \quad (1)$$

$$E[P(degree(v_i == k))] = np \quad (2)$$

$$Var(P(degree(v_i == k))) == np(1-p) \quad (3)$$

As *n* progresses towards infinity, the distribution shifts towards a *poisson* distribution which is given by equation (4).

$$P(degree(v_i == k)) = \frac{e^{-np}(np)^k}{k!}$$

The graphs depicting the degree distribution for various realization of the Erdos Renyi graph are depicted in Fig.1-5 respectively. As can be seen from all the figures, as *p* increases from $2x10^{-3}$ to 10^{-2}

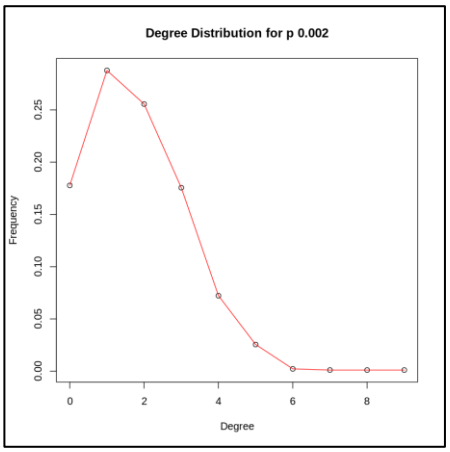

Fig. 1 Degree Distribution for $p=2x10^{-3}$

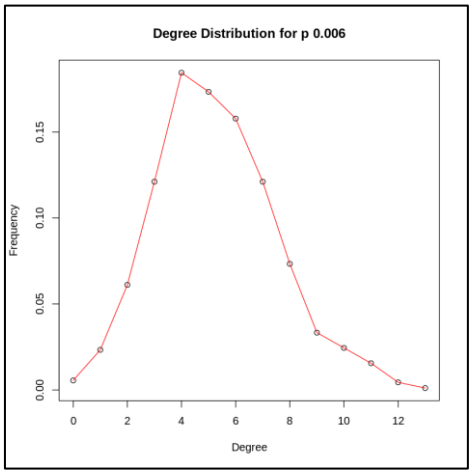

Fig. 2 Degree Distribution for $p=6x10^{-3}$

the graph starts from having a *Poisson* distribution and moves towards a *Binomial* distribution. While we have mentioned that as n progresses towards infinity, the distribution starts to behave more like a Poisson distribution. The exact conditions are when $np \ll n$ and this is the case if we look at Table 1 which has the theoretical and actual values achieved for the mean and variance of the degree distribution of the Erdos Renyi graph.

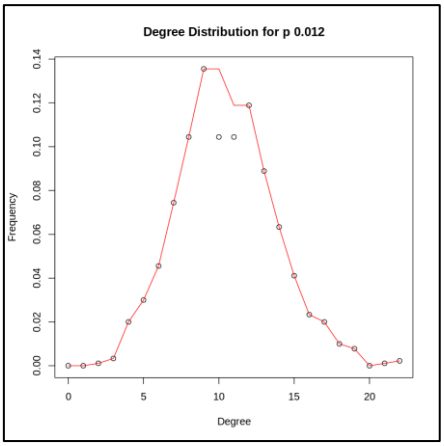

Fig. 3 Degree Distribution for $p=1.2x10^{-2}$

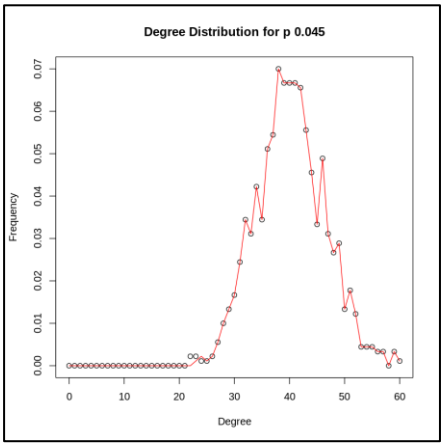

Fig. 4 Degree Distribution for $p=4.5x10^{-2}$

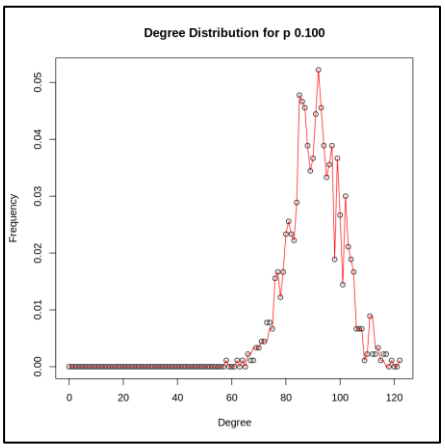

Fig. 5 Degree Distribution for $p=10^{-1}$

Table. 1 Theoretical and Actual values of Mean and Variance for Erdos-Renyi Graphs

Probability Value	Theoretical Mean	Actual Mean	Theoretical Variance	Actual Variance
0.002	1.8	1.782	1.7964	1.806
0.006	5.4	5.278	5.3676	4.955
0.012	10.8	10.484	10.6704	10.399
0.045	40.5	40.1093	38.6775	38.795
0.1	90	90.860	81	86.619

Question 1b

For this question, the task was to sweep p again from $2x10^{-3}$ to 10^{-1} and numerically estimate the probability that the Erdos-Renyi graph is connected, furthermore for a single instance of an unconnected graph, to find the diameter of its GCC (Giant Connected Component), which is only defined for graphs that are not connected as the largest set of vertices that are connected or in other words within this group there exists a path from each vertex to another.

For an undirected graph, the probability that the graph is connected increases non-linearly with the probability p as can be seen in Table 2. In between $6x10^{-3}$ and $1.2x10^{-2}$, the probability value p doubles, but the probability that the graph is connected increases from 4% to 99%. Secondly, as p increases, the diameter of its GCC decreases since larger and larger parts of the graph are now connected to each other, the largest cluster that is disconnected from the rest of the graph gets smaller and smaller until the graph becomes fully connected upon which the GCC no longer exists since the GCC is the Erdos Renyi graph.

In regards to whether all random realizations of the graph are connected, the probability of the realization depends on the probability value p as well as the size of the network n. When $np \ll n$, the probability of the graph being connected is low and close to 0%, as p increases, when np is in the same magnitude as n, the probability of generating a fully-connected graph increases non-monotonically and at the same time the diameter of its GCC decreases non-monotonically as well.

Table 2 – Probability of an ER graph being connected and the diameter of its GCC

Probability	All Realizations are connected	Probability of getting a connected graph	Diameter of its GCC
$2x10^{-3}$	No	0%	25
$6x10^{-3}$	No	4%	9
$1.2x10^{-2}$	No	99%	5
$4.5x10^{-2}$	Yes	100%	-
10^{-2}	Yes	100%	-

Question 1c

For this question the probability value p was swept from 0 to 10^{-2}, in steps of 10^{-4}. The limit of 10^{-2} was used since when n is 900, the value of $\frac{\ln n}{n} = 7.56x10^{-3}$, therefore it was rounded up to 10^{-2} and also from the last question, we did see that by p = 10^{-2}, the graph is always connected. For each value of p, a Erdos-Renyi graph was generated 100 times and the expected value of the normalize GCC size was

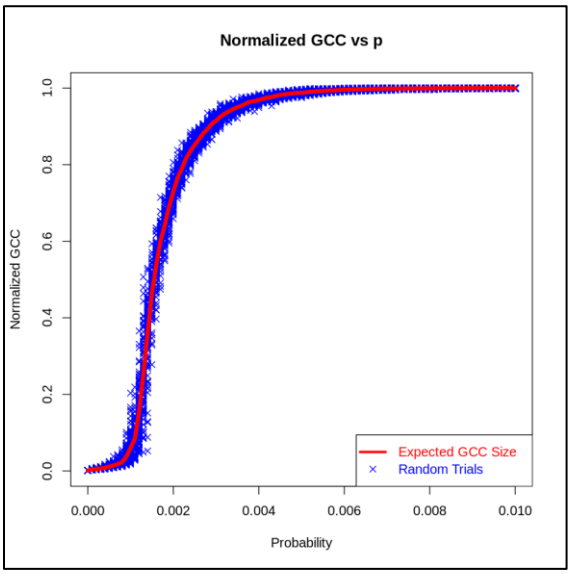

Fig. 6 Normalized GCC vs p for p ranging from 0 to 10^{-2}

derived from these 100 trials and is visualised in Fig.6. Furthermore, we can see from Fig.6 that as p increases, the size of the GCC increases non-linearly with p. The theoretical and actual values achieved for both inflection points are shown in Table 3.

- The first inflection point happens at $p = \frac{1}{n}$ and for $n = 900$, the value of p comes to $1.1x10^{-3}$ and from Cayley's theorem the expected size of the GCC is given by equation(4).

$$E(|GCC|) = \epsilon \cdot n^2 \cdot p \quad (4)$$

  The criterion of emergency was set at when the size of the GCC covers more than 10% of the total graph, and for this value ϵ is $1.1x10^{-4}$. The theoretical value at which the GCC starts to emerge is $p = 1.1x10^{-3}$ and the actual value at which it starts to emerge is at $p = 1.2x10^{-3}$.
- The second inflection point at which takes up over 99% happens at $p = c\frac{\ln n}{n}$, for a value of c set to 1.00001, the theoretical probability value comes to $7.56x10^{-3}$ and the actual realised value is $7.6x10^{-3}$

Table 3 – Comparison of Actual and Theoretical Values for points of inflection

Inflection Point	Theoretical Probability Value	Actual Probability Value
Emergence of GCC	$1.1x10^{-3}$	$1.2x10^{-3}$
GCC takes up more than 99% of the graph	$7.56x10^{-3}$	$7.6x10^{-3}$

Question 1d

In this question, the task was to sweep n from 10^2 to 10^4 and the probability p should follow the equation (5), and c is varied to control how quickly p changes with respect to $\frac{1}{n}$. Similar to the previous question, for each combination of n and p, a total of 100 trials were done and used to calculate the expected size of GCC.

$$c = np \tag{5}$$

1. For $c = 0.5$, the results and trend present can be seen in Fig. 7, and similar to what we've seen so far. As n increases, the GCC size increases piece-wise linearly. The inflection point at which the slope changes is around $n = 2000$ beyond which the slope at which the GCC size increases with respect to n, decreases.

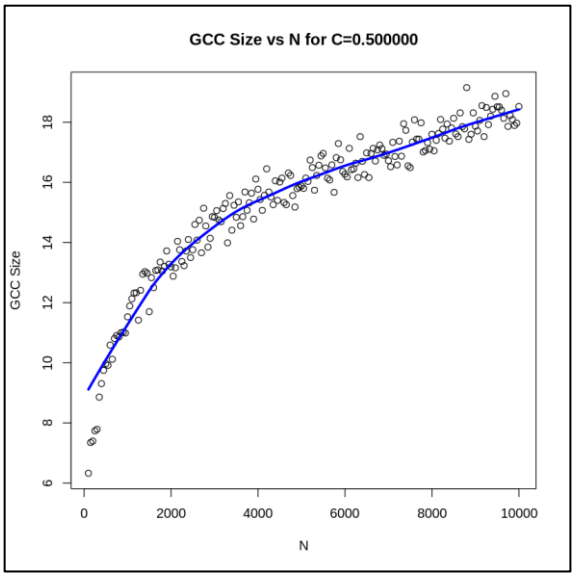

Fig. 7 Expected GCC size vs N when c is set to 0.5

For c<1, the number of connected components is low for an undirected Erdos-Renyi graph and is the size of the GCC is upper bounded by $O(ln(n))$, what that means is that, the number of vertices can be no more than $O(ln(n))$ when c<1.

2. For c =1, the results and trends present are depicted in Fig. 8 and are slightly different from the results we've seen when c = 0.5 , the expected GCC size increases logarithmically with an increase in the n and the second thing that can be seen that the magnitude of the expected GCC size increases much more than it did when $c=0.5$, this is because the rate at which the size of the GCC grows for an undirected Erdos Renyi graph when c is set to 1 is given by equation (6). Thus, from equation (6), we can see that the GCC size does increase with an increase in n, albeit linearly; it does grow much faster and does not have an upper bound like for when c<1.

$$E(|GCC|) = O(\sqrt{n}) \vee O(n^{\frac{2}{3}}) \tag{6}$$

3. For the last experiment, the goal was to calculate the expected size of the GCC for values of $c>1$. Three cases were considered when $c = 1.15, 1.25$ and 1.35. The results from this are shown in Fig. 9 and what we see is that it grows linearly with respect to n, even more so than when $c=1$, and as c increases, slope of the expected GCC line increases, which means that it is on the $O(n)$, instead of $O(\sqrt{n})$ *or* $O(n^{\frac{2}{3}})$. The actual relationship between the expected GCC size and n is given in equation (7).

$$E(|GCC|) = \mathrm{O(n)} == \epsilon \cdot c \cdot n \tag{7}$$

4. The expected size of the GCC and n for each case when *c<1, c=1 and c>1* is tabulated below in Table 4.

Table 4 – Expected Size of GCC vs C

Case for C	$E(\|GCC\|)$
c<1	$\epsilon \cdot c \cdot \ln n$
c=1	$\epsilon \cdot c \cdot \sqrt{n}$ *or* $\epsilon \cdot c \cdot n^{\frac{2}{3}}$
c>1	$\epsilon \cdot c \cdot n$

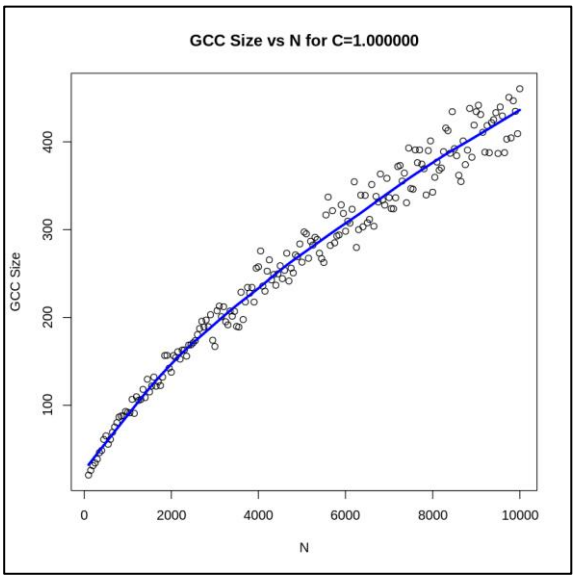

Fig. 8 Expected GCC size vs N when c is set to 1

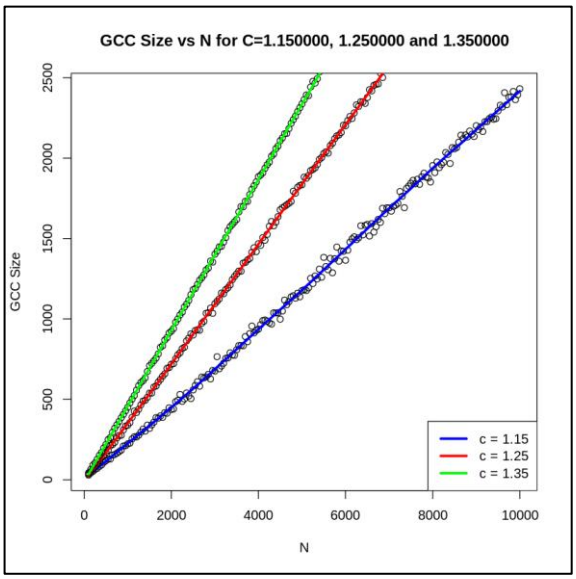

Fig. 9 Expected GCC size vs N when c is set to 1.15,1.25 and 1.35

Question 2a

In this experiment, an undirected preferential attachment network was generated with parameters *n=1050* and *m=1*. In a preferential attachment network, each new vertex that is added to the graph connects to *m* other vertices wherein the probability of being connected is directly proportional to the degree of the vertex under question, therefore high degree nodes are more likely to get new edges connecting it to other vertices and this is a positive feedback cycle that continues, and the knowledge of the degree distribution of the nodes can usually be inferred from taking a random walk with a probabilistically significant number of steps to ensure that the random walker has seen enough nodes before making a decision. The graph generated for *n =1050* and *m=1* is visualised in Figure 10.

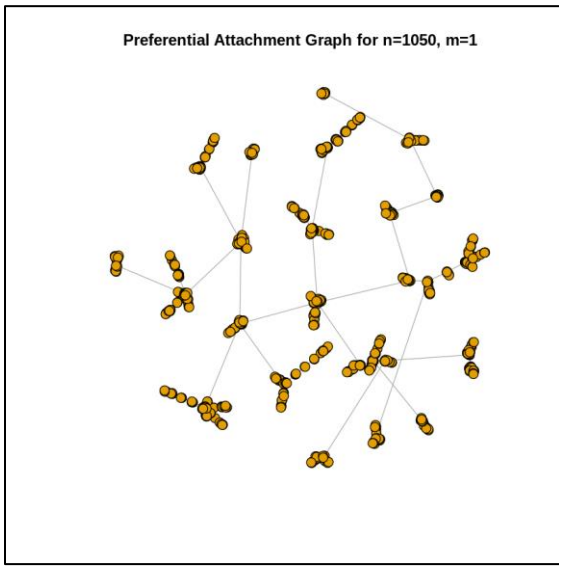

Fig. 10 Barabasi-Albert Model for n=1050 and m=1

In theory, the preferential attachment graph is always a connected graph since each node is connected to *m* other nodes and each of the *m* nodes is in turn connected to other nodes and from this recursive-like nature, it be inferred that a preferential attachment graph is always connected. However, to test this, similar to what we've done before, 100 different preferential attachment graphs were generated and checked for connectedness and the probability of a preferential attachment graph being connected came to 100%.

Question 2b

In this experiment, the community structure of the graph is found and the graph's modularity and assortativity are measured. The definitions of modularity and assortativity are as follows. *Modularity* measures the strength of division into network modules and can range from -0.5 to 1, where the higher the value, the more "modular" the graph is or in simpler words, the easier it is to define communities inside the graph which have dense connections within the community but sparse connectivity to vertices outside of the community.

Assortativity measures the level of homophyly of the graph based on some value, or in simpler words it measures the tendency of nodes to connect to nodes of 'similar' value over 'dissimilar' value where the value is defined as a structural property. For this project, the degree of a node was used as its value, therefore the metric was measuring whether nodes of certain degree were connected to nodes of similar degree or were they connecting to nodes with degrees 'dissimilar' from its own degree. The equation used for calculating the assortativity coefficient is given in equations (8) and (9).

$$q_i = \sum_i e_{ij} \tag{8}$$

$$Assortativity = \frac{1}{\sigma_q^2} \sum_{jk} jk\left(e_{jk} - q_j q_k\right) \tag{9}$$

σ_q^2 is the variance over q, and q_i measures the fraction of vertices connecting type i and type j which in our case measures the fraction of vertices connecting with degree type i to type j.

Furthermore, for this question, the community structure was found using Newman's algorithm which works on maximising the Newman's modularity score and is shown in equation (10).

$$modularity(Cluster_i) = \sum_{i,j \epsilon Cluster_i} A_{ij} - \frac{k_i k_j}{2m} \tag{10}$$

A_{ij} is an element of the node-node incidence matrix and k_i/k_j are the degree of nodes inside $Cluster_i$. In the community-finding algorithm, the goal is to maximize the expected value of modularity over all clusters by iteratively updating the clusters by merging them if and only if it increases the total expected modularity of the graph.

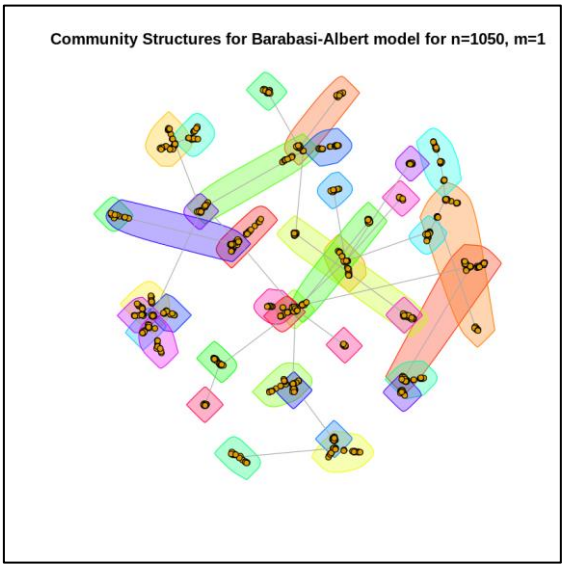

Fig. 11 Community Structure for a Barabasi-Albert Model for *n=1050* and *m=1*

At the beginning the algorithm initialises all nodes to be their own cluster which are then iteratively merged to maximise the expected modularity of the graph. The modularity and assortativity scores have been mentioned in Table 5.

The modularity an assortativity score highlight how the preferential attachment network is highly modular since it can be broken apart into communities because of its high modularity score. The assortativity score highlights how the nodes in the graph do not connect to similar nodes which is true since each new node added to the graph is more likely to connect to high-degree vertices, therefore similar nodes do not tend to connect to each other. The community structure has been visualised in Fig.11.

Table 5 – Modularity and Assortativity Score for an undirected preferential attachment network for $n=1050$ and $m=1$

Modularity Score	0.9333
Assortativity Score	-0.0885

Question 2c

In this experiment, a new undirected preferential attachment graph is generated with *n=10500* and *m=1* is generated and its modularity and assortativity score is compared to the previous graph generated with *n=1050*. The results are tabulated in Table 6 and the graph and its community structure are visualised in Fig. 12 and 13.

Table 6 - Modularity and Assortativity Score for an undirected preferential attachment network for *n=10500* and *m=1*

Modularity	0.9791
Assortativty	-0.0494

Fig.12 Barabasi-Albert Model Visualization for *n=10500* and *m=1*

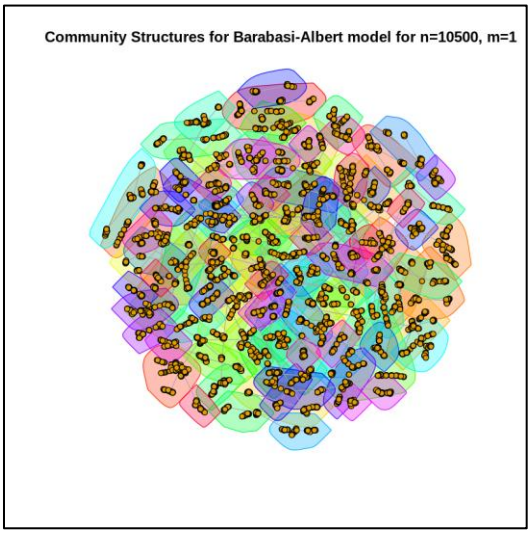

Fig. 13 Community Structure for a Barabasi-Albert Model for *n=10500* and *m=1*

The larger network has a higher modularity score which is intuitive as well since there are 10 times more nodes and in the preferential attachment network nodes attach to other nodes of high degree, therefore for larger and larger graphs, nodes are more likely to connect to high degree nodes which means that communities are more likely to form in larger networks.

The larger network has an assortativity score which I lower than the assortativity score of the smaller network and the logic behind that is that because of the presence of 10 times more nodes, there is a larger number of vertices from which a new node can choose from leading to nodes of varying degrees and a lower likeliness to connect to highly dissimilar nodes, thus the assortativity score for the larger network is still negative, but closer to zero.

Question 2d

In this question, the goal was to plot the degree distribution of two preferential attachment graphs on a log-log scale for *n=1050* and *n=10500* and estimate the slope of the plot using linear regression. The degree distribution of a preferential attachment network follows the power-law exponent which is given by equation (11), where *m* is the number of nodes each new node attaches to.

$$P(degree(v_i == k)) == \frac{2(m)(m+1)}{k(k+1)(k+2)} \sim k^{-3} \tag{11}$$

The log-log distribution would then be linear with a slope near -3 as we'll show below in equation (12).

$$\log P(degree(v_i == k) \sim 2log(2m) - 3log(k) \tag{12}$$

The theoretical approximation and practical values can be seen in Table 6. What can be seen is that as n increases, the values start to approach the expected values, this could be attributed to the discretization error when sampling from a continuous probability distribution. The results on a log-log scale can be seen in Fig. 14.

Table 6 – Theoretical and Actual values for Linear Regression on Log-Log Scale

Value of n	Theoretical Slope	Actual Slope
1050	-3	-2.7922
10500	-3	-2.9647

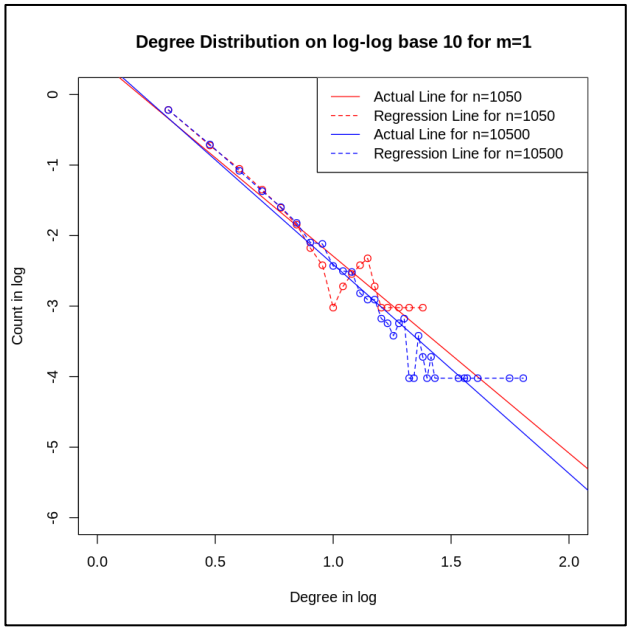

Fig.14 The degree distribution for *n=1050* and *n=10500* plotted on a log-log scale

Question 2e

In this experiment, the goal is to randomly pick a node *i* and then randomly pick its neighbour *j* and plot the degree distribution of j on a log-log scale and perform linear regression on the degree distribution. This experiment was done for both *n=1050* and *n=10500*. The degree distribution can be seen in Fig.15 and the theoretical and actual slope of the graph are tabulated in Table 7.

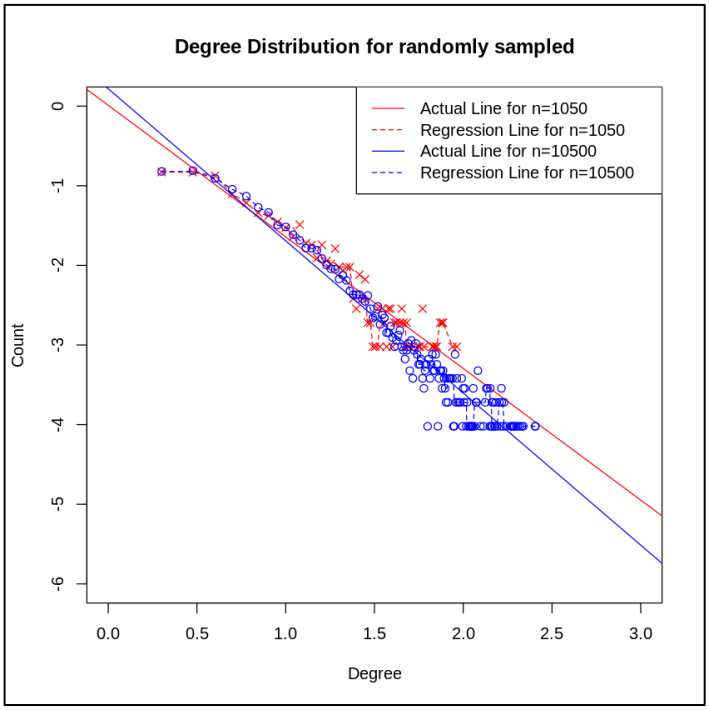

Fig.15 Degree Distribution on a log-log scale for *n=1050* and *n=10500*

Table 7 – Theoretical and Actual values for Linear Regression on Log-Log Scale

Value of n	Theoretical Slope	Actual Slope
1050	-3	-1.65284
10500	-3	-1.9106

The degree distribution does follow a linear trend, but it does not comply with the theoretical degree distribution as much as the previous experiment and the reason being that in this experiment, we have simulated a random walk with a single step, which in large networks with 1050 or 10500 nodes, means that we've only partially seen all the nodes, therefore it lacks global knowledge of the graph. Furthermore because we are randomly sampling with only a single timestep, the results can vary from one run to another since we are not reaching the steady state response which is time-invariant.

While in theory if we carry out an infinite number of runs, we should be able to recreate the degree distribution perfectly, however because of the fact that *m=1* and generation of *pseudo-random* numbers, there is high chance that some nodes will be visited more than others thus there will be a parity between the theoretical and actual values obtained.

Question 2f

In this experiment, we had to simulate the expected degree of a node that is added at timesteps i as $1 \leq i \leq 1050$. The practical results are shown in Fig 16, As we can see, the expected degree of a node increases monotonically with the age of the node, which is intuitive since the older the node is, the more chances it has to get a new nodes attached to it, which would increase its degree and thus reinforce its chance at getting new nodes attached to it. The theoretical formula that models the degree of a node is shown in equation (13), where v_j is the $vertex_j$ (added at time step j) under question and i is the timestep at which we're seeing the degree of a node.

$$k(j,i) = m\sqrt{\frac{i}{j}} \tag{13}$$

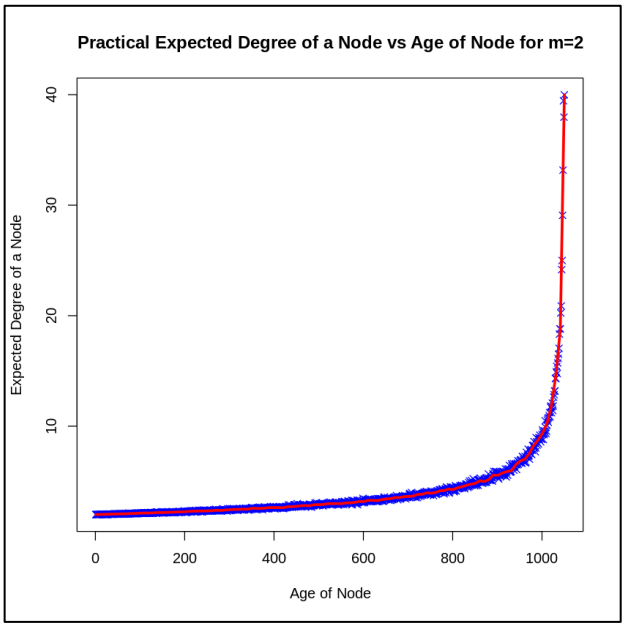

Fig 16 Actual Simulation between the age of a node and its expected degree

Question 2g

In this experiment, the task is to repeat the exercises 2a-2f for *m=2* and *m=6*.

1. Creating an undirected network and checking if it is always connected

For *m=2* and *m=6*, a total of *100* trials were conducted to see if these graphs were always connected and the results are that both graphs are connected 100% of the time, which is similar to the results for the graph when *m=1*, since in a preferential attachment graph network, each node is guaranteed to attach to other nodes which are in turn attached to other nodes. The results for *m=2* and *m=6* can be seen in Fig. 17 and Fig.18 respectively.

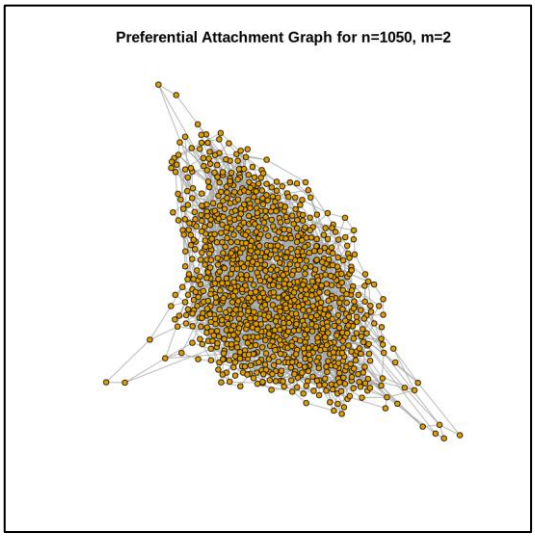

Fig. 17 Visualisation of a Barabasi-Albert Graph for *m=2*

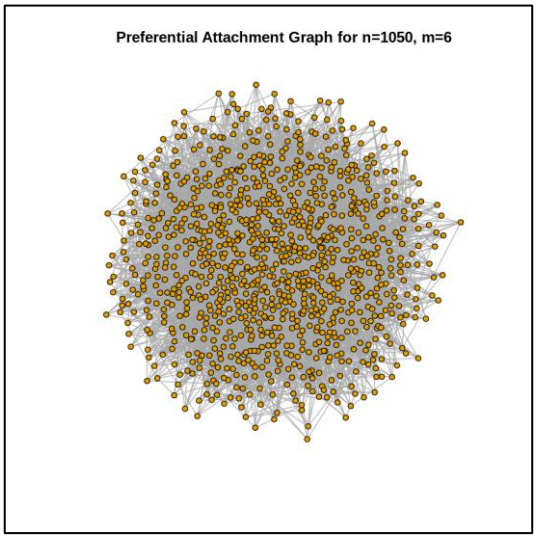

Fig. 18 Visualisation of a Barabasi-Albert Graph for *m=6*

2. Find the community structure and find the modularity and the assortativity.

The modularity and assortativity of the both graphs are noted in Table 8 and the respective visualisation with the community structures are depicted in Fig. 19 and Fig.20.

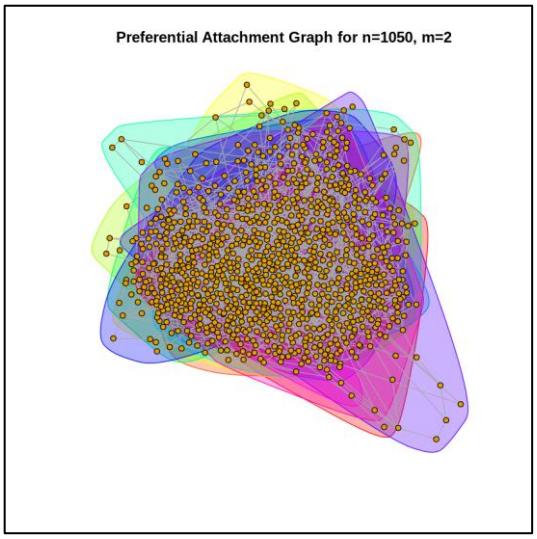

Fig. 19 Visualisation of a Barabasi-Albert Graph for *m=2*

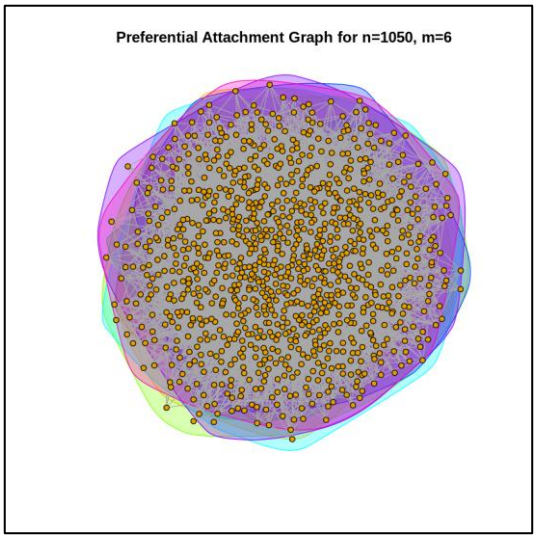

Fig. 20 Visualisation of a Barabasi-Albert Graph for *m=6*

From our earlier experiments and from this we can conclude two things:

- As *n* increases, the modularity of a graph increases which is intuitive since there are more nodes which are of higher degree and as new nodes get attached to it, it is easier to form communities because of the presence of a handful of high-degree nodes which are in turn connected to most nodes.
- As *m* increases, the modularity of a graph decreases, because since each node is being attached to more and more nodes, it becomes increasingly more difficult to separate it into a community by itself since it can be connected to nodes of different communities and according to equation (10) which is used for forming communities by maximising the estimated modularity score, which is inversely proportional to *2m*, thus if we were to increase *m*, the expected value of modularity would decrease.

Table 8 – Modularity and Assortativity Scores for *m=2* and *m=6*

m	Modularity	Assortativity
2	0.525378	-0.061935
6	0.242335	-0.029726

3. Generate a network with *n=10500* and compare its modularity and assortativity with the smaller network

The visualisations of the community structures for *m=2* and *m=6* are shown in Fig. 21 and Fig.22 respectively and the modularity and assortativity scores are tabulated in Table. 9

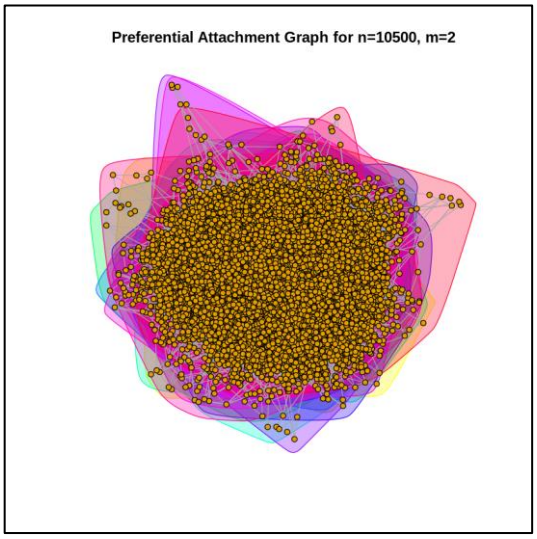

Fig. 21 Visualisation of a Barabasi-Albert Graph for *m=2*

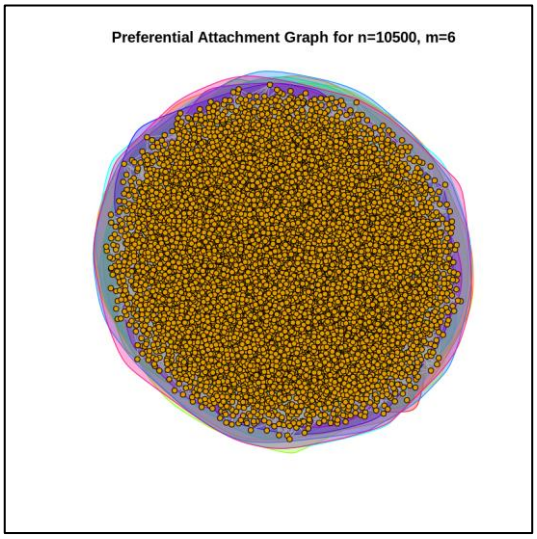

Fig. 22 Visualisation of a Barabasi-Albert Graph for *m=6*

Table 9 – Modularity and Assortativity Scores for *m=2 and m=6*

m	Modularity	Assortativity
2	0.528976	-0.009399
6	0.245573	0.000565

Besides the observations made in the previous experiment on the effect of *n* and *m* on modularity. Other conclusions can be made on the assortativity as well such as:

- As *n* increases, the assortativity of a graph increases. Assortativity measures the likeliness of nodes connecting to other similar nodes, and since *n* increases, there are more possible options from which a vertex can choose from, thus the assortativity score increases since in a smaller network, a handful of high degree nodes dominate the entire node matching procedure, but in a larger network there are several high degree nodes and other vertices as well. Moreover from Table 9, the assortativity scores for the larger graphs are around 0, indicating that as if the graph was built from random matching which does make sense especially since there are large number of possible vertices to chose from that the probability of choosing one over the other comes close to random choice.
- As *m* increases, the assortativity of a graph increases, since each node now connects to many other nodes and in the process connects to other 'similar' nodes. When *m* is small, there isn't much choice because each node is inclined to attach to the high-degree node. But as *m* is increased, each node needs to look beyond a single high-degree node and needs to search for other viable nodes to match with. Thus there's a linear relationship between *m* and the assortativity.

4. Degree Distribution for *n=1050,10500* on a log-log scale and estimate the slope of the plot

The results for both *m=2* and *m=6* have been visualised in Fig.23 and Fig.24 respectively and the theoretical and actual results achieved for the slope are present in Table.10

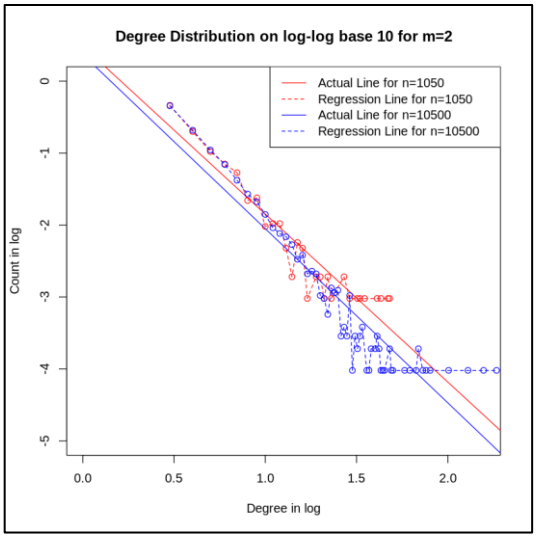

Fig. 21 Degree Distribution on a log-log scale for a Preferential Attachment network for *m=2*

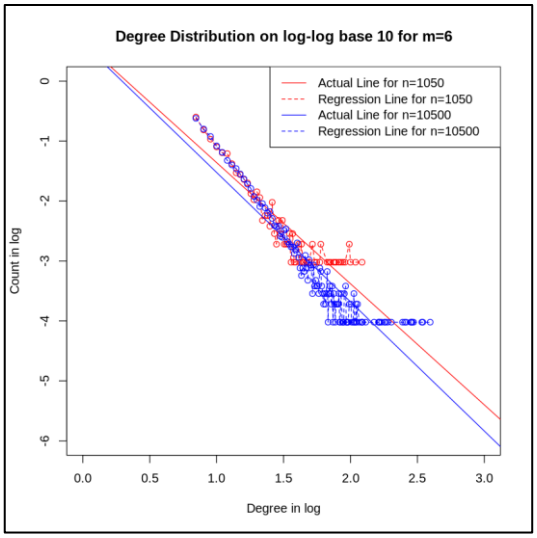

Fig. 22 Degree Distribution on a log-log scale for a Preferential Attachment network for *m=6*

Table 10 – Theoretical and Actual Slopes for different values of m

Value of m	Value of n	Theoretical Slope	Actual Slope
2	1050	-3	-2.33
2	10500	-3	-2.42
6	1050	-3	-2.02
6	10500	-3	-2.16

The following inferences can be made:

- As *n* increases, the graph comes closer and closer to the theoretical limits, since as previously argued one can think of the difference from the theoretical limit as the discretization error and as n increases, the degree distribution becomes smoother and smoother and is more continuous thus coming closer to the theoretical values
- As *m* increases, the graph strays away from the theoretical limits, since each node now connects to *m* other nodes which makes the average degree of the node increase, thus causing the degree distribution to shift to the right, making it less negative

5. Randomly picking a node *i* and then its neighbour j and plotting the degree distribution

The results for *m=2* and *m=6* can be seen in Fig.23 and Fig.24 respectively and the results pertaining to the theoretical and actual regression results are shown in Table 11.

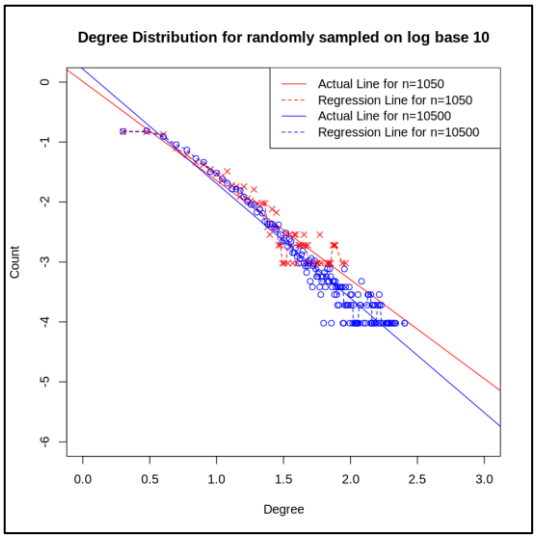

Fig. 23 Degree Distribution on a log-log scale for a randomly sampled set of neighbours for *m=2*

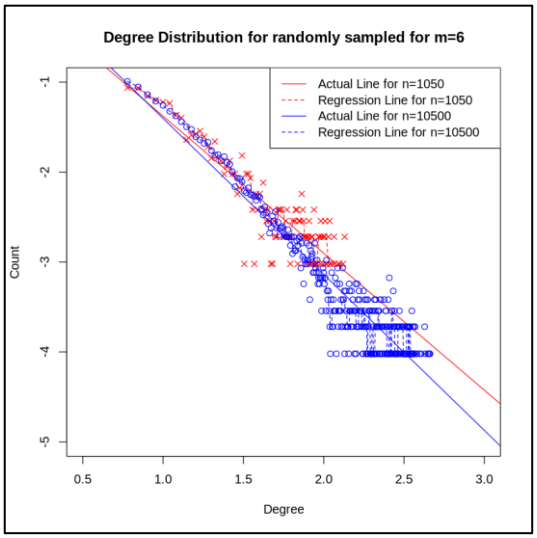

Fig. 24 Degree Distribution on a log-log scale for a randomly sampled set of neighbours for *m=6*

Table 11. Theoretical and Actual Slopes for different values of m when randomly sampled

Value of m	Value of n	Theoretical Slope	Actual Slope
2	1050	-3	-1.65
2	10500	-3	-1.91
6	1050	-3	-1.52
6	10500	-3	-1.73

The following inferences can be made from the results:

- As *n* increases the linear regression results come closer to the theoretical limit with the argument that can be made as before about the sampling noise and discretization error introduced into a continuous distribution.
- As *m* increases, the linear regression results stray further away from the theoretical limit and as previously argued, one interpretation of it is the fact that as each node connects to more and more vertices, the average degree of a node increases, causing the graph to shift the right and making the slope less and less negative, or more and more positive.

The results do stray far from the theoretical limit, because of the fact that we are only randomly sampling a single neighbour and are not repeatedly doing this, which further means that we aren't reaching the steady-state values of the random-walker and are instead sampling the transient values which are time-variant. Furthermore sampling a single node doesn't build a proper representation of the entire network and therefore we get much more noisier degree distribution data which is reflected in the disparity between the theoretical values and actual values.

6. Relationship between the degree of a node and the age

The results for $m=2$ and $m=6$ are shown in Fig.25 and Fig.26 respectively.

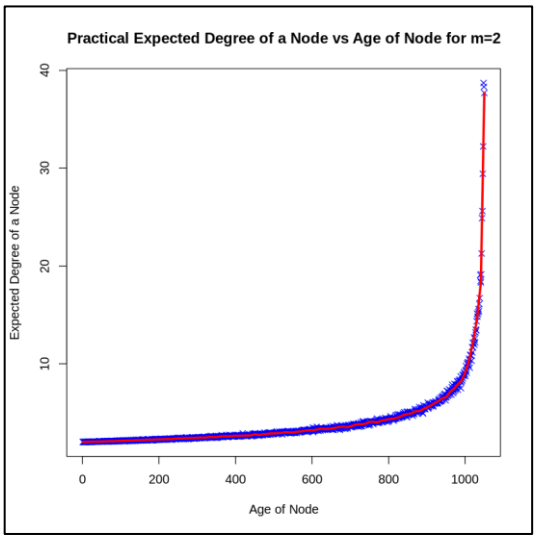

Fig. 25 Relationship between the degree of a node vs the age for $m=2$

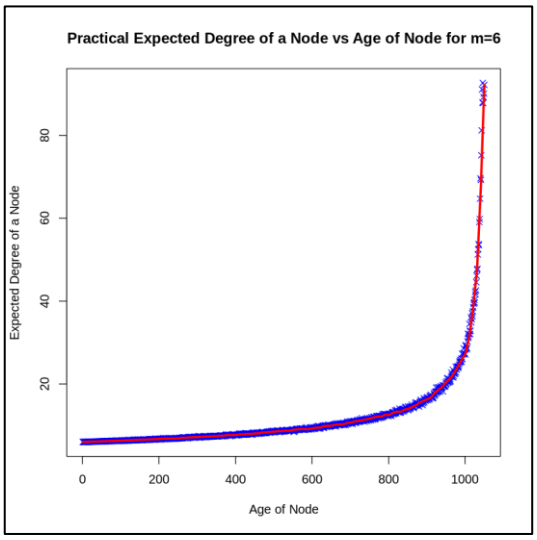

Fig. 26 Relationship between the degree of a node vs the age for $m=6$

The only inference that can be made is that as m increases, the degree of a node increases as well, which makes intuitive sense since each vertex needs to attach to m other vertices at each timestep and as m increases, the degree of m nodes increases by 1 and coupled with preferential attachment, some nodes get attached more than others. Thus because of m nodes needing to be attached and the preferential attachment model, the rate at which the graph climbs increases with an increase in m, the mathematical relationship between the two is shown in equation (13).

Question 2h

In this experiment, firstly a preferential attachment model was first generated, then its degree sequence is used along with a stub matching procedure to create a new graph. For this experiment the *Vigor-Latapy* algorithm was used for stub-matching and the results are as follows. The visualisation of both graphs can be seen in Fig.27 and Fig.28 respectively. The modularity of the preferential attachment graph came out to be 0.929855 and the modularity of the model generated from stub matching came out to be 0.935078, which means that the stub matching procedure gave a comparatively more modular

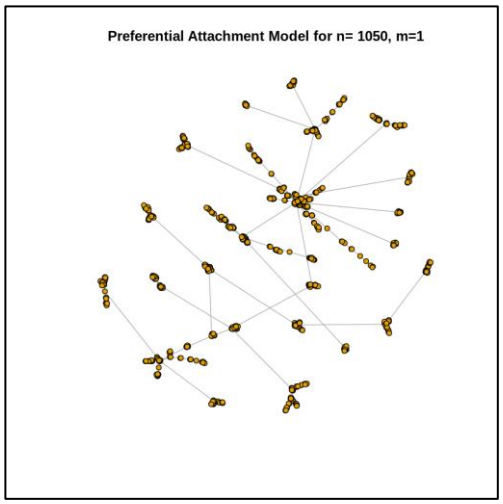

Fig. 27 Visualisation of a Preferential Attachment Graph for *n=1050* and *m=1*

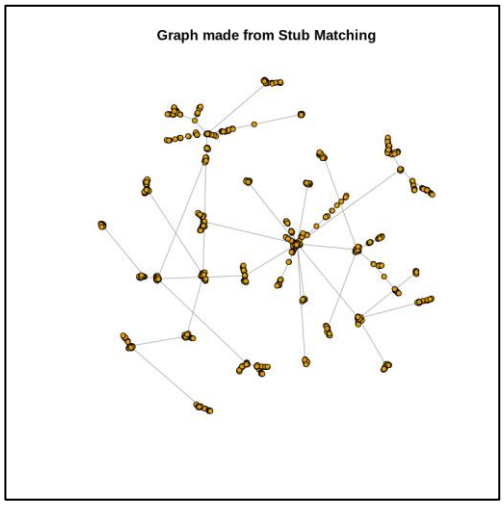

Fig.28 Visualisation of a Preferential Attachment Graph for *n=1050* and *m=1,* generated from stub matching

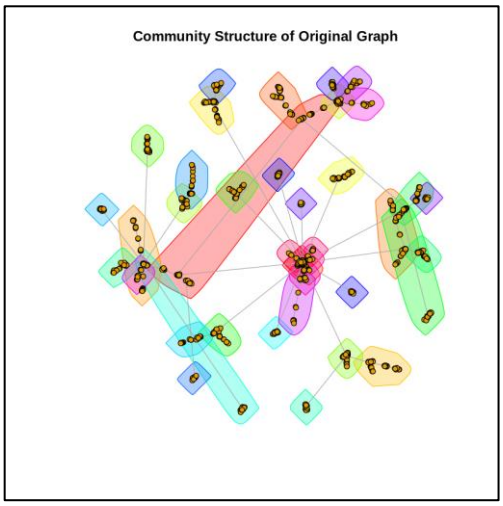

Fig.29 Community Structure for a Preferential Attachment Model

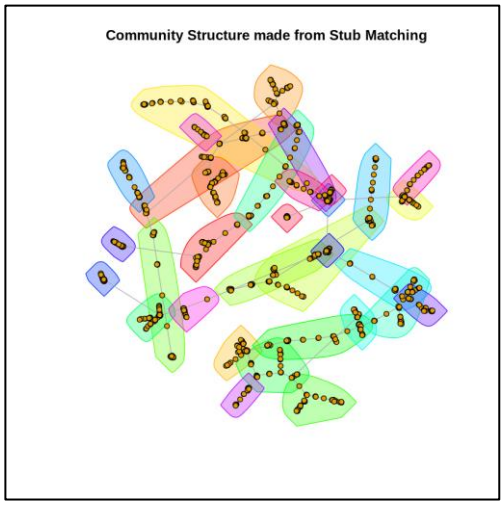

Fig.30 Community Structure for the graph generated from stub matching

graph. While there are other stub matching algorithms as well, the *Vigor-Latay* algorithm was chosen because it prevents graphs from having loops and works well on undirected graphs. In a preferential attachment model graph, a node attaches to other nodes with the probability of attaching to a higher degree node is higher. In stub matching, each node has a degree value assigned to it and at run-time, vertices are randomly sampled and connected together (there is no inherent preference in this unlike in Barabasi-Albert Model). Compared to simple stub matching algorithms which are poor at forming modular graphs, the preferential attachment model develops more modular graphs. In comparison to

the stub matching algorithm *Vigor-Latay*, which uses monte-carlo simulations to maximise the modularity of its stub matching, the result is a more modular graph which can be seen in the modularity score and in Fig. where there are more communities in Fig.29 than in Fig. 30.

Question 3a

In this experiment, we modified the probability model used to add a new vertex to the network by penalizing the age of the node. The graph and its degree distribution plotted on a log-log scale are shown in Fig.31 and Fig.32 respectively. The power law exponent for this graph can be inferred from the degree distribution when plotted on a log-log scale since the distribution becomes linear when mapped to the log-log scale.

By calculating the slope of the linear regression line, it can be seen that the slope of the graph is -3.3479. Thus the power law exponent of this graph is 3.3479, which does come close to the theoretical power law exponent of 3.

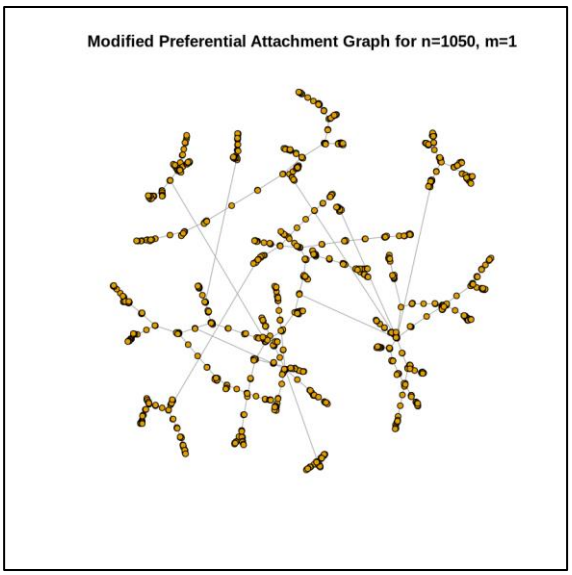

Fig.31 Visualisation of the Modified Preferential Attachment Node

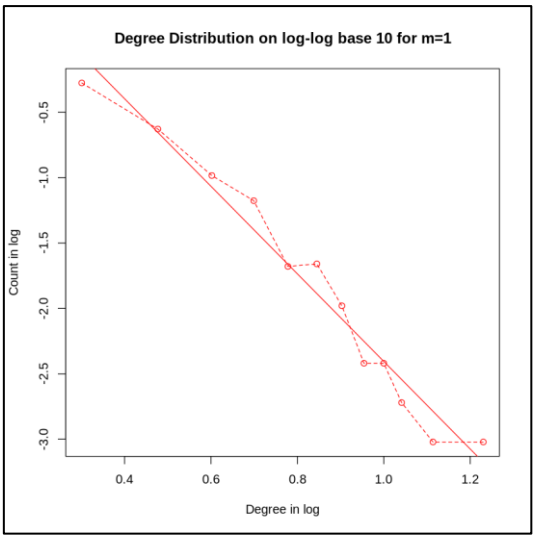

Fig.32 Degree Distribution plotted on a log-log scale.

Question 3b

After using the fast-clustering algorithm proposed by Mark Newman, the modularity score came out to be 0.936, which indicates that the graph is highly modular and can be broken down into many communities which can be seen in Fig.33 From the graph it can be seen that most clusters have similar community sizes and that is the case since the algorithm penalizes the node from selecting older nodes and thus prevents a positive reinforcement of high degree nodes becoming even more high degree.

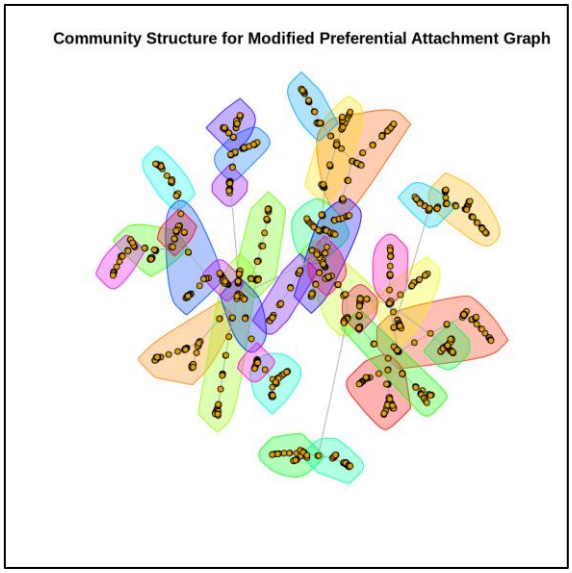

Fig.33 Community Structure for Modified Preferential Attachment Graph

Random Walk on Networks

Question 1a

In this experiment, the goal was to generate an undirected Erdos-Renyi graph with $n=900$ and $p=0.015$. The visualisation of this graph can be seen in Fig.34

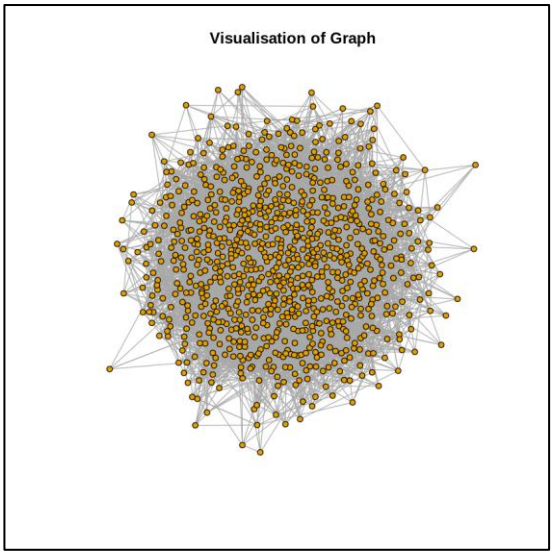

Fig.34 Visualisation of Erdos-Renyi graph used for the random walk experiment

Question 1b

In this experiment, we had to simulate a random walker and plot the shortest distance of the random walker from some vertex to the starting point over a *100* timesteps. To receive probabilistically significant results, we had carried out 900 random starting points and each random walker was then made to randomly walk for a 100 timesteps. These results were then used as the basis for calculating the mean and variance of the shortest path of the random walker against timesteps *t*. The results for this can be seen in Fig.35 and Fig.36 respectively.

From the graphs its can be inferred that the random walker does reach a steady state after seeing less than 2% of all nodes, The expected shortest distance settles at 2.87 and the variance of the shortest distance peaks at 0.55 before settling at 0.30. Because this is an undirected graph, and the nodes are randomly connected to each other, the random walker would end up in the vicinity of the starting point.

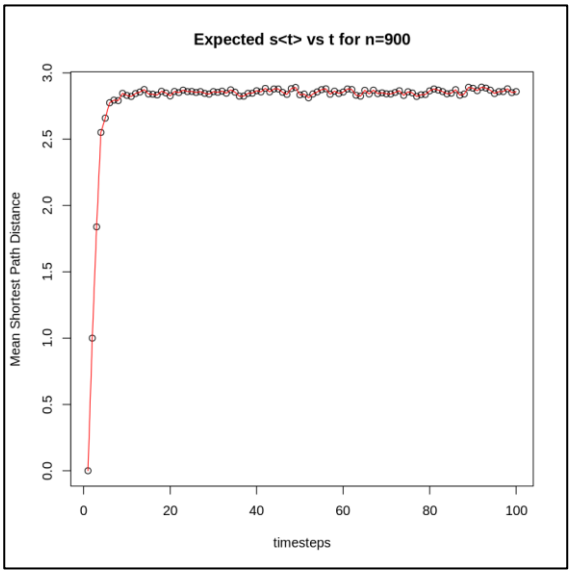

Fig. 35 Expected Shortest Distance of Random Walker against timesteps *t*

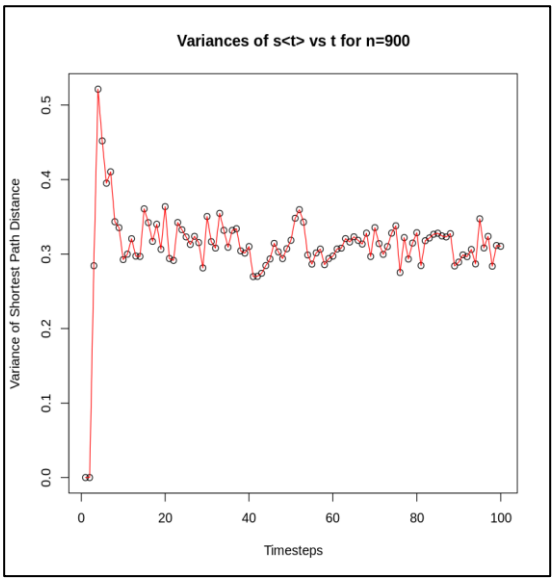

Fig.36 Variance of Shortest Distance of Random Walker against timesteps *t*

Question 1c

In this experiment, the degree distribution of the vertices the random walker has visited is collected and its degree distribution is plotted on Fig.37 and the actual degree distribution of the graph in Fig.38.

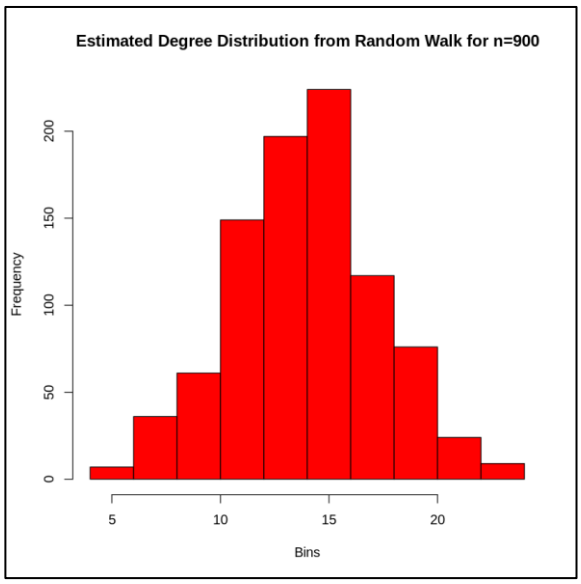

Fig.37 Degree Distribution of Nodes visited by Random Walker

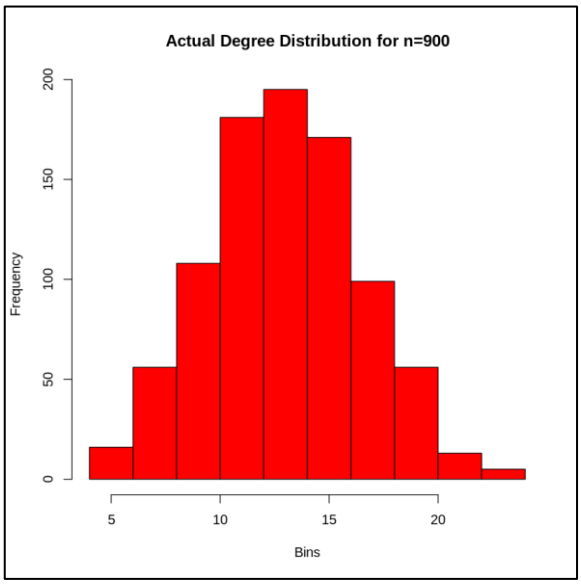

Fig.38 Actual Degree Distribution of the Graph

We have already established that the degree distribution of an Erdos-Renyi graph is a *Binomial distribution* that approaches a *Laplacian distribution*. The distribution of the random walker is a *Binomial distribution* as well with a similar shape and frequency as that of the actual degree distribution. For a couple of bins, the estimated degree is more than the actual which solely means that the random walker has visited the same node multiple times in a random walk, thus these nodes contribute more than once in the histogram, otherwise both the estimated degree distribution and the actual degree distribution are both of similar distributions and shape.

Question 1d

In this experiment, we had to generate an Erdos Renyi graph with *n=9000* and compare the results we get when we do a random walk on it, such as the estimated shortest distance, variance in the shortest distance. The visualisation of the graph can be seen in Fig.39 In this example, the number of random walks were increased to 9000 instead of 900 to make sure that it is statistically significant. The results from the random walk are shown in Fig.40 and Fig.41 respectively.

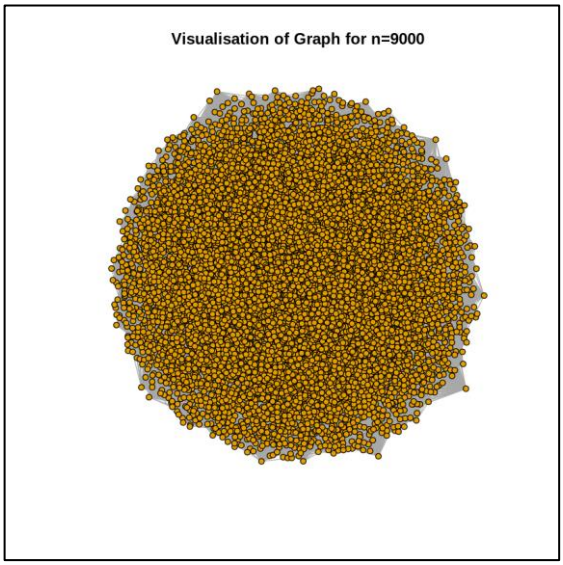

Fig.39 Visualisation of an Erdos – Renyi Graph for *n=9000* and *n=0.015*

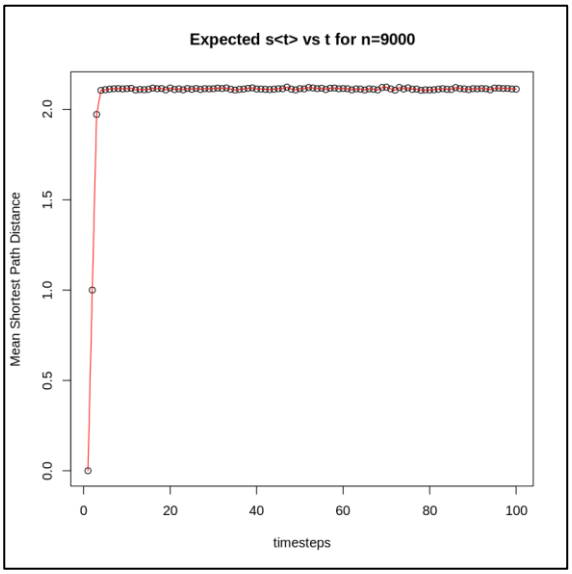

Fig.40 Expected Shortest Distance of Random Walker against timesteps *t*

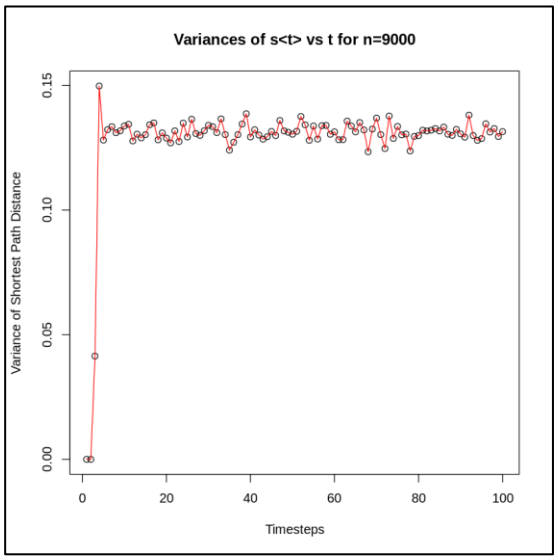

Fig.41 Variance of Shortest Distance of Random Walker against timesteps *t*

Similar to the previous random walk experiment, the random walker also reaches a steady state even when we increase the graph to 10 times its original size. The steady state values of the expected shortest distance of the random walker are 2.11 and the steady state values of the variance of the shortest distance

of the random walker is 0.13. Moreover, the random walker reaches the steady state values in fewer timesteps.

From a qualitative lens, we can see that as we increased the graph size the steady state values decreased. The diameter of the graph when $n=900$ is 5 and the diameter of the graph when $n=9000$ is 3, which would mean that the diameter of the graph does play a role in the random walker's performance.

The diameter represents how far any two nodes can possible be in the graph and the smaller the diameter is, that means that the graph is highly connected and a random walker can reach to any node in fewer steps. Therefore, the random walker reaches a steady state value quicker since it can iterate through the entire graph much more quickly and because of the decreased diameter, it is at any time no less than 3 hops away from the starting point, thus it's steady state value decreased, because in the original graph the diameter was 5, therefore it could be more comparatively far away from the source.

The decrease in the variance can be explained as well, since the model reaches steady state faster, there is fewer variance in between runs of the random walk.

Question 2a

In this experiment, the goal was to generate a preferential-attachment model with *n=900* and *m=1*. The visualisation of this graph can be seen in Fig.42.

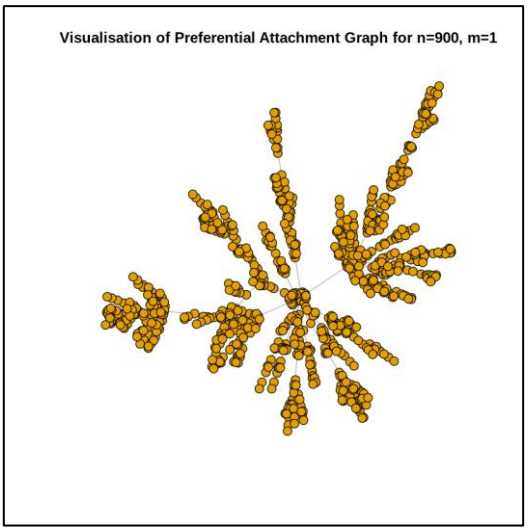

Fig.42. Visualisation of Barabasi-Albert model for *n=900* and *m=1*

Question 2b

In this experiment, we had to simulate a random walker and plot the shortest distance of the random walker from some vertex to the starting point over a *100* timesteps. To receive probabilistically significant results, we had carried out 900 random starting points and each random walker was then made to randomly walk for a 100 timesteps. These results were then used as the basis for calculating the mean and variance of the shortest path of the random walker against timesteps *t*. The results for this can be seen in Fig.43 and Fig.44 respectively.

From the graphs its can be inferred that the random walker does not reach a steady state, unlike the random walker used in an Erdos-Renyi graph, because there is a preference when nodes are being attached, the random walker would progressively go towards higher degree nodes,

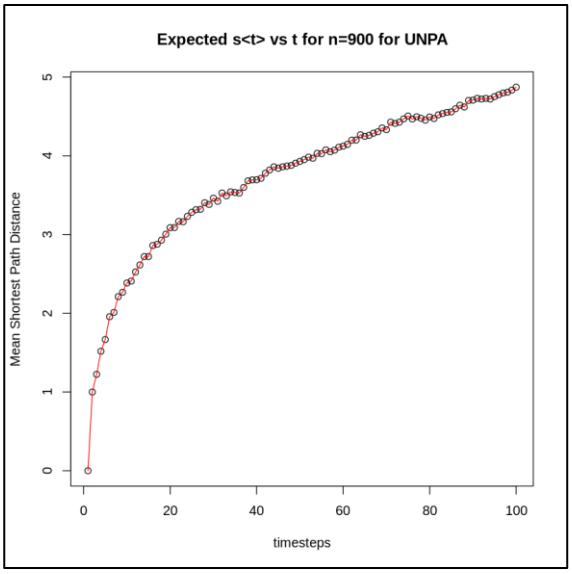

Fig.43 Expected Shortest Distance of Random Walker against timesteps *t*

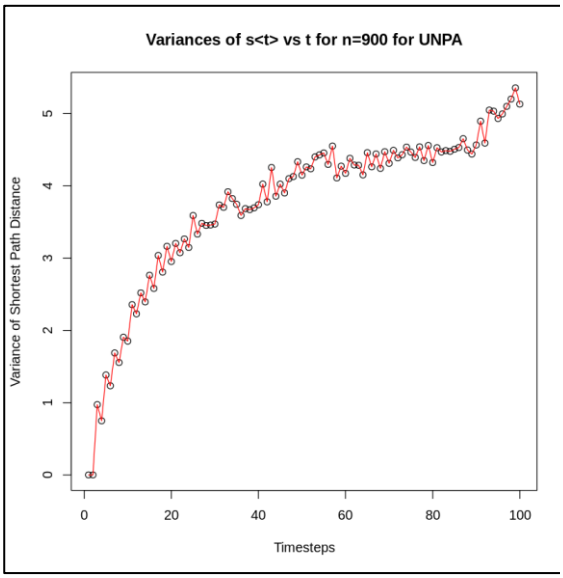

Fig.44 Variance of Shortest Distance of Random Walker against timesteps *t*

Question 2c

In this experiment, the degree distribution of the vertices the random walker has visited is collected and its degree distribution is plotted on Fig.45 and the actual degree distribution of the graph in Fig.46.

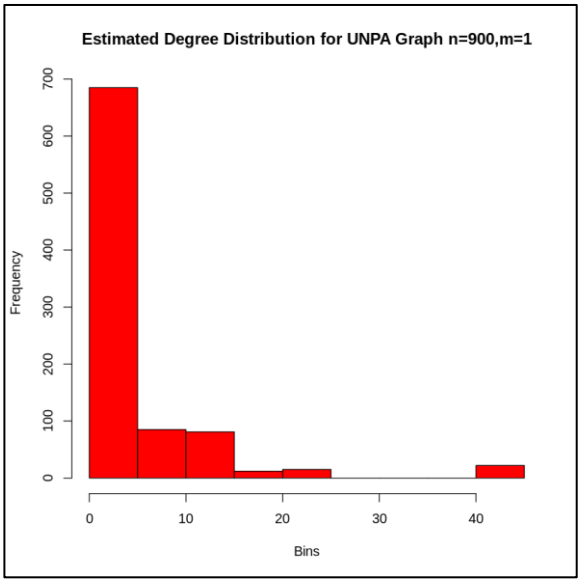

Fig.45 Degree Distribution of Nodes visited by Random Walker

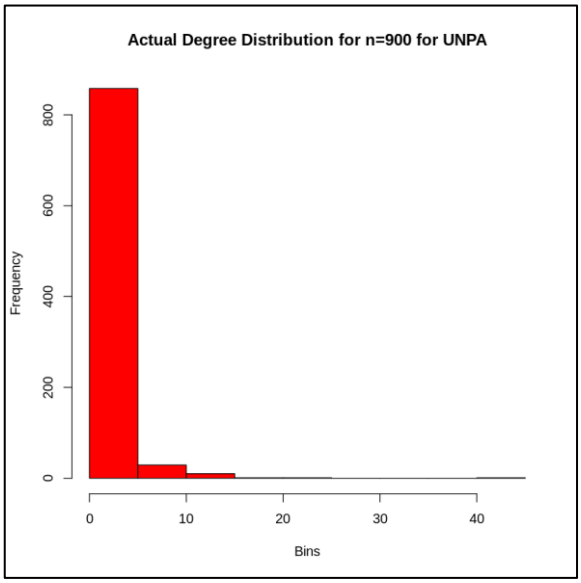

Fig.46 Actual Degree Distribution of the Graph

We have already established that the degree distribution of a Preferential Attachment graph follows the power law distribution. The distribution of the random walker is a power law distribution as well with a similar shape and frequency as that of the actual degree distribution. For a couple of bins, the estimated degree is more than the actual which solely means that the random walker has visited the same node multiple times in a random walk, thus these nodes contribute more than once in the histogram, otherwise both the estimated degree distribution and the actual degree distribution are both of similar distributions and shape.

Question 2d

In this experiment, two Barabasi-Albert graphs were generated for *n=90* and *n=9000* for *m=1*. The visualisation of both graphs can be seen in Fig.47 and Fig.48 respectively.

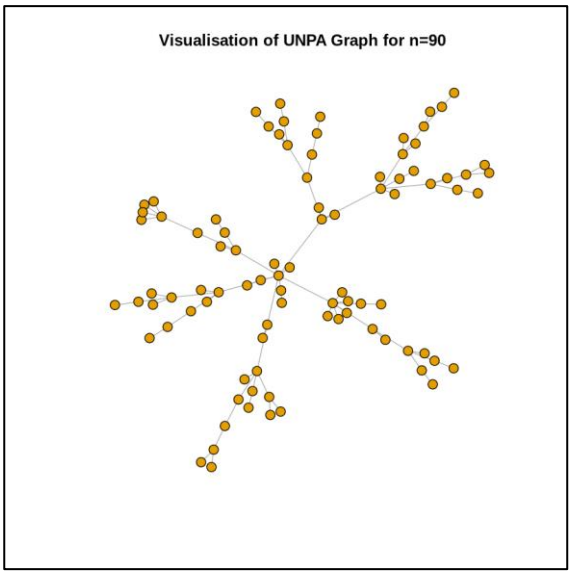

Fig.47 Visualisation of a Barabasi-Albert Graph for *n=90* and *m=1*

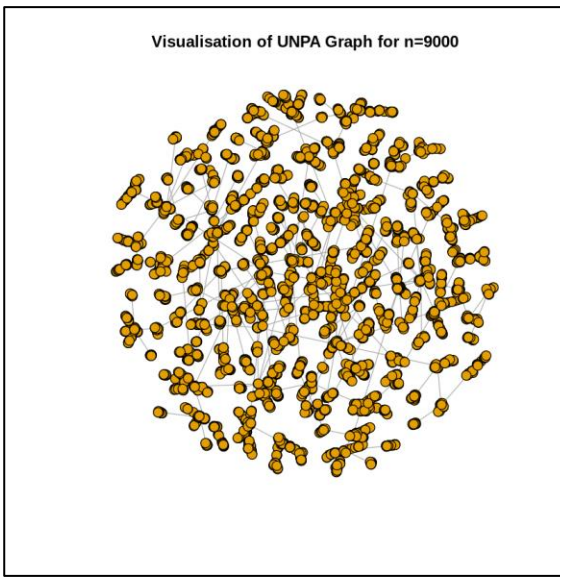

Fig.48 Visualisation of a Barabasi-Albert Graph for *n=9000* and *m=1*

The results for the random walker for *n=90* are shown in Fig.49 and Fig.50 and the results for the random walker for *n=9000* are shown in Fig.51 and Fig.52.

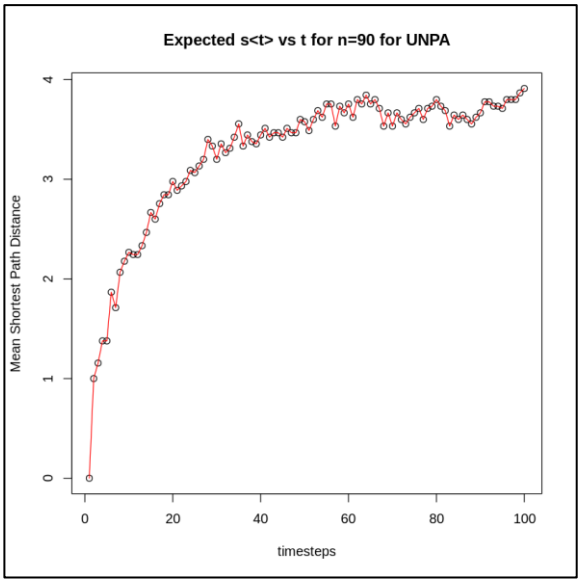

Fig.49 Degree Distribution of Nodes visited by Random Walker

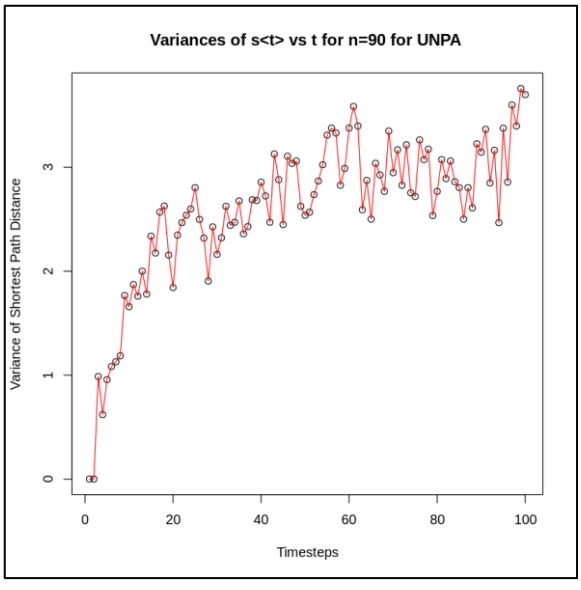

Fig.50 Actual Degree Distribution of the Graph

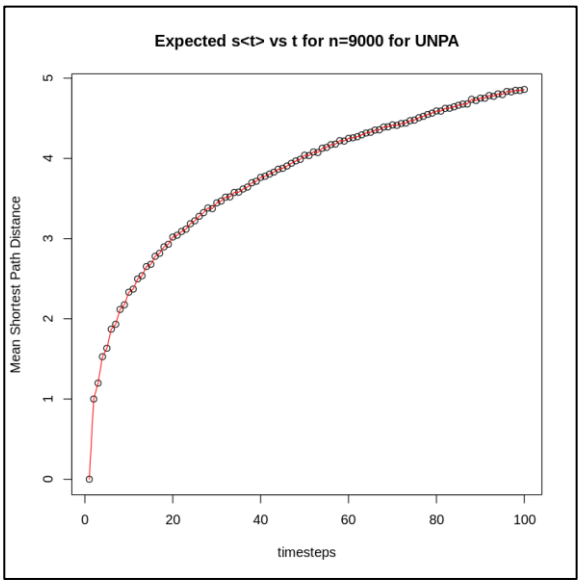

Fig.51 Degree Distribution of Nodes visited by Random Walker

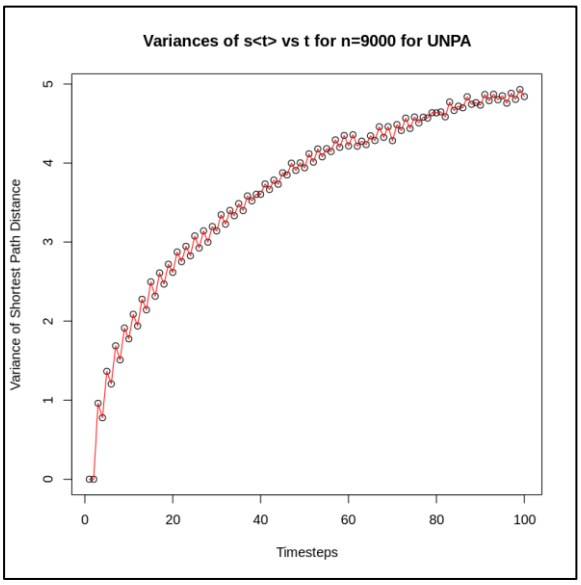

Fig.52 Actual Degree Distribution of the Graph

In between *n=90* and *n=900*, the distance the random walker covers within a *100* timesteps is less for *n=90*, and this reflects in both the mean distance and variance. For *n=900,* the random walker is on average 5 hops away from the origin by the end of the random walk. For *n=90*, the random walker is 4 hops away from the origin by the end of the random walk. For *n=900* the diameter of the graph came out to be 20 while for *n=90* the diameter came out to be 11. This plays a role in the distance as well. A smaller diameter coupled with a smaller graph, the random walker within a shorter time starts reaching the higher degree (older) nodes within the graph and

In between *n=900* and *n=9000*, the distance the random walker covers in the same timesteps has increased from 4 to 5 and the variance is higher as well. The diameter of the graph for *n=9000* is 28. With a similar argument we can argue that since the graph is extremely large with much higher degree nodes being present and the preference for any node to be attached to the higher degree nodes, the random walker in the same number of timesteps is able to cover a much larger distance from the random walker. The variance is higher because of the potential nodes the random walker can choose from at each point owing to the network being much larger.

The diameter does play a role in the estimated distance from the starting point, or the variance of the distance from the starting point. Unlike in an Erdos-Renyi graph where there was an inverse relationship. It is linear in a Barabasi-Albert graph. As the diameter increases, the mean of the shortest distance from the starting point increases for the same number of timesteps because there are much higher degree nodes which the walker can go to and can quickly travel far within a few timesteps. The variance is higher as the diameter increases because there are potentially more possible nodes for the random walker to choose from, thus the uncertainty at each timestep is high as well.

Question 3a

In this experiment, the goal was to calculate the *vanilla* PageRank scores without any teleportation or any other constraints as such. To prevent a black-hole from being formed in the graph if the random-walker starts at node 1, two directed preferential attachment graphs were generated and the second graph was permuted and added back to the original so that any node is connected in the 'directed' sense and there is no blackhole.

To compute the PageRank scores, a random walker was used and to overcome the fact that it takes *ln(n)* steps to reach a steady state, the final node the random walker visits at the end of n timesteps is used since *ln(n)*<<*n*, we do not have to worry about recording the non-steady state response of the random walker. The visualisation of the PageRank scores can be seen in Fig.53 and the relationship between the PageRank Scores and the degree of the vertex is shown in Fig.54.

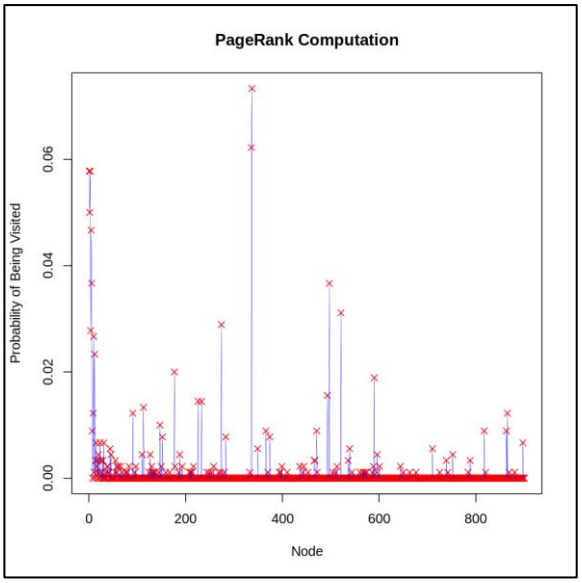

Fig.53 PageRank Scores simulated by a Random Walker

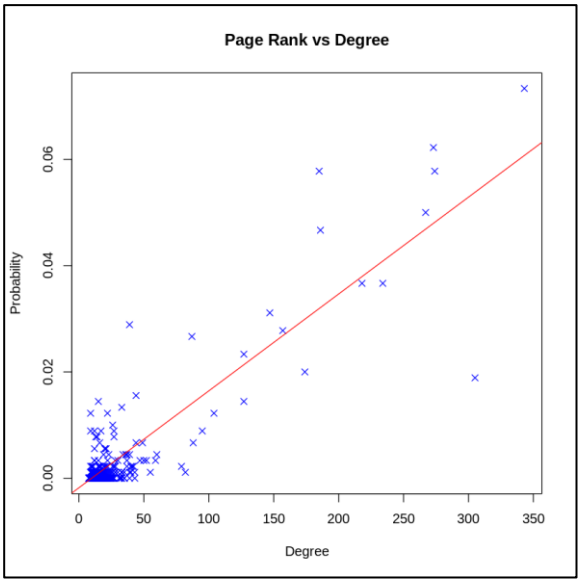

Fig.54 Relationship between the Degree of a Node and its PageRank Scores

The PageRank scores seen in Fig.53 do show that certain nodes are more important than others, and the results from Fig.54 show that there is a linear relationship between the degree and the PageRank scores. A Pearson's correlation score of 0.894 shows that there is a strong relationship between the two. The equation between the PageRank scores and the degree of a node are given in equation (14).

$$\pi(i) = \sum_{j=1}^{|V|} \frac{A_{ji}}{degree(vertex_j)} \pi(j) \quad (14)$$

Question 3b

In this experiment, the goal was to further the estimation of PageRank by accounting for teleportation with $a=0.2$. The calculations used for getting the raw PageRank scores are the same as the previous question where the last node of the random walk is used to overcome the issue of sampling before a steady state response. The visualisation of the PageRank scores with teleportation are seen in Fig.55 and the relationship between the degree and the PageRank scores can be seen in Fig.56.

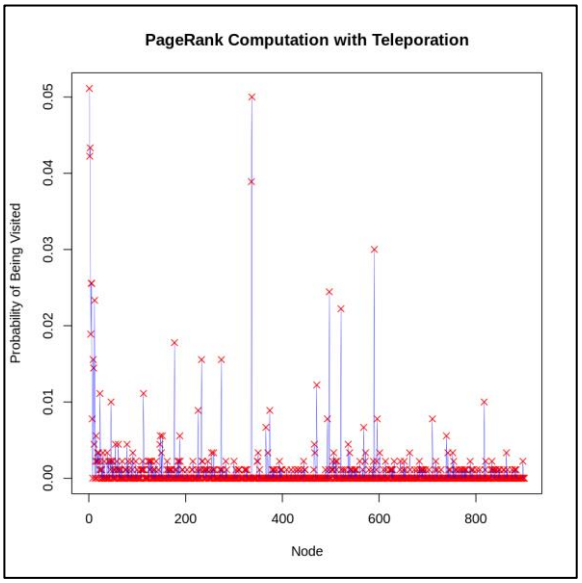

Fig.55 PageRank Scores simulated by a Random Walker with Teleportation

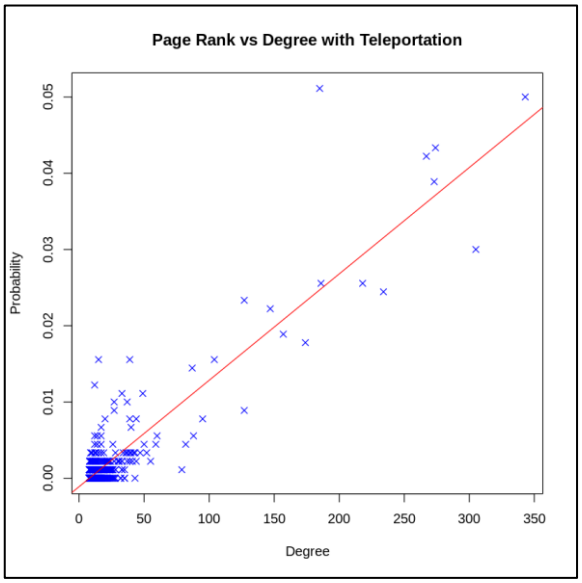

Fig.56 Relationship between the Degree of a Node and its PageRank Scores wit Teleportation

From comparing Fig.55 with Fig.53, we can see that the PageRank scores have decreased for most important nodes and other nodes have increased, since the random walker does not solely depend on the degree of a node to move around the graph. The equation for the PageRank scores is given by equation(15).

$$\pi(i) = (1-\alpha)\sum_{j=1}^{|V|}\frac{A_{ji}}{degree(vertex_j)}\pi(j) + \frac{\alpha}{|V|} \quad (15)$$

From equation (15) we can see that if we increase α, the PageRank scores become more and more closer to random choice and less important on the degree of a node. Therefore the PageRank scores generally decrease for more important nodes since the random walker can teleport to other nodes as well.

Question 4a

In this experiment, the aim was to further the calculation of PageRank scores by setting the teleportation probabilities proportional to the PageRank scores. The results for this are shown in Fig.57 and Fig.58 respectively.

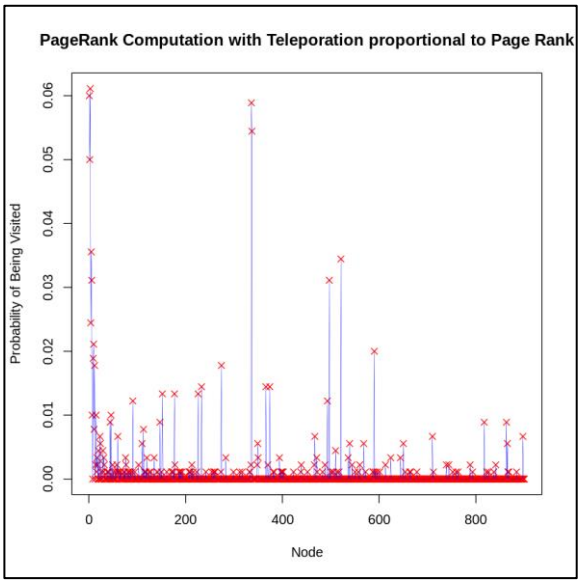

Fig.57 PageRank Scores simulated by a Random Walker with Teleportation

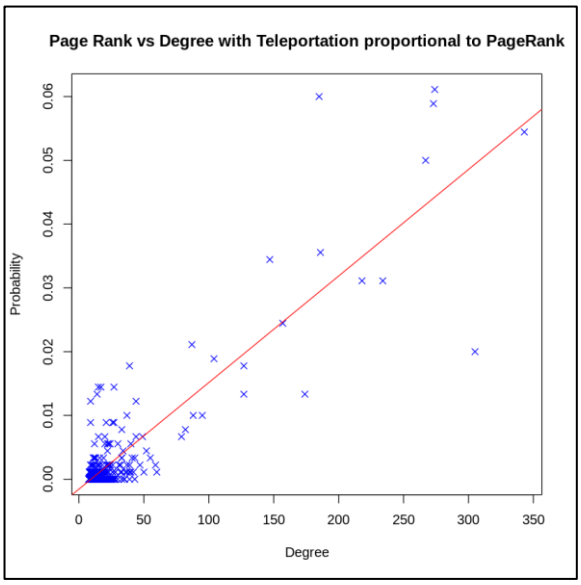

Fig.58 Relationship between the Degree of a Node and its PageRank Scores wit Teleportation

Comparing the results from Fig. 53 and Fig. 57, there are more nodes above 5% probability when the PageRank scores than there is in Fig.53, this highlights the "Rich gets richer and poor gets poorer" framework in PageRank which is a positive feedback cycle. And from Fig.58 we can see that the positive linear relationship between the degree of a node and the PageRank scores still holds valid. However, with the optimal PageRank scores will not be different from the original PageRank scores which will proved below.

$$\pi(i) = (1-\alpha)\sum_{j=1}^{|V|}\frac{A_{ji}}{degree(v_j)}\pi(j) + \alpha\sum_{j=1}^{|V|}\pi^*(i)\pi(j)$$

$$\pi(i) = (1-\alpha)\sum_{j=1}^{|V|}\frac{A_{ji}}{degree(v_j)}\pi(j) + \alpha\pi^*(i)$$

$$\pi^*(i) = \sum_{j=1}^{|V|}\frac{A_{ji}}{degree(v_j)}\pi(j)$$

The proof highlights that at the optimal PageRank scores, there are the same. Which is highlighted by the correlation score which came out to be 0.89 which is similar to the score we received for the original PageRank scores.

Question 4b

In this experiment, the goal was that the random walker can only teleport to only two nodes which hold the median page rank scores. The results from this are shown in Fig. 59 and Fig.60 respectively.

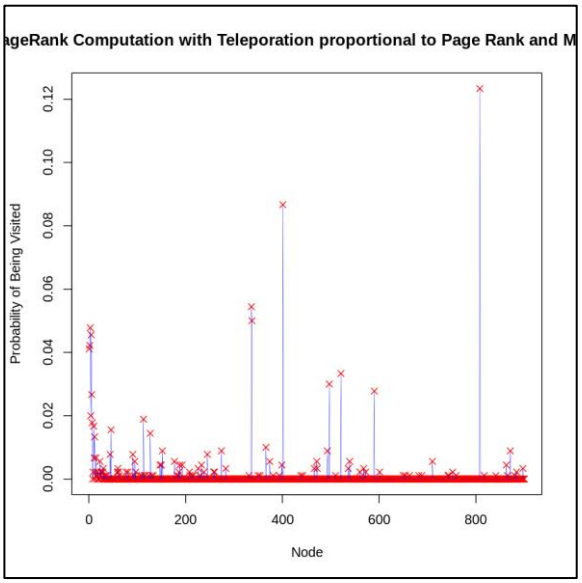

Fig.59 PageRank Scores simulated by a Random Walker with Teleportation

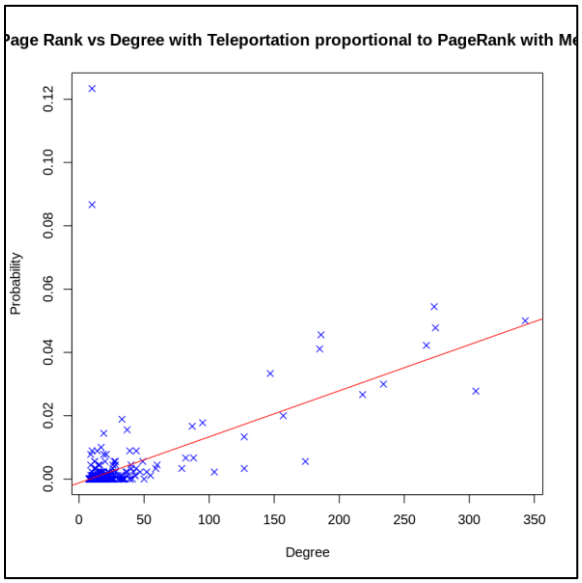

Fig.60 Relationship between the Degree of a Node and its PageRank Scores wit Teleportation

From Fig.60 we can that a lot of the smaller degree nodes now have PageRank scores close to zero, while high degree nodes have smaller PageRank scores, the slope of the line is much lower now indicating that the relationship is not as strong as before which can be ratified with the correlation scores between the two variables which came out to be 0.6, which is comparatively lower. The PageRank for the median nodes is 12.3% and 8.6% respectively, which are the highest PageRank scores since the random walker always teleports to one of these nodes, and the sum of median PageRank scores is equivalent to the α. The PageRank equation then becomes as shown in equation (16). Where *I(i)* is an indicator function that tests whether *i* is a member of set *M* which contains the nodes that the random walker can teleport to, if it does not belong in it, the PageRank scores are calculated using the vanilla algorithm given by 14.

$$\pi(i) = (1-\alpha)\sum_{j=1}^{|V|}\frac{A_{ji}}{degree(vertex_j)}\pi(j) + \frac{\alpha I(i)}{|M|} \tag{16}$$

$$I(i) = \begin{cases} 1, & i \in M \\ 0, & i \notin M \end{cases}$$

Question 4c

In this question, the goal is to modify the PageRank algorithm to account for the user only teleporting a set of trusted websites, since the original PageRank algorithm assumes that the user is interested in all webpages the same. We propose the following modifications from equation 16 and 14 which are shown in equation 17.

$$\pi(i) = (1-\alpha)\sum_{j=1}^{|V|}\frac{A_{ji}}{degree(vertex_j)}\pi(j) + \frac{\alpha}{|M|}, i \in M \tag{17}$$

$$\pi(i) = \sum_{j=1}^{|V|} \frac{A_{ji}}{degree(vertex_j)} \pi(j) \ , i \notin M \tag{17}$$

Therefore, the PageRank scores are higher for webpages that are in the trusted set *M*, and account for the teleportation. If the user is browsing any other website, their interest in websites outside of *M* would be the same thus, we use the original vanilla PageRank equation since in the websites outside of set *M*.