PROJECT ONE

Random Graphs and Random Walks

Group Members:

Junyu Wang (UID: 405729050)

Zhaoxi Yu (UID: 005432230)

Wayne Zhang (UID: 905845716)

UCLA

ECE 232: Large Scale Social and Complex Networks

Contents

List of Figures & Tables

1 Generating Random Networks

1.1 Create random networks using ER model

1.1.a

We created five graphs with 1000 nodes based on given probabilities {0.003, 0.004, 0.01, 0.05, 0.1}. Then we plot the distribution using the degree.distribution function to plot five graphs below.

We observe binomial distribution in all these graphs:

$$Prob\,(degre = k) = C_{n-1}^{k} p^{k}(1-p)^{n-k-1} \simeq C_{n}^{k} p^{k}(1-p)^{n-k} \tag{1.1}$$

The quantity of C_{n-1}^{k} means that choosing k edges as degree among all n-1 possible choices, and $p^{k}(1-p)^{n-k-1}$ means that k edges are selected, and the remaining n-k-1 are not. Because n is a large number (1000 here), so we can replace n-1 with n. So that we get the equation for binomial distribution.

According to the definition of binomial distribution, $E = np, Variance = n[(1-p)]$. Here is a table to represent the experimental numbers and the theoretical numbers.

Table 1-1

p	*Experimental Mean*	*Theoretical Mean*	*Experimental Variance*	*Theoretical Variance*
0.003	3.138	3	3.122	2.99
0.004	3.886	4	3.724	3.98
0.01	10.110	10	9.689	9.89
0.05	50.076	50	44.775	47.45
0.1	100.996	100	91.261	89.91

As we can see from the above form, the experimental results closely match the expected theoretical results. And with the increasement of the size of sample number, the difference between the two results is decreasing.

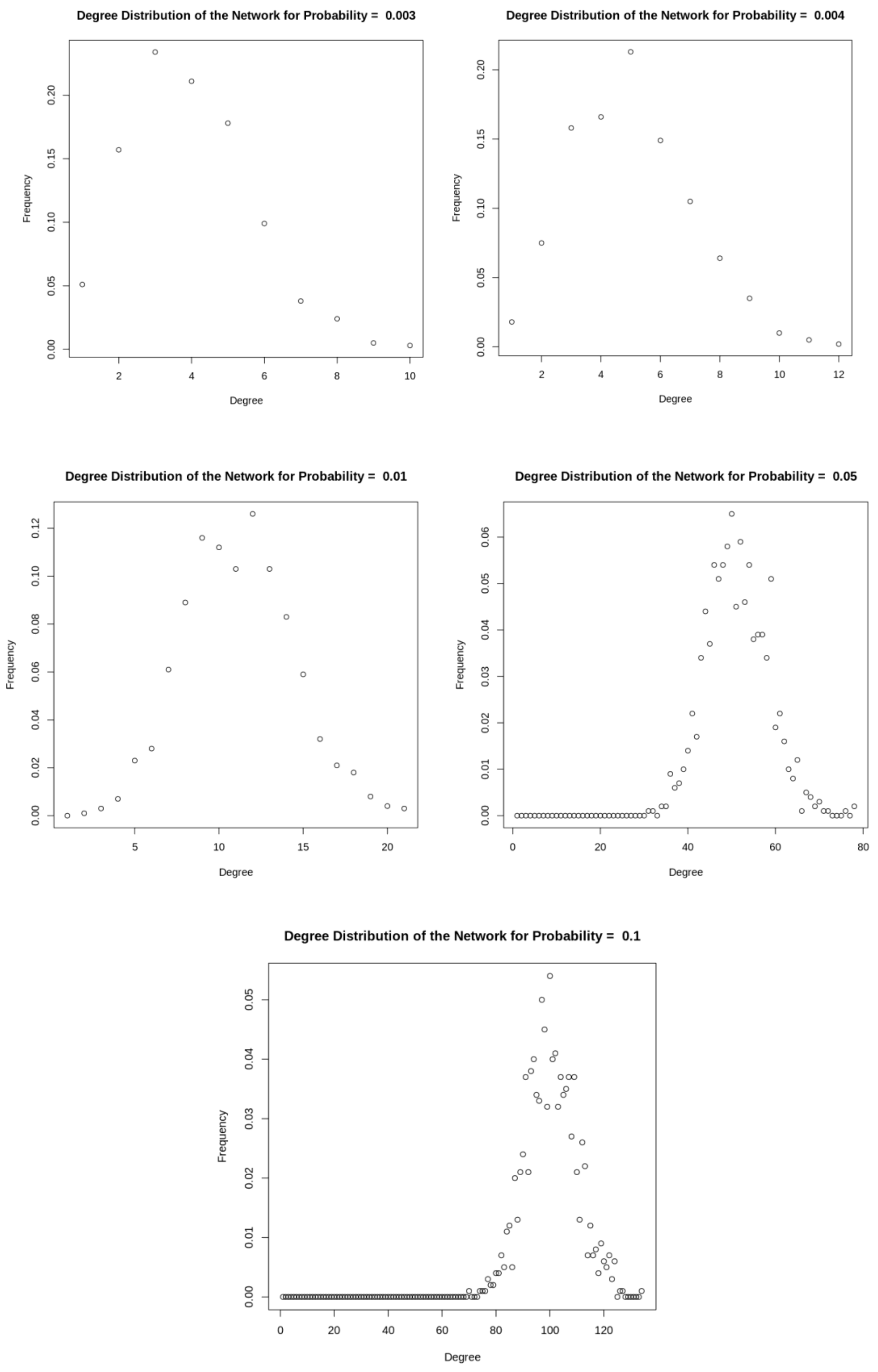

Degree Distribution of the Network for Probability = 0.003
Degree Distribution of the Network for Probability = 0.004
Degree Distribution of the Network for Probability = 0.01
Degree Distribution of the Network for Probability = 0.05
Degree Distribution of the Network for Probability = 0.1
Frequency
Degree

Figure 1-1

1.1.b

According to the code results, not all random realization of ER network is connected. For each probability p, we generate 1000 graphs under that p, and calculate the percentage of connected graph. Experiments results are show in the form below.

Table 1-2

p	*Diameter of GCC*	*Size of GCC*	*Percentage of connected graph*
0.003	14	955	0
0.004	12	984	0
0.01	6	1000	0.961
0.05	3	1000	1
0.1	3	1000	1

1.1.c

We set the upper bound of p to 0.09 and create a sequence of probs from 0 to 0.09 with step of 0.001. At each p step, we generate 100 graphs to record its normalized GCC size. Then we create the graph below.

i. Our definition of emergence is that when the normalization size of GCC's increasing rate is high and it reaches around 50%, we call the GCC starts to emerge. From the graph, we can see that the p is around 0.01. This result matches with the theoretical value which should be $1/n = 1/1000$.

ii. According to the result, when p is around 0.005, the GCC takes up over 99% of the nodes in almost every experiment. This also matches the result.

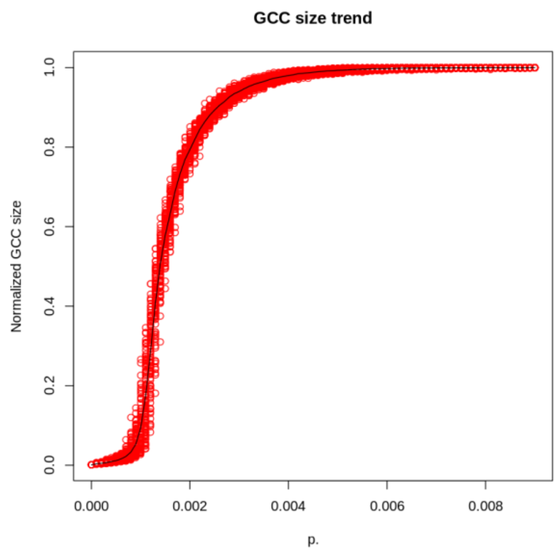


Figure 1-2

1.1.d

In this problem, we define a function to calculate the average of GCC size by generating 100 graphs given p and number of nodes n.

i. We generate a sequence from 100 to 10000 with step of 150. For every number of nodes, we record the number of p and the average size of GCC. We can see that p is decreasing when number of nodes increases because we fix c value here. And the size of GCC increases with the increase of number of nodes. And the trend is around logn.

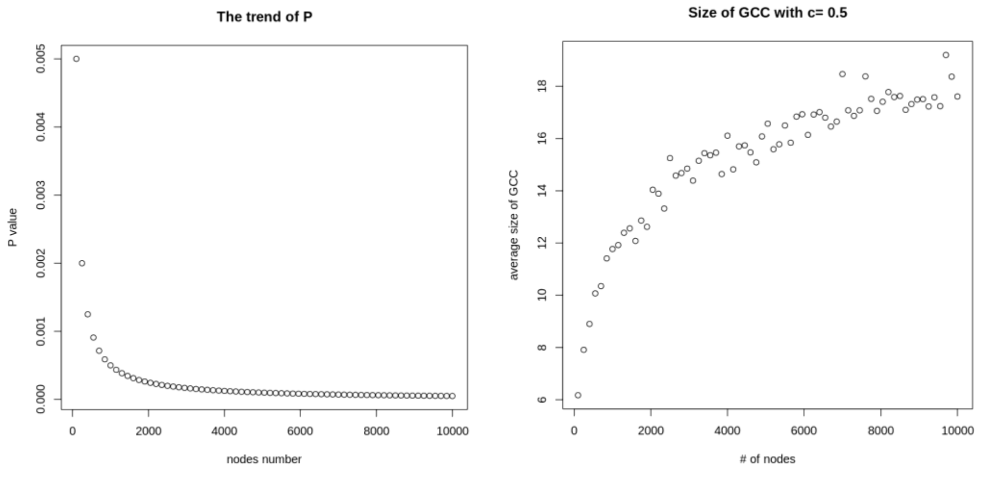


Figure 1-3

ii. Then we repeat this process and set the value of c to 1. We can see the same trend compared to problem i. However, we could see that the average size of GCC is larger than the size where the value of C is 0.5. Also, the relationship between size of GCC and number of nodes is transforming from log to linear relationship.

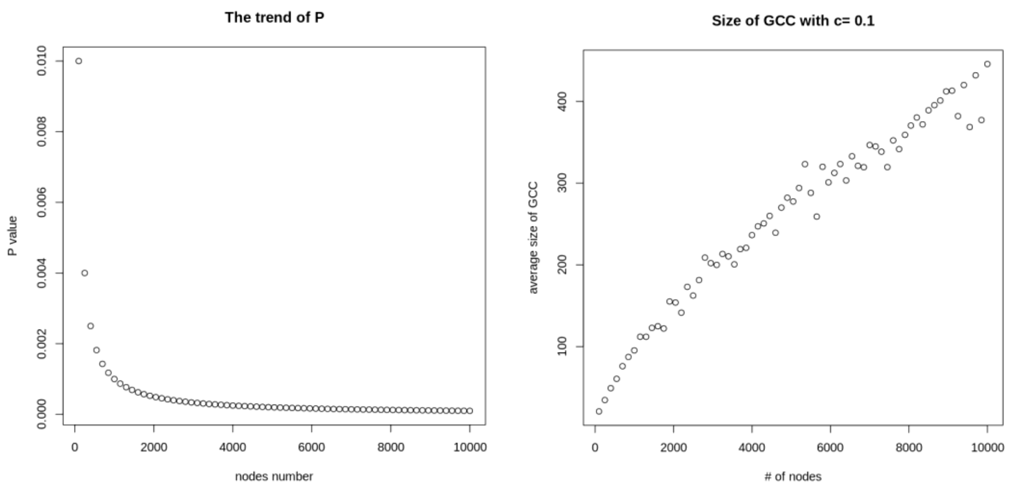


Figure 1-4

iii. Here are the results for c = 1.1, 1.2, 1.3.

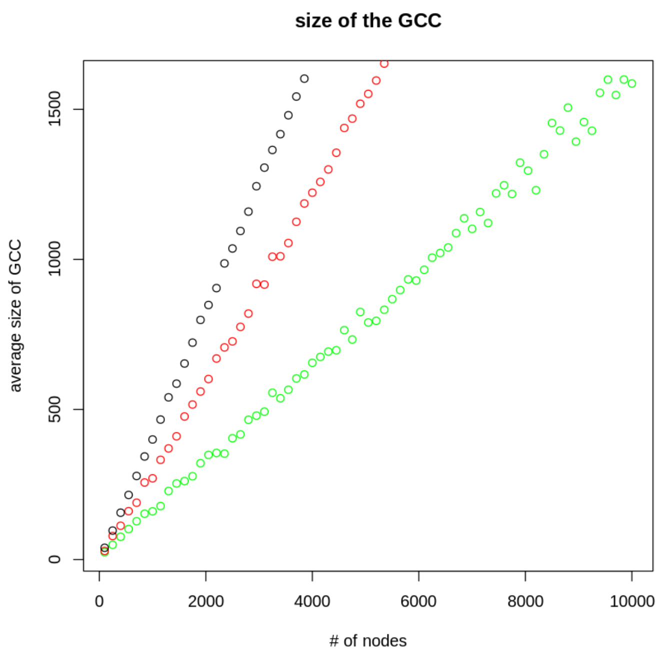


Figure 1-5

iv. From the above we can see that the relationship between number of nodes and average size of GCC is affected by the value of c here.

When the value of c is less than 1, the relationship between them is log relation. It is around $average\_size\_gcc = O(log(numberofnodes))$.

When the value of c is equal to 1, the relation ship between them is transforming from log relation to linear relation. Also, we can see given same number of nodes, the average size of GCC is larger because we have higher number of c.

When the value of c is larger than 1, we can see that they are in linear relationship. And the slope is proportional to c, which means given same number of does, the average size of GCC is larger given larger c.

1.2 Create networks using preferential attachment model

1.2.a

The undirected network using preferential attachment model with 1000 nodes, where each new node attaches to 1 old node is shown below. According to the definition of the preferential attachment model, the network is always connected.

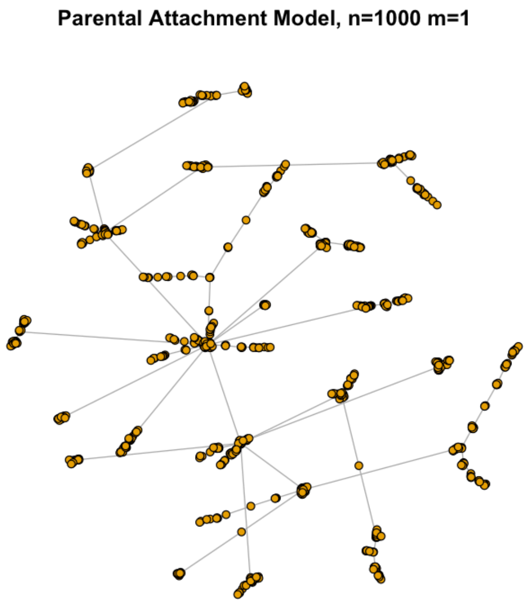


Figure 1-6

1.2.b

The community structure of the network generated in 1.2.a is shown below. The modularity is 0.929322.

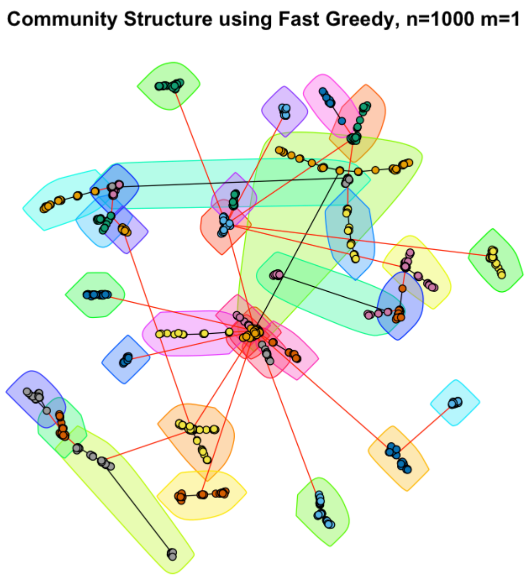


Figure 1-7

1.2.c

In this question, we are asked to create an undirected network using preferential attachment model with 10000 nodes, where each new node attaches to 1 old node. The left figure below shows the generated network, and the right figure below shows its community structure.

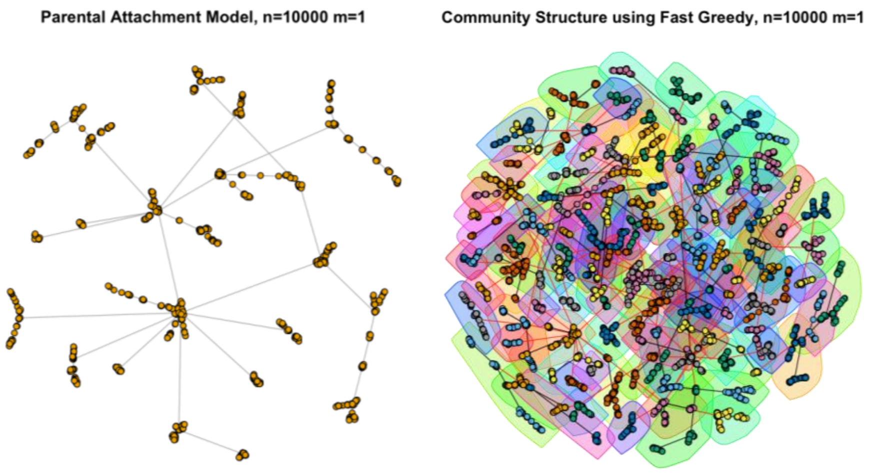

Parental Attachment Model, n=10000 m=1
Community Structure using Fast Greedy, n=10000 m=1

Figure 1-8

The modularity of the network with 10000 nodes is 0.978315, larger than the modularity of the smaller network. This indicates the connections between communities tend to be sparser in larger networks, as a new node is more likely to be attached to nodes with a greater degree in larger networks.

1.2.d

The following figures show the degree distribution for the two preferential attachment models with 1000 nodes and 10000 nodes.

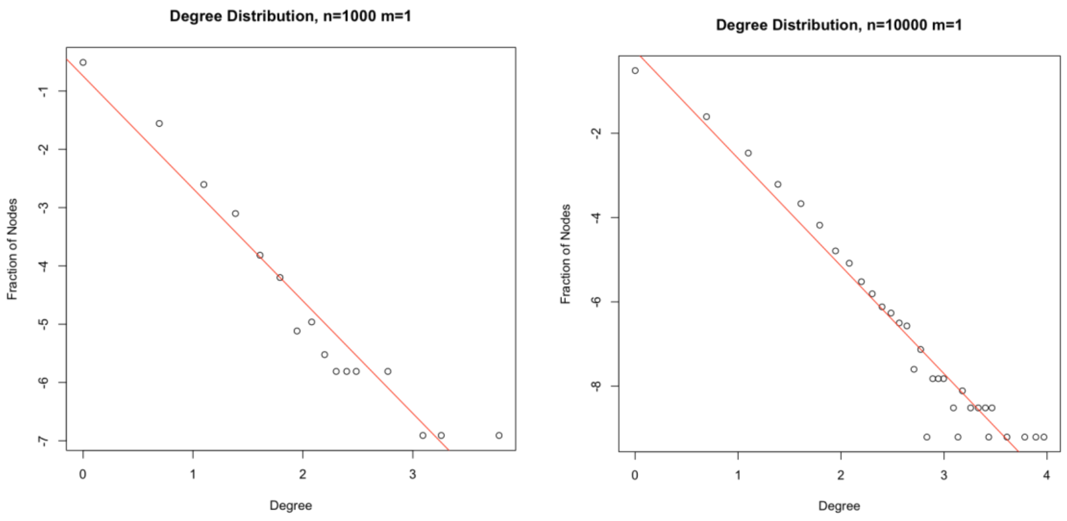


Figure 1-9

The slope of the linear regression line for n = 1000 is -2.069, and the slope of the linear regression line for n = 10000 is -2.4325. The slope is steeper in a larger network.

1.2.e

The log-log degree distributions of neighbor nodes (base e) for both networks are shown below.

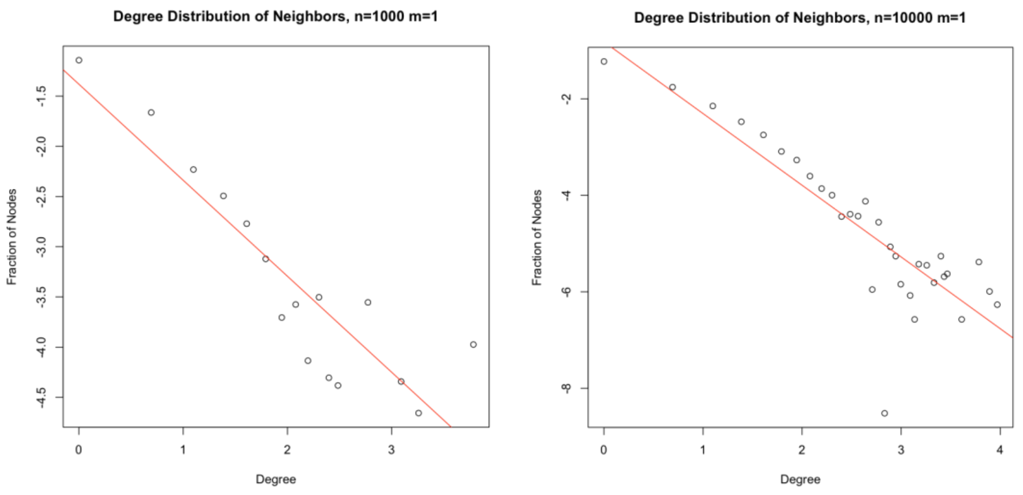


Figure 1-10

From the above figures we can see that the distribution is roughly linear in the log-log scale. The slope for n = 1000 is -0.9318 and the slope for n = 10000 is -1.330. The slopes of neighbor degree distribution are smaller than the slopes of node degree distribution for both n.

1.2.f

The figure below demonstrates the relationship between the age of nodes and their expected degree.

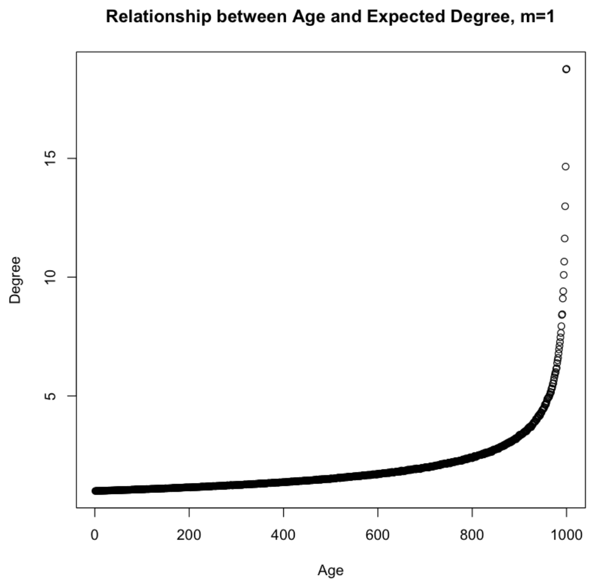


Figure 1-11

From the figure, we can see that the expected degree increases as the age of nodes increases. The reason for the monotonical increase is that newer nodes are more likely to be connected to those older nodes with larger degrees in the preferential attachment model, thus the degree of older nodes increases along each time step.

1.2.g

We repeated the previous parts for m = 2 and m = 5. The undirected networks with preferential attachment are shown below. All networks are always connected.

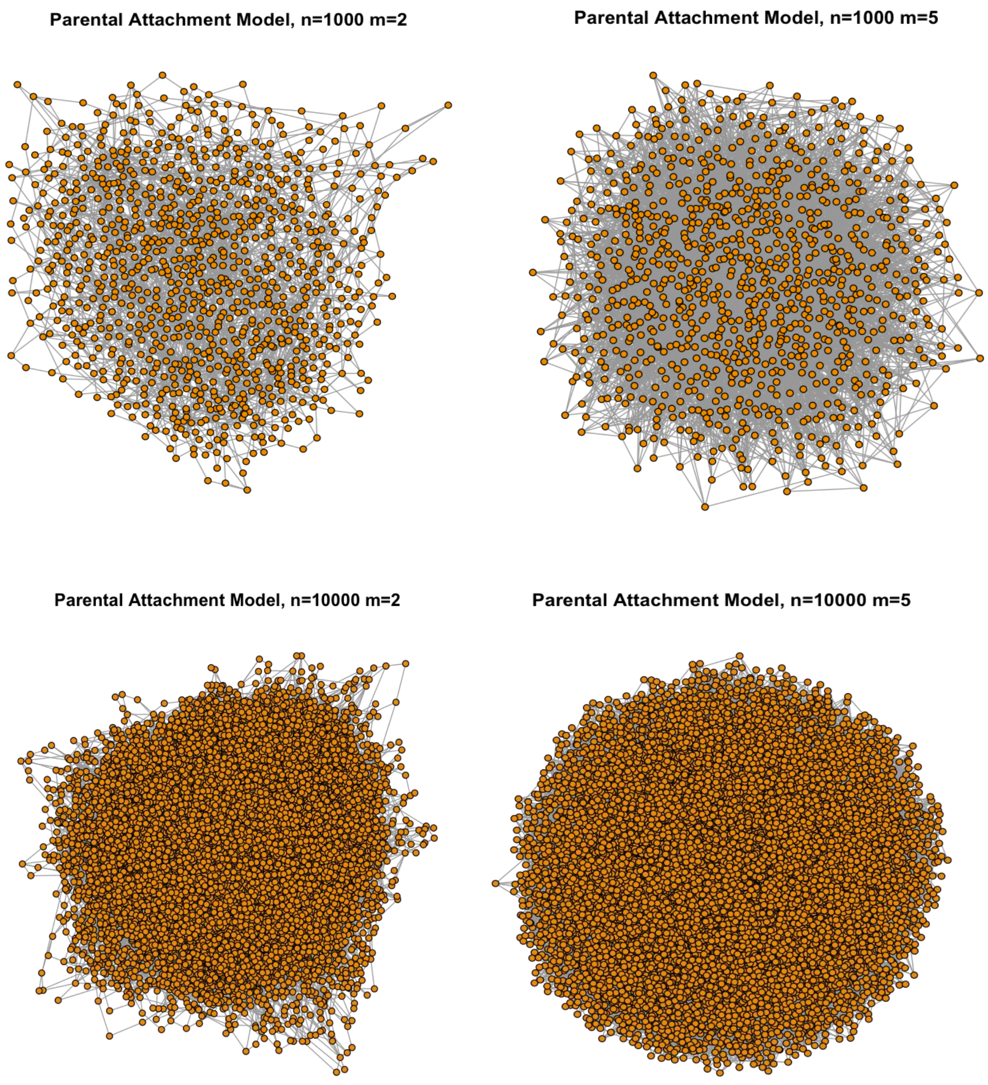

Parental Attachment Model, n=1000 m=2
Parental Attachment Model, n=1000 m=5
Parental Attachment Model, n=10000 m=2
Parental Attachment Model, n=10000 m=5

Figure 1-12

The figures below show the community structures of each network.

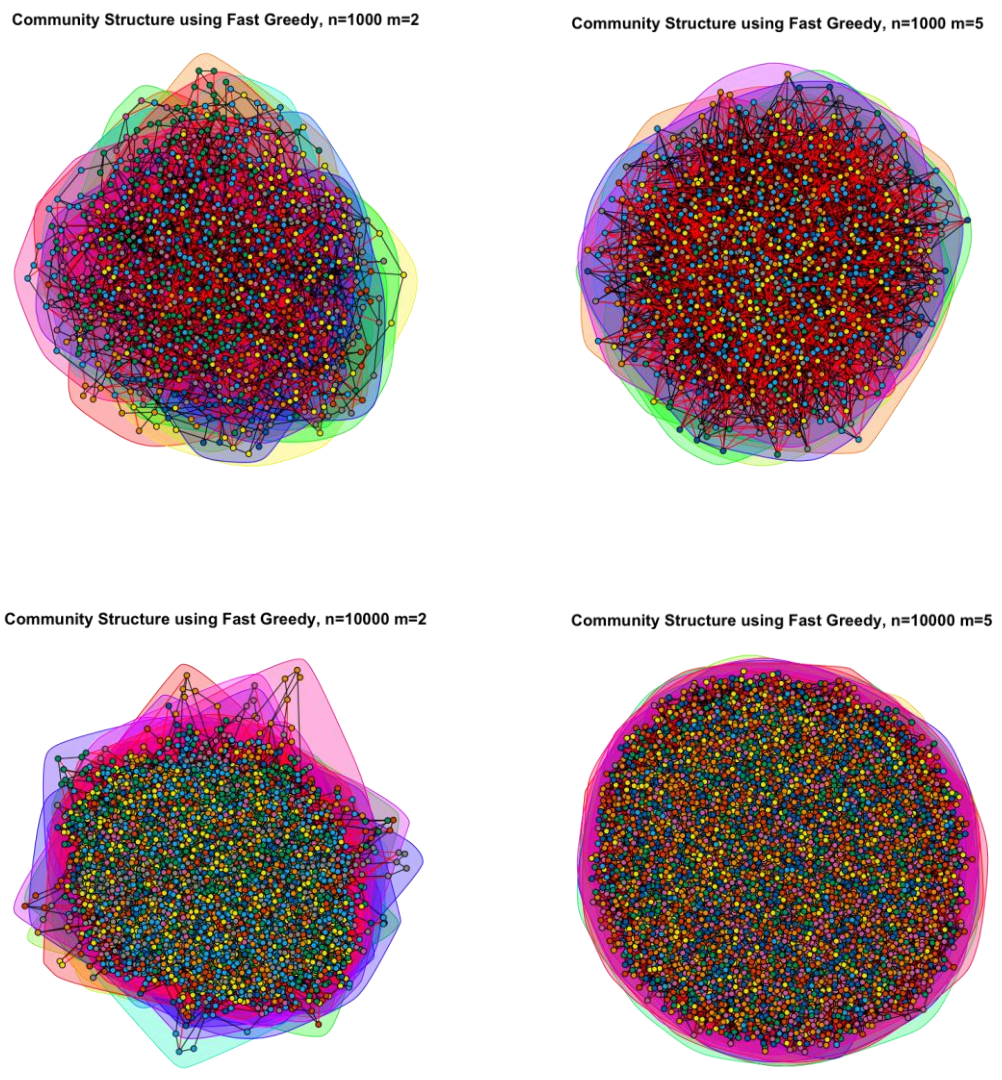

Community Structure using Fast Greedy, n=1000 m=2
Community Structure using Fast Greedy, n=1000 m=5
Community Structure using Fast Greedy, n=10000 m=2
Community Structure using Fast Greedy, n=10000 m=5

Figure 1-13

The table below describes the modularity, the slopes for linear regression of node degree distribution, and neighbor degree distribution of each network, following the figures of degree distribution in the log-log scale.

Table 1-3

n	*m*	*modularity*	*node deg. dist.*	*neighbor deg, dist.*
1000	1	0.929322	-2.069	-0.9318
	2	0.519967	-2.0620	-1.033
	5	0.275619	-1.988	-0.9729
10000	1	0.978315	-2.4325	-1.330
	2	0.530440	-2.3602	-1.3717
	5	0.273123	-2.148	-1.2122

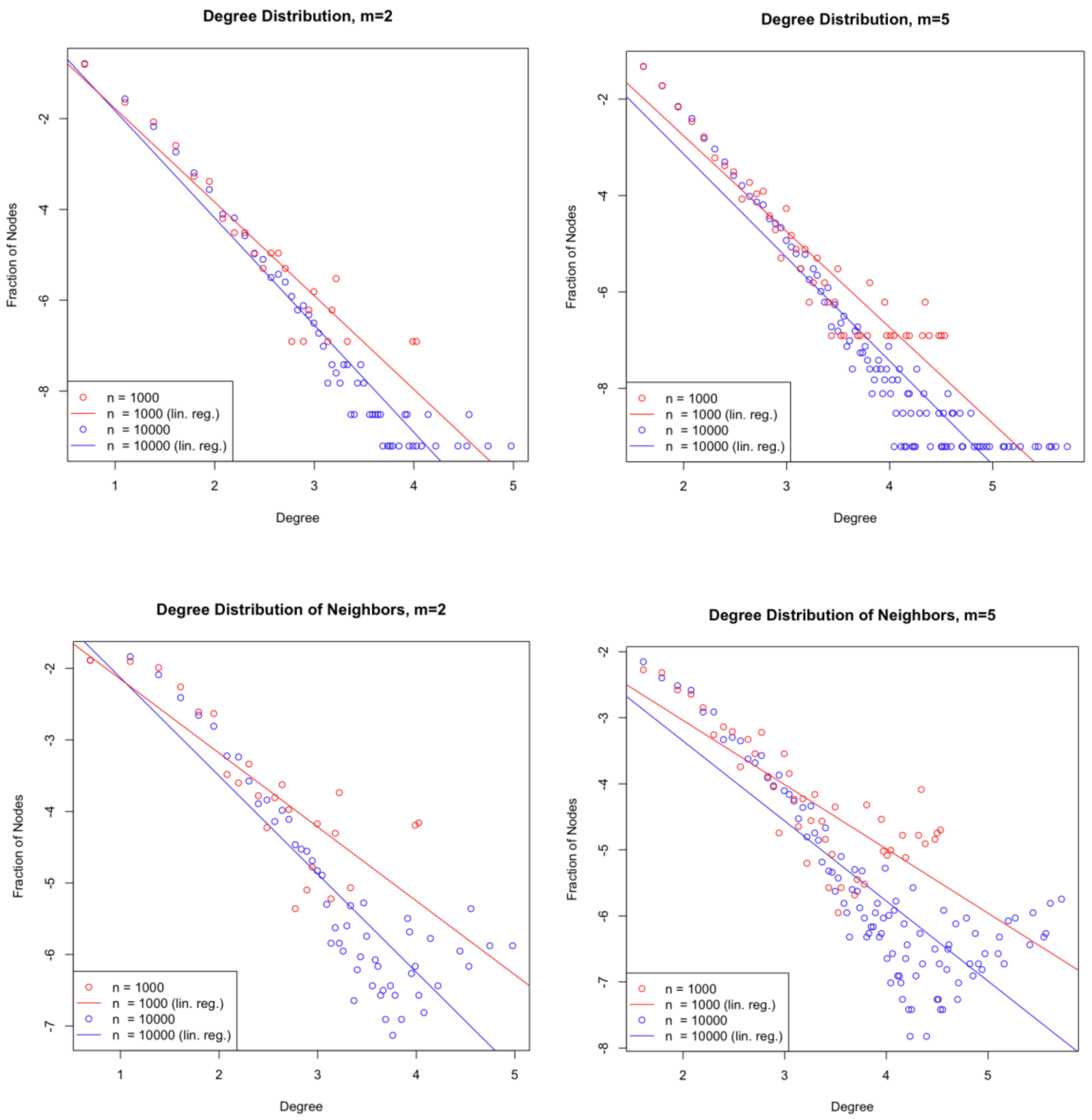


Figure 1-14

From the table and the figures above, we can observe that:

i. For each fixed n, the modularity decreases as m increases, indicating the global sparsity among different communities drops due to larger m.

ii. For each fixed m, the modularity increases as n increases. According to the lecture notes, for the steady state degree distribution of the power law network we have

$$\lim_{k\to\infty} \propto \frac{1}{k^3} \tag{1.2}$$

The power law exponent is closer to 3 for the network with n = 10000 than n = 1000.

iii. The degree distributions in the log-log scale are roughly linear for both node degree distribution and neighbor degree distribution.

The following two figures reflect the relationship between age and expected degree for m = 2 and m = 5. The observation that the expected degree increases as the node gets older applies to both networks. Besides, the expected degree at the same node age increases as m increases, which confirms the equation $Average\ degree = 2mt/t = 2m$ from lecture notes.

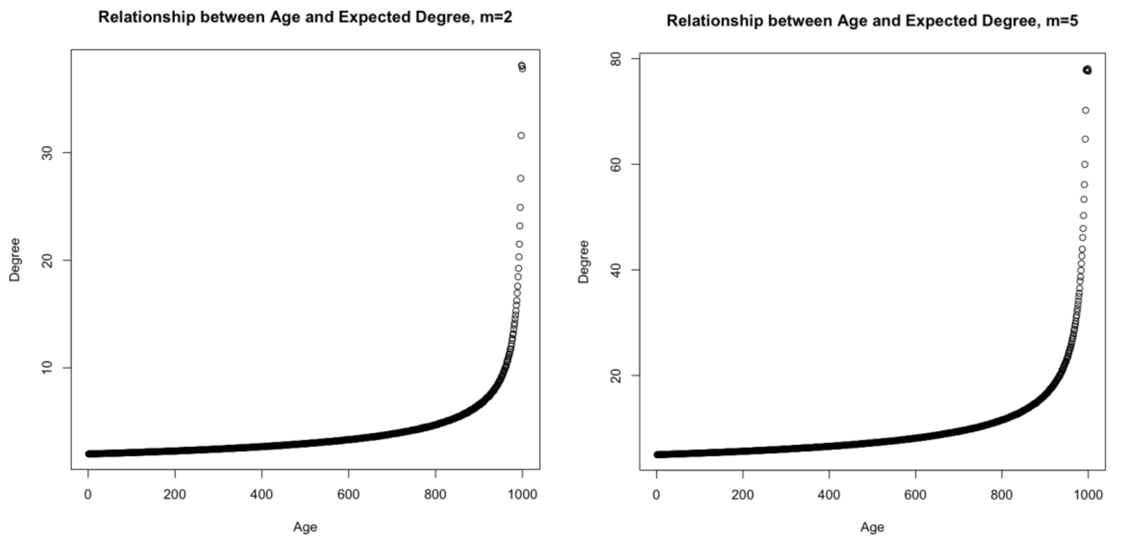


Figure 1-15

1.2.h

The preferential attachment network with n = 1000 and m = 1 and the network with the same degree sequence using stub-matching (simple, no multiple) procedure and their community structures are shown below. The modularities of both networks are 0.932881 and 0.835802 respectively. We can clearly observe that the preferential attachment network is guaranteed to be connected. However, it is very likely that the network generated through stub-matching is not connected. We can also see that there is a considerable number of individual communities around the GCC of the second network, resulting in a smaller modularity.

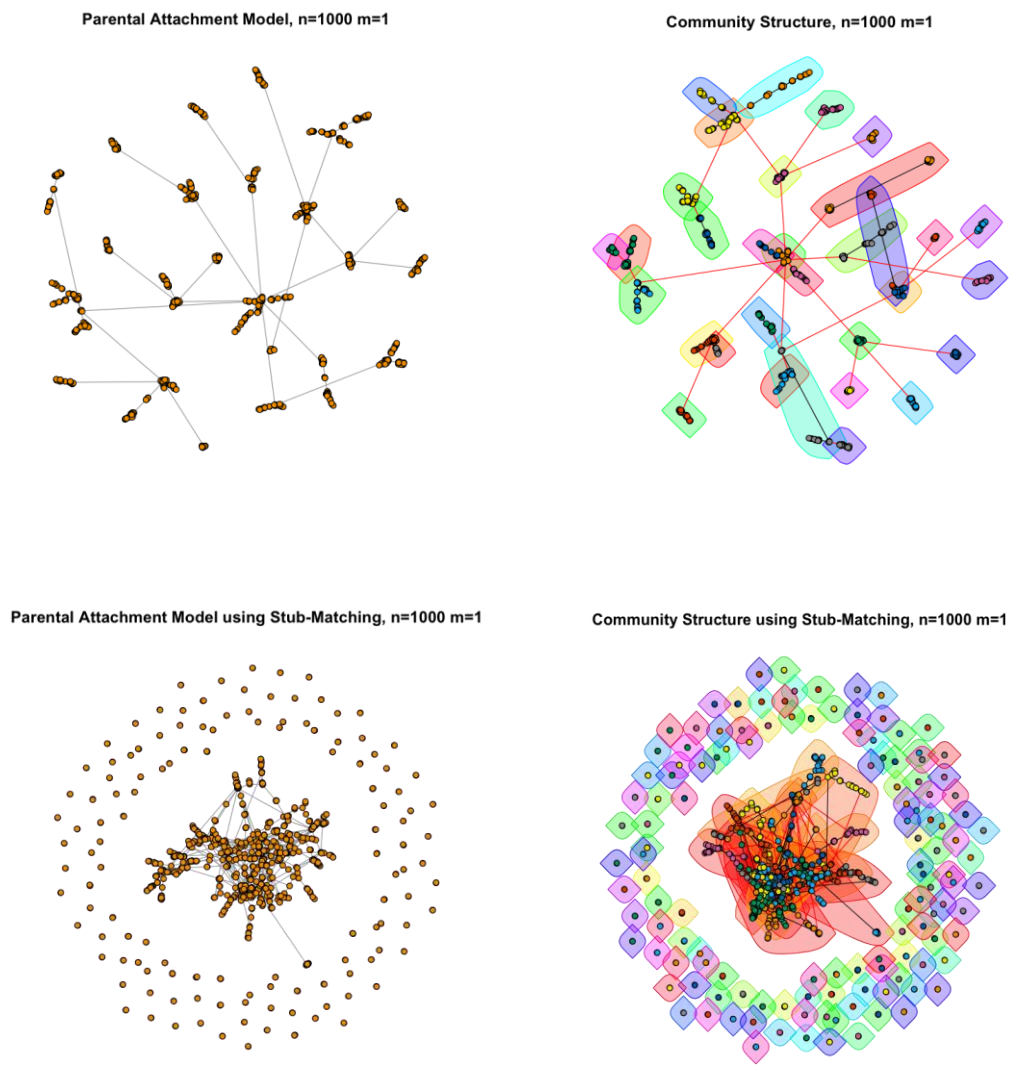

Parental Attachment Model, n=1000 m=1
Community Structure, n=1000 m=1
Parental Attachment Model using Stub-Matching, n=1000 m=1
Community Structure using Stub-Matching, n=1000 m=1

Figure 1-16

1.3 Create a modified preferential attachment model that penalizes the age of a node

1.3.a

We create a network generated by the preferential attachment model that penalizes the age of a node using *sample_pa_age*, passing the parameters stated in the question. We plot the degree distribution of the generated network in log-log scale in Figure 1-17.

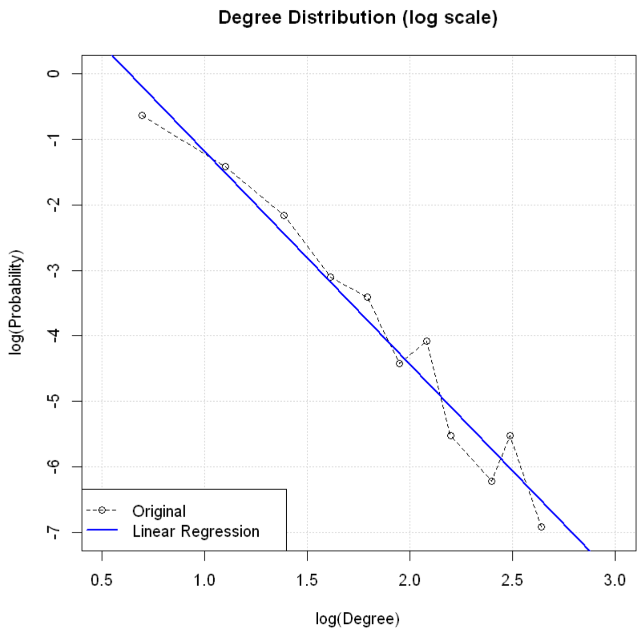


Figure 1-17

The slope or the power-law component obtained from the log-log plot is 3.249. This is in nice agreement with the theoretical value 3.

1.3.b

We use the fast greedy method provided by igraph *cluster_fast_greedy* to find the community structure of the network. The result is shown in Figure 1-18. We find the modularity of the network to be 0.934845. Modularity is a measure of the community structure of the network, a high modularity value means, by penalizing the age of a node during network creation, we construct a network with stronger inner-cluster connections and weaker inter-cluster connections.

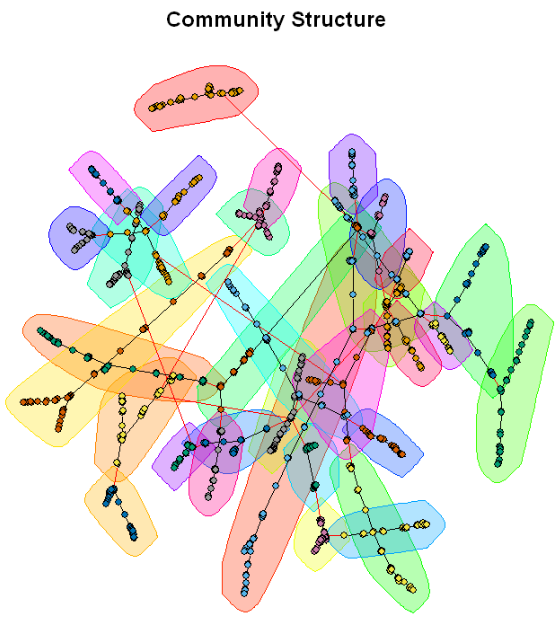


Figure 1-18

2 Random Walk on Networks

2.1 Random walk on Erdos-Renyi networks

2.1.a

We create an undirected Erdos-Renyi graph with n = 1000 and p = 0:01 using *sample_gnp*. The graph is shown in Figure 2-1.

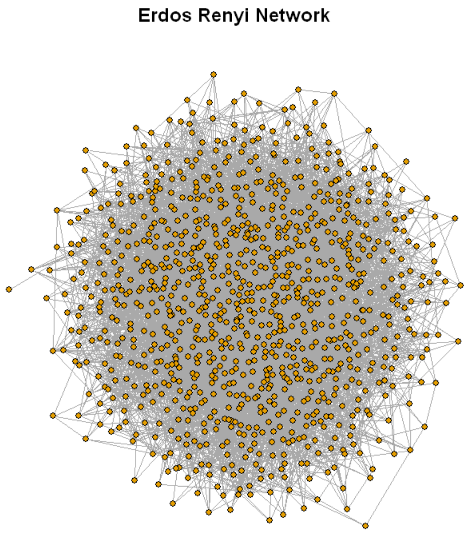


Figure 2-1

2.1.b

We started a random walk at a randomly selected node in the Giant Connected Component (GCC) of the graph and walked for $T=100$ steps. For each step $t \in (0,T)$, we measured the distance $s(t)$ between the 'standing' node from the starting node. We calculated the mean and variance of $s(t)$ over 1000 such random walks to get $\langle s(t) \rangle$ and $\sigma^2(t)$. The results are shown in Figure 2-2. We can see that both the mean and variance increase initially and then converge to a steady state where $t = t_{ss} \approx 10$ and $\langle s(t_{ss}) \rangle \approx 3.3$, $\sigma^2(t_{ss}) \approx 0.42$. This can be explained by the theory of random walk: for any non-bipartite graph, conducting random walk for sufficiently large number of steps provides the steady state node occupancy probability vector π. Without preferential attachment, the walker is likely to end on a random node within the circular region of the average distance from a random starting node.

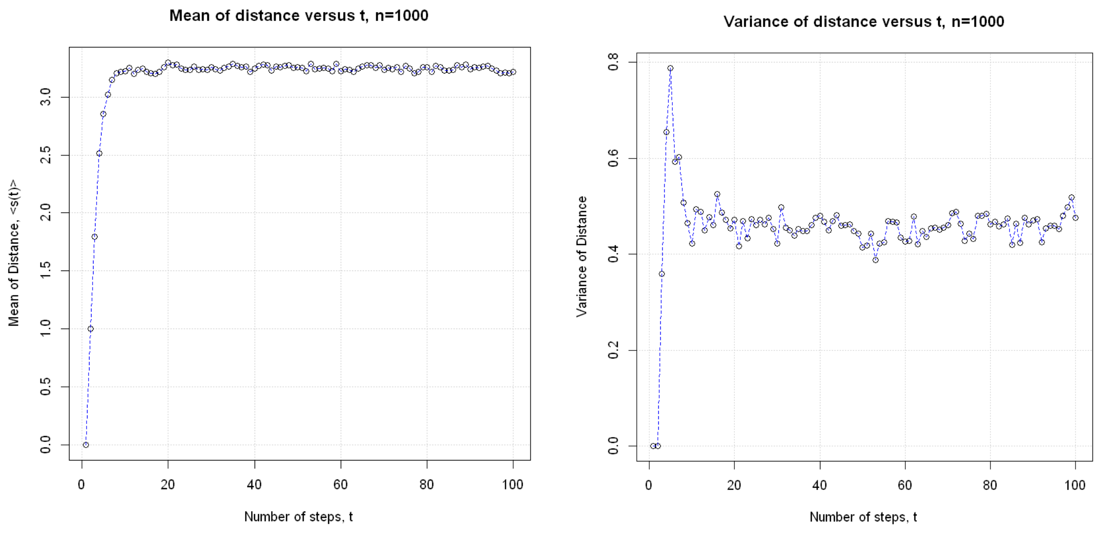


Figure 2-2

2.1.c

We calculate the degree of the end/terminal node for each of the 1000 random walks in 2.1.b and compare it with the degree distribution of the original graph in 2.1.a. The result is shown in Figure 2-3. We can see that both distributions follow a approximately similar binomial distribution, as expected for E-R networks.

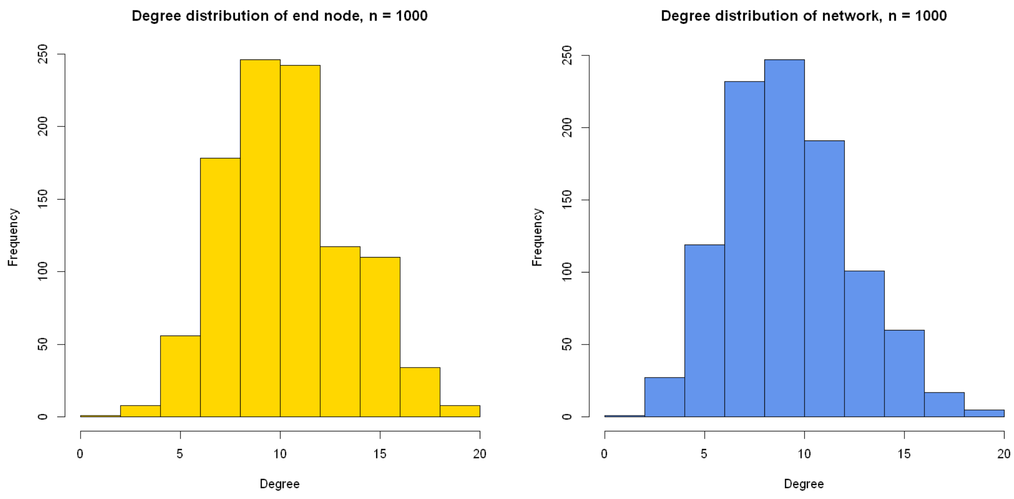


Figure 2-3

2.1.d

We change the parameter in 2.1.b to $n = 10000$ and run the same simulation. The result of $\langle s(t) \rangle$ and $\sigma^2(t)$ are shown in Figure 2-4. We can see that both the mean and variance increase initially and then converge to a steady state where $t = t_{ss} \approx 5$ and $\langle s(t_{ss}) \rangle \approx 2.4$, $\sigma^2(t_{ss}) \approx 0.25$.

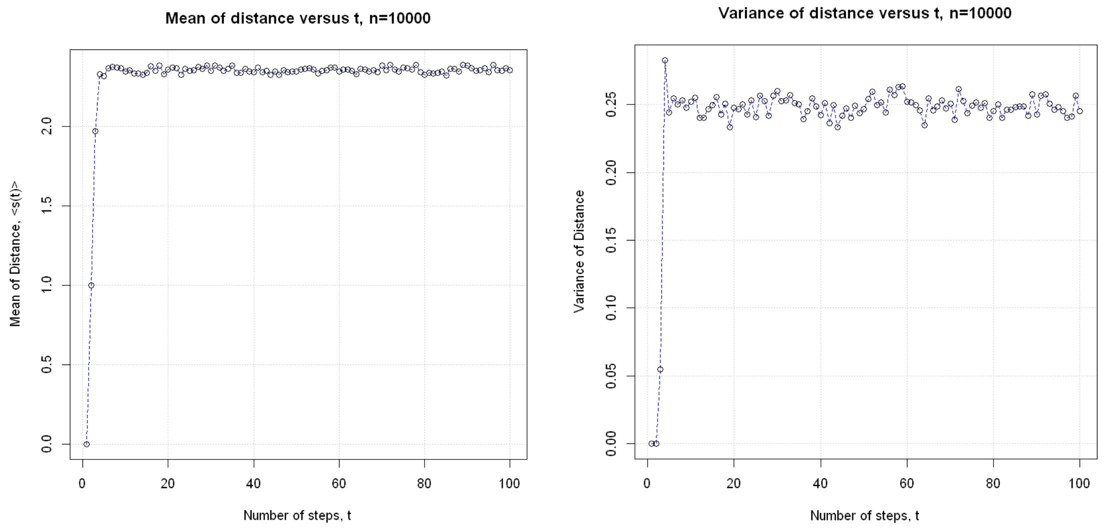


Figure 2-4

We compare the result to 2.1.b in Table 2-1.

Table 2-1

n	*Diameter*	t_{ss}	$\langle s(t_{ss}) \rangle$	$\sigma^2(t_{ss})$
1,000	5	10	3.3	0.42
10,000	3	5	2.4	0.25

While n is increased, all other variables decrease. Specifically, this means that the larger network converges faster to the steady state, with lower and more stable distance to the starting node at convergence. This is reasonable due to the decrease of the diameter of the network. With higher n on construction, the network is more densely connected and thus more concentrated around the starting node, resulting in a lower distance.

2.2 Random walk on networks with fat-tailed degree distribution

2.2.a

The figure of the undirected network through preferential attachment model with n = 1000, m = 1 is generated below.

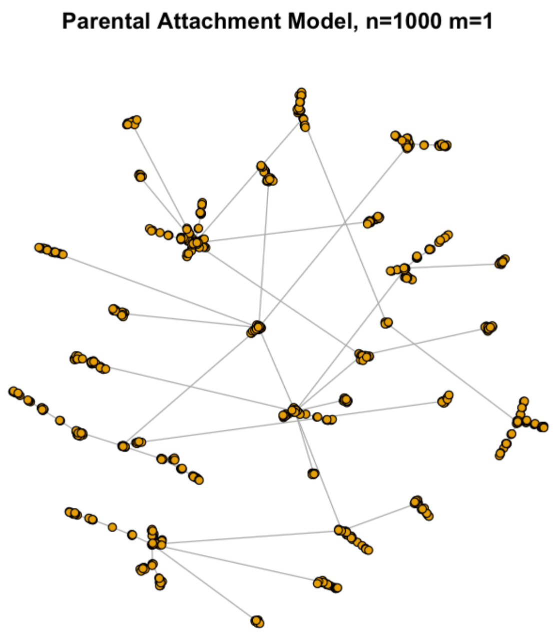


Figure 2-5

2.2.b

We repeated similar steps in last question. The following figure shows the average distance and the variance over t.

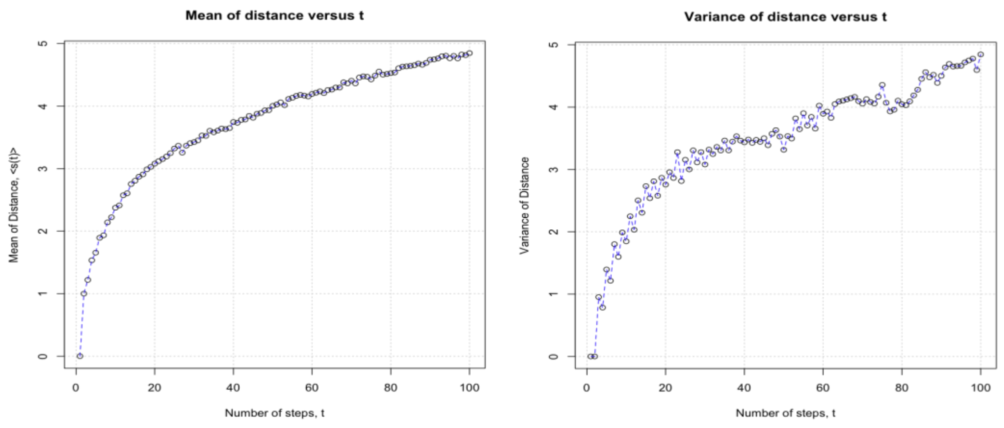


Figure 2-6

2.2.c

The degree distributions of the terminal nodes and the graph are shown below. We observe that the degree distribution corresponds to a power-law distribution.

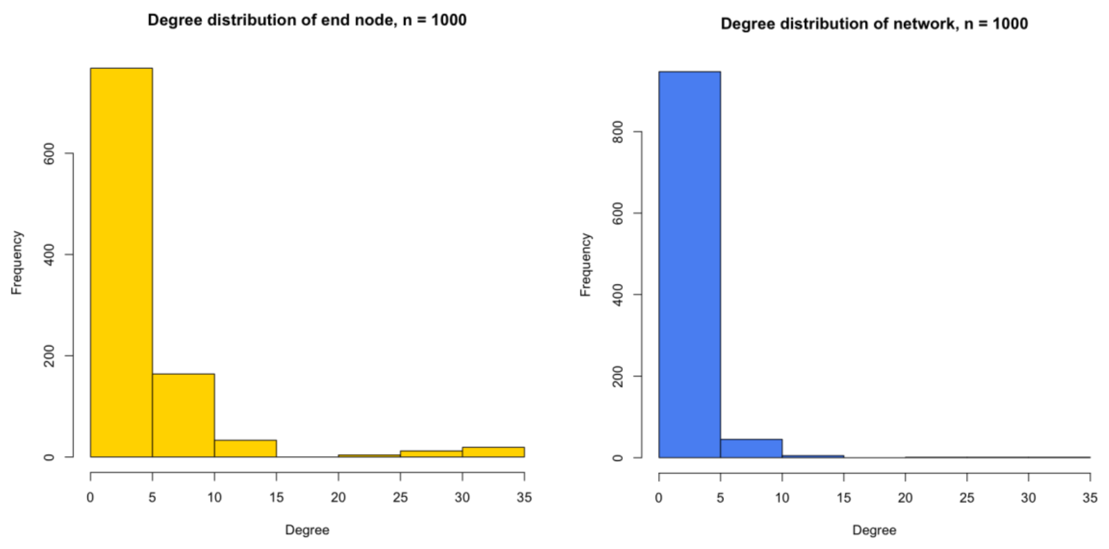
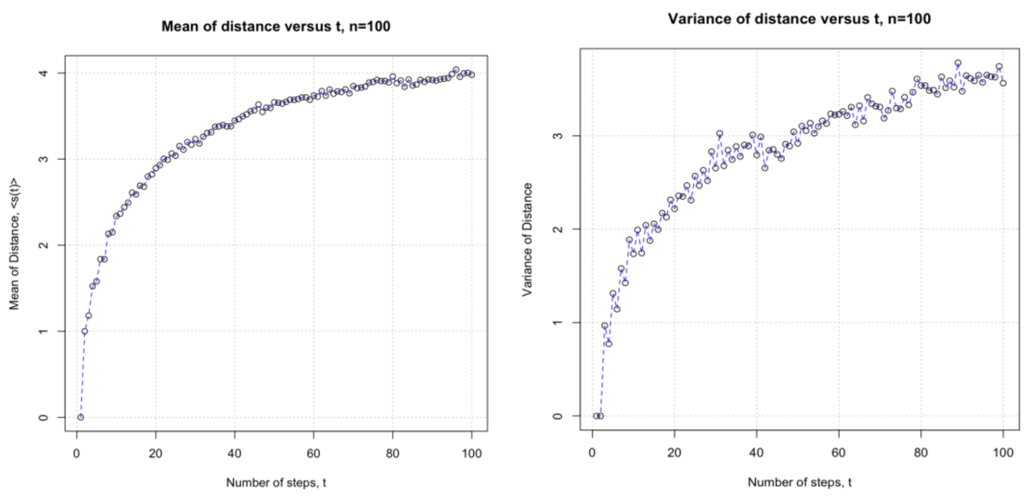
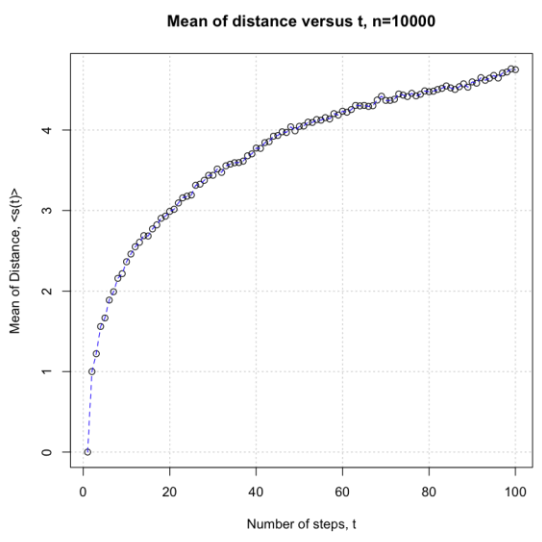


Figure 2-7

2.2.d

The figures below reflect the average distance and the variance of the preferential attachment network with n = 100 and n = 10000 versus t. It was faster for the average distance of n = 100 to converge at a stable value than n = 1000 and n = 10000. Besides, variance for n = 100 fluctuates more than other variances, due to fewer starting nodes to calculate the mean.





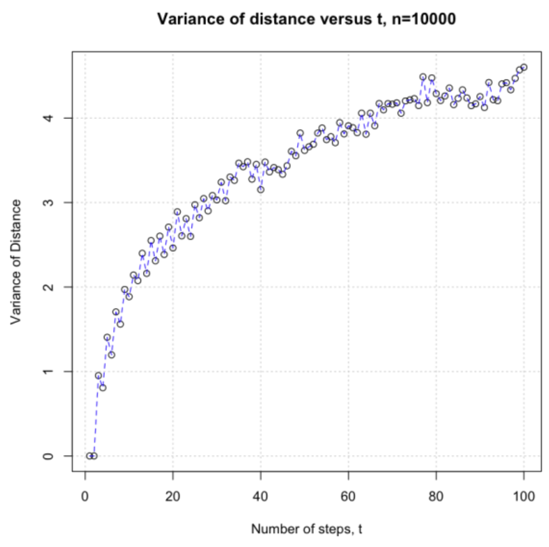


Figure 2-8

2.3 PageRank

2.3.a

In this program, we perform random walk on networks to find the frequency of arriving at each node. **The conclusion is that the probability of arriving at a node is proportional to the degree of that node.** We draw the graph indication relationship between the probability of arriving nodes and degree of nodes.

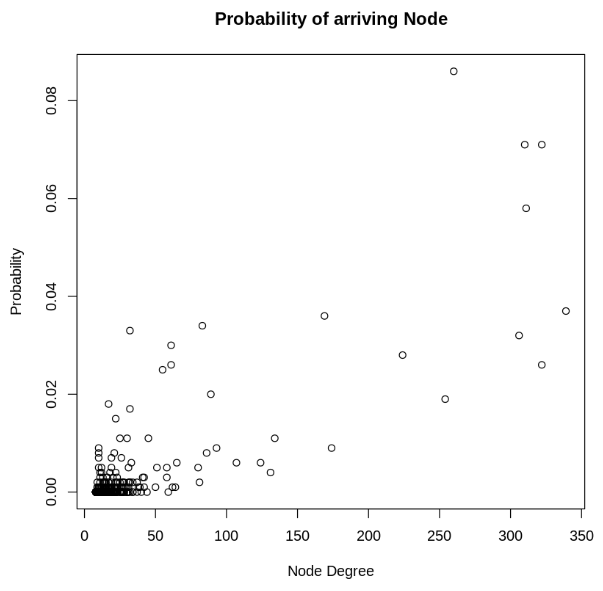


Figure 2-9

After constructing the original graph and adding a shuffled graph to it, we perform a 1000 times random walk on this graph with steps of 250. Every end of a random walk, we record its destination. Then we construct a map data structure, where the key is the node, and the value is the node's arriving frequency.

We can easily figure out that the node with high possibility of arriving also has a high node degree.

2.3.b

In this problem, we add teleportation probability to the algorithm. The result is that the probability of arriving at a node is also **proportional** to the degree of the node.

The difference is that we can see the maximum of frequency (probability) here is around 0.06 while in the previous problem the maximum probability is around 0.08 on average. This is because we perform a teleportation here, which means the next transition or next walk will have an alpha probability to perform a random transition. This prevents our iterator from stucking at the nodes with higher degree. So, we can find that the node area with lower degree is denser than the previous problem. Also, the slope of probability over node degree also is smaller than the previous problem because we have higher possibility to reach low degree node with teleportation than the situation without it.

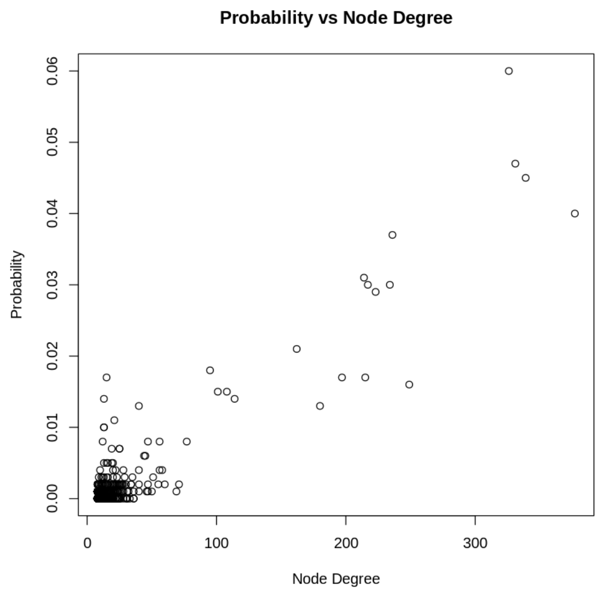


Figure 2-10

2.4 Personalized PageRank

2.4.a

Here, we could get the page rank result as the guidance to do random walk. In the teleportation process, instead of going to other nodes with equal probability, our personalized page rank algorithm will select the next node based on our page rank heuristic. Here is the result of this algorithm.

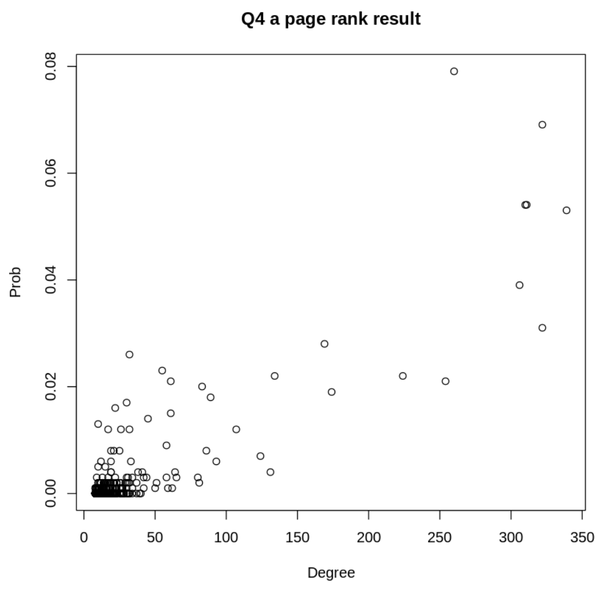


Figure 2-11

Compared to the result in 2.3.a, we could find the result are similar. We can see the distribution if the node degree is similar and the slope prob over degree is similar. There is no big difference between them.

In other words, this also indicates that our random walk without teleportation can find the node with higher page rank. And its algorithm ability is equivalent to teleportation with given page rank. Also, because alpha is just 0.15 here, so it does not dominate the whole process. If we have a higher alpha, we will need high probability of higher degree node.

2.4.b

For this problem, we do the exact same operation compared with previous problems except setting probability of transition to make it transit to two nodes with 0.5 probability respectively in the teleportation stage. Here is the result we get.

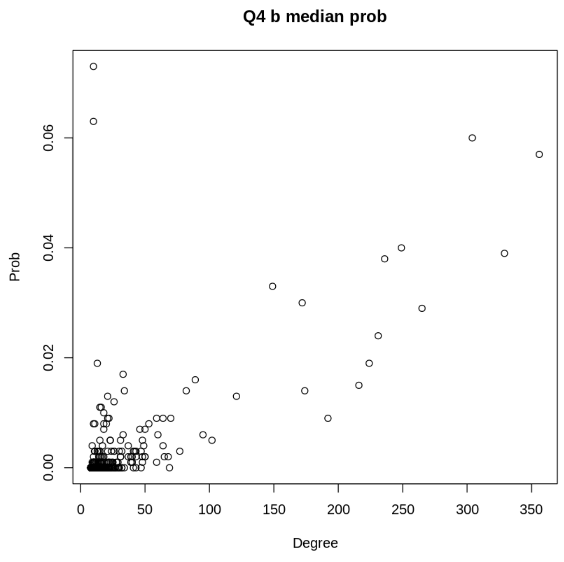


Figure 2-12

We can see that the distribution is sparser compared to previous cases. Here are two main differences. 1) We can see there are some points with low degree but high probability. This is because we also let the random walker transit to two points, and the points next to these two points may have the higher probability to reach. So random walker may arrive at the point with low degree. 2) We can see that the slope of prob over degree is smaller than the previous examples. This is because, in the teleportation stage the random walker will not always go to the points that tend to have higher page rank. However, they are always going to two designated points. This will affect the overall result.

2.4.c

According to the lecture notes, here is our original and normal pagerank equation:

$$PR(i) = (1-\alpha)\sum_{j=1}^{n}\frac{1}{k(j)}A_{ji}PR(j) \tag{2.1}$$

Then the equation with teleportation should be the following equation. This equation takes in account for the teleportation page rank.

$$PR(i) = (1-\alpha)\sum_{j=1}^{n}\frac{1}{k(j)}A_{ji}PR(j) + \alpha\sum_{j=1}^{n}\frac{1}{|T|}PR(j) \tag{2.2}$$

Then if we consider the effect of this self-reinforcement, which means that people tend to be interested only in trusted web pages and not all pages. So that we need to only enable interested page to be calculated into the page rank calculation equation. So that we get the following equation:

$$PR(i) = (1-\alpha)\sum_{j=1}^{n}\frac{1}{k(j)}A_{ji}PR(j) + I_{\text{trusted}} \odot \sum_{j=1}^{n}\frac{1}{|T|}PR(j)$$
$$I_{\text{trusted}} = \begin{cases} 1 & \text{page } i \text{ is a trusted web page} \\ 0 & \text{page } i \text{ is not a trusted web page} \end{cases} \tag{2.3}$$