ECE 232E - Project 1

Random Graphs and Random Walks

Zhechen Xu (805030074), Yufei Hu (404944367), Qi Zeng (204946904), Liangkun Zhao (204947003)

April 22, 2018

Part 1: Generating Random Networks:

1. Create random networks using Erdös-Rényi (ER) model

(a)

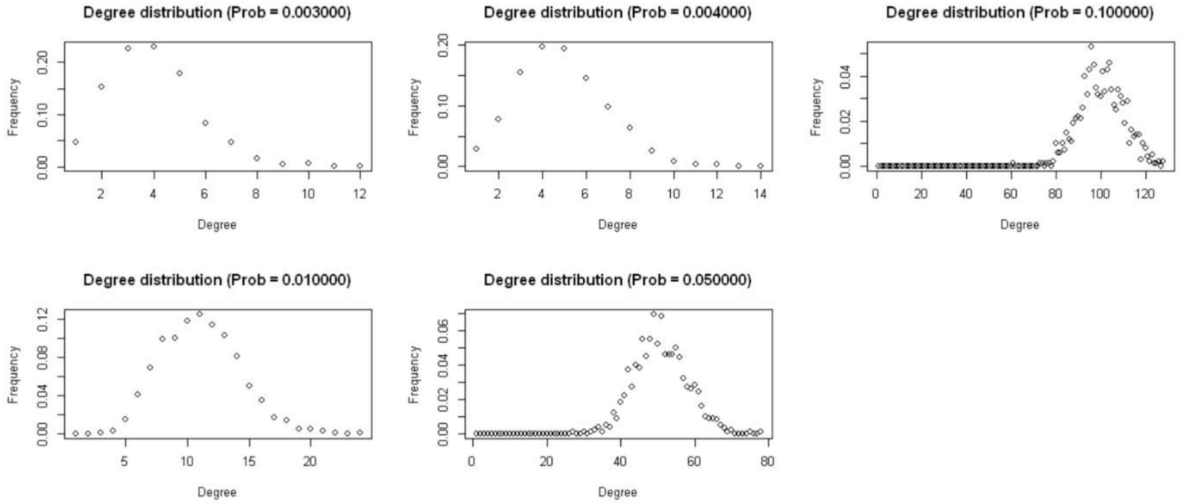

Figure 1: Network degree distribution with different probabilities

Table 1: Statistics of degree distribution with different probabilities

Probability	Mean	Variance
0.003	3.052	2.816
0.004	4.122	4.051
0.010	10.114	10.327
0.05	50.330	46.858
0.1	99.498	92.557

Erdös-Rényi (ER) model is used to generate 5 random networks with different edge connection probabilities. Based on the observation in Figure 1, the distribution of the network degree follows a Bernoulli distribution. The reason is due to the equation depicted as follows:

$$p(\deg = k) = \binom{n}{k} \cdot p^k \cdot (1-p)^{n-k} \tag{1}$$

As the number of nodes or the product of node number and probability increases, the Bernoulli distribution will gradually approach Poisson distribution following the equation below:

$$p(\deg = k) = \frac{e^{-np} \cdot (np)^k}{k!} \tag{2}$$

Based on the two equations above, our team has drawn the theoretical degree distribution as follows:

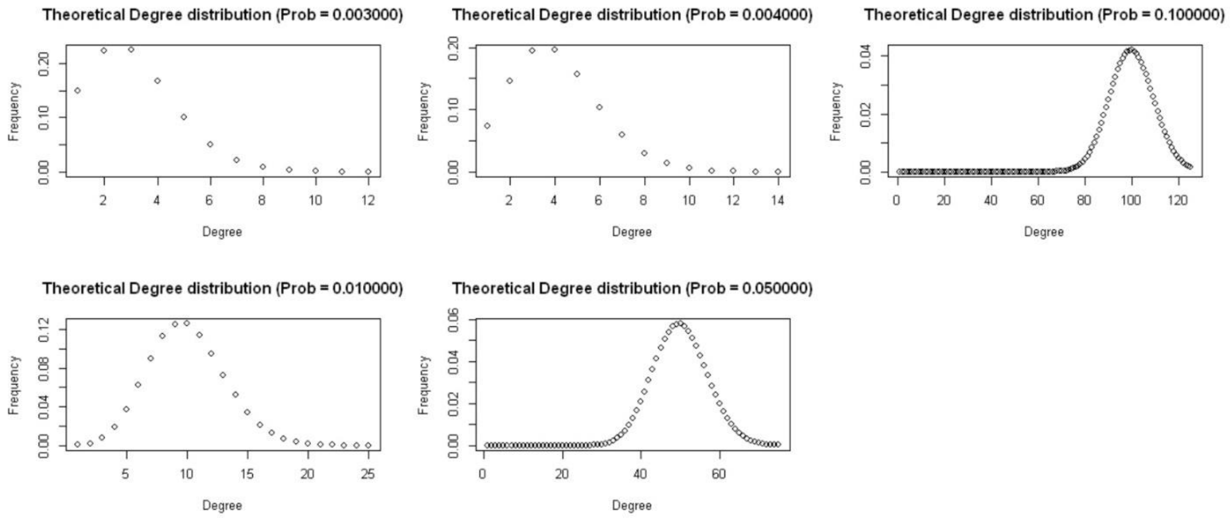

Figure 2: Theoretical Network degree distribution with different probabilities

Table 2: Theoretical statistics of degree distribution with different probabilities

Probability	Mean	Variance
0.003	2.999796	2.989512
0.004	3.999712	3.981884
0.010	9.999586	9.897282
0.05	49.980009	47.947600
0.1	99.416601	131.201305

Comparing Figure 1 with Figure 2 and Table 1 with Table 2, it is found that the simulation result matches the theoretical results pretty well.

(b)

Table 3: Connection of the five networks

Probability	Connected	GCC Details
0.003	No	Diameter=12, Number of nodes=940
0.004	No	Diameter=11, Number of nodes=979
0.010	Yes	N/A
0.05	Yes	N/A
0.1	Yes	N/A

Based on the Erdős and Rényi's analysis, the size of GCC and the network connection conditions are described as follows:

$$\begin{aligned} &\text{When } np < 1,\ E(|GCC|) = O(\ln n) \\ &\text{When } np = 1,\ E(|GCC|) = O(\sqrt{n}) \\ &\text{When } np > 1,\ E(|GCC|) = \varepsilon(np) \cdot n \end{aligned} \tag{3}$$

In our generated networks, *np* is always bigger than 1. *ε(np)* gets close to 1 when *np* increases. This analysis matches our simulation perfectly. When *np* is big enough, the generated network is connected. The theoretical probability for a connected network is around 0.01 when *np* is one order of magnitude bigger than 1.

(c)

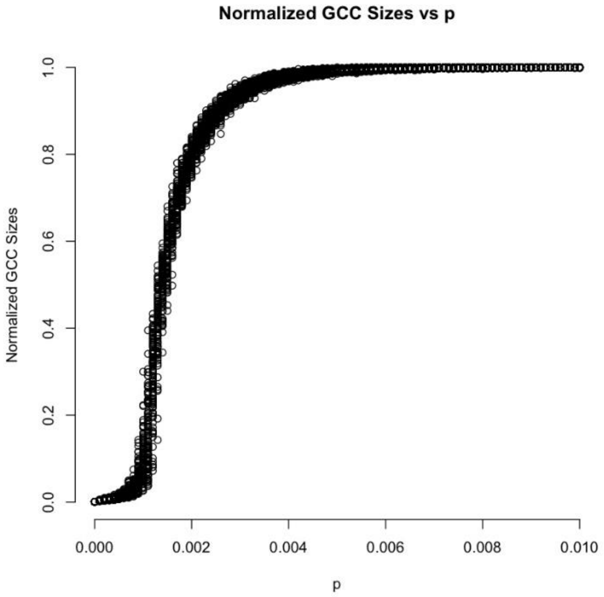

Figure 3: Normalized GCC size against probability

Based on Equation 3, the normalized GCC size is considered to stay at 0 when *np* is 1. When *np* is actually 1, the theoretical scale of the normalized GCC size is around $n/\sqrt{n}=0.03$, therefore the emergence point is defined as when the normalized GCC size reaches $1-n/\sqrt{n}=0.97$. Based on our simulation, the emergence edge connection probability is 0.0036. This number matches the experiment results listed in Table 3 very well.

(d)

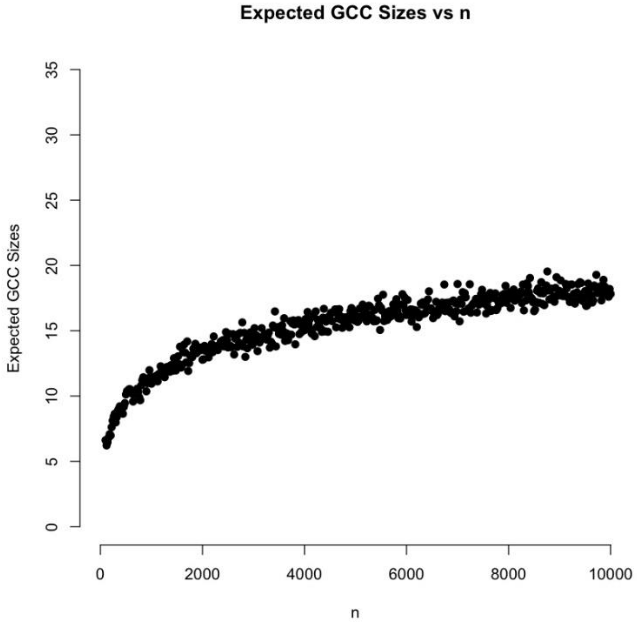

Figure 4: Expected GCC sizes when c is 0.5

Based on Figure 4, it is observed that the expected GCC size gradually increases when *n* gets bigger. According to Equation 3, when *c* is below 1, the expected GCC size is around the scale of *ln(n)*. This is reason why the increase of GCC size is rather slow.

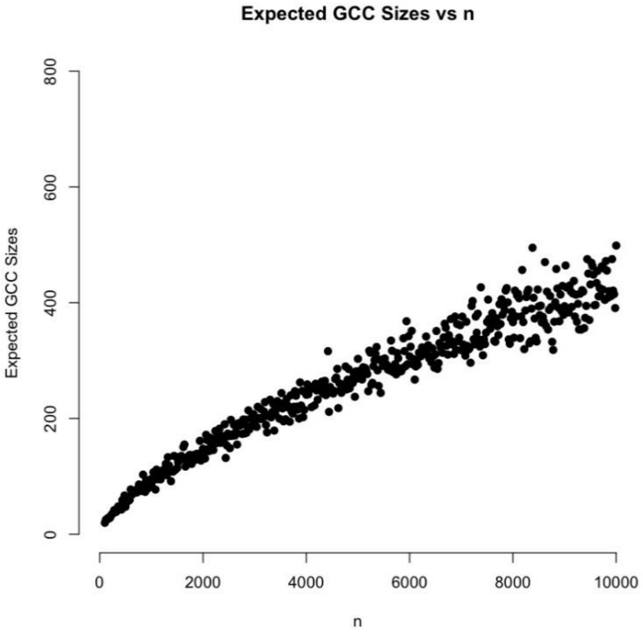

Figure 5: Expected GCC sizes when c is 1

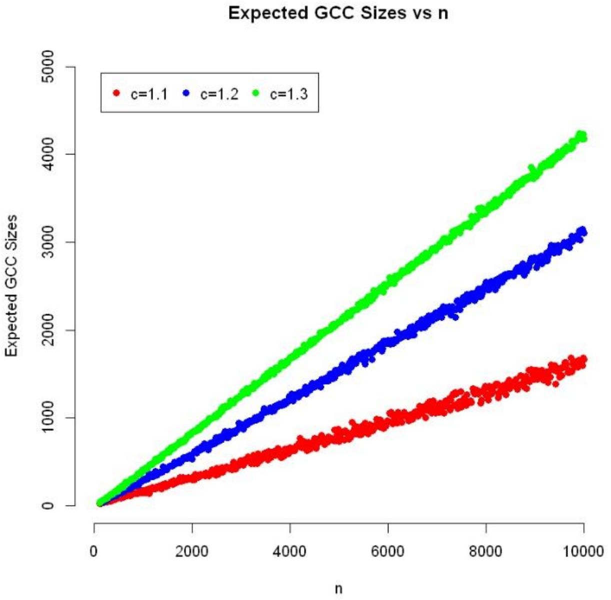

Figure 6: Expected GCC sizes when c is 1.1, 1.2, and 1.3

Figure 5 and Figure 6 matches with Equation 3 perfectly as the former one increases at *O(log(n))* while the latter one increases linearly.

2. Create networks using preferential attachment model

(a)

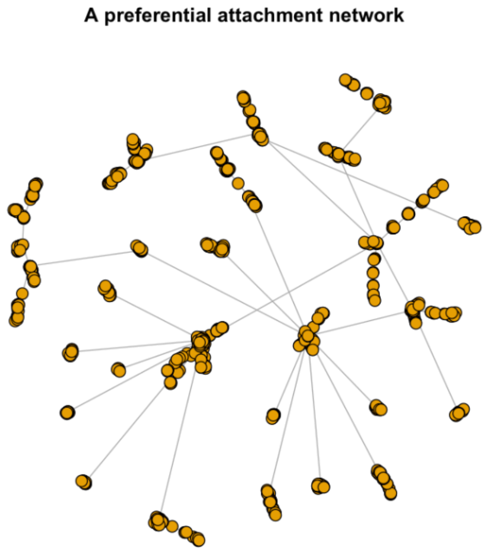

Figure 7: A network generated by preferential attachment model

A network generated through preferential attachment model is always connected. For preferential attachment model, a new node is going to pick an existing node to attach at each step during modeling procedure.

(b)

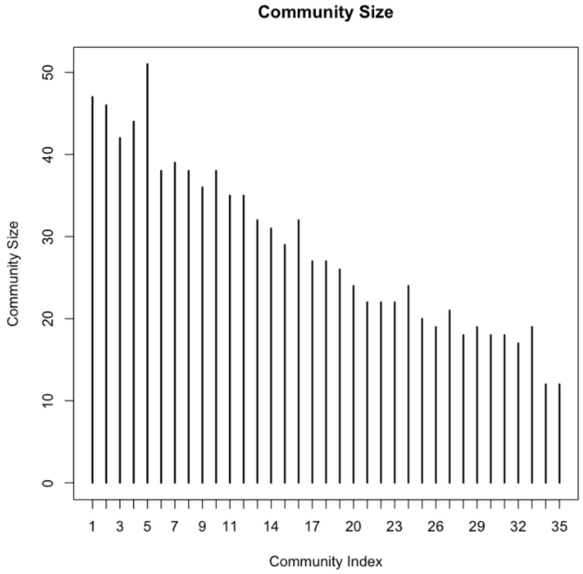

Figure 8: Community size vs Community Index

There are 35 communities and the modularity is 0.933 which means that the network has dense connections with other nodes in its community but sparse connection with nodes in a different community.

(c)

The larger network's modularity is 0.978, which is larger compared to that of the smaller network. A preferential attachment network follows power law and the rule "richer get richer". During the modeling process, the community size gets bigger. The largest community in this network has 154 nodes, and the connection between nodes within a community get denser in the larger preferential attachment network.

(d)

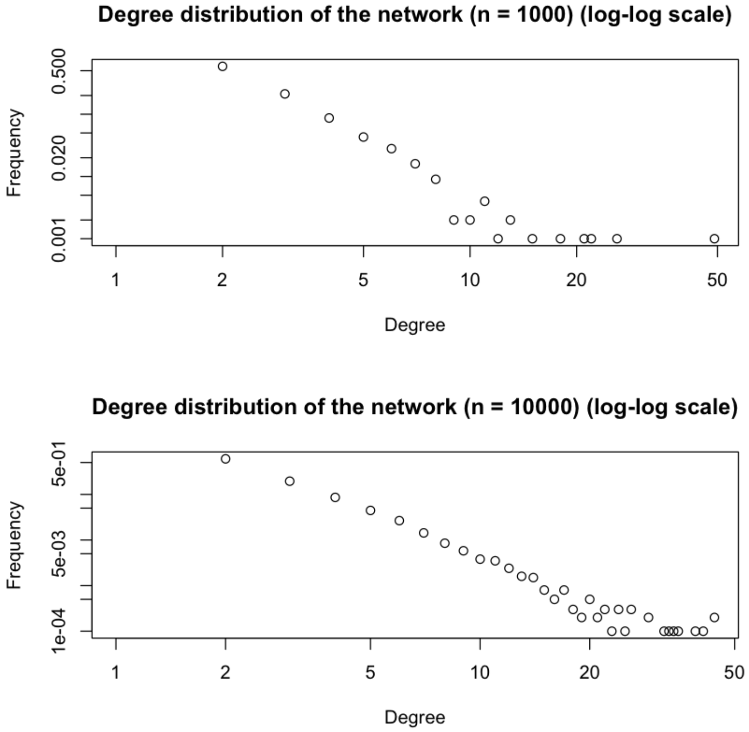

Figure 9: Distribution of a preferential attachment network (log-log scale)

The slope of each plot is estimated by using linear regression model. When there are 1000 nodes in the network, the slope is -2.385. When there are 10000 nodes in the network, the slope is -3.069.

(e)

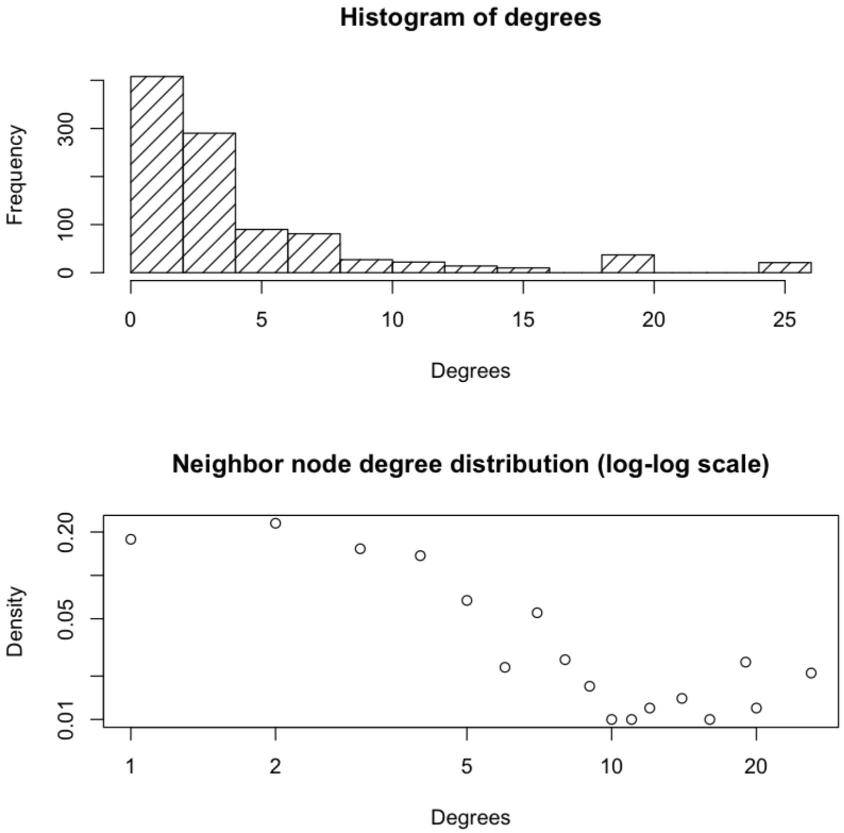

Figure 10: Random picked neighbor node degree distribution

The result of the experiment is shown below. From the plot, we can find out that the network follows power's law ~ x^2, whose pattern is very similar to node distribution of a preferential attachment network.

(f)

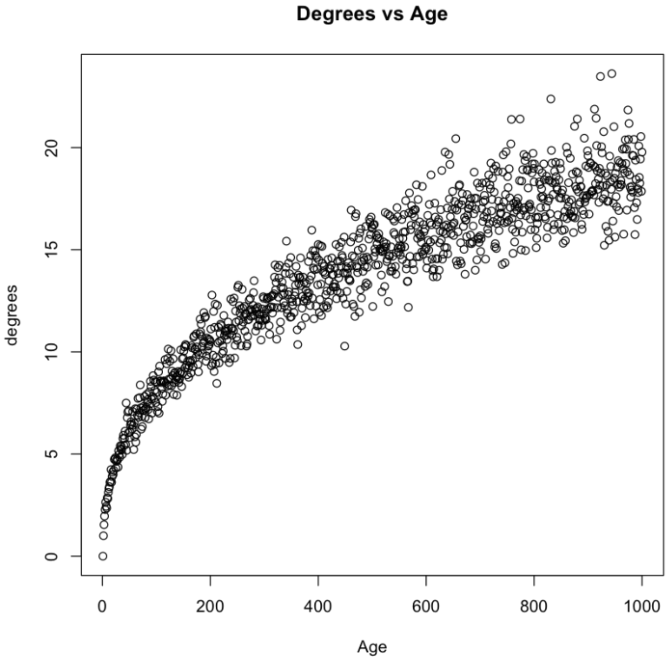

Figure 11: Node degree vs Node age

(g)

Modularity for m = 1 is 0.927.

Modularity for m = 2 is 0.523.

Modularity for m = 5 is 0.275.

From the mathematical definition of modularity,

$$Q = \frac{1}{2e} \cdot \sum_{i,j} \frac{A_{ij} - k_i \cdot k_j}{2e} \cdot \delta(c_i, c_j)_{i,j} \tag{4}$$

where e is the number of edges. The value of modularity is inverse-proportional to the number of edges. Thus, as m gets larger, more edges will be added into network at each step. Also, as new nodes are added into the network, the division between different vertex types get worse.

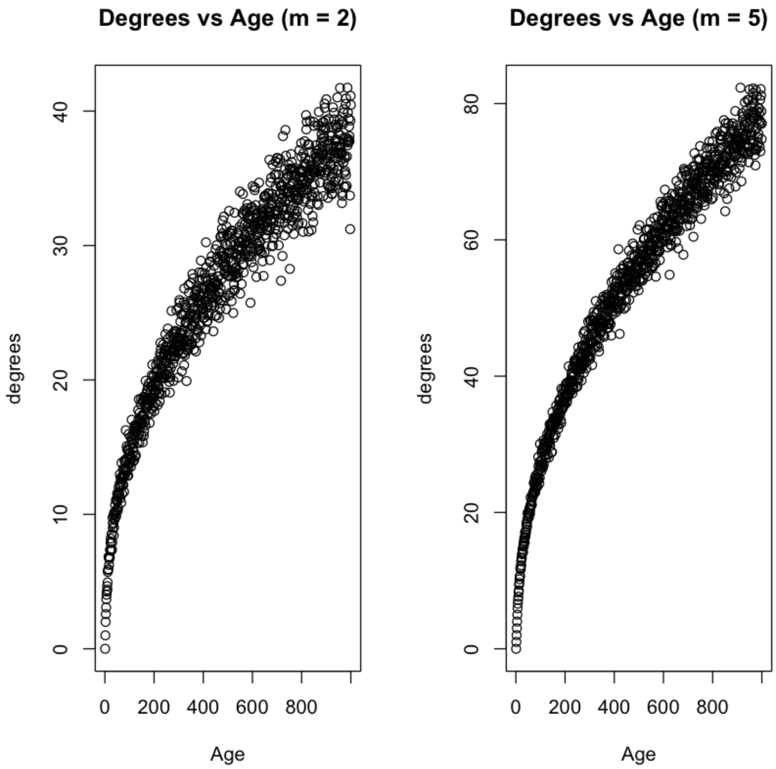

Figure 12: Node degree vs Node age

(h)

The modularity of the original network is 0.9336348.

The modularity of the new network is 0.8385052.

Form graphs of two networks below, we can tell that the separations of different vertex types in the original network are larger than the separations in the new network.

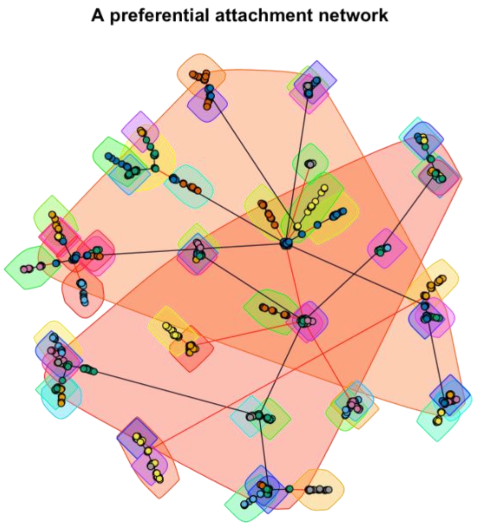

Figure 13: A network generated by preferential attachment method

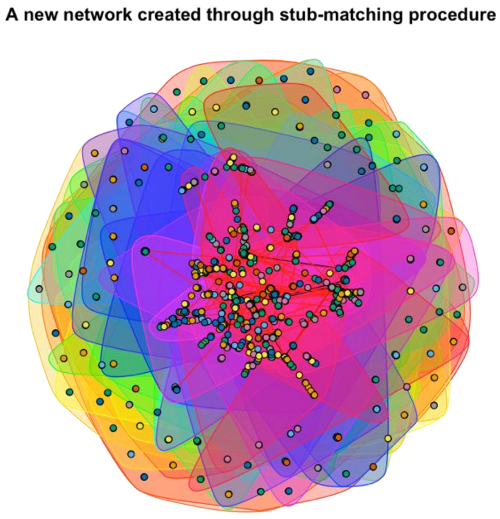

Figure 14: A new network generated through stub-matching procedure

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

(a)

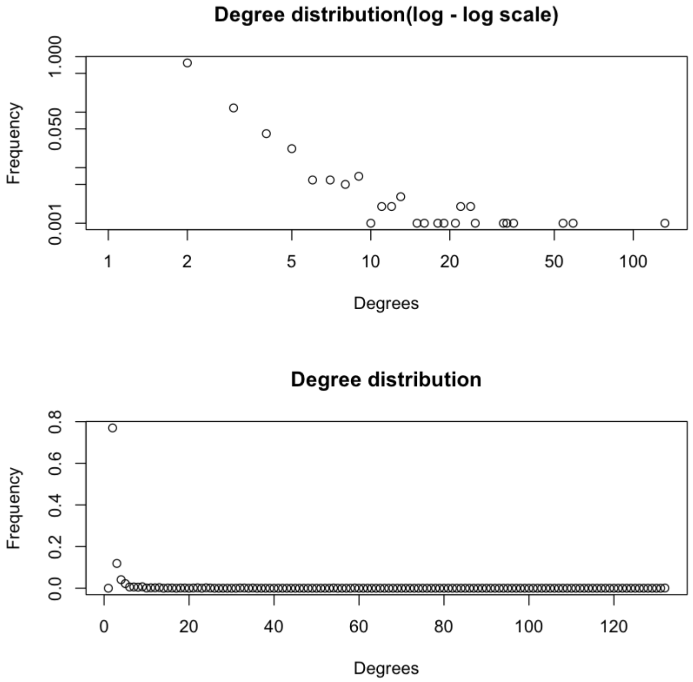

Figure 15: Degree distribution of a preferential-attachment network

The value of power law exponent is 1.569.

(b)

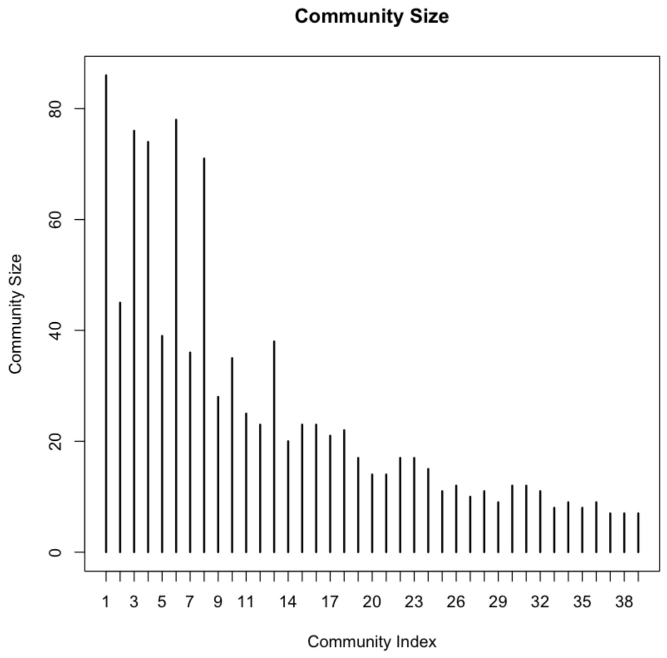

Figure 16: Community size vs Community index

The modularity of the network is 0.916. We can tell that the network is well clustered within a community but highly separated across communities.

Part 2: Random Walk on Networks:

1. Random walk on Erdös-Rényi networks

(a)

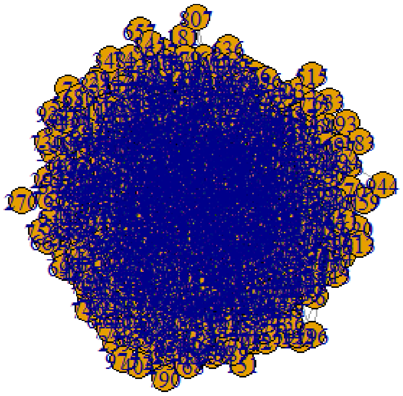

Figure 17: Undirected random network

(b)

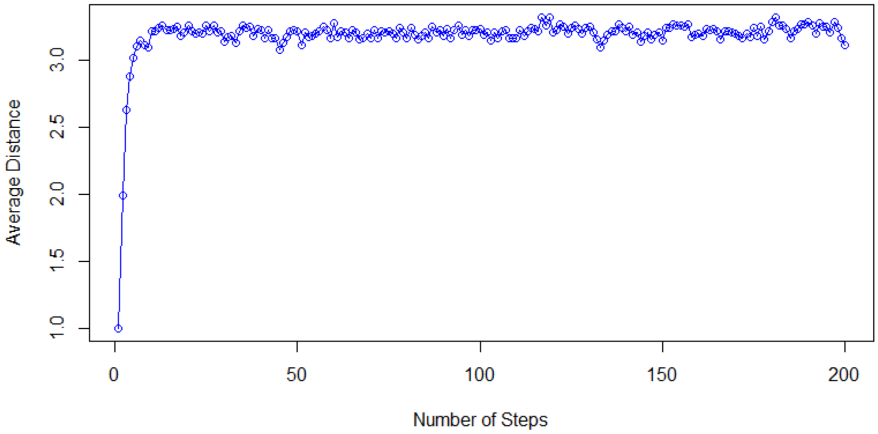

Figure 18: Average distance against number of steps

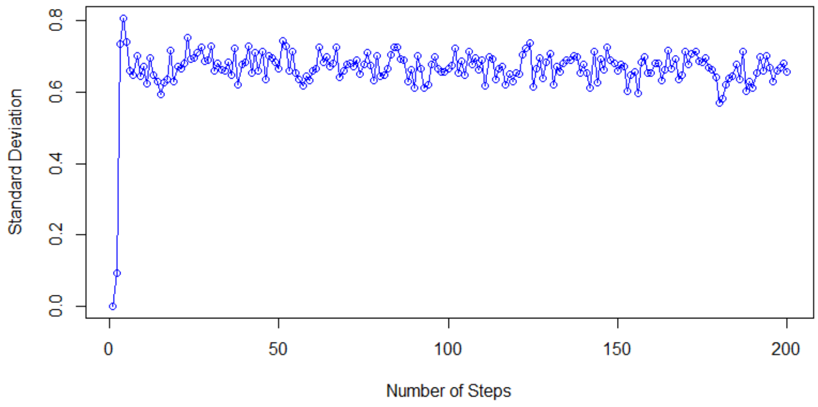

Figure 19: Standard deviation against number of steps

As we can see in the two figures above, both average distance and standard deviation tend to be converged. The results are given by averaging over 250 different networks with 200 random walk steps.

(c)

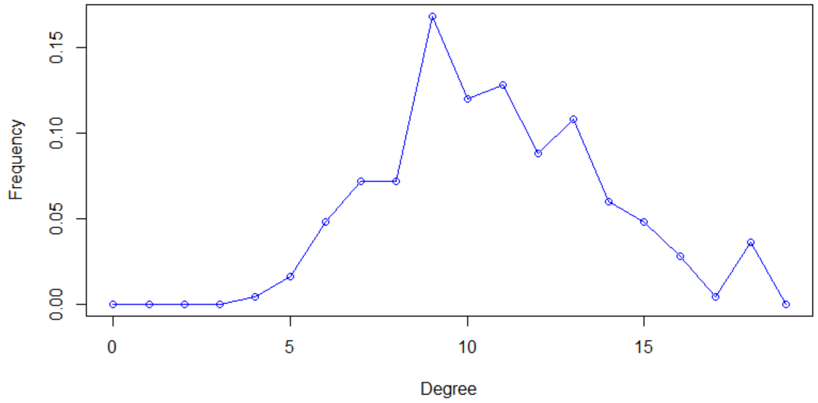

Figure 20: End node degree

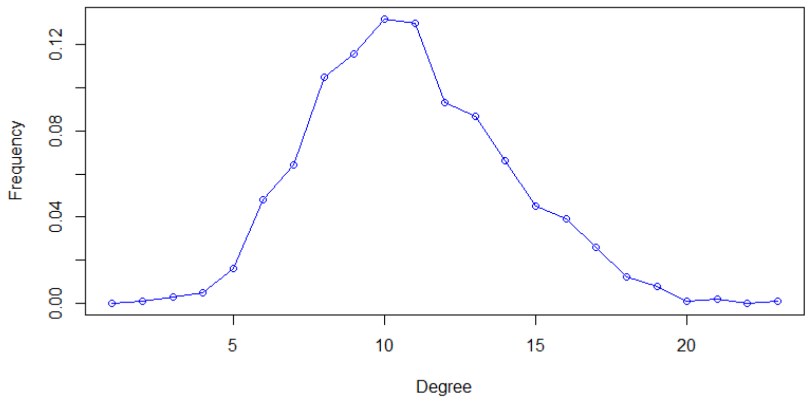

Figure 21: Graph degree

Same as the network settings for task (b), here we plot the end node degree and graph degree separately. After 200 steps of random walk, the end points tend to be at those points with larger degrees, which can also be verified by the theory that:

$$f(k) = \frac{kp(k)}{\sum_k kp(k)} \tag{5}$$

where *p(k)* represents the degree distribution of the network.

(d)

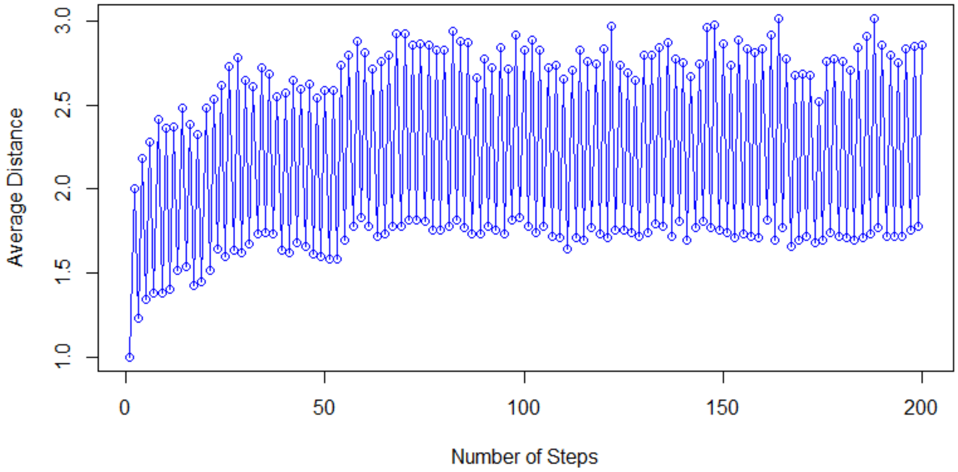

Figure 22: Average distance of 100-node network

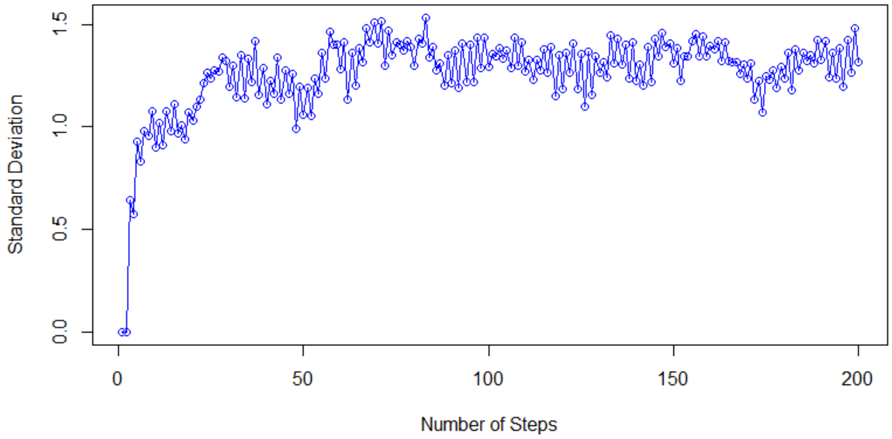

Figure 23: Standard deviation of 100-node network

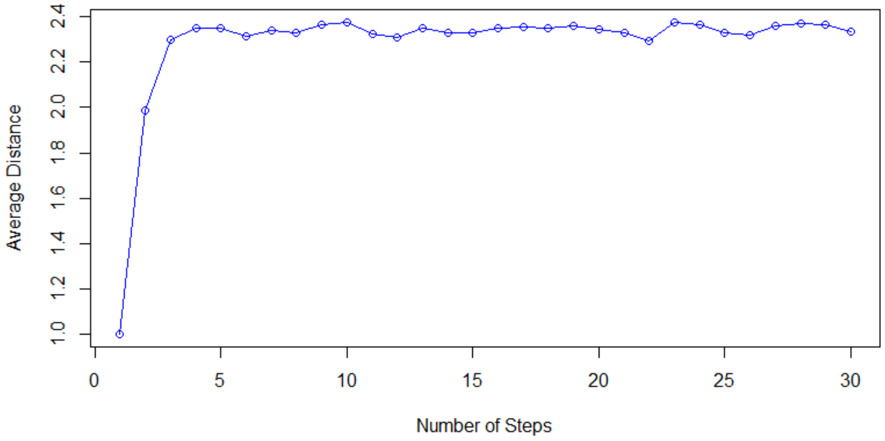

Figure 24: Average distance of 10000-node network

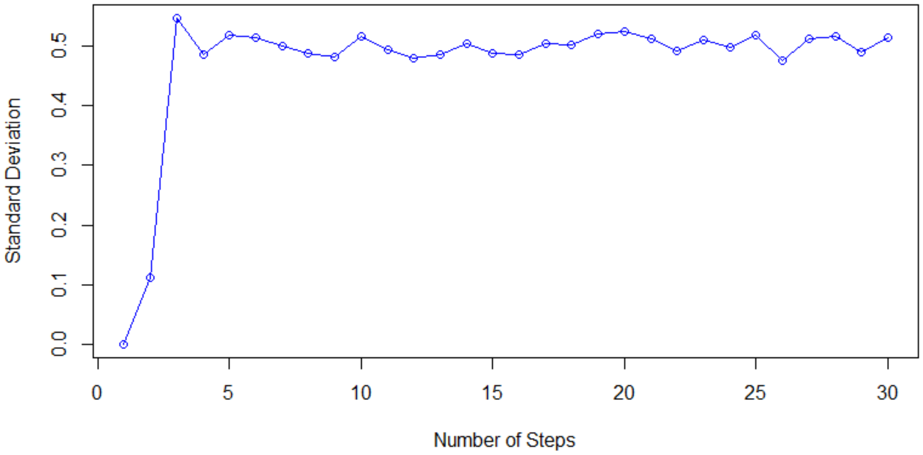

Figure 25: Standard deviation of 10000-node network

Here, we draw two sets of plots for the network with 100 nodes and 10000 nodes separately. For 100-node network, the test settings are the same (250 networks with 200 steps), but for 10000-node network, the test settings are different, we use 250 different with 30 steps. The reason why we did so is that, after 30 steps, the plots are already to be converged and we could save computing expense in the same time.

For 100-node network, due to its high sparsity (with p=0.01 and only 100 nodes), the average distance is not converged. And for 10000-node network, the results are satisfying. Since the distance cannot be larger than the diameter (upper bound) and those networks we used have relatively small diameter, the average distance is not that big which means the walking agent is basically around its original position.

The diameter and average distance for each network are shown in the table below:

Table 4: Diameter and average distance for each network

Number of nodes	**Diameter**	**Average Distance (200 steps)**
100	14	1.7 - 2.8
1000	6	~3.25
10000	3	~2.3

The diameter now has a strong impact to the average distance especially for the network with big number of nodes. Because the diameter for such large network is small, and the diameter strictly restrict the average distance.

2. Random walk on networks with fat-tailed degree distribution

(a)

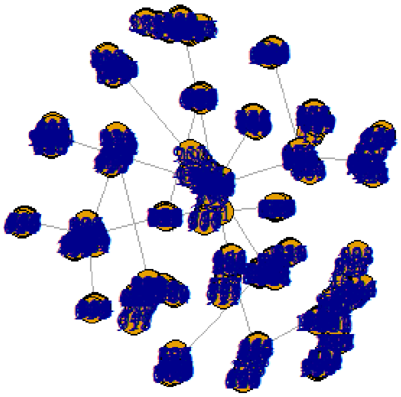

Figure 26: Undirected preferential attachment network

(b)

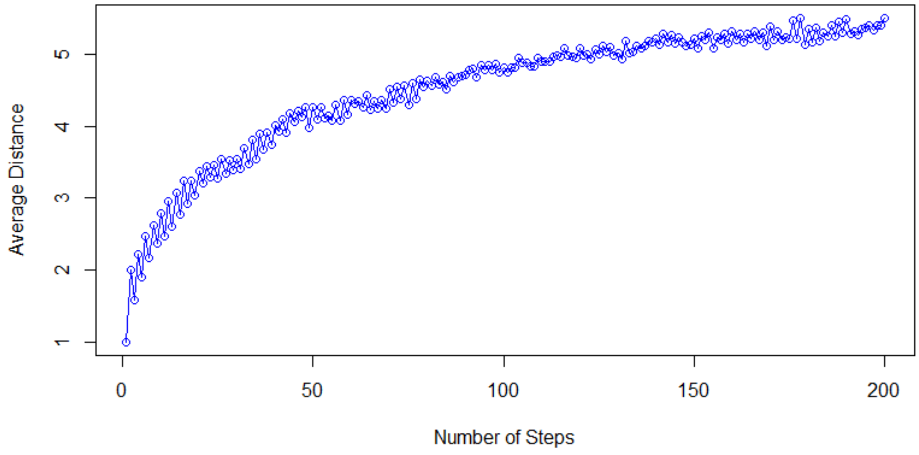

Figure 27: Average distance with 200-step random walk

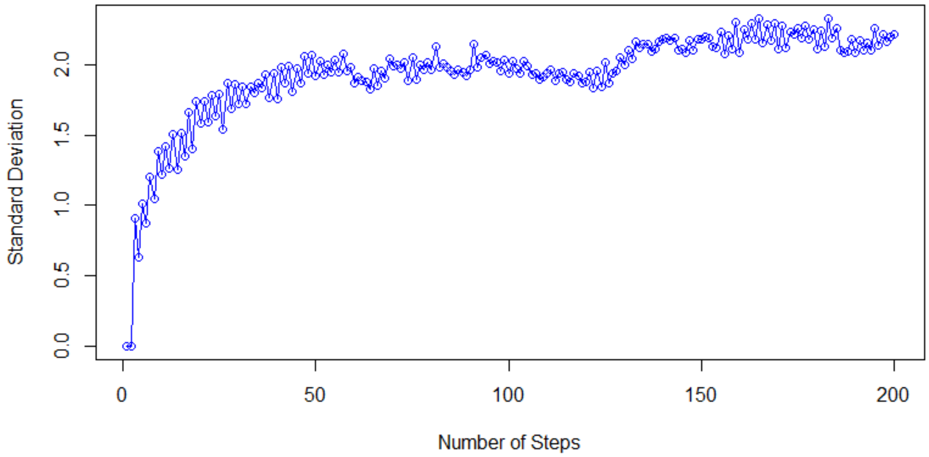

Figure 28: Standard deviation with 200-step random walk

The results are given by averaging over 250 different networks with 200 random walk steps. Different from the results in last problem, the average distance didn't converge after 200 steps. Considering it's a fat-tailed network, that makes sense. And we also tried 500-step random walk:

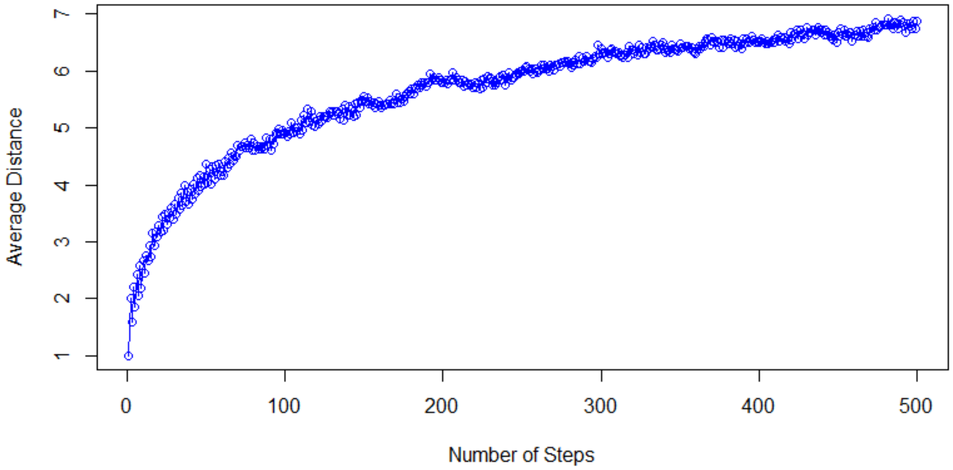

Figure 29: Average distance with 500-step random walk

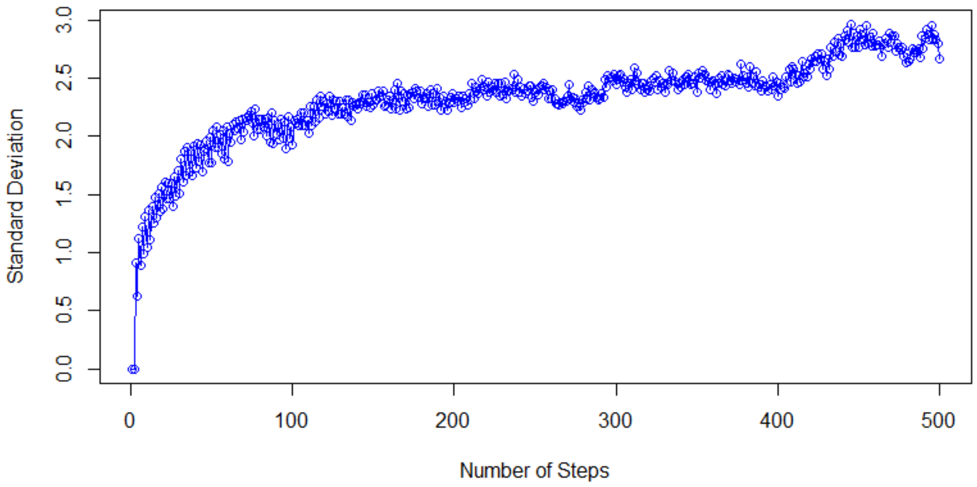

Figure 30: Standard deviation with 200-step random walk

It also indicates that growing tendency. It is because that the fat-tailed network tends to have bigger diameter than the ER network with same number of nodes.

(c)

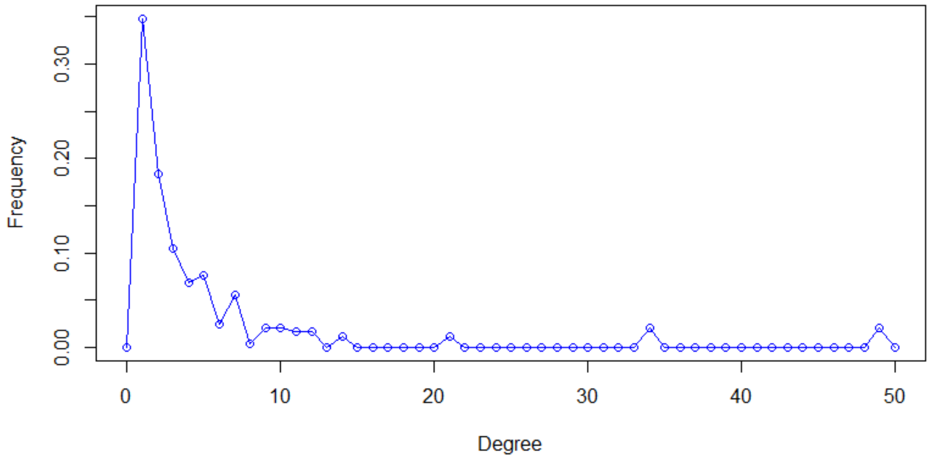

Figure 31: End node degree

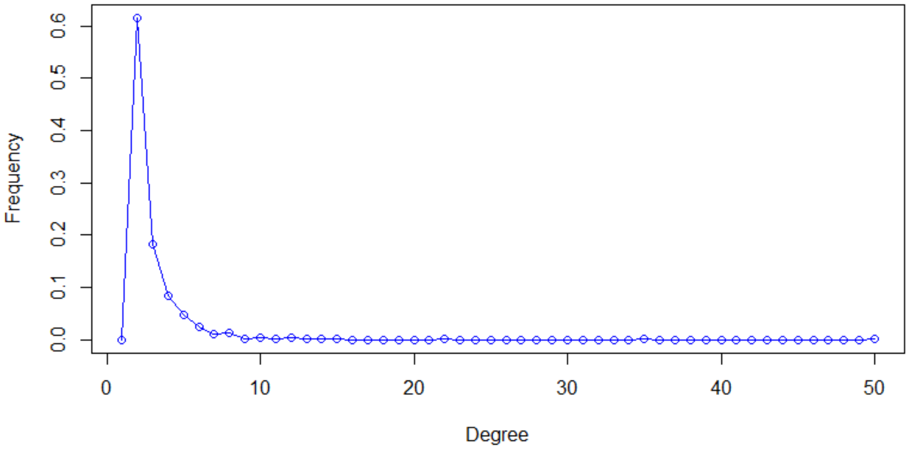

Figure 32: Graph degree

Similarly, after 200 steps of random walk, the end points tend to be at those points with larger degrees, just like the results given by ER network. Though the distribution itself is very different.

(d)

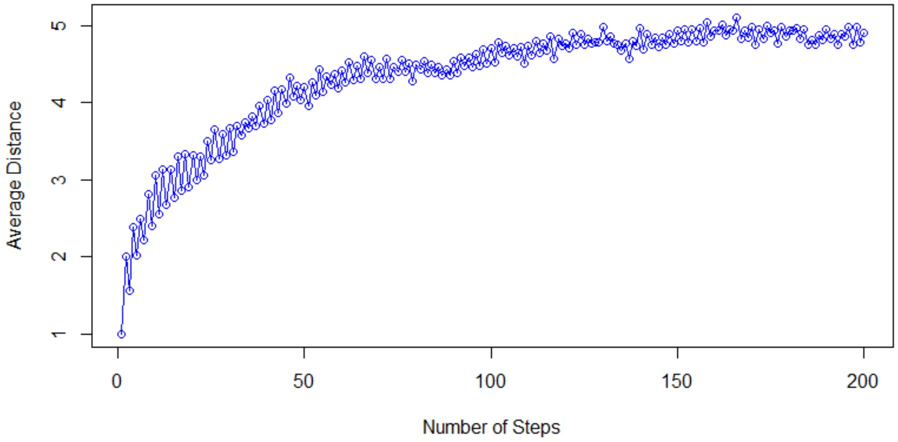

Figure 33: Average distance of 100-node network

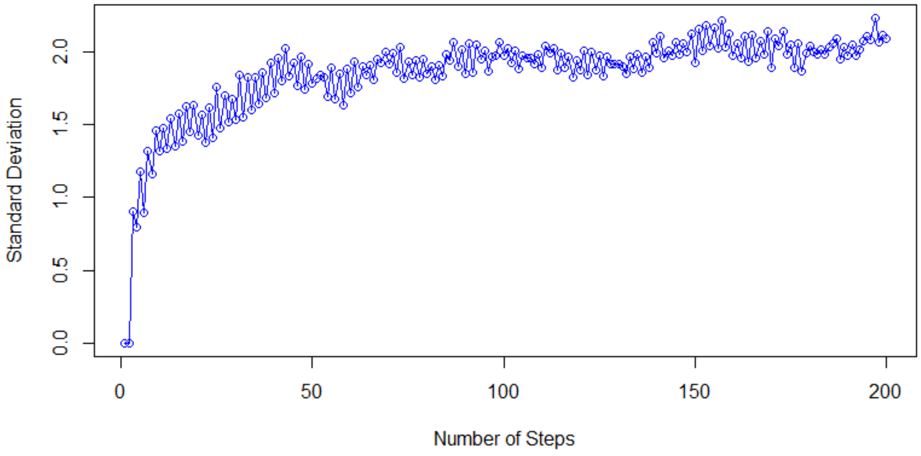

Figure 34: Standard deviation of 100-node network

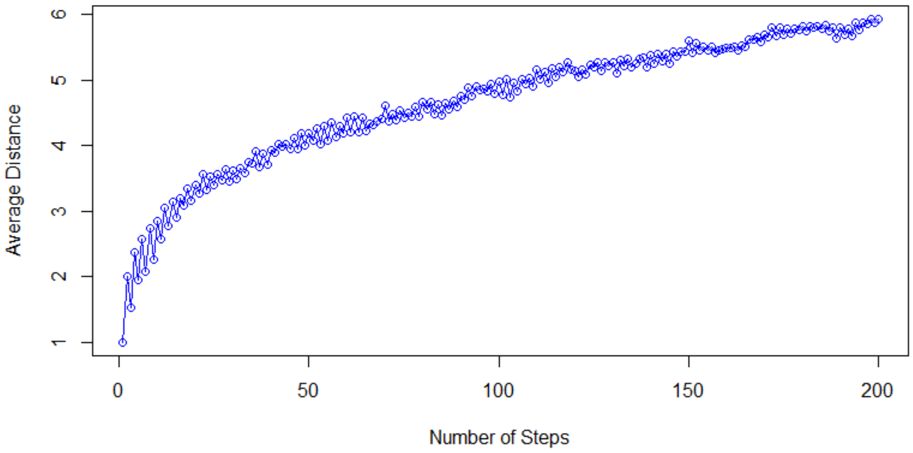

Figure 35: Average distance of 10000-node network

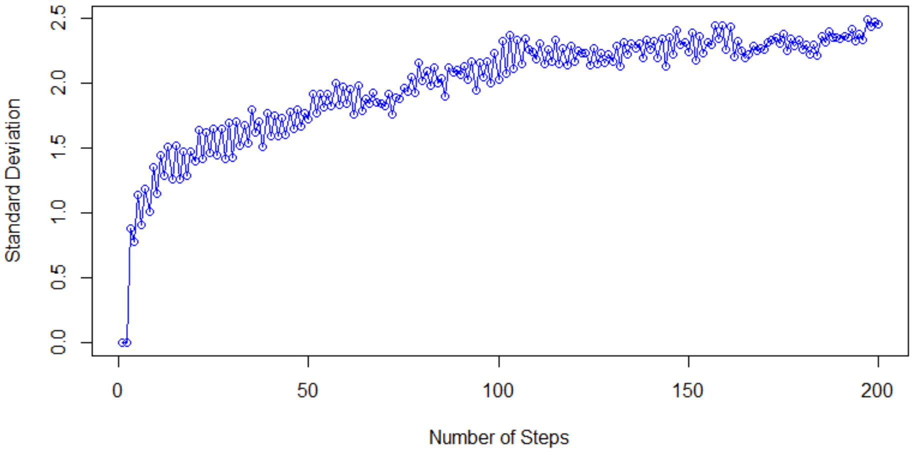

Figure 36: Average distance of 10000-node network

For 100-node fat tailed network, after 200 steps of random walk, the average distance seems to be converged. But for the 10000-node one, obviously, the average distance is not converged. Comparing with the ER network, when walking on a fat-tailed network, it needs more steps to make the average distance to converge. We analyze this with the degree distribution plots as well.

The diameter and average distance for each network are shown as follow:

Table 5: Diameter and average distance for each network

Number of nodes	**Diameter**	**Average Distance (200 steps)**
100	11	~5
1000	19	~5.5
10000	27	~6

From the observation, as the diameter increases, the average distance a random walker can reach with certain number of steps also becomes bigger. However, we could also consider them as the same value, since the diameter should not have strong impact on the random walker's average distance for a fat-tailed network. But of course, the diameter is still the upper bound of distance.

3. PageRank

(a)

Since it is a graph with m=4, meaning that the in-degrees follow a power law distribution. And for each node i, there is a path to node 1. Because by the definition m is the number of edges to add in each time step, every node after node1 must has a path to node1. In our file Part2_3_random_walk, the steady state after each random walk is always node1, then the probability that the walker visits each node is [1, 0,0,0 …,0] with node1 of probability 1. This result is close to the result using page_rank functions, but there are still a lot of data missing using random walk methodology. So, we decided to use page_rank functions to generate results in all following questions.

The plot of probability that the walker visits each node, and the plot of degree of nodes are as follow:

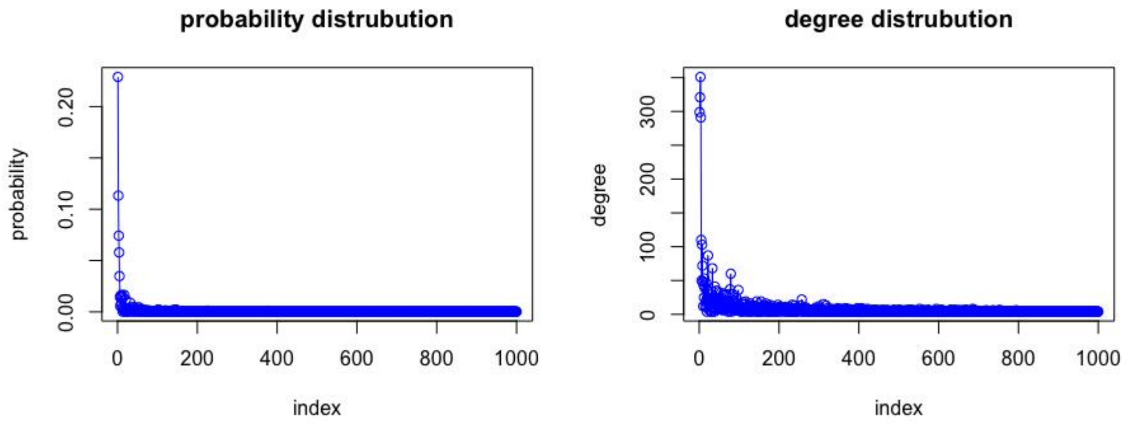

Figure 37: Left: probability distribution of 3(a). Right: degree distribution

The cosine similarity of the two vectors is 0.7713765, which means these two vectors are very related. This is because by the definition of Page Rank algorithm, the more edges points to the node, the more important the node is. If we calculate the similarity of the probability vector and the in-degree vector of nodes the result is 0.795458.

(b)

Now, we use a teleportation probability of α = 0.15. By performing random walks on the network created in 3(a), the plot of probability distribution that the walker visits each node is:

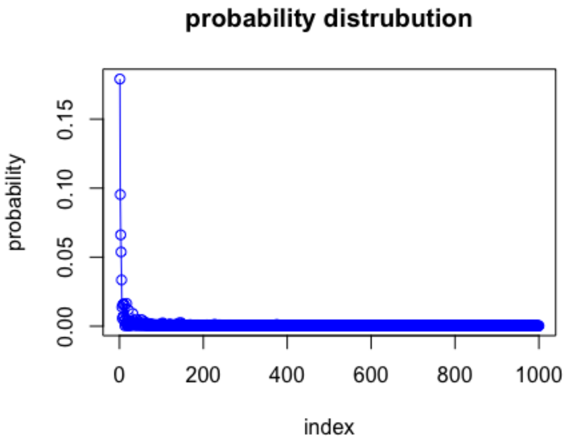

Figure 38: Probability distribution of 3(b)

The cosine similarity of the vector of probability distribution and the vector of degree of nodes is 0.8122541, which also indicates that the probability distribution is closely related to the degree of nodes. If we use in-degree instead of degree of nodes, the cosine similarity is 0.8333458. The reason is by the definition of Page Rank algorithm, the more edges points to the node, the more important the node is. So the nodes with high in-degree value has large page rank value.

4. Personalized PageRank

(a)

We use random walk on network generated in part 3 to simulate this personalized PageRank. The teleportation probability to each node is proportional to its PageRank. let the teleportation probability be equal to α = 0.15. By the equation of Page Rank,

$$\mathbf{PR} = \alpha \cdot \mathbf{d} + (1-\alpha) \cdot L \cdot \mathbf{PR} \tag{6}$$

here $\mathbf{d} = \mathbf{PR}$, which leads to $\mathbf{PR} = \alpha \cdot \mathbf{d} + (1-\alpha) \cdot L \cdot \mathbf{PR} = \mathbf{I} \cdot \mathbf{PR}$ if $L = \mathbf{I}$. The plot of probability distribution that the walker visits each node is:

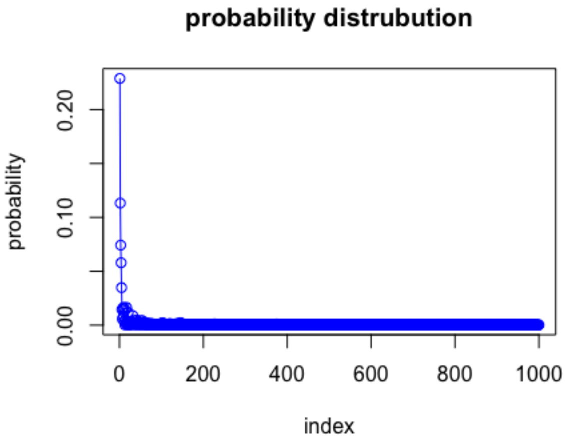

Figure 39: Probability distribution of 4/(a)

Compare with the probability distribution of 3(b), the cosine similarity between the probability distribution of 3(b) and 4(a) is 0.997583, which means the two vectors are co-related for sure. And by comparing the top 10 values:

Table 6: Top 10 value of probability distribution of 3/(b) and 4/(a)

	3(b)	4(a)
1	0.179048254	0.228908102
2	0.095363355	0.113257246
3	0.066190752	0.074220980
4	0.053771062	0.057936228
5	0.033498093	0.034843340
6	0.013758311	0.014427606
7	0.005597532	0.005632217

8	0.015542932	0.015575168
9	0.007171765	0.007796766
10	0.016264237	0.016551642

We noticed that the top values of 4(a) are all greater than the corresponding values of 3(b). Especially the head of vector in 4(a) is about 0.05 greater than that in 3(b). It is because the page rank vector is not the uniform distribution, which makes the top nodes having more importance than the tails since they can be reached with higher probability in 4(a) than that in 3(b).

(b)

We chose node 13 and 22 to set teleportation land only on those two nodes. The result is:

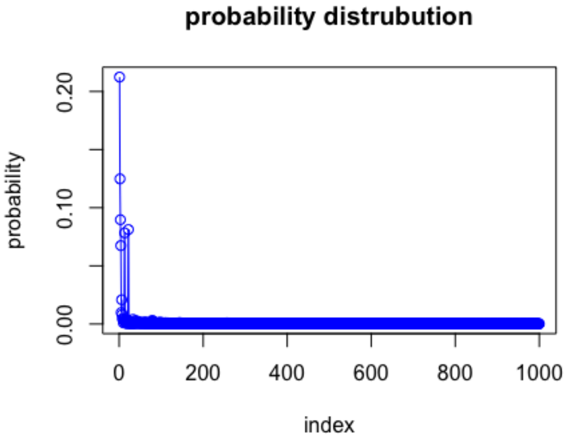

Figure 40: Probability distribution of 4(b)

From the graph we can notice that there are two nodes with far greater probability than neighbor nodes. We guessed that those nodes are exactly the two we picked. From 3(b) we got the probability values of node 13 and 22 are 0.000302191 and 0.000302191. In 4(b) the probability values are 0.07831773 and 0.08122546. Their value increased about 200 times than those in 3(b). The PageRank value changed a lot. But the top nodes still took the dominance. This is because the teleportation forced the random walker to visit node 13 and 22 in the beginning cycles. But after a number of cycles the nodes with much higher in-degree value beat the two we chose.

(c)

The formula for personalized PageRank is as follows:

$$\mathbf{PR} = \alpha \cdot \mathbf{d} + (1-\alpha) \cdot (\frac{\mathbf{PR}(T_1)}{\mathbf{C}(T_1)} + ... + \frac{\mathbf{PR}(T_n)}{\mathbf{C}(T_n)}) \tag{7}$$

Where **PR** is the PageRank value, α is teleportation value, **C** is the count, or number, of outgoing links for each node.