EE232E Project 1

Random Graphs and Random Walks

Hongyan Gu – 205025476

Xin Liu – 505037053

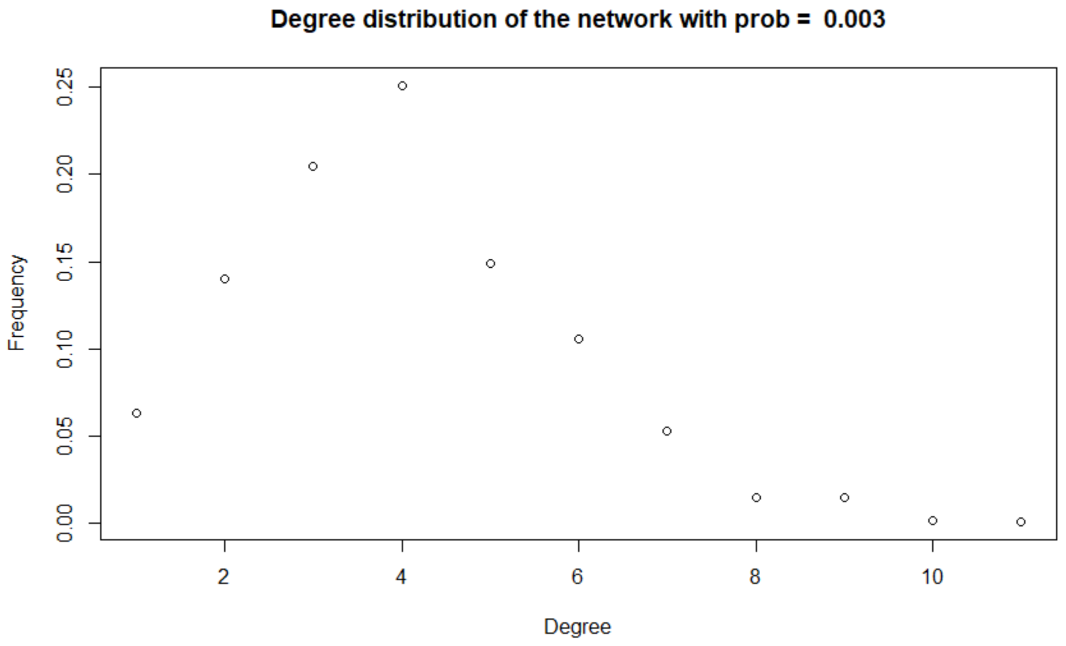
Aoxuan (Douglas) Li - 905027231

Jiawei Du - 404943853

I. Generating Random Networks.

1. Create random networks using Erdös-Rényi (ER) model
(a)
The distribution plots are shown below:

p = 0.003



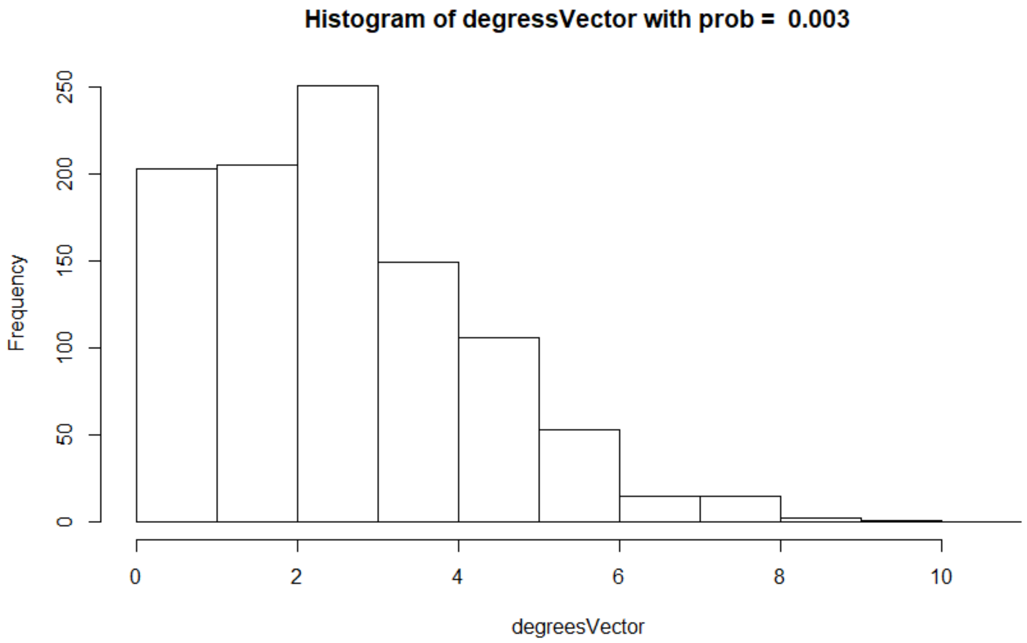
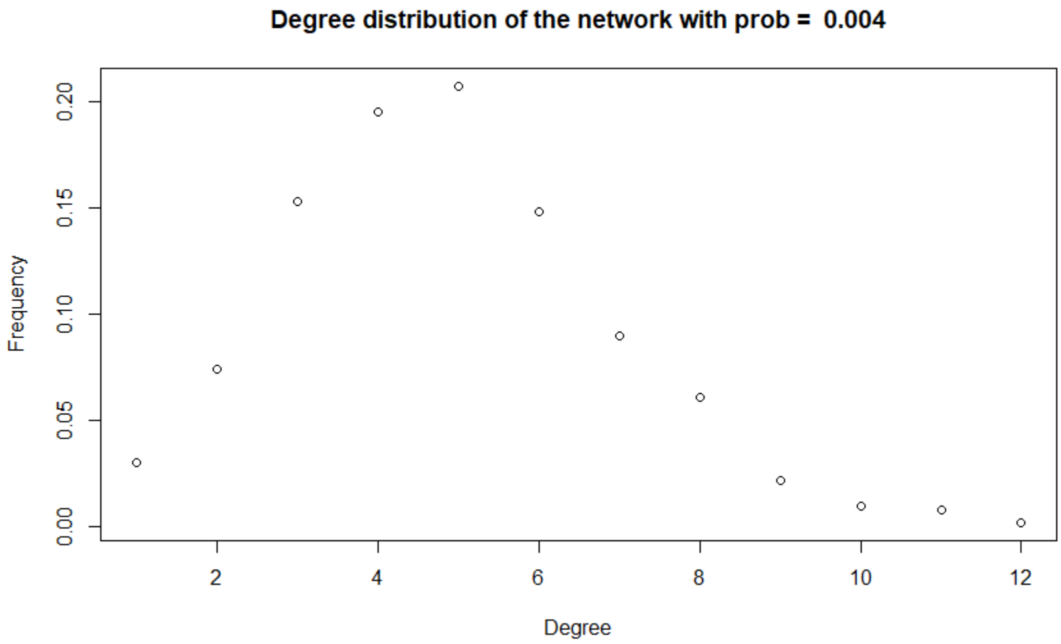


Figure 1(a)(b). Degree distribution and histogram for probability = 0.003

p = 0.004



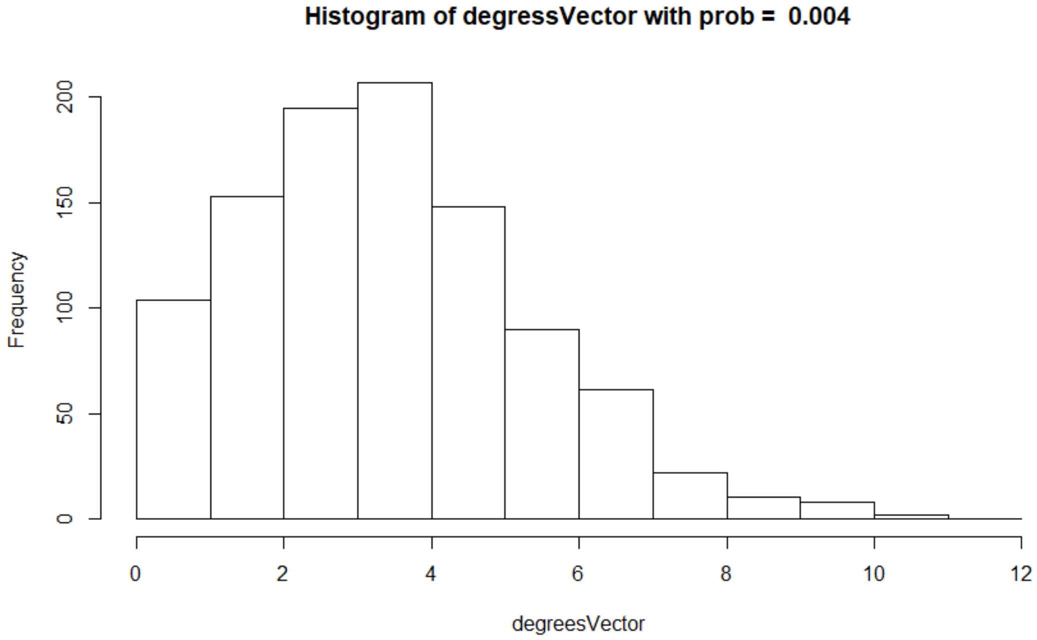
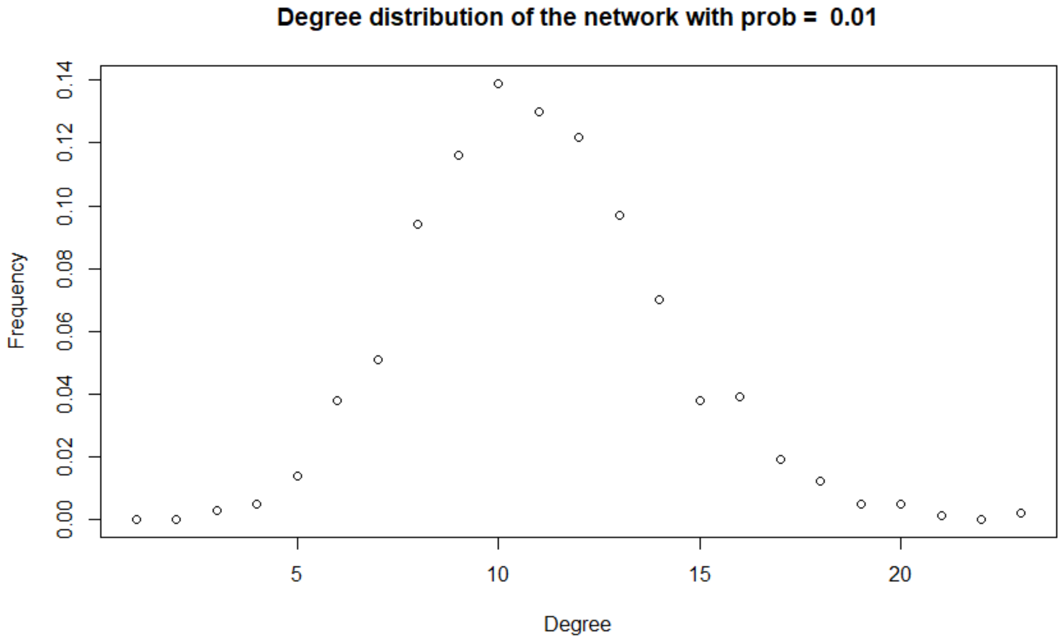


Figure 2(a)(b). Degree distribution and histogram for probability = 0.004

p = 0.01



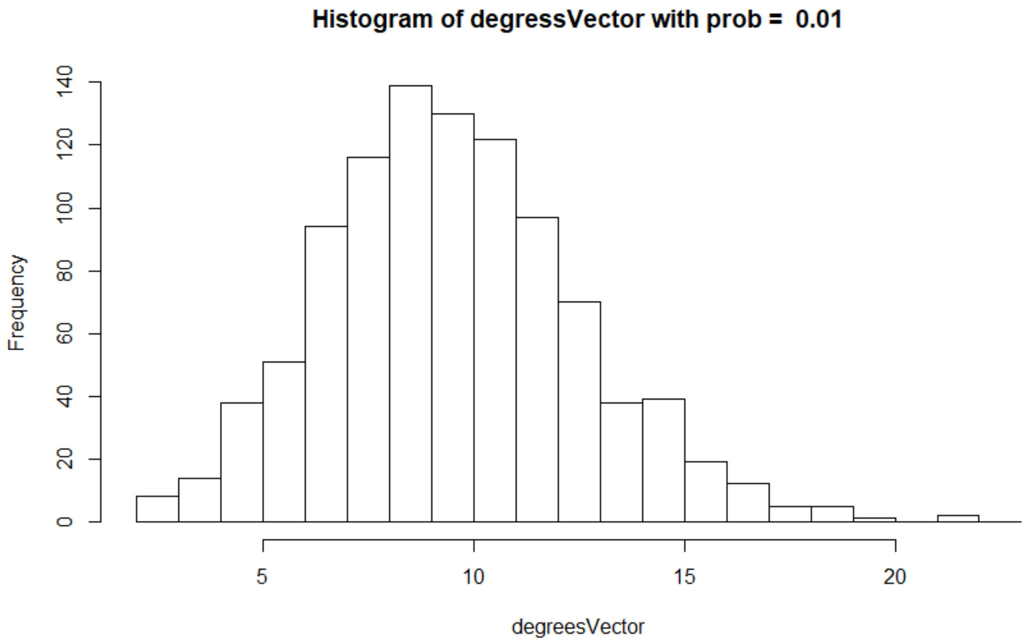
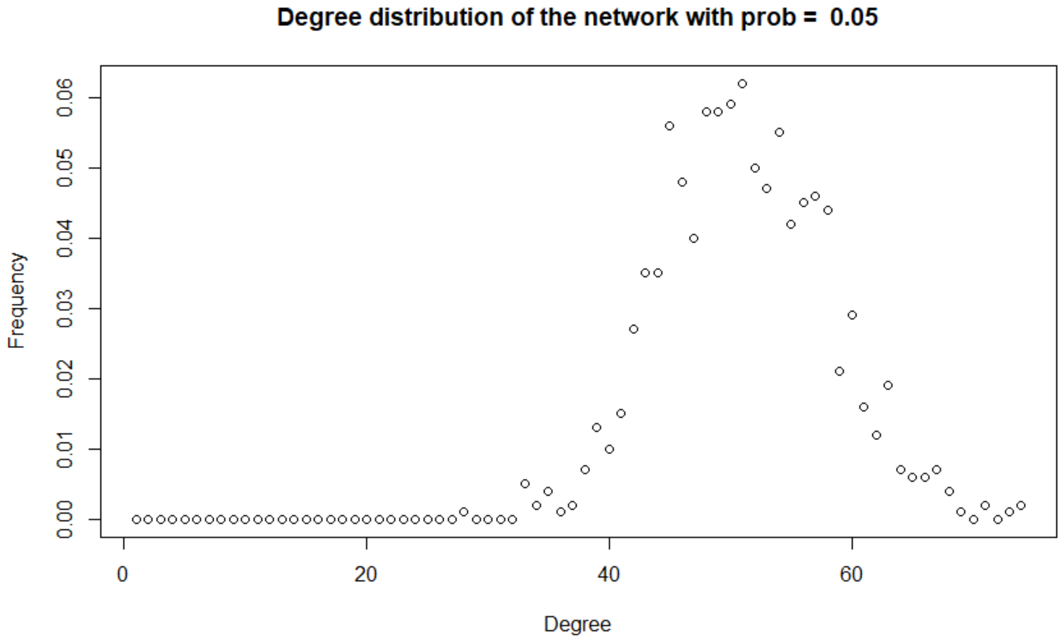


Figure 3(a)(b). Degree distribution and histogram for probability = 0.01

p = 0.05



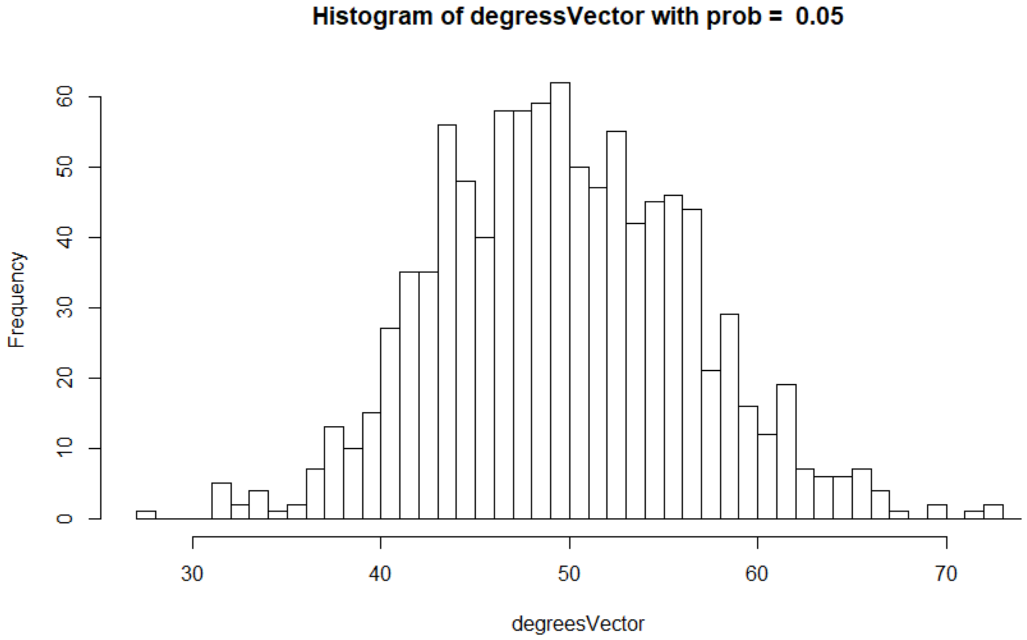
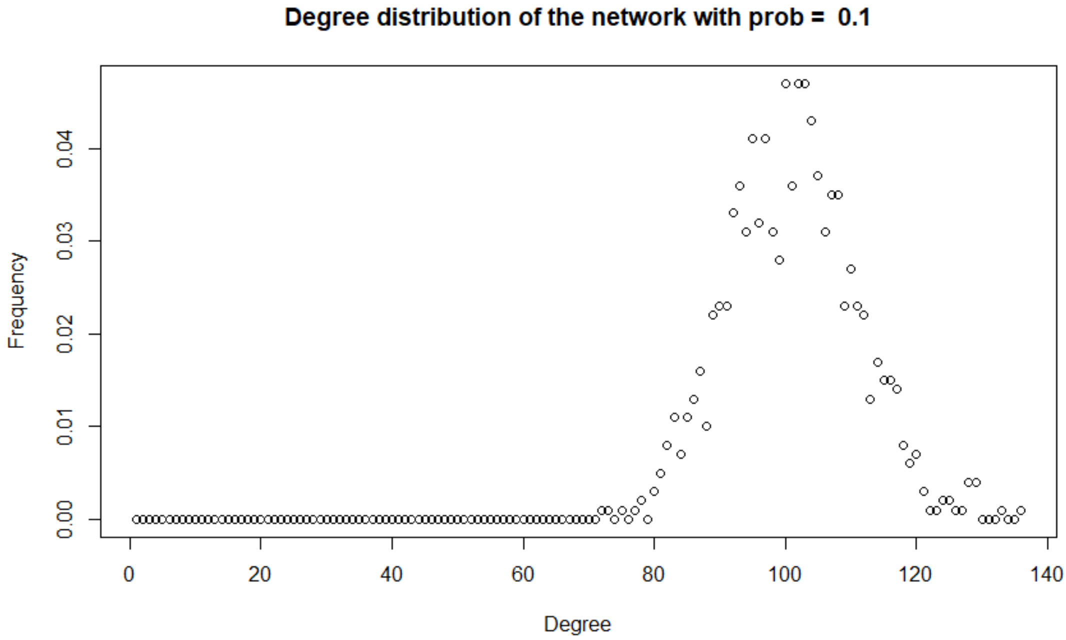


Figure 4(a)(b). Degree distribution and histogram for probability = 0.05

p = 0.1



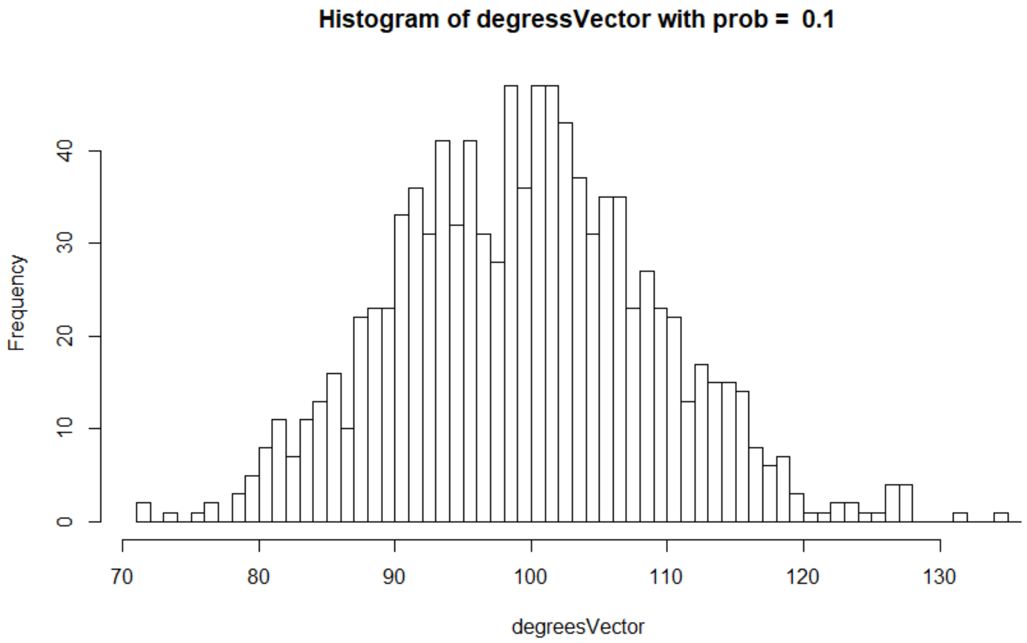


Figure 5(a)(b). Degree distribution and histogram for probability = 0.1

The distribution of degree is a **binomial distribution**. For Erdös-Rényi model, edges of two nodes are selected with equal possibility p. Therefore, in a graph with n+1 nodes, the possibility of a node to be degree k is

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

It is a binomial distribution.

The mean values and variances for different p is shown below:

Table 1. Tested mean values and variances for ER model

Possibility	**0.003**	**0.004**	**0.01**	**0.05**	**0.1**
Mean	3.00	3.86	10.00	50.19	100.05
Variance	3.12	3.98	9.27	46.10	93.45

For a binomial distribution, the theoretical mean value and variance is calculated by:

$$\mu = np$$
$$\sigma^2 = np(1-p)$$

For this particular example, n = 999. The theoretical mean value and variance is:

Table 2. Mean values and variances for ER model

Possibility	**0.003**	**0.004**	**0.01**	**0.05**	**0.1**
Mean	3.00	4.00	9.99	49.95	99.90
Variance	2.99	3.98	9.89	47.45	89.91

Due to the number of samples 1000 is not a value that is extremely large, the results we obtained had slight difference with the theoretical ones, but they are reasonably very close to each other. Additionally, notice that when p → 0, the mean values and variances are theoretically the same, which is also consistent with the results we obtained.

(b)

We iterated 1000 times to calculate the possibilities that a ER graph is connected for a particular possibility p and reported the properties of GCC and its largest diameter. The results are shown below:

Table 3. Results for a ER graph with different possibilities

Possibility	**0.003**	**0.004**	**0.01**	**0.05**	**0.1**
Possibility to be connected	0	0	0.951	1	1
Number of nodes in GCC in a particular example	951	982	1000	1000	1000
Diameter of that GCC	13	11	5	3	3

(c)

We plotted the curve of normalized GCC sizes v.s. probabilities.

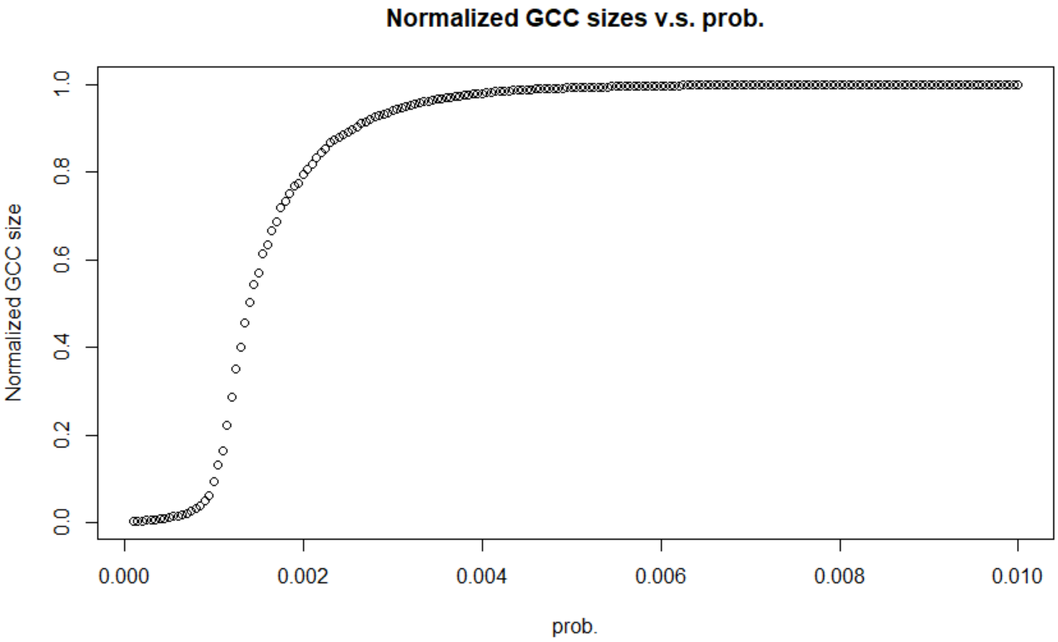


Figure 6. Normalized GCC sizes v.s. probability

For our criterion, **an GCC's emergence could be regarded as a turning point of a curve**. From the plot given, we could see that the GCC started to emerge at probability around **0.001**, which is consistent to theory that the **critical turning point** is given by p = 1/n = 0.001.

More specifically, we assume the average degree of graph is $\alpha = \mathrm{np}$, then if $\alpha < 1$ all the components are in size of $\mathrm{O}(\log n)$; if $\alpha < 1$, there is one component of size $\mathrm{O(n)}$, with others have size $\mathrm{O}(\log n)$.

We could also define the **'completed emergence' of GCC as there are few isolated nodes existing in the graph**. For this definition, we could get the conclusion that when $\mathrm{p} = \ln(n)/\mathrm{n}$, there are fewer isolated nodes, and when $\mathrm{p} = 2\ln(n)/\mathrm{n}$, there would be completely no isolated nodes with high possibility. This is typically what we say interesting properties for $\mathrm{p} = \mathrm{O}\left(\frac{\ln(n)}{n}\right)$. Next is the proof.

We assume $\mathrm{np} = \alpha$. The probability for us to have an isolated node is:

$$\mathrm{P(degree} = 0) = (1-p)^n = \left(1 - \frac{\alpha}{n}\right)^n \to e^{-\alpha} \qquad (n \to \infty)$$

The event we are asking about is I = 'some nodes are isolated' $= \bigcup_{v \in N} I_v$, where I_v is the event that a node v is isolated. We have:

$$\mathrm{P(I)} = P\left(\bigcup_{v \in N} I_v\right) \le \sum_{v \in N} P(I_v) = n\, e^{-\alpha}$$

This inequality follows the union bound inequality.

Clearly, when $\mathrm{np} = \alpha = \ln(\mathrm{n})$, $\mathrm{P(I)} \le 1$, which means there would be fewer isolated nodes. When $\mathrm{np} = \alpha = 2\ln(\mathrm{n})$, $\mathrm{P(I)} = \frac{1}{n} \to 0 \quad (n \to \infty)$, there would be no isolated nodes. For this part, ln(n) /n= ln 1000 /1000=0.007 and 2ln(n)/n = 0.014. From the plot above, we could see that both the results showed that the graph almost have no isolated nodes, where **GCC completely emerged**.

(d)

i. The plot is given below:

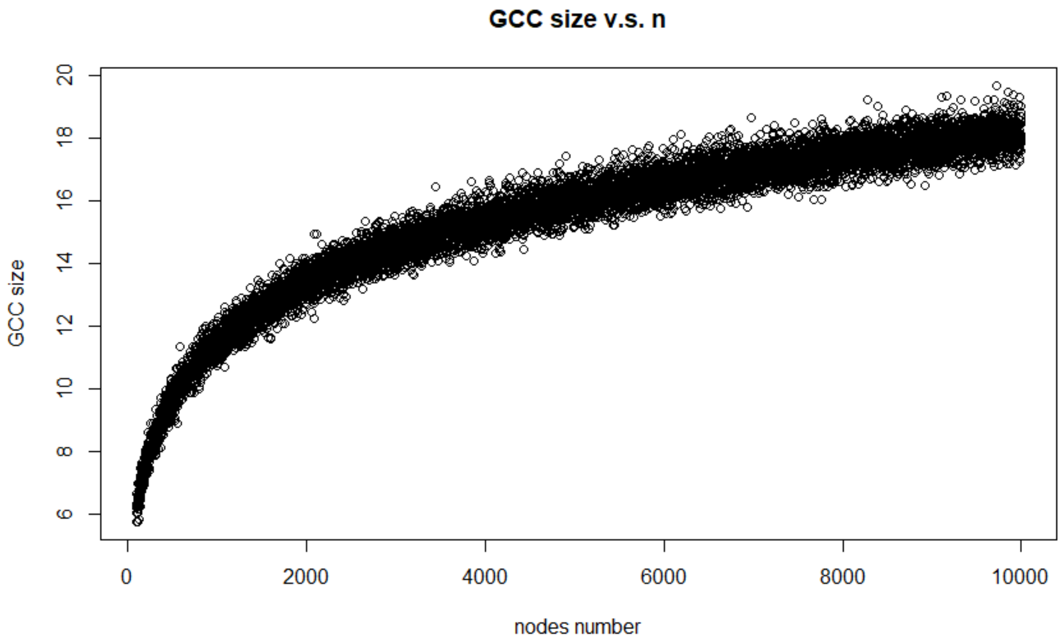


Figure 7. GCC sizes v.s. n with np = 0.5

From the plot, we could see that the trend is a **log function**. As stated in (c), for $\alpha = \text{np} = 0.5 < 1$, all the components are in size of $\mathbf{O}(\log \boldsymbol{n})$, so that the function is a log function. To see this, we plot the GCC sizes v.s. log(# of nodes), the results turned out to be a linear trend, which indicates that the original trend is a log function.

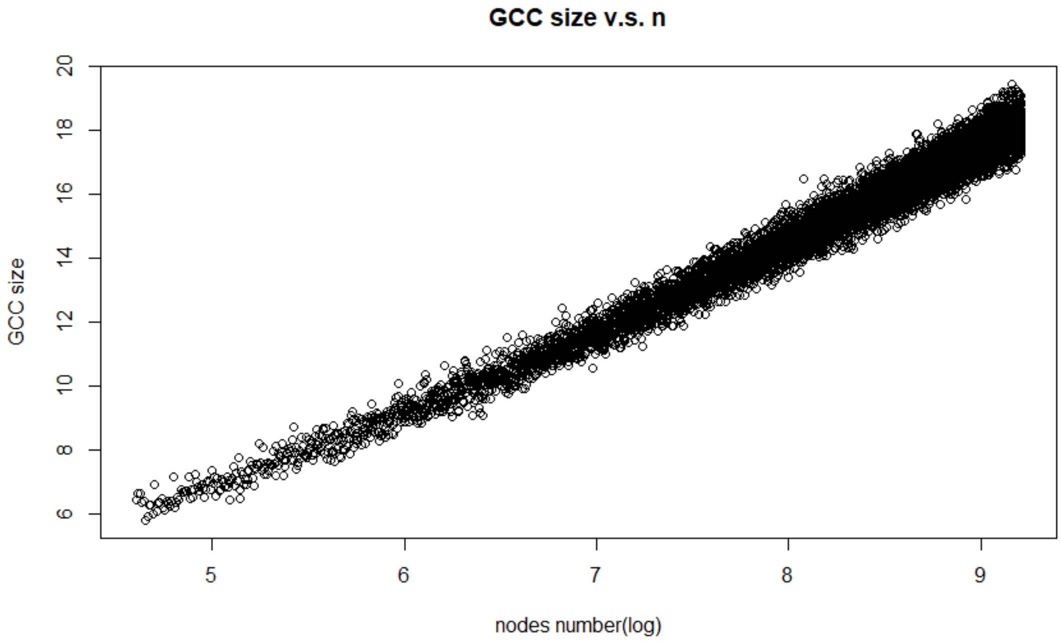


Figure 8. GCC sizes v.s. log(n) with np = 0.5, an almost linear trend

ii.

The plot is given below:

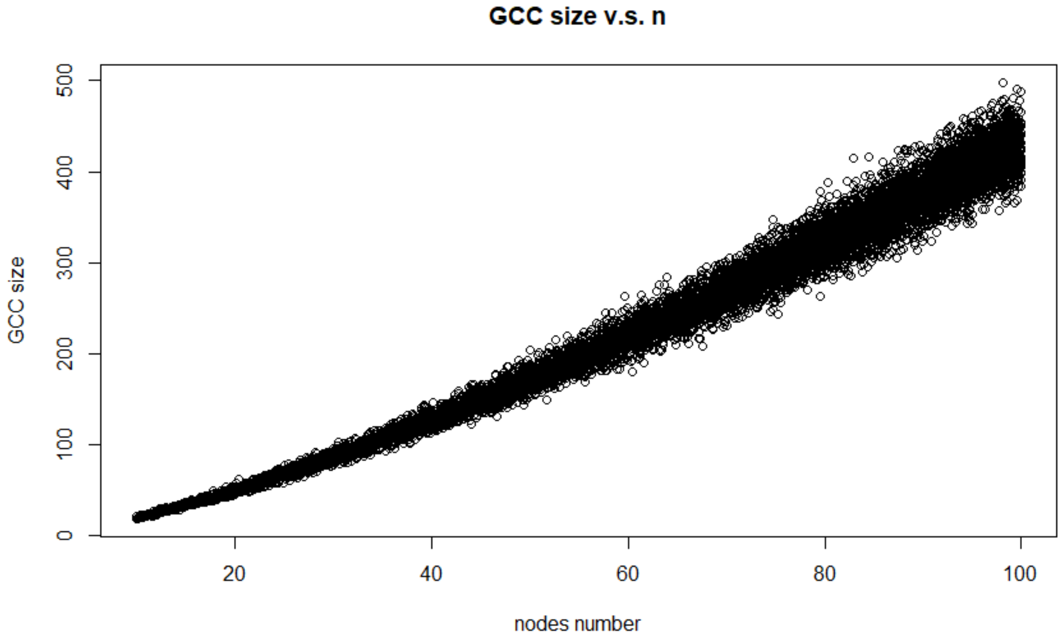


Figure 9. GCC sizes v.s. n with np = 1

In the lecture, the professor stated that for $\alpha = \text{np} = 1$, the number of vertices in the largest component of the graph is $\mathbf{O(n^{1/2})}$. On Wikipedia, it is stated as $\text{O}(n^{2/3})$. After testament, we found that $\mathbf{O(n^{2/3})}$ **could better describe the situation (See following)**. The following is the comparison, from which we could see the plot showed better linearity when we drew GCC sizes over $n^{2/3}$.

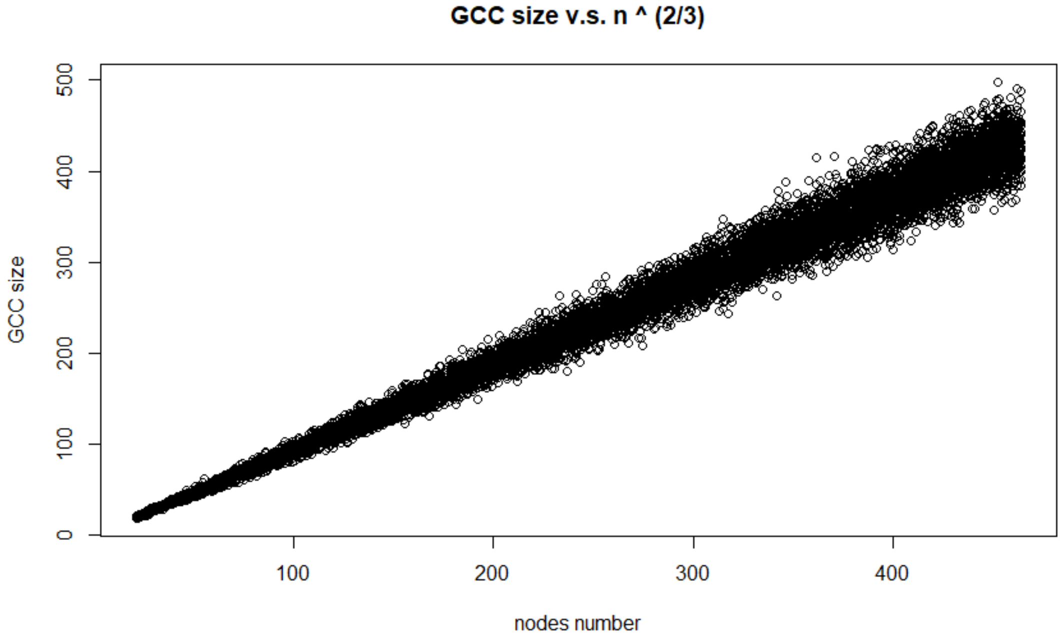


Figure 10. GCC sizes v.s. n ^ (2/3) with np =1

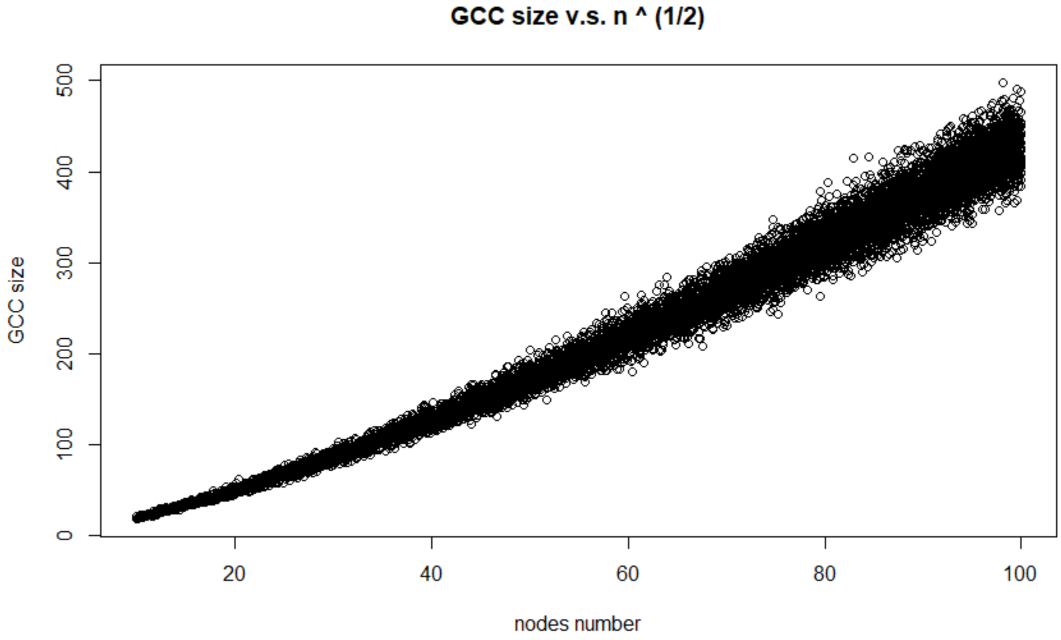


Figure 11. GCC sizes v.s. n ^ (1/2) with np =1

iii.

For $\alpha = np > 1$, the number of vertices in the largest component of the graph is O(n). So that for all c = 1.1, 1.2 and 1.3, the GCC sizes are almost linear function with respect to n.

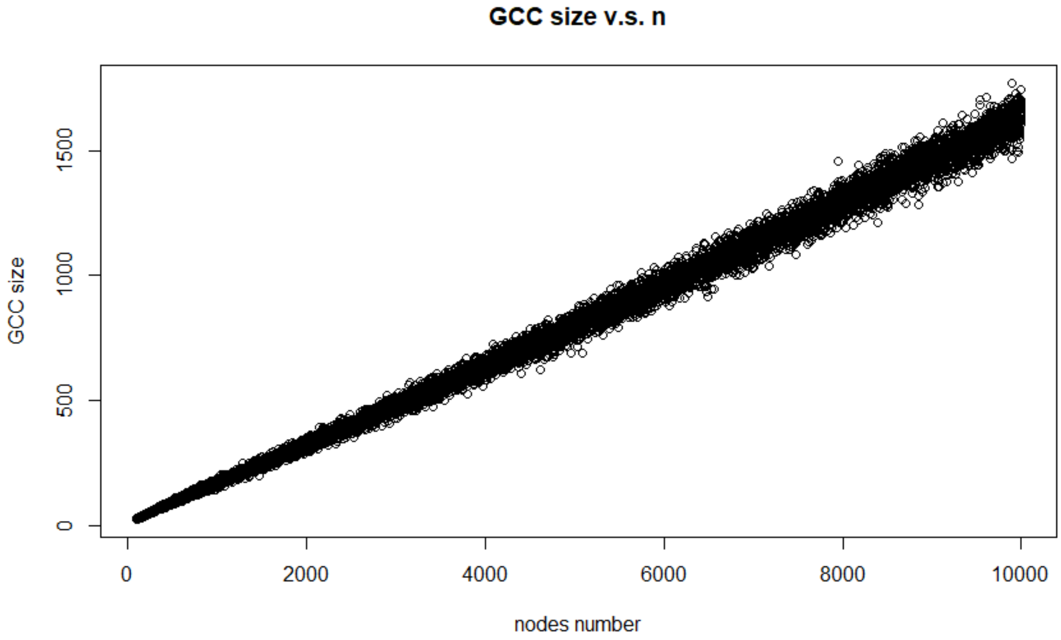


Figure 12. GCC sizes v.s. n with np =1.1

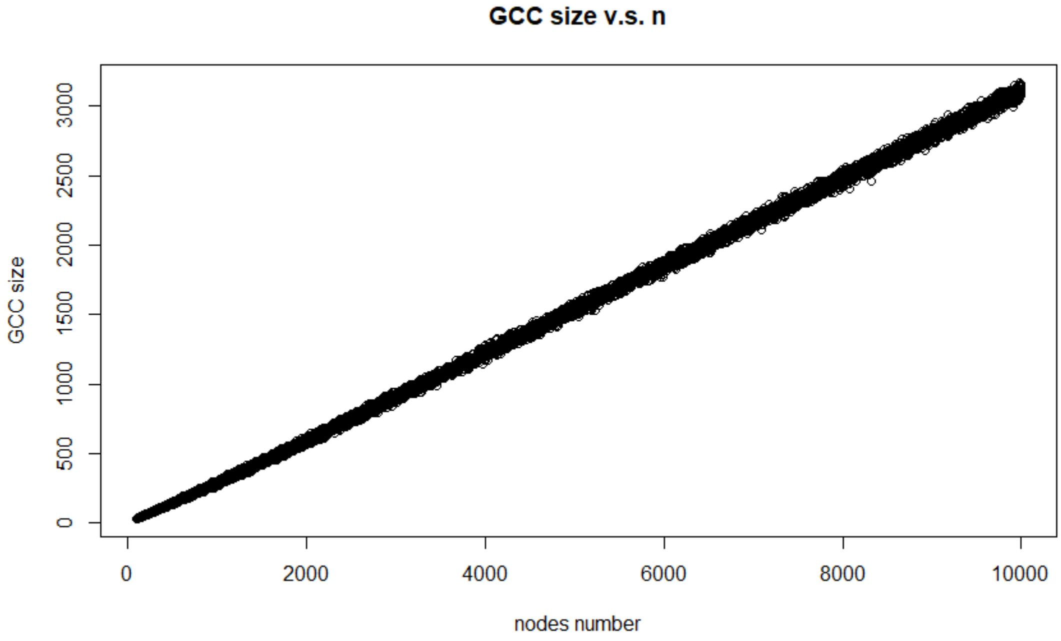


Figure 12. GCC sizes v.s. n with np =1.2

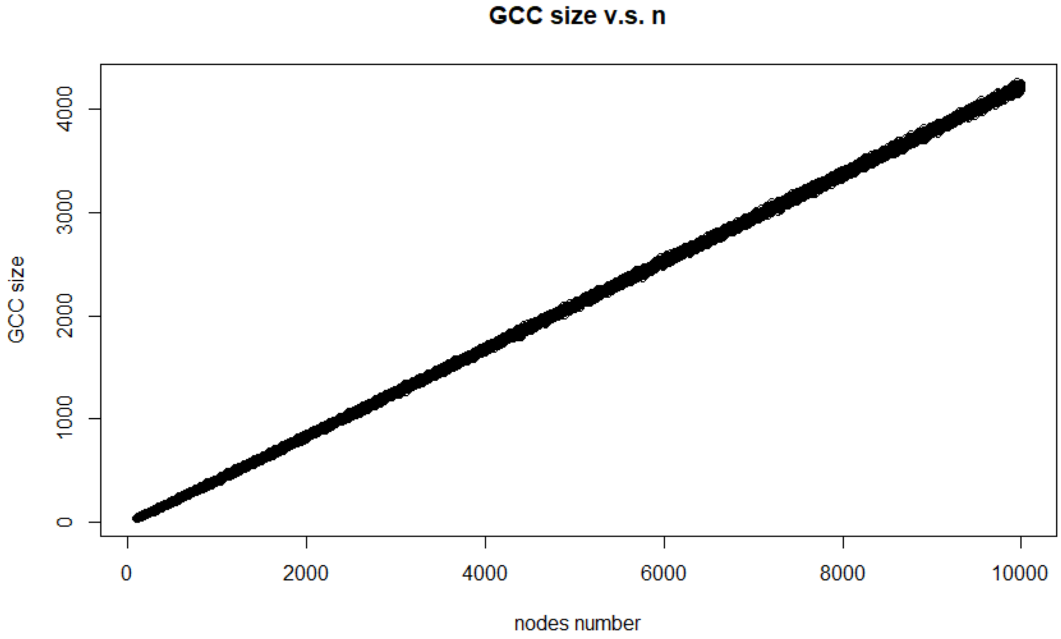


Figure 12. GCC sizes v.s. n with np =1.3

To explain that GCC sizes is in O(n) for np > 1, we could consider the expectation of a single node, which is np as stated in previous part. For the probability that the node is connected with a node which is k steps away from it is $(np)^k$. This increases exponentially as k increases. Thus, with high probability, there would be a subgraph which could cover most part of the graph. This is GCC and therefore its size is in **O(n)**.

The curve for p v.s. n is shown below:

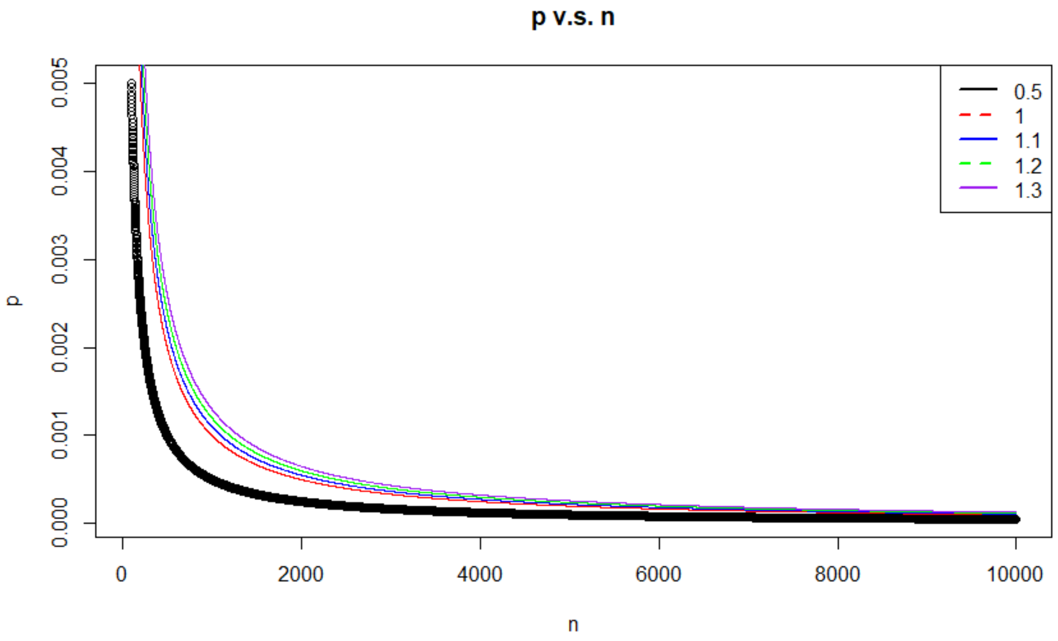


Figure 13. probability v.s number of nodes

1.2 Create networks using preferential attachment model

(a) n=1000,m=1

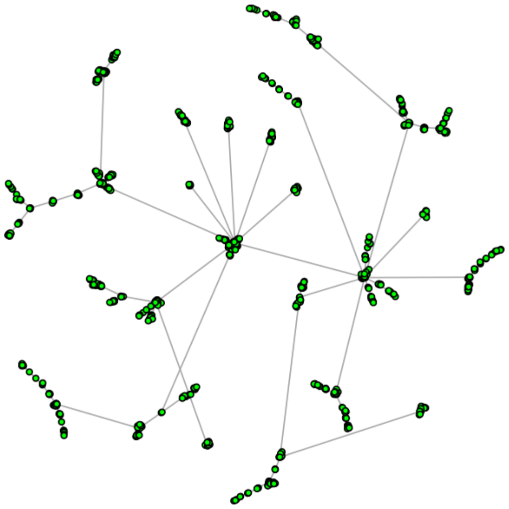
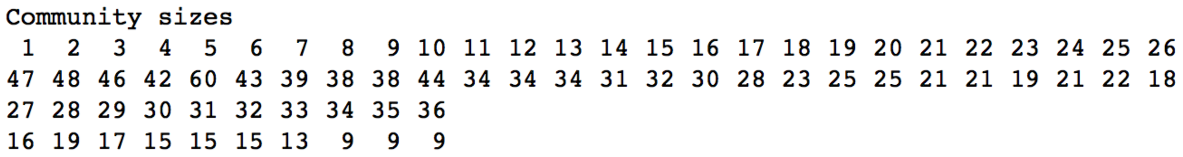


Figure 14. Network created by preferential attachment model(n=1000,m=1)

Based on the mechanism of preferential attachment, such network is always connected.

(b)

Derived by fast-greedy community finding algorithm, there are totally 36 communities in this network. Fig 15 demonstrates the size of each community detected in detail.



```
Community sizes
 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 
47 48 46 42 60 43 39 38 38 44 34 34 34 31 32 30 28 23 25 25 21 21 19 21 22 18 
27 28 29 30 31 32 33 34 35 36 
16 19 17 15 15 15 13  9  9  9 
```

Figure 15. Community Size of network by preferential attachment model(n=1000,m=1)

Modularity is one measure of the structure of networks or graphs It was designed to measure the strength of division of a network into modules.Networks with high modularity have dense connections between the nodes within modules but sparse connections between nodes in different modules.

Here, the modularity of this network is 0.931.

(c)
n=10000, m=1

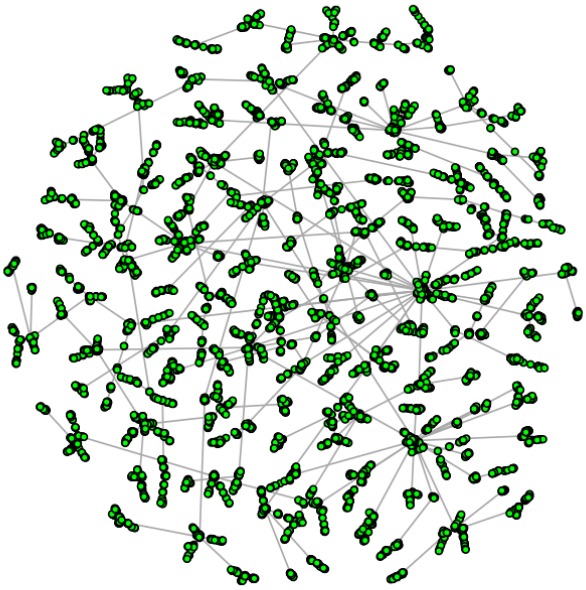


Figure 16. Network created by preferential attachment model(n=10000,m=1)

The modularity of this network is 0.978, which is larger than that of smaller network.

(d)

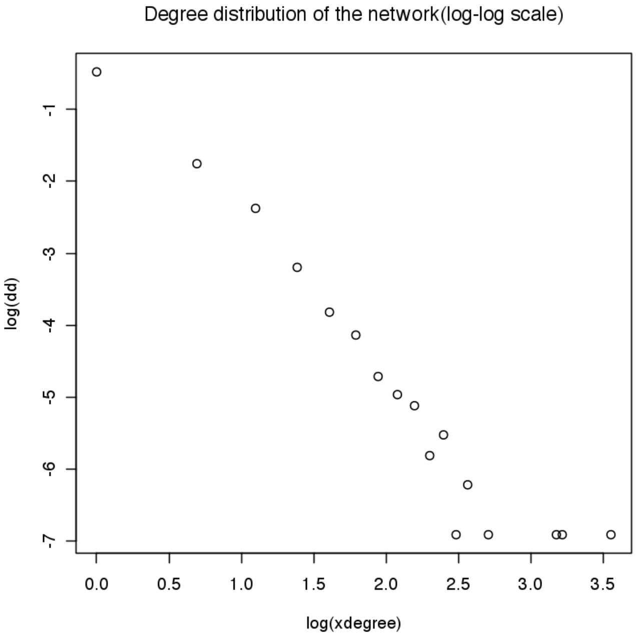


Figure 17. Degree distribution of the network with 1000 nodes(m=1)

The slope of the plot is approximately -2.191.

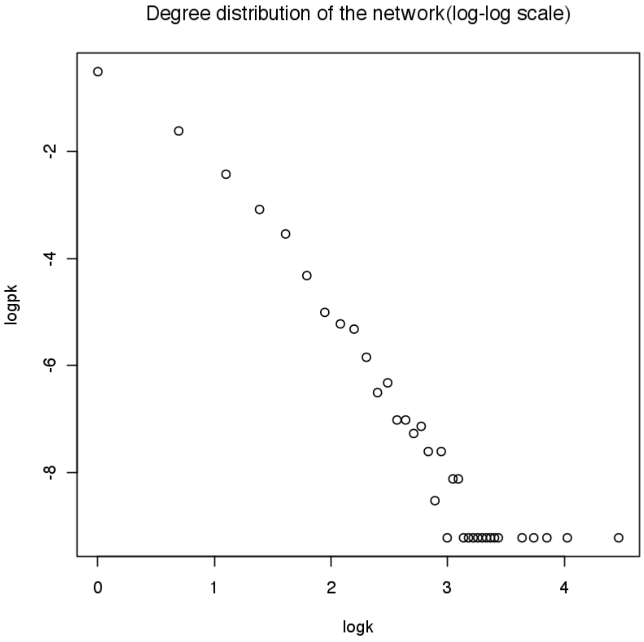


Figure 18. Degree distribution of the network with 10000 nodes(m=1)

The slope of the plot is approximately -2.194.

(e)

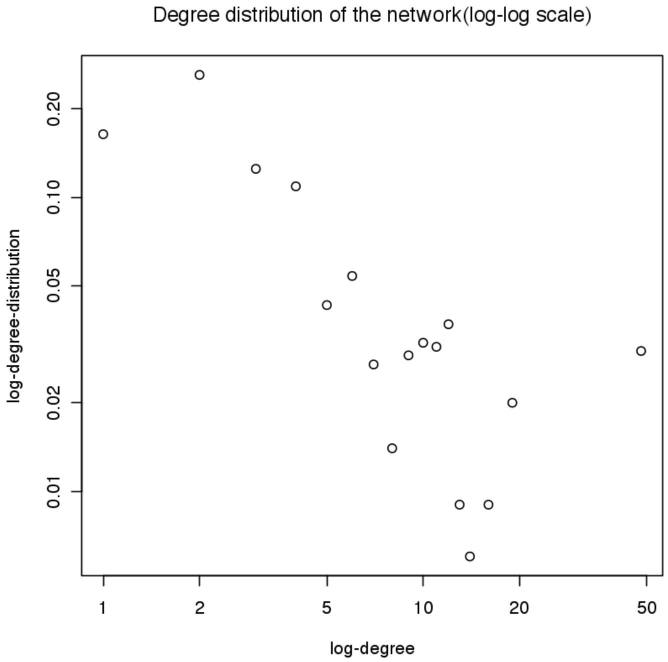


Figure 19. Degree distribution of the node picked as a random neighbor of a random node(m=1)

The relationship has less monotonicity compared with normal degree distribution. In addition, the probability of degree equaling to 2 is larger than that of degree being 1, which is apparently correlated to the tree-like structure of the network generated by preferential attachment model.

(f)

$$E[degree\ of\ node\ added\ at\ time\ i] = m\sqrt{(t/i)}$$

So, based on the relationship between the timestep at which the node is added and its age, we can derive that

$E[degree\ of\ node\ whose\ age\ is\ 'age'] = m * \sqrt{\frac{t}{t-age+1}} = \sqrt{\frac{t}{t-age+1}}$ (t=1,2, ...,1000)

When t=1000, the relationship between expected degree and age of node can be depicted in the figure.

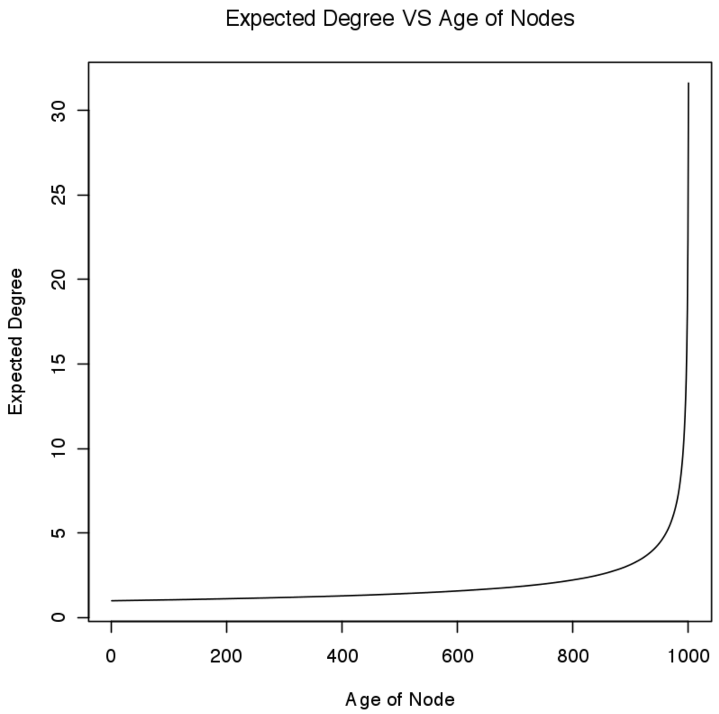


Figure 20. The relationship between expected degree and age of node when timestep=1000 and m=1

(g)
Repeat (a)~(f) for **m=2**

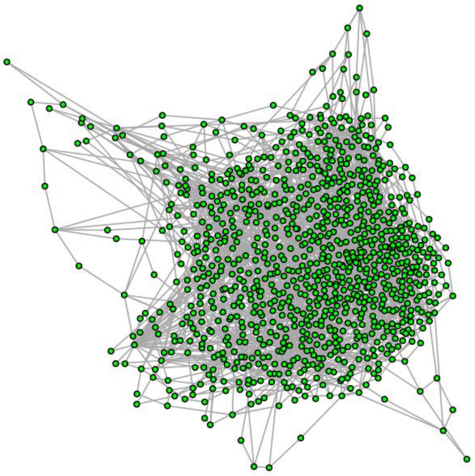
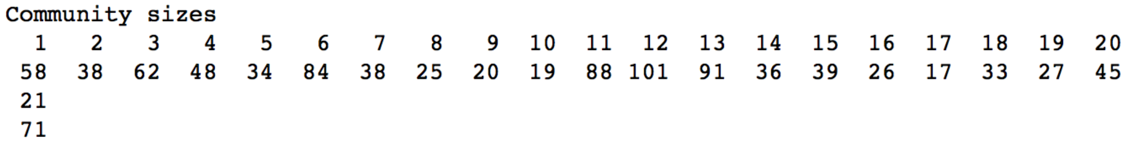


Figure 21. Network created by preferential attachment model(n=1000,m=2)

Fig 22 demonstrates the size of each community detected in detail.



```
Community sizes
  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20 
 58  38  62  48  34  84  38  25  20  19  88 101  91  36  39  26  17  33  27  45 
 21 
 71 
```

Figure 22. Community Size of network by preferential attachment model(n=1000,m=2)

The modularity of this network is 0.526.

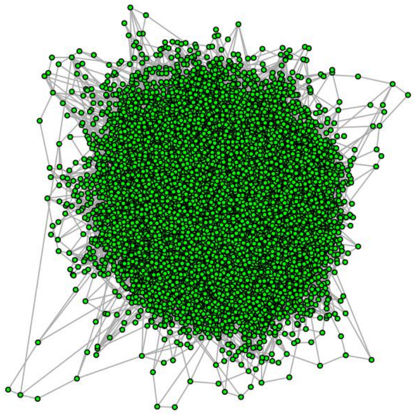


Figure 23. Network created by preferential attachment model(n=10000,m=2)

The modularity of this network is 0.528, which is larger than that of smaller network.

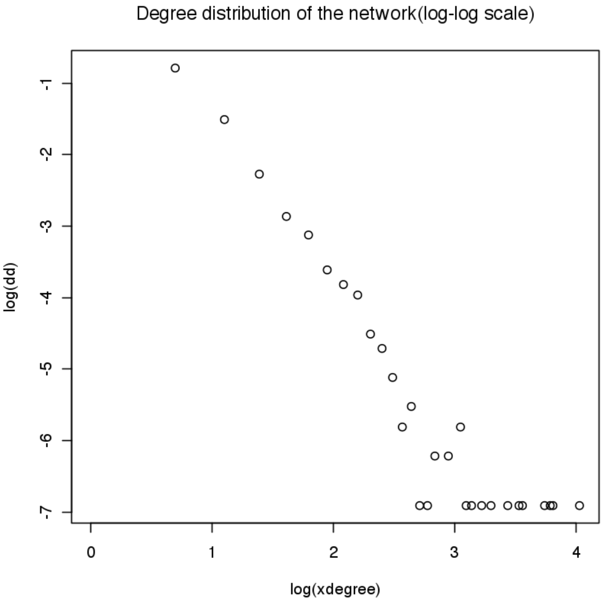


Figure 24. Degree distribution of the network with 1000 nodes (m=2)

The slope of the plot is approximately -2.450.

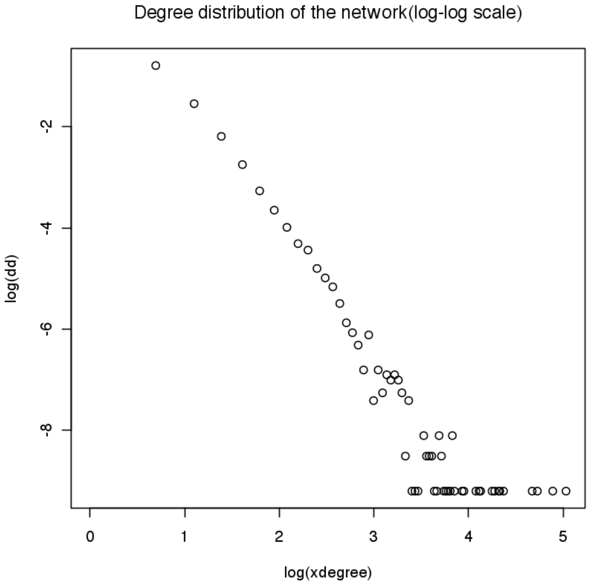


Figure 25. Degree distribution of the network with 10000 nodes (m=2)

The slope of the plot is approximately -2.454.

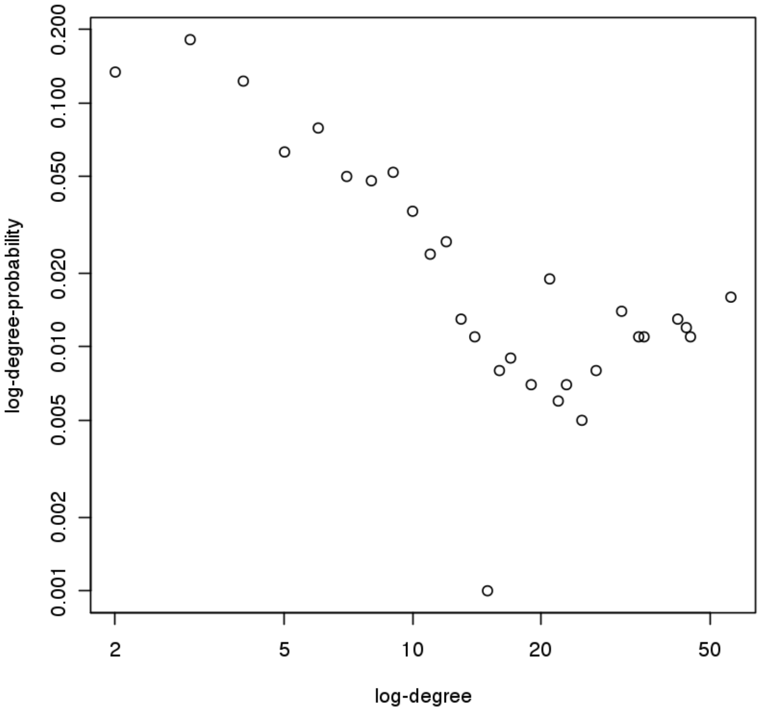


Figure 26. Degree distribution of the node picked as a random neighbor of a random node(m=2)

$E[degree\ of\ node\ whose\ age\ is\ 'age'] = m * \sqrt{\frac{t}{t-age+1}} = 2\sqrt{\frac{t}{t-age+1}}$ (t=1,2, … ,1000)

When t=1000, the relationship between expected degree and age of node can be depicted in Fig 27.

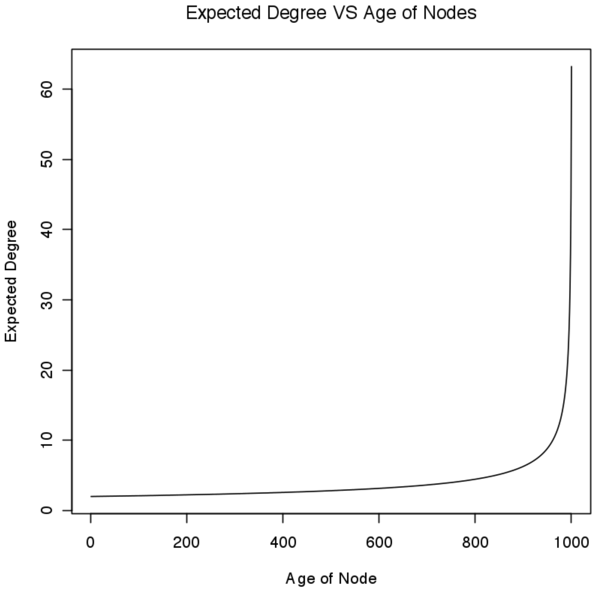


Figure 27. The relationship between expected degree and age of node when timestep=1000 and m=2

Repeat (a)~(f) for **m=5**

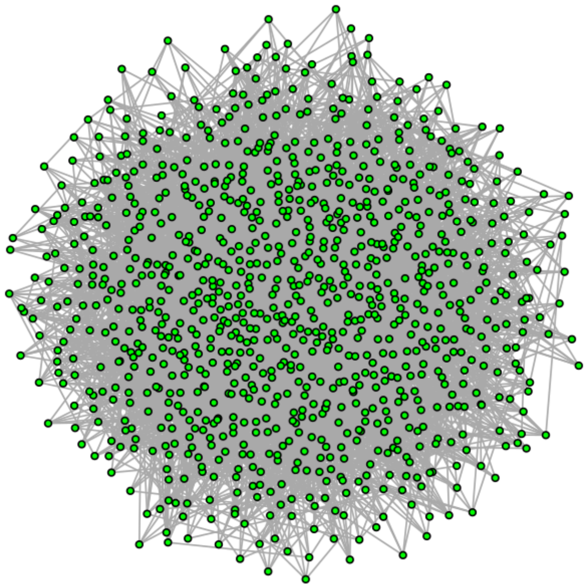
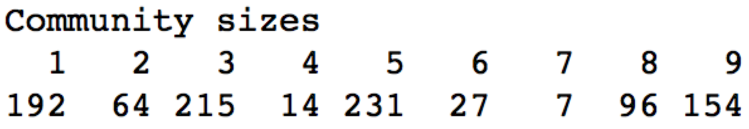


Figure 28. Network created by preferential attachment model(n=1000,m=5)

Fig 29 demonstrates the size of each community detected in detail.



```
Community sizes
  1   2   3   4   5   6   7   8   9
192  64 215  14 231  27   7  96 154
```

Figure 29. Community Size of network by preferential attachment model(n=1000,m=5)

The modularity of this network is 0.278.

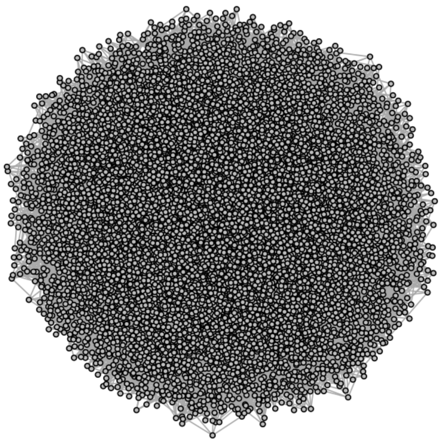


Figure 30. Network created by preferential attachment model(n=10000,m=5)

The modularity of this network is 0.274, which is larger than that of smaller network.

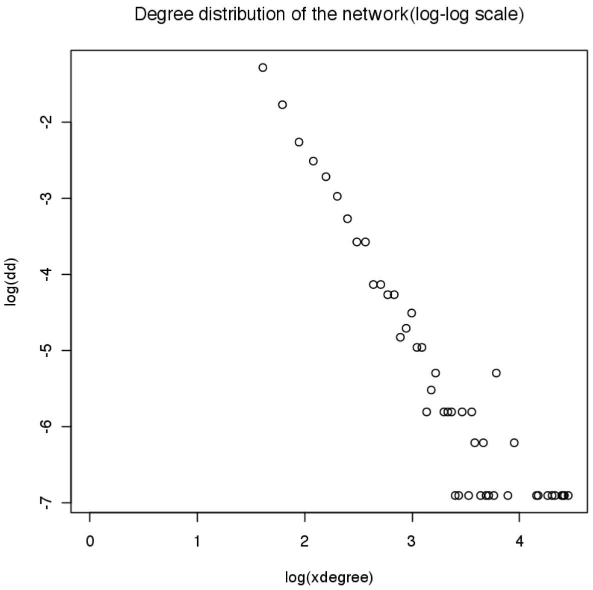


Figure 31. Degree distribution of the network with 1000 nodes (m=5)

The slope of the plot is approximately -2.690.

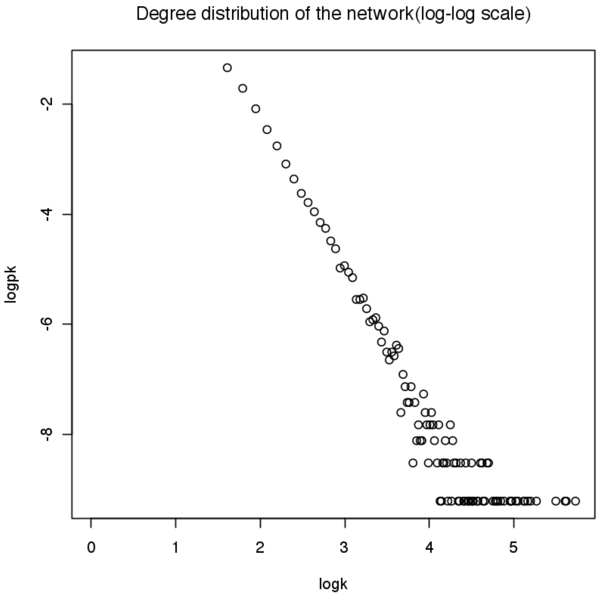


Figure 32. Degree distribution of the network with 10000 nodes (m=5)

The slope of the plot is approximately -2.718.

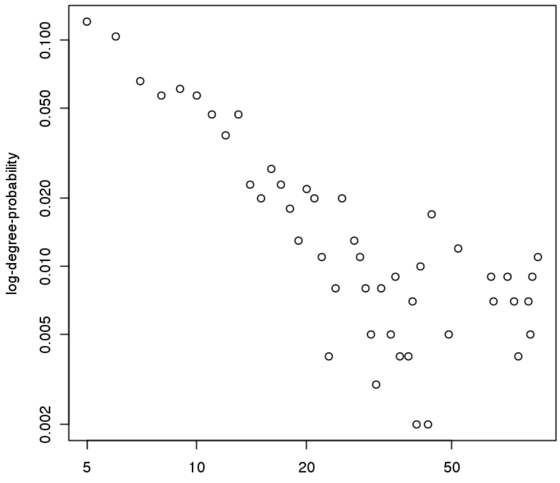


Figure 33. Degree distribution of the node picked as a random neighbor of a random node(m=5)

$E[degree\ of\ node\ whose\ age\ is\ 'age'] = m * \sqrt{\frac{t}{t-age+1}} = 5\sqrt{\frac{t}{t-age+1}}$ (t=1,2,..1000)

When t=1000, the relationship between expected degree and age of node can be depicted in Fig.

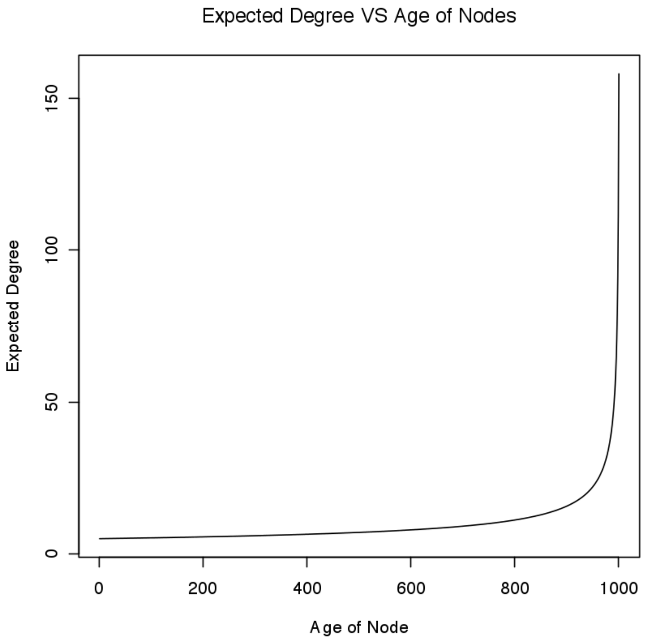


Figure 34. The relationship between expected degree and age of node when timestep=1000 and m=5

Comparison of modularity between m=1,2,5

The modularity when m=1 is highest because the network is prone to form a tree-like structure and in this scenario, large cycles would not exist. The probability of overlapping between distinct communities also diminishes.

(h)

In this section, we calculate the modularity of each network using walktrap community finding algorithm.

Figure 35 is the network generated by preferential attachment model

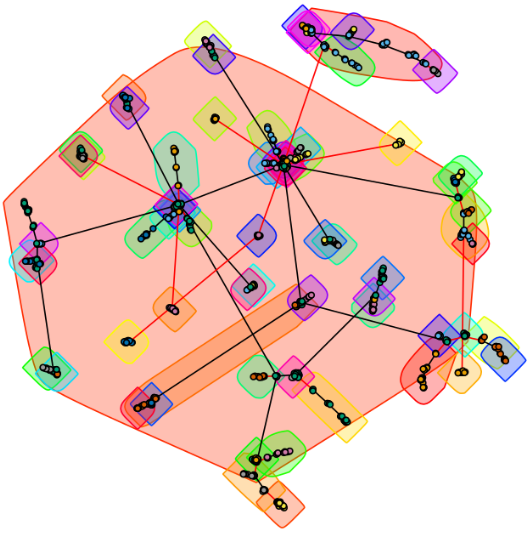


Figure 35. Network generated by preferential attachment model and its community

The modularity of this network is 0.850.

Figure 36 is the network generated by stub-matching with the same degree sequence

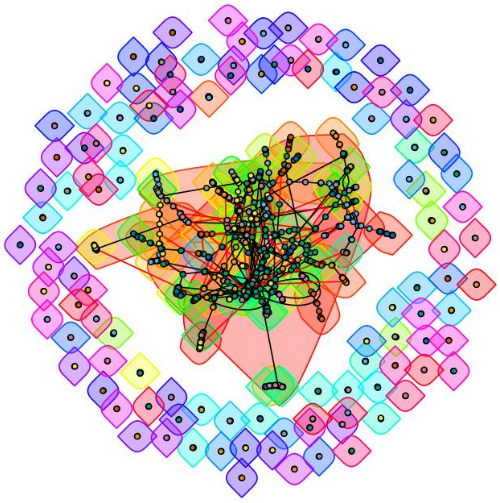


Figure 36. Network generated by stub-matching procedure and its community

The modularity of this network is 0.747.

There are three main differences between two procedures for creating power-low random networks.

1. Before stub-matching procedure is performed, the degree sequence of the network has been configured. However, the degree sequence is dynamically generated during the procedure of preferential attachment.

2. It is likely that there are multiple edges between two nodes for stub-matching procedure since there is no prohibition that a node cannot directly connect to another node twice.

3. The network generated by preferential attachment is always connected. By contrast, the network created by stub-matching is not necessarily connected all the time.

1.3 Create a modified preferential attachment model that penalize the age of a node
(a)

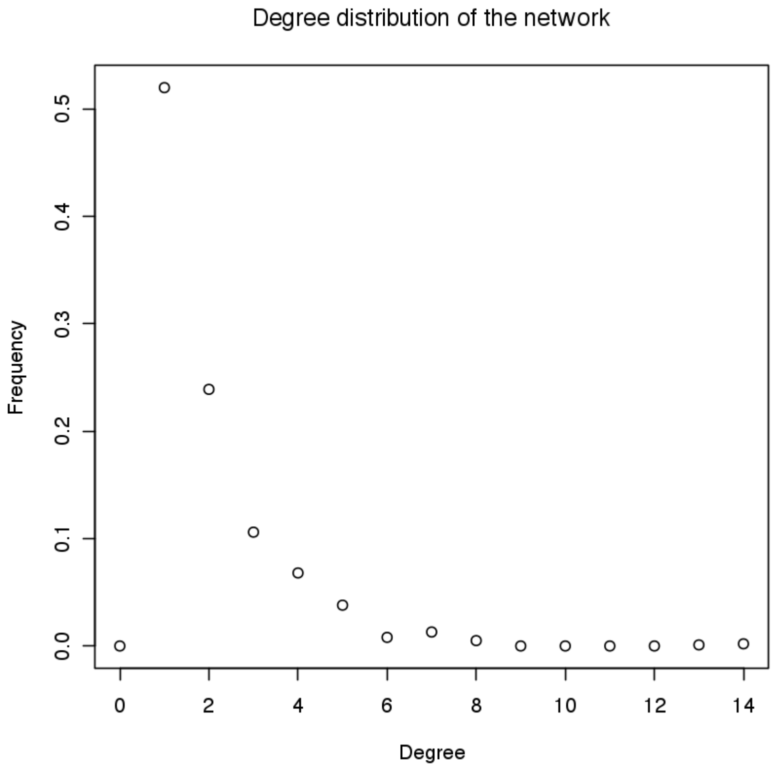
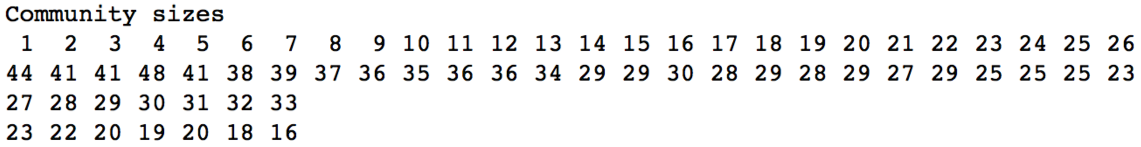


Figure 37. Degree Distribution of Network generated by PA which penalizes the age (n=1000, m=1)

The power-law exponent is 2.092.

(b)

Figure 38 demonstrates the size of each community detected in detail.



```
Community sizes
 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 
44 41 41 48 41 38 39 37 36 35 36 36 34 29 29 30 28 29 28 29 27 29 25 25 25 23 
27 28 29 30 31 32 33 
23 22 20 19 20 18 16 
```

Figure 38. Community Size of Network by PA which penalizes the age (n=1000, m=1)

The modularity of this network is 0.935.

II. Random Walk on Networks.

2.1 Random walk on Erdös-Rényi networks

(a)

We created the undirected E-R model graph as instructed.

The graph was a connected graph with the diameter 4.

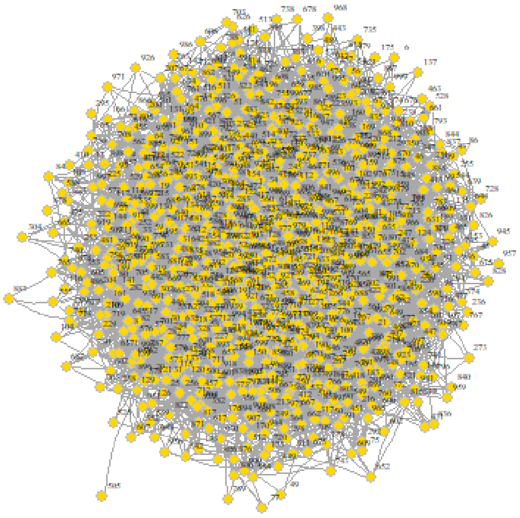


Figure 39. ER model with G(n,p) = G(1000,0.01)

(b)

We iterated t from 1 to 100, beginning with a random start node, and calculated the shortest distance from it to a tail node through random walk. We experimented it 100 times for each step size t and plotted the mean and standard deviations for each step sizes.

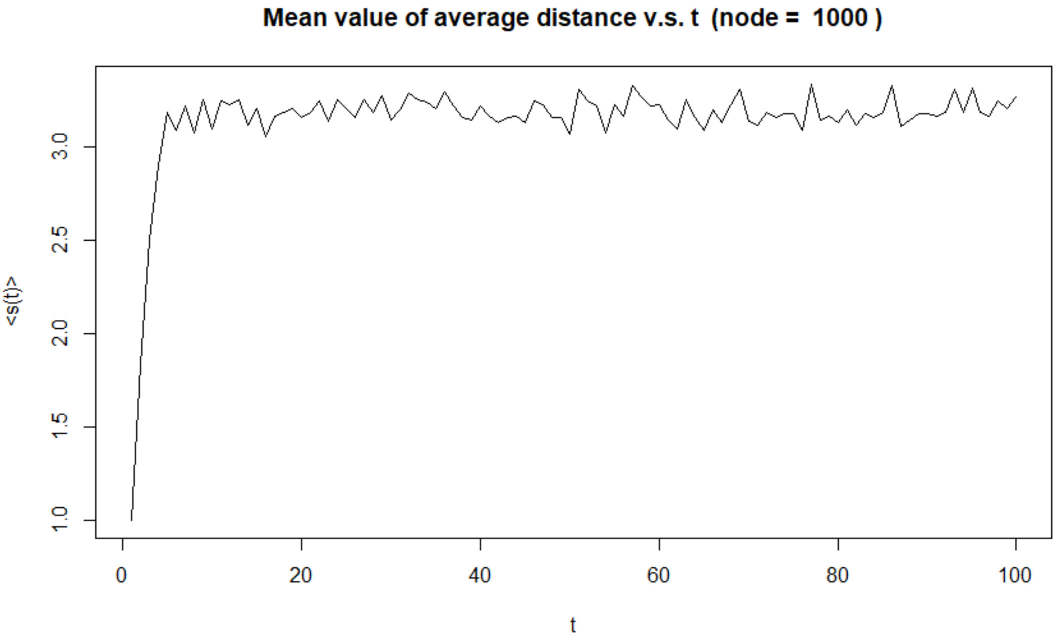


Figure 40. Mean value of distance <s(t)> v.s. t

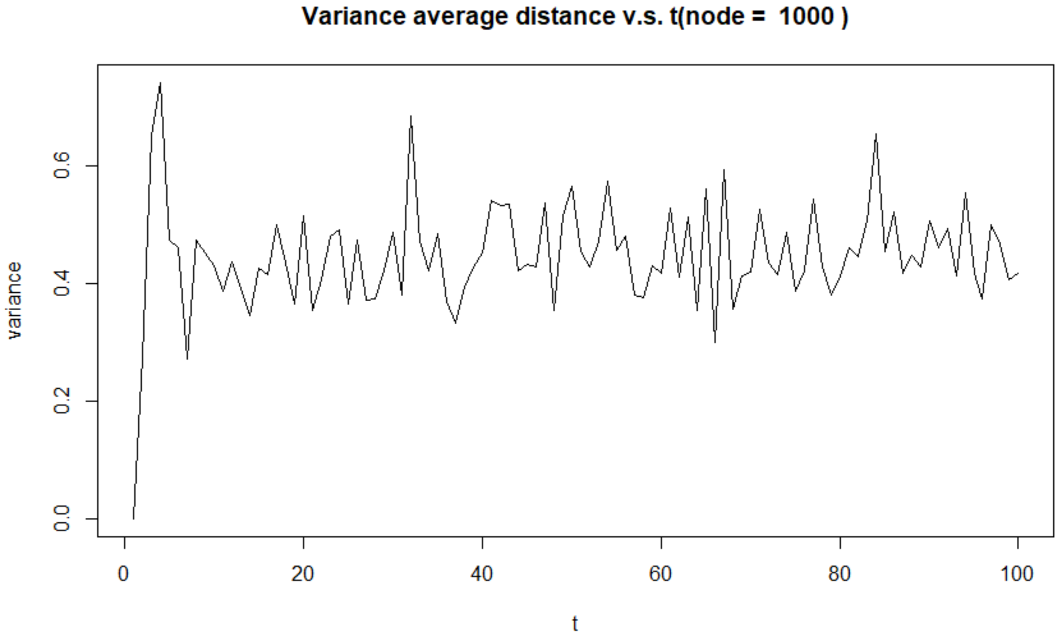


Figure 41. standard deviation of distance v.s. t

(c)

We experimented 100 times for the degree of the last node for a random walker for the graph G(1000, 0.01). Actually, as proved in lecture, O(ln(n)) steps are enough for the result to converge. The degree distribution is shown as below:

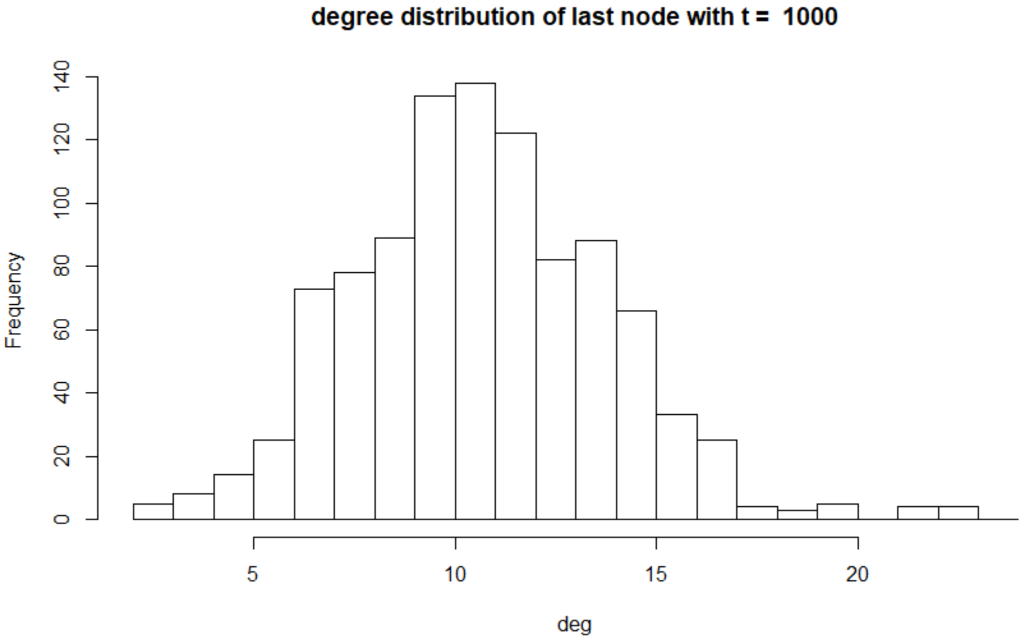


Figure 42. degree distribution of the end node

The distribution is largely similar with the degree distribution of nodes for the same graph, as is shown in part 1-1. They both have a mean value at degree 10, which is equal to np = 10.

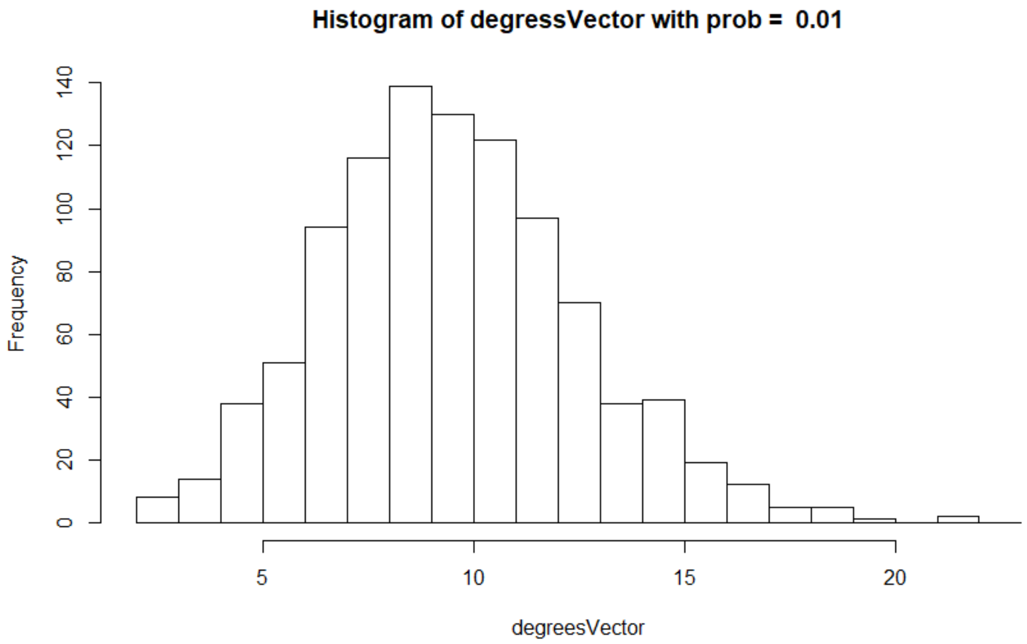
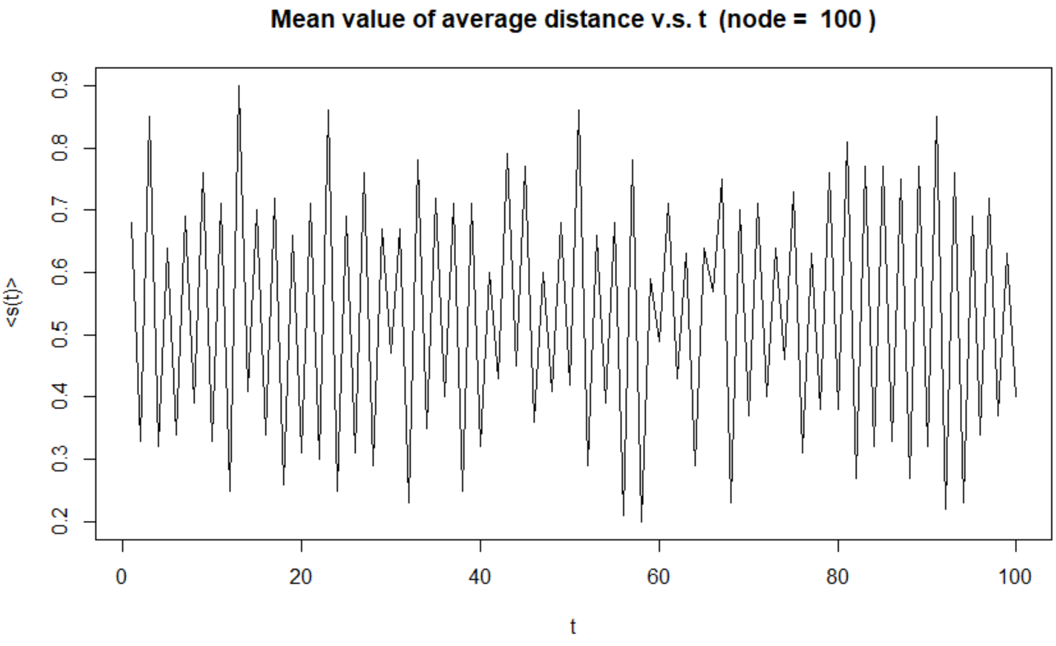


Figure 43. degree distribution for a G(n,p) = G(1000, 0.01)

It is proved in the lecture that when reaching a steady state, node occupancy probabilities vector is the same as the node-picked possibility vector, which is a binomial distribution B(n,p).

(d)

By changing the node number to 100 and 10000, we got their average distance and deviation for distance separately.



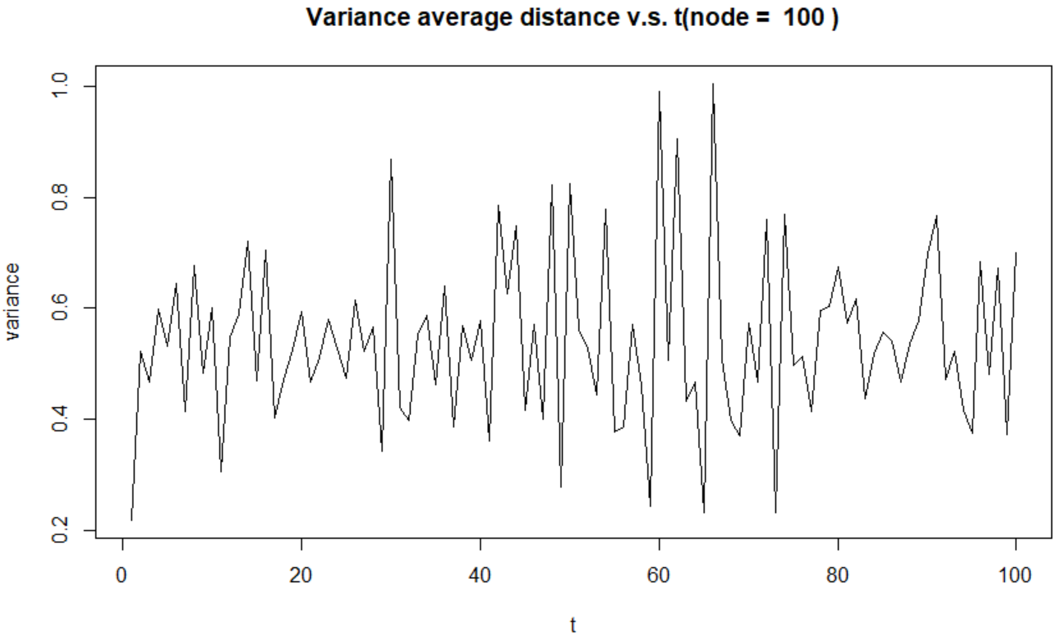
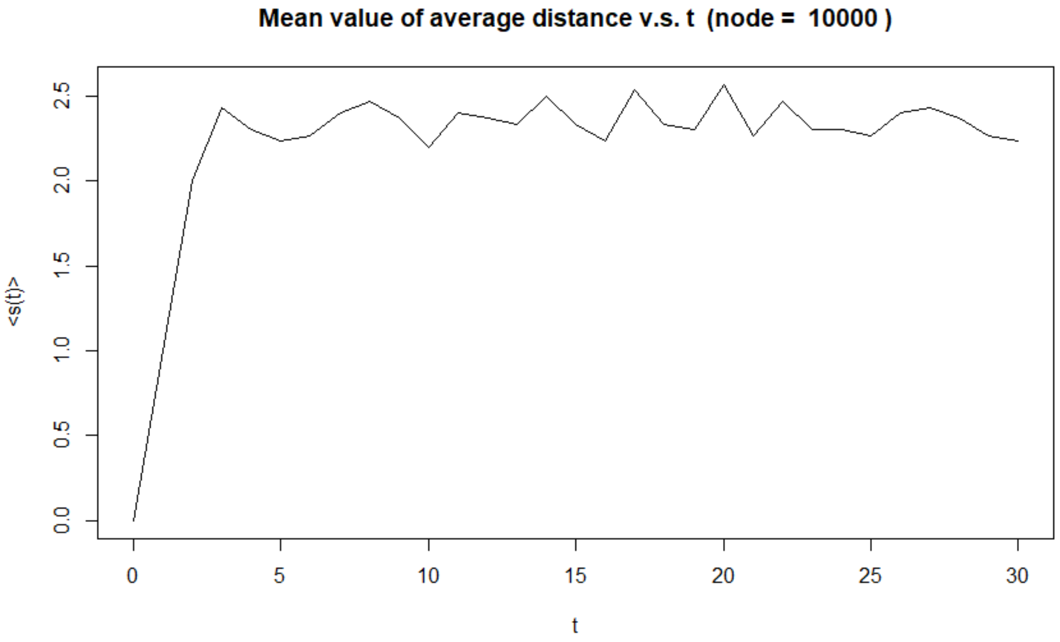


Figure 44. Average and deviation for distance v.s. t at node = 100



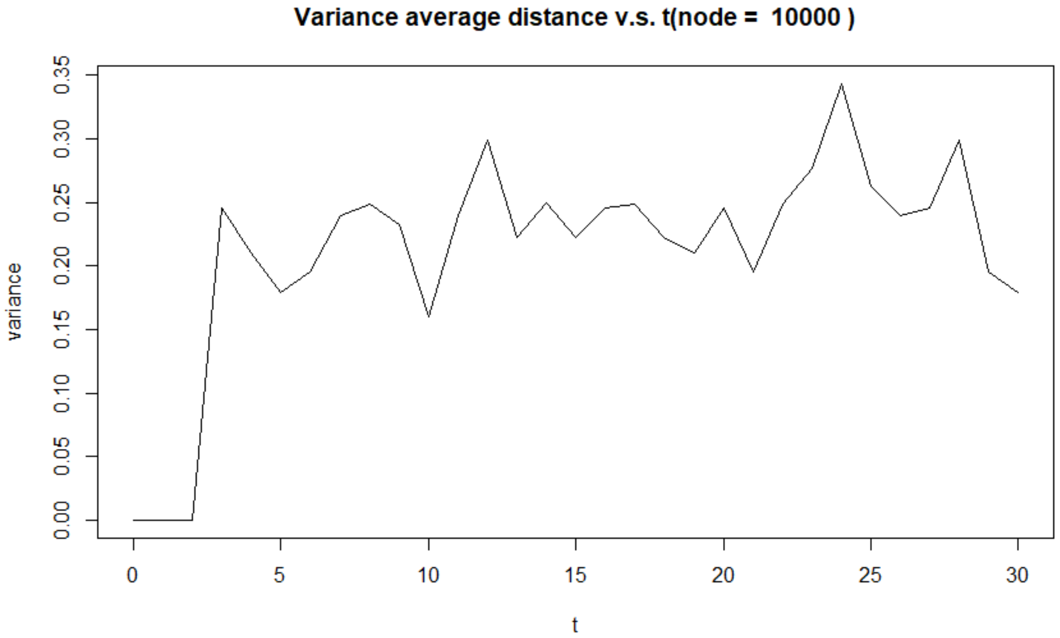


Figure 45. Average and standard deviation for distance v.s. t at node = 10000

For node = 10000, because of the big computational complexity, we shrink the t to 30. Actually, the curve reached its steady point at step = 4, which converges to about 2.5. Therefore, t = 30 is sufficient.

From the results, we could see that for node = 10000, the walker could converge to a steady point where the average of distance is more steady and standard deviation is less (around 0.25). By comparison, for node = 100, the results seemed to be more random, with std deviation = 0.6.

For the case n = 100, np = 1, which is actually a turning point that GCC started to emerge. This means the graph may still contain large amount of isolated nodes or small clusters of nodes. Therefore, it is not difficult to see the average of distance is almost less than one, and the standard deviation is not steady.

For the case n = 10000, np=100>>1, which means the graph is connected with super large possibility. Thus, the walker would finally converge to a point, at which degree is consistent with the degree distribution of the whole graph. Plus, the average distance should be consistent with the diameter of the graph (GCC), which was 3 as we tested.

We recorded the diameter for the three situations with node number = 100, 1000 and 10000.

Table 3. Node Number and diameters

Node Number	Diameter
100	7
1000	4
10000	3

Compared the previous results, we could conclude that the **converged average distance for node number =1000 and 10000 are consistent with the diameter of that graph**, while for the case node number = 100, diameter was not a good indicator. The reason is that for both case of node number = 1000 and 10000, the whole graphs are connected, so that the distance could converge to diameter. For case of node = 100, however, the graph is not connected and there were many isolated nodes or small clusters of nodes. The diameter for GCC could not indicate well the distance because the chosen start node could have large possibility to be any node outside GCC.

2.2 Random walk on networks with fat-tailed degree distribution

(a) Preferential Attachment Network Generation (n=1000)

As shown in Fig 45, a preferential attachment network is generated with 1000 nodes, where each new node attaches to m=1 old nodes.

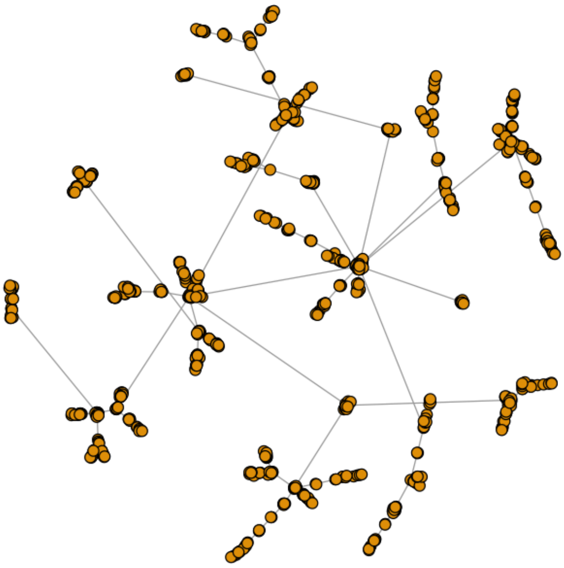


Figure 46. Preferential Attachment Network with 1000 nodes

(b) Random Walk on Preferential Attachment Network (n=1000)

In this section, the average distance (donated as $< s(t) >$) and standard deviation (donated as $\sigma^2(t)$ of random walk on preferential attachment network is measured, where the average is over random choices of the starting nodes. This experiment is set as following:

1) Sample 100 starting points. From each starting point,
2) Random walk on step(s) 1 to 100

The result is shown in Fig. It can be concluded that $< s(t) >= O(log(t))$ and $\sigma^2(t) = O(log(t))$

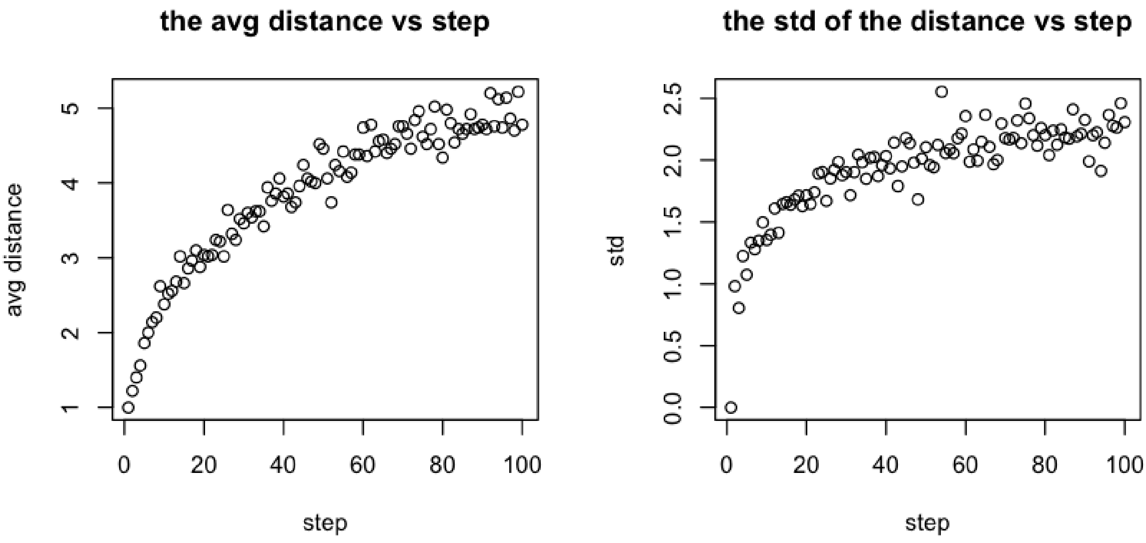


Figure 47. Average Distance vs Steps (left) and Standard Deviation (right) of 1000 nodes

(c) Degree Distribution on Preferential Attachment Network (n=1000)

In this section, the degree distribution of the nodes reached at the end of the random walk on this network is measured. This experiment has the same setting as in 2.2.2. Comparing with the degree distribution of the graph (shown in Fig 47. right), this distribution (shown in Fig 47. left) is also matched with power law distribution. According to the law of large numbers, if this experiment is repeated much more times, the distribution may become indistinguishable from the degree distribution of the graph.

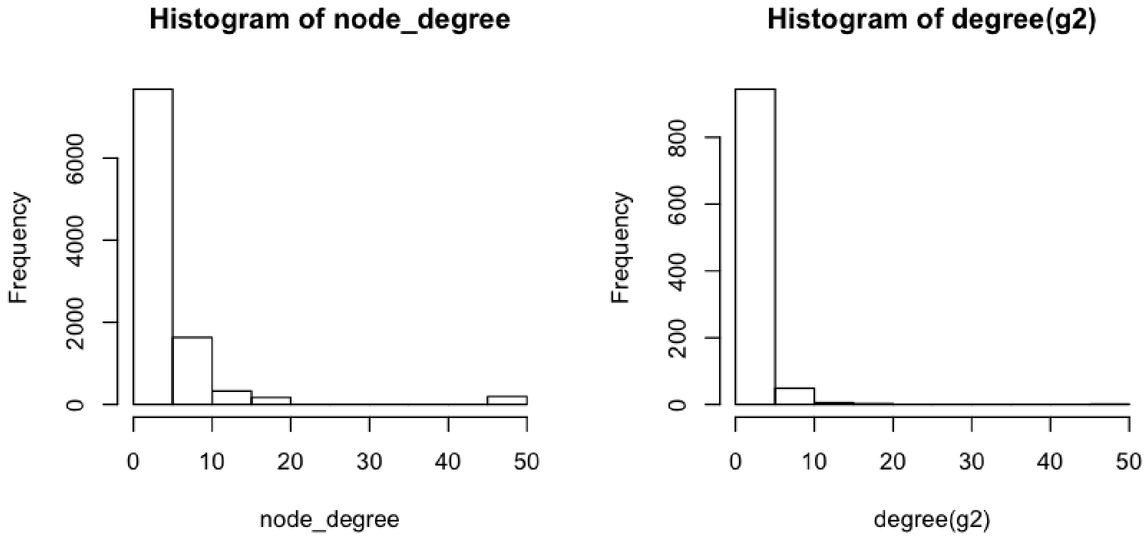


Figure 48. Random Walk Degree Distribution (left) and Graph Degree Distribution (right) of 1000 nodes

(d) Preferential Attachment Network Generation (n=100 & 10000)

In this section, $< s(t) >$and $\sigma^2(t)$ of preferential attachment network n=100 and n=10000 are measured. As shown in Fig 48 and Fig 49, two networks with 100 and 10000 nodes, and m=1, are generated. The setting of experiments is similar to (b). However, on preferential attachment networks with 10000 nodes, due to the limitation of computation ability, random walks start from 30 random selected nodes. The results of two experiments are shown in Fig 50 and Fig 51. Both are matched with $< s(t) >= O(log(t))$and $\sigma^2(t) = O(log(t))$, and the diameter of the network does not influence the complexity. However, the network with 10000 nodes has a low rate of convergence comparing the network with 100 nodes. Similar to the degree distribution in random walk, the steps t should ∝ diameter of networks.

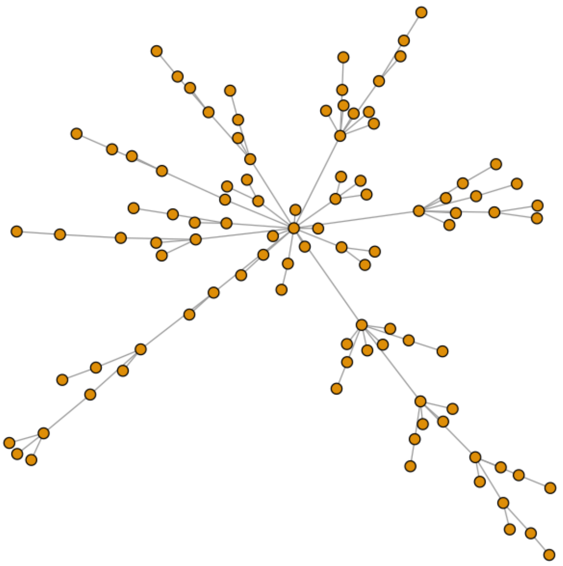


Figure 49. Preferential Attachment Network with 100 nodes

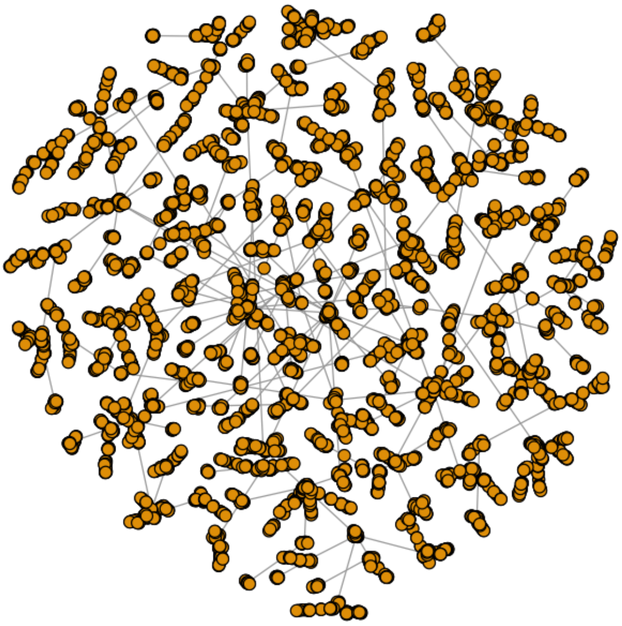


Figure 50. Preferential Attachment Network with 10000 nodes

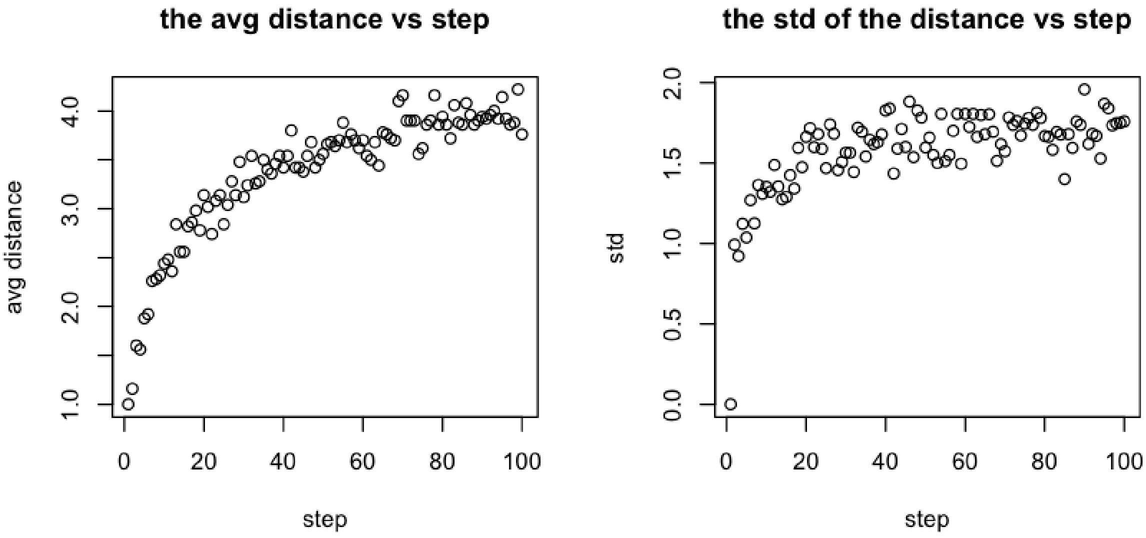


Figure 51. Average Distance vs Steps (left) and Standard deviation (right) of 100 nodes

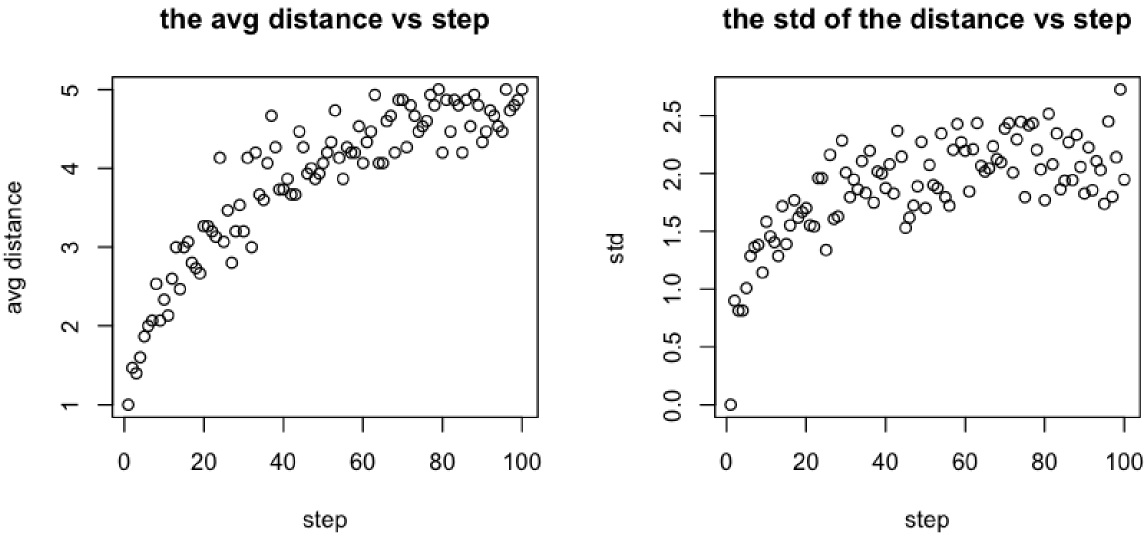


Figure 52. Average Distance vs Steps (left) and Standard deviation (right) of 10000 nodes

2.3. PageRank

In this section, we would like to use random walk to simulate PageRank algorithm. In 2.3(a), we simulated PageRank with random walk in a directed preferential attachment network. In 2.3(b), we modified random walk with teleportation and measured the correlation between vertex visiting probability and degree.

(a) Basic random walk setup

According to problem statement, we first generated a directed random network with preferential attachment model, with 1000 vertices, m=4. Fig. 52 illustrates the network. Noting that in a directed preferential attachment network, the first node has out-degree of zero, which means the 1# vertex is a dead node.

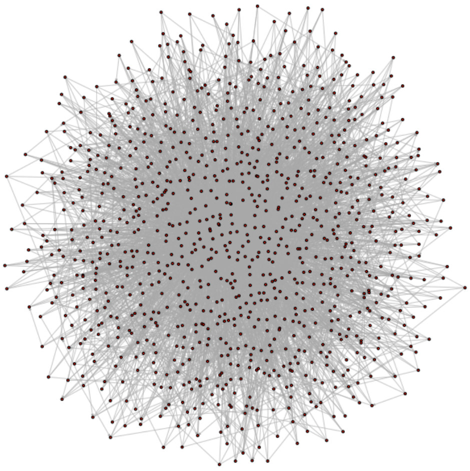


Figure 53. Illustration of preferential attachment network

Here, in a random walk, we set start node as random and walk steps as 1000. By repeating the experiment 10,000 times, we then have sufficient data and could measure the vertices visiting probability by calculating the frequency count that the last step of each random walk. The probability scattering against vertex degree is shown in Fig. 53. From the figure, we can see that all there is only one data point in the figure with probability of 1, which means that all random walk ended up in a vertex.

This is a not surprising result, as discussed above, vertex 1# has an out-degree of 0, and all random walks would go to this vertex and get trapped in several steps, whatever the start vertex is. In conclusion, in this problem, we do not have sufficient proof to say there is a correlation between visiting probability and vertex degree, since all walks ended up in one vertex.

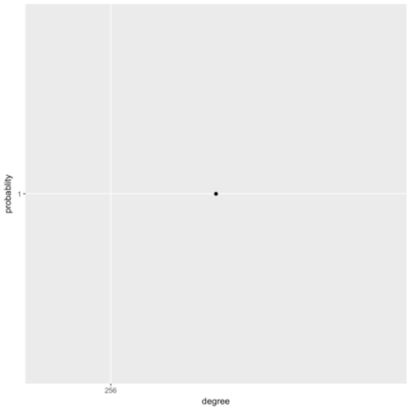


Figure 54. Probability distribution of visiting against vertex degree

(b) Random walk with teleportation

In this section we added teleportation to the random walk. The teleportation is an analog of real network. The intuition of teleportation is that, if a random walk meets a vertex without-degree=0, jump to a random vertex; in each transaction process, with probability α, the random walk would jump to a random vertex instead of its proceeding neighbors.

After repeating the experiment 10,000 times with teleportationα=0.15, we got the result shown in Fig. 54. In Fig. 54, the data points are real data, and the blue line with grey shaded region is their linear regression with 95% confidence interval.

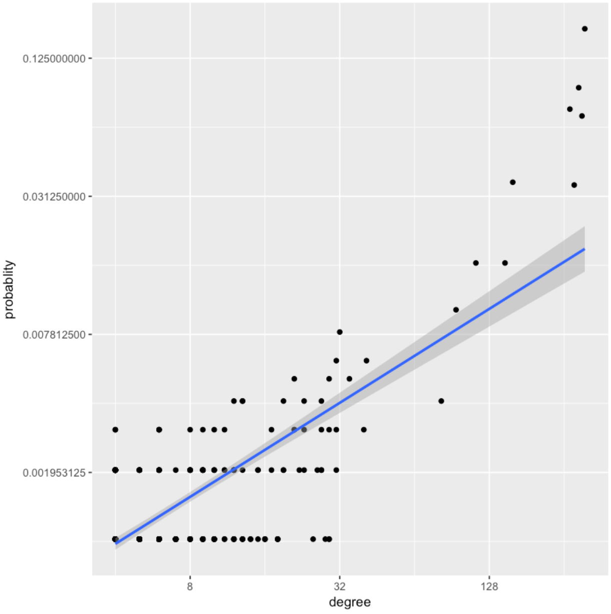


Figure 55. Probability distribution of visiting against vertex degree, with teleportation α=0.15

We used pearson correlation test to measure the vertex degree and vertex visiting probability. The result turns out to be that the Pearson's product-moment correlation between vertex degree and visiting probability is 0.835, with p-value < 2.2e-16. Based on this, we can conclude that vertex degree and visiting probability has linear correlation.

2.4 Personalized PageRank

In this part, we implemented personalized PageRank and compared with regularized PageRank, and discussed other scenarios with personalized PageRank algorithm.

(a) Comparison with regular PageRank

In this section, we first generate a random walk with teleportation with α=1/N=0.001, as a baseline of regularized PageRank. After that, we used a Personalized PageRank algorithm instead, whose α is in proportional to its regularized PageRank (in this problem, we set coefficient of 0.5), and compared the visiting probability with both cases. Fig. 56 shows the scattering plot of Regularized PageRank versus Personalized PageRank. The blue line with grey area is their linear regression with 95% confidence interval. From the plot we can see that, with PageRank with higher probability, Personalized PageRank tends to be higher than Regularized PageRank.

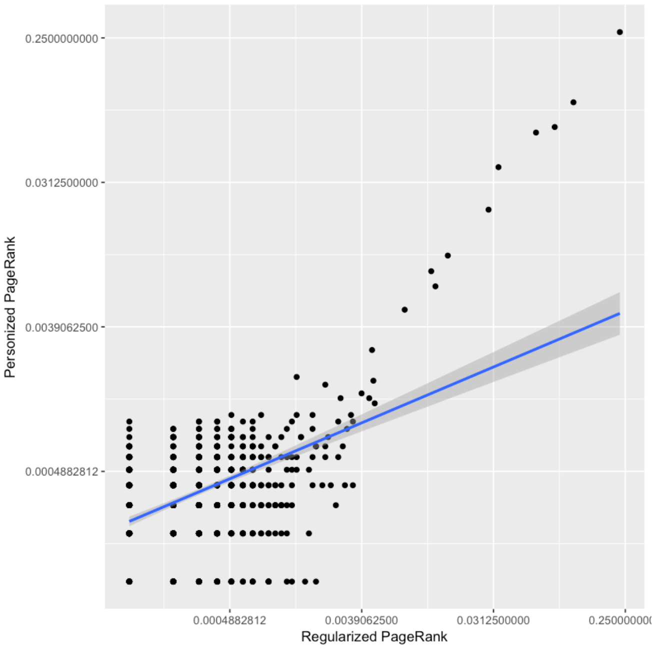


Figure 56. Regularized PageRank versus Personalized PageRank

We then calculated the absolute value of the difference between Regularized PageRank and Personalized PageRank, and compared it with vertex degree, since in the previous question, we have shown that vertex has a higher degree is more likely to have a higher visiting probability. The plot is shown in Fig. 57. From the plot we can see that, in Personalized PageRank the probability is more likely to be affected when the vertex degree is high.

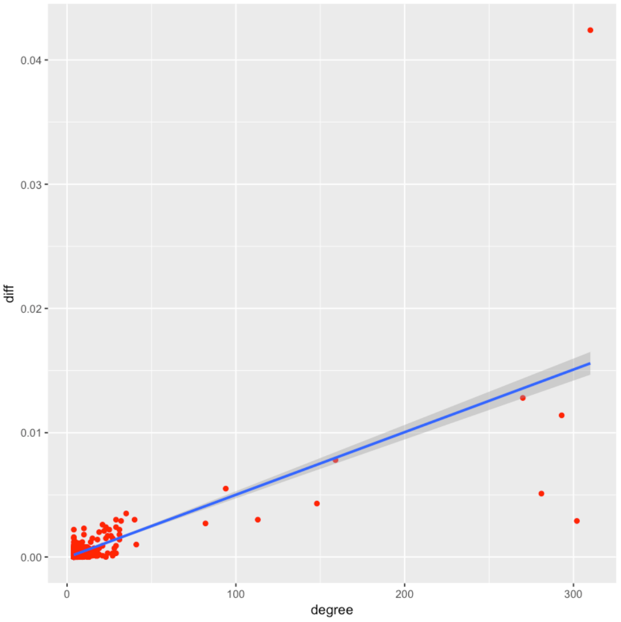


Figure 57. Difference between Regularized PageRank and Personalized PageRank versus vertex degree

We also did Pearson Correlation Test between the two variables. The result turns out to be that the correlation value is 0.733 with p-value 2.2e-16, which can also verify the assertion above. A probable reason behind can be explained that, in personalized PageRank, a vertex has a higher PageRank has more teleportation α, and is more likely to jump to a random vertex, instead of its neighbors, who may have relatively high visiting probabilities. This phenomenon devotes a lower Personalized PageRank with higher vertex degrees.

(b) Teleportation only in median PageRank

In this section, teleportation only jumps to 2 vertices which have the median probability (if it travels to a dead node, jumps to one of the 2 vertices). In the graph we generated, the vertices have median visiting probability is vertex 364 and 370. Fig. 58 shows the new visiting probability and shows the difference with that in 4-1. From the plot we can see that the visiting probability of 2 median vertices has been enlarged, in comparison to other nodes.

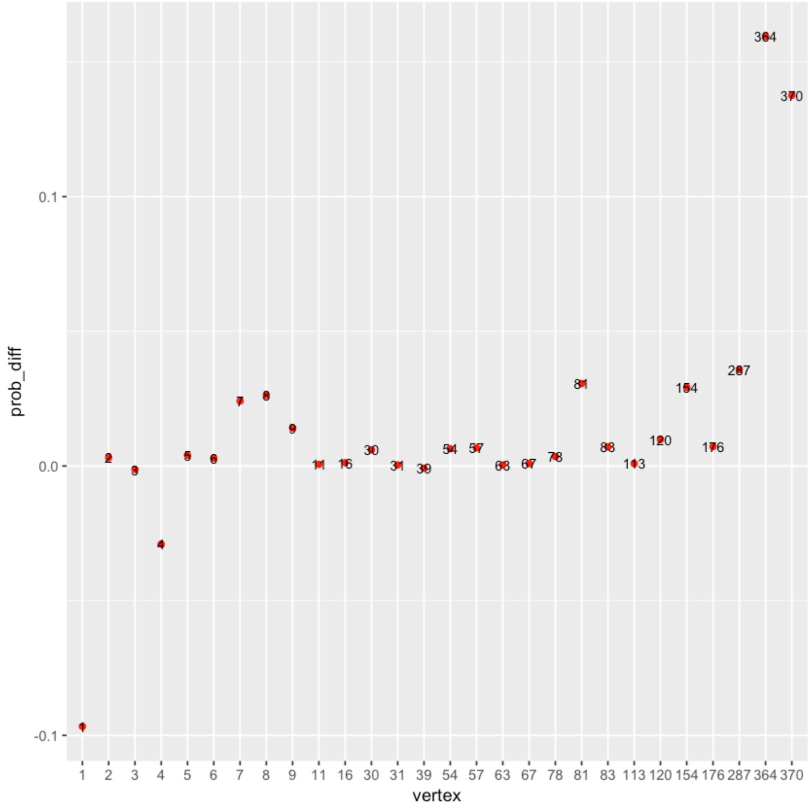


Figure 58. Probability difference with Personalized PageRank

(c) Self-reinforcement PageRank equation

According to problem statement, we can adjust the transition probability to the equation below. In this equation, we changed the teleportation visiting probability from 1/N into personalized PageRank.

A personalized view of PageRank PG can be interpreted as the equation follows:

$$PG = [1 - \alpha] \cdot PG \cdot T + \alpha\sigma$$

Where α is teleportation probability, and σ is a non-uniform preference vector specific to a user, with $\sum \sigma = 1$. T is for transition matrix.

We can do random walk and update the PageRank with transition probability depicted above recursively, and update PageRank in a self-reinforcement fashion every time we did a batch of random walk and get the PG with visiting probability.