Answers to Project Sheet 2, Modelling Complex Systems

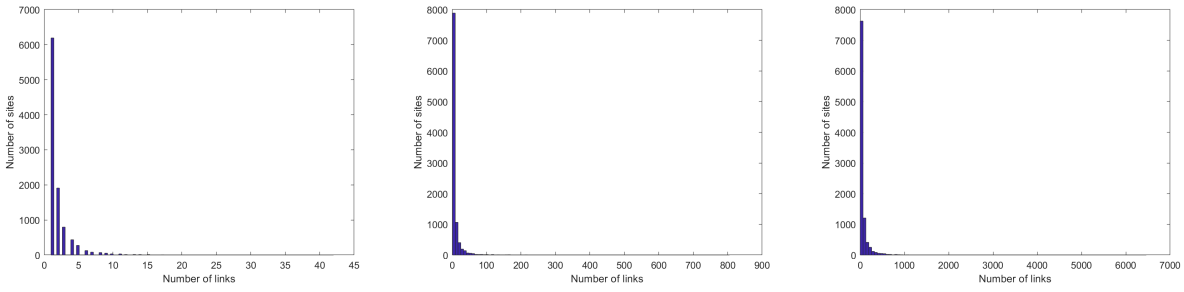
Fredrik Gustafsson

May 2019

1 Growth of World Wide Web

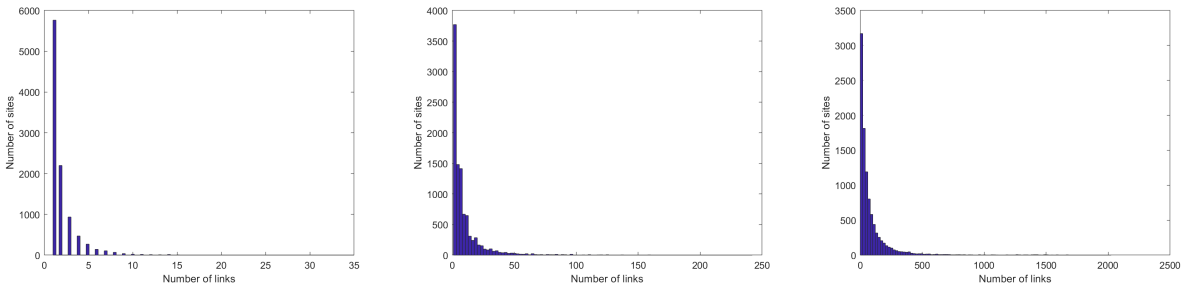
1.1 Not Caring about Who Links to Whom

In this part we increase the number of sites by one in each time step. Each new site will then have m links to other sites, a site can link more than once to the same site. The sites to link with was chosen at random with probability p and chosen in proportion to already existing links with probability 1-p. The proportion probability was calculated as $\frac{k_i}{\sum_{j=1}^{t+1} k_i}$. In order to investigate the effects of m and p on the final distribution, plotted as a histogram and as a loglog plot, both were varied. The result, as histograms, are show in figure 1-3 below and as loglog plots in figure 4 -6. For each simulation 10000 steps were taken with an initial number of sites being 100.



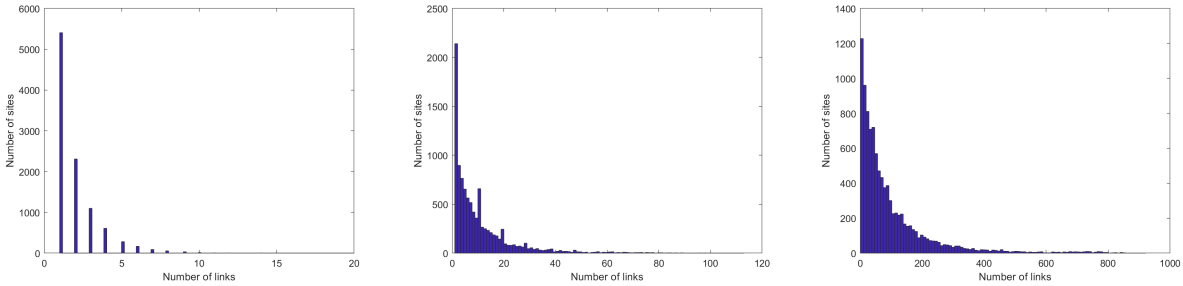
(a) p = 0.25 m = 1 (b) p = 0.25, m = 10 (c) p = 0.25 m = 100

Figure 1: Histograms for varying values of m with p = 0.25



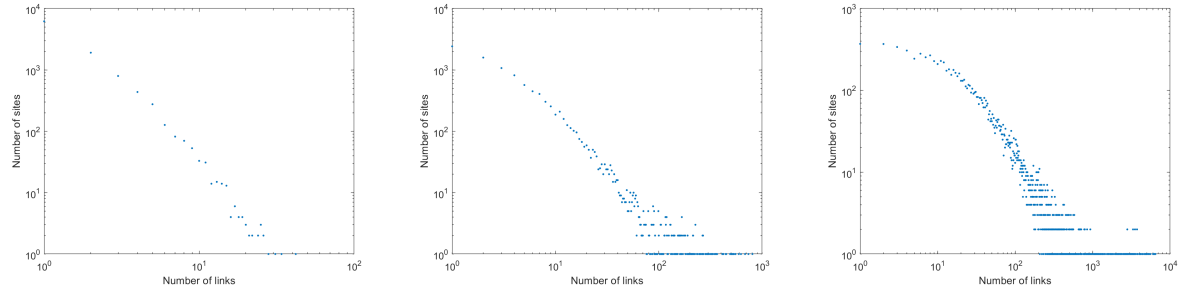
(a) p = 0.5 m = 1 (b) p = 0.5, m = 10 (c) p = 0.5 m = 100

Figure 2: Histograms for varying values of m with p = 0.5



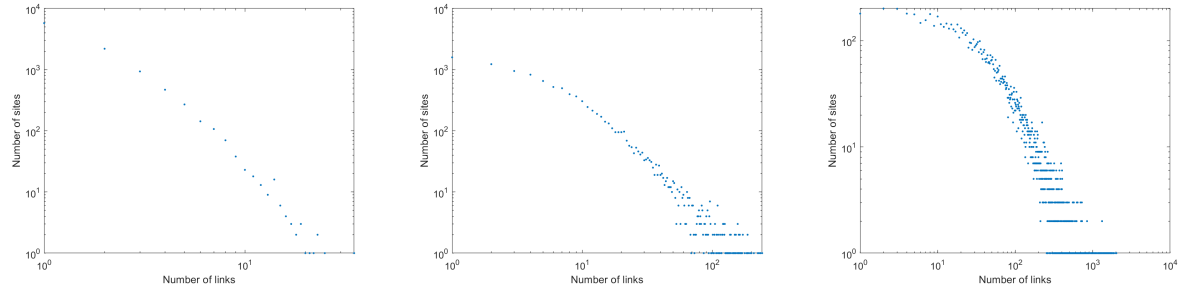
(a) p = 0.75 m = 1 (b) p = 0.75, m = 10 (c) p = 0.75 m = 100

Figure 3: Histograms for varying values of m with p = 0.75



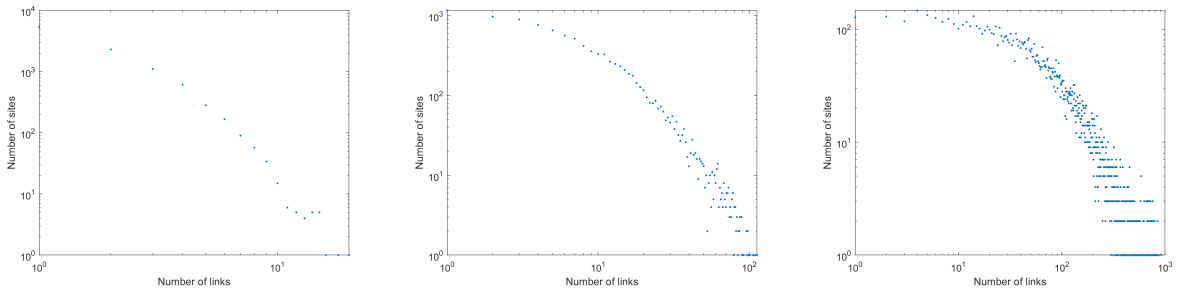
(a) p = 0.25 m = 1 (b) p = 0.25, m = 10 (c) p = 0.25 m = 100

Figure 4: Loglog plot for varying values of m with p = 0.25



(a) p = 0.5 m = 1 (b) p = 0.5, m = 10 (c) p = 0.5 m = 100

Figure 5: Loglog plot for varying values of m with p = 0.5



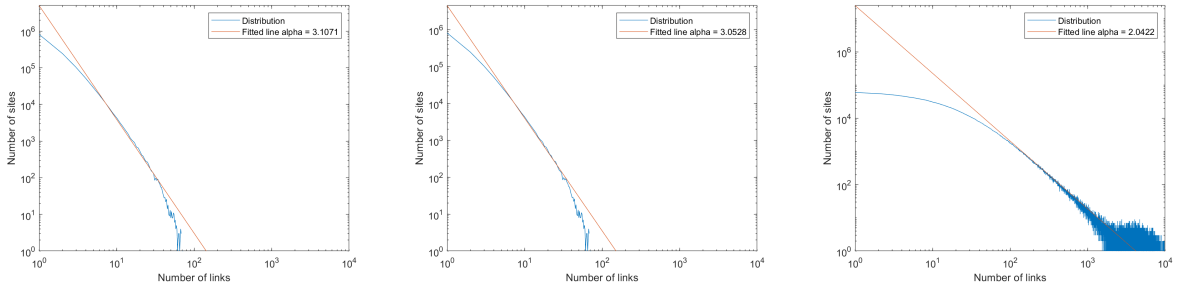
(a) p = 0.75 m = 1 (b) p = 0.75, m = 10 (c) p = 0.75 m = 100

Figure 6: Loglog plot for varying values of m with p = 0.75

As we can see from figure 1-3 and 4-6 the distribution appears to approach the power distribution for lower values of p while it appears as the negative exponential distribution or geometric distribution. One can also see, unsurprisingly, that with a higher m value the spread in regards to how many links websites gets increases, shown as more bars in the histogram plots or as the width of the loglog plot.

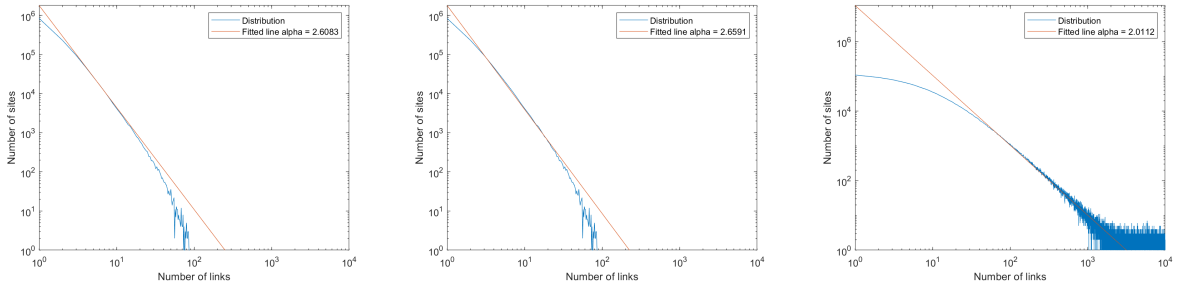
1.2 Fitting a Line to the Power Law

In order to try to fit a line to the power law. The slope was calculated from the loglog plots from the methods shown in the lectures, Newman 2005. The resulting line was then plotted in the same loglog plot. This was done for p values of 0.2, 0.1 and 0 for each value of p m was varied as 1, 10 and 100. For the p values of 0.2 and 0.1 the 13000 steps was taking while for p = 0 15000 steps was taken. Once again the inital number of sites was set to 100.



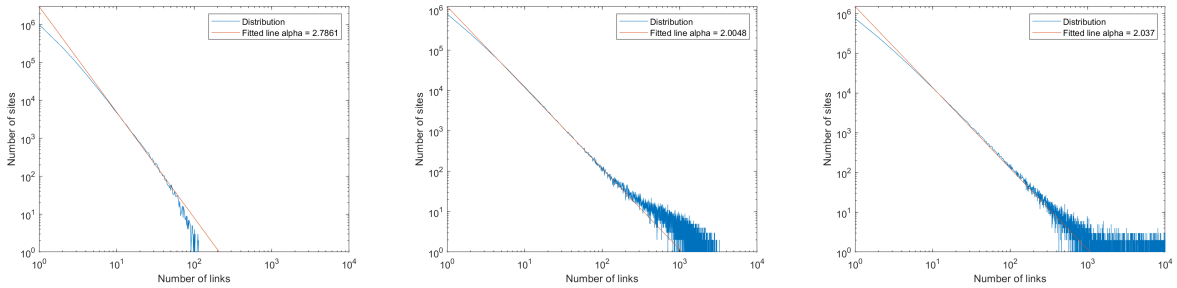
(a) p = 0.2 m = 1 (b) p = 0.2, m = 10 (c) p = 0.2 m = 100

Figure 7: Fitted lines in loglog plot for varying values of m with p = 0.2



(a) p = 0.1 m = 1 (b) p = 0.1, m = 10 (c) p = 0.1 m = 100

Figure 8: Fitted lines in loglog plot for varying values of m with p = 0.1



(a) p = 0.0 m = 1 (b) p = 0.0, m = 10 (c) p = 0.0 m = 100

Figure 9: Fitted lines in loglog plot for varying values of m with p = 0.75

As we can see from figure 7-9 as either p decreases or m increases the slope of the loglog plot approaches 2. This would then imply that as the probability that we chose a site with the preferential method increases and as each new site creates new links the slope approaches 2 and therefore approaches an infinite mean. Comparing this result with the given equation $\alpha = 1 + \frac{m+1}{m(1-p)}$ the simulations for p = 0.0, m = 100 and p = 0.0, m = 10 seems to corresponds fairly well with the equation. That the others doesn't agree as much could be from either the simulation not being run for long enough, the given equation is meant to hold for $t \to \infty$ or more likely measurement error. However the same trend can be observed theoretically and practically, namely the slope approaching 2 as m increases and p decreases, the network becomes more preferentially based.

1.3 Who Links to Whom, Largest Connection After Attacks

In order to store which sites are connected to each others the connections were saved in a sparse matrix. The network had one way connections resulting in a lower triangular matrix, when expanded from its sparse form. The connections was calculated as before however the result was stored in the sparse matrix instead of an array. Different networks were created with p as 0.1, 0.5 and 0.75 and m as 1, 10 and 100 in order to investigate how these values affected the size of the largest component. Two different attacks were simulated on in which 100 random sites and their links where removed and one in which the 100 sites with most links to it were removed. The size of largest strongly connected component for the original unattacked network and the two attacked networks are shown below in table 1-3.

Table 1: Size of largest strongly connected component before attacks for different combinations of p and m

	P =0.0	P = 0.05	P = 0.1	P = 0.5	P = 0.75
m = 1	1281	699	767	1171	872
m = 10	12100	12100	12100	12100	12100
m = 100	12100	12100	12100	12100	12100

Table 2: Size of largest strongly connected component after random attacks removing 100 sites for different combinations of p and m

	P = 0.0	P = 0.05	P = 0.1	P = 0.5	P = 0.75
m = 1	1270	693	757	1140	854
m = 10	12000	12000	11999	12001	12000
m = 100	12000	12000	12000	12000	12001

Table 3: Size of largest strongly connected component after directed attacks removing 100 most popular sites for different combinations of p and m

	P = 0.0	P = 0.05	P = 0.1	P = 0.5	P = 0.75
m = 1	143	276	194	299	200
m = 10	11942	11997	12000	12000	12000
m = 100	11991	12000	12000	12000	12000

Table 4: Size of largest strongly connected component before attacks for different combinations of p and m

The tables above 1-3 shows the robustness of the network increases as the number of new links increases. One can also see that the largest connected component when each new site only produces one link happens at p = 0.5 i.e. the chance is equal for the linking to be random or preferential.

2 Appendix

2.1 Network

2.1.1 main.m

```
%%
clear all
close all
n = 100;
t = 10000;
m = 1;
p = 1;
net = sites(n,t,m,p);

figure
hist(net,100)
xlabel('Number of links')
ylabel('Number of sites')

bins = histcounts(net, 1:t);
figure()
loglog(1:t-1,bins,'.')
ylabel('Number of sites')
xlabel('Number of links')

%alpha = 1+length(mNet)*(1/sum(log(mNet./min(mNet))));
%%
clear all
close all
n = 100;
t = 15000;
m = 100;
p = 0.0;
reps = 100;
for i = 1:reps
    net(i,:) = sites(n,t,m,p);
end
%mNet = mean(net);
%net = mNet;
figure
hist(net(:),100)
xlabel('Number of links')
ylabel('Number of sites')

bins = histcounts(net(:), 1:t);
figure()
loglog(1:t-1,bins,'.')
ylabel('Number of sites')
xlabel('Number of links')

%alpha = 1+length(mNet)*(1/sum(log(mNet./min(mNet))));

%%
clear all
close all
n = 100;
t = 12000;
m = [1 10 100];
p = [0.1 0.5 0.75];
```

```
nToRemove = 100
for i = 1:length(m)
    for j = 1: length(p)
        net = networkGraph(n,t,m(i),p(j));
        [o(i,j) r(i,j) d(i,j)] = clusters(net,nToRemove);
    end
end
```

2.1.2 sites.m

```
function [sites] = sites(n,t,m,p)
sites = ones(1,n);
for i = 1:t
    newSite = 0;
    tempSites = sites;
    for j = 1:m
        if rand() < p
            index = randi(length(tempSites),1);
            tempSites(index) = tempSites(index)+1;
        else
            tot = sum(sites);
            for k = 1:length(tempSites);
                prob(k) = sites(k)./tot;
            end
            index = randsample(1:length(tempSites),1,1,prob);
            tempSites(index) = tempSites(index) + 1;
        end
    end
    sites = tempSites;
    sites = [sites 1];
end
end
```

2.1.3 networkGraph.m

```
function [network] = networkGraph(n,t,m,p)
network = speye(n);
for i = 1:t
    tempNet = network;
    tot = sum(sum(network));
    for k = 1:length(network)
        prob(k) = sum(network(:,k))./tot;
    end
    prob = full(prob);
    tempNet((length(network)+1),(length(network)+1)) = 1;
    for j = 1:m
        if rand() < p
            index = randi(length(network),1);
            tempNet((length(network)+1),index) = tempNet((length(network)+1),index)+1;

        else
            index = randsample(1:length(network),1,1,prob);
            tempNet((length(network)+1),index) = tempNet((length(network)+1),index)+1;

        end
    end
    network = tempNet;
```

```
end
end
```

2.1.4 clusters.m

```
function [origSize, randSize, dirSize] = clusters(net,nToRemove)
%Calculate largest cluster size after directed and random attack
net1 = net;
net2 = net;
for i = 1:nToRemove
    x = randi(length(net1));
    net1(x,:) = 0;
    net1(:,x) = 0;
end
for i = 1:nToRemove
    sums = sum(net2,1);
    maximum = max(sums);
    [x,y] = find(sums == maximum);
    net2(y(1),:) = 0;
    net2(:,y(1)) = 0;
end
[numOrigComp, origComp] = graphconncomp(net,'Directed',false);
[randAttNComp, randAttComp] = graphconncomp(net1,'Directed',false);
[dirAttNComp, dirAttCom] = graphconncomp(net2,'Directed',false);
origBins = histcounts(origComp,1:length(net));
randBins = histcounts(randAttComp,1:length(net));
dirBins = histcounts(dirAttCom,1:length(net));
origSize = max(origBins);
randSize = max(randBins);
dirSize = max(dirBins);
end
```