**自动化1901 林宇航 201906060308**

代码：

clear all;

tic

N = 1000;

K = 10;

P = [0,0.0001,0.00025,0.0005,0.001,0.002,0.004,0.008,0.018,0.03,0.065,0.13,0.25,0.5,1];

num\_of\_P = length(P);

%P=[0:0.0001:0.001,0.001:0.001:0.01,0.01:0.01:0.1,0.1:0.1:1];

C = zeros(1,num\_of\_P);

L = zeros(1,num\_of\_P);

%开始初始化

A = sparse(N,N);

%连接边

for i= 1:K/2

A = A + diag(ones(1,N-i),i); %构造一个第i条对角线全为1的N\*N矩阵，加到原矩阵

end

for i = 1:K/2

A(N-K/2+i:N,i)=1; % 环形的首尾相接处的边

end

A = A + A';

%初始化完成

A\_BACK = A;

%以不同的概率P对原来的边进行重新连接

for i = 1:num\_of\_P

A = A\_BACK; %防止A被修改

p = P(i);

%P(i) = p;

i

for ii = 1:50

A = A\_BACK;

%重新连接

for j = 1:K/2 %只需要顺时针就行 每个节点的顺时针就能遍历所有的边

NeedRW = find(rand(1,N) <= p); %通过概率得到这些点是需要重新连接的

for x = 1:length(NeedRW)

%连接新节点

while 1

NewVertice = fix(rand()\*N+1); %找到要连接的新节点

if NewVertice~=NeedRW(x) && A(NeedRW(x),NewVertice) == 0 %不能自连，重复连接

A(NeedRW(x),NewVertice)=1;

A(NewVertice,NeedRW(x))=1;

break;

end

end

%删除老节点

if NeedRW(x) <= N-j %原来连接的老节点 比节点大 ，这个节点在（1～N-i）

A(NeedRW(x),NeedRW(x)+j)=0;

A(NeedRW(x)+j,NeedRW(x))=0;

else %原来连接的老节点 比节点小 ，这个节点在（N-i～N）

A(NeedRW(x)-N+j,NeedRW(x)) = 0;

A(NeedRW(x),NeedRW(x)-N+j) = 0;

end

end

end%当前概率 重新连接结束

%计算当前概率的 聚类系数和平均路径

%聚类系数：

Ci = zeros(1,N);

for y = 1:N

%先算每个节点的聚类系数： Ci

Ne = find(A(y,:)==1); %与y节点相连接的邻居节点集合

NeNum = length(Ne); %邻居个数

if NeNum <= 1

Ci(y)=0;

else

Ei = length(find(A(Ne,Ne)==1)); %邻居与邻居之间边的数量（三角形）

Ci(y)=Ei\*2/(NeNum\*(NeNum-1)); %（三角形/三元组）

end

end%当前p 聚类系数求解完毕

Cc(ii) = sum(Ci)/N;

%平均最短路径

Paths = tril(graphallshortestpaths(tril(sparse(A)),'directed',false));

Ll(ii) = sum(sum(Paths)) / (N\*(N-1)/2);

clear NeedRW Ci Ei NeNum Paths A;

end

C(i) = sum(Cc)/length(Cc);

L(i) = sum(Ll) /length(Ll);

clear Cc Ll p;

end

toc

%画图

figure

semilogx(P(2:end),C(2:end)/C(1),'d')

hold on

semilogx(P(2:end),L(2:end)/L(1),'s','MarkerFaceColor','k')

hold off

xlabel('P')

legend('C(p)/C(0)','L(p)/L(0)')

结果：

