**实验1： 连通试验**

**一、实验目的**

通过本实验的学习，理解和掌握信息熵的概念和物理意义；试着求解图像的信息熵。

**二、实验内容**

理解无线传感器网络的拓扑生成方法；

学习网络的连通度计算方法；

学习对网络连通度进行调整。

**三、实验原理、方法和手段**

**3.1 网络结构**

是一个简单图（网络可看为一个‘图’），其中为点集，边集。若为定点的度，*A*(*G*)为*G*的领接矩阵。

**3.2 连通度**

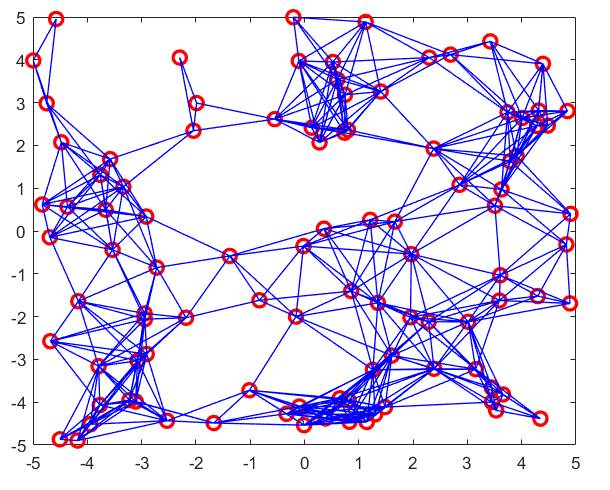
代数连通度是Laplace图谱的次小特征值，是研究图谱问题的重要指标。在前述的图中，*A*(*G*)为*G*的领接矩阵，，则*G*的Laplace矩阵为。也可以被定义为满足以下不等式的二次型：

其中的特征值为。因此的列和为0，所以是对应于0的一个特征向量。当且仅当图*G*是连通的，有。则称为图*G*的代数连通度，记为。

**四、实验内容**

**4.1 生成网络**

设随机生成一个具有100个节点的传感器网络*G*。节点随机放置在10x10方形区域内，通信距离为2，下图所示。请你也生成一个同样参数的网络。



**4.2 网络连通度**

接上题（**4.1**），计算网络*G*的代数连通度，记为。

**4.3 更多网络**

接上题（**4.1和4.2**），修改网络参数（例如节点改为50个，或者半径改为5），重新生成网络并计算代数连通度。

**五、实验组织运行要求**

单人实验，单独完成。

**六、实验条件**

（1）计算机（win、linux、osx不限）

（2）MATLAB或其他任何编程工具/语言，工具不是我们实验目的。

**八、实验结果**

**8.1 网络结构**

一个具有100个节点的传感器网络G。节点随机放置在10x10方形区域内，通信距离为2。

**实验代码**

|  |
| --- |
| DrawNetworks.m  function fig = DrawNetworks( Network )  %画?向图 ?络连接图 Network Connection.  % function fig = DrawNetworks( Network, Dists ) %把所有的边的长度（通信距离）都标出来了  num = Network.Conf.NodeNumber; %节点个数  loc = Network.Nodes.loc; %节点坐标  square = Network.Conf.Square; %?形区域的边长  Neighbors = Network.Nodes.neighbors; %邻接节点的id  fig = figure;  plot(loc(:,1),loc(:,2),'ro','MarkerSize',8,'LineWidth',2); %节点是红?圆圈  side=ceil(square/2);  axis([-side,side,-side,side]);  for i=1:num  for k = 1:length(Neighbors{i})  j = Neighbors{i}(k);  % c = num2str(Dists(i,j),'%.2f');  % text((loc(i,1) + loc(j,1))/2,(loc(i,2) + loc(j,2))/2,c,'Fontsize',10); %把所有的边的长度（通信距离）都标出来了  % hold on;  line([loc(i,1),loc(j,1)],[loc(i,2),loc(j,2)],'LineWidth',0.8,'Color','b'); %线是蓝?  end  end  set(gcf, 'Color', 'w'); %??  end  CreateNetworksFunc.m  function [ Network, Dists ] = CreateNetworksFunc(Conf)  % 创建?向图 ?络连接图 Network Connection.  num = Conf.NodeNumber; %节点个数  square = Conf.Square; %?形区域的边长  maxDist = Conf.CommDist; %最?通信距离  loc = square\*rand(num,2) - square/2; %num\*2的随机数 节点坐标  Dists = Euclid\_Dist(loc(:,1),loc(:,2)); %节点数\*节点数，对?线元素为0  % without self-loop 不存在节点??到??的路径，对?线上的元素为?穷?  Dists = Dists + 10\*maxDist\*eye(num);  Neighbors = cell(num,1);  maxDegree = 0; %节点的最?度，与节点相邻的最?边数  edges = 0; %图的总边的个数，?向图的度/2  for i=1:num  Neighbors{i} = find(Dists(i,:)<=maxDist); %找邻接节点的id  if length(Neighbors{i}) > maxDegree  maxDegree = length(Neighbors{i}); %节点的最?度  end  edges = edges + length(Neighbors{i});  end  Nodes.loc = loc;  Nodes.neighbors = Neighbors;  Network.maxDegree = maxDegree;  Network.edges = edges/2; %% undirected graph  Network.Conf = Conf;  Network.Nodes = Nodes;  end  function dist = Euclid\_Dist(X,Y)  % 求两两节点之间的距离，输出[节点\*节点]的矩阵，距离矩阵  len = length(X);  xx = repmat(X,1,len); %节点数\*节点数  yy = repmat(Y,1,len);  dist = sqrt((xx-xx').^2+(yy-yy').^2); %节点数\*节点数  end  demo\_Create\_Network\_Connection.m  %创建?向图 ?络连接图 Network Connection.  clc;  close all;  clear;  Conf.Square = 10; %?形区域的边长  Conf.NodeNumber = 100; %节点个数  Conf.CommDist = 2; %最?通信距离  is\_create\_network = 1;  if is\_create\_network == 1  [ Network, Dists ] = CreateNetworksFunc(Conf);  save Network\_1.mat Network  else  load Network\_1.mat  end  nodenum = size(Network.Nodes.loc,1); %节点个数  lap\_matrix = zeros(nodenum); %节点数\*节点数 图的Laplace矩阵：diag(d1,d2,...dn)-邻接矩阵，di为节点i的度  for i=1:nodenum  idx = Network.Nodes.neighbors{i}; %邻接节点的id  lap\_matrix(i,idx) = -1; %负的邻接矩阵  lap\_matrix(i,i) = length(idx); %对?线元素为节点的度  end  eig\_val = eig(lap\_matrix); %lap\_matrix的特征值  eig\_val = sort(eig\_val,'ascend'); %从?到?排序，最?特征值为0  algeb\_conn = eig\_val(2) % algebraic connectivity 代数连通度：lap\_matrix的第??特征值>0，连通图  avg\_deg = sum(diag(lap\_matrix))/nodenum % average values 节点度的均值  DrawNetworks(Network);  % DrawNetworks(Network, Dists); %把所有的边的长度（通信距离）都标出来了  print(gcf,'-dpng','Network\_1.png'); %保存图? |

**实验结果（图片）**

|  |
| --- |
| Network_1 |

**8.2 连通度**

**实验代码**

|  |
| --- |
| %创建?向图 ?络连接图 Network Connection.  clc;  close all;  clear;  Conf.Square = 10; %?形区域的边长  Conf.NodeNumber = 100; %节点个数  Conf.CommDist = 2; %最?通信距离  is\_create\_network = 1;  if is\_create\_network == 1  [ Network, Dists ] = CreateNetworksFunc(Conf);  save Network\_1.mat Network  else  load Network\_1.mat  end  nodenum = size(Network.Nodes.loc,1); %节点个数  lap\_matrix = zeros(nodenum); %节点数\*节点数 图的Laplace矩阵：diag(d1,d2,...dn)-邻接矩阵，di为节点i的度  for i=1:nodenum  idx = Network.Nodes.neighbors{i}; %邻接节点的id  lap\_matrix(i,idx) = -1; %负的邻接矩阵  lap\_matrix(i,i) = length(idx); %对?线元素为节点的度  end  eig\_val = eig(lap\_matrix); %lap\_matrix的特征值  eig\_val = sort(eig\_val,'ascend'); %从?到?排序，最?特征值为0  algeb\_conn = eig\_val(2) % algebraic connectivity 代数连通度：lap\_matrix的第??特征值>0，连通图  avg\_deg = sum(diag(lap\_matrix))/nodenum % average values 节点度的均值  DrawNetworks(Network);  % DrawNetworks(Network, Dists); %把所有的边的长度（通信距离）都标出来了  print(gcf,'-dpng','Network\_1.png'); %保存图? |

**实验结果**

|  |
| --- |
|  |

**8.3 修改参数**

**（1）网络1：**一个具有200个节点的传感器网络*G*。节点随机放置在10x10方形区域内，通信距离为2。

**实验代码**

|  |
| --- |
| %创建?向图 ?络连接图 Network Connection.  clc;  close all;  clear;  Conf.Square = 10; %?形区域的边长  Conf.NodeNumber = 200; %节点个数  Conf.CommDist = 2; %最?通信距离  is\_create\_network = 1;  if is\_create\_network == 1  [ Network, Dists ] = CreateNetworksFunc(Conf);  save Network\_1.mat Network  else  load Network\_1.mat  end  nodenum = size(Network.Nodes.loc,1); %节点个数  lap\_matrix = zeros(nodenum); %节点数\*节点数 图的Laplace矩阵：diag(d1,d2,...dn)-邻接矩阵，di为节点i的度  for i=1:nodenum  idx = Network.Nodes.neighbors{i}; %邻接节点的id  lap\_matrix(i,idx) = -1; %负的邻接矩阵  lap\_matrix(i,i) = length(idx); %对?线元素为节点的度  end  eig\_val = eig(lap\_matrix); %lap\_matrix的特征值  eig\_val = sort(eig\_val,'ascend'); %从?到?排序，最?特征值为0  algeb\_conn = eig\_val(2) % algebraic connectivity 代数连通度：lap\_matrix的第??特征值>0，连通图  avg\_deg = sum(diag(lap\_matrix))/nodenum % average values 节点度的均值  DrawNetworks(Network);  % DrawNetworks(Network, Dists); %把所有的边的长度（通信距离）都标出来了  print(gcf,'-dpng','节点200.png'); %保存图? |

**实验结果（包括网络结构和连通度）**

|  |
| --- |
| 节点200 |

**（2）网络2：**一个具有100个节点的传感器网络*G*。节点随机放置在15x15方形区域内，通信距离为2。

**实验代码**

|  |
| --- |
| %创建?向图 ?络连接图 Network Connection.  clc;  close all;  clear;  Conf.Square = 15; %?形区域的边长  Conf.NodeNumber = 100; %节点个数  Conf.CommDist = 2; %最?通信距离  is\_create\_network = 1;  if is\_create\_network == 1  [ Network, Dists ] = CreateNetworksFunc(Conf);  save Network\_1.mat Network  else  load Network\_1.mat  end  nodenum = size(Network.Nodes.loc,1); %节点个数  lap\_matrix = zeros(nodenum); %节点数\*节点数 图的Laplace矩阵：diag(d1,d2,...dn)-邻接矩阵，di为节点i的度  for i=1:nodenum  idx = Network.Nodes.neighbors{i}; %邻接节点的id  lap\_matrix(i,idx) = -1; %负的邻接矩阵  lap\_matrix(i,i) = length(idx); %对?线元素为节点的度  end  eig\_val = eig(lap\_matrix); %lap\_matrix的特征值  eig\_val = sort(eig\_val,'ascend'); %从?到?排序，最?特征值为0  algeb\_conn = eig\_val(2) % algebraic connectivity 代数连通度：lap\_matrix的第??特征值>0，连通图  avg\_deg = sum(diag(lap\_matrix))/nodenum % average values 节点度的均值  DrawNetworks(Network);  % DrawNetworks(Network, Dists); %把所有的边的长度（通信距离）都标出来了  print(gcf,'-dpng','矩形区域长15.png'); %保存图? |

**实验结果（包括网络结构和连通度）**

|  |
| --- |
| 矩形区域长15 |

**（3）网络3：**一个具有100个节点的传感器网络*G*。节点随机放置在10x10方形区域内，通信距离为3。

**实验代码**

|  |
| --- |
| %创建?向图 ?络连接图 Network Connection.  clc;  close all;  clear;  Conf.Square = 10; %?形区域的边长  Conf.NodeNumber = 100; %节点个数  Conf.CommDist = 3; %最?通信距离  is\_create\_network = 1;  if is\_create\_network == 1  [ Network, Dists ] = CreateNetworksFunc(Conf);  save Network\_1.mat Network  else  load Network\_1.mat  end  nodenum = size(Network.Nodes.loc,1); %节点个数  lap\_matrix = zeros(nodenum); %节点数\*节点数 图的Laplace矩阵：diag(d1,d2,...dn)-邻接矩阵，di为节点i的度  for i=1:nodenum  idx = Network.Nodes.neighbors{i}; %邻接节点的id  lap\_matrix(i,idx) = -1; %负的邻接矩阵  lap\_matrix(i,i) = length(idx); %对?线元素为节点的度  end  eig\_val = eig(lap\_matrix); %lap\_matrix的特征值  eig\_val = sort(eig\_val,'ascend'); %从?到?排序，最?特征值为0  algeb\_conn = eig\_val(2) % algebraic connectivity 代数连通度：lap\_matrix的第??特征值>0，连通图  avg\_deg = sum(diag(lap\_matrix))/nodenum % average values 节点度的均值  DrawNetworks(Network);  % DrawNetworks(Network, Dists); %把所有的边的长度（通信距离）都标出来了  print(gcf,'-dpng','通信距离3.png'); %保存图? |

**实验结果（包括网络结构和连通度）**

|  |
| --- |
| 通信距离3 |

**（4）网络4：**一个具有200个节点的传感器网络*G*。节点随机放置在15x15方形区域内，通信距离为3。

**实验代码**

|  |
| --- |
| %创建?向图 ?络连接图 Network Connection.  clc;  close all;  clear;  Conf.Square = 15; %?形区域的边长  Conf.NodeNumber = 200; %节点个数  Conf.CommDist = 3; %最?通信距离  is\_create\_network = 1;  if is\_create\_network == 1  [ Network, Dists ] = CreateNetworksFunc(Conf);  save Network\_1.mat Network  else  load Network\_1.mat  end  nodenum = size(Network.Nodes.loc,1); %节点个数  lap\_matrix = zeros(nodenum); %节点数\*节点数 图的Laplace矩阵：diag(d1,d2,...dn)-邻接矩阵，di为节点i的度  for i=1:nodenum  idx = Network.Nodes.neighbors{i}; %邻接节点的id  lap\_matrix(i,idx) = -1; %负的邻接矩阵  lap\_matrix(i,i) = length(idx); %对?线元素为节点的度  end  eig\_val = eig(lap\_matrix); %lap\_matrix的特征值  eig\_val = sort(eig\_val,'ascend'); %从?到?排序，最?特征值为0  algeb\_conn = eig\_val(2) % algebraic connectivity 代数连通度：lap\_matrix的第??特征值>0，连通图  avg\_deg = sum(diag(lap\_matrix))/nodenum % average values 节点度的均值  DrawNetworks(Network);  % DrawNetworks(Network, Dists); %把所有的边的长度（通信距离）都标出来了  print(gcf,'-dpng','网络四.png'); %保存图? |

**实验结果（包括网络结构和连通度）**

|  |
| --- |
| 网络四 |