最短路问题

山东理工大学数学院 周世祥

Contents

生物信息学工具箱

P=biograph(R,nodes,'ShowWeights','on') % 建立有向图对象P view(P)

假设一个图有n个节点，则可以用一个nxn矩阵R表示，R(i,j)=k表示第i个节点和第j个节点 边权值为k，若两点之间无边，可设R(i,j)=0或R(i,j)=无穷大，大部分算法都设为无穷大。 由三个向量可以构成图的关联矩阵。 [redacted],

,起始与终止节点

,边权值向量

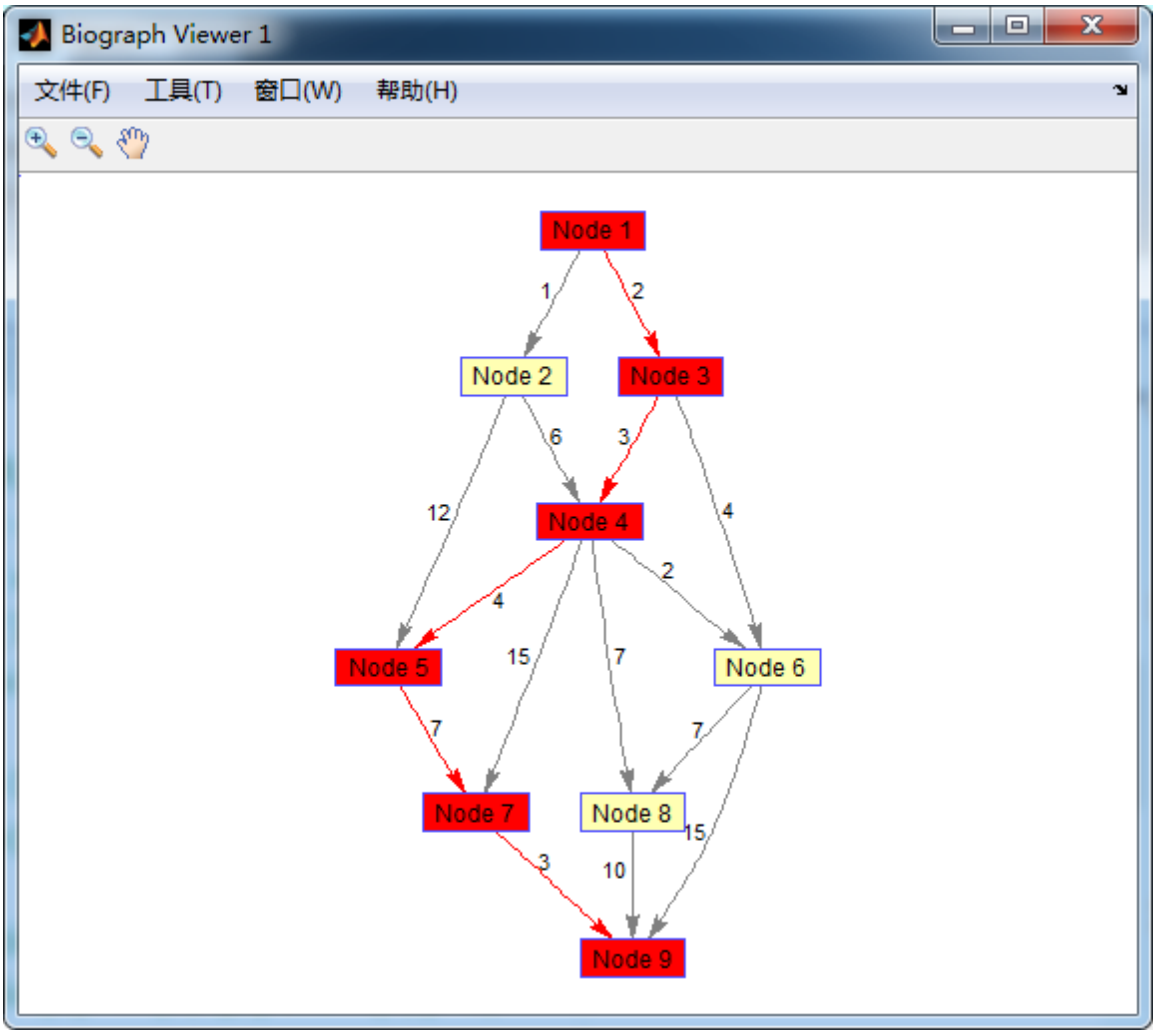
；关联矩阵的稀疏表示

令 $R(n,n)=0$ 将其变为方阵，稀疏矩阵可以由命令full( )变为常规矩阵。常规矩阵由sparse()转 为稀疏矩阵。

```
aa=[1 1 2 2 3 3 4 4 4 4 5 6 6 7 8];
bb=[2 3 5 4 4 6 5 7 8 6 7 8 9 9 9 ];
w=[1 2 12 6 3 4 4 15 7 2 7 7 15 3 10];
R=sparse(aa,bb,w); %关联矩阵稀疏表示
R(9,9)=0; %变成方阵
full(R)

h=view(biograph(R,[],'ShowWeights','on'))

[d,p]=graphshortestpath(R,1,9) %求节点1到9的最短路径
set(h.Nodes(p),'Color',[1 0 0])
edges=getedgesbynodeid(h,get(h.Nodes(p),'ID')) ;%获得最优边句柄
set(edges,'LineColor',[1 0 0])
```

```
ans =

     0     1     2     0     0     0     0     0     0
     0     0     0     6    12     0     0     0     0
     0     0     0     3     0     4     0     0     0
     0     0     0     0     4     2    15     7     0
     0     0     0     0     0     0     7     0     0
     0     0     0     0     0     0     0     7    15
     0     0     0     0     0     0     0     0     3
     0     0     0     0     0     0     0     0    10
     0     0     0     0     0     0     0     0     0

Biograph object with 9 nodes and 15 edges.

d =

    19
```

```
p =

     1     3     4     5     7     9
```



Dijkstra最短路径算法

```
%下面函数试图计算从初始节点到任意节点之间的最优路径
% function [d,p0]=dijkstra(W,s,t)
% [n,m]=size(W); ix=(W==0); W(ix)=Inf; %将权值设为无穷大
% if n~=m, error('Square W required'); end %关联矩阵设为方阵
% visited(1:n)=0; dist(1:n)=Inf; dist(s)=0; d=Inf; w=[1:n]';
% for i=1:(n-1)
%    ix=(visited==0); vec(1:n)=Inf; vec(ix)=dist(ix);
%    [a,u]=min(vec); visited(u)=1;
%    for v=1:n
%       if (W(u,v)+dist(u)<dist(v))
%          dist(v)=dist(u)+W(u,v); parent(v,i)=u;
% end; end; end
% u=parent(:,1)==s; p0(u,1)=s; p0(u,2)=w(u); w(u)=0;
% for k=1:n, vec=parent(k,:); vec=vec(vec~=0);
%    if length(vec)>0, vec=vec(end);
%        if w(vec)==0
%            v1=p0(vec,:); v1=v1(v1~=0); aa=[v1, k]; w(k)=0;
%            for j=1:length(aa), p0(k,j)=aa(j); end
% end, end, end
% p0=p0(t,:); d=dist(t);
```

假设节点个数为n，起始节点为s，则算法具体步骤：

初始化，建立三个向量存储各节点的状态，其中，visited表示各个节点是否更新，初始值为0；dist存储起始节点到本节点的最短距离，初始值为无穷大，parent向量存储到本节点的上一个节点，默认值为0
另设起始节点处dist(s)=0.
循环求解。 让i做n-1次循环，更新能由本节点经过一个边到达的节点距离与上级节点信息，并更新由本节点可以到达的未访问节点的最短路径信息。循环直到所有未访问节点完全处理完成。
提取到终止节点t的最短路径。利用parent向量逐步提取最优路径。

```
aa=[1 1 2 2 3 3 4 4 4 4 5 6 6 7 8];
bb=[2 3 5 4 4 6 5 7 8 6 7 8 9 9 9 ];
w=[1 2 12 6 3 4 4 15 7 2 7 7 15 3 10];
R=sparse(aa,bb,w); %关联矩阵稀疏表示
R(9,9)=0; %变成方阵
% openvar R    可视化方式编辑关联矩阵
W=ones(9);
[d,p]=dijkstra(R.*W,1,9) %搜索最优路径

 % 一次性地构造出起始节点到各个节点的最优路径信息，只需将目标节点设置成向量即可。
 [d,p]=dijkstra(R.*W,1,2:9) %搜索1到各个节点的最优路径
 % Tab=[1*ones(8,1),(2:9)',p,d']
```

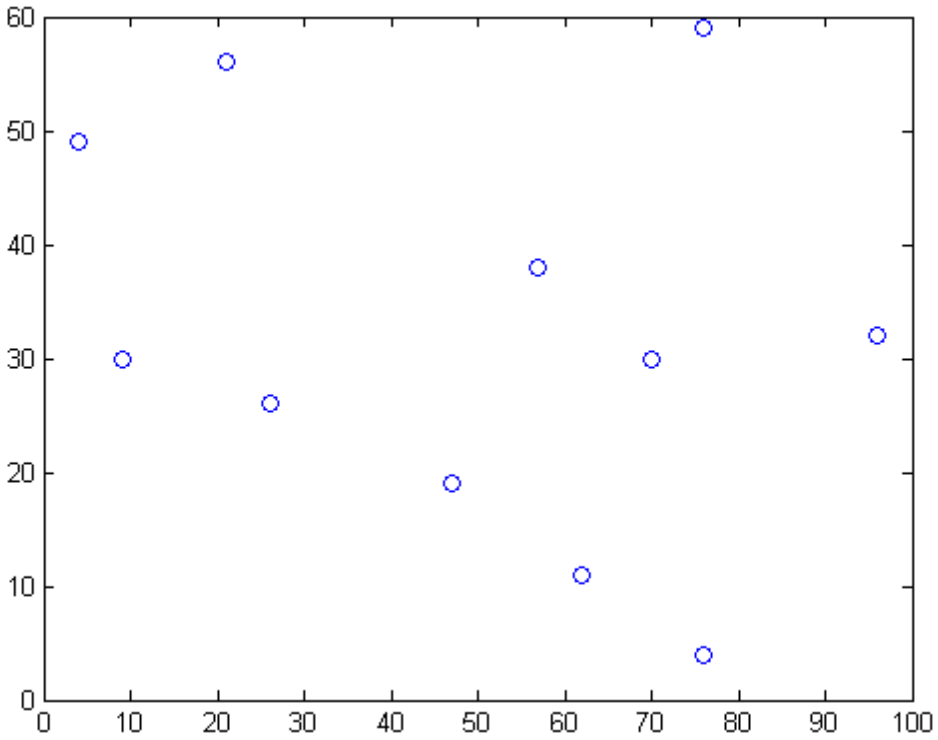
```
d =

    19


p =

     1     3     4     5     7     9


d =

     1     2     5     9     6    16    12    19


p =

     1     2     0     0     0     0
     1     3     0     0     0     0
     1     3     4     0     0     0
     1     3     4     5     0     0
     1     3     6     0     0     0
     1     3     4     5     7     0
     1     3     4     8     0     0
     1     3     4     5     7     9
```

练习

假设有11个城市，其分布坐标分别为(4,49),(9,30),(21,56),(26,26),(47,19),(57,38),(62,11),(70,30),(76,59),(76,4),(96,4) 其间的公路如图示，求出城市A(点1)到城市B(点11)的最短路径，如果城市6和8之间修理，重新搜索最优路径。

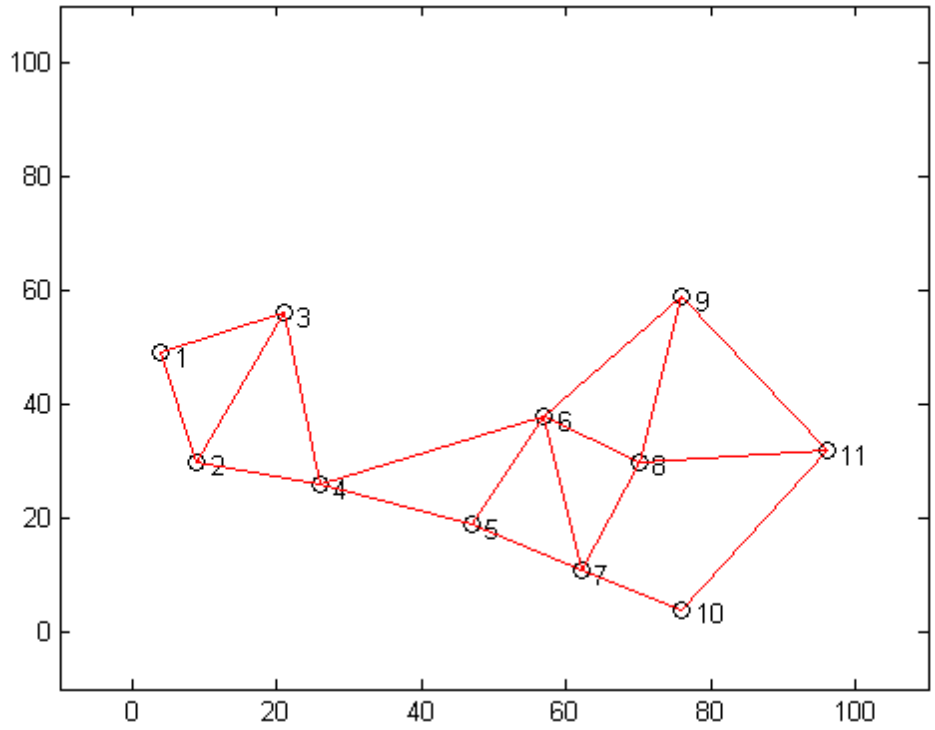
```
 clear all
 x=[4  9 21 26 47 57   62 70 76 76 96];
 y=[49 30 56 26 19 38 11 30 59 4 32];
 plot(x,y,'o')
%  line([4,49],[9,30])
%  line([4,49],[21,56])
%  line([21,56],[26,26])
```

```
%  line([9,30],[26 26])
%  line([26,26],[47,19])
%  line([47,19],[57,38])
%  line([57,38],[26,26])
%  line([57,38],[62,11])
%  line([57,38],[70,30])
%  line([57,38],[47,19])
%  line([70,30],[76,59])
%   line([70,30],[57,38])
%    line([70,30],[62,11])
%     line([70,30],[96,32])
%       line([76,59],[96,32])
%         line([76,4],[96,32])
```




```
 A1 = [4,49];
A2 = [9,30];
A3 = [21,56];
A4 = [26,26];
A5 = [47,19];

A6 = [57,38];

A7 = [62,11];
A8 = [70,30];
A9 = [76,59];

A10 = [76,4];
A11=[96,32];

for p = 1:11  % 放进一个矩阵内方便操作
    c = num2str(p);
    s = ['A(p,:) = A' c ';'];
    eval(s);
end
% scatter(A(:,1),A(:,2));
for p = 1:11
```

```
    c = num2str(p);
    plot(A(p,1),A(p,2),'ko');
    hold on
    axis([-10 110 -10 110]);
    text(A(p,1)+2,A(p,2)-1,c);
end
% 需要的顺序为A1-A2-A3-A4-A5-A6-A7-A8-A9-A11-A10
sx = [1 2 3 4 5 6 7 8 9   11 10];
for p = 1:length(sx)-1
    P1 = A(sx(p),:);
    P2 = A(sx(p+1),:);
    line([P1(1) P2(1)],[P1(2) P2(2)],'color','r');
end
sy = [1  3 2 4   6  5 7   10 11];
for p = 1:length(sy)-1
    P1 = A(sy(p),:);
    P2 = A(sy(p+1),:);
    line([P1(1) P2(1)],[P1(2) P2(2)],'color','r');
end
sz = [6 8 11 9 6];
for p = 1:length(sz)-1
    P1 = A(sz(p),:);
    P2 = A(sz(p+1),:);
    line([P1(1) P2(1)],[P1(2) P2(2)],'color','r');
end

hold off

% 最短距离为122.9394,路径为1->2->4->5->7->8->11
```



遗传算法工具箱安装

```
str=[matlabroot,'\mcr\toolbox\gatbx'];
addpath(str)
```

参考文献

薛定宇，《MATLAB最优化计算》，清华大学出版社，2020年1月。

Published with MATLAB® R2014a