Article

Maximum matching of the network: A Matlab program and application

WenJun Zhang
School of Life Sciences, Sun Yat-sen University, Guangzhou 510275, China; International Academy of Ecology and Environmental Sciences, Hong Kong
E-mail: zhwj@mail.sysu.edu.cn, wjzhang@iaees.org

Received 7 November 2016; *Accepted 12 December 2016*; *Published online 1 December 2017*

Abstract
In this article, I present full Matlab codes of Hungarian Algorithm for maximum matching in the network.

Keywords network; maximum matching; Hungarian Algorithm; Matlab.

Selforganizology
ISSN 2410-0080
URL: http://www.iaees.org/publications/journals/selforganizology/online-version.asp
RSS: http://www.iaees.org/publications/journals/selforganizology/rss.xml
E-mail: selforganizology@iaees.org
Editor-in-Chief: WenJun Zhang
Publisher: International Academy of Ecology and Environmental Sciences

1 Introduction

Matching problem refers to the optimal assignment problem. Let *M* be a subset of the edge set *E* of a graph (network) *X*. If any of two links of *M* are disjoint in *X*, *M* is a matching of *X*. The two nodes of a link of *M* are matched in *M*. If a link of the matching *M* is associated with the node *v*, *M* saturates the node *v*, and otherwise *v* is *M*-unsaturated. *M* is called a maximum matching of *X*, if there is not other matching *M*' in *X*, such that |*M*'|>|*M*|. The matching *M* of graph *X* is a maximum matching if and only if there is not any *M*-augmenting path in *X* (Chan et al., 1982; Zhang, 2012). In this article, I present full Matlab codes of Hungarian Algorithm for maximum matching in the network.

2 Algorithm

Hungarian Algorithm is used for solving the maximum matching problem. Here it is used for a bipartite graph problem (Zhang, 2012). Suppose we want to find an optimal plan to deploy *m* objects on *n* areas. Assume any of *m* objects can be deployed on one or more areas. Not all objects will certainly be deployed on any area. We wonder that whether we can assign each object to an area. We denote *m* objects and *n* areas as $G=\{g_1, g_2, \dots, g_m\}$ and $H=\{h_1, h_2, \dots, h_n\}$, respectively, where g_i and h_j are adjacent if and only if the object g_i can be deployed on the area h_j. We try to to find the maximum matching of a network (graph) (Zhang, 2012).

Suppose $X=(G,H,E)$ is a bipartite graph (network), where $G=\{g_1, g_2, \dots, g_m\}$, $H=\{h_1, h_2, \dots, h_n\}$. First, arbitrarily find an initial matching *M* of *X*, and (Chan et al., 1982; Zhang, 2012)

(1) Let $S=\phi$, $T=\phi$, return (2).

(2) If *M* saturates all nodes of *G*-*S*, then *M* is the maximum matching of *X*. Otherwise, arbitrarily find a

M-unsaturated node $u \in G\text{-}S$, set $S=S\cup\{u\}$, return (3).

(3) Let $N(S)=\{v|u\in S,\ uv\in E\}$. Return (2), if $N(S)=T$,. Otherwise, find $h\in N(S)\text{-}T$. If *h* is *M*-saturated, return (4), and otherwise, return (5).

(4) Suppose $gh\in M$, then let $S=S\cup\{g\}$, $T=T\cup\{h\}$, return (3).

(5) Known *u*-*h* path is a *M*-augmenting path, denoted by *C*, and let $M=M\oplus C$, return (1), where $M\oplus C=M\cup C\text{-}M\cap C$.

The Matlab codes, maxMatch.m, of Hungarian Algorithm for maximum matching problems are as follows. Data is $a= (a_{ij})_{v*n}$,where *v* is the number of elements in *G*, and *n* is the number of elements in *H*.

```
a=input('Input the excel file name of data matrix for maximum matching problem (e.g., a.xls, etc. Data is a=(aij)v*n,where v is
the number of elements in G, and n is the number of elements in H): ','s');
a=xlsread(a);
v=size(a,1); n=size(a,2);
c=zeros(n,5); m=zeros(v,n);
g=zeros(1,v); h=zeros(1,n); hh=zeros(1,n);
cp=0;
for i=1:v
for j=1:n
if (a(i,j)~=0)
m(i,j)=1;
break;
end; end
if (m(i,j)~=0) break; end
end
while (v>0)
g=zeros(1,v); h=zeros(1,n);
for i=1:v
cd=1;
for j=1:n
if (m(i,j)~=0) cd=0; break; end
end
if (cd~=0) g(i)=-n-1; end
end
cd=0;
while (v>0)
gi=0;
for i=1:v
if (g(i)<0)
gi=i;
break;
end; end
if (gi==0)
cd=1;
break;
end
```

```
g(gi)=-g(gi);
k=1;
for j=1:n
if ((a(gi,j)~=0) & (h(j)==0))
h(j)=gi;
hh(k)=j;
k=k+1;
end; end
if (k>1)
k=k-1;
for j=1:k
cp=1;
for i=1:v
if (m(i,hh(j))~=0)
g(i)=-hh(j);
cp=0;
break;
end; end
if (cp~=0) break; end
end
if (cp~=0)
k=1;
j=hh(j);
while (v>0)
c(k,2)=j;
c(k,1)=h(j);
j=abs(g(h(j)));
if (j==(n+1)) break; end
k=k+1;
end
for i=1:k
if (m(c(i,1),c(i,2))~=0) m(c(i,1),c(i,2))=0;
else m(c(i,1),c(i,2))=1; end
end
break;
end; end
end
if (cd~=0) break; end
end
fprintf(['Maximum matching:' '\n']);
m
```

3 Application Example

Suppose there are 6 objects in *G* and 5 areas in *H*. The data matrix for maximum matching is

$$\begin{array}{ccccc}
1 & 0 & 1 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 \\
0 & 1 & 1 & 0 & 1 \\
1 & 0 & 1 & 0 & 0 \\
0 & 1 & 0 & 1 & 1 \\
1 & 0 & 1 & 0 & 1
\end{array}$$

Using the algorithm, we obtained the maximum matching matrix as the following

$$\begin{array}{ccccc}
1 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 1 \\
0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0
\end{array}$$

And the maximum matching is: (object 1, area 1), (object 2, area 2), (object 3, area 3, area 5), (object 4, area 3), (object 5, area 4).

References

Chan SB, et al. 1982. Graph Theory and Its Applications. Science Press, Beijing, China

Zhang WJ. 2012. Computational Ecology: Graphs, Networks and Agent-based Modeling. World Scientific, Singapore