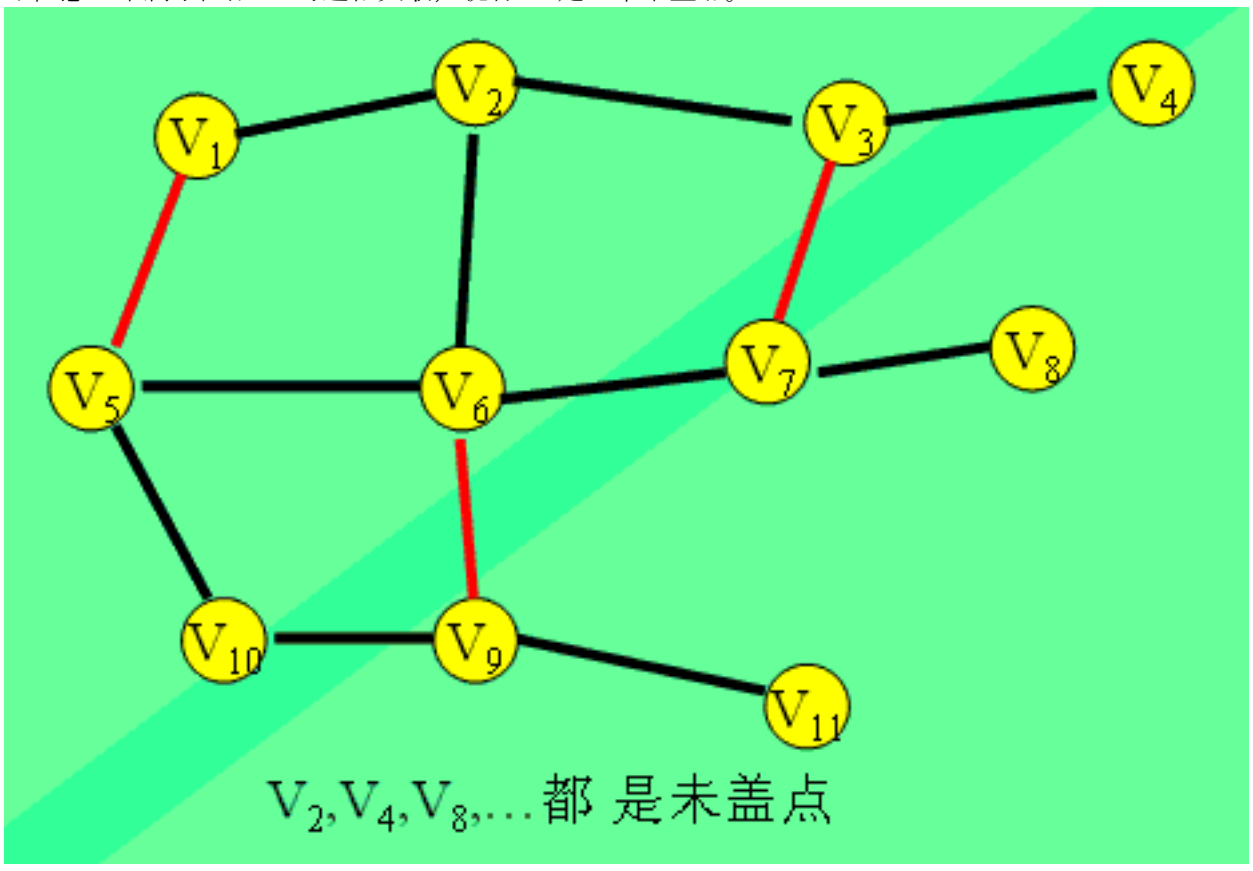
匈牙利算法

分类： 1.3 数据结构和算法 2009-08-07 10:13 1221 人阅读 评论(0) 收藏 举报

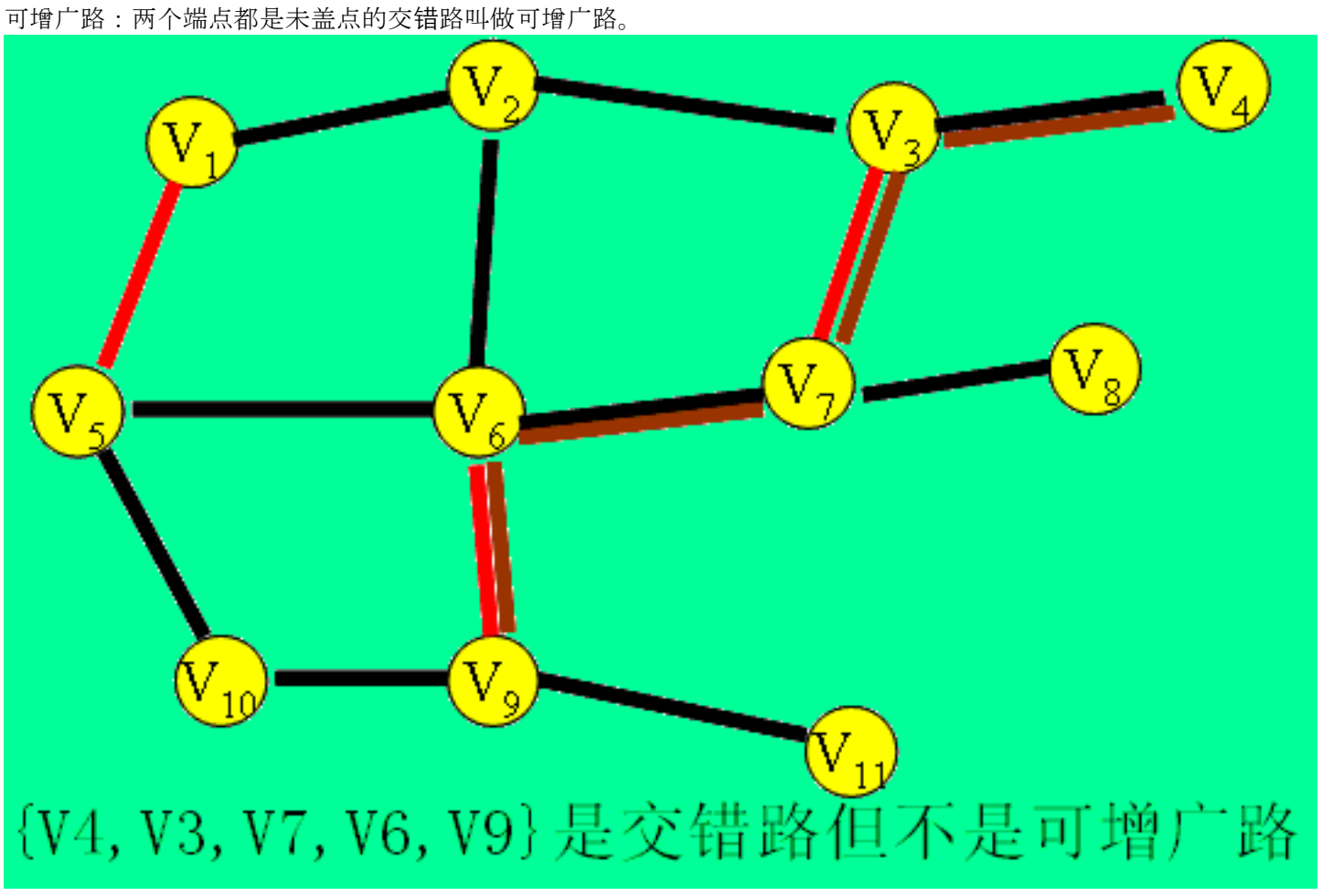
本文转自大牛博客：http://www.byvoid.com/blog/hungary/

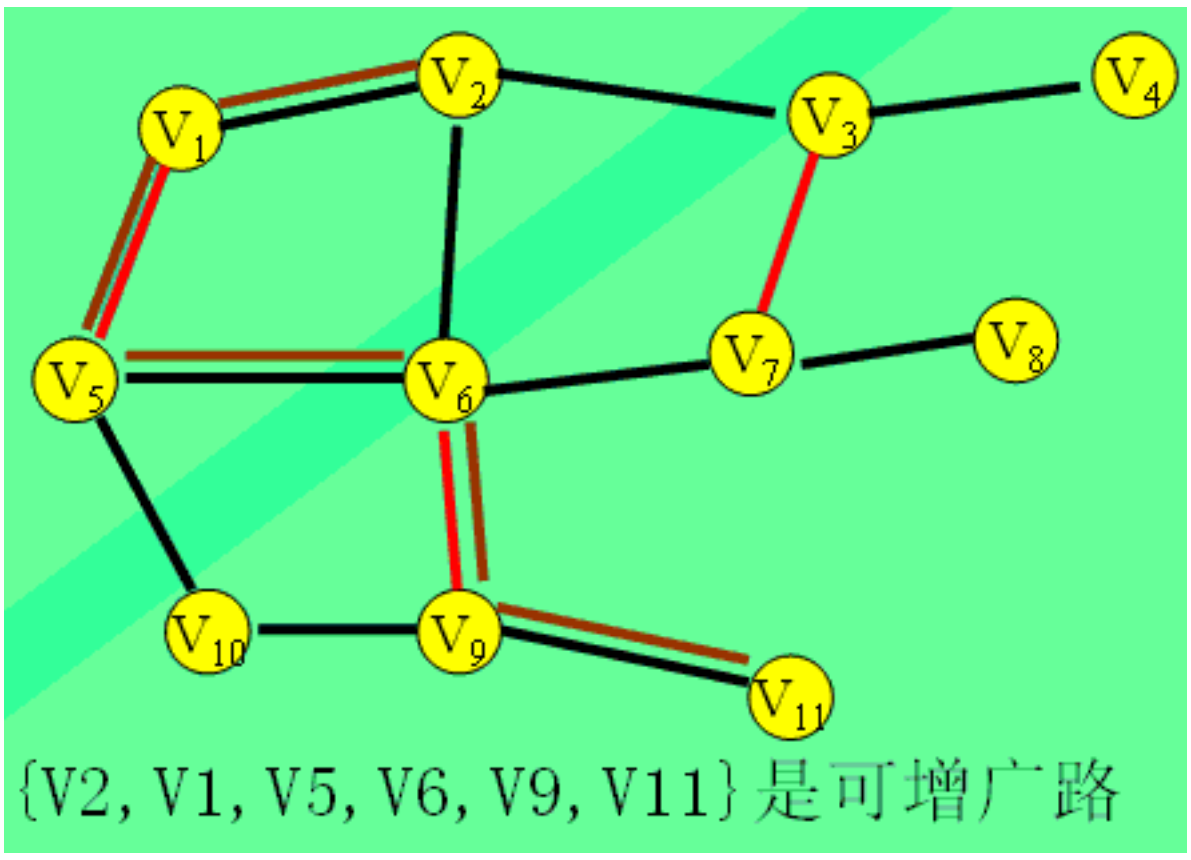
这是一种用增广路求二分图最大匹配的算法。它由匈牙利数学家 Edmonds 于 1965 年提出，因而得名。 定义 未盖点：设 Vi 是图 G 的一个顶点，如果 Vi 不与任意一条属于匹配 M 的边相关联，就称 Vi 是一个未盖点。



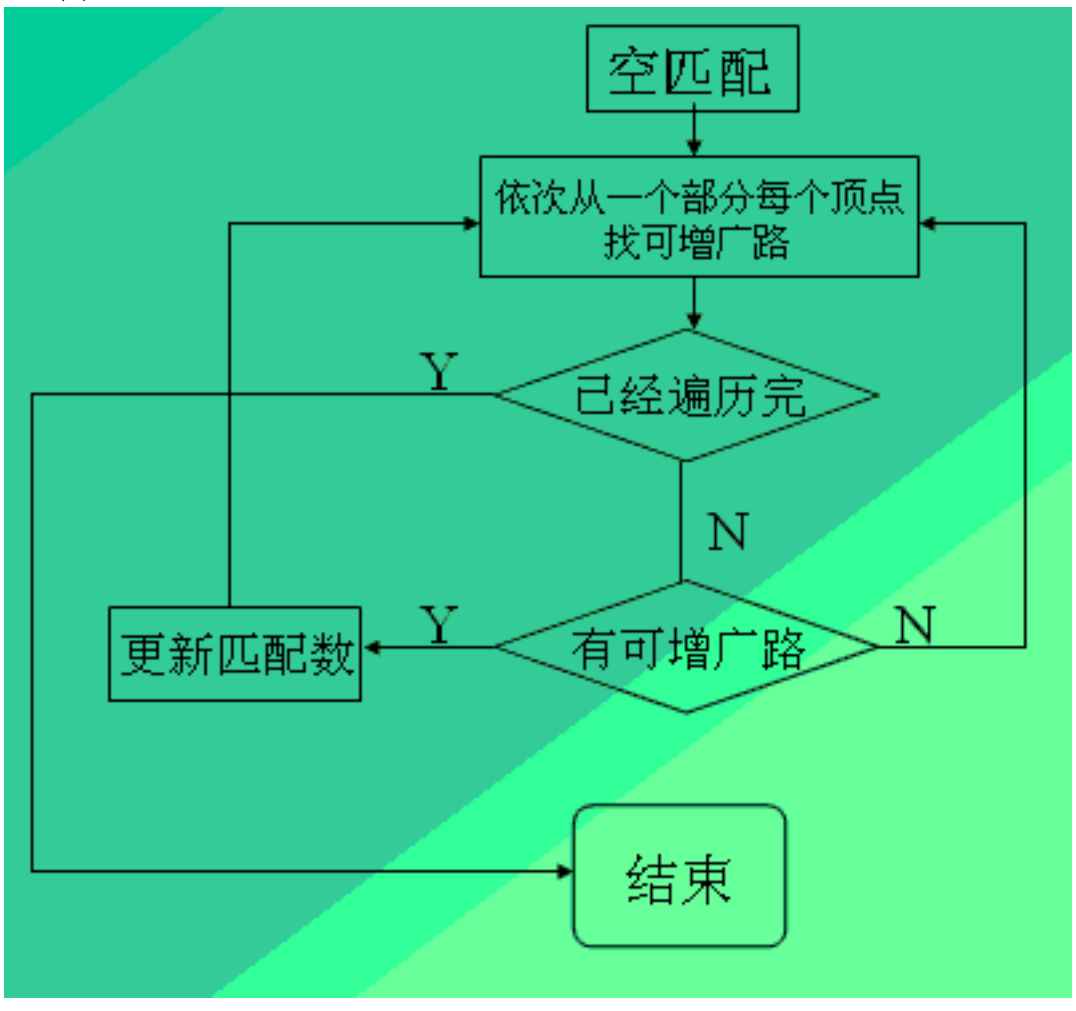
交错路：设 P 是图 G 的一条路，如果 P 的任意两条相邻的边一定是一条属于 M 而另一条不属于 M，就称 P 是一条交错路。

可增广路：两个端点都是未盖点的交错路叫做可增广路。





流程图



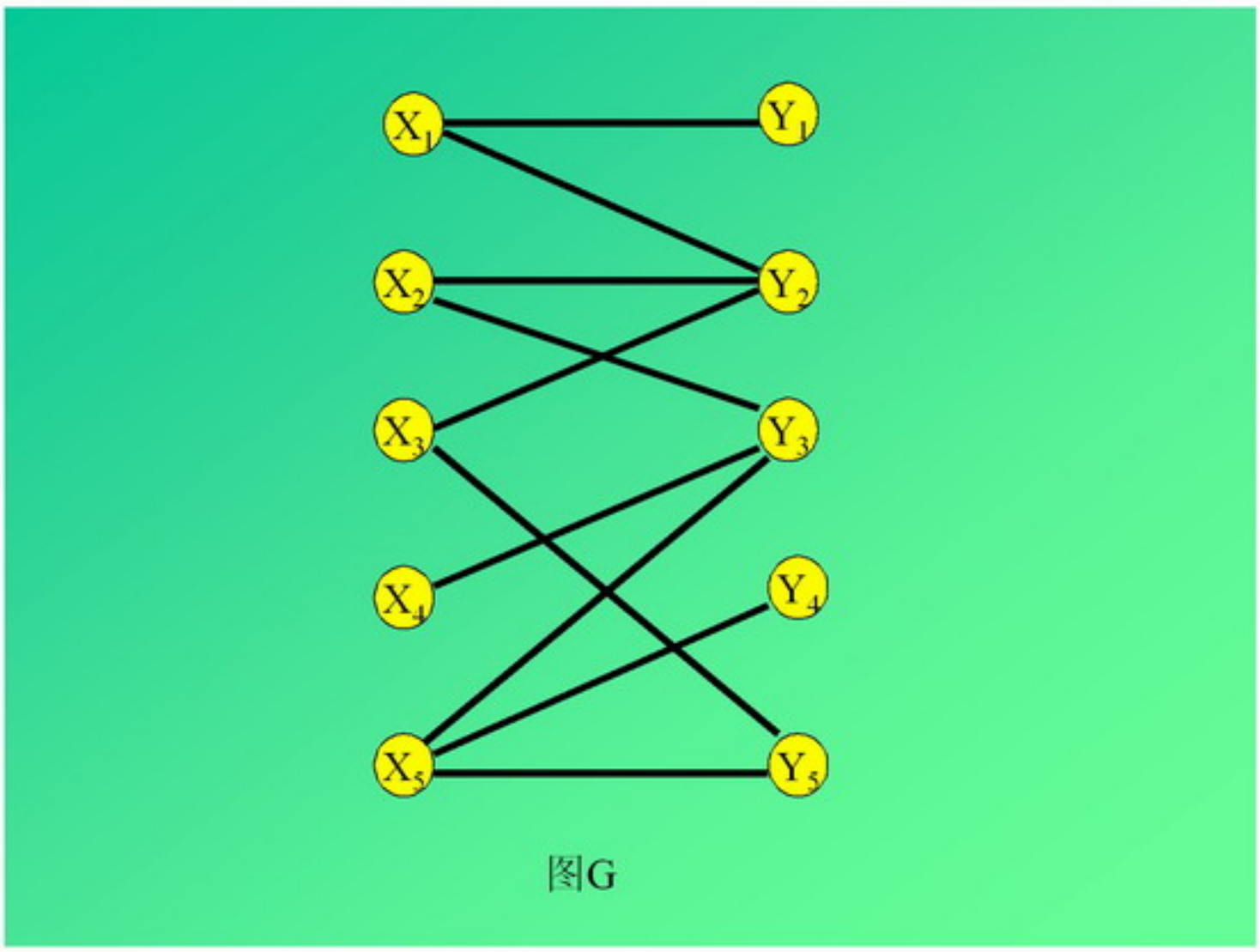
伪代码：

[cpp] view plaincopy

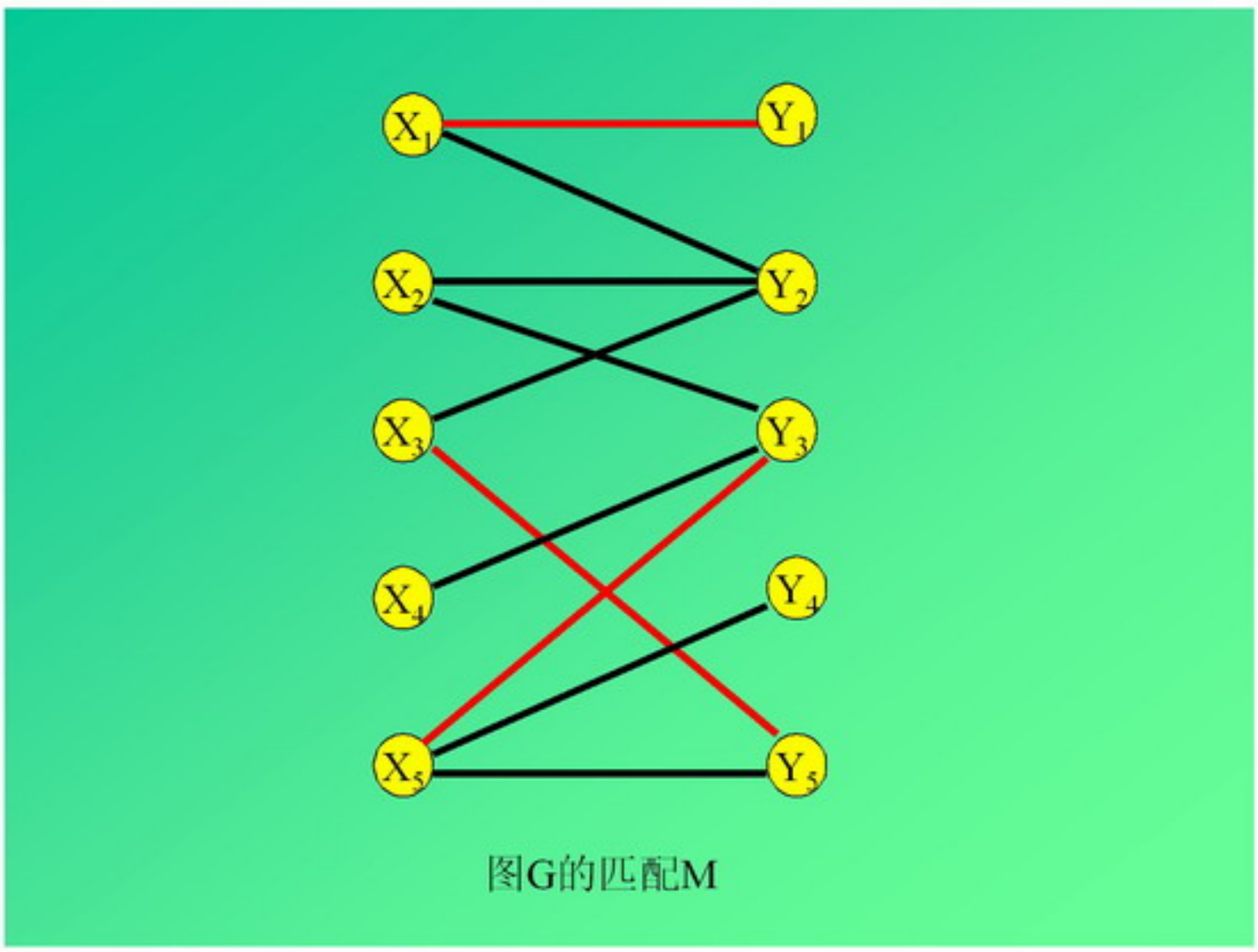
```
bool 寻找从 k 出发的对应项出的可增广路
{
    while (从邻接表中列举 k 能关联到顶点 j)
    {
        if (j 不在增广路上)
        {
            把 j 加入增广路;
            if (j 是未盖点 或者 从 j 的对应项出发有可增广路)
            {
                修改 j 的对应项为 k;
                则从 k 的对应项出有可增广路,返回 true;
            }
        }
    }
    则从 k 的对应项出没有可增广路,返回 false;
}

void 匈牙利 hungary()
{
    for i->1 to n
    {
        if (则从 i 的对应项出有可增广路)
            匹配数++;
    }
    输出 匹配数;
}
```

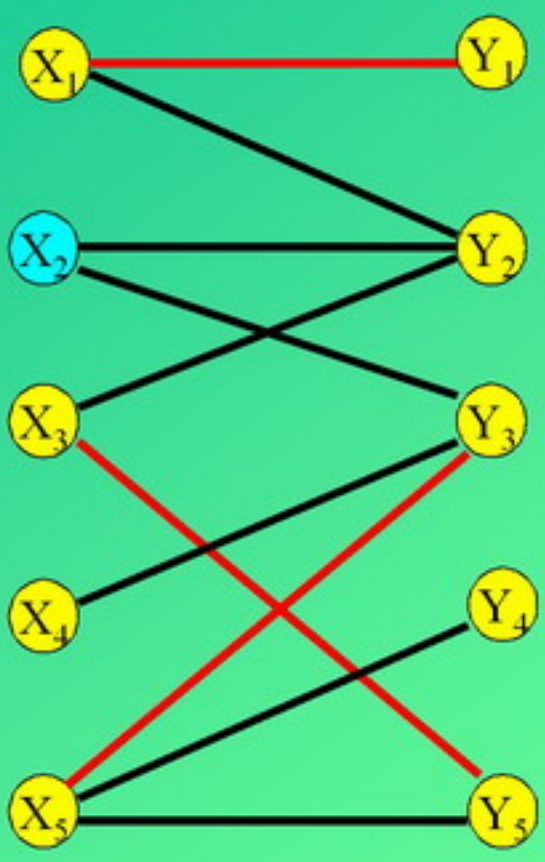
演示：



图G



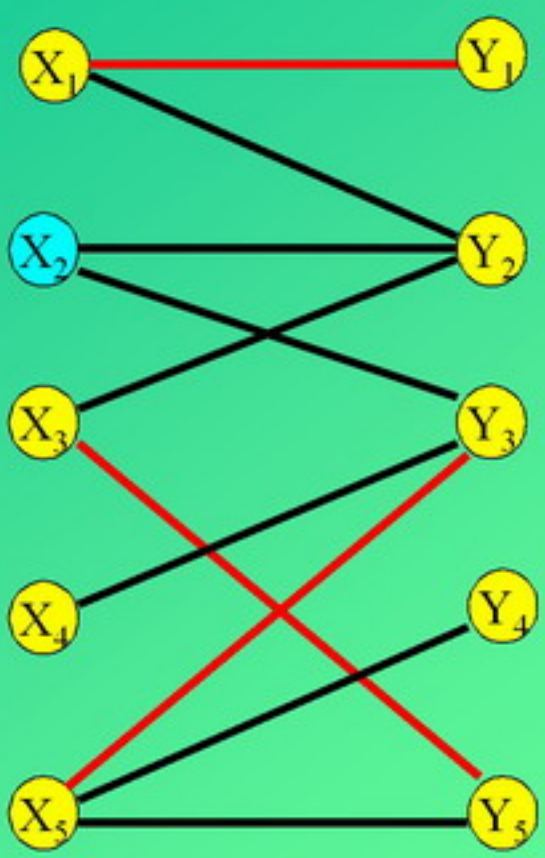
图G的匹配M



X尚未饱合，找出未饱合点X_2

图G的匹配M

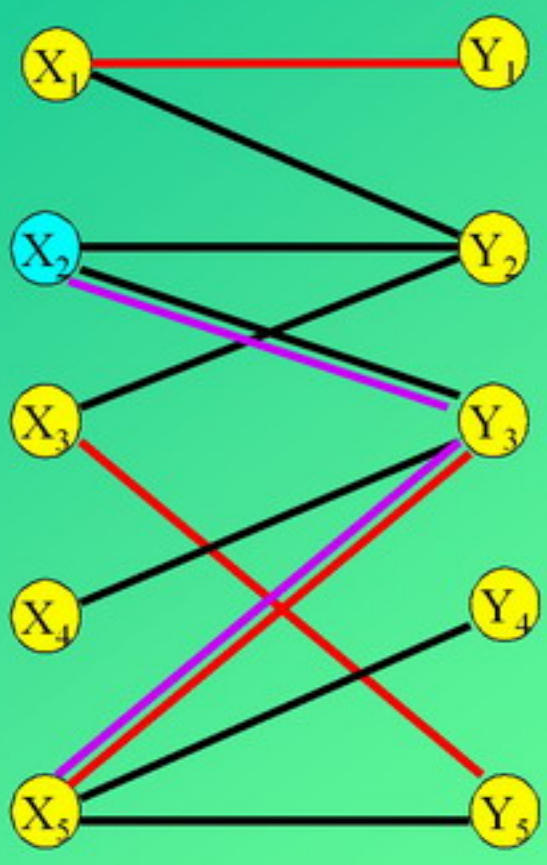
S={X2}
T={}



X尚未饱合，找出未饱合点X_2

图G的匹配M

S={X_2}
N(S)={Y_2,Y_3}
T={Y_3}

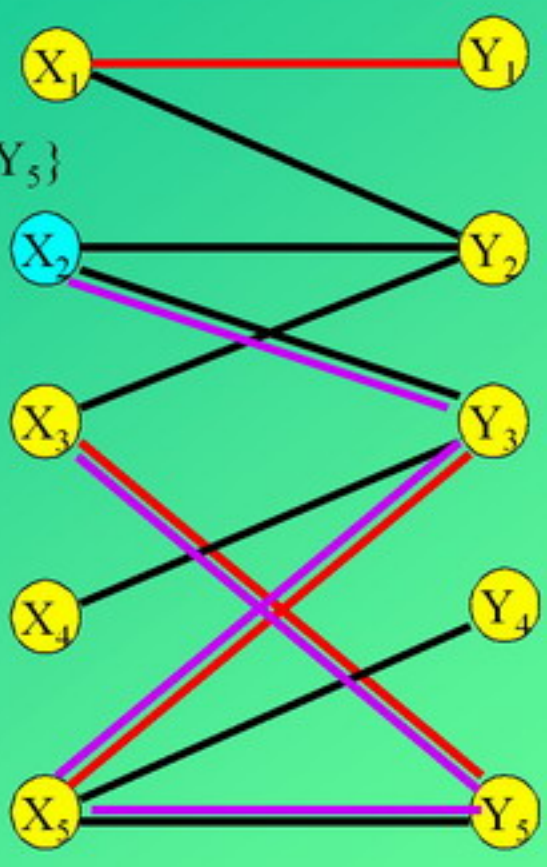


图G的匹配M

取Y_3

Y_3已饱合且存在Y_3X_5属于M，Y_3加入集合T，X_5加入集合S

S={X_2，X_5}
N(S)={Y_2,Y_3,Y_4,Y_5}
T={Y_3}

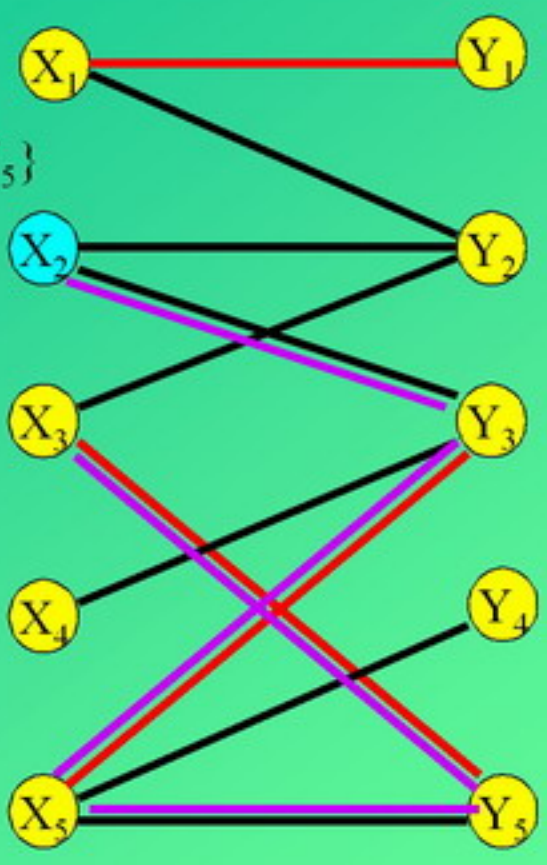


图G的匹配M

取Y_5

Y_5已饱合且存在Y_5X_3属于M，Y_5加入集合T，X_3加入集合S

S={X_2，X_5，X_3}
N(S)={Y_2,Y_3,Y_4,Y_5}
T={Y_3，Y_5}

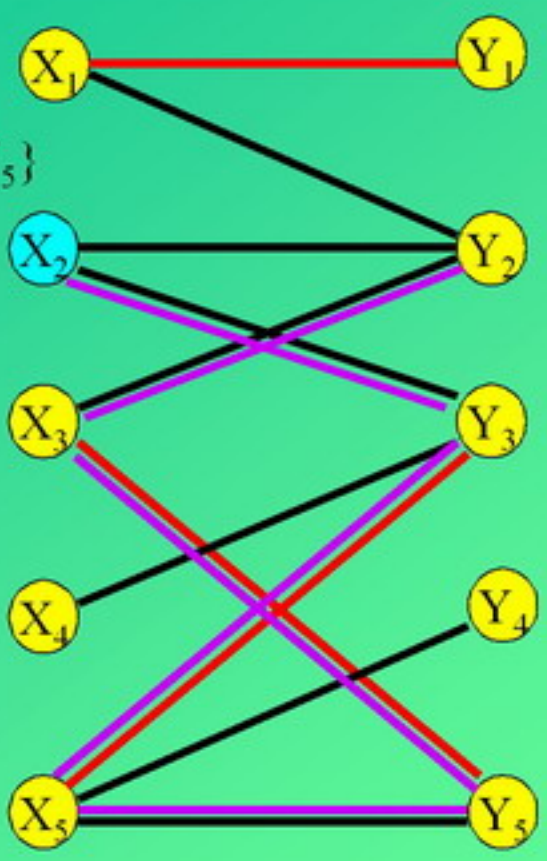


图G的匹配M

取Y_5

Y_5已饱合且存在Y_5X_3属于M，Y_5加入集合T，X_3加入集合S

S={X_2，X_5，X_3}
N(S)={Y_2,Y_3,Y_4,Y_5}
T={Y_3，Y_5}



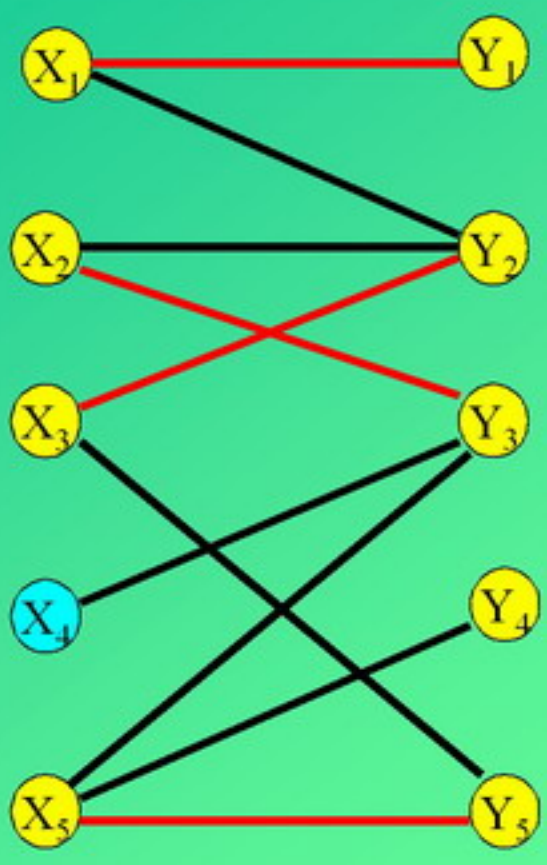
图G的匹配M

取Y_2

Y_2未饱合，存在$P= X_2\ Y_3X_5\ Y_5\ X_3Y_2$的可改进路

对P改进

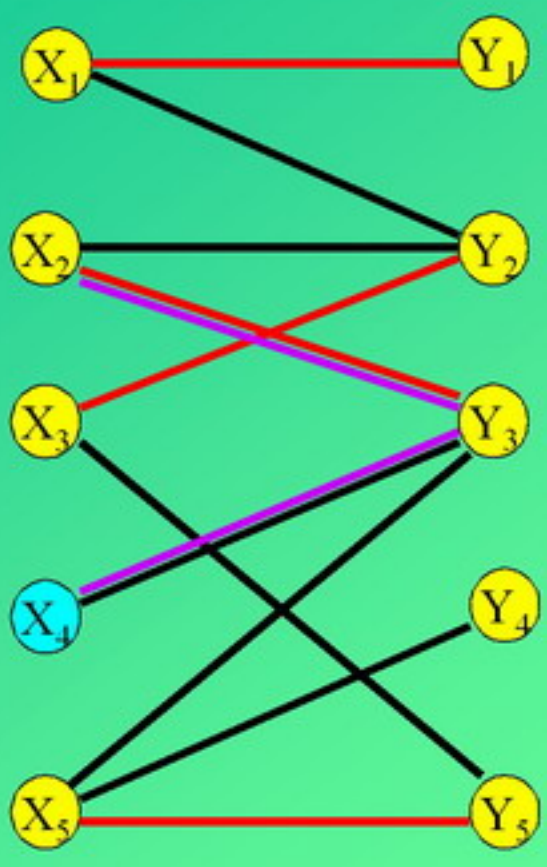
S={X_4}
N(S)={Y_3}
T={}



图G的匹配M

X尚未饱合，找出未饱合点X_4

S={X_4,X_2}
N(S)={Y_2, Y_3}
T={Y_3}

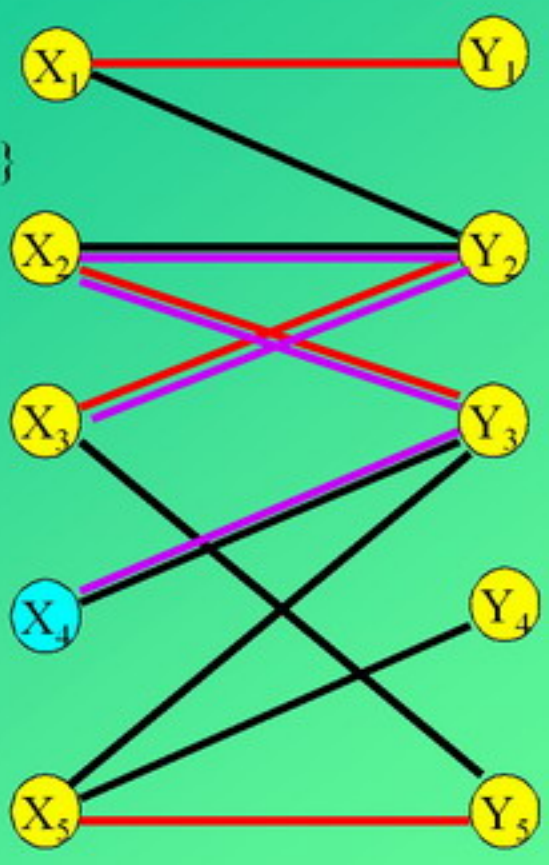


图G的匹配M

取Y_3

Y_3已饱合且存在Y_3X_2属于M，Y_3加入集合T，X_2加入集合S

S={X_4,X_2,X_3}
N(S)={Y_2, Y_3, Y_5}
T={Y_3, Y_2}

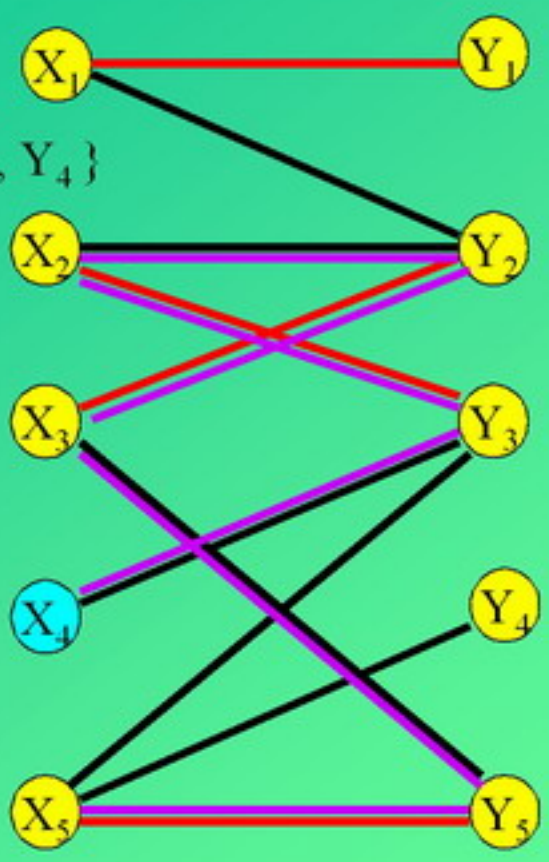


图G的匹配M

取Y_2

Y_2已饱合且存在Y_2X_3属于M，Y_2加入集合T，X_3加入集合S

S={X_4,X_2,X_3,X_5}
N(S)={Y_2, Y_3, Y_5, Y_4}
T={Y_3, Y_2, Y_5}

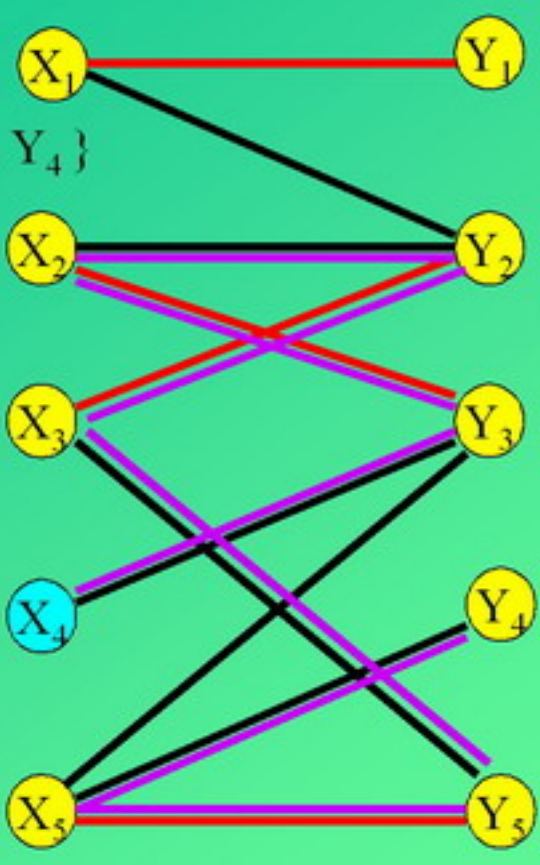


图G的匹配M

取Y_5

Y_5已饱合且存在Y_2X_5属于M，Y_5加入集合T，X_5加入集合S

S={X_4,X_2,X_3,X_5}
N(S)={Y_2, Y_3, Y_5, Y_4}
T={Y_3, Y_2, Y_5}

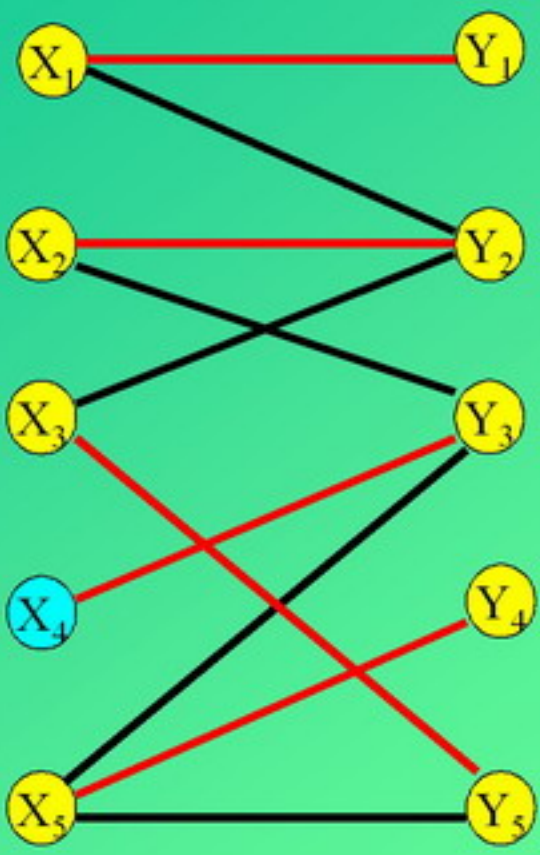


取Y_4

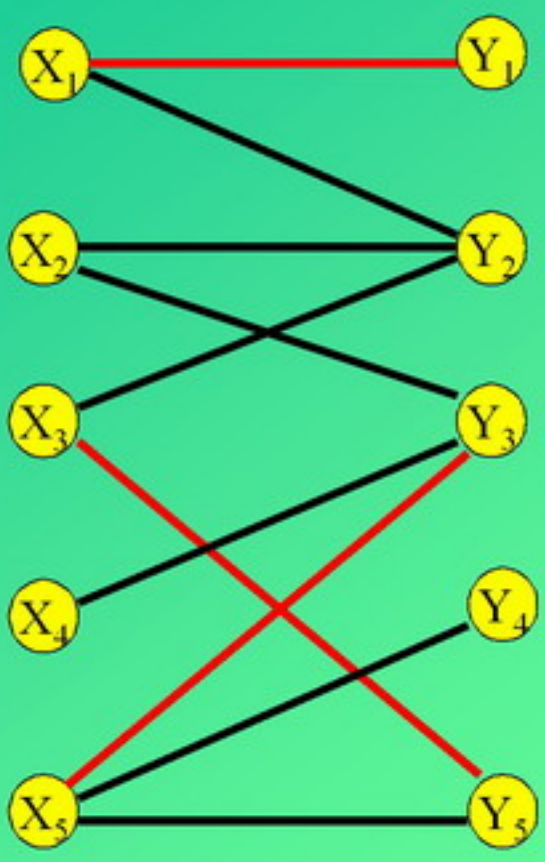
Y_4未饱合，存在
P= $X_4Y_3X_2Y_2X_3Y_5X_5Y_4$
的可改进路

对P改进

图G的匹配M

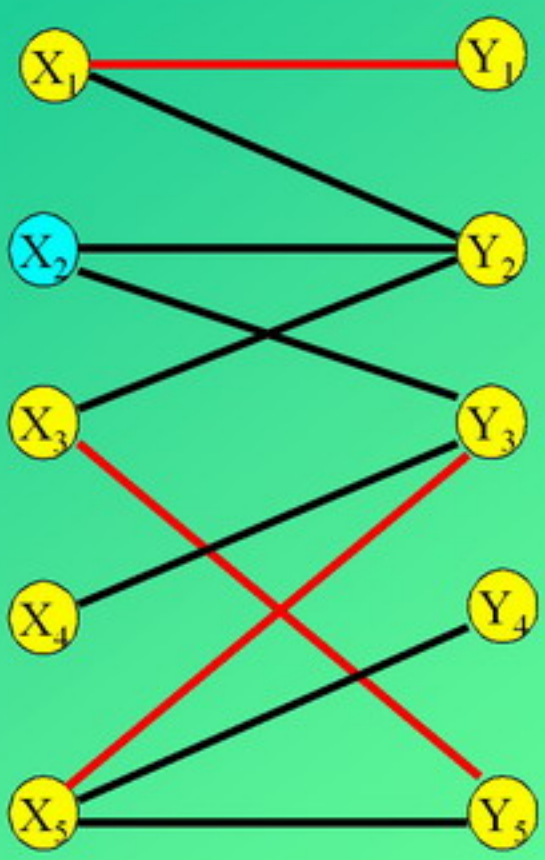


X已饱合得图G的最大匹配M



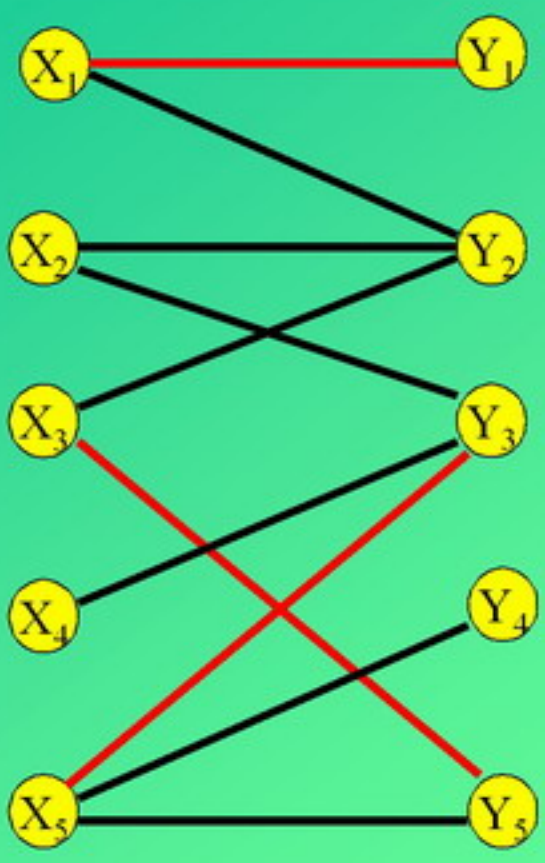
图G的匹配M

S={X2}
T={}

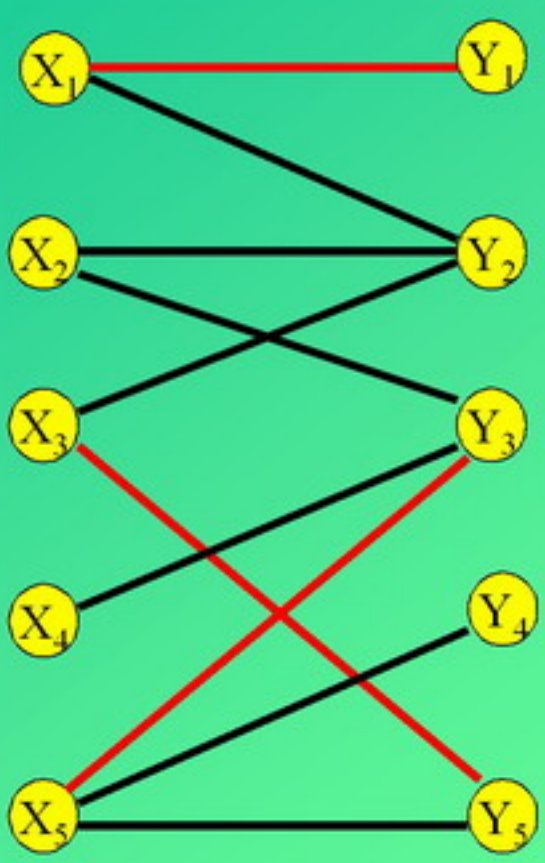


X尚未饱合，找出未饱合点X2

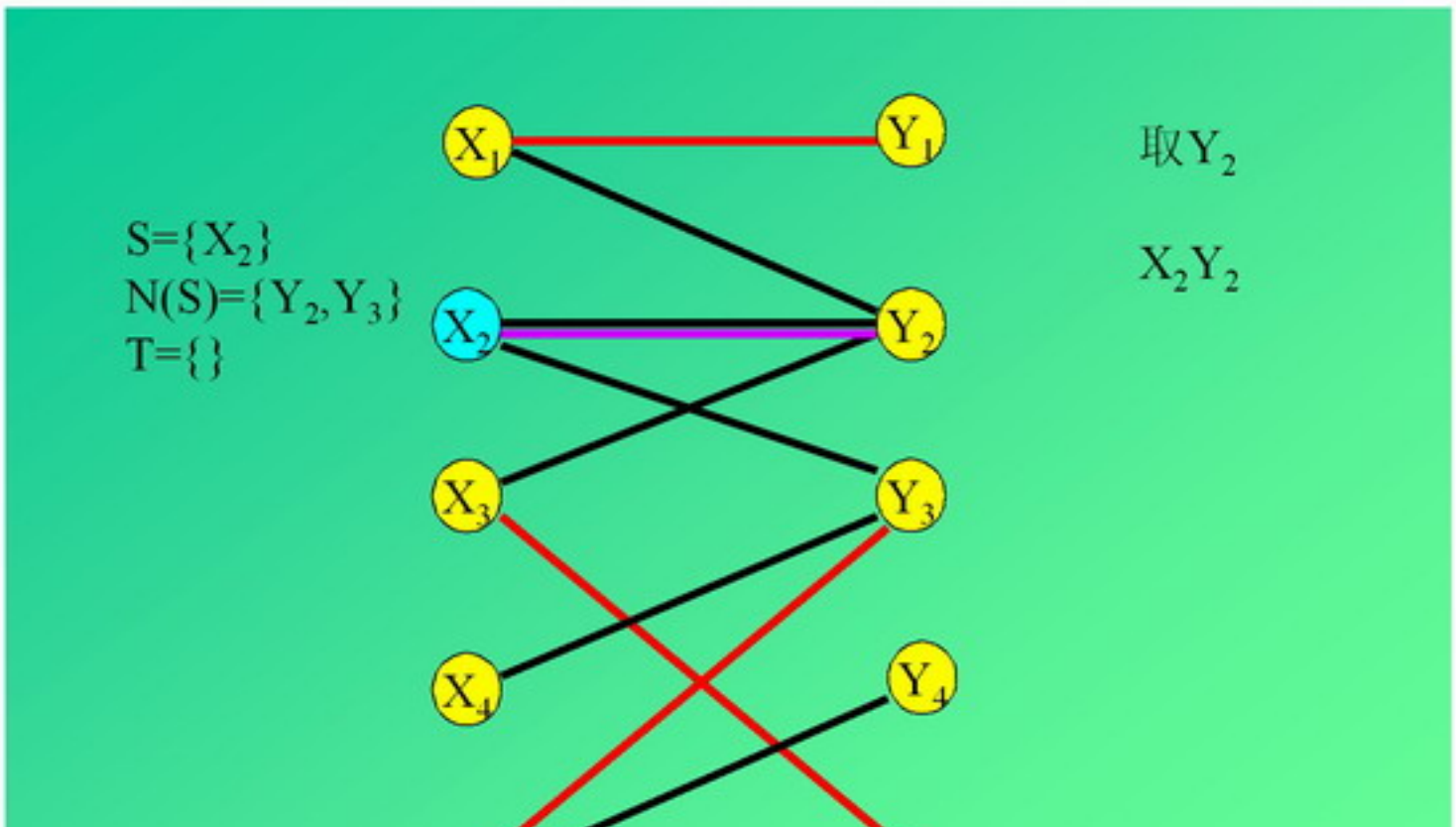
图G的匹配M



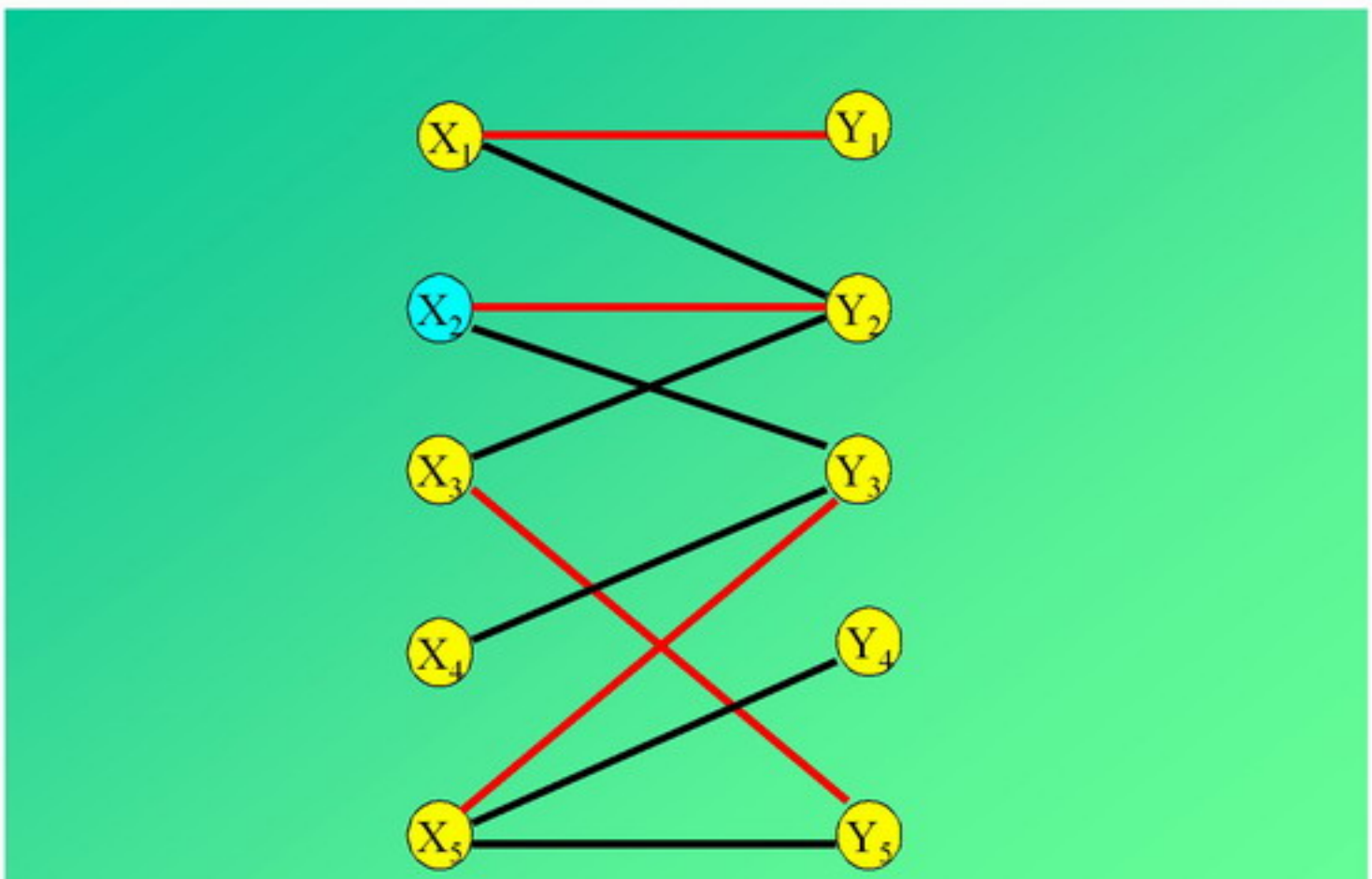
第二种变化



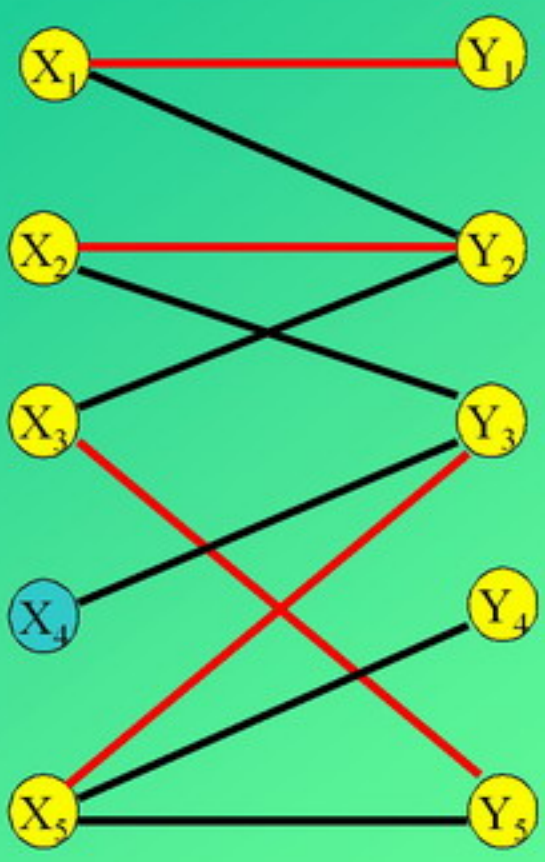
图G的匹配M



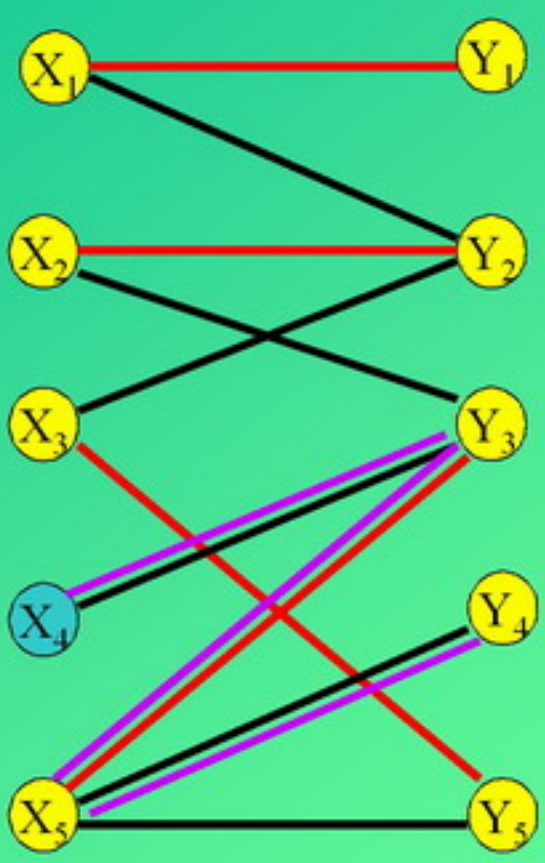
图G的匹配M



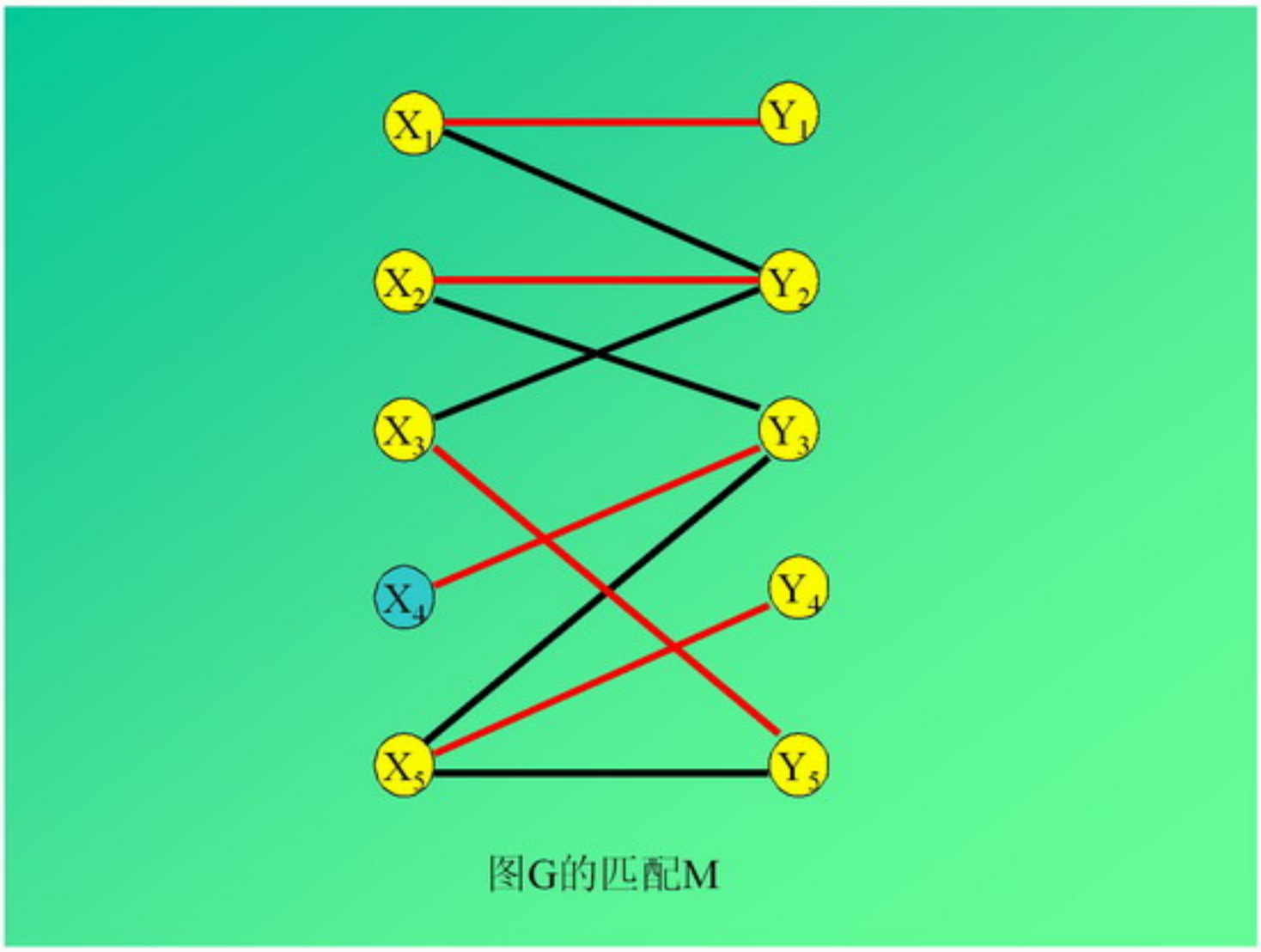
图G的匹配M



图G的匹配M



图G的匹配M



图G的匹配M

C 实现（作者 BYVoid）

[cpp] view plaincopy

```cpp
#include <stdio.h>
#include <string.h>
#define MAX 102

long n,n1,match;
long adjl[MAX][MAX];
long mat[MAX];
bool used[MAX];

FILE *fi,*fo;

void readfile()
{
    fi=fopen("flyer.in","r");
    fo=fopen("flyer.out","w");
    fscanf(fi,"%ld%ld",&n,&n1);
    long a,b;
    while (fscanf(fi,"%ld%ld",&a,&b)!=EOF)
        adjl[a][ ++adjl[a][0] ]=b;
    match=0;
}

bool crosspath(long k)
{
    for (long i=1;i<=adjl[k][0];i++)
    {
        long j=adjl[k][i];
        if (!used[j])
        {
            used[j]=true;
            if (mat[j]==0 || crosspath(mat[j]))
            {
                mat[j]=k;
                return true;
            }
        }
    }
    return false;
}

void hungary()
{
    for (long i=1;i<=n1;i++)
    {
        if (crosspath(i))
            match++;
        memset(used,0,sizeof(used));
```

```
    }
}

void print()
{
    fprintf(fo,"%ld",match);
    fclose(fi);
    fclose(fo);
}

int main()
{
    readfile();
    hungary();
    print();
    return 0;
}
```

Pascal 实现（作者魂牛）

[delphi] view plaincopy

```
var
  a:array[1..1000,1..1000] of boolean;
  b:array[1..1000] of longint;
  c:array[1..1000] of boolean;
  n,k,i,x,y,ans,m:longint;

function path(x:longint):boolean;
var
  i:longint;
begin
  for i:=1 to n do
  if a[x,i] and not c[i] then
  begin
    c[i]:=true;
    if (b[i]=0) or path(b[i]) then
    begin
      b[i]:=x;
      exit(true);
    end;
  end;
  exit(false);
end;

procedure hungary;
var
  i:longint;
begin
  fillchar(b,sizeof(b),0);
  for i:=1 to m do
  begin
    fillchar(c,sizeof(c),0);
    if path(i) then inc(ans);
  end;
end;

begin
  fillchar(a,sizeof(a),0);
  readln(m,n,k);
  for i:=1 to k do
  begin
    readln(x,y);
    a[x,y]:=true;
  end;
  ans:=0;
  hungary;
  writeln(ans);
end.
```