

# [caojia321的博客](#)

[z oj 1495 哈 又一道dp](#) - | [回首页](#) | [2005年索引](#) | - - [最小路径覆盖](#)

## 最大二分图匹配(匈牙利算法)

最近队里流行二分图,不能落下,呵呵我也看了。二分图指的是这样一种图:其所有的顶点分成两个集合M和N,其中M或N中任意两个在同一集合中的点都不相连。二分图匹配是指求出一组边,其中的顶点分别在两个集合中,并且任意两条边都没有相同的顶点,这组边叫做二分图的匹配,而所能得到的最大的边的个数,叫做最大匹配。

计算二分图的算法有网络流算法和匈牙利算法(目前就知道这两种),其中匈牙利算法是比较巧妙的,具体过程如下(转自组合数学):

令 $g=(x,*,y)$ 是一个二分图,其中 $x=\{x_1,x_2,\dots\}$ , $y=\{y_1,y_2,\dots\}$ .令 $m$ 为 $g$ 中的任意匹配。

1.将 $x$ 的所有不与 $m$ 的边关联的顶点表上 $\forall$ ,并称所有的顶点为未扫描的。转到2。

2.如果在上一步没有新的标记加到 $x$ 的顶点上,则停,否则,转3

3.当存在 $x$ 被标记但未被扫描的顶点时,选择一个被标记但未被扫描的 $x$ 的顶点,比如 $x_i$ ,用 $(x_i)$ 标

记 $y$ 的所有顶点,这些顶点被不属于 $m$ 且尚未标记的边连到 $x_i$ 。

现在顶点 $x_i$ 是被扫描的。如果不存在被标记但未被扫描的顶点,转4。

4.如果在步骤3没有新的标记被标记到 $y$ 的顶点上,则停,否则转5。

5.当存在 $y$ 被标记但未被扫描的顶点时。选择 $y$ 的一个被标记但未被扫描的顶点,比如 $y_j$ ,

用 $(y_j)$ 标记 $x$ 的顶点,这些顶点被属于 $m$ 且尚未标记的边连到 $y_j$ 。现在,顶点 $y_j$ 是被扫描的。

如果不存在被标记但未被扫描的顶点则转道2。

由于每一个顶点最多被标记一次且由于每一个顶点最多被扫描一次,本匹配算法在有限步内终止。

代码实现:

bfs过程:

```
#include<stdio.h>
#include<string.h>
main()
{
    bool map[100][300];
    int
    i,i1,i2,num,num1,que[300],cou,stu,match1[100],match2[300],pque,p1,now,prev[300],
    scanf("%d",&n);
```

```
for(i=0;i<n;i++)
{
    scanf("%d",&cou,&stu);
    memset(map,0,sizeof(map));
    for(il=0;il<cou;il++)
    {
        scanf("%d",&num);
        for(i2=0;i2<num;i2++)
        {
            scanf("%d",&num1);
            map[il][num1-1]=true;
        }
    }
    num=0;
    memset(match1,int(-1),sizeof(match1));
    memset(match2,int(-1),sizeof(match2));
    for(il=0;il<cou;il++)
    {
        pl=0;
        pque=0;
        for(i2=0;i2<stu;i2++)
        {
            if(map[il][i2])
            {
                prev[i2]=-1;
                que[pque++]=i2;
            }
            else
                prev[i2]=-2;
        }
        while(pl<pque)
        {
            now=que[pl];
            if(match2[now]==-1)
                break;
            pl++;
            for(i2=0;i2<stu;i2++)
            {
                if(prev[i2]==-2&&map[match2[now]][i2])
                {
                    prev[i2]=now;
                    que[pque++]=i2;
                }
            }
        }
        if(pl==pque)
            continue;
        while(prev[now]>=0)
        {
            match1[match2[prev[now]]]=now;
            match2[now]=match2[prev[now]];
            now=prev[now];
        }
        match2[now]=il;
        match1[il]=now;
        num++;
    }
    if(num==cou)
        printf("YES\n");
}
```

```
    else
        printf("NO\n");
    }
}
```

dfs实现过程：

```
#include<stdio.h>
#include<string.h>
#define MAX 100

bool map[MAX][MAX],searched[MAX];
int prev[MAX],m,n;

bool dfs(int data)
{
    int i,temp;
    for(i=0;i<m;i++)
    {
        if(map[data][i]&&!searched[i])
        {
            searched[i]=true;
            temp=prev[i];
            prev[i]=data;
            if(temp!=-1||dfs(temp))
                return true;
            prev[i]=temp;
        }
    }
    return false;
}

main()
{
    int num,i,k,temp1,temp2,job;
    while(scanf("%d",&n)!=EOF&&n!=0)
    {
        scanf("%d%d",&m,&k);
        memset(map,0,sizeof(map));
        memset(prev,int(-1),sizeof(prev));
        memset(searched,0,sizeof(searched));
        for(i=0;i<k;i++)
        {
            scanf("%d%d%d",&job,&temp1,&temp2);
            if(temp1!=0&&temp2!=0)
                map[temp1][temp2]=true;
        }
        num=0;
        for(i=0;i<n;i++)
        {
            memset(searched,0,sizeof(searched));
            dfs(i);
        }
        for(i=0;i<m;i++)
        {
            if(prev[i]!=-1)
                num++;
        }
        printf("%d\n",num);
    }
}
```

```
}  
}
```

【作者: [caojia321](#)】 【访问统计:1539】 【2005年07月27日 星期三 01:49】 【[加入博采](#)】 【[打印](#)】

## Trackback

你可以使用这个链接引用该篇文章

<http://publishblog.blogchina.com/blog/tb.b?diaryID=2398867>


## 回复



- 评论人: 看不懂 2006-04-19 21:30:48 


我根本看不懂!!!!!!!!!!!!!!  
SOS!!!!



- 评论人: wcswswws 2005-08-30 17:42:19 

太感谢你了



- 评论人: wcswswws 2005-08-27 21:22:21 

这位仁兄语文很好

验证

码:

评论内容:

5112

提交

重置

[2003-2004 BOKEE.COM All rights reserved](#)  
[Powered by BlogDriver 2.1](#)