ACM 程序设计

计算机学院 刘春英

今天，

请个假，
下周调课(西安)

每周一星（8）：

05072120

第九讲

二分图及其应用

（Bipartite Graph & Applications）

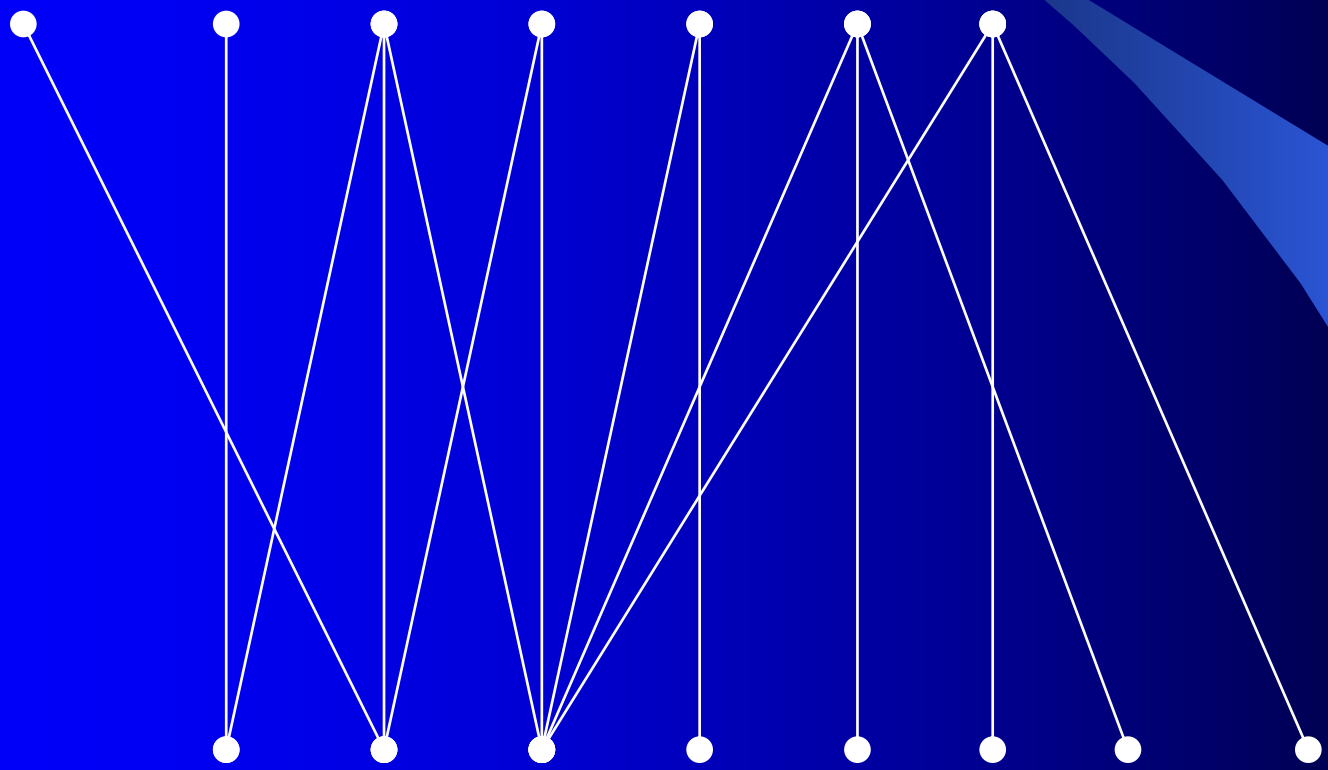
主要内容

- 什么是二分图
- 二分图的最大匹配
- 匈牙利算法
- 二分图的最小顶点覆盖
- DAG图的最小路径覆盖
- 二分图的最大独立集
- 处理技巧

什么是二分图？

- 如果一个图的顶点可以分为两个集合X和Y，图的所有边一定是有一个顶点属于集合X，另一个顶点属于集合Y，则称该图为“二分图”（ Bipartite Graph ）

例：婚配问题



二分图的最大匹配

在二分图的应用中，最常见的就是最大匹配问题，很多其他的问题都可以通过转化为匹配问题来解决。

如何求二分图的最大匹配呢？

经典算法：

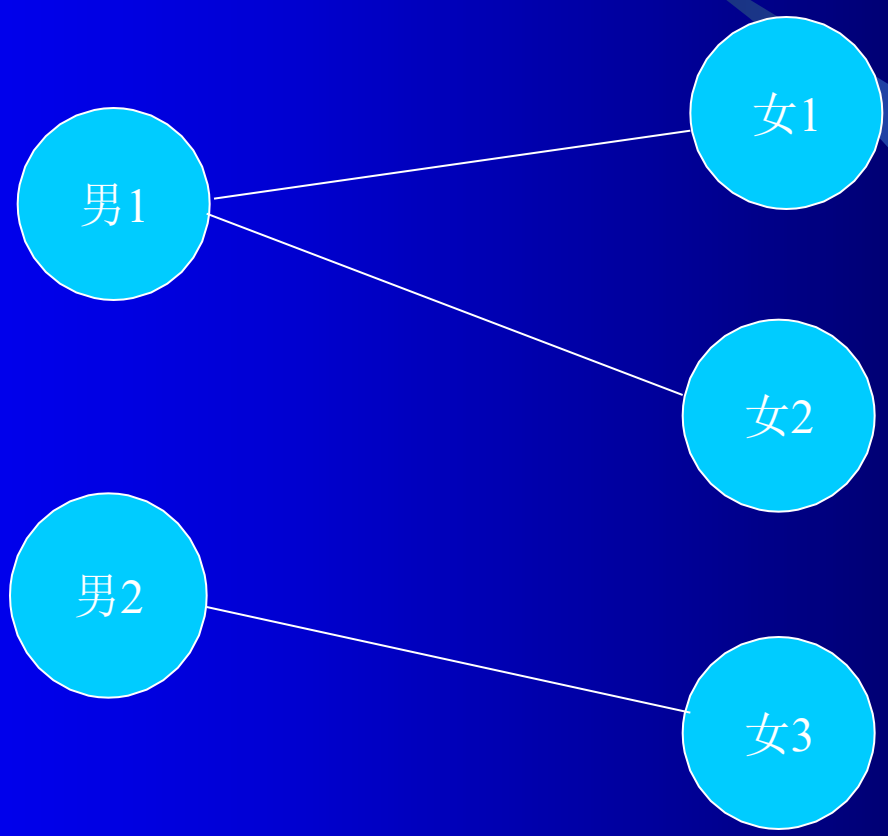
匈牙利算法

```
/*hdoj 1150匈牙利算法 月下版 */
#include<iostream>
#include<string>
#include<vector>
using namespace std;

bool mark1[100],mark2[100];
int list[100];
int n,m,edge,num;
vector<vector<int> > v;
bool dfs(int to)
{
register int i,point,s = list[to];
for(i=0;i<v[s].size();i++)
{
point = v[s][i];
if(!mark2[point])
continue;
mark2[point] = false;
if(list[point]==-1 || dfs(point)){
list[point] = s;
return true;
}
}
return false;
}
void Solve()
{
int i,j,point;
bool flog = false;
memset(mark1,true,sizeof(mar
k1));
memset(list,-1,sizeof(list));
num=0;
for(i=0;i<n;i++)
{
for(j=0;j<v[i].size();j++)
if(list[v[i][j]] == -1)
{
mark1[i] = false;
list[v[i][j]] = i;
num++;
if(i==0) flog = true;
break;
}
}
for(i=0;i<n;i++)
{
if(mark1[i])
{
if(!v[i].empty()){
memset(mark2,true,sizeof(mark2));
for(j=0;j<v[i].size();j++)
{
point = v[i][j];
if(!mark2[point]) continue;
mark2[point] = false;
if(list[point] == -1 || dfs(point))
{
list[point] = i;
num++;
break;
}
}
}
mark1[i] = false;
}
}
if(flog || list[0] != -1)
cout << num-1 << endl;
else cout << num << endl;
}

int main()
{
int i,j,s,d;
while(cin>>n)
{
if(n == 0)break;
v.clear();
v.resize(n);
cin >> m >> edge;
for(i=0;i<edge;i++)
{
cin >> j >> s >> d;
v[s].push_back(d);
}
Solve();
}
return 0;
}
```

匈牙利算法(求二分图最大匹配)

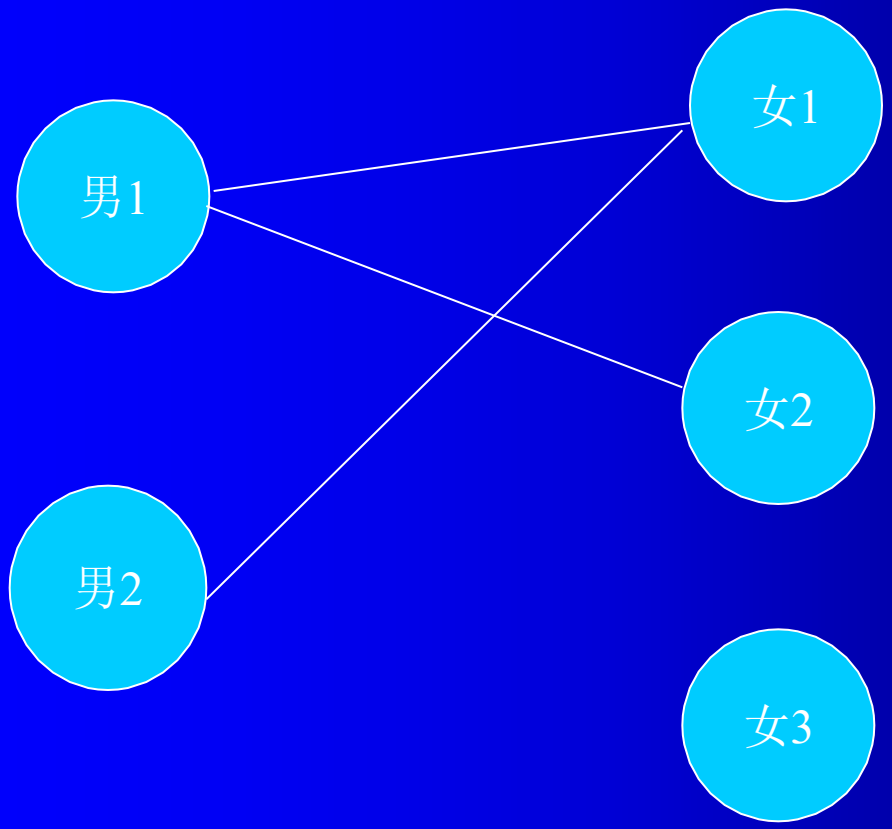
- 谈匈牙利算法自然避不开Hall定理
- Hall定理：对于二分图G，存在一个匹配M，使得X的所有顶点关于M饱和的充要条件是：对于X的任意一个子集A，和A邻接的点集为T(A)，恒有： |T(A)| >= |A|

图示（1）：



返回

图示（2）：



X0=男2

V1={男2}

V2 = Φ

T(V1)={女1}

Y=女1

V1={男2，男1}

V2 ={女1}

Y=女2

M←M⊕E(P)

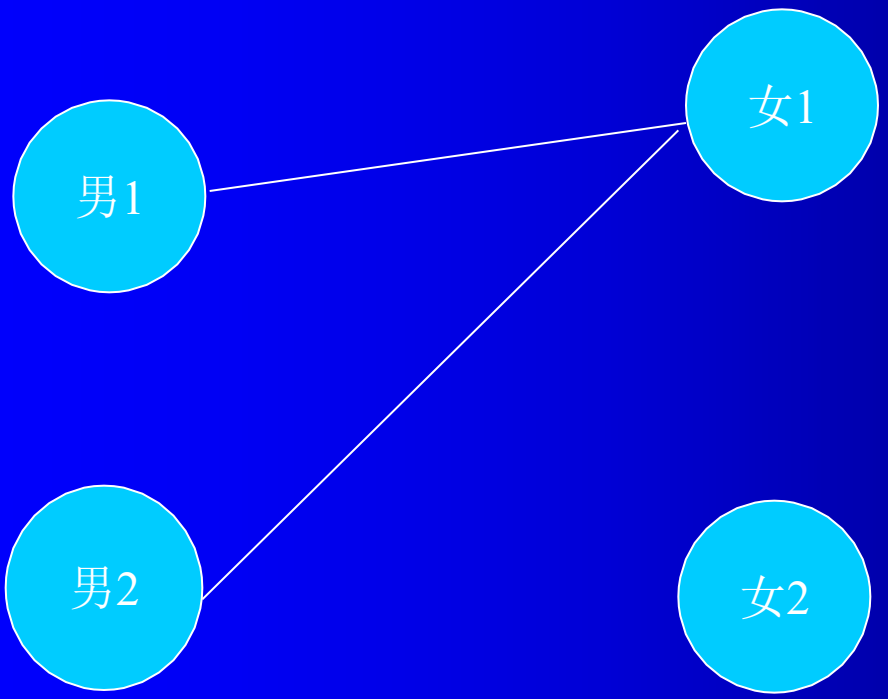
(其中，P是从x0 →y 的可增广道路)

返回

匈牙利算法是基于Hall定理中充分性证明的思想，其基本步骤为：

- 1．任给初始匹配M；
- 2．若X已饱和则结束，否则进行第3步；
- 3．在X中找到一个非饱和顶点x0，
  作V1 ← {x0}，V2 ← Φ；
- 4．若T(V1) = V2则因为无法匹配而停止，否则任选一点y ∈T(V1)\V2；
- 5．若y已饱和则转6，否则做一条从x0 →y的可增广道路P，M←M⊕E(P)，转2；
- 6．由于y已饱和，所以M中有一条边(y,z)，作V1 ← V1 ∪{z}, V2 ← V2 ∪ {y}， 转4；

图示（3）：



X0=男2

V1={男2}

V2 = Φ

T(V1)={女1}

T(V1) != V2

Y=女1

V1={男2，男1}

V2 ={女1}

T(V1)=V2

返回

NOTE:

真正求二分图的最大匹配的题目很少，往往做一些简单的变化，比如——

变种1：
二分图的最小顶点覆盖

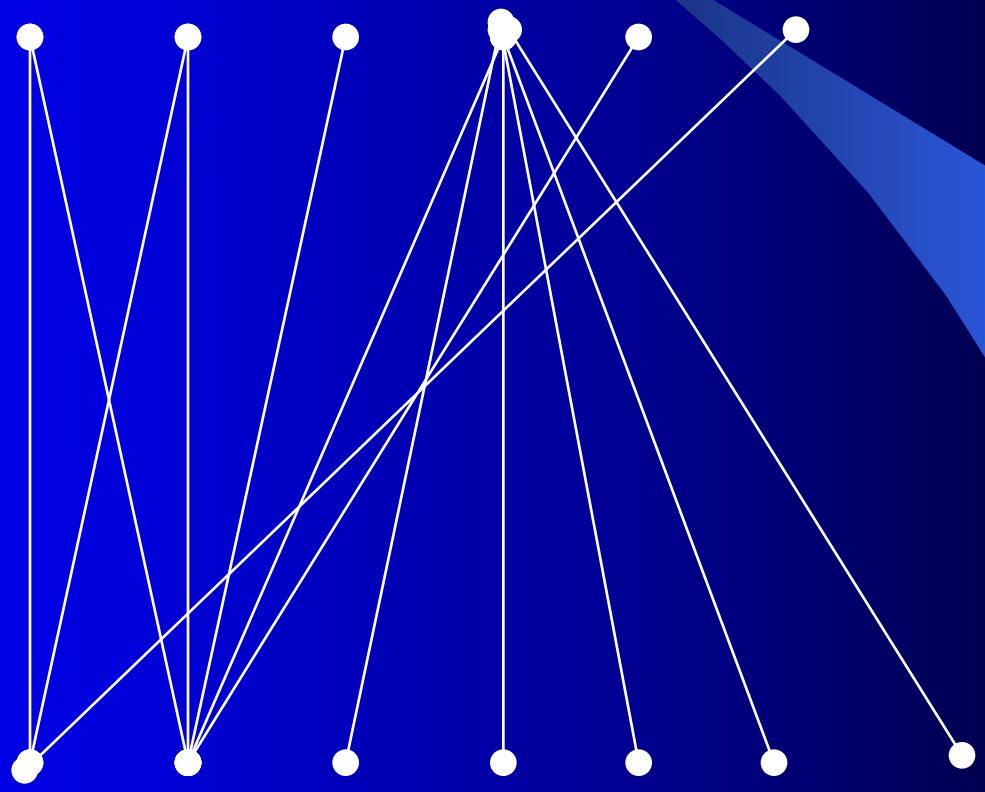
在二分图中求最少的点，让每条边都至少和其中的一个点关联，这就是

“二分图的**最小顶点覆盖**”。

实 例 分 析

例：严禁早恋，违者开除！

男生



女生

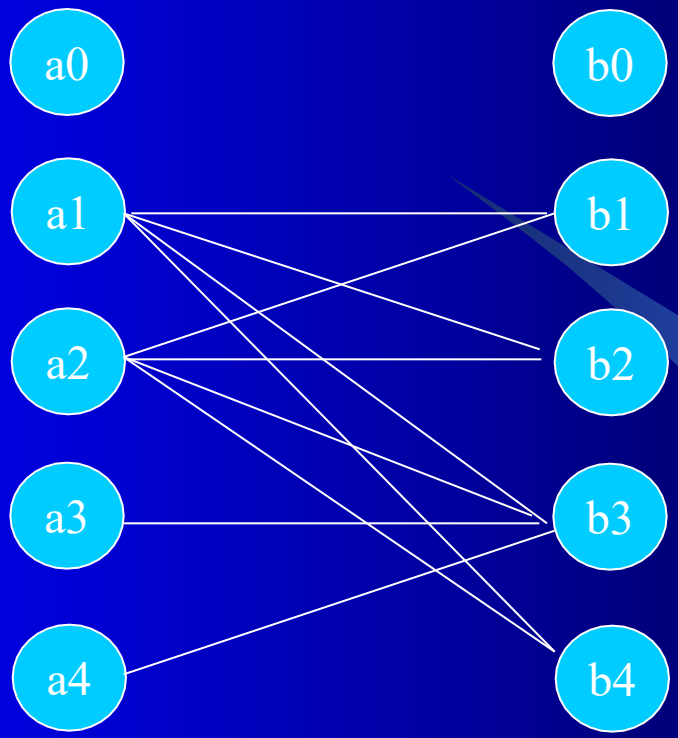
例：任务安排

有两台机器A和B以及N个需要运行的任务。每台机器有M种不同的模式，而每个任务都恰好在一台机器上运行。如果它在机器A上运行，则机器A需要设置为模式ai,如果它在机器B上运行，则机器A需要设置为模式bi。每台机器上的任务可以按照任意顺序执行，但是每台机器每转换一次模式需要重启一次。请合理为每个任务安排一台机器并合理安排顺序，使得机器重启次数尽量少。

——hdoj 1150（LRJ_p331）

——ACM/ICPC Beijing 2002

图示：



结论：

二分图的最小顶点覆盖数 = 二分图的最大匹配数

特别说明:

- 此题需要注意的一点，具体参见：
  http://acm.hdu.edu.cn/forum/read.php?tid=61&keyword=%B6%FE%B7%D6

变种2：
DAG图的最小路径覆盖

用尽量少的不相交简单路径覆盖有向无环图(DAG)G的所有顶点，这就是DAG图的最小路径覆盖问题。

例：空袭（Air Raid）

有一个城镇，它的所有街道都是单行的，并且每条街道都是和两个路口相连。同时已知街道不会形成回路。

你的任务是编写程序求最小数量的伞兵，这些伞兵可以访问（visit）所有的路口。对于伞兵的起始降落点不做限制。

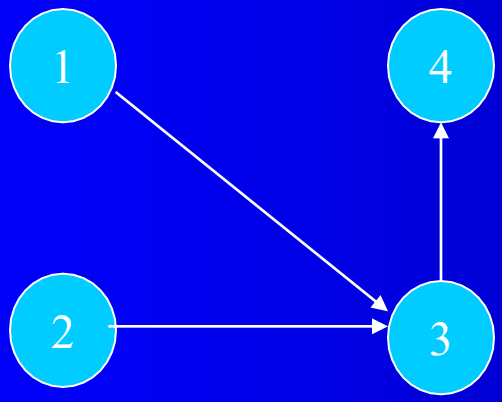
——hdoj 1151

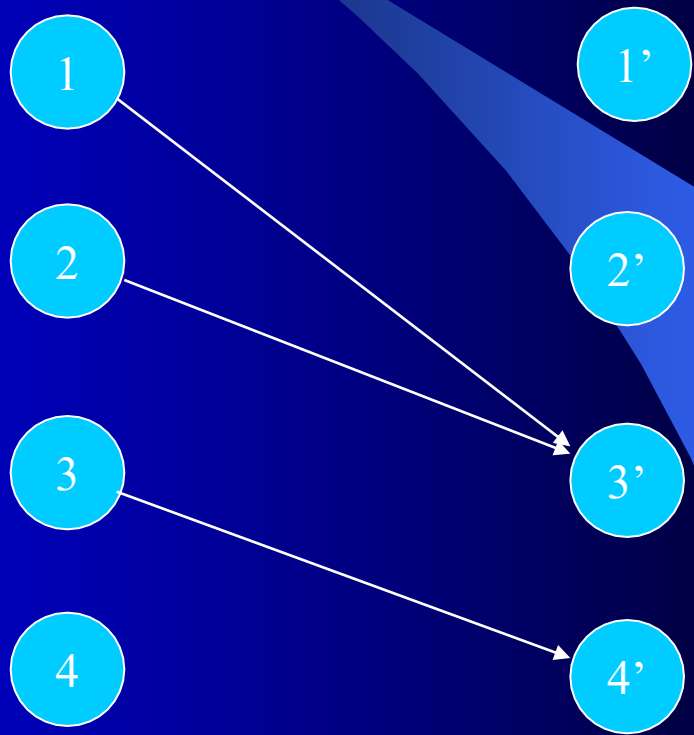
Sample input:

```
4
3
3 4
1 3
2 3
```

Output: 2

“空袭”示意图





结论：

DAG图的最小路径覆盖数=
节点数（n）- 最大匹配数（m）

关键：求二分图的最大匹配数

变种3:

二分图的最大独立集

Girls and Boys

大学二年级的时候，一些同学开始研究男女同学之间的缘分。研究者试图找出没有缘分同学的最大集。程序的结果就是要输出这个集合中学生的数量。

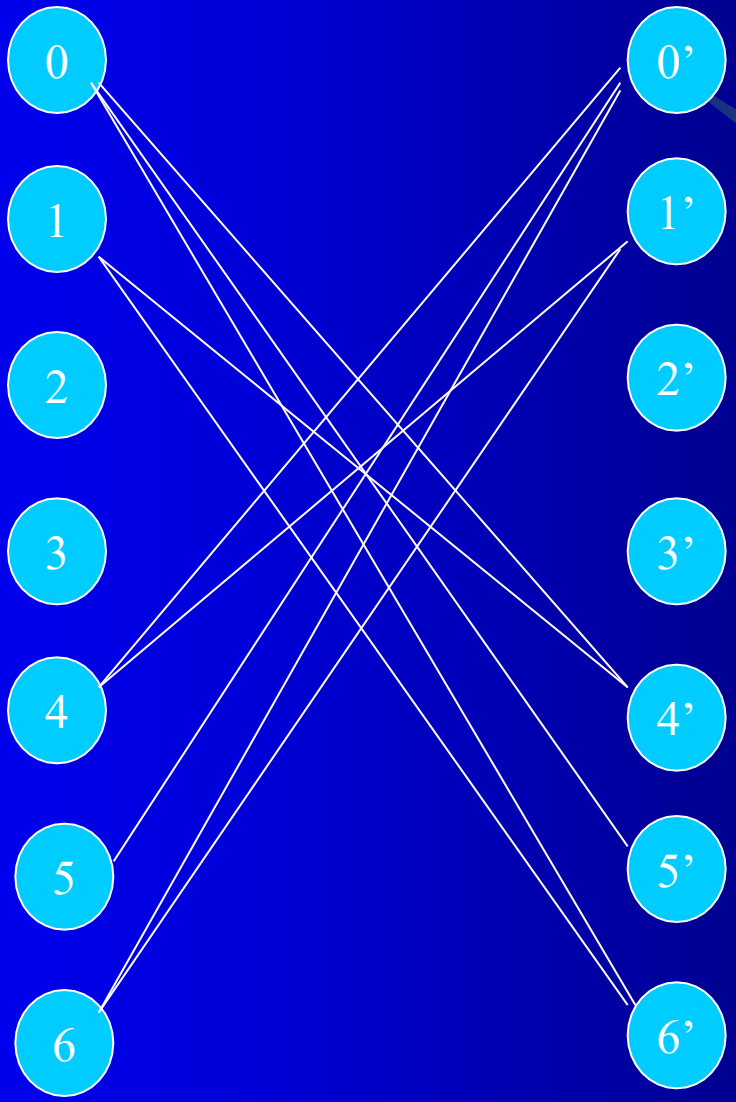
——hdoj_1068

输入数据格式：

7
0: (3) 4 5 6
1: (2) 4 6
2: (0)
3: (0)
4: (2) 0 1
5: (1) 0
6: (2) 0 1

输出：5

“Girls and Boys”示意图

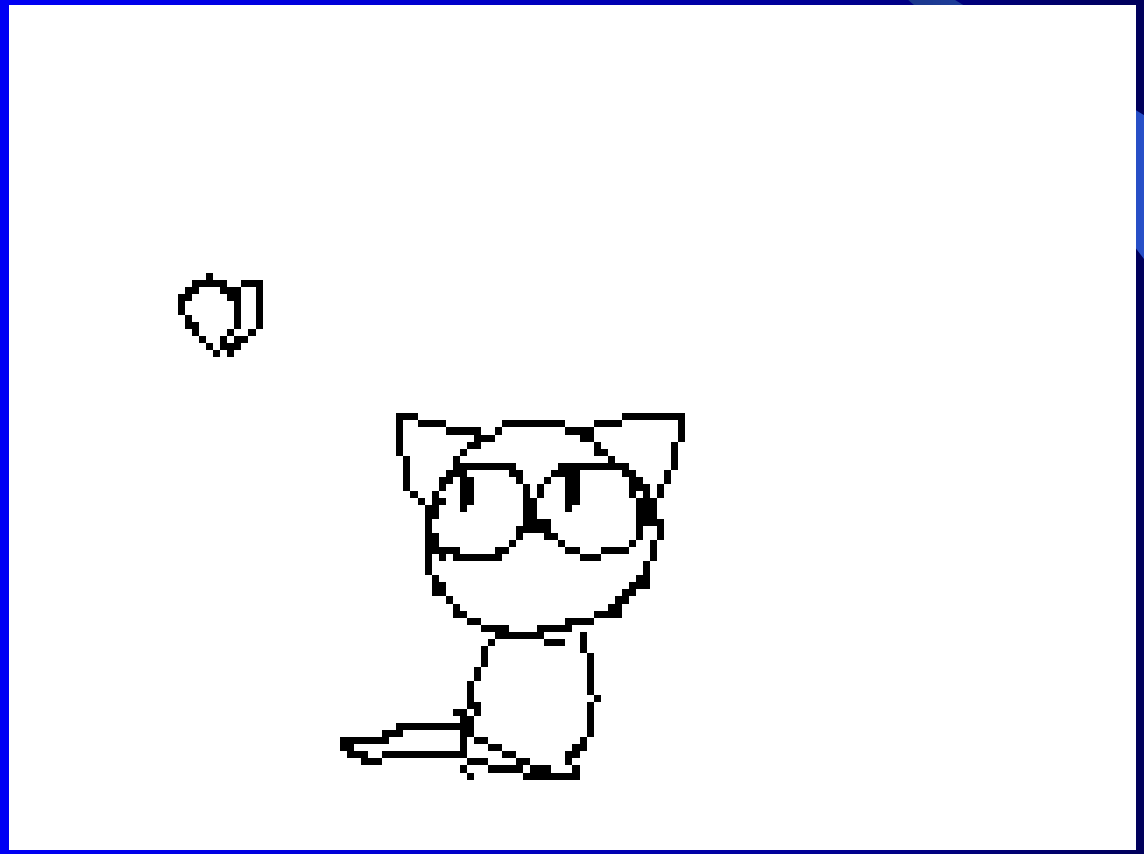


结论：

二分图的最大独立集数=
节点数（n）— 最大匹配数（m）

关键：求二分图的最大匹配数

Any Questions?



附：二分匹配练习题：(HDOJ)

- 1068 (二分图最大独立集= n - m)
- 1150 (二分图最小顶点覆盖= m)
- 1151 (二分图最小路径覆盖= n - m)

别忘了，

下周调课(西安)

课后一定要练习！

ACM,
天天见!