二分图匹配入门

by Hzyoi

定义

- 二分图又称作二部图，是图论中的一种特殊模型。设G=(V,E)是一个无向图，如果顶点V可分割为两个互不相交的子集(A,B)，并且图中的每条边（i，j）所关联的两个顶点i和j分别属于这两个不同的顶点集(i in A,j in B)，则称图G为一个二分图。 ——from百度百科
- 用更通俗的话来讲，二分图是一个可以被分为两个互不相交的子图，没有边连接同一子图内的两个点的图。

更多的定义

- 匹配：在图论中，一个匹配（matching）是一个边的集合，其中任意两条边都没有公共顶点。
- 匹配点、匹配边、未匹配点、非匹配边：它们的含义非常显然，我就不说了。
- 最大匹配：一个图所有匹配中，所含匹配边数最多的匹配，称为这个图的最大匹配。
- 完美匹配：如果一个图的某个匹配中，所有的顶点都是匹配点，那么它就是一个完美匹配。显然，完美匹配一定是最大匹配，但并非每个图都存在完美匹配。

匈牙利算法

- 定义们：
- 交替路：从一个未匹配点出发，依次经过非匹配边、匹配边、非匹配边，……，形成的路径叫交替路。
- 增广路：从一个未匹配点出发，走交替路，如果结束于另一个未匹配点（出发的点不算），则这条交替路称为增广路。
- 增广路有一个重要特点：非匹配边比匹配边多一条。因此，研究增广路的意义是改进匹配。只要把增广路中的匹配边和非匹配边的身份交换即可。由于中间的匹配节点不存在其他相连的匹配边，所以这样做不会破坏匹配的性质。交换后，图中的匹配边数目比原来多了 1 条。
- 我们可以通过重复找增广路来增加匹配中的匹配边。找不到增广路时，达到最大匹配（~~我不会证明的~~增广路定理）。

- 代码：

```cpp
#include <cstdio>
#include <cstring>
const int N=1005,M=3005;
struct edge{int en,Last;}E[M];
int num[N],Last[N],Cnt;
bool Cant[N];
inline void Link(const int &st,const int &en)
{
    E[Cnt]=(edge){en,Last[st]},Last[st]=Cnt++;
}
bool find(int x)
{
    for (int i=Last[x];~i;i=E[i].Last)
        if (!Cant[E[i].en])    //不在交替路中
        {
            Cant[E[i].en]=1;    //加入交替路
            if (!num[E[i].en] || find(num[E[i].en]))
                //如果找到的点未覆盖则该交替路是增广路，交换匹配边和未匹配边
                return num[E[i].en]=x,1;
        }
    return 0;
}
int main()
{
    int n,m,Ans=0;
    memset(Last,-1,sizeof(Last)),scanf("%d %d",&n,&m);
    for (int i=1,a,b;i<=m;++i) scanf("%d %d",&a,&b),Link(a,b);
    for (int i=1;i<=m;++i)
    {
        memset(Cant,0,sizeof(Cant));
        if (find(i)) ++Ans;
    }
    printf("%d\n",Ans);
}
```

- 时间复杂度：O(n*(n+m))
- 其实上面的代码并没有卵用，二分图匹配的主要难度在建图上。

题目们

- poj1274 - The Perfect Stall
- 题目大意：有n个奶牛和m个谷仓，现在每个奶牛有自己喜欢去的谷仓，并且它们只会去自己喜欢的谷仓吃东西，问最多有多少奶牛能够吃到东西
- $0<n,m<=200$
- 很显然的二分图最大匹配对吧，每头牛向每个谷仓连边，然后直接跑匈牙利算法就好了。

题目们

- poj1274 - Asteroids
- 题目大意：有n个点，横纵坐标都在0到k的范围内，每次可以消掉一行或一列上的所有点，为最少需要几次才能才能消掉所有点。
- $0<n<=10000,0<k<=500$

题解们

- 首先引入一些定义
- 点覆盖：对于图$G=(V,E)$中的一个点覆盖是一个集合$S\subseteq V$使得每一条边至少有一个端点在S中。
- 然后我们找出所有出现过的横纵坐标，然后将每个点的横坐标向它的纵坐标连边，此时每条图上的每条边代表一个点，如果一条边连着的两个点中有任意一个被选中，这条边代表的点最小点就被消掉了。所以最小点覆盖就是答案。这个比较显然。
- 然后有一个的定理，最小点覆盖数等于最大匹配数，所以直接在图上跑匈牙利算法就好了。

题目们

- poj3692 - Kindergarten
- 题目大意：有一群人，里面有两类人，分别有n个和m个。里面同一类的人都互相认识，一部分人还认识一些另一类的人。求最多能选出多少人使得所有人都互相认识。
- 0<n,m<=200

题解们

- 继续引入定义：
- 独立集：图的顶点集的子集，其中任意两点之间没有边。
- 把彼此不认识的人连边，图的最大独立集就是最大的彼此认识的人的集合。然后有一个定理就是最大独立集=总点数-最大匹配=总点数-最小点覆盖数。

- 另外二分图还有
- 最小边覆盖（每个点都有至少一个连它的边在集合里，如果某点没有边相连就直接+1）= 二分图点数 - 最大匹配=最大独立集

定义们

- 在一个DAG中，找出最少的路径，使得这些路径经过了所有的点。
- 最小路径覆盖分为最小不相交路径覆盖和最小可相交路径覆盖。
- 最小不相交路径覆盖：每一条路径经过的顶点各不相同。
- 最小可相交路径覆盖：每一条路径经过的顶点可以相同。
- 特别的，每个点自己也可以称为是路径覆盖，只不过路径的长度是0。

最小不相交路径覆盖

- 把原图的每个点V拆成Vx和Vy两个点，如果有一条有向边A->B，那么就加边Ax−>By。这样就得到了一个二分图。那么最小路径覆盖=原图的结点数-新图的最大匹配数。
- 简易证明：Vy有匹配，说明覆盖中有边连到它，Vx有匹配，说明覆盖中有边从它出去。我们每次在二分图里找一条匹配边就相当于把两条路径合成了一条路径，也就相当于路径数减少了1。所以找到了几条匹配边，路径数就减少了多少。一开始有n条，所以有最小路径覆盖=原图的结点数-新图的最大匹配数。
- 因为路径之间不能有公共点，所以加的边之间也不能有公共点，这就是匹配的定义。

最小可相交路径覆盖

- 求出原图的传递闭包（一般用floyd）
- 然后把原图的每个点V拆成Vx和Vy两个点，如果传递闭包中A能到达B，那么就加边Ax－>By。
- 这也叫最小链覆盖
- 链：一些点的集合，链中任意两点x, y，一定满足x能到达y或者y能到达x
- 反链：一些点的集合，链中任意两点x, y，一定满足x不能到达y，y也不能到达x
- **最长反链 = 最小链覆盖**

GL & HF