

# 三元环计数

## 算法例题

### 例题一

洛谷p1989

#### 题意

给定  $n$  个点和  $m$  条边的无向图。定义三元环为满足所有点两两之间有连边的三元组数，求图中的三元环数。

#### 题解

将无向图转化为有向图，其中每条边由原图中度数小的点指向度数大的点，当两个点度数相同时由编号小的点指向编号大的点。

由于上述定义指三元环为满足  $(u,v,w)$  一定满足连边为  $u \rightarrow v, u \rightarrow w, v \rightarrow w$

考虑枚举  $u$  标记  $u \rightarrow w$  的  $w$  后枚举  $v$  然后检查  $v \rightarrow w$  的  $w$  是否被标记。

时间复杂度为  $O(\sum_{i=1}^m \text{out}_{v_i})$  若原图中  $\text{deg}_{v_i} \leq \sqrt{m}$  则当前图中一定也有  $\text{out}_{v_i} \leq \sqrt{m}$

若原图中  $\text{deg}_{v_i} > \sqrt{m}$  由于与当前图中  $v_i$  相邻的点一定满足  $\text{deg}_w \geq \text{deg}_{v_i} > \sqrt{m}$  于是这样的  $w$  不超过  $\sqrt{m}$  个，即也有  $\text{out}_{v_i} \leq \sqrt{m}$

综上，总时间复杂度为  $O(m\sqrt{m})$

```
const int MAXN=1e5+5,MAXM=2e5+5;
struct Edge{
    int to,next;
}edge[MAXM];
int head[MAXN],edge_cnt;
void Insert(int u,int v){
    edge[++edge_cnt]=Edge{v,head[u]};
    head[u]=edge_cnt;
}
int deg[MAXN],vis[MAXN];
pair<int,int> E[MAXM];
int main()
{
    int n=read_int(),m=read_int();
    _for(i,0,m){
```

```
int u=read_int(),v=read_int();
deg[u]++,deg[v]++;
E[i]=make_pair(u,v);
}
_for(i,0,m){
int u=E[i].first,v=E[i].second;
if(deg[u]<deg[v]||(deg[u]==deg[v]&&u<v))
Insert(u,v);
else
Insert(v,u);
}
int ans=0;
_rep(u,1,n){
for(int i=head[u];i;i=edge[i].next)
vis[edge[i].to]=u;
for(int i=head[u];i;i=edge[i].next){
int v=edge[i].to;
for(int j=head[v];j;j=edge[j].next){
int w=edge[j].to;
ans+=(vis[w]==u);
}
}
}
enter(ans);
return 0;
}
```

## 例题二

[HDU6184](#)

### 题意

给定  $n$  个点和  $m$  条边的无向图。

定义好的四元组  $(A,B,C,D)$  为满足存在连边  $AB,BC,CD,DA,AC$  的四元组，求图中好的四元组数。

### 题解

统计每条边出现在三元组中的次数  $c_i$  最终答案为  $\sum_{i=1}^n \binom{c_i}{2}$

```
const int MAXN=1e5+5,MAXM=2e5+5;
struct Edge{
int to,next,c;
}edge[MAXM];
int head[MAXN],edge_cnt;
```

```

void Insert(int u,int v){
    edge[++edge_cnt]=Edge{v,head[u],0};
    head[u]=edge_cnt;
}
int deg[MAXN],vis[MAXN],in[MAXN];
pair<int,int> E[MAXM];
int main()
{
    int n,m;
    while(cin>>n>>m){
        _rep(i,1,n)head[i]=deg[i]=vis[i]=in[i]=0;
        edge_cnt=0;
        _for(i,0,m){
            int u=read_int(),v=read_int();
            deg[u]++,deg[v]++;
            E[i]=make_pair(u,v);
        }
        _for(i,0,m){
            int u=E[i].first,v=E[i].second;
            if(deg[u]<deg[v]||(deg[u]==deg[v]&&u<v))
                Insert(u,v);
            else
                Insert(v,u);
        }
        _rep(u,1,n){
            for(int i=head[u];i;i=edge[i].next){
                int w=edge[i].to;
                vis[w]=u,in[w]=i;
            }
            for(int i=head[u];i;i=edge[i].next){
                int v=edge[i].to;
                for(int j=head[v];j;j=edge[j].next){
                    int w=edge[j].to;
                    if(vis[w]==u)
                        edge[i].c++,edge[j].c++,edge[in[w]].c++;
                }
            }
        }
        LL ans=0;
        _rep(i,1,m)
            ans+=1LL*edge[i].c*(edge[i].c-1)/2;
        enter(ans);
    }
    return 0;
}

```

From:  
<https://wiki.cvbbacm.com/> - **CVBB ACM Team**

Permanent link:  
[https://wiki.cvbbacm.com/doku.php?id=2020-2021:teams:legal\\_string:jxm2001:%E4%B8%89%E5%85%83%E7%8E%AF%E8%AE%A1%E6%95%B0&rev=1622009755](https://wiki.cvbbacm.com/doku.php?id=2020-2021:teams:legal_string:jxm2001:%E4%B8%89%E5%85%83%E7%8E%AF%E8%AE%A1%E6%95%B0&rev=1622009755)

Last update: 2021/05/26 14:15

