BZOJ.PA2009 Cakes(三元环)

思路：三元环的裸题。第一次写还是稍微整理一下求三元环的思路。

1.无向图(无重边，无自环）转化为有向无环图。

转化的方法：对于任意边$edge(u,v)$,度数大的结点指向度数小的结点，若度数相同，则编号小的结点指向编号大的结点。

用表达式即：$if(deg[u]<deg[v]||deg[u]==deg[v]\&\&u>v)\quad swap(u,v)$

$e[u].push\_back(v)$.变为有向边。

2.接下来就是遍历每个结点看能否形成三元环。

对于结点i的所有相邻结点u打上标记，再对每个u遍历它的相邻结点v，若v有i给的标记，说明是一个三元环。

说明：这里每个三元环只会被记录一次。

时间复杂度：$O(m\sqrt{m})$

简单证明下时间复杂度：

显然我们考虑一条边$u\to v$对时间的贡献是out_v，

则总时间复杂度为：$\sum_{i=1}^{m}=out_{v_i}$

out_v是结点v的出度。

显然分两种情况。

当$deg(v)\le\sqrt{m}$时，因为$out_v\le deg(v)$，所以时间复杂度是$out_v=\sqrt{m}$。

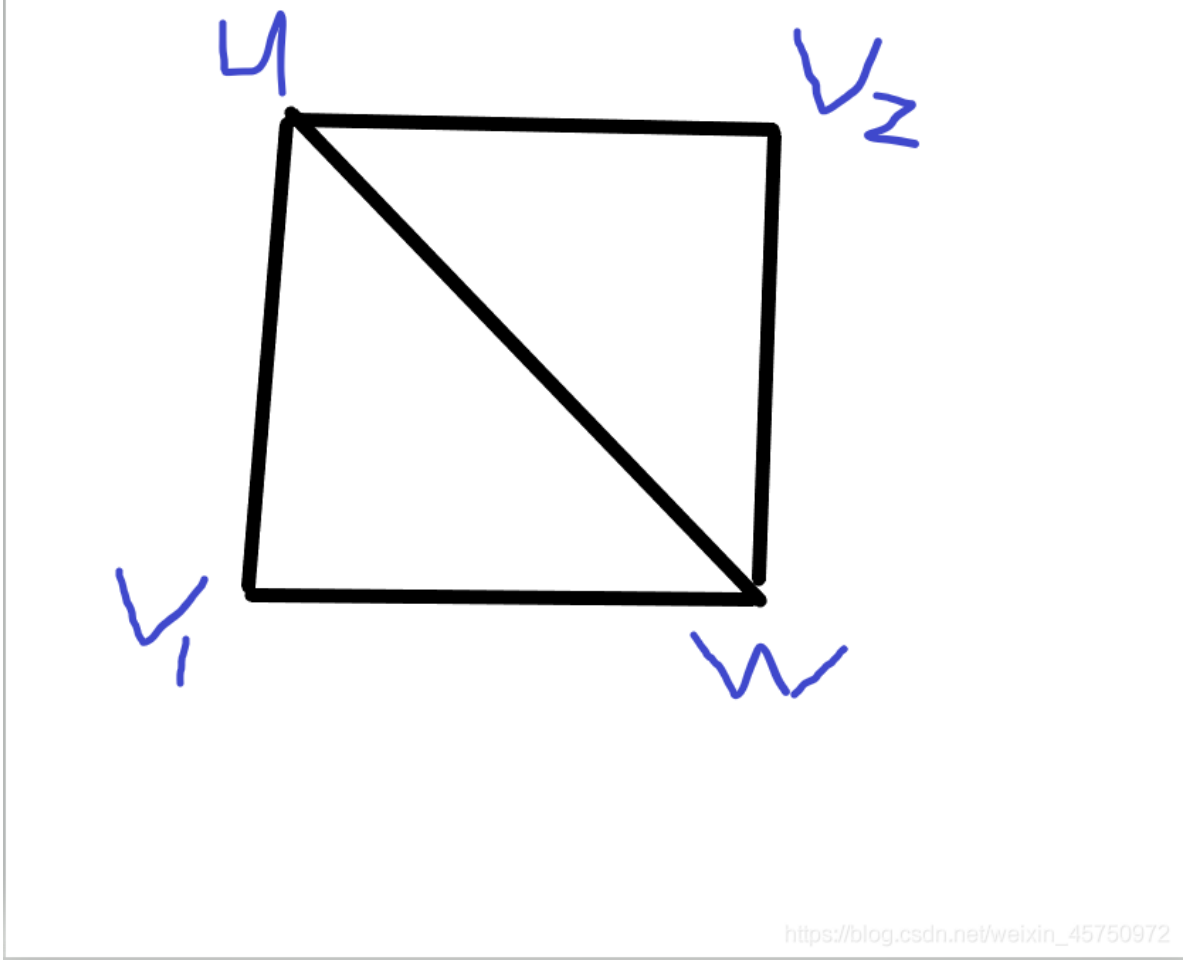
当$dge(v)>\sqrt{(m)}$时，因为图的总度数是$O(m)$的，所以度数$>\sqrt{m}$的结点至多有$O(\sqrt{m})$个，而v只能连向度数不小于它的结点，所以时间复杂度：$out_v=\sqrt{m}$.

综上：总时间复杂度：$\sum_{i=1}^{m}=out_{v_i}=m\sqrt{m}$.

```
#include<bits/stdc++.h>
using namespace std;
typedef long long ll;
const int N=1e5+5,M=1e6+5,inf=0x3f3f3f3f,mod=1e9+7;
#define mst(a) memset(a,0,sizeof a)
#define lx x<<1
#define rx x<<1|1
#define reg register
#define PII pair<int,int>
#define fi first
#define se second
int n,m,deg[N],vis[N],val[N];
vector<int>e[N];
```

```
struct node{
    int u,v;
}a[N];
int main(){
    scanf("%d%d",&n,&m);
    for(int i=1;i<=n;i++) scanf("%d",&val[i]);
    for(int i=1;i<=m;i++){
        int u,v;
        scanf("%d%d",&u,&v);
        deg[u]++,deg[v]++;
        a[i]={u,v};
    }
    for(int i=1;i<=m;i++){
        int u=a[i].u,v=a[i].v;
        if(deg[u]>deg[v]||deg[u]==deg[v]&&u>v) swap(u,v);   //无向图转有向无环图.
        e[u].push_back(v);
    }
    ll ans=0;
    for(int i=1;i<=n;i++){
        for(auto v:e[i]) vis[v]=i;
        for(auto u:e[i])
            for(auto v:e[u])
                if(vis[v]==i) ans+=max({val[u],val[v],val[i]});
    }
    printf("%lld\n",ans);
    return 0;
}
```

hdu.6184 Counting Stars(三元环)



思路：显然这样结构的图像可以转化为三元环来做。

我们用一个数组vis来维护每条边是否在三元环里。

我们可以通过枚举每条边是三元环的边的次数来计算出该边对答案的贡献。

$contribution = \frac{cnt[i] \times (cnt[i]-1)}{2}$。

时间复杂度：$O(m\sqrt{m})$。

m取$1e5$的话，时间复杂度大约在$3e7$左右。

```
#include<bits/stdc++.h>
using namespace std;
typedef long long ll;
const int N=1e5+5,M=1e6+5,inf=0x3f3f3f3f,mod=1e9+7;
#define mst(a) memset(a,0,sizeof a)
#define lx x<<1
#define rx x<<1|1
#define reg register
#define PII pair<int,int>
#define fi first
#define se second
int n,m,deg[N],cnt[N<<1],vis[N];
PII a[N<<1];
vector<PII>e[N];
int main(){
    while(~scanf("%d%d",&n,&m)){
```

```
    for(int i=1;i<=m;i++){
        scanf("%d%d",&a[i].fi,&a[i].se);
        deg[a[i].fi]++,deg[a[i].se]++;
    }
    for(int i=1;i<=m;i++){
        int u=a[i].fi,v=a[i].se;
        if(deg[u]>deg[v]||deg[u]==deg[v]&&u>v) swap(u,v);
        e[u].push_back({v,i});
    }
    for(int u=1;u<=n;u++){
        for(auto v:e[u]) vis[v.fi]=v.se;
        for(auto v:e[u])
            for(auto w:e[v.fi])
                if(vis[w.fi]) cnt[v.se]++,cnt[w.se]++,cnt[vis[w.fi]]++;
        for(auto v:e[u]) vis[v.fi]=0;
    }
    ll ans=0;
    for(int i=1;i<=m;i++){
        ans+=1LL*cnt[i]*(cnt[i]-1)/2;
        cnt[i]=0;
    }
    printf("%lld\n",ans);
    for(int i=1;i<=n;i++) e[i].clear(),deg[i]=0;
    }
    return 0;
}
```