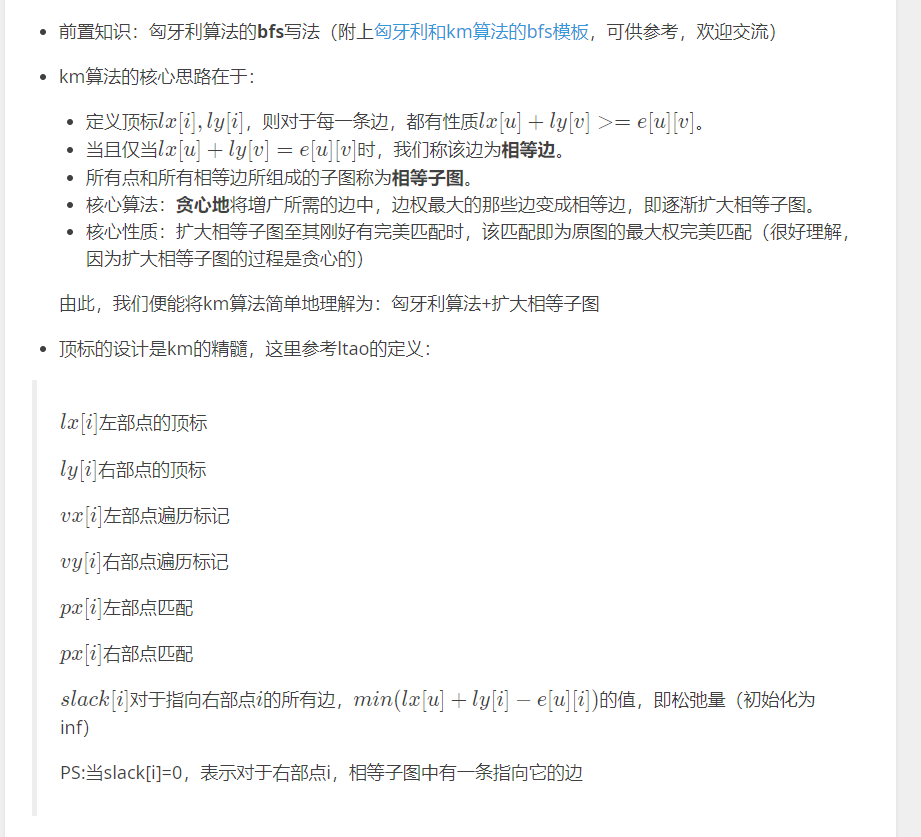
luogu6577：给定一张二分图，左右部均有 n个点，共有 m条带权边，且保证有完美匹配。

求一种完美匹配的方案，使得最终匹配边的边权之和最大。（完美匹配的意思就是最后左边和右边的所有点都要被匹配）

https://www.luogu.com.cn/blog/SingerCoder/solution-p6577

km算法（Kuhn－Munkres算法），是一种在二分图上求解最大权完美匹配的算法，用邻接矩阵存图即可。

相比费用流较高的复杂度，km算法有着更为优秀的效率，但局限性在于只能做匹配，而不像费用流那样灵活可变。





首先提示坑点：

边权最小约-1e7，所以如果要添加虚边将图补成完全图，记得初始边权设为-inf。（不补全而直接忽视不存在边在本题也可行，依照个人习惯即可）

权值和需要用long long​存储。





km+bfs

#include<bits/stdc++.h>

#define ll long long

#define inf 1e18

#define mn 505

using namespace std;

ll vx[mn],vy[mn],lx[mn],pre[mn],px[mn],py[mn],

ly[mn],a[mn][mn],slack[mn],

ans,cnt=0,n,m;

void aug(ll x)

{

if(px[pre[x]])aug(px[pre[x]]);

px[py[x]=pre[x]]=x;

}

void bfs(ll s)

{ll i,x,y,d;

queue<ll>q;

q.push(s);

vx[s]=1;

while(1)

{

while(q.empty()==0)

{

x=q.front();q.pop();

for(y=1;y<=n;y++)

if(!vy[y])

{

if(slack[y]>lx[x]+ly[y]-a[x][y])

{

slack[y]=lx[x]+ly[y]-a[x][y];

pre[y]=x;

}

if(slack[y]==0)

{

vy[y]=1;

if(!py[y]){aug(y);return;}

else{q.push(py[y]);vx[py[y]]=1;}

}

}

}

d=inf;

for(i=1;i<=n;i++)if(vy[i]==0)d=min(d,slack[i]);

for(i=1;i<=n;i++)

{

if(vx[i])lx[i]-=d;

if(vy[i])ly[i]+=d;

else slack[i]-=d;

}

for(y=1;y<=n;y++)if(vy[y]==0)

{

if(slack[y]==0)

{

vy[y]=1;

if(py[y]==0){aug(y);return ;}

else {q.push(py[y]);vx[py[y]]=1;}

}

}

}

}

void km()

{ll i,j;

for(i=1;i<=n;i++)

{

memset(pre,0,sizeof(pre));

memset(vx,0,sizeof(vx));

memset(vy,0,sizeof(vy));

for(j=1;j<=n;j++)slack[j]=inf;

bfs(i);

}

ans=0;

for(i=1;i<=n;i++)ans+=lx[i]+ly[i];

}

int main()

{

ll x,y,k,i,j,z;

char ch;

cin>>n>>m;

for(i=1;i<=n;i++)lx[i]=-inf;

for(i=1;i<=n;i++)

for(j=1;j<=n;j++)

a[i][j]=-inf;

while(m--)

{

scanf("%lld%lld%lld",&x,&y,&z);

a[x][y]=z;

lx[x]=max(lx[x],z);

}

km();

cout<<ans<<endl;

for(i=1;i<=n;i++)

printf("%lld ",py[i]);

return 0;

}

km+dfs（慢一点，可以尽量不用）

#include<bits/stdc++.h>

#define ll long long

#define inf 1e18

#define mn 505

using namespace std;

ll vx[mn],vy[mn],lx[mn],

ly[mn],a[mn][mn],slack[mn],

my[mn],ans=0,cnt=0,tot=0,sign=0,n,m;

ll dfs(ll i)

{ll j;

vx[i]=1;

for(j=1;j<=n;j++)

if(vy[j]==0)

{

ll t=lx[i]+ly[j]-a[i][j];

if(t==0)

{

vy[j]=1;

if(my[j]==0||dfs(my[j])){my[j]=i;return 1;}

}

else slack[j]=min(slack[j],t);

}

return 0;

}

void km()

{ll i,j,tmp,d;

ans=0;

for(i=1;i<=n;i++)

{

lx[i]=-inf;

for(j=1;j<=n;j++)

lx[i]=max(lx[i],a[i][j]);

}

memset(ly,0,sizeof(ly));

memset(my,0,sizeof(my));

for(i=1;i<=n;i++)

{

for(j=1;j<=n;j++)

slack[j]=inf;

while(1)

{

memset(vx,0,sizeof(vx));

memset(vy,0,sizeof(vy));

if(dfs(i))break;

d=inf;

for(j=1;j<=n;j++)

if(vy[j]==0)d=min(d,slack[j]);

for(j=1;j<=n;j++)if(vx[j])lx[j]-=d;

for(j=1;j<=n;j++)if(vy[j])ly[j]+=d;

else slack[j]-=d;

}

}

for(j=1;j<=n;j++)

if(my[j]>0)ans+=a[my[j]][j];

}

int main()

{

ll x,y,k,i,j,z;

char ch;

cin>>n>>m;

for(i=1;i<=n;i++)

for(j=1;j<=n;j++)

a[i][j]=-inf;

while(m--)

{

scanf("%lld%lld%lld",&x,&y,&z);

a[x][y]=z;

}

km();

cout<<ans<<endl;

for(i=1;i<=n;i++)

printf("%lld ",my[i]);

return 0;

}