模板

[图论 3](#_Toc241320427)

[二分图最大匹配匹配匈牙利算法 O(n^3) 3](#_Toc1829082541)

[带权二分图最佳匹配匹配KM算法 O(n^3) 3](#_Toc447125069)

图论

二分图最大匹配匹配匈牙利算法 O(n^3)

使用时对每个x顶点dfs(x)返回1匹配数加一

int k,nx,ny;//边数 两边顶点数

int cnt;

int vis[MAXN],py[MAXN],head[MAXN];//vis为当前轮y点是否被访问

//py为总的y已经匹配到的x编号

struct edge{

int v,next;

}p[MAXN\*MAXN];//边总数

void addedge(int u,int v){

p[cnt].v=v;

p[cnt].next=head[u];

head[u]=cnt++;

}

bool dfs(int x){

for(int i=head[x];~i;i=p[i].next){ //遍历每个x点的边

int y=p[i].v;

if(!vis[y]){ //如果当前轮这个y还没匹配

vis[y]=1;

if(py[y]==0||dfs(py[y])){ //如果y在前面的轮次被匹配寻找是否能给这个y匹配的x寻找新的y

py[y]=x; //这个y匹配到本次搜索的x

return 1;

}

}

}

return 0;

}

带权二分图最佳匹配匹配KM算法 O(n^3)

int k,nx,ny;//边数 两边顶点数

int eg[MAXN][MAXN];

int minn;

int visy[MAXN],visx[MAXN],py[MAXN],slack[MAXN],head[MAXN];

int wx[MAXN],wy[MAXN];

bool dfs(int x){

visx[x]=1;

for(int i=1;i<=ny;i++){

if(!visy[i]){

int t=wx[x]+wy[i]-eg[x][i];

if(t==0){

visy[i]=1;

if(py[i]==0||dfs(py[i])){

py[i]=x;

return 1;

}

}

else if(slack[i]>t){

slack[i]=t;

}

}

}

return 0;

}

int km(){

memset(py,0,sizeof(py));

memset(wx,0,sizeof(wx));

memset(wy,0,sizeof(wy));

for(int i=1;i<=nx;i++){

for(int j=1;j<=ny;j++){

if(eg[i][j]>wx[i])

wx[i]=eg[i][j];

}

}

for(int i=1;i<=nx;i++){

for(int j=1;j<=ny;j++) slack[j]=INF;

while(true){

minn=INF;

memset(visx,0,sizeof(visx));

memset(visy,0,sizeof(visy));

if(dfs(i)) break;

for(int j=1;j<=ny;j++){

if(!visy[j]&&minn>slack[j])

minn=slack[j];

}

for(int j=1;j<=nx;j++){

if(visx[j]) wx[j]-=minn;

}

for(int j=1;j<=ny;j++){

if(visy[j]) wy[j]+=minn;

else slack[j]-=minn;

}

}

}

int ans=0;

for(int i=1;i<=ny;i++){

ans+=eg[py[i]][i];

}

return ans;

}