A题题解：最小割

题意：求n个节点m条边的无向连通图的最小割

算法：Stoer Wagner算法 ~~（网络流会爆）~~

整个算法基于这样一个事实：最小割会将整个图分成两个连通块。对于图中任意两个节点s,t，他俩要么在同一个连通块，要么分别在两个连通块上。

算法步骤：

1. 首先固定一个节点，每次循环从这个节点开始用一种类似prim的思想（从点开始扩展）扩展出一颗最大生成树。

   1）找最大边：

   ```
   max=0;
   max_v=0;
   for(int k=1;k<=n;++k){
    if(!vis[k]&&!combine[k]){//某个节点既没有被访问也没有被合并
     if(dist[k]>max||!max_v){//满足条件，更新
      max=dist[k];
      max_v=k;
     }
    }
   }
   t=max_v;
   ```

   2）更新边权：

   ```
   //dist[]存的是各个节点到已经拓展的最大生成树的点集的距离
    for(int k=1;k<=n;++k){
     if(!combine[k]&&!vis[k])
      dist[k]+=map[t][k];
    }
   ```

2. 记录扩展到最后剩下的两个点s,t还有最后剩的的边的边权。

3. 如果这个边权比用来原先计算的最小割小，那么就更新最小割。

4. 把最后两个点s,t合并。（note：由于上文中提到的那个事实，假如这两条边不在同一个连通块，那么他们俩之间的最小割就是我们所求的整个图的最小割；假如他俩在同一个连通块，那么合并他俩也没啥影响。最坏情况合并n-1次点后才能找到不在同一个连通块上的两个点。所以循环n-1次。）

   ```
   combine[s]=1;//s已经合并了
    for(int j=1;j<=n;++j){
      map[t][j]+=map[s][j];
      map[j][t]+=map[j][s];
    }//所有s相关联的边都加到t上
   ```

5. 从步骤2开始循环n-1次。

时间复杂度O（n^3):

循环n-1次，O（n）时间复杂度，每次要拓展个最大生成树，拓展每个节点要查遍所有出边，O（n）时间复杂度,要拓展n-1次，O（n）时间复杂度。乘起来就是O（n^3)时间复杂度。