# 斯坦纳树

1. 什么是斯坦纳树？

       斯坦纳树问题是组合优化学科中的一个问题。将指定点集合中的所有点连通，且边权总和最小的生成树称为最小斯坦纳树（Minimal Steiner Tree），其实最小生成树是最小斯坦纳树的一种特殊情况。而斯坦纳树可以理解为使得指定集合中的点连通的树，但不一定最小。

2. 如何求解最小斯坦纳树？

      可以用DP求解，dp[i][state]表示以i为根，指定集合中的点的连通状态为state的生成树的最小总权值。

      转移方程有两重：

      第一重，先通过连通状态的子集进行转移。

      dp[i][state]=min{ dp[i][subset1]+dp[i][subset2] }

      枚举子集的技巧可以用 for(sub=(state-1)&state;sub;sub=(sub-1)&state)。

      第二重，在当前枚举的连通状态下，对该连通状态进行松弛操作。

      dp[i][state]=min{ dp[i][state], dp[j][state]+e[i][j] }

      为什么只需对该连通状态进行松弛？因为更后面的连通状态会由先前的连通状态通过第一重转移得到，所以无需对别的连通状态松弛。松弛操作用SPFA即可。

      复杂度 O(n\*3^k+cE\*2^k)

      c为SPFA复杂度中的常数，E为边的数量，但几乎达不到全部边的数量，甚至非常小。3^k来自于子集的转移sum{C(i,n)\*2^i} (1<=i<=n)，用二项式展开求一下和。

/\*

\* Steiner Tree：求，使得指定K个点连通的生成树的最小总权值

\* st[i] 表示顶点i的标记值，如果i是指定集合内第m(0<=m<K)个点，则st[i]=1<<m

\* endSt=1<<K

\* dptree[i][state] 表示以i为根，连通状态为state的生成树值

\*/

#define CLR(x,a) memset(x,a,sizeof(x))

int dptree[N][1<<K],st[N],endSt;

bool vis[N][1<<K];

queue<int> que;

int input(){

/\*

\* 输入，并且返回指定集合元素个数K

\* 因为有时候元素个数需要通过输入数据处理出来，所以单独开个输入函数。

\*/

}

void initSteinerTree(){

CLR(dptree,-1);

CLR(st,0);

for(int i=1;i<=n;i++) CLR(vis[i],0);

endSt=1<<input();

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

dptree[i][st[i]]=0;

}

void update(int &a,int x){

a=(a>x || a==-1)? x : a;

}

void SPFA(int state){

while(!que.empty()){

int u=que.front();

que.pop();

vis[u][state]=false;

for(int i=p[u];i!=-1;i=e[i].next){

int v=e[i].vid;

if(dptree[v][st[v]|state]==-1 ||

dptree[v][st[v]|state]>dptree[u][state]+e[i].w){

dptree[v][st[v]|state]=dptree[u][state]+e[i].w;

if(st[v]|state!=state || vis[v][state])

continue; //只更新当前连通状态

vis[v][state]=true;

que.push(v);

}

}

}

}

void steinerTree(){

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

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

if(st[i] && (st[i]&j)==0) continue;

for(int sub=(j-1)&j;sub;sub=(sub-1)&j){

int x=st[i]|sub,y=st[i]|(j-sub);

if(dptree[i][x]!=-1 && dptree[i][y]!=-1)

update(dptree[i][j],dptree[i][x]+dptree[i][y]);

}

if(dptree[i][j]!=-1)

que.push(i),vis[i][j]=true;

}

SPFA(j);

}

}

## BZOJ2595



