目录

1 新的板子

1.1 MstPrim

```
class MstPrim{
public:
    struct node{
        int id;int dis;
        bool operator<(const node&a)const{
            return a.dis<dis;
        }
    };
    int maxD[N][N],mp[N][N];
    bool mst[N][N];
    int dis[N];
    bool vis[N];
    int pre[N];
    int n;
    MstPrim(){}
    void init(int _n){
        n=_n;
        for(int i=1;i<=n;i++){
            for(int j=1;j<=n;j++)
                mp[i][j]=INF,mst[i][j]=maxD[i][j]=0;
            vis[i]=0;
            dis[i]=INF;
            pre[i]=1;
        }
    }
    int prim(int s=1)
    {
        priority_queue<node>q;
        q.push({s,dis[s]=0});
        int ans=0;
        while(!q.empty())
        {
            auto [x,W]=q.top();q.pop();
```

```
34              if(vis[x])continue;
35              vis[x]=1;
36              ans+=W;
37              mst[x][pre[x]]=mst[pre[x]][x]=1;
38              for(int y=1;y<=n;y++){
39                  if(vis[y]&&mp[x][y]<INF)
40                      maxD[x][y]=maxD[y][x]=max(maxD[pre[x
                            ]][y],dis[x]);
41                  if(mp[x][y]<dis[y]){
42                      dis[y]=mp[x][y];
43                      pre[y]=x;
44                      q.push({y,dis[y]});
45                  }
46              }
47          }
48          return ans;
49      }
50      void add(int x,int y,int w){mp[x][y]=w;}
51      bool had(int x,int y){return mst[x][y];}
52      int getmaxD(int x,int y){return maxD[x][y];}
53  }_mst;
```

1.2 MstKru

```
class MstKru{
public:
    struct node{int x,y,w,fl;}p[maxn];
    bool mst[maxn][maxn];
    int maxD[maxn][maxn];
    vector<int>v[maxn];
    int n,m;
    MstKru(){}
    void init(int _n){
        n=_n;m=0;
        for(int i=1;i<=n;i++){
            for(int j=1;j<=n;j++){
                mst[i][j]=0;
                maxD[i][j]=0;
            }
            v[i]={i};
        }
    }
    void add(int x,int y,int w){
        p[++m]={x,y,w};
    }
    int run()
    {
        sort(p+1,p+1+m,[&](node a,node b){return a.w<b.w
           ;});
        dsu.init(n);
        ll ans=0;
        int cnt=n;
        for(int i=1;i<=m;i++)
        {
            auto &[x,y,w,fl]=p[i];
            if(!dsu.same(x,y))
            {
                cnt--;
                int fa=dsu.find(x);
```

```
35                  int fb=dsu.find(y);
36                  ans+=w; fl=1;
37                  dsu.merge(x,y);
38                  for(auto _i:v[fa])
39                      for(auto _j:v[fb])
40                          maxD[_i][_j]=maxD[_j][_i]=w;
41                  for(auto _i:v[fb])
42                      v[fa].push_back(_i);
43              }
44              if(cnt==1)break;
45          }
46          if(cnt!=1)return -2;
47          ll res=INF;
48          for(int i=1;i<=m;i++)
49          {
50              if(p[i].fl)continue;
51              auto [x,y,w,fl]=p[i];
52              res=min(res,ans+w-maxD[x][y]);
53          }
54          if(res==INF)return -1;
55          return res;
56      }
57  }mst;
```

1.3 STR_Tr

```
class STR_Tr{
public:
    struct Edge{int to,next,w;}edge[maxn];
    int head[maxn],cnt;
    int f[maxn][22];
    int dep[maxn];
    int maxx[maxn][22];
    int minn[maxn][22];
    void add(int from,int to,int w){
        edge[++cnt].w=w;
        edge[cnt].to=to;
        edge[cnt].next=head[from];
        head[from]=cnt;
    }
    void Add(int x,int y,int w){
        add(x,y,w);
        add(y,x,w);
    }
    void dfs(int x,int fa)
    {
        dep[x]=dep[fa]+1;
        f[x][0]=fa;
        minn[x][0]=-INF;
        for(int i=1;(1<<i)<=dep[x];i++)
        {
            f[x][i]=f[f[x][i-1]][i-1];
            int kk[4]={maxx[x][i-1],maxx[f[x][i-1]][i-1],
                    minn[x][i-1],minn[f[x][i-1]][i-1]};
            sort(kk,kk+4);
            maxx[x][i]=kk[3];
            int ptr=2;
            while(ptr>=0&&kk[ptr]==kk[3])ptr--;
            minn[x][i]=(ptr==-1?-INF:kk[ptr]);
        }
        for(int i=head[x];i;i=edge[i].next)
```

```
36              if(edge[i].to!=fa){
37                  maxx[edge[i].to][0]=edge[i].w;
38                  dfs(edge[i].to,x);
39              }
40      }
41      int lca(int x,int y){
42          if(dep[x]<dep[y])swap(x,y);
43          for (int j = 0, D = dep[x] - dep[y]; D; ++j, D >>=
                1)
44              if (D & 1) x = f[x][j];
45          if(x==y)return x;
46          for (int j = 20; ~j; --j)
47              if (f[x][j] != f[y][j])
48                  x = f[x][j], y = f[y][j];
49          return f[x][0];
50      }
51      int querySTR(int x,int y,int val){
52          int res=-INF;
53          for(int i=21;i>=0;i--){
54              if(dep[f[x][i]]>=dep[y]){
55                  if(val!=maxx[x][i])
56                      res=max(res,maxx[x][i]);
57                  else
58                      res=max(res,minn[x][i]);
59                  x=f[x][i];
60              }
61          }
62          return res;
63      }
64      int query(int x,int y)
65      {
66          int res=-INF;
67          for(int i=21;i>=0;i--){
68              if(dep[f[x][i]]>=dep[y]){
69                  res=max(res,maxx[x][i]);
70                  x=f[x][i];
71              }
```

```
72          }
73          return res;
74      }
75  }tr;
```

1.4 Tr

```
class Tr{
public:
    struct Edge{int to,next,w;}edge[maxn];
    int head[maxn],cnt;
    int maxx[maxn][22];
    int f[maxn][22];
    int dep[maxn];
    void add(int from,int to,int w){
        edge[++cnt].w=w;
        edge[cnt].to=to;
        edge[cnt].next=head[from];
        head[from]=cnt;
    }
    void Add(int x,int y,int w){
        add(x,y,w);
        add(y,x,w);
    }
    void dfs(int x,int fa)
    {
        dep[x]=dep[fa]+1;
        f[x][0]=fa;
        for(int i=1;(1<<i)<=dep[x];i++)
        {
            f[x][i]=f[f[x][i-1]][i-1];
            maxx[x][i]=max(maxx[x][i-1],maxx[f[x][i-1]][i
               -1]);
        }
        for(int i=head[x];i;i=edge[i].next)
            if(edge[i].to!=fa){
                maxx[edge[i].to][0]=edge[i].w;
                dfs(edge[i].to,x);
            }
    }
    int lca(int x,int y){
        if(dep[x]<dep[y])swap(x,y);
```

```
35          for (int j = 0, D = dep[x] - dep[y]; D; ++j, D >>=
                1)
36              if (D & 1) x = f[x][j];
37          if(x==y)return x;
38          for (int j = 20; ~j; --j)
39              if (f[x][j] != f[y][j])
40                  x = f[x][j], y = f[y][j];
41          return f[x][0];
42      }
43      int query(int x,int y)
44      {
45          int res=-INF;
46          for(int i=21;i>=0;i--){
47              if(dep[f[x][i]]>=dep[y]){
48                  res=max(res,maxx[x][i]);
49                  x=f[x][i];
50              }
51          }
52          return res;
53      }
54  }_tr;
```

1.5 Kru

```
class Kru{
public:
    struct node{int x,y,w,fl;};
    vector<node>p;
    int n,m;
    ll ans;
    Kru(){}
    void init(int _n){
        n=_n;ans=0;m=0;
        p.clear();
        p.resize(m+1);
        dsu.init(n);
    }
    void add(int x,int y,int w){
        // p[++m]={x,y,w};
        p.push_back({x,y,w});
    }
    int run(){
        sort(p.begin()+1,p.end(),[&](node a,node b){return
            a.w<b.w;});
        int num=n;
        for(int i=1;i<=m;i++)
        {
            auto &[x,y,w,fl]=p[i];
            if(!dsu.same(x,y)){
                num--;
                dsu.merge(x,y);
                tr.Add(x,y,w);
                ans+=w;
                fl=1;
            }
            if(num==1)break;
        }
        if(num!=1)ans=-1;
        return ans;
```

```
35      }
36      ll query()
37      {
38          tr.dfs(1,0);
39          ll res=INF64;
40          for(int i=1;i<=m;i++)
41          {
42              if(p[i].fl)continue;
43              auto [x,y,w,fl]=p[i];
44              int _lca=tr.lca(x,y);
45              int tmpa=tr.query(x,_lca);
46              int tmpb=tr.query(y,_lca);
47              if(max(tmpa,tmpb)>-INF)
48                  res=min(res,ans-max(tmpa,tmpb)+w);
49          }
50          return (res==INF64?-1:res);
51      }
52  }_k;
```