宝藏

一. 考察内容：

动态规划 状态压缩

二. 题目分析：

[题目大意]

在一个无向图上构建最小生成树，每条边的花费为边权乘该边距根的节点数。

[写题思路]

如果用kruskal跑最小生成树，由于边权与根节点位置有关，所以结果不一定是最优的。考虑用状态压缩动态规划实现。

设状态f[i][j]：最大深度为i，选取的点∈j的最小化费。转移：f[i][j]=f[i-1][k]+（在第i层添加j-k这些节点所需要的最小花费），其中 新加节点的花费 需要枚举每种情况来求。对于部分状态，有的转移是错误的，因为我们没有记录i-1层都包含哪些节点，但是正解一定被包含到状态中，所以该转移的正确性不会受到影响。

三. 代码实现：

```
#define _CRT_SECURE_NO_DEPRECATE
/************************
*创建时间：2018 08 11
*文件类型：源代码文件
*题目来源：COGS
*当前状态：已通过
*备忘录：动态规划 状态压缩
*作者：HtBest
************************/
#include <stdio.h>
#include <iostream>
#include <string>
#include <stdlib.h>
#include <string.h>
#include <algorithm>
#include <queue>
using namespace std;
#define MAXN 13
#define lowbit(x) (x&-x)
int n,m,d[MAXN][MAXN],U;
long long f[MAXN][1<<MAXN],ans;
/* Variable explain:

*/
void read()
{
    int ls1,ls2,ls3;
    // freopen("2017treasure.in","r",stdin);
    // freopen("2017treasure.out","w",stdout);
    scanf("%d%d",&n,&m);
    for(int i=1;i<=n;++i)for(int j=1;j<=n;++j)d[i][j]=1e9;
    for(int i=1;i<=m;++i)scanf("%d%d%d",&ls1,&ls2,&ls3),d[ls1][ls2]=d[ls2]
[ls1]=min(ls3,d[ls1][ls2]);
    return;
}
bool check(int n,int a)
{
    if(n==1&&a!=lowbit(a))return false;
```

```
    int ans=0;
    while(a)ans++,a-=lowbit(a);
    return ans>=n;
}
void dp()
{
    //设状态f[i][j]:最深为i层，用到j这些点
    U=(1<<n)-1;
    for(int i=1;i<=n;++i)for(int j=0;j<=U;++j)f[i][j]=1e9;
    for(int i=0;i<n;++i)f[1][1<<i]=0;
    for(int i=2;i<=n;++i)
    {
        for(int j=3;j<=U;++j)
        {
            if(!check(i,j)) continue;
            for(int k=(j-1)&j;k;k=(k-1)&j)
            {
                if(!check(i-1,k))   continue;
                long long ls1=j-k,cols1=0,mincost=0;
                while(ls1)//当前状态
                {
                    ++cols1;
                    if(ls1&1)
                    {
                        long long ls2=k,cols2=0,mincost2=1e9;
                        while(ls2)//转移状态
                        {
                            ++cols2;
                            if(ls2&1)
                            {
                                mincost2=min(mincost2,(long long)d[cols1][cols2]);
                            }
                            ls2>>=1;
                        }
                        mincost+=mincost2;
                        // printf(" d[%d]:%d\n",cols1,mincost2);
                    }
                    ls1>>=1;
                }
                // printf("k=%d: %d\n",k,f[i-1][k]+(mincost*(i-1)<0?
100000000:mincost*(i-1)));
                if(mincost<1000000000)f[i][j]=min(f[i][j],f[i-1][k]+mincost*(i-1));
            }
            // printf("#f[%d][%d]:%d#\n",i,j,f[i][j]);
        }
    }
}
int main()
{
    read();
    dp();
    ans=1e9;
    for(int i=1;i<=n;++i)ans=min(ans,f[i][U]);
    printf("%lld\n",ans);
    return 0;
}
```

<题目跳转> <查看代码>