提高组300分试题 第三组

赛艇表演

题解

30pts：n<=10，m<=20。

给不会最短路的人的部分分。

50pts：n<=100，m<=500。

往返路费其实就是把边权乘以2。

用floyd求出最短路，然后暴力枚举就可以了。

70pts：n<=1500，m<=2000。

考虑门票钱最小的那个城市，肯定是自己在自己城市看的。

我们从这个点出发跑Dijkstra，求出到其他城市的最短路，更新其他城市的答案。

然后删掉这个点，重复以上操作。

一共要跑n遍Dijkstra，但是点的数量越来越少，所以常数很小。

85pts：图的结构随机生成。

考虑DP？

设f[u]表示点u的最小花费。

转移是用f[u] + w更新f[v]，其中v和u相邻。

这个转移有环怎么办啊？

你发现这其实是个spfa。

对带环DP跑spfa就可以了。

100pts。

把图建出来，把spfa换成Dijkstra就可以了。

当然这个题可以直接建模最短路模型，省略前面所有的部分分。

在原图的基础上加一个超级源S，S到i的连边是在i城市看表演的费用。

那么S到i的最短路就是i的最小费用。

相似题目：NOIp14 寻找道路

标准代码

C++

```
#include <bits/stdc++.h>
using namespace std;
typedef long long ll;
const int N = 200010;
const int M = 1000010;
int _w;

namespace G {
        int head[N], nxt[M], to[M], eid;
        ll len[M];

        void init() {
                eid = 0;
                memset(head, -1, sizeof head);
        }
        void adde( int u, int v, ll w ) {
                to[eid] = v, len[eid] = w;
                nxt[eid] = head[u], head[u] = eid++;
                to[eid] = u, len[eid] = w;
                nxt[eid] = head[v], head[v] = eid++;
        }
}

int n, m;
ll dis[N];
bool done[N];

struct Node {
        int u;
        ll d;
        Node() {}
        Node( int u, ll d ):
                u(u), d(d) {}
        bool operator<( const Node &rhs ) const {
                return d > rhs.d;
        }
};
priority_queue<Node> pq;

void dijkstra() {
        using namespace G;
        memset(dis, 0x3f, sizeof dis);
        dis[0] = 0, pq.push( Node(0, 0) );
```

```
        while( !pq.empty() ) {
                Node o = pq.top(); pq.pop();
                int u = o.u;
                if( done[u] ) continue;
                done[u] = 1;
                for( int i = head[u]; ~i; i = nxt[i] ) {
                        int v = to[i];
                        ll w = len[i];
                        if( dis[u] + w < dis[v] ) {
                                dis[v] = dis[u] + w;
                                pq.push( Node(v, dis[v]) );
                        }
                }
        }
}

int main() {
        _w = scanf( "%d%d", &n, &m );
        G::init();
        for( int i = 0; i < m; ++i ) {
                int u, v;
                ll w;
                _w = scanf( "%d%d%lld", &u, &v, &w );
                G::adde(u, v, w*2);
        }
        for( int i = 1; i <= n; ++i ) {
                ll a;
                _w = scanf( "%lld", &a );
                G::adde(0, i, a);
        }
        dijkstra();
        for( int i = 1; i <= n; ++i )
                printf( "%lld ", dis[i] );
        puts("");
        return 0;
}
```

逮虾户

题解

假设已知 c，那么总时间 T 为 $\sum_{i=1}^{n} \frac{di}{si+c}$ 随着 c 的增大，T 只会越来越小；反之随着 c 的减小，T只会越来越大（存在单调性）。 二分查找 c，使得 T = t 即可。注意二分的上下界。 时间复杂度 O(n logw)。

标准代码

C++

```cpp
#include<bits/stdc++.h>
using namespace std;
const int maxn = 1005;
int n;
double t,d[maxn],s[maxn];
double check(double x){
    double res = 0;
    for(int i=0;i<n;i++){
        if(s[i]+x<=0){
            return 10000000.0;
        }
        res += d[i]/(s[i]+x);
    }
    return res;
}
int main(){
    cin>>n>>t;
    for(int i=0;i<n;i++)
        cin>>d[i]>>s[i];
    double l=-1e9,r=1e9;
    for(int cas=0;cas<=100;cas++){
        double mid = (l+r)/2.0;
        if(check(mid)>t)l=mid;
        else r=mid;
    }
    printf("%.12f\n",l);
}
```

战略威慑

题解

由题意可知，城市与道路组成了一棵树。我们从这棵树去掉一条边，从得到的两棵树上分别寻找最长路径即可不交叉。

由于n只有200，枚举去掉的边，再分别用dfs求树上最长路径即可。

标准代码

C++

```cpp
#include <cstdio>
#include <algorithm>
#include <cstring>
#include <vector>
using namespace std;

const int N = 2e3 + 100;
```

```cpp
int n;
vector<int> V[N];
int dep[N];

int dfs1(int pre, int fa, int root, int deep) {
        dep[pre] = deep;
        int ct = pre, res;
        for (int i = 0; i < V[pre].size(); i++) {
                int v = V[pre][i];
                if (v == fa || v == root) continue;
                res = dfs1(v, pre, root, deep + 1);
                if (dep[ct] < dep[res])
                        ct = res;
        }
        return ct;
}

int solve(int pre, int fa) {
        int ct = dfs1(pre, pre, fa, 0);
        ct = dfs1(ct, ct, fa, 0);
        return dep[ct];
}

int dfs(int pre, int fa) {
        int ans = 0;
        for (int i = 0; i < V[pre].size(); i++) {
                int v = V[pre][i];
                if (v == fa) continue;
                ans = max(ans, solve(v, pre) * solve(pre, v));
                ans = max(ans, dfs(v, pre));
        }
        return ans;
}

int main() {
        int a, b;
        scanf("%d", &n);
        for (int i = 1; i < n; i++) {
                scanf("%d%d", &a, &b);
                V[a].push_back(b);
                V[b].push_back(a);
        }
        printf("%d\n", dfs(1, 1));
        return 0;
}
```