

# J. Random Walk

## 题意

给定  $n$  个点  $m$  条边，每条边两个权值  $(a_i, b_i)$  在  $t$  时刻经过第  $i$  条边的收益为  $\max(\lfloor \frac{a_i}{2^t} \rfloor, b_i)$  且一条边经过多次收益也计算多次。

每个点一个点权  $w_i$  起点位于点  $s$  的概率为  $\frac{w_s}{\sum_{i=1}^{n-1} w_i}$  每个时间单位人物从当前点等概率移动到相邻点，且移动到点  $n$  后游戏结束。

问人物的期望收益，然后接下来两种修改操作，每次修改后再次询问人物收益：

1. 将某条边  $(u, v)$  的边权修改为  $(a, b)$  保证这条边原图中存在
2. 将某个点的  $w_i$  修改

数据保证  $1 \leq b_i \leq a_i \leq 100$  且第二类修改操作不超过  $n$  个。

## 题解

设  $\text{dp}(u, i)$  表示  $t=i$  时人物位于点  $u$  的概率  $E(u)$  表示人物经过点  $u$  的期望次数  $G(u)$  表示与  $u$  相邻的边  $p(u) = \frac{w_u}{\sum_{i=1}^{n-1} w_i}$

不难发现  $t \geq 6$  后每条边收益一定等于  $b_i$  于是答案等于

$$\sum_{u=1}^{n-1} \frac{1}{\deg(u)} \sum_{e \in G(u)} \sum_{t=0}^{+\infty} \max(\lfloor \frac{a_i}{2^t} \rfloor, b_i) \text{dp}(u, t) = \sum_{u=1}^{n-1} \frac{1}{\deg(u)} \sum_{e \in G(u)} \left( b_i \left( E(u) - \sum_{t=0}^5 \text{dp}(u, t) \right) + \sum_{t=0}^5 \max(\lfloor \frac{a_i}{2^t} \rfloor, b_i) \text{dp}(u, t) \right)$$

关于  $\text{dp}(u, t) (0 \leq t \leq 5)$  可以  $O(6m)$  暴力计算，接下来考虑计算  $E(u)$

考虑建立有向图，如果原图中边  $(u, v)$  满足  $u, v \neq n$  则连边  $u \to v (w = \frac{1}{\deg(u)})$ ,  $v \to u (w = \frac{1}{\deg(v)})$

否则，假设  $v = n$  则只连边  $u \to v (w = \frac{1}{\deg(u)})$  同时建立超级源点  $s$  连边  $s \to u (w = p(u))$

对每个点  $u$  考虑所有入边  $v_1 \to u, v_2 \to u \dots v_k \to u$  不难发现  $E(u) = \sum_{i=1}^k w_i E(v_i)$  初始条件  $E(s) = 1$

于是上述方程可以表示成如下矩阵：

$$\left[ \begin{array}{cccc|c} c & c & c & c & c \end{array} \right] \text{ %这里的c表示数组中元素对其方式 |c居中|r右对齐|l左对齐, 竖线表示2、3列间插入竖线 } 1 \& a_{1,2} \& \dots \& a_{1,n-1} \& p(1) \backslash 1 \& a_{2,2} \& \dots \& a_{2,n-1} \& p(2) \backslash \dots \& \dots \& \dots \& \dots \backslash a_{n-1,1} \& a_{n-1,2} \& \dots \& 1 \& p(n-1) \backslash \end{array} \right]$$

$O(n^3)$  即可计算出所有  $E(u)$  因此计算初始答案复杂度为  $O(n^3 + 6m)$

接下来考虑修改操作，发现第一种修改操作对  $\text{dp}(u, t), E(u)$  均无影响，如果  $u \neq n$  对答案影响为

$$\frac{1}{\deg(u)} \left( b_i \left( E(u) - \sum_{t=0}^5 \text{dp}(u,t) \right) + \sum_{t=0}^5 \max \left( \lfloor \frac{a_i}{2^t} \rfloor, b_i \right) \text{dp}(u,t) \right)$$

同理  $v \neq n$  时也产生类似影响，因此单次处理复杂度  $O(6)$

第二种操作对  $\text{dp}(u,t), E(u)$  均产生影响，同样  $\text{dp}(u,t)$  可以暴力计算，但  $E(u)$  重新计算成分过高。

注意到  $E(u)$  的增广矩阵仅有增广部分被修改，因此可以预处理出原矩阵的逆  $A^{-1}$  则有

$$\begin{pmatrix} E(1) \\ E(2) \\ \vdots \\ E(n-1) \end{pmatrix} = A^{-1} \begin{pmatrix} p(1) \\ p(2) \\ \vdots \\ p(n-1) \end{pmatrix}$$

因此可以  $O(n^2+6m)$  重新计算一次答案。总时间复杂度  $O(n^3+6nm+6q)$

```
const int MAXN=505,MAXM=1e4+5,MAXT=5,mod=998244353;
struct Edge{
    int to,next;
}edge[MAXM<<1];
int head[MAXN],edge_cnt;
void Insert(int u,int v){
    edge[++edge_cnt]=Edge{v,head[u]};
    head[u]=edge_cnt;
}
int w[MAXN],deg[MAXN],inv[MAXN];
int a[MAXN][MAXN],b[MAXN][MAXN],c[MAXN][MAXN],d[MAXN][MAXN],E[MAXN];
int dp[8][MAXN];
int quick_pow(int n,int k){
    int ans=1;
    while(k){
        if(k&1)ans=1LL*ans*n%mod;
        n=1LL*n*n%mod;
        k>>=1;
    }
    return ans;
}
void calc1(int n){
    static int temp[MAXN][MAXN<<1];
    n--;
    _rep(i,1,n){
        _rep(j,1,n){
            if(c[i][j])
                temp[i][j]=mod-inv[deg[j]];
            else
                temp[i][j]=0;
        }
        temp[i][i]=1;
        temp[i][i+n]=1;
    }
    _rep(i,1,n){
        int pos=-1;
```

```

    _rep(j,i,n){
        if(temp[j][i]){
            pos=j;
            break;
        }
    }
    if(pos!=i)swap(temp[i],temp[pos]);
    int div=quick_pow(temp[i][i],mod-2);
    for(int j=n*2;j>=i;j--)
        temp[i][j]=1LL*temp[i][j]*div%mod;
    _rep(j,1,n){
        if(j==i)continue;
        for(int k=n*2;k>=i;k--)
            temp[j][k]=(temp[j][k]-1LL*temp[j][i]*temp[i][k])%mod;
    }
}
_rep(i,1,n)_rep(j,1,n)
d[i][j]=(temp[i][j+n]+mod)%mod;
}
void add_edge(int u,int v,int &ans){
    _rep(t,0,MAXT)
    ans=(ans+1LL*dp[t][u]*inv[deg[u]]%mod*max(a[u][v]>>t,b[u][v]))%mod;
    ans=(ans+1LL*E[u]*inv[deg[u]]%mod*b[u][v])%mod;
}
void del_edge(int u,int v,int &ans){
    _rep(t,0,MAXT)
    ans=(ans-1LL*dp[t][u]*inv[deg[u]]%mod*max(a[u][v]>>t,b[u][v]))%mod;
    ans=(ans-1LL*E[u]*inv[deg[u]]%mod*b[u][v])%mod;
    ans=(ans+mod)%mod;
}
int calc2(int n){
    int ans=0,s=0;
    mem(dp,0);
    n--;
    _rep(i,1,n)s+=w[i];
    s=quick_pow(s,mod-2);
    _rep(i,1,n){
        dp[0][i]=1LL*w[i]*s%mod;
        E[i]=0;
        _rep(j,1,n)
        E[i]=(E[i]+1LL*w[j]*d[i][j])%mod;
        E[i]=1LL*E[i]*s%mod;
        E[i]=(E[i]+mod-dp[0][i])%mod;
    }
    _rep(t,1,MAXT){
        _rep(u,1,n){
            for(int i=head[u];i;i=edge[i].next){
                int v=edge[i].to;
                dp[t][v]=(dp[t][v]+1LL*dp[t-1][u]*inv[deg[u]])%mod;
            }
        }
    }
}

```

```
    _rep(u, 1, n)
        E[u] = (E[u] + mod - dp[t][u]) % mod;
    }
    _rep(u, 1, n) {
        for (int i = head[u]; i; i = edge[i].next) {
            int v = edge[i].to;
            add_edge(u, v, ans);
        }
    }
    return ans;
}
int main() {
    int n = read_int(), m = read_int(), q = read_int();
    _for(i, 1, n)
        w[i] = read_int();
    while (m--) {
        int u = read_int(), v = read_int();
        a[u][v] = a[v][u] = read_int();
        b[u][v] = b[v][u] = read_int();
        c[u][v] = c[v][u] = 1;
        Insert(u, v); Insert(v, u);
        deg[u]++; deg[v]++;
    }
    inv[1] = 1;
    _rep(i, 2, n)
        inv[i] = 1LL * (mod - mod / i) * inv[mod % i] % mod;
    calc1(n);
    int ans = calc2(n);
    enter(ans);
    while (q--) {
        int opt = read_int();
        if (opt == 1) {
            int u = read_int(), v = read_int();
            if (u > v) swap(u, v);
            del_edge(u, v, ans);
            if (v != n) del_edge(v, u, ans);
            a[u][v] = a[v][u] = read_int();
            b[u][v] = b[v][u] = read_int();
            add_edge(u, v, ans);
            if (v != n) add_edge(v, u, ans);
        }
        else {
            int u = read_int();
            w[u] = read_int();
            ans = calc2(n);
        }
        enter(ans);
    }
    return 0;
}
```

}

From: <https://wiki.cvbbacm.com/> - CVBB ACM Team

Permanent link: [https://wiki.cvbbacm.com/doku.php?id=2020-2021.teams:legal\\_string:%E7%BB%84%E9%98%F%E8%AE%AD%E7%BB%83%E6%AF%94%E8%B5%9B%E8%AE%B0%E5%BD%95%E7%BC%93%E5%86%B2%E5%8C%BA&rev=1630419404](https://wiki.cvbbacm.com/doku.php?id=2020-2021.teams:legal_string:%E7%BB%84%E9%98%F%E8%AE%AD%E7%BB%83%E6%AF%94%E8%B5%9B%E8%AE%B0%E5%BD%95%E7%BC%93%E5%86%B2%E5%8C%BA&rev=1630419404)

Last update: 2021/08/31 22:16