01Trie

```
#include <bits/stdc++.h>

using namespace std;

const int N=1e5+10;

int n;

int e[N<<1],ne[N<<1],h[N],idx,w[N<<1];
int d[N];
int nex[N*31][2],root,tot;

void add(int x,int y,int z){e[++idx]=y,ne[idx]=h[x],h[x]=idx;w[idx]=z;}

int newcode(){return tot++;}

void init(){
    tot=1;
    root=newcode();
}

void insert(int x){
    int p=root;
    for(int i=30;~i;i--){
        int t=(x>>i)&1;
        if(!nex[p][t]) nex[p][t]=newcode();
        p=nex[p][t];
    }
}

int query(int x){
    int p=root;
    int res=0;
    for(int i=30;~i;i--){
        int t=(x>>i)&1;
        if(nex[p][t^1]){
            res=res*2+1;
            p=nex[p][t^1];
        }
        else{
            res=res*2;
```

```
            p=nex[p][t];
        }
    }
    return res;
}

void dfs(int x,int fa){
    for(int i=h[x];i;i=ne[i]){
        int j=e[i];
        if(j==fa) continue;
        d[j]=d[x]^w[i];
        insert(d[j]);
        dfs(j,x);
    }
}

int main(){
    ios::sync_with_stdio(0);
    cin.tie(0),cout.tie(0);
    cin>>n;
    for(int i=1;i<n;i++){
        int u,v,w;
        cin>>u>>v>>w;
        add(u,v,w),add(v,u,w);
    }
    init();
    d[1]=0;
    insert(0);
    dfs(1,1);
    int ans=0;
    for(int i=1;i<=n;i++){
        ans=max(ans,query(d[i]));
    }
    cout<<ans<<'\n';
}
```

ODT

jiangly的896E代码，用map实现ODT

有四种类型的操作：

- 1 l r x：对于每个 i，使得 $l \le i \le r$，将 $a_i + x$ 分配给 a_i。
- 2 l r x：对于每个 i，使得 $l \le i \le r$，将 x 赋值给 a_i。
- 3 l r x：打印索引范围 $[l, r]$ 中第 x 个最小的数字，即第 x 个位置的元素如果将 $l \le i \le r$ 的所有元素 a_i 都取出并排序到一个非递减整数数组中。保证为 $1 \le x \le r - l + 1$。
- 4 l r x y：打印 a_i 的 x 次方之和，使得 $l \le i \le r$ 模 y，即。

```cpp
#include <bits/stdc++.h>

using i64 = long long;

int power(int a, int b, int p) {
    int res = 1 % p;
    for (; b; b /= 2, a = 1LL * a * a % p) {
        if (b % 2) {
            res = 1LL * res * a % p;
        }
    }
    return res;
}

int main() {
    std::ios::sync_with_stdio(false);
    std::cin.tie(nullptr);

    int n, m, seed, vmax;
    std::cin >> n >> m >> seed >> vmax;

    auto rnd = [&]() {
        int ret = seed;
        seed = (seed * 7LL + 13) % 1000000007;
        return ret;
    };

    std::vector<int> a(n);

    std::map<int, i64> f;
    for (int i = 0; i < n; i++) {
        a[i] = rnd() % vmax + 1;
        f[i] = a[i];
    }
    f[n] = -1;

    auto split = [&](int i) {
        auto it = std::prev(f.upper_bound(i));
        if (it->first != i) {
            f[i] = it->second;
        }
    };

    for (int i = 0; i < m; i++) {
        int op = rnd() % 4 + 1;
        int l = rnd() % n + 1;
```

```
        int r = rnd() % n + 1;
        if (l > r) {
            std::swap(l, r);
        }
        l--;

        int x;
        if (op == 3) {
            x = rnd() % (r - l);
        } else {
            x = rnd() % vmax + 1;
        }

        split(l);
        split(r);
        if (op == 1) {
            for (auto it = f.find(l); it->first != r; it++) {
                it->second += x;
            }
        } else if (op == 2) {
            for (auto it = f.find(l); it->first != r; it = f.erase(it))
                ;
            f[l] = x;
        } else if (op == 3) {
            std::vector<std::array<i64, 2>> v;
            for (auto it = f.find(l); it->first != r; it++) {
                v.push_back({it->second, std::next(it)->first - it->first});
            }
            std::sort(v.begin(), v.end());
            for (auto [a, b] : v) {
                if (x < b) {
                    std::cout << a << "\n";
                    break;
                }
                x -= b;
            }
        } else {
            int y = rnd() % vmax + 1;
            int ans = 0;
            for (auto it = f.find(l); it->first != r; it++) {
                ans = (ans + 1LL * (std::next(it)->first - it->first) * power(it-
>second % y, x, y)) % y;
            }
            std::cout << ans << "\n";
        }
    }
```

```
    return 0;
}
```