Problem Menu Back

Statement

Submissions

Your score not attempted

Spoilers

Show difficulty Show tags

solved by 2 / 5

Top users by score

#	User	Score
1	Zanite	100
2	Raphela	100
4	filiomerel	0
5	kaka_jayapura	0

Arkavidia 9.0 - Penyisihan CP > A

Submission #4625747

Arkavidia 9.0 - Penyisihan CP / A. Another Tree Cost Problem

Accepted • Raphela • C++17 • November 15, 2025 at 19:32:44

▶ Test Data Results

```cpp
#include <bits/stdc++.h>
using namespace std;

const int MAXN = 100000;
const int MAXK = 7;
const long long INF = (long long)4e18;

struct Edge {
    int to;
    long long w[MAXK];
};

struct Anc {
    int cid;
    long long dist[MAXK];
};

long long S[MAXK][MAXN + 5];
long long distR[MAXK][MAXN + 5];

bool removedNode[MAXN + 5];
int subSize[MAXN + 5];
vector<int> centroids;
vector<Anc> anc[MAXN + 5];

void dfsDistR(int v, int p) {
    for (const auto &e : g[v]) {
        int to = e.to;
        if (to == p) continue;
        for (int c = 0; c < K; ++c) {
            distR[c][to] = distR[c][v] + e.w[c];
        }
        dfsDistR(to, v);
    }
}

void dfsSize(int v, int p) {
    subSize[v] = 1;
    for (const auto &e : g[v]) {
        int to = e.to;
        if (to == p || removedNode[to]) continue;
        dfsSize(to, v);
        subSize[v] += subSize[to];
    }
`   Anc a;
    a.cid = cid;
    for (int c = 0; c < K; ++c) a.dist[c] = distArr[c];
    anc[v].push_back(a);
    for (const auto &e : g[v]) {
        int to = e.to;
        if (to == p || removedNode[to]) continue;
        long long nd[MAXK];
        for (int c = 0; c < K; ++c) nd[c] = distArr[c] + e.w[c];
        dfsDistCentroid(to, v, cid, nd);
    }
}

void decompose(int v) {
    dfsSize(v, 0);
    int size = subSize[v];
    int parent = 0;
    int cur = v;
    bool changed = true;
    while (changed) {
        changed = false;
        for (const auto &e : g[cur]) {
            int to = e.to;
            if (to == parent || removedNode[to]) continue;
            if (subSize[to] > size / 2) {
                parent = cur;
                cur = to;
                changed = true;
    }
    int cnode = cur;
    int cid = (int)centroids.size();
    centroids.push_back(cnode);
    removedNode[cnode] = true;
    long long distArr[MAXK];
    for (int i = 0; i < K; ++i) distArr[i] = 0;
    dfsDistCentroid(cnode, 0, cid, distArr);
    for (const auto &e : g[cnode]) {
        int to = e.to;
        if (removedNode[to]) continue;
        decompose(to);
    }
}

long long F[MAXK][MAXN + 5];
long long Bval[MAXN + 5];
long long best[MAXK][MAXN + 5];

```

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

    if (!(cin >> N >> K >> R)) return 0;

    for (int i = 0; i < N - 1; ++i) {
        int U, V;
        cin >> U >> V;
        for (int c = 0; c < K; ++c) {
            long long x;
            cin >> x;
            e1.w[c] = x;
            e2.w[c] = x;
        }
        g[U].push_back(e1);
        g[V].push_back(e2);
    }

    for (int c = 0; c < K; ++c) {
        for (int j = 1; j <= N; ++j) {
            long long x;
            cin >> x;
            S[c][j] = x;
        }
    }

    dfsDistR(R, 0);

    decompose(1);
    int C = (int)centroids.size();

    for (int c = 0; c < K; ++c) {
        for (int cid = 0; cid < C; ++cid) {
            best[c][cid] = INF;
        }
    }
            long long val = distR[c][i];
            long long q = INF;
            for (const auto &a : anc[i]) {
                int cid = a.cid;
                long long cand = best[c][cid] + a.dist[c];
                if (cand < q) q = cand;
            }
            if (q < val) val = q;
            F[c][i] = val;
        }

        long long bi = INF;
        for (int c = 0; c < K; ++c) {
            long long cand = S[c][i] + F[c][i];
            if (cand < bi) bi = cand;
        }
        Bval[i] = bi;
            int cid = a.cid;
            for (int c = 0; c < K; ++c) {
                long long cand = Bval[i] + a.dist[c];
                if (cand < best[c][cid]) best[c][cid] = cand;
            }
        }
    }

    for (int P = 1; P <= N; ++P) {
        long long ans = INF;
        for (int c = 0; c < K; ++c) {
            if (F[c][P] < ans) ans = F[c][P];
        }
        if (P == R) ans = 0;
        if (P > 1) cout << ' ';
        cout << ans;
    }
    cout << '\n';

    return 0;
}
```