



Problem Menu

[Back](#)[Statement](#)[Submissions](#)

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](#)

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

```
1 #include <bits/stdc++.h>
2 using namespace std;
3
4 const int MAXN = 100000;
5 const int MAXK = 7;
6 const long long INF = (long long)4e18;
7
8 struct Edge {
9     int to;
10    long long w[MAXK];
11 };
12
13 struct Anc {
14     int cid;
15    long long dist[MAXK];
16 };
17
18 long long S[MAXK][MAXN + 5];
19 long long distR[MAXK][MAXN + 5];
20
21 bool removedNode[MAXN + 5];
22 int subSize[MAXN + 5];
23 vector<int> centroids;
24 vector<Anc> anc[MAXN + 5];
25
26 void dfsDistR(int v, int p) {
27     for (const auto &e : g[v]) {
28         int to = e.to;
29         if (to == p) continue;
30         for (int c = 0; c < K; ++c) {
31             distR[c][to] = distR[c][v] + e.w[c];
32         }
33         dfsDistR(to, v);
34     }
35 }
36
37 void dfsSize(int v, int p) {
38     subSize[v] = 1;
39     for (const auto &e : g[v]) {
40         int to = e.to;
41         if (to == p || removedNode[to]) continue;
42         dfsSize(to, v);
43         subSize[v] += subSize[to];
44     }
45 }
46
47 Anc a;
48 a.cid = cid;
49 for (int c = 0; c < K; ++c) a.dist[c] = distArr[c];
50 anc[v].push_back(a);
51 for (const auto &e : g[v]) {
52     int to = e.to;
53     if (to == p || removedNode[to]) continue;
54     dfsSize(to, v);
55     subSize[v] += subSize[to];
56 }
57
58 void dfsDistCentroid(int v, int cid, long long nd) {
59     for (int c = 0; c < K; ++c) nd[c] = distArr[c] + e.w[c];
60     dfsDistCentroid(to, v, cid, nd);
61 }
62
63
64 void decompose(int v) {
65     dfsSize(v, 0);
66     int size = subSize[v];
67     int parent = 0;
68     int cur = v;
69     bool changed = true;
70     while (changed) {
71         changed = false;
72         for (const auto &e : g[cur]) {
73             int to = e.to;
74             if (to == parent || removedNode[to]) continue;
75             if (subSize[to] > size / 2) {
76                 parent = cur;
77                 cur = to;
78                 changed = true;
79             }
80         }
81         int cnode = cur;
82         int cid = (int)centroids.size();
83         centroids.push_back(cnode);
84         removedNode[cnode] = true;
85         long long distArr[MAXK];
86         for (int i = 0; i < K; ++i) distArr[i] = 0;
87         dfsDistCentroid(cnode, 0, cid, distArr);
88         for (const auto &e : g[cnode]) {
89             int to = e.to;
90             if (removedNode[to]) continue;
91             decompose(to);
92         }
93     }
94 }
95
96 long long F[MAXK][MAXN + 5];
97 long long Bval[MAXN + 5];
98 long long best[MAXK][MAXN + 5];
99
100
```

```

101 int main() {
102     ios::sync_with_stdio(false);
103     cin.tie(nullptr);
104
105     if (!(cin >> N >> K >> R)) return 0;
106
107     for (int i = 0; i < N - 1; ++i) {
108         int U, V;
109         cin >> U >> V;
110         for (int c = 0; c < K; ++c) {
111             long long x;
112             cin >> x;
113             e1.w[c] = x;
114             e2.w[c] = x;
115         }
116         g[U].push_back(e1);
117         g[V].push_back(e2);
118     }
119
120     for (int c = 0; c < K; ++c) {
121         for (int j = 1; j <= N; ++j) {
122             long long x;
123             cin >> x;
124             S[c][j] = x;
125         }
126     }
127
128     dfsDistR(R, 0);
129
130     decompose(1);
131     int C = (int)centroids.size();
132
133     for (int c = 0; c < K; ++c) {
134         for (int cid = 0; cid < C; ++cid) {
135             best[c][cid] = INF;
136         }
137     }
138
139     for (int i = 0; i < N; ++i) {
140         long long val = distR[c][i];
141         long long q = INF;
142         for (const auto &a : anc[i]) {
143             int cid = a.cid;
144             long long cand = best[c][cid] + a.dist[c];
145             if (cand < q) q = cand;
146         }
147         if (q < val) val = q;
148         F[c][i] = val;
149     }
150
151     long long bi = INF;
152     for (int c = 0; c < K; ++c) {
153         long long cand = S[c][i] + F[c][i];
154         if (cand < bi) bi = cand;
155     }
156     Bval[i] = bi;
157
158     for (int c = 0; c < K; ++c) {
159         int cid = a.cid;
160         for (int c = 0; c < K; ++c) {
161             long long cand = Bval[i] + a.dist[c];
162             if (cand < best[c][cid]) best[c][cid] = cand;
163         }
164     }
165
166     for (int P = 1; P <= N; ++P) {
167         long long ans = INF;
168         for (int c = 0; c < K; ++c) {
169             if (F[c][P] < ans) ans = F[c][P];
170         }
171         if (P == R) ans = 0;
172         if (P > 1) cout << ' ';
173         cout << ans;
174     }
175     cout << '\n';
176
177     return 0;
178 }
179
180
181
182
183 }
```