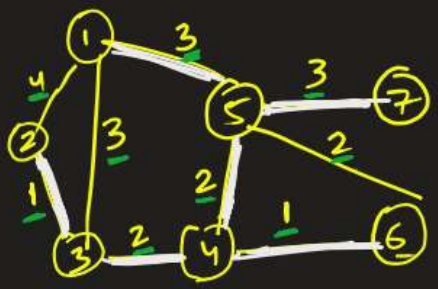
Trees & Graphs Lecture 6

Thursday, 22 August 2024 5:57 AM

Minimum Spanning Trees

Prims Algorithm



w, u, v (node, parent)

3	7	5

visited

T	T	T	T	T	T	T
F	F	F	F	F	F	F
1	2	3	4	5	6	7

min heap (pq)

<1, <2, 3>>
<2, <3, 4>>
<1, <6, 4>>
<2, <6, 5>>
<2, <4, 5>>
<3, <7, 5>>
<4, <2, 1>>
<3, <3, 1>>
<3, <5, 1>>

<w, <u, v>> edge

w: weight, u: node, v: parent

mst edges :- {{1,5}, {4,5}, {4,6}, {3,4}, {2,3}, {5,7}}

mst weight :- 0 3 5 6 8 9 12

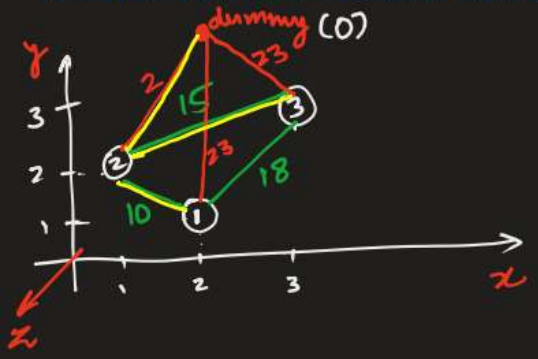
Space :- O(V + E)

Time :- O(E log E) = O(E log V)

https://www.geeksforgeeks.org/problems/minimum-spanning-tree/1

```cpp
class Solution {
    public:
    //Function to find sum of weights of edges of the Minimum Spanning Tree.
    int spanningTree(int V, vector<vector<int>> adj[]) {
        priority_queue<pair<int, int>, vector<pair<int, int>>, greater<pair<int, int>>> pq;
        vector<bool> vis(V, false);
        pq.push({0, 0});
        int ans = 0;
        while(!pq.empty()) {
            pair<int, int> e = pq.top();
            pq.pop();
            if(!vis[e.second]) {
                vis[e.second] = true;
                ans += e.first;
                for(auto e2: adj[e.second])
                    if(!vis[e2[0]])
                        pq.push({e2[1], e2[0]});
            }
        }
        return ans;
    }
};
```

https://codeforces.com/contest/1245/problem/D



$C_1 = 23,\ C_2 = 2,\ C_3 = 23$ ←

$K_1 = 3,\ K_2 = 2,\ K_3 = 3$

$W_{ij} = (|x_i - x_j| + |y_i - y_j|) * (K_i + K_j)$ ←

MST → ↓ It is necessary to connect all the node

SP ↓ Not necessary to connect all the nodes

```
int find(int x, vi &p) {
    if(p[x] >= 0)
        return p[x] = find(p[x], p);
    return x;
}
bool union_(int a, int b, vi &p) {
    a = find(a, p);
    b = find(b, p);
    if(a==b)    return false;
    if(p[a] < p[b]) p[b] = a;
    else if(p[a] > p[b]) p[a] = b;
    else {
        p[a] = b;
        p[b]--;
    }
    return true;
}
```

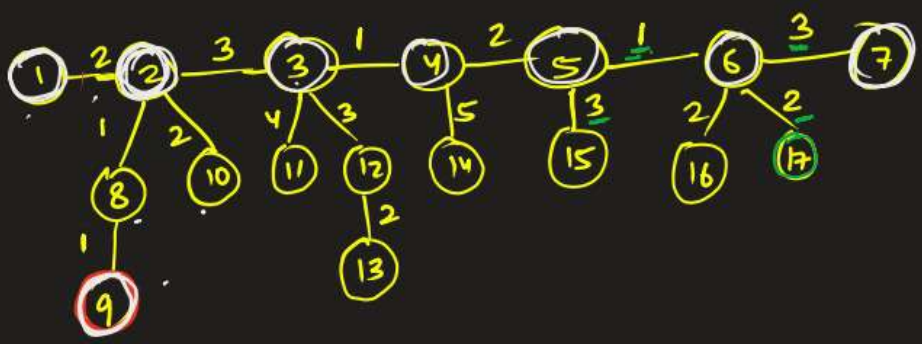
```cpp
void solve() {
    ll n, w;
    cin >> n;
    vll x(n+1), y(n+1), k(n+1);
    vector<pair<ll, pair<int, int> > > edges; // {w, {u, v}}
    FOR(i, 1, n+1) cin >> x[i] >> y[i];
    FOR(i, 1, n+1) {
        cin >> w;
        edges.pb(mp(w, mp(0, i)));
    }
    FOR(i, 1, n+1) cin >> k[i];
    FOR(i, 1, n+1)
        FOR(j, i+1, n+1)
            edges.pb(mp((k[i]+k[j])*(llabs(x[i]-x[j])+llabs(y[i]-y[j])), mp(i, j)));
    sort(edges.begin(), edges.end());
    // Kruskals MST algorithm
    vi ps;
    vpii con;
    ll ans = 0;
    vi p(n+1, -1);
    for(auto e: edges) {
        if(union_(e.second.first, e.second.second, p)) {
            ans += e.first;
            if(e.second.first == 0)
                ps.pb(e.second.second);
            else
                con.pb(e.second);
        }
    }
    cout << ans << endl;
    cout << ps.size() << endl;
    for(int s: ps) cout << s << " ";
    cout << endl << con.size() << endl;
    for(auto p: con) cout << p.first << " " << p.second << endl;
}
```

https://codeforces.com/contest/1004/problem/E

Diameter:- path on the tree with maximum distance

① Find the diameter of the tree.

Task
- From any node, find the farthest node (a)
- From a, find the farthest node (b)
- a-b is a diameter



```cpp
umap<int, int> a[100001];
bool vis[100001];
int n,k,u,v,d;
pii dfs(int s) {
    vis[s] = true;
    pii dis;
    int md = 0, node = s;
    for(auto p: a[s]) {
        if(!vis[p.first]) {
            dis = dfs(p.first);
            if(dis.second+p.second > md) {
                md = dis.second+p.second;
                node = dis.first;
            }
        }
    }
    return mp(node, md);
}
```

```cpp
pii dia_nodes() {
    FOR(i,0,n+1) vis[i] = false;
    pii p1 = dfs(1);
    FOR(i,0,n+1) vis[i] = false;
    pii p2 = dfs(p1.first);
    return mp(p1.first, p2.first);
}
void solve() {
    cin >> n >> k;
    FOR(i, 0, n-1) {
        cin >> u >> v >> d;
        a[u][v] = d;
        a[v][u] = d;
    }
    pii dn = dia_nodes();
    cout << dn.first << " " << dn.second;
}
```