> C-2024-打印准考证
>
> 23371041-李一鸣

题目描述

- 给定 $G(V,E)$ 和 k 个点 $p_1p_2\dots p_k$，求至少经过 P 中任意一点且从 $1\sim n$ 的最短路径

解题思路

- 最短路问题，但是要求经过 P 中任意一点
- 可以考虑用 $Dijkstra$ 求出 1 到任意一点的最小距离和 n 到任意一点的最小距离，然后进行松弛
- 然后枚举 p_i，更新 $ans = min_{1\le i\le k}(dis_1(p_i)+dis_n(p_i))$
- 复杂度分析：$\Theta(n\lg n)$

代码实现

```
struct Node {
    int now, val;
    Node(int _now, int _val) : now(_now), val(_val) {}
    bool operator < (const Node& b) const {
        return val > b.val;
    }
};

int to[1000010], next[1000010], head[100010], val[1000010];
int tot;

void addEdge(int u, int v, int w) {
    to[++tot] = v, next[tot] = head[u], head[u] = tot, val[tot] = w;
}

int dist[2][100010];

std :: priority_queue<Node> q;

int n;

void dijkstra(int s, int index) {
    // memset(dist, 0x7f, sizeof(dist));
    for(int i = 1; i <= n; ++i)
        dist[index][i] = 1ll << 61;
    q.push(Node(s, 0));
    dist[index][s] = 0;
    while(!q.empty()) {
        Node now = q.top();
        q.pop();
        int u = now.now, d = now.val;
        if(d > dist[index][u])
            continue;
        for(int i = head[u]; i; i = next[i]) {
            int v = to[i], w = val[i];
            if(dist[index][u] + w < dist[index][v]) {
```

```cpp
                dist[index][v] = dist[index][u] + w;
                q.push(Node(v, dist[index][v]));
            }
        }
    }
}

int p[100010];

signed main() {
    int T = read();
    // int T = 1;
    while(T--) {
        n = read();
        int m = read(), k = read();
        for(int i = 1; i <= k; ++i)
            p[i] = read();
        memset(head, 0, sizeof(head));
        tot = 0;
        for(int i = 1; i <= m; i++) {
            int u = read(), v = read(), w = read();
            addEdge(u, v, w);
            addEdge(v, u, w);
        }
        int ans = 1ll << 61;
        dijkstra(1, 0);
        dijkstra(n, 1);
        for(int i = 1; i <= k; i++) {
            int pp = p[i];
            ans = std::min(ans, dist[0][pp] + dist[1][pp]);
        }
        write(ans > 1ll << 60 ? -1 : ans);
        putchar('\n');
    }
}
```