memset0's Notebook

方知暮然回首之时
那人却已不在灯火阑珊处

关于我
友情链接
文章聚合

洛谷4208 [JSOI2008]最小生成树计数

2019-03-06 | 题解

一道还是挺有意思的题，讲下某神仙课件里的 Matrix Tree 定理做法。

首先最小生成树有个很显然的性质，就是对于任意两颗同一个图的最小生成树，他们的边权序列排序后应当相同。可以考虑克鲁斯卡尔算法的过程用归纳法证明。

也就是说如果两棵最小生成树不同，一定是克鲁斯卡尔算法加边时，对 **同一边权** 的 **相同数量** 条边的选择不同。

容易发现，每次以任意顺序加完所有相同边权的边后，图的连通性相同，而新加的边，一定是将原图的一些不连通的联通块连接起来。考虑暴力做法 2^{10} 枚举，即可有一种朴素的 AC 算法。

然而此题可以跑到 $O(n+mk^2)$ ，可以发现选边的过程类似于生成树计数，我们可以直接用 Matrix Tree 定理求出对于每个连边后联通的联通块的方案树，根据乘法原理相乘即可。

需要注意：

1. 题目给出的数不是质数~~（我就因为这个调了很久~~

2. 需要把加边前已经联通的联通块缩点

代码：

```
// ================================
//   author: memset0
//   date: 2019.03.05 14:57:27
//   website: https://memset0.cn/
// ================================
#include <bits/stdc++.h>
#define ll long long
namespace ringo {
template <class T> inline void read(T &x) {
    x = 0; register char c = getchar(); register bool f = 0;
    while (!isdigit(c)) f ^= c == '-', c = getchar();
    while (isdigit(c)) x = x * 10 + c - '0', c = getchar();
    if (f) x = -x;
}
template <class T> inline void print(T x) {
    if (x < 0) putchar('-'), x = -x;
    if (x > 9) print(x / 10);
    putchar('0' + x % 10);
}
template <class T> inline void print(T x, char c) { print(x), putchar(c); }

const int N = 110, M = 1010, mod = 31011;
int n, m, ans, fa[N], vis[N], map[N], A[N][N];
std::vector <int> G[N];

inline int dec(int a, int b) { a -= b; return a < 0 ? a + mod : a; }
inline int inc(int a, int b) { a += b; return a >= mod ? a - mod : a; }
inline int mul(int a, int b) { return (ll)a * b - (ll)a * b / mod * mod; }
inline int inv(int x) { return x < 2 ? 1 : (ll)(mod - mod / x) * inv(mod % x)

inline int find(int x) { return fa[x] == x ? x : fa[x] = find(fa[x]); }
```

memset0's Notebook

方知蓦然回首之时
那人却已不在灯火阑珊处

关于我
友情链接
文章聚合

Theme Ringo by memset0
Proudly powered by Typecho

```
struct edge {
    int u, v, w;
    inline bool operator < (const edge &other) const {
        return w < other.w;
    }
} e[M];

void dfs(int u, std::vector <int> &node) {
    vis[u] = 1, map[u] = node.size(), node.push_back(u);
    for (int i = 0, v; i < G[u].size(); i++) {
        v = G[u][i];
        if (!vis[v]) dfs(v, node);
    }
}

int calc(int a[N][N], int n) {
    if (n == 1) return a[1][1];
    static double tmp, f[N][N];
    for (int i = 1; i <= n; i++)
        for (int j = 1; j <= n; j++)
            f[i][j] = a[i][j];
    for (int i = 1, j; i <= n; i++) {
        for (j = i; j <= n; j++)
            if (fabs(f[j][i]) > 1e-7) break;
        if (j > n) return 0;
        if (i != j) std::swap(f[i], f[j]);
        for (int j = i + 1; j <= n; j++) {
            tmp = f[j][i] / f[i][i];
            for (int k = i; k <= n; k++) f[j][k] -= f[i][k] * tmp;
        }
    }
    double ans = 1;
    for (int i = 1; i <= n; i++) ans = ans * f[i][i];
    return (int)(ans + .5);
}

void solve(int u) {
```

```
    std::vector <int> node; dfs(u, node);
    if (node.size() == 1) return;
    for (int i = 0; i < node.size(); i++)
        for (int j = 0; j < node.size(); j++)
            A[i][j] = 0;
    for (int i = 0; i < node.size(); i++) A[i][i] = G[node[i]].size();
    for (int i = 0; i < node.size(); i++)
        for (int v, j = 0; j < G[node[i]].size(); j++) {
            v = G[node[i]][j];
            A[i][map[v]]--;
        }
    ans = (ll)ans * calc(A, node.size() - 1) % mod;
}

void main() {
    read(n), read(m);
    for (int i = 1, u, v, w; i <= m; i++) {
        read(u), read(v), read(w);
        e[i] = (edge){u, v, w};
    }
    for (int i = 1; i <= n; i++) fa[i] = i;
    std::sort(e + 1, e + m + 1);
    ans = 1;
    for (int L = 1, R = 0; L <= m; L = R + 1) {
        while (1) { ++R; if (e[R + 1].w != e[R].w) break; }
        memset(G, 0, sizeof(G));
        memset(vis, 0, sizeof(vis));
        for (int u, v, i = L; i <= R; i++) {
            u = find(e[i].u), v = find(e[i].v);
            if (u != v) G[u].push_back(v), G[v].push_back(u);
        }
        for (int i = 1; i <= n; i++) if (!vis[i]) solve(i);
        for (int i = L, u, v; i <= R; i++) {
            u = find(e[i].u), v = find(e[i].v);
            if (u != v) fa[u] = v;
        }
    }
```

```
    for (int i = 1; i <= n; i++) if (find(i) != find(1)) { ans = 0; break; }
    print(ans, '\n');
}

} signed main() { return ringo::main(), 0; }
```

Matrix Tree

洛谷5070 [Ynoi2015]即便看不到未来
上一篇 «

洛谷4647 [IOI2007] sails 船帆
» 下一篇

memset0's Notebook

方知暮然回首之时
那人却已不在灯火阑珊处

在这里输入关键字哦 ~（回车搜索）

关于我
友情链接
文章聚合

↑