「WC2011」Xor - 线性基

2017-02-26 | OI

给一个无向连通图，求一条从 1 到 n 的路径（可以不是简单路径），使经过的边权的异或和最大。

链接

BZOJ 2115

题解

首先，一个显然的结论是，图中所有简单环的异或和，都是可以直接获得的 —— 因为我们可以从起点走到一个环，由 v 进入这个环，走一圈回到 v 然后原路返回起点，这样起点和 v 之间的边都经过了两次，相当于没有经过这些边，而环上的边权被留在了答案中。

我们可以 BFS 求出所有简单环的边权异或和，并求出它们的线性基。任意找一条从 1 到 n 的路径，然后从大到小考虑线性基中的每个元素加入到答案中会不会使答案更大即可。

证明：设当前答案为 s，考虑到的线性基中的当前元素为 x，x 控制着线性基中的第 k 位，那么分为三种情况：

1. s 中第 k 位为 1，x 中第 k 位为 1，此时不选 x；
2. s 中第 k 位为 0，x 中第 k 位为 1，此时选 x；
3. x 中第 k 位为 0，此时一定有 $x = 0$，不会影响。

根据归纳法，最终答案是正确的。

代码

```
#include <cstdio>
#include <vector>
#include <queue>

const int MAXN = 50000;
```

```
const int MAXL = 60;

struct Node {
    std::vector<struct Edge> e;
    Node *fa;
    long long dist;
    int dep;
} N[MAXN + 1];

struct Edge {
    Node *s, *t;
    long long w;

    Edge(Node *s, Node *t, long long w) : s(s), t(t), w(w) {}
};

inline void addEdge(int s, int t, long long w) {
    N[s].e.push_back(Edge(&N[s], &N[t], w));
    N[t].e.push_back(Edge(&N[t], &N[s], w));
}

inline std::vector<long long> bfs() {
    std::queue<Node *> q;
    q.push(&N[1]);
    N[1].dep = 1;

    std::vector<long long> circles;
    while (!q.empty()) {
        Node *v = q.front();
        q.pop();

        for (Edge *e = &v->e.front(); e <= &v->e.back(); e++) {
            if (!e->t->dep) {
                e->t->dep = v->dep + 1;
                e->t->dist = v->dist ^ e->w;
                e->t->fa = v;
                q.push(e->t);
            } else if (e->t != v->fa) {
                circles.push_back(e->t->dist ^ v->dist ^ e->w);
            }
        }
    }

    return circles;
}

struct LinearBasis {
    std::vector<long long> v;

    void build(std::vector<long long> x) {
        std::vector<long long> a(MAXL + 1);
```

```
    for (int i = 0; i < int(x.size()); i++) {
        long long t = x[i];

        for (int j = MAXL; j >= 0; j--) {
            if (!(t & (1ll << j))) continue;

            if (a[j]) t ^= a[j];
            else {
                for (int k = 0; k < j; k++) if (t & (1ll << k)) t ^= a[k];
                for (int k = j + 1; k <= MAXL; k++) if (a[k] & (1ll << j)) a[k] ^= t
                a[j] = t;
                break;
            }
        }
    }

    v.clear();
```

BZOJ # 线性基

‹ 「HDU 3949」XOR - 线性基

「BZOJ 2844」albus 就是要第一个出场 - 线性基 ›

0条评论 Menci's Blog Disqus 隐私政策 1 登录

推荐 推文 分享 评分最高

开始讨论…

通过以下方式登录

或注册一个 DISQUS 帐号 ?

姓名

来做第一个留言的人吧!

订阅 在您的网站上使用 Disqus添加 Disqus添加 不要出售我的数据

© 2015 – 2021 ♥ Menci

运行于 GigsGigsCloud 云平台 | 由 Upyun 提供 CDN 服务

由 Hexo 强力驱动 | 主题 — NexT.Pisces v5.1.2