

Atcoder Rugular Contest 127

[比赛链接](#)

D - Sum of Min of Xor

题意

给定序列 A, B 求 $\sum_{1 \leq i < j \leq n} \min(a_i \oplus a_j, b_i \oplus b_j)$

题解

从高到低考虑每个位，当 $a_i \oplus a_j$ 和 $b_i \oplus b_j$ 第一个位出现不同时已经可以判断 $a_i \oplus a_j$ 和 $b_i \oplus b_j$ 的大小关系。

注意到这也等价于 $a_i \oplus b_i$ 和 $a_j \oplus b_j$ 的第一个位出现不同。

假设当前考虑到第 k 位，此时将所有下标 i 划分成 S, T 其中 S 中每个 i 满足 $a_i \oplus b_i$ 第 k 位为 0 ， T 中每个 i 满足 $a_i \oplus b_i$ 第 k 位为 1 。

递归处理 $\{i, j\} \subseteq S$ 或 $\{i, j\} \subseteq T$ 的 (i, j) 对的贡献。现考虑如何处理 $i \in S, j \in T$ 的 (i, j) 对贡献。

根据 a_i 的第 k 位是否为 0 可以将 S 分为 S_0, S_1 同理将 T 分为 T_0, T_1

不难发现，对 $i \in S_0, j \in T_0$ 有 $\min(a_i \oplus a_j, b_i \oplus b_j) = a_i \oplus a_j$ 这转化为一个 $O(n \log V)$ 的经典问题。

其余的 $(S_0, T_1), (S_1, T_0), (S_1, T_1)$ 也有类似的解法。算上分治，总复杂度为 $O(n \log^2 V + 2^V)$

需要注意的是，递归最后一层为 $k = -1$ 即 $a_i \oplus b_i = a_j \oplus b_j$ 此时有 $\min(a_i \oplus a_j, b_i \oplus b_j) = a_i \oplus a_j = b_i \oplus b_j$ 直接处理即可。

```
const int MAXN = 2.5e5 + 5, MAXD = 18;
int a[MAXN], b[MAXN], cnt[2];
LL ans;
void calc(vector<int> L, vector<int> R) {
    _for(i, 0, MAXD) {
        LL s = 0;
        cnt[0] = cnt[1] = 0;
        for (int v : L)
            cnt[(v >> i) & 1]++;
        for (int v : R)
            s += cnt[!((v >> i) & 1)];
        ans += (1LL << i) * s;
    }
}
```

```
    }
}
void solve(vector<pair<int, int>> c, int d) {
    if (c.size() == 0) return;
    if (d == -1) {
        _for(i, 0, MAXD) {
            LL s = 0;
            cnt[0] = cnt[1] = 0;
            for (pair<int, int> p : c)
                cnt[(p.first >> i) & 1]++;
            ans += (1LL << i) * cnt[0] * cnt[1];
        }
        return;
    }
    vector<pair<int, int>> L, R;
    for (pair<int, int> p : c) {
        if ((p.first ^ p.second) & (1 << d))
            R.push_back(p);
        else
            L.push_back(p);
    }
    solve(L, d - 1);
    solve(R, d - 1);
    vector<int> temp[2][4];
    for (pair<int, int> p : L) {
        if (p.first & (1 << d)) {
            temp[0][2].push_back(p.first);
            temp[0][3].push_back(p.second);
        }
        else {
            temp[0][0].push_back(p.first);
            temp[0][1].push_back(p.second);
        }
    }
    for (pair<int, int> p : R) {
        if (p.first & (1 << d)) {
            temp[1][2].push_back(p.first);
            temp[1][1].push_back(p.second);
        }
        else {
            temp[1][0].push_back(p.first);
            temp[1][3].push_back(p.second);
        }
    }
    _for(i, 0, 4)
        calc(temp[0][i], temp[1][i]);
}
int main() {
    int n = read_int();
    _for(i, 0, n)
```

```
    a[i] = read_int();
    _for(i, 0, n)
        b[i] = read_int();
    vector<pair<int, int>> c;
    _for(i, 0, n)
        c.push_back(make_pair(a[i], b[i]));
    solve(c, MAXD);
    enter(ans);
    return 0;
}
```

E - Priority Queue

题意

给定一个长度为 $n+m$ 的操作序列，有 n 次 push 操作和 m 次 pop 操作。

操作过程中维护一个大根堆的优先队列，同时 push 的所有元素恰好是 $1 \sim n$ 的一个排列。问最后优先队列中的元素集合的种数。

题解

不难构造出令最后优先队列中剩余元素尽可能大的方案：第 i 次 push 元素 i

假定上述方案中

From:

<https://wiki.cvbbacm.com/> - CVBB ACM Team

Permanent link:

https://wiki.cvbbacm.com/doku.php?id=2020-2021:teams:legal_string:jxm2001:contest:arc_127&rev=1632731222

Last update: 2021/09/27 16:27