可持久化数据结构

杨尚霖

July 2024

简介

我们称每次修改操作后的数据结构为一个历史版本，而数据结构可持久化后可以访问任意历史版本并作出修改。

OI 常见的大概只有可持久化线段树，所谓的可持久化字典树一般指可持久化 0-1 Trie，可以看作是值域线段树。

可持久化线段树

我们以洛谷 P3834 这道模板为例，求静态区间第 k 小。

考虑如果是静态全局第 k 小怎么做？当然是~~sort~~权值线段树了，在线段树上二分即可，用权值线段树也可以支持动态全局第 k 小。

然后可以想到在静态区间第 k 小中，对于每个前缀 $a_{1,2,\ldots,i}$ 都建一个权值线段树 T_i，T_i 中代表区间 $[l,r]$ 的节点中存前缀 $a_{1,2,\ldots,i}$ 中有多少数在 $[l,r]$ 内。

对于询问 $[l,r]$，我们只要将 T_r 与 T_{l-1} 两棵树中的对应结点相减，就得到了区间 $[l,r]$ 的权值线段树。

然而这样一共需要 $O\left(n^2\right)$ 个结点，考虑优化。

可持久化线段树

观察 T_i 相较于 T_{i-1} 有什么变化，实际上就是在 T_{i-1} 的基础上修改了从 a_i 对应的叶子结点到根结点的路径上的权值，而这只有 $O(\log n)$ 个结点。

从 T_{i-1} 到 T_i 的过程中，我们只需要新建发生变化的 $O(\log n)$ 个结点即可，剩下的结点可以与 T_{i-1} 共用。

区间修改同理，同样只有 $O(\log n)$ 个结点被修改。

题外话：动态区间第 k 小若允许离线则可以用整体二分做，强制在线则可以用 BIT 套权值线段树解决。

实现

```
struct node {
    int ls, rs, sum;
} T[N * 100];

void upd(int &p, int vp, int l, int r, int gk, int k) {
    T[p = ++tot] = T[vp];
    T[p].sum += k;
    if (l == r) return;
    int mid = (l + r) >> 1;
    if (mid >= gk) upd(T[p].ls, T[vp].ls, l, mid, gk, k);
    else upd(T[p].rs, T[vp].rs, mid + 1, r, gk, k);
}
```

P3899 [湖南集训] 更为厉害

n 个点的树，q 次询问，每次给出 p, k，求有序三元组 (p, x, y) 的数量，满足：

- p, x, y 两两不同。
- p, x 均为 y 的祖先节点。
- p, x 距离不超过 k。

$n, q \leq 3 \times 10^5$。

sol

首先有 p 是 x 的祖先或 x 是 p 的祖先。如果 x 是 p 的祖先，这很好办。

如果 p 是 x 的祖先，问题就转化为求满足 x 在 p 子树内且 $dep_x <= dep_p + k$ 的 siz_x 之和。我们将子树的限制用 dfn 序表示，这就变成了一个二维数点。

可持久化与二维数点

可持久化线段树一般可以用来在线处理二维数点询问。在这题中，每个点 x 可以看作二维平面上的点 (dfn_x, dep_x)，有点权 siz_x，我们问的就是一块矩形中所有点的权值和。按 dfn 序（第一维）把所有点排序，版本 T_i 这棵线段树上区间 $[l, r]$ 存储对于前 i 个点，深度（第二位）在 $[l, r]$ 中的所有点的权值和。

对于询问 $[l1, r1] \times [l2, r2]$，先差分成 $[1, r1] \times [l2, r2]$ $- [1, l1 - 1] \times [l2, r2]$。在 T_{r1}, T_{l1-1} 上分别查询 $[l2, r2]$ 区间和即可。

当然本题没有强制在线，可以离线下来只用线段树解决。

P2633 Count on a tree

给定一棵 n 个节点的树，每个点有一个权值。有 m 个询问，每次给你 u, v, k，你需要回答 u xor last 和 v 这两个节点间第 k 小的点权。
其中 last 是上一个询问的答案，定义其初始为 0，即第一个询问的 u 是明文。

$n, m \leq 10^5$。

sol

静态区间第 k 小变成了静态路径第 k 小。

实际上思路是一样的，序列可以差分，树上路径也可以差分。

具体地，做区间第 k 小的时候 T_i 是从 T_{i-1} 修改来的；挪到树上，我们只需要让 T_u 从 T_v 修改来即可，其中 v 是 u 的父结点。

SP11470 To the moon

一个长度为 n 的数组 $\{A\}$，m 次操作，有 4 种：

- C l r d：区间 $[l, r]$ 中的数都加 d，同时当前的时间戳加 1。
- Q l r：查询当前时间戳区间 $[l, r]$ 中所有数的和。
- H l r t：查询时间戳 t 区间 $[l, r]$ 的和。
- B t：将当前时间戳置为 t。

$n, m \leq 10^5$。

可持久化的标记问题

核心是把所有信息发生改变的结点都建新的结点，由于只有我们访问到的结点会发生改变，所以时间正确空间也肯定正确。

由于在一个过去版本 push_down 会影响之后的版本，所以我们需要标记永久化。

可持久化并查集

实际上就是把并查集的 fa 和 siz 数组可持久化。

注意这里要用启发式合并。路径压缩的复杂度是均摊意义下的，一次 find 操作可以达到 $O(n)$，故不能使用。

我基本上没见过可持久化并查集，大概没人会出强制在线吧，都可以离线解决。

版本树

我们将每个版本看成一个结点，版本 i 是从版本 j 的基础上修改来的则从 j 向 i 连边，形成以最初版本为根节点的外向树。

如果没有强制在线，则可以这样离线，dfs 这颗版本树，每条边代表一个操作，进入某个子树则执行操作，退出子树则撤销操作。通过离线我们可以降低空间复杂度。

比如可持久化并查集就可以直接建版本树降到 $O(n)$ 空间复杂度。当然由于需要撤销我们还是不能用路径压缩。

可持久化 01Trie

实际上你可以把 01Trie 看成是值域为 $[0, 2^\omega)$ 的动态开点权值线段树。它相较于普通权值线段树的一个优势是由于它还是一个二进制下的字典树，它可以高效执行一些按位运算。

当然它的可持久化方法和可持久化线段树完全一致。

P4735 最大异或和

给定一个非负整数序列 $\{a\}$，初始长度为 n。
有 m 个操作，有以下两种操作类型：

- A x：添加操作，表示在序列末尾添加一个数 x，序列的长度 N 加 1。
- Q l r x：询问操作，你需要找到一个位置 p，满足 $l \leq p \leq r$，使得：$a[p] \oplus a[p+1] \oplus ... \oplus a[N] \oplus x$ 最大，输出最大值。

$n, m \leq 3 \times 10^5$。

sol

看到这一串连续的下标异或起来，应该马上想到转化成前缀异或和，记为 s_i。

就是求区间 $[l-1, r-1]$ 中与 $s_n \oplus x$ 异或和最大的数。

也就是求区间 $[l-1, r-1]$ 的 01Trie，还是利用差分即可。

P9329 [JOISC 2023 Day1] Two Currencies

n 个结点的树，经过一条边 w 需要支付 1 枚金币或 c_q 枚银币。

有 q 个人，第 i 个初始持有 x_i 枚金币和 y_i 枚银币，并想从 s_i 到达 t_i，问他最多能剩多少枚金币。

sol

等价于问最多能选多少条边支付银币，显然答案可二分。

设选了 k 条边支付银币，那么我们肯定是要选需要支付银币的数量前 k 小的对吧。于是建完可持久化线段树后利用树上差分求出路径 $s_i \to t_i$ 的权值线段树，线段树上二分即可。

可以离线用整体二分做到时间 $O(n \log n)$，空间 $O(n)$。

P2839 [国家集训队] middle

一个长度为 n 的序列 a，设其排过序之后为 b，其中位数定义为 $b_{n/2}$，其中 a, b 从 0 开始标号，除法下取整。
给你一个长度为 n 的序列 s。回答 Q 个这样的询问：s 的左端点在 $[a, b]$ 之间，右端点在 $[c, d]$ 之间的区间中最大的中位数，其中 $a < b < c < d$。
位置也从 0 开始标号。

强制在线。

sol

中位数是可二分的。设二分答案为 k。把序列中 $\geq k$ 的位置设为 1，$< k$ 的位置设为 -1，则序列的中位数 $\geq k$ 当且仅当和非负。

我们二分答案，然后就是查询 $[a, b]$ 的最大后缀和，$[c, d]$ 的最大前缀和以及 (b, c) 的区间和。

这需要对 n 个值每个值都建一棵线段树，我们可持久化一下就好了。

P4098 [HEOI2013] ALO

给定长为 n 的序列 $a_1, a_2, \ldots, a_n$。你可以任意选取一个区间 $[l, r]$，获得区间次大值与区间中另一个任意选取的数的异或和。问最大能得到的异或和是多少。

sol

直接枚举次大值是 a_i，考虑有哪些区间以 a_i 为次大值，常见的套路。

设 a_i 左侧第一个大于 a_i 的位置是 $l1$，第二个是 $l2$，$r1, r2$ 同理。

那我们只需要关心区间 $[l2+1, r1-1]$ 与 $[l1+1, r2-1]$ 就好了，其它所有以 a_i 为次大值的区间都是这两个区间的子区间。

可持久化 01Trie 上查询区间 $[l2+1, r2-1]$ 中的数与 a_i 异或的最大值即可。

P4592 [TJOI2018] 异或

现在有一颗以 1 为根节点的由 n 个节点组成的树，节点从 1 至 n 编号。树上每个节点上都有一个权值 v_i。现在有 q 次操作，操作如下：

- 1 x z：查询节点 x 的子树中的节点权值与 z 异或结果的最大值。
- 2 x y z：查询节点 x 到节点 y 的简单路径上的节点的权值与 z 异或结果最大值。

LOJ#3628「2021 集训队互测」树上的孤独

两棵树，A 树有 n 个结点，B 树有 m 个，每个结点都有颜色。q 次操作，有两种：

- P1 P2 P3 P4：记上次询问答案是 lst，$D1 = P3 \oplus lst$，$D2 = P4 \oplus P4$，求 A 树上距 P1 不超过 $D1$ 和 B 树上距 $P2$ 不超过 $D2$ 的所有点中，共有多少种颜色。
- S1 S2 将 A 树的 S1 点颜色改为 $S2$。

$n \leq 20, m \leq 2 \times 10^5, q \leq 10^6$。

sol

重点的是考虑怎么求 $P2$ 子树中距离 $P2$ 不超过 $D2$ 的点的颜色种数。剩下的我们可以 $O(nq)$ 做。

先重剖。对于这种颜色种数考虑启发式合并。对于一个子树中的一种颜色，我们只记录它在子树中出现的深度最浅的位置，这可以通过启发式合并完成。如果我们要做的是子树中全部点的颜色种数，启发式合并即可完成。而这题多了一维深度的限制，于是可以考虑用可持久化线段树做查询。
具体地，T_u 上的区间 $[l,r]$ 存储 u 子树中有多少种颜色最浅深度在 $[l,r]$ 内。dfs 到 u 时，先 dfs u 的所有轻子树，然后 dfs u 的重儿子 v，令 T_u 从 T_v 的基础上修改来，从 v 回到 u 时，再遍历 u 的轻子树，如果一种颜色出现在了更浅的位置（设原来是 $d1$ 更浅的深度是 $d2$），则在 T_u 中在 $d1$ 处减一，在 $d2$ 处加一。

由于一个点只会被淘汰一次，故总复杂度
$O(nq+(q+m)\log m)$。