关于可持久化数据结构 (2014-09-19 00:38:56)

转 载 ▼

学到一个非常猎奇的trick可以可持久化任意度数为O(1)的数据结构,且不产生额外复杂度。（比如要记parent的数据结构就可以用这个方法持久化了）

正常可持久化有两个方法，一个是OI界最常见的path copy（但记父亲会跪），另一个所谓“fat node”：对每个结点开个set记录所有历史修改，查询时候直接查set（但复杂度要乘log t）。

现在我们把两个方法结合起来：
每个结点依然是fat node，但只开一个额外域，也就是set里只能有一个元素。 如果set满了，就直接做path copy，而最tricky的地方是，这时候，path copy修改它的父结点时候，可以直接用fat node的方法去改！也就是如果父亲结点的set是空的，我们可以直接用fat node的方法记录对父结点的这次修改！（如果父亲结点set也满了就继续path copy父结点）

这样一来，我们可以发现这个数据结构能在保证时间复杂度的情况下支持记录（一个）父亲了。（想象一下修改的复杂度，每2次操作就要对父亲层改1次，每4次操作就要对祖父层改1次，etc..... 类似等比数列求和，公比是1/2，不影响均摊复杂度）

而有多个父亲怎么办呢？ 比如有两个父亲，那容易发现上面方法就跪了，因为每2次操作就要对父亲层改2次，每4次操作就要对祖父层改4次，etc..... 这个等比数列的和就不再和首项同阶了.....
怎么办呢？更加猎奇的trick：只要更改set的最大大小即可！ 假设最多有k个父亲，那么设置set最大大小为k！这样那个等比数列公比就是k/k+1，就可以了.....

应用：
1. 要求记录父亲的数据结构的可持久化
2. 改进可持久化（无序）链表的复杂度到修改O(1)遍历O(n)（不再需要用可持久化平衡树维护链表了，直接用上面的方法去维护，这样修改就O(1)了）