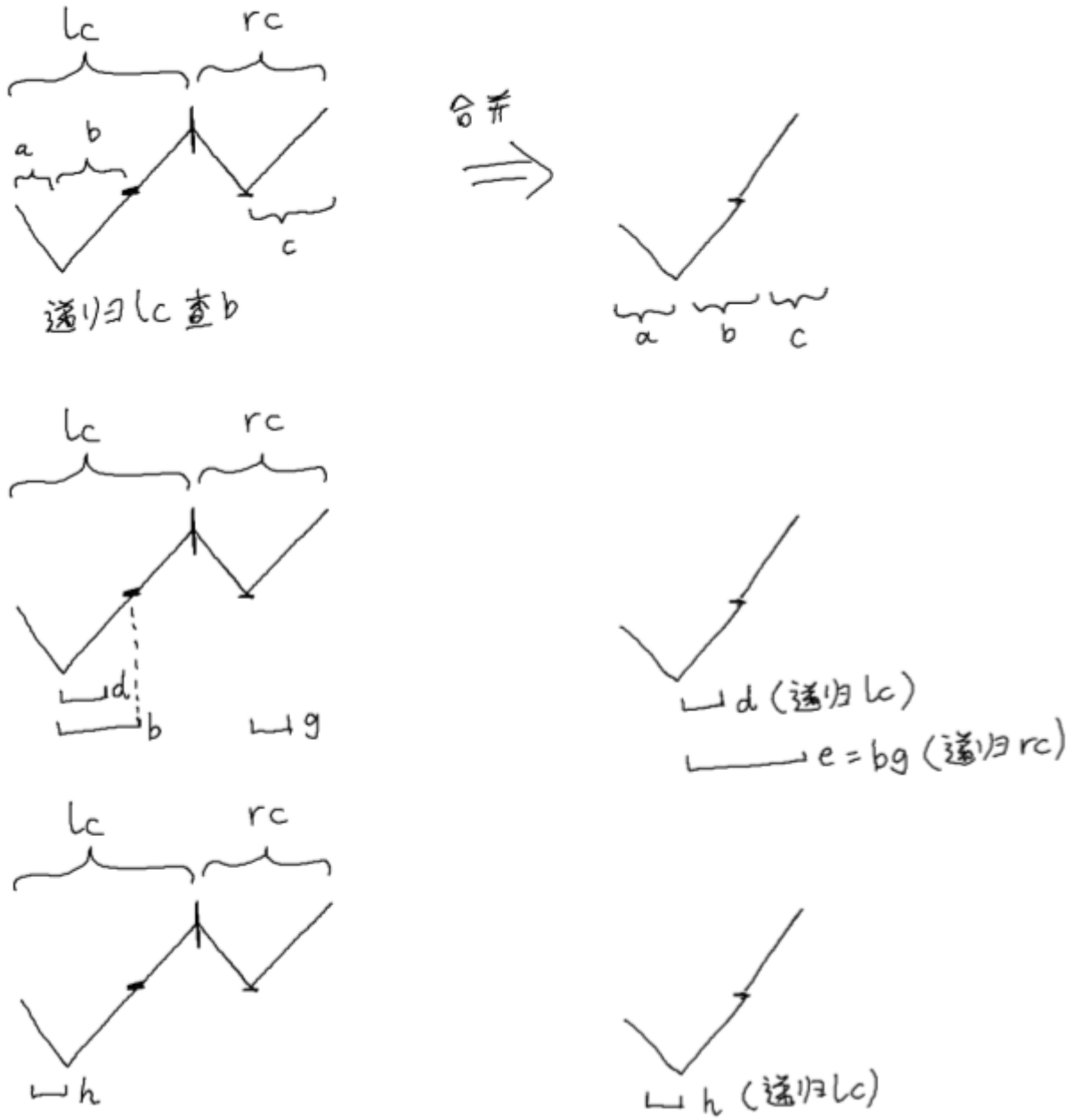
题解 By nzhtl1477

题目中的 `max` 和 `nand` 是为了让信息的性质更少，只保留半群的性质。

考虑使用一棵 leafy 的平衡树维护这个序列，每个叶子对应一个括号，每个节点维护子树对应的不匹配括号，不匹配括号一定形如 `"))))((((("`，上传信息时需要递归查询。

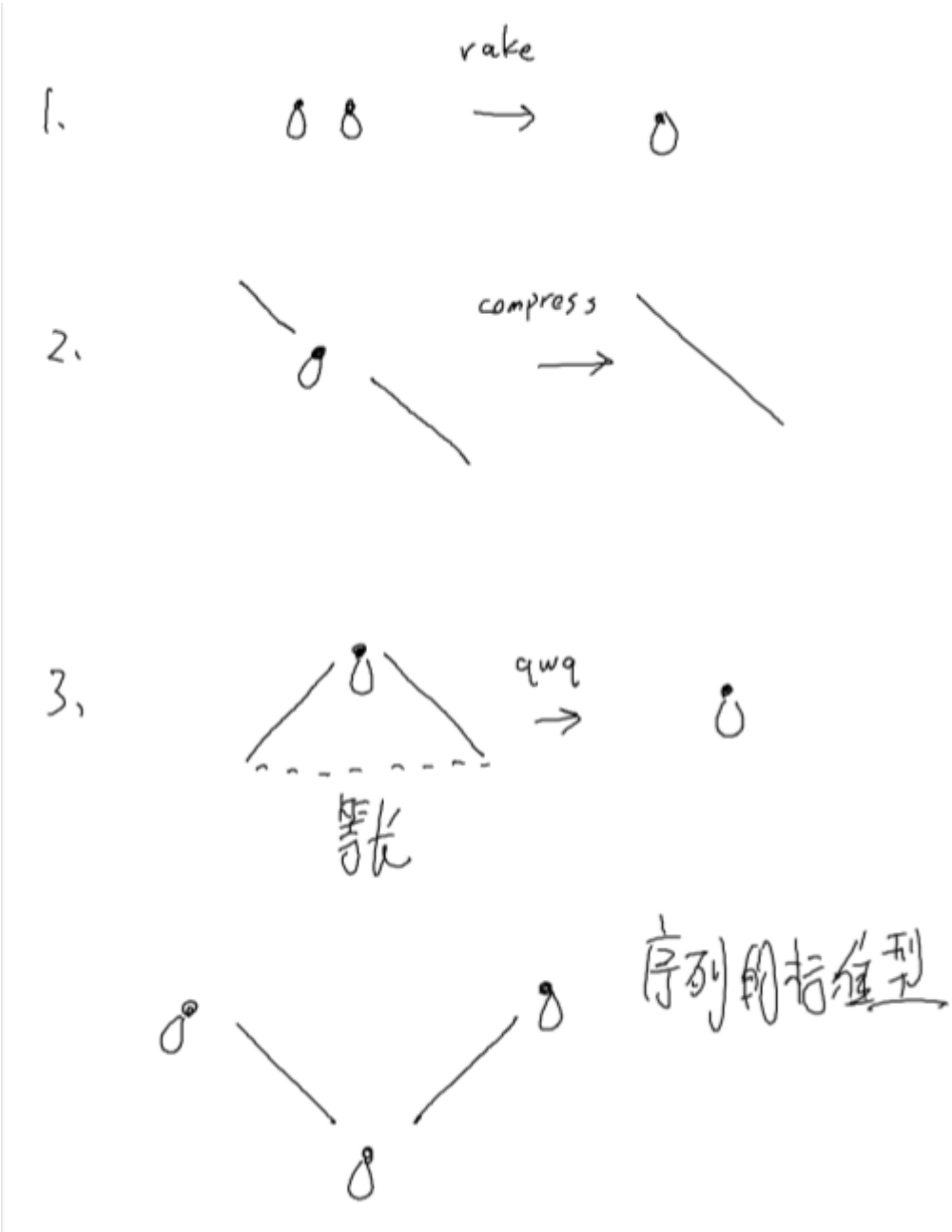
可以发现每次合并两个节点的时候是两个 `"))))((((("` 形式的信息合并，如下图所示我们考虑每种可能的合并方式，发现在合并过程中如果我们记下中间结果，就可以保证单向递归。



由于是单向递归的合并，所以合并两个节点信息的复杂度为 $O(\log n)$。

建树复杂度为 $O(n \log n)$，每个操作会访问到 $O(\log n)$ 个节点，总复杂度为 $O(\log^2 n)$。

总时间复杂度 $O(n\log n+m\log^2 n)$，总空间复杂度 $O(n)$。



std设计了一种在序列上的类top tree的分治结构，如上图所示。

cluster有三种：

1. 一个合法的括号序列（即没有不匹配括号）
2. 一段左括号（相邻两个左括号之间可以插入一个合法的括号序列）
3. 一段右括号（相邻两个右括号之间可以插入一个合法的括号序列）

最终序列会合并为一个可以简单计算结果的标准形式。

由于其中一个操作的势能不太对，所以并不能证明这个方法是 $O(\log n)$ 的，因为这样的方法在部分情况下也需要单侧的递归，所以只能证明一个 $O(\log^2 n)$ 的上界，并且各种观测发现其行为的确类似于 $O(\log^2 n)$。

感觉从这个设计上可以感受到这类问题的在线做法比较难优化到 $O(\log n)$。

总时间复杂度 $O(n\log n+m\log^2 n)$，总空间复杂度 $O(n)$。