leetcode刷题优化思路总结

2018-07-07　Leetcode

1. O（N）到 O(logN)

（1）数组、矩阵类的题，暴力求解是每个元素依次访问一遍，之后，在看题目中是不是蕴含着元素之间有着有序的规律，如果有，那么进行二分搜索，这样就不再是线性访问元素了，而是可以跳过一部分不可能是答案的元素，从而达到优化效果。

潜在有序：（1）BST，永远要思考中序遍历下会不会更快

（2）前面全部是0，后面全部是1

（3）顺序不重要，意味着可以排序

1. 图类的问题：优化数据结构很重要，其中，选择什么样的数据作为结点很重要，有时候，数据看似多，其实数量有限，这个时候就果断的使用这些数据单独作为结点。
2. 从O(N^2)到O(N)的技巧：

（1）注意数据是不是整数类型，是的话就可以采用bit操作，这样整个解的空间可以降到O(32N)。

（2）如果是双层遍历，那么看看是不是可以使用双指针来跳过双重循环中的不需要计算的循环。

（3）如果题目是字符，那么考虑是不是26次循环就能解决问题，或者题目是整数，那么考虑是不是32次循环可以解决。

（4）hash，例如在一个集合（元素是整数或者pair或者类）里面找出符合条件的两个数，此时将所有数存入hash，然后对一个元素，根据特征找出他的“补数”就可以了，而原来的暴力求解是O(N^2)。

（5）第二次循环计算的数可以之前进行O(N)的预处理之后变为O(1)的操作，例如要知道一个坐标左边最大（最小）的数。

（6）利用前一步的结果，例如String to Integer，不用每次都计算10^n，而是使用变量记录上一个10^n-1次方，这次只需要乘10就得出10^n。

1. 看似不可解到O(N^2):
   (1) 看看是不是确定了某些东西，其他的就可以生成，那么接下来只需要双重循环找出起点就可以了。
   (2) 如果同向双指针无法将问题复杂度降到O(mn)，那么考虑使用二维dp让复杂度降到O(mn)。

2. 明知道可以时间优化但是想不出来

（1）看是不是通过使用加空间的方法，例如增加hashmap，先把中间结果存起来，这样扫描一遍其他的信息时候查找只需要O(1)时间。

（2）看本身传入的数据结构是不是可以压缩，压缩后进行处理，降低处理的时间复杂度。例如压缩前是O(N^3)的复杂度，但是先只花费O(N)进行压缩到N1，那么复杂度变为O(N + N1^3）。

1. 从组合复杂度O(N^k)到O(N^3)或者O(N^2)，甚至O(N)

（1）组合选择的问题联想到背包问题从而降低复杂度到O(N^3)或者O(N^2)。

（2）使用优先级队列，把所有元素按顺序（有必要的话排序）存入优先级队列，队列满的时候，选择最不符合要求的元素，弹出去，最后队列中剩下的就是答案。

1. 随机数一个个处理从O(N)降到O(logN)：

使用树结构，看问题能不能转化成为子问题，子问题作为树的子节点。

1. n里面选k，从O(n^k)变成O(nlogk)

暴力解考虑是从n个里面任选K个，优化方法是只扫描一遍n，期间放入优先级队列，先放入K个，然后对新加入点，把堆顶不符合要求的去除，所以一遍下来就能选出符合要求的K个元素。

1. O(nlogn)变O(n)

（1）思考题目是不是实际类型的题目，实际类型题目参数其实是有上限的,例如身高，年龄。此时就可以考虑使用bucket，然后使用有序的hash函数把元素hash到相应位置，例如年龄23，那么就hash到23的位置，这样，我们可以对hash数组进行一些有序基础上处理，例如可以计算年龄介于23到30之间的人数，只需要在23的地方放年龄小于23的人数，年龄30的地方放年龄小于30的人数，相减即可。

（2）优先级队列的造成的nlogn，看能不能使用滑动窗口解决，滑动窗口每次落在窗口外的可以O(1)时间被丢弃掉，而不像优先级队列需要logn时间。

1. O(n)变O(logn * logn)

（1）题目往往是树，分治法，使得其中一棵子树时间复杂度降为1。

（2）特殊结构下，操作时间复杂度发生变化，例如完全二叉树求高度只需要花费logn时间。

1. O(n^2)进行剪枝

（1）单词两两比较的时候，可以把之前的元素存入trie树，这样当前元素可以一次性和之前所有元素进行比较，好处是之前如果单词含有相同前缀，那么就更快。

（2）同向双指针，左右指针在不符合要求的情况下跳转

1. 必要联想

（1）凡是对称结构，那么想到是否是区间，是否可以用递归上下界来解。

（2）凡是BST，一定想到inorder来解，inorder过程中定义上下界来解。

（3）凡是二分搜索，首先想到是先要排好序的。

1. 逆向思维

（1）有向图，方向逆过来考虑，起点和终点互换考虑。

（2）删除也可以理解为从其中选出需要的，也就变为了组合问题，如果是一个个的删除，那么也可以理解为子集问题。

Comments

Write a Comment