4 寻找两个正序数组的中位数

2020年6月13日 9:53

4. 寻找两个正序数组的中位数

难度 困难 2751 收藏 分享 切换为英文 关注 反馈

给定两个大小为 m 和 n 的正序（从小到大）数组 `nums1` 和 `nums2`。

请你找出这两个正序数组的中位数，并且要求算法的时间复杂度为 O(log(m + n))。

你可以假设 `nums1` 和 `nums2` 不会同时为空。

示例 1:

```
nums1 = [1, 3]
nums2 = [2]

则中位数是 2.0
```

示例 2:

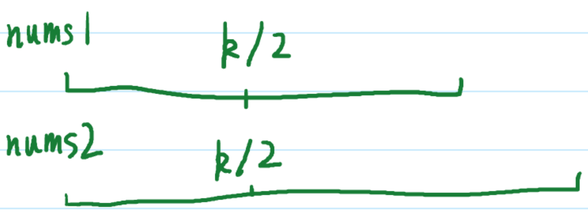
```
nums1 = [1, 2]
nums2 = [3, 4]

则中位数是 (2 + 3)/2 = 2.5
```

```
class Solution{
public:
    double findMedianSortedArrays(vector<int>& nums1, vector<int>& nums2) {
        int total = nums1.size() + nums2.size();
        if (total % 2 == 0) {                                   //当两个数组元素个数和为偶数时
            int left = find(nums1, 0, nums2, 0, total/2);
            int right = find(nums1, 0, nums2, 0, total/2+1);
            return (left + right) / 2.0;                        //中位数为中间两个元素的平均值，注意返回值是double，所以要除以2.0
        } else {
            return find(nums1, 0, nums2, 0, total/2+1);         //如果元素总和为奇数，则直接返回中间的数
        }
    }
    int find(vector<int>& nums1, int i, vector<int>& nums2, int j, int k) {  //i,j是两个数组的起始位置
        if (nums1.size() - i > nums2.size() - j) {              //我们假设 nums1 是元素个数较少的一个数组
            return find(nums2, j, nums1, i, k);                 //所以如果 nums1 元素个数大于 nums2 元素个数，就需要把两个数组对调一下
        }
        if (k == 1) {                                           //如果要找的数是第一个数(即最小的数)，需要特判一下（这里其实是递归边界）
            if(nums1.size() == i) {                             //因为 nums1 较短，所以有可能为空
                return nums2[j];                                //若为空就返回 nums2 的第一个数
            } else {
                return min(nums1[i], nums2[j]);                 //否则返回两个数组的第一个数中较小的那个
            }
        }
        if (nums1.size() == i) {                                //特判，如果 nums1 为空
            return nums2[j+k-1];                                //则返回 nums2 的第 k 个数就好了
        }
        int si = min(int(nums1.size()), i+k/2), sj = j+k-k/2;   // si, sj 是两个数组的第 k/2 个数
        if (nums1[si-1] > nums2[sj-1]){
            return find(nums1, i, nums2, sj, k-(sj-j));
        } else{
            return find(nums1, si, nums2, j, k-(si-i));
        }
    }
};
```

这里要取 min 是因为 i+k/2 有可能越界，nums1.size() 的返回值为 size_t，所以要强制类型转换为 int

砍掉 nums2 的前面 $k-\frac{k}{2}$ 个元素，
所以 nums2 从 sj 开始，返回新的两个数组的第 k-(sj-j) 个元素

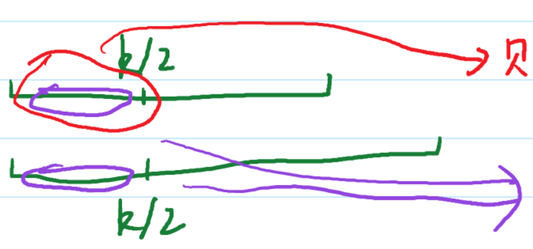


假设 nums1.size() 为 n，nums2.size() 为 m
要找的中位数 $k=\frac{n+m}{2}$

我们先判判 nums1[k/2] 和 nums2[k/2] 的关系

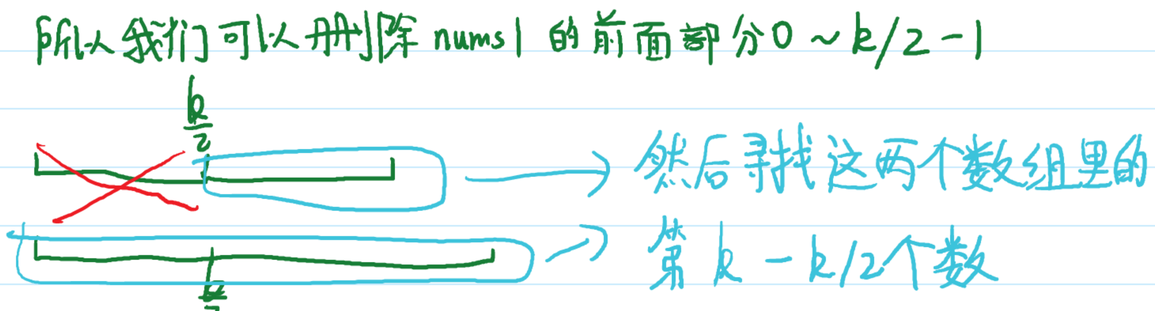
实际上这里是 $k-\frac{k}{2}$，k是偶数时，$k-\frac{k}{2}=k$
k是奇数时，$k-\frac{k}{2}$ 比 $\frac{k}{2}$ 大 1
这里为了讨论方便，先取 nums[k/2]

如果 nums1[k/2] > nums2[k/2]



则小于 nums1[k/2] 的元素个数一定小于 k
因为这两个部分相加为 k，nums1[k/2] 又较小

所以我们可以删除 nums1 的前面部分 0 ~ k/2-1



然后寻找这两个数组里的第 k-k/2 个数

nums1[k/2] > nums2[k/2] 时同理，删去 nums2 的前面部分

nums1[k/2] == nums2[k/2] 时就找到了第 k 个数（nums1[k/2] 或 nums2[k/2]）