

2014
07-12[求两个有序数组的中位数-算法导论](#)

coder

算法分析, 递归和分治

围观3250次

14条评论

Question

There are 2 sorted arrays A and B of size n each. Write an algorithm to find the median of the array obtained after merging the above 2 arrays(i.e. array of length 2n). The complexity should be $O(\log(n))$.

有两个排序的数组，长度都为n，求合并后的排序数组的中位数。

题目是《算法导论》上的一道习题，不过已多次出现在面试题当中。注意，此题中两个数组的长度是相等的。当然，长度不等的话也可以做，只是要多些判断条件。参考leetcode题目 [Median of Two Sorted Arrays](#)

方法1 直接遍历

直接的解法是遍历两个数组并计数，类似归并排序里面的有序数组的合并，复杂度为 $O(n)$ 。代码如下：

```
01 #include <iostream>
02 #include <stdio.h>
03 using namespace std;
04
05 double getMedian(int arr1[],int arr2[], int n){
06     int i=0,j=0; //分别是 arr1, arr2的当前下标
07     int m1=-1,m2=-1; //保存两个中位数. 由于是2n个, 肯定有两个中位数
08     for(int cnt=0; cnt<=n; cnt++){
09         if( i<n && (arr1[i] < arr2[j] || j >= n )){
10             m1 = m2;
11             m2 = arr1[i++];
12         }else{
13             m1 = m2;
14             m2 = arr2[j++];
15         }
16     }
17     return (m1+m2)/2.0;
18 }
19 int main()
20 {
21     int ar1[] = {1, 12, 15, 26, 38};
22     int ar2[] = {2, 13, 17, 30, 45};
23
24     int n1 = sizeof(ar1)/sizeof(ar1[0]);
25     int n2 = sizeof(ar2)/sizeof(ar2[0]);
26     if (n1 == n2)
27         printf("Median is %lf", getMedian(ar1, ar2, n1));
28     else
29         printf("Doesn't work for arrays of unequal size");
30     return 0;
31 }
```

方法2 分治法

要求的复杂度为 $O(\log(m+n))$ ，很显然需要用分治法求解。

假设数组A的中位数为m1，数组B为m2，例如：

ar1[] = {1, 12, 15, 26, 38}

ar2[] = {2, 13, 17, 30, 45}

m1 = 15，m2 = 17。由于m1 < m2，则可以确定中位数即为下面两个子数组的中位数：

[15, 26, 38] 和 [2, 13, 17]

重复这个步骤，可以得到 m1 = 26 m2 = 13. 得到两个子数组：

[15, 26] 和 [13, 17]

这时，由于n=2，无法在继续分下去了。可以直接计算得：

U B E R
优步注册优步
顺风车

上班顺路赚油费 >

标签

A*算法 BFS C++ DFS Dijkstra Google
HOJ Java KMP LeetCode UVA 二分图
二叉树 优先队列 位运算 凸包问题 分治
动态规划 博弈论 卡特兰数 后缀数组 回溯
图 大数据 字典树 字符串 并查集 微软
快速幂 拓扑排序 排序 数论 最小生成树
最短路径 栈 概率 模拟 模拟退火 母函数
状态压缩 百度 算法讲解 线段树 组合数学
编程大赛 网络流 考研机试 背包问题
计算几何 记忆化搜索 贪心 递推 遗传算法
阿里 随机算法 面试题 高精度

```

1 median = (max(ar1[0], ar2[0]) + min(ar1[1], ar2[1]))/2
2           = (max(15, 13) + min(26, 17))/2
3           = (15 + 17)/2
4           = 16

```

代码如下：

```

01 int median(int arr[], int n)
02 {
03     if (n%2 == 0)
04         return (arr[n/2] + arr[n/2-1])/2;
05     else
06         return arr[n/2];
07 }
08
09 int getMedian(int ar1[], int ar2[], int n) {
10     int m1;
11     int m2;
12     if (n <= 0)
13         return -1;
14     if (n == 1)
15         return (ar1[0] + ar2[0]) / 2;
16
17     if (n == 2)
18         return (max(ar1[0], ar2[0]) + min(ar1[1], ar2[1])) / 2;
19
20     m1 = median(ar1, n);
21     m2 = median(ar2, n);
22     /* 相等可直接返回 */
23     if (m1 == m2)
24         return m1;
25     if (m1 < m2) {
26         if (n % 2 == 0)
27             return getMedian(ar1 + n/2-1, ar2, n/2 + 1);
28         else
29             return getMedian(ar1 + n/2, ar2, n/2+1);
30     } else {
31         if (n % 2 == 0)
32             return getMedian(ar2 + n/2-1, ar1, n/2+1);
33         else
34             return getMedian(ar2 + n/2, ar1, n/2+1);
35     }
36 }
37
38 int main()
39 {
40     int ar1[] = {1, 12, 10, 26, 38};
41     int ar2[] = {2, 13, 17, 30, 45};
42
43     int n1 = sizeof(ar1)/sizeof(ar1[0]);
44     int n2 = sizeof(ar2)/sizeof(ar2[0]);
45     if (n1 == n2)
46         printf("Median is %d", getMedian(ar1, ar2, n1));
47     else
48         printf("Doesn't work for arrays of unequal size");
49     return 0;
50 }

```

时间复杂度为 $O(\log n)$ 。

参考：<http://www.geeksforgeeks.org/median-of-two-sorted-arrays/>

坚持 1 个月，看美剧不用字幕

vipabc 专业英美系外教，免费试听课程 每天45分钟，忙里偷闲学英语！



分治, 面试题

← 数组滑动窗口中的最大值[单调队列]

寻找缺失的数字 →

您可能还会对这些文章感兴趣！

- HDU 3244-Inviting Friends-背包问题-[解题报告]H ▸ HDU 4260-The End of The World-动态规划-[解
- HDU 4742-Pinball Game 3D-分治-[解题报告]HOJ ▸ C++ 优先级队列 (priority_queue)
- HDU 3829-Cat VS Dog-图-[解题报告]HOJ ▸ HDU 2832-Snail' s trouble-分治-[解题报告]HOJ
- hdu 1960 Taxi Cab Scheme-二分图-[解题报告]C++ ▸ 回溯法 (2)

社交帐号登录: 微信 微博 QQ 人人 更多»

JD.C

¥ 4999.00

以实现卓越

不相信吗

节省





说点什么吧...

发布

14条评论

最新 最早 最热



j lucky

失误失误，转载错了。已经改正，抱歉。

2015年7月6日 回复 顶 转发



WalkCoder

第二块代码if(it != mp.end())应改为if(it != mp.end() && (i+1)!=(it->second +1))；因为第二种解法如果数组有重复元素 就不正确

2015年6月11日 回复 顶 转发



wynk0804

正解，应该是200！

2015年5月2日 回复 顶 转发



zclzcllove

很好的思路，可以借鉴

2015年3月10日 回复 顶 转发



Lint

第一句可以忽略不计了吧。从第二句开始分析，说明这个花色下的所有牌都会在其他里面出现，那么还剩下♣和♦。第三句，可以排除2和7，因为在两种花色里有。现在是第四句，因为♣还剩下多个，只有是♦B才能知道答案。

2015年3月5日 回复 顶 转发



magichu

很好。代码比书上的简练！

2015年3月4日 回复 顶 转发



cherry_odd

补充：此算法求出的不一定是最优解，之前理解错了

2015年2月22日 回复 顶 转发



la0bei

递归实现的代码部分

14句和18句的 wt[n - 1]是不是应该改为 wt[n]？

2014年12月31日 回复 顶 转发



reking

```
#include <stdio>
#include <algorithm>
```

```
struct LWPair{
    int l,w;
};

int main() {
    //freopen("input.txt","r",stdin);
    const int MAXSIZE=5000, MAXVAL=10000;
    LWPair sticks[MAXSIZE];
    int store[MAXSIZE];
    int ncase, nstick, length,width, tmp, time, i,j;
    if(scanf("%d",&ncase)!=1) return -1;
    while(ncase-- && scanf("%d",&nstick)==1) {
        for(i=0;i<nstick;++i) scanf("%d",&sticks[i].l,&sticks[i].w);
        std::sort(sticks,sticks+nstick,[](const LWPair &lhs, const LWPair &rhs) { return lhs.l>rhs.l ||
        lhs.l==rhs.l && lhs.w>rhs.w; });
        for(time=-1,i=0;i<nstick;++i) {
            tmp=sticks[i].w;
            for(j=time;j>=0 && store[j]>=tmp;--j); // search from right to left
            if(j==time) { store[++time]=tmp; }
            else { store[j+1]=tmp; }
        }
        printf("%dn",time+1);
    }
    return 0;
}
```



158123

理解错了。。。。。谢谢楼主~~

2014年8月14日 [回复](#) [顶](#) [转发](#)



158123

有谁知道直接遍历的方法中第8行是什么意思嘛？

2014年8月12日 [回复](#) [顶](#) [转发](#)



光速小子

总数为 $2n$ ，遍历到第 n 个，就是中间的位置

2014年8月12日 [回复](#) [顶](#) [转发](#)



1173954900

这是往年的笔试题，非在线，主要考察基础，Java会在面试时间问到

2014年8月10日 [回复](#) [顶](#) [转发](#)



Ding Qiangyuan

您没有考虑 树的根节点是负数的情况，若树的根节点是个很大的负数，那么就要考虑过不过另外一边子树了

2014年8月6日 [回复](#) [顶](#) [转发](#)