实验一 用分治法求解数组的中位数和最大子集

一、实验原理

1. 分治法的概念

分治算法的基本思想是将一个规模为 N 的问题,分解成 K 个规模较小的子问题,这些子问题相互独立且月原问题性质相同。求解出子问题的解,合并得到原问题的解。分治法在每一层递归上有 3 个基本步骤。

（1）分解:将原问题逐层分解为若干个较小规模,相互独立且与原问题表示形式相同的子问题;

（2）求解:若分解后的子问题规模较小且求解容易则直接求解,否则将相应的子问题再不断分解为更小的子问题,直到容易求解为止;

（3）合并:将已经求解出的各子问题的解逐步合并,最后得到原问题的解。

2. 适用分治策略解决的问题

对于一个输入规模为 n 并且取值比较大的问题,若能满足下面条件,使用分治法的思想就可以提高解决问题的效率。

（1）保证这 n 个数据能分解为 k 个不同的子集合,同时得到的 k 个子集合为可以独立求解的子问题（$1 \le k \le n$）;

（2）分解后的各子问题和原问题具有相似的表示形式,便于使用循环或递归机制;

（3）得到这些子问题的解之后,可以方便递推出原问题的解。

3. 分治算法框架

分治法的一般的算法设计模式如下:

$Divide-and-Conquer(p)$

{ {if(| p |<= n0)

{解子问题；

return(子问题的解)；}

for(i = 1; i <= k; i = i + 1)

{分解原问题为更小的子问题 Pi：

yi = Divide-and-conquer(| Pi |);}

T = MERGE(y1, y2, ⋯, yk);

return(T);

}

其中$|p|$为问题 p 的规模大小;$n0$是闽值,当问题 p 的规模$|p|$小于$n0$时,问题比较容易直接求解,不需要再继续分解。

4. 分治法的分割原则

将原问题分割为规模大致相等的子问题的分割方法是基于子问题平衡思想的,在大多数情况下,这种方法总是优于子问题规模不等的分割方法。分治法将规模为n的问题分割成k个规模为n/m的子问题去求解。

用递归方程的分析方法来分析分治法的计算效率：

$$T(n)=\begin{cases} o(1) & n=1 \\ kT(n/m)+f(n) & n>1 \end{cases}$$

假设将原问题分解为 k 个子问题及将 k 个子问题的解合并为原问题的解均需耗费$f(n)$个单位时间。$T(n)$表示该分治法中解规模为n的问题所需的计算时间。

$$T(n)=n^{\log_m^k}+\sum_{j=0}^{\log_m^{n-1}}k^j f(n/m^j)$$

二、实验要求

完成下面两段来自 LeetCode 上的分治法设计，并分析其时间复杂度的公式，是否满足要求？为什么？

Pro1：Median of Two Sorted Arrays

There are two sorted arrays nums1 and nums2 of size m and n respectively.Find the median of the two sorted arrays. The overall run time complexity should be O(log (m+n)).

Pro2：Maximum Subarray

Find the contiguous subarray within an array (containing at least one number) which has the largest sum.

For example, given the array [-2,1,-3,4,-1,2,1,-5,4],

the contiguous subarray [4,-1,2,1] has the largest sum = 6.

注意：当节课实验结束前找老师检查，不需提交实验报告