

算法分析与设计 II

2022-2023-2

数学与计算机学院
数据科学与大数据技术

LAST MODIFIED: 2023.1.16



2. 基础算法

2.1 枚举法

枚举法

也叫穷举法, 是一种暴力搜索的方法, 特点是将给定条件的所有情况都进行计算, 直到找到符合要求的解, 随着参与计算的参数的增加, 算法复杂度可能成指数级增加, 所以只有当所有情况的总数在一个较小的范围时才能进行

- 枚举法的优点就是建模简单, 几乎不用考虑任何算法。在计算过程中, 排除一些不可能的情况, 能够适当减少一定的计算量
- 程序设计中一些排序算法(选择排序、冒泡排序、插入排序)、查找算法(顺序查找、二叉树的遍历等) 都是枚举法的具体应用

■ Description

- ▶ ...Each block is a cube, 1 inch by 1 inch by 1 inch. Donald wants to stack the blocks together into a rectangular solid and wrap them all up in brown paper for shipping. How much brown paper does Donald need?

■ Input

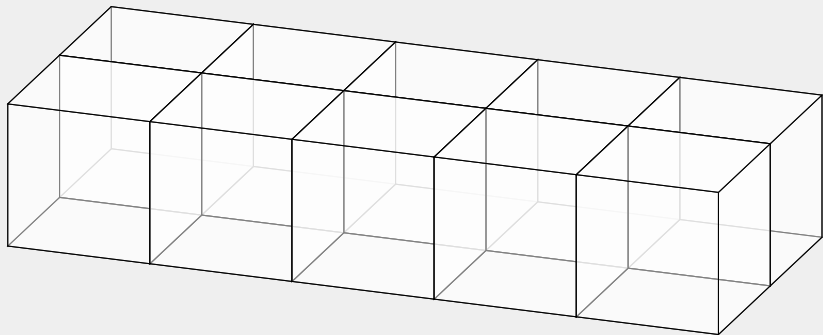
- ▶ The first line of input contains C , the number of test cases. For each case there is an additional line containing N , the number of blocks to be shipped. N does not exceed 1000.

■ Output

- ▶ Your program should produce one line of output per case, giving the minimal area of paper (in square inches) needed to wrap the blocks when they are stacked together.

Sample

- Sample Input: 10
- Sample Output: 34



2.2 递归法

■ 递归法思路:

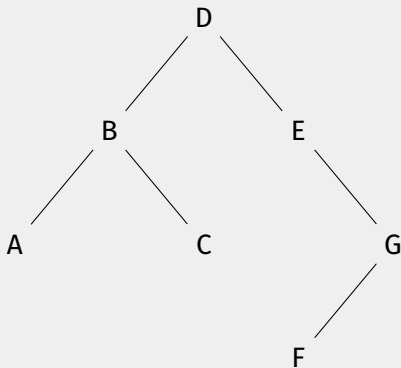
1. 原问题分解为一个或多个规模更小、但具有类似于原问题特性的子问题
2. 确定无须分解、可直接求解的最小子问题（递归的终止条件）

■ 递归的两个基本要素是:

1. 递归关系式：确定递归的方式，即原问题是如何分解为子问题的
2. 递归出口：确定递归到何时终止，即递归的终止（结束、边界）条件

■ 但是，递归方法并不能降低程序的时间复杂度，而且，递归时函数的嵌套调用使用系统堆栈，控制不好会导致堆栈溢出 (stack overflow)

- 已知二叉树
 - ▶ 先序遍历为
DBACEGF
 - ▶ 中序遍历为
ABCDEFG
- 求后序遍历
 - ▶ ACBFGED



1731 – Orders (poj.org)

- 给出字母序列，将这个序列中字母组成的所有可能排列/置换按字典序输出

Sample

- Sample Input: bbjd
- Sample Output: bbdj bbjd bdbj bdjb bjbd bjdb dbbj dbjb djbb jbbd jbdb jdbb

- 实现：
 1. 递归（手写）
 2. C++ STL 中的 `next_permutation`¹函数

¹https://cplusplus.com/reference/algorithm/next_permutation/

2.3 分治法

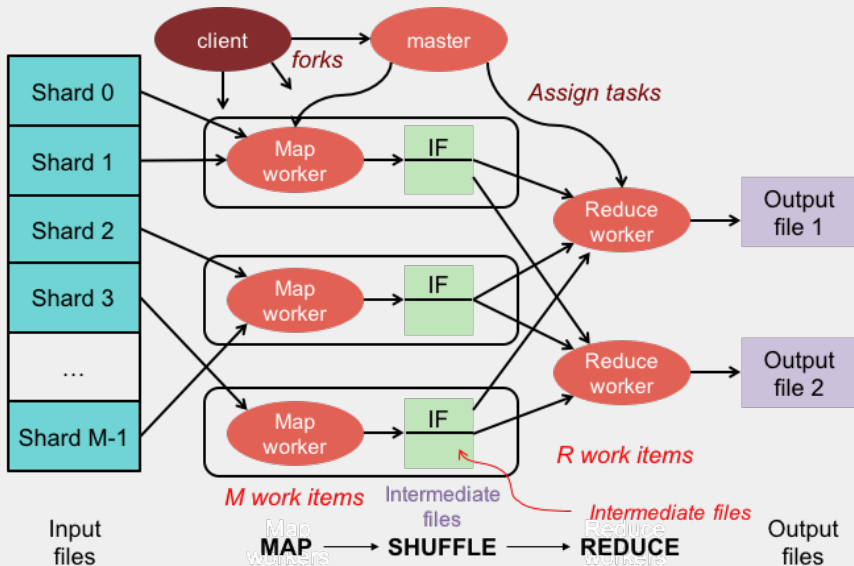
分治法

将原问题分解成规模较小但是与原问题类似的子问题，通过递归来求解这些子问题，再将子问题的解合并来求出原问题的解的方法

■ 分治在递归的过程中有三步：

1. 分解：将原问题分解成规模较小但是与问题类似的子问题
2. 求解：递归求解子问题，子问题规模足够小时直接求解
3. 合并：将子问题的解合并，求出原问题的解

MapReduce



2388 – Who's in the Middle (poj.org)

■ 求 n 个数的中位数， n 为奇数

■ 分析

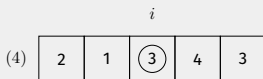
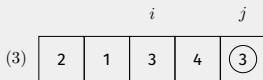
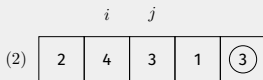
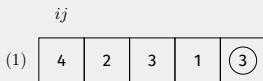
1. 如果 n 个数是有序的，那么第 $\frac{n+1}{2}$ 个数就是所求的中位数
2. 在 C 语言提供了排序函数 `qsort`²，C++ 语言提供了 `sort`³，它们的实现都是基于快速排序
3. **快速排序**(Quicksort)，又称**划分交换排序**(partition-exchange sort)，简称**快排**

²<https://cplusplus.com/reference/cstdlib/qsort/>

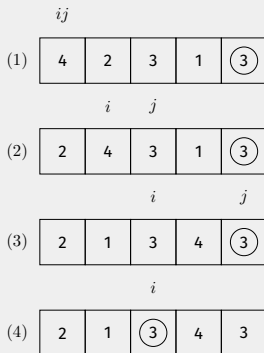
³<https://cplusplus.com/reference/algorithm/sort/>

快速排序

(1) 选取数组 **a** 最右边元素作为基准值 **x**，两个指针 **i** 和 **j** 最初都在序列最左边

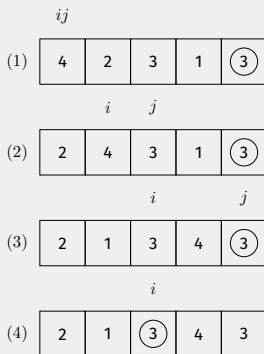


快速排序



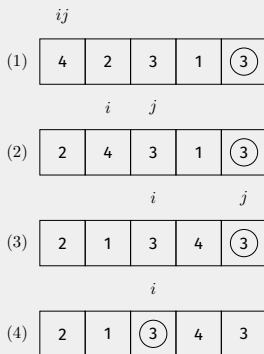
- (1) 选取数组 a 最右边元素作为基准值 x ，两个指针 i 和 j 最初都在序列最左边
- (2) 用 $a[j]$ 和 x 的值进行比较，如果 $a[j]$ 小于 x ，则交换 $a[i]$ 和 $a[j]$ ，同时将 i 和 j 向右移动；如果 $a[j]$ 大于等于 x ，则 j 右移， i 不动

快速排序



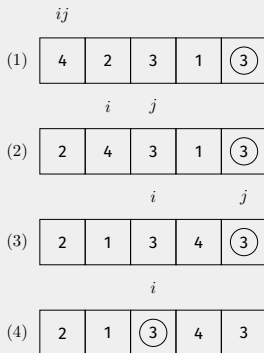
- (1) 选取数组 a 最右边元素作为基准值 x ，两个指针 i 和 j 最初都在序列最左边
- (2) 用 $a[j]$ 和 x 的值进行比较，如果 $a[j]$ 小于 x ，则交换 $a[i]$ 和 $a[j]$ ，同时将 i 和 j 向右移动；如果 $a[j]$ 大于等于 x ，则 j 右移， i 不动
- (3) 当 j 移动到最右边时， i 的左侧元素都小于 x ，右侧都大于等于 x

快速排序



- (1) 选取数组 a 最右边元素作为基准值 x ，两个指针 i 和 j 最初都在序列最左边
- (2) 用 $a[j]$ 和 x 的值进行比较，如果 $a[j]$ 小于 x ，则交换 $a[i]$ 和 $a[j]$ ，同时将 i 和 j 向右移动；如果 $a[j]$ 大于等于 x ，则 j 右移， i 不动
- (3) 当 j 移动到最右边时， i 的左侧元素都小于 x ，右侧都大于等于 x
- (4) 最后将 x 和 $a[i]$ 值交换。序列就被 i 分成了左右两部分

快速排序



- (1) 选取数组 a 最右边元素作为基准值 x ，两个指针 i 和 j 最初都在序列最左边
- (2) 用 $a[j]$ 和 x 的值进行比较，如果 $a[j]$ 小于 x ，则交换 $a[i]$ 和 $a[j]$ ，同时将 i 和 j 向右移动；如果 $a[j]$ 大于等于 x ，则 j 右移， i 不动
- (3) 当 j 移动到最右边时， i 的左侧元素都小于 x ，右侧都大于等于 x
- (4) 最后将 x 和 $a[i]$ 值交换。序列就被 i 分成了左右两部分

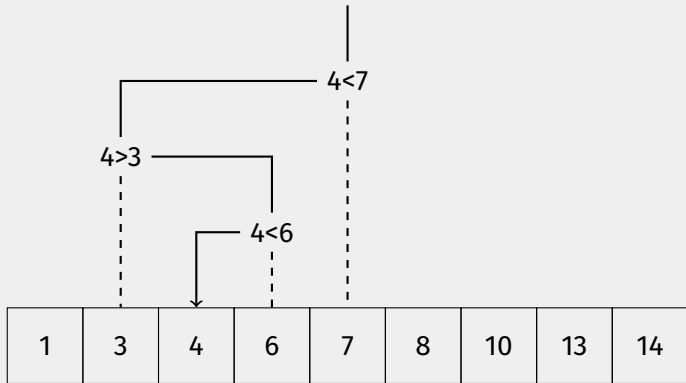
交换过程结束后，两个 3 改变了之前的顺序，这种情况称为排序的不稳定性，所以这种快速排序算法是**不稳定**的排序算法

- 通过交换相邻元素进行排序，对于一个给定的无序序列 $a[1 \cdots n]$ ，计算最终完成升序排列需要交换的次数

分析

1. **冒泡排序**，所求的交换次数也就是计算无序序列的**逆序对**数目，冒泡排序的时间复杂度是 $O(n^2)$
2. 求逆序对数目效率更高的方法是利用**归并排序**，可以做到 $O(n \log n)$ 的时间复杂度
3. 归并的具体方法是将排好序的两个数组的元素依次比较，将其中较小的依次放入一个辅助数组 **b** 中，因为两个数组已经排序，比较的过程直接从左边到最右边即可。最后用 **b** 替换原来的 **a**，进行上一层的合并操作

二分查找

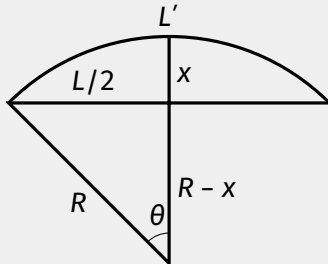


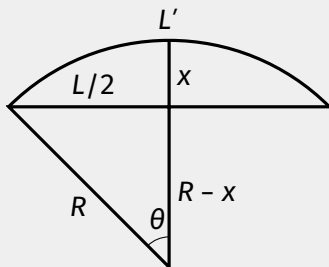
1905 – Expanding Rods (poj.org)

- 细棒长度 L ，两端固定，加热后膨胀为一段圆弧，长度变为

$$L' = (1 + n \cdot C) \cdot L$$

n 为变化的温度， C 为膨胀系数，求细棒中心移动的距离



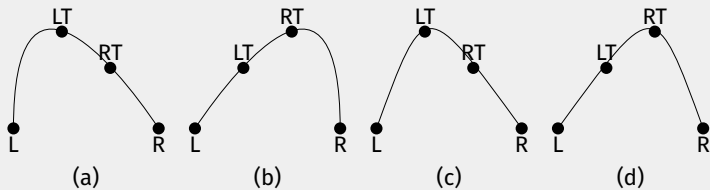


$$R^2 = (R - x)^2 + (L/2)^2 \Rightarrow R = \frac{x^2 + (L/2)^2}{2x}$$

$$\sin \theta = \frac{L/2}{R} \Rightarrow \theta = \sin^{-1}\left(\frac{L/2}{R}\right)$$

代入 $L' = 2R\theta$ 可得 x 和 L' 的关系

三分查找



- 由 (a)(c) 可以看出, $f(LT)$ 大于 $f(RT)$ 时, 可以将右侧区间删掉
- 由 (b)(d) 可以看出, $f(LT)$ 小于 $f(RT)$ 时, 可以将左侧区间删掉

2420 – A Star not a Tree? (poj.org)

■ 给出平面上 n 个点的坐标，找到一点，到这些点的距离和最小

■ 分析

1. 所求的就是几何学中的费马点，显然几何方法不适用于计算机，在空间范围内搜索该点，可以采用分治策略进行，不断缩小求解空间，直到找到符合精度要求的解
2. 本题的坐标空间 x 和 y 都在 0 到 10000 之间，随着 x 和 y 的增加，解并不是单调变化，无法用二分搜索来实现。问题区间不满足单调性，但是符合凸函数，的性质，求区间最值可以采用三分的方法