**两种比较领先的排序思维对比**

罗瑶光手稿

今天我有阅读了一种这几年比较流行的快速排序思想，基于中轴分离进行左右分配的数列递归排序法。地址：github 上的 dongxingrong。 我仔细的研究了下，发现有几个地方可取，于是进行评价：

|  |
| --- |
|  |

public int getMiddle(Integer[] list, int low, int high) {

int tmp = list[low]; //数组的第一个作为中轴

while (low < high) {

while (low < high && list[high] >= tmp) {

high--;

}

list[low] = list[high]; //比中轴小的记录移到低端

while (low < high && list[low] <= tmp) {

low++;

}

list[high] = list[low]; //比中轴大的记录移到高端

}

list[low] = tmp; //中轴记录到尾

return low; //返回中轴的位置

}

我思考了很久，在同频算子减少的方向中，直接减少temp的 swap变量是一种先进的思想。但是利弊交错，我进行了和我的4代小高峰过滤快排做了详细对比：

|  |  |  |
| --- | --- | --- |
| **1000万数列排序** | 瑶光高峰过滤 | xingrong中轴分配 |
| 算子减少 | 有 | 有 |
| 条件减少 | 有 | 有 |
| 离散效率 | 高 | 中 |
| 条件过滤 | 有 | 无 |
| 高频降解 | 无 | 有 |
| 计算性能 | 高 | 高 |
| 代码缩进 | 有 | 有 |
| 平均高峰先排耗时 | 1137毫秒 | 1186毫秒 |
| 平均分配先排耗时 | 1072毫秒 | 1178毫秒 |

**注解：这里的函数效率是指单个函数运行的计算消耗，包括堆栈消耗，内存阻塞，计算耗时等因素。**

**计算性能指的是数组的最大长度，单位消耗的时间，内存大小。**

比较可以发现，因为递归的小高峰的存在，中轴心分配所需要的位移变换成为了必要执行功能，增大了开销，这种开销可以最大限度的达到balanced递归逻辑的层数，但是对于指数方的多维算子条件缺少了过滤层，所以，性能耗损大。这种排序思想是完美的，为追求完美，丢失了一些东西，比较可惜。但是这种思想需要进行解析，在很多算法应用领域有宝贵的价值。

下面的4张图是堆1000万个随机数进行排序的比较耗时测试。







