# 关于对排序的自创方案

## ——平衡分治思想及切换排序的应用

### 202330453151 于博宇 计科一班

### 目录

### 摘要 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 2

### 关键词 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 2

### 核心思想 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 2

### 研究历程 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 2

### 关键代码截图 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3

### 阈值的选取 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 4

### 优劣性分析 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 5

### 拓展研究 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 6

### 参考文献 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 6

### 摘要

随着大数据时代的到来，数据处理和分析的需求日益增长。排序算法作为数据处理的基础，其效率和性能直接影响到整个数据处理流程的效率。传统的排序算法如快速排序、归并排序等在大数据处理中表现出色，但在某些特定场景下，仍然存在优化空间。本文提出了一种新的排序算法——“二分数组——快速排序算法”，该算法结合了分治法和冒泡排序的思想，通过将原始数组分成小数组和大数组，并在递归过程中动态调整数组长度，最终在数组大小小于特定阈值时切换到冒泡排序。实验结果表明，该算法在处理大数据时具有较好的性能和稳定性。

**关键词：**排序算法优化；大数据处理；分治法

**核心思想：**

“二分数组——快速排序算法”的核心思想是将原始数组分成两个部分：小数组和大数组。初始时，选择数组中的两个元素作为分界点，大数插入到大数组的最小分界位；小数插入到小数组中的最大分界位。对于后续的每个元素，根据其与2个分界点的关系，插入到相应的数组中。通过这种方法，我们可以**干净利落的把原始数组分成大数及小数两份**。

具体的后续插入方案是：如果元素位于两个分界点之间，则根据小数组和大数组的插入次数，选择插入次数较少的数组，并更新相应的分界点。通过这种方式，算法能够在递归过程中动态调整数组大小，尽量实现有效的二等分，这也是我所谓的**平衡分区思想**。最终在数组大小小于特定阈值时切换到快速排序，以防止递归深度过大导致时间复杂度过高。

**研究历程：**

在数据结构上课时，在老师提出如何有效按照成绩高低分开卷纸时突发奇想：可不可以通过两张试卷分数作为基准点来进行全班的排序？在此时此算法仅停留在分治法的层面。

课后，我又想到可能因为基准点过于极端导致出现分治效果不好的情况，于是一开始想法是隔一个插一个，一边分一个，但后续想到不如用两个t来记录插入次数，有在两个分界点之间的值就插到目前实时较小的那个数组里面。这是我平衡分区的思路历程。

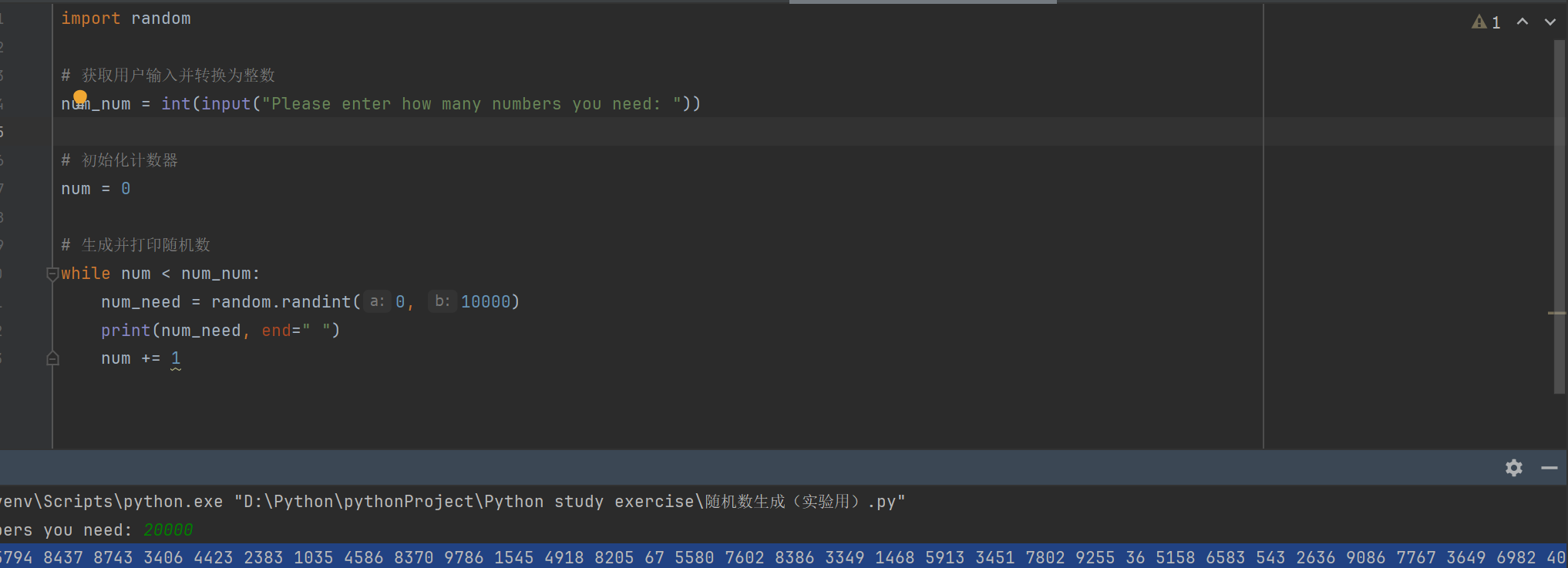
紧接着，我又想到似乎不断的递归在后期效果不如快速排序效率高，于是切换快速排序。

**关键代码截图：**

**算法部分：**

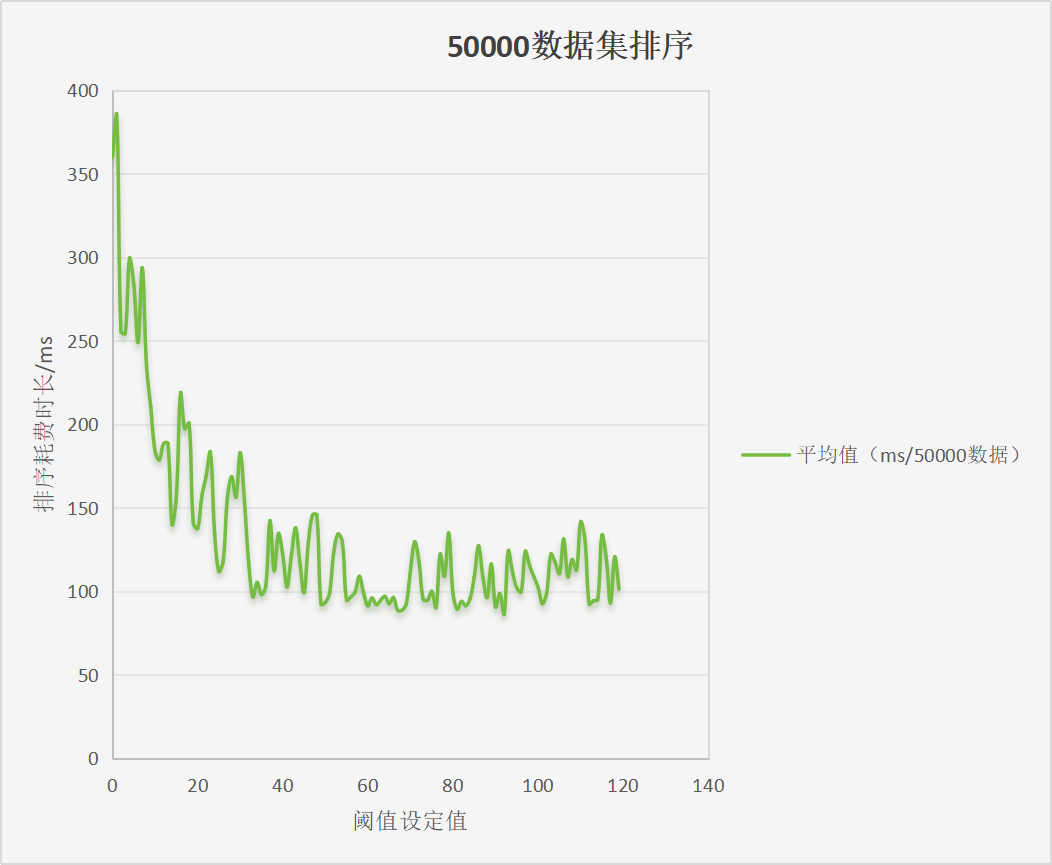


**生成大规模数据集部分：**



**阈值的选取：**

我对阈值进行了循环测试最佳阈值，其中二分数组—冒泡排序结果如下：



分析：测试多次取平均值绘制这个表格，发现阈值选大概46左右就可以，之所以呈现震荡我觉得是因为测试次数不够，测试次数足够多我觉得这个阈值改变应该在第一象限会呈现逼近类似双钩函数第一象限的图案，快速下降在46左右到达谷点，之后随着增加逐渐上升，直到逼近冒泡排序的耗时线。

下面是二分数组—快速排序结果：

分析：测试多次取平均值绘制这个表格，发现整个曲线呈逼近快速排序耗时，且比快速排序耗时明显稳定，后续由于阈值增大采用快速排序较多而不太稳定，最终权衡下来选择88作为最佳阈值。

**优劣势分析：**

**优势**

动态调整数组大小：

算法在递归过程中动态调整数组大小，避免了传统分治算法中数组大小固定的问题。这使得算法在处理不同分布的数据时更加灵活。

平衡性：

通过比较小数组和大数组的插入次数，算法能够尽量保持两个数组的平衡，从而提高排序效率。相比之下，快速排序在处理已经部分有序的数据时，可能会导致不平衡的分区，从而影响性能。

稳定性：

在数组大小小于特定阈值时，算法切换到 std::sort，确保了排序的稳定性。快速排序本身是不稳定的排序算法。

适应性：

算法在处理不同类型的数据分布时，可能会有更好的适应性。例如，在处理包含大量重复元素的数据时，可能会表现得更好，因为它可以根据插入次数动态调整数组大小

**劣势**

复杂度分析：该算法的时间复杂度较为复杂，取决于递归的深度和冒泡排序的使用频率。在某些情况下，算法的性能可能不如传统的快速排序或归并排序。

**拓展研究：**

在此基础上我尝试使用混合二分数组——快速排序算法，就是利用我的算法在排序同时平衡分区以及快速排序不平衡但效率高的二种优势尝试集合起来。横轴是每隔n次就进行一次二分数组排序，但我们从纵轴可以清楚的发现这种优化效果不理想。

**结论：**

“二分数组——快速排序算法”是一种结合了分治法和插入排序思想的新型排序算法。通过动态调整数组大小和在特定条件下切换到冒泡排序，该算法在处理大数据时表现出色。尽管该算法在复杂度分析和初始元素选择方面存在一定的挑战，但其独特的思想和优势使其在大数据处理领域具有广泛的应用前景。目前混合排序优化已经被证实是不可行的，未来的研究可以进一步优化初始元素选择策略，并探索该算法在不同数据分布下的性能表现。

**参考文献**

[1] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 3rd ed. MIT Press, 2009.

[2] D. E. Knuth,*The Art of Computer Programming, Volume 3: Sorting and Searching*, 2nd ed. Addison-Wesley, 1998.

[3] R. Sedgewick and K. Wayne,*Algorithms*, 4th ed. Addison-Wesley, 2011.

[4] J. L. Bentley and M. D. McIlroy, "Engineering a sort function," *Software: Practice and Experience*, vol. 23, no. 11, pp. 1249-1265, 1993.

[5] C. A. R. Hoare, "Quicksort," *The Computer Journal*, vol. 5, no. 1, pp. 10-15, 1962.