数据结构与算法基础课程实验报告

实验 5：排序算法实验比较

<table>
<tr><td>姓名</td><td>李大鑫</td><td>院系</td><td>计算机学院</td><td>学号</td><td>1170300825</td></tr>
<tr><td>任课教师</td><td colspan="2">臧天仪</td><td>指导教师</td><td colspan="2">臧天仪</td></tr>
<tr><td>实验地点</td><td colspan="2">GW</td><td>实验时间</td><td colspan="2">12/29/2018</td></tr>
<tr><td rowspan="2">实验课表现</td><td>出勤、表现得分 10%</td><td></td><td rowspan="2">实验报告得分 40%</td><td rowspan="2"></td><td rowspan="2">实验总分</td><td rowspan="2"></td></tr>
<tr><td>操作结果得分 50%</td><td></td></tr>
</table>

实验目的：

排序算法是数据处理的最基本和最重要的操作。其目的是将一组“无序”的记录序列调整为“有序”的记录序列。

实验内容：

实现一组经典的排序算法，通过实验数据的设计，考察不同规模和分布的数据对排序算法运行时间影响的规律，验证理论分析结果的正确性。

实验要求：（学生对预习要求的回答）（10 分）　　得分：

1、 首先选择 (冒泡排序，快速排序) 一组算法进行测试。

2、 产生不同规模和分布的数据，以“图或表”的方式给出输入规模和分布对排序方法运行时间变化趋势的影响（画出 T(n)的曲线）。并与理论分析结果比较。

设计实验数据：一共设计三种类型的实验数据，分别是：顺序、逆序、乱序。在画图时，分别对冒泡排序和快速排序在不同规模的三种数据下进行测试，对于冒泡排序，其数据规模范围为[0,10000]间隔为 100，一共 100 组数据；对于快速排序，数据规模为[0,100000]间隔为 1000，一共 100 组数据。将测出的时间结果输出到文件中，用来作图。

3、 如何作图？

使用Excel进行作图，将测出的数据复制粘贴到Excel中，通过其图表功能作图。

实验过程中遇到的问题如何解决的？（10 分）（着重从软件调试、质量保证、结果分析方面进行阐述）　　得分：

问题 1： 如何实现快速排序

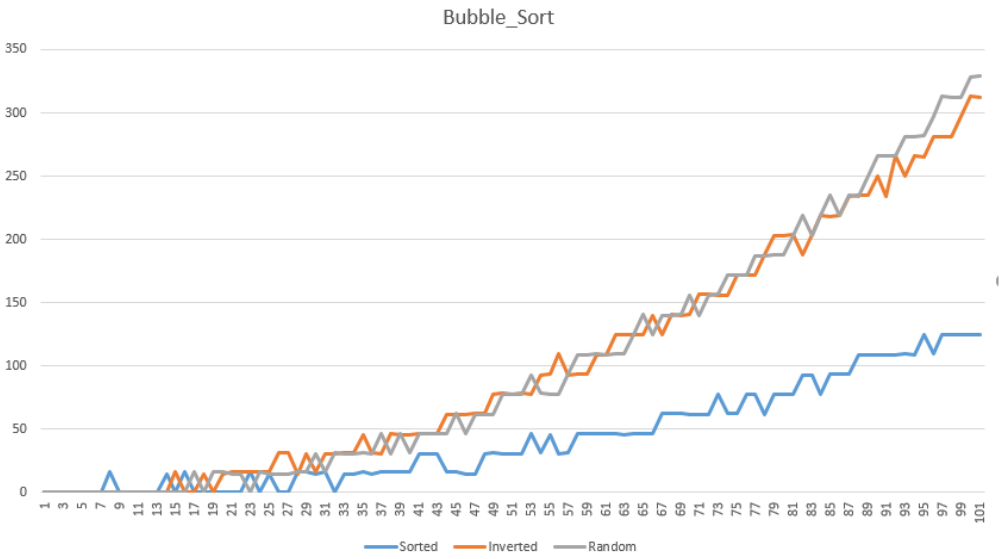
快速排序这一排序算法是基于分治算法的，即将大问题划分为相同类型的小问题，然后通过递归解决小问题从而将整个问题解决。快速排序一共分为三个大步骤，1）寻找基准点 2）通过基准点将区间内部的所有数组进行划分，划分为左右两部分，中间是基准点，左部分比基准点都要小，右部分比基准点都要大 3）分别递归解决左右部分。

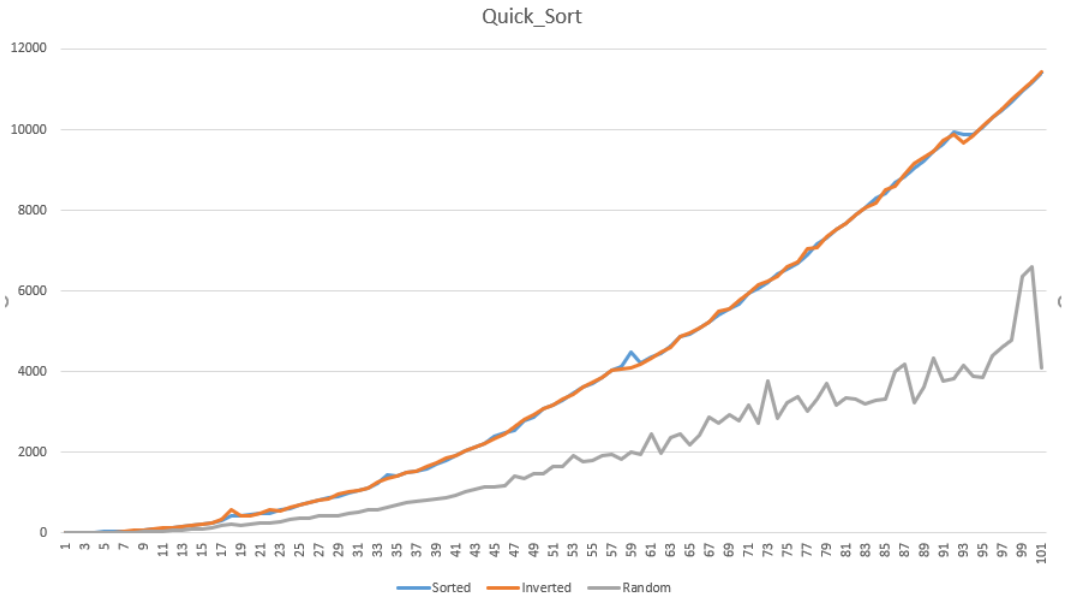
问题 2： 如何选取基准点？

在算法实现中，我们选取的是从区间最左边开始，第一组不相同的两个数中的较大的那个作为划分的基准点，这种选取基准点的方法是对直接选取区间最左边数据方法的改进，适用于这种情况，如果最左边有一部分相等的数的话，这些数最好不要移动，当出现第一个数大于这些相等的数的时候，显然就满足了不移动的情况，有效降低了交换次数。在这里如果随机选取基准点平均性能会更好一些。

本次实验的体会（结论）（10 分）	得分：

首先将冒泡排序与快速排序算法在三种不同类型、不同规模的数据下的运行时间的图表列在下面：





（两图中横坐标为数据序号，数据按照在上面实验要求中提到的给出 100 组；竖坐标为时间/ms）

在冒泡排序中，Sorted 数据最快，而 Inverted 和 Random 数据相近，原因是冒泡排序的思

想是在一次循环选取最小值的过程中如果后一个比前一个要小则交换，在顺序数据中，显然都满足后一个比前一个要小，所以就不需要进行交换，相比于其他两种则省去了交换的时间。

在快速排序中，Random 数据最快，Inverted 和 Sorted 数据相近，原因是对于 Inverted 和 Sorted 数据，在一个区间操作中，我们选取基准点为最小或者最大的，所以对于每一个处于区间中的数都需要进行交换，对于 Random 的情况，这时候我们选取的基准点就是随机的了，所以交换的次数降低，因此此时是选取基准点的算法限制了程序在 Sorted 和 Inverted 两类数据中的发挥。因此如果随机选取基准点算法会更高效一些。

思考题：（10 分）

思考题 1：（10 分）　　得分：

思考题 1：请说明实验中的排序算法的时间复杂度，并给出解释

冒泡排序：最优 O(n^2) 最差 O(n^2) 平均 O(n^2)。

冒泡排序使用两个 for 循环完成，第一层 for 循环代表选取第 i 个最小的数据到前部，第二个循环代表比较相邻值选出第 i 个最小的数据。最优的情况是初始数据已经排好序了，所以无需进行交换，则时间花销为 n*(n-1)/2，所以最优情况下时间复杂度为 O(n^2)；最差的情况就是开始的数据是完全逆序的，所以每一次循环每一次比较最终都需要进行元素的交换，所以时间花销为 4*n*(n-1)/2（比较、交换的三个步骤），所以最差情况是 O(n^2)。

快速排序：最好 O(nlogn) 平均时间复杂性为 O(nlogn) 最坏情况下为 O(n^2)

关于快速排序的时间复杂度证明比较复杂，查阅资料之后将时间复杂度的证明总结如下：

平均情况的分析

这是最难的部分。对于平均情况，我们假设对于 S_1，每一个文件大小都是等可能的，因此每个大小均有概率 $1/N$。这个假设对于我们这里的枢纽元选取和分割方法实际上是合理的，不过，对于某些其他情况它并不合理。那些不保持子文件(subfile)随机性的分割方法不能使用这种分析方法。有趣的是，这些方法看来导致程序在实际运行中花费更长的时间。

由该假设可知，$T(i)$(从而 $T(N-i-1)$)的平均值为 $(1/N)\sum_{j=0}^{N-1}T(j)$。此时方程(7.1)变成

$$T(N)=\frac{2}{N}\left[\sum_{j=0}^{N-1}T(j)\right]+cN \tag{7.14}$$

如果用 N 乘以方程(7.14)，则有

$$NT(N)=2\left[\sum_{j=0}^{N-1}T(j)\right]+cN^2 \tag{7.15}$$

我们需要除去求和符号以简化计算。注意，我们可以再套用一次方程(7.15)，得到

$$(N-1)T(N-1)=2\left[\sum_{j=0}^{N-2}T(j)\right]+c(N-1)^2 \tag{7.16}$$

若从(7.15)减去(7.16)，则得到

$$NT(N)-(N-1)T(N-1)=2T(N-1)+2cN-c \tag{7.17}$$

移项、合并并除去右边无关紧要的项 $-c$，我们得到

$$NT(N)=(N+1)T(N-1)+2cN \tag{7.18}$$

现在有了一个只用 $T(N-1)$表示 $T(N)$的公式。再用叠缩公式的思路，不过方程(7.18)的形式不适合。为此，用 $N(N+1)$除方程(7.18)：

$$\frac{T(N)}{N+1}=\frac{T(N-1)}{N}+\frac{2c}{N+1} \tag{7.19}$$

现在可以进行叠缩

$$\frac{T(N-1)}{N}=\frac{T(N-2)}{N-1}+\frac{2c}{N} \tag{7.20}$$

$$\frac{T(N-2)}{N-1}=\frac{T(N-3)}{N-2}+\frac{2c}{N-1} \tag{7.21}$$

$$\vdots$$

$$\frac{T(2)}{3}=\frac{T(1)}{2}+\frac{2c}{3} \tag{7.22}$$

将方程(7.19)到(7.22)相加，得到

$$\frac{T(N)}{N+1}=\frac{T(1)}{2}+2c\sum_{i=3}^{N+1}\frac{1}{i} \tag{7.23}$$

该和大约为 $\log_e(N+1)+\gamma-\frac{3}{2}$，其中 $\gamma\approx0.577$ 叫做欧拉常数(Euler's constant)，于是

$$\frac{T(N)}{N+1}=O(\log N) \tag{7.24}$$

从而

$$T(N)=O(N\log N) \tag{7.25}$$

虽然这里的分析看似复杂，但是实际上并不复杂——一旦你看出某些递推关系，这些步骤是很自然的。该分析实际上还可以再进一步。上面描述的高度优化的形式也已经被分析过，结果的获得非常困难，涉及到一些复杂的递归和高深的数学。相等关键字的影响也已仔细地进行了分析，实际上所介绍的程序就是这么做的。

最好情况的分析

在最好的情况下，枢纽元正好位于中间。为了简化数学推导，我们假设两个子数组恰好各为原数组的一半大小，虽然这会给出稍微过高的估计，但是由于我们只关心大 O 答案，因此结果还是可以接受的。

$$T(N)=2T(N/2)+cN \tag{7.7}$$

用 N 去除方程(7.7)的两边，

$$\frac{T(N)}{N}=\frac{T(N/2)}{N/2}+c \tag{7.8}$$

我们反复套用这个方程，得到

$$\frac{T(N/2)}{N/2}=\frac{T(N/4)}{N/4}+c \tag{7.9}$$

$$\frac{T(N/4)}{N/4}=\frac{T(N/8)}{N/8}+c \tag{7.10}$$

$$\vdots$$

$$\frac{T(2)}{2}=\frac{T(1)}{1}+c \tag{7.11}$$

将从(7.7)到(7.11)的方程加起来，并注意到它们共有 $\log N$ 个，于是

$$\frac{T(N)}{N}=\frac{T(1)}{1}+c\log N \tag{7.12}$$

由此得到

$$T(N)=cN\log N+N=O(N\log N) \tag{7.13}$$

注意，这和归并排序的分析完全相同，因此，我们得到相同的答案。

最坏情况的分析

枢纽元始终是最小元素。此时 $i=0$，如果我们忽略无关紧要的 $T(0)=1$，那么递推关系为

$$T(N)=T(N-1)+cN,N>1 \tag{7.2}$$

反复使用方程(7.2)，我们得到

$$T(N-1)=T(N-2)+c(N-1) \tag{7.3}$$

$$T(N-2)=T(N-3)+c(N-2) \tag{7.4}$$

$$\vdots$$

$$T(2)=T(1)+c(2) \tag{7.5}$$

将所有这些方程相加，得到

$$T(N)=T(1)+c\sum_{i=2}^{N}i=O(N^2) \tag{7.6}$$

这正是我们前面宣布的结果。

指导教师特殊评语：
指导教师签字： 日期：