**深 圳 大 学 实 验 报 告**

**课程名称：­ 算法设计与分析**

**实验项目名称： 算法性能分析**

**学院： 计算机与软件学院**

**专业： 计算机科学与技术**

**指导教师： 杨烜**

**报告人： 郑雨婷 学号：2021150122 班级： 高性能**

**实验时间： 2023/3/2——2023/3/16**

**实验报告提交时间： 2023/3/19**

**教务部制**

|  |  |  |
| --- | --- | --- |
| 实验目的与要求：  1.掌握选择排序、冒泡排序、合并排序、快速排序、插入排序算法原理  2.掌握不同排序算法时间效率的经验分析方法，验证理论分析与经验分析的一致性。 | | |
| 方法、步骤：  按照选择排序、冒泡排序、合并排序，快速排序、插入排序的顺序，给出每一种算法的原理、伪代码、并且通过对大量样本的测试结果，统计不同排序算法的时间效率与输入规模的关系，通过经验分析方法，展示不同排序算法的时间复杂度，并与理论分析的基本运算次数做比较，验证理论分析结论的正确性。  通过分析算法的效率，在有10亿的数据，选出快速挑选出最大的十个数的算法。并且通过实测，验证算法的正确性。 | | |
| 实验过程及内容：   1. 选择排序   1.1算法原理  选出未排序中最小的放在已排序队列尾端。步骤：   1. 第一趟选出最小的元素，放入排好序的序列第一个。 2. 之后每一趟选出还未排序的最小的数，放在已经排好的序列之后。 3. 重复②，一直到所有元素都排序完成。     图1 选择排序示意图  1.2 伪代码实现  For i=1 to n-1;  min = i+1;  for j=i+1 to n  if( A[j]<A[min]) min=j;  swap(A[i],A[min]);  1.3 算法实际执行时间  选择排序算法各数据规模下运行平均时间如下表1所示：    表1 选择排序在不同规模下实测运行时间  1.4 算法效率分析  选择排序算法时间复杂度平均值为，经过实际运行，我选取排序100000个数据作为基准，通过之间的比例关系，计算得到各实验数据中算法执行时间的理论值。通过程序执行，求得选择排序各数据规模理论值与误差如下表2。    图 2 选择排序不同数据规模下的理论值和实验值    表2 选择排序理论值于实际值的对比  由上方图表可以发现，理论曲线和实际曲线几乎重合，并且符合x的平方的趋势。   1. 冒泡排序   2.1 算法原理  n个元素，共n-1趟，两两比较，第i趟把未排的最大的元素放到n+1-i的位置。  步骤：  ①依次比较相邻两个元素的大小，若前大于后，则交换。  ②这样，最大的元素就放在最后面了。  ③n-1趟交换即可排序完成。    图3 冒泡排序示意图  2.2 伪代码实现  for i= 1 to n-1  tag=0;  for j= 1 to n-i  if Al] > Ai + 1] swap(A],A + 1]),tag=1;  if tag=0 break;  2.3 算法实际执行时间  冒泡排序算法各数据规模下运行平均时间如下表3所示：    表3 冒泡排序在不同规模下实测运行时间  表3 冒泡排序在不同规模下实测运行时间  2.4 算法效率分析  冒泡排序算法时间复杂度平均值为，经过实际运行，我选取排序100000个数据作为基准，通过之间的比例关系，计算得到各实验数据中算法执行时间的理论值。通过程序执行，求得冒泡排序各数据规模理论值与误差如下表4。    图 4 冒泡排序不同数据规模下的理论值和实验值    表4 冒泡排序理论值于实际值的对比  由上方图表可以发现，理论曲线和实际曲线几乎重合，并且符合x的平方的趋势。在数据规模增大时，误差也增大，我认为可能是数据量为100000时，我所定下的基准，即用以计算理论值的数值较低，导致实验误差较大，探其原因，可能是运行过程中电脑环境不同所导致。   1. 合并排序   3.1 算法原理  原理：分治思想，先分成小问题，再继续分，解决子问题，合并求出大问题的解。用递归的思想。  步骤：   1. 把一个序列分成两个子序列。 2. 求出子序列的解。 3. 用子序列的解合并出主序列的解。 4. 至于子序列如何求解，继续分成更小的子问题。这里体现出递归的思想。   3.2 伪代码实现    图5 合并排序示意图  Merge\_sort(A,p,r)  If p < r  q=( p + r )/2  Merge\_sort(A,p,q)  Merge\_sort(A,q+1,r)  Merge(A,p,q,r)  3.3 算法实际执行时间  合并排序算法各数据规模下运行平均时间如下表5所示：    表5 合并排序在不同规模下实测运行时间  3.4 算法效率分析  合并排序算法时间复杂度平均值为，经过实际运行，我选取排序100000个数据作为基准，通过之间的比例关系，计算得到各实验数据中算法执行时间的理论值。通过程序执行，求得合并排序各数据规模理论值与误差如下表6。    表6 合并排序理论值于实际值的对比  由上方图表可以发现，理论曲线和实际曲线几乎重合，曲线看起来像一条直线是因为nlogn在n大于100000时，曲线 斜率接近1。    图 6 合并排序不同数据规模下的理论值和实验值   1. 快速排序   4.1 算法原理  小于pivot的全部放在前面，大于于pivot全部放在后面。用递归解决左右两边子序列。   1. 从数列中挑出一个元素，称为 "基准"（pivot）; 2. 重新排序数列，所有元素比基准值小的摆放在基准前面，所有元素比基准值大的摆在基准的后面（相同的数可以到任一边）。基准处于数列的中间位置。 3. 递归地把基准值左侧的元素和基准值右侧的元素排序；     图7 快速排序示意图  4.2 伪代码实现  QUICKSORT(A,low,high)  if(low>=high) return;  pivot = A[low];  Quick(A,low,high);  QUICKSORT (A,low,pivot -1);  QUICKSORT (A,pivot +1,high)  4.3 算法实际执行时间    表7 快速排序在不同规模下实测运行时间  4.4 算法效率分析  快速排序算法时间复杂度平均值为，经过实际运行，我选取排序100000个数据作为基准，通过之间的比例关系，计算得到各实验数据中算法执行时间的理论值。通过程序执行，求得快速排序各数据规模理论值与误差如下表。      图 8 快速排序不同数据规模下的理论值和实验值    表8 快速排序理论值于实际值的对比  由上方图表可以发现，理论曲线和实际曲线几乎重合，曲线看起来像一条直线是因为nlogn在n大于100000时，曲线 斜率接近1   1. 插入排序   5.1 算法原理  原理：每次排序时前面的元素已经排好了。  ①从第一个元素开始，该元素可以认为已经被排序  ②取出下一个元素，在已经排序的元素序列中从后向前扫描  ③如果被扫描的元素（已排序）大于新元素，将该元素后移一位  ④重复步骤3，直到找到已排序的元素小于或者等于新元素的位置  ⑤将新元素插入到该位置后  ⑥重复步骤2~5    图9 插入排序示意图  5.2 伪代码实现  NSERTION-SORT(A)  for i=2 to n:  key=A[i]  j=i-1  while j>0 and A[j]>key  A[j+1]= A[j]  j=j-1  A[j+1]=key  5.3 算法实际执行时间    表9 插入排序在不同规模下实测运行时间  5.4 算法效率分析  插入排序算法时间复杂度平均值为，经过实际运行，我选取排序100000个数据作为基准，通过之间的比例关系，计算得到各实验数据中算法执行时间的理论值。通过程序执行，求得插入排序各数据规模理论值与误差如下表。    表10 插入排序理论值于实际值的对比    图 10 插入排序不同数据规模下的理论值和实验值  由上方图表可以发现，理论曲线和实际曲线几乎重合，并且符合x的平方的趋势。  六、误差分析  在五种算法中，理论运行时间和实际平均运行时间都有一定的差距，可能原因如下：  1.基准数据测量不准确，导致理论值计算有偏差。  2.在测试样本数据时，电脑在运行其他程序，导致CPU资源分配，使得算法程序运行较慢。  3.算法设计有问题。有些算法可能在设计时做了无用功，导致程序运行偏慢。 |
| 数据处理分析：   1. 排序算法比对及误差分析   将时间复杂度为的三种算法（选择排序、冒泡排序、插入排序）放在一起比较分析。三种算法的实际运行时间比较如下表：    图11 选择、冒泡、排序比较    表11 选择、冒泡、排序比较  根据图表可以看出，插入排序的效率最高，选择排序的效率居中、冒泡排序的效率源低于前面两者，呈现这个结果的可能原因如下：  1.冒泡排序是相邻两个元素两两比较，只要逆序就要执行一次交换，交换的次数远大于其他两种算法。  2.选择排序虽然每一次循环都要遍历所有元素选出最小的，但是只是进行比较吗，实际交换的操作只有一次，因此，效率要高于冒泡排序。  3.插入排序的比较次数小于选择排序，因此效率高于选择排序。  将时间复杂度为的两种算法（合并排序、快速排序）放在一起比较分析。两种算法的实际运行时间比较如下表：    表12 合并、快速排序比较    图12 合并、快速排序比较  根据图表可以看出，快速排序的效率高于合并排序，呈现这个结果的可能原因为合并排序在合并时需要用到辅助空间，而快速排序是就地操作，在连续的地址进行操作所耗的时间较少。   1. TopK问题   现在有10亿的数据（每个数据四个字节），请快速挑选出最大的十个数，并在小规模数据上验证算法的正确性。  1.选择排序，因为选择排序只需要遍历十次即可选出最大的十个数（冒泡排序也是遍历十次，但是冒泡的效率远低于选择排序）。经过测试，得出在不同规模下的运行时间：    2.不完全推排序。首先将前十个数建成小顶堆，从第11个数开始，每个数都与堆顶进行比较，若小于堆顶，说明一定不是最大的十个数。若大于堆顶，说明有资格竞选最大的十个数，然后更新堆。经过测试，得出在不同规模下的运行时间：    3.将以上两种算法的运行时间放在一起比较，可以看出，堆排序的效率高于选择排序的效率。      我们分析可知，堆排序每次插入一个新的元素的时间复杂度为在要对数据规模为n的数据进行比较操作时，时间复杂度为。选择排序在数据规模为n时的时间复杂度为,当k=10,n=10亿时，堆排序的时间复杂度低于选择排序，和实测结果一致，说明算法正确。 |

|  |
| --- |
| 实验结论：  在这次实验中，我们主要研究了关于各排序算法的实现原理即其时间复杂度分析。通过这次实验，我才知道时间复杂度相同的算法的效率也会有很大的差别的。在冒泡算法中，利用标志位提前结束循环在大规模数据下并没有提高效率，因为在大规模数据下很难提前排好序，反而会增加比较次数。  同时，在TopK问题中，通过比较不同的算法，更深刻体会到算法的效率在大规模数据下的重要性，算法设计正确可以大大提高效率。 |
| 指导教师批阅意见：  成绩评定：  指导教师签字：  年 月 日 |
| 备注： |

注：1、报告内的项目或内容设置，可根据实际情况加以调整和补充。

2、教师批改学生实验报告时间应在学生提交实验报告时间后10日内。