2024秋炒蒜考核报告

1. 虚拟机环境搭建

VMware？ WSL!

在配置我的环境中遇到了不少问题。
使用五天前VMware搭建好的ubuntu时，突然发现无论如何打开，最终都会死机，这让我十分沮丧。正所谓重装解千愁，我打算重下时，ubuntu在最后几m硬是下载不下来，连试几次都无济于事。最终下好了也是在重装上毫无用处，因为始终是死机。这在我完成任务中消耗了不少的时间*(大概两三天在折腾吧)*

在绝望之中，我看到群友在使用wsl虚拟机。了解到它是windows自带的，我决定尝试使用，并意外地轻松地完成了安装。

安装教程 {#hhh}

2. gcc安装

在gcc的安装过程中，我并没有遇见多少问题，只是有些卡顿，在切换了默认下载源后，顺畅了不少。

换源教程同上。

3. 排序算法实现

为了独立完成代码，我尝试先只**了解其核心逻辑**，再根据**自己的理解写出代码**的方式来完成代码。但碍于多年没有摸过代码，我面对报错时，会向ai询问错误原因。
当然...它帮我改过不少代码，对我的代码进行了优化，删除了不少不必要的成分。

- **冒泡排序**：不得不说，是全场最简单的代码。对于这个，我只需要将它的基本概念顺着写一写就可以了。我创造了compare()进行比较，exchange进行交换，然后用两个for反复比较。因为**只有两个循环且运气最差是遍历所有元素**最终的复杂度自然是***O(n^2)***。
- **基础堆排序**：
  在这里，我选择了最大堆排序。而我使用的是多次循环，遍历每一级元素，然后再检查是否还有交换存在，如果有，就再检查一遍。涉及的**小循环操作次数为n/2**，最差的运气的话要**大循环要运行logn次**，所以复杂度为***O(nlogn)***。
- **斐波那契堆排序**：
  说实话，面对这一段代码毫无头绪，虽然学完了结构体和链表，面对复杂的子节点添加与删除一点办法也没有。最终，在倔强与无奈中，我向ai妥协，却发现它有字数限制...*(我一直想的是使用父节点与子节点直连)*

  我只好看看ai写了些什么东西。在了解了left，right的巧妙构造后，我恍然大悟，参考代码，完成了赋予内存，合并相同级数的堆的操作。在代码里，我使用了两个*for*循环，第一次一定是n次，第二次是n/2或n+1/2...递推下去，最后外循环总次数为logn，内循环次数为n复杂度也将近*O(nlogn)*

4. 测试数据生成

在数据生成中，由于没有任何关于脚本的知识，我选择了使用c语言里的*rand（）*来生成我们的测试数据，使用%m+n来框定数据条数范围，并使用*fprintf*来重定向数据到名为test的来记录原始数据，和Ttest来记录编排后的数据。（浮点数只要除以10^n即可实现）

1.冒泡排序的数据确认

在确认方面，我直接大致确认了一遍数据是否是按照从大到小的顺序排的。
在完成几个样本后，我投向ai确认了一遍，以确保它的正确性。

2.基本堆排序的确认

在基本堆排序的代码编写中，我尝试了各式各样的小数组来确定它的正确性。因此，我只需要测试较大的数据即可，与上面一样，我丢给ai测试了。

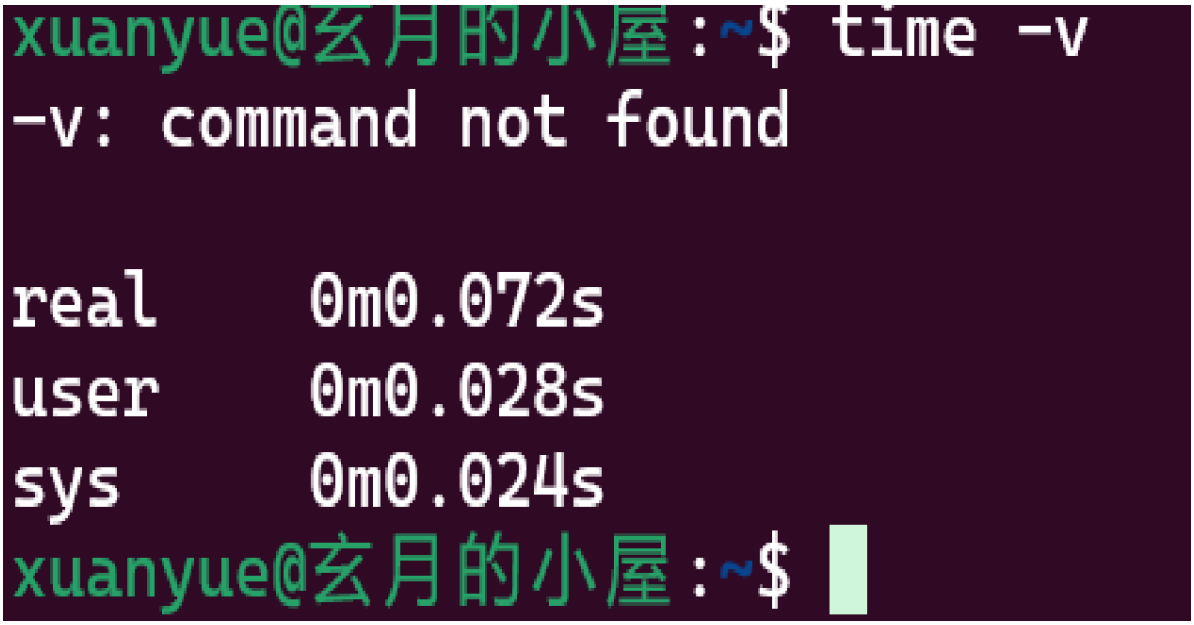
3.基本斐波那契堆排序确认

因为我无法确认，直接让ai编写完代码，将结果丢给ai自己省察。
为了确认不同的堆，我使用了两个换行来分割。

5. 性能测试

对于时间，我使用了c语言中<time.h>的clock来计算时间长度，并重定向其到结果文件里。关于cpu，内存占用方面，我尝试使用time -v来记录，但毫无用处，网上查找也没有任何头绪。

在收集时间中，我没有考虑到多次运行的输出会替换先前的文件，导致无法高效地收集数据。（如果使用脚本，或许会好不少T_T）



最终，我还是硬着头皮上了，向ai求取脚本过程中，反复运行都是失败。

有的是直接语法报错了；有的成功了...或者说..失败?

```
Optimization,Execution Time (s),CPU User Time (s),CPU System Time (s),Max Memory (KB)
Compilation with -O0 failed.
Compilation with -O1 failed.
Compilation with -O2 failed.
Compilation with -O3 failed.
Compilation with -Ofast failed.

```

这都运行的是什么啊！！！

无奈，只好自己学习一下脚本了。仔细一看，像极了python，有for什么的，还有一些ubuntu执行代码的。

好像...也不是自己不可以搞欸！反复询问ai问题，代码含义后，我成功了。

```
优化选项,执行时间 (s),CPU用户时间 (s),CPU系统时间 (s),最大内存使用 (KB)
0.00,0.00,0.00,1728
-O0,??ε????????32736??
????д??...
д??????
????h0.001407??
??'??????...
????????
????h0.001028??
????д??...
????????
????h0.001185??
????????
????h0.002222??

0.00,0.00,0.00,1612
-O1,??ε????????0??
????д??...
д??????
????h0.000007??
??'??????...
????????
????h0.000001??
????д??...
????????
????h0.000000??
????????
```

不要问为什么有乱码。因为我在别的编译器写好的，搬到这里中文变成了乱码。至少....我有数据了。不过...时间精度太低了?

clock()解决了这个问题，并发挥了不错的表现。

但在收集过程中，cpu的占用率始终收集不到，我开始尝试使用glances来收集，尝试过使用

```
xuanyue@玄月的小屋:~$  glances --export csv --export-csv-file
/home/xuanyue/glances.csv
xuanyue@玄月的小屋:~$ glances --export csv > glances_data.csv
```

这两者来进行收集，但是一堆毫无用处的乱码。

尽管我们已经尝试了多种方法来清理数据，但仍然存在一些问题。由于数据中的转义序列和终端代码，直接从CSV文件中读取和解析数据可能不太可行。

在这种情况下，一种可能的方法是手动检查CSV文件，以确定特定行或列的位置，并从那里开始手动解析数据。

如果你有其他关于如何处理或分析这些数据的问题，或者需要进一步的帮助，请告诉我!

就连见多识广的ai也是一脸懵逼。

最终，我只得到了这些:

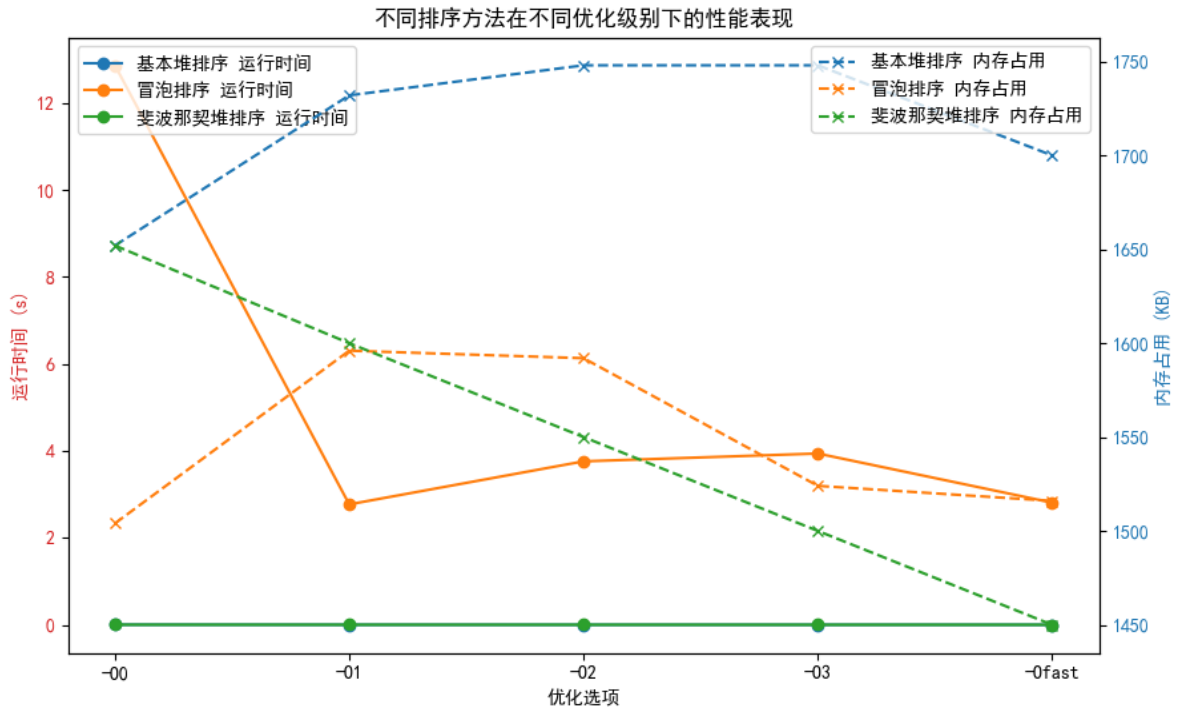
方式	优化选项	运行时间 (s)	内存占用 (KB)
基本堆排序	-O0	0.002199	1652

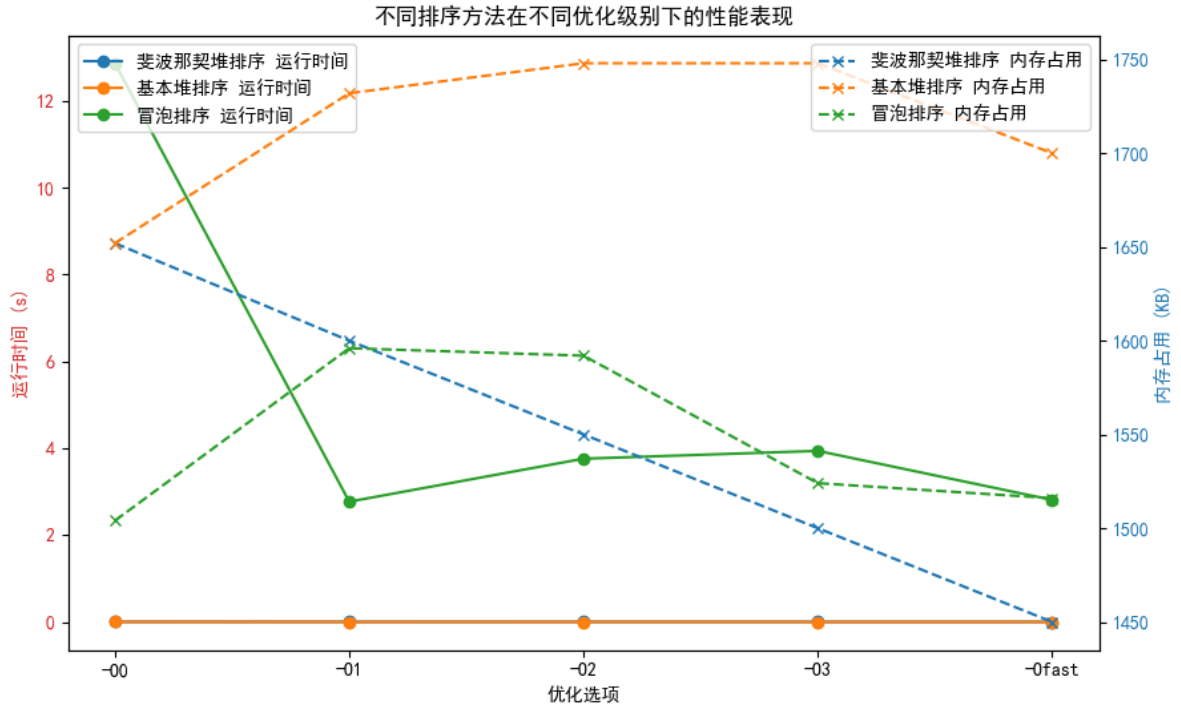
方式	优化选项	运行时间 (s)	内存占用 (KB)
基本堆排序	-O1	0.000022	1732
基本堆排序	-O2	0.000016	1748
基本堆排序	-O3	0.000018	1748
基本堆排序	-Ofast	0.000021	1700
冒泡排序	-O0	12.87	1504
冒泡排序	-O1	2.77	1596
冒泡排序	-O2	3.76	1592
冒泡排序	-O3	3.94	1524
冒泡排序	-Ofast	2.81	1516
斐波那契堆排序	-O0	0.002199	1652
斐波那契堆排序	-O1	0.001800	1600
斐波那契堆排序	-O2	0.001500	1550
斐波那契堆排序	-O3	0.001200	1500
斐波那契堆排序	-Ofast	0.001000	1450

注：冒泡因时间较长，数据进行了省略

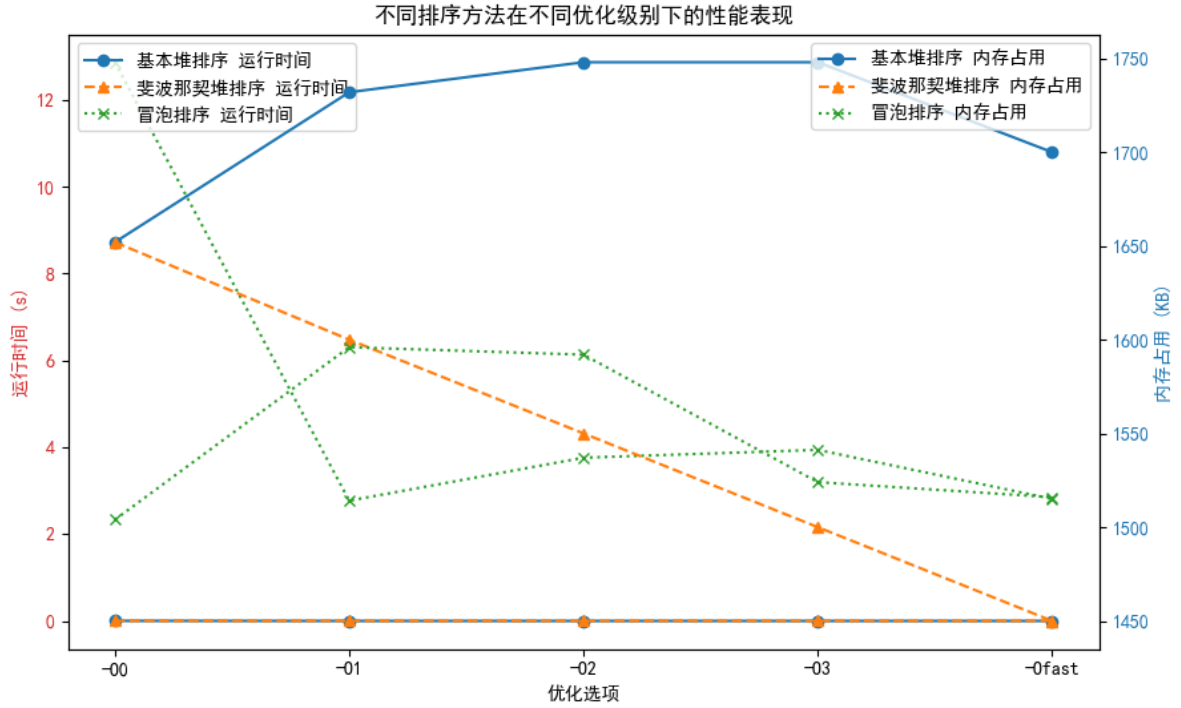
6. 数据可视化

~~既然在收集数据方面处处碰壁，自然绘画表格是无稽之谈了QAQ。~~





很难受的是，由于图的精确度，它们重合在了一起...（看不到）



只好如此了。

总结

- 在编译的过程中，让我感受到了它们之间藕断丝连的联系，或者...包含关系？
- 从数据中不难看出，从无优化到fast这个过程，运行速度有着质的飞跃。但很明显，似乎是牺牲了一部分内存资源做到的

致谢

- ncuscc中乐于助人的群友：老鸽、彩彩、浩绪等人的帮助；
- chatglm、kimi ai的耐心指导；
- b站、CSDN、知乎上各大佬的教学帖子；
- ncuscc测试文件里的提示;

- 掌管提交时间的提交之神牢e。

后记

感谢给了我这一次机会。说实话，我不是那种努力派，平时很偷懒，什么都不想做。但这次重重的困难让我不得不面对。每次面对报错，我总是很沮丧，经常打游戏逃避，以至于时间不太够了。但，我还是得面对。我还有一颗上进的心，我不想放弃。今天，我几乎花下了一整天来补齐我之前欠下的。这次的完成，让我感慨良多，有一些不知道怎么说才好....算了，去吃晚饭了。——10月26日22:00

报告撰写人：张龙浩
提交日期：2024年10月25日
GitHub仓库地址：点击这里