*[金山办公]*

*[文本监听工具]*

性能测试报告

版本号： *V1.0*

*2022年 8月 18 日*

# 概述

## 编写目的

编写该测试总结报告主要有以下几个目的：

1. 熟练对性能测试分析工具的使用，方便对后续代码的优化
2. 通过对测试结果的分析，对系统整体质量有所了解
3. 根据分析报告中得出的结论进行性能优化

## 阅读对象

主要读者：开发

其他读者：无

# 1.3范围

本次测试是针对整个文本监听工具的性能测试，包括监听路径的添加/删除/编辑、监听日志的查询等。

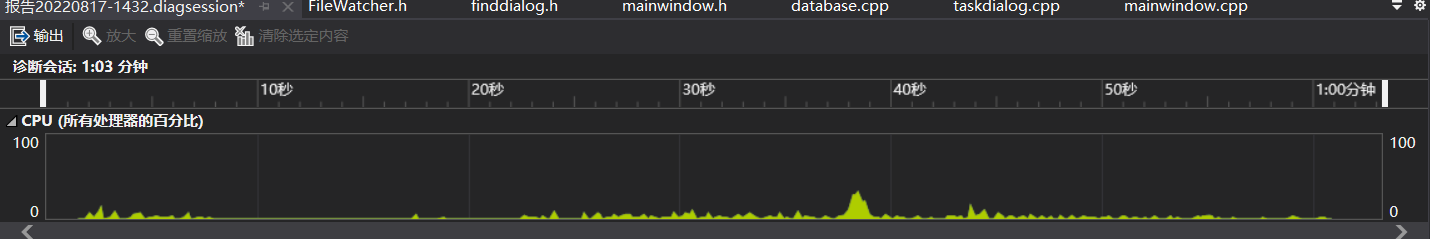
# 测试工具

VS的性能探查器：1、概要:整个报告概要说明 调用树:以树形表格的方式展开函数之间的关系。 2、模块:分析调用的不同的程序模块， 如不同的DLL、lib模块的耗时. 调用方/被调用方:以数值显示的调用与被调用的关系 3、函数:以数值显示的各个函数的执行时间和执行次数统计值 4、进程：显示当前进程的信息 5、函数详细信息：以图表的方式形象地显示， 调用函数-当前函数-被调用子函数之间的关系和时间比例。

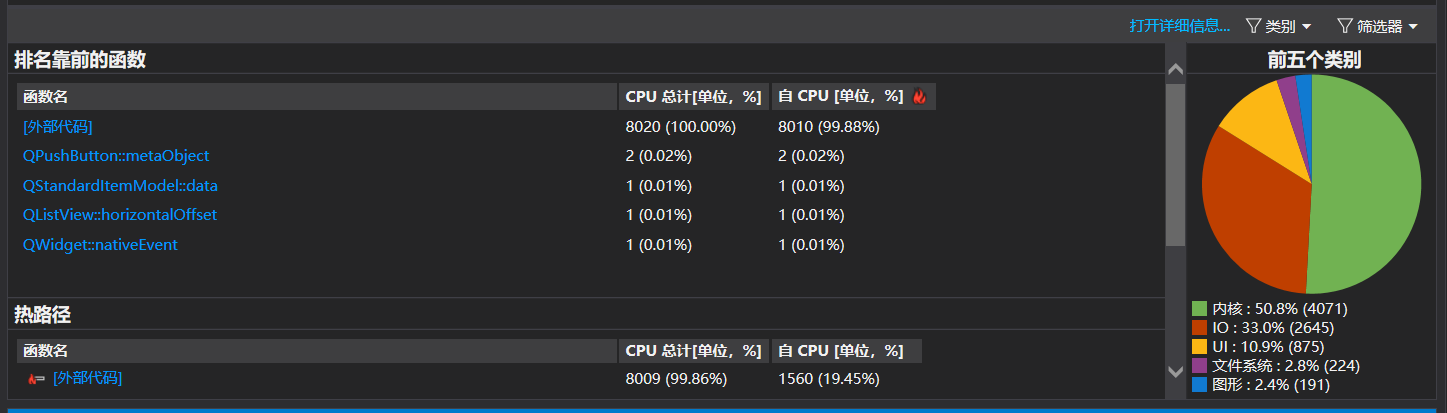
VS的诊断工具：这里主要用到的分析数据就是的CPU使用率，通过选取合适时间段的CPU使用情况，找出CPU占用比较多的函数进行瓶颈突破。

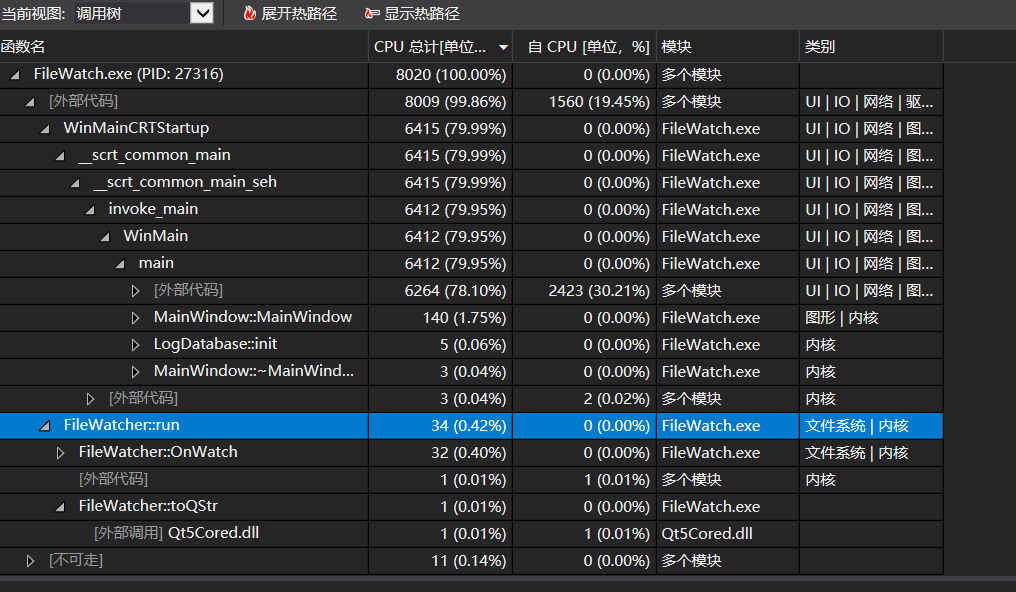
# 测试结果及分析

1. vs性能探测器分析结果如下：



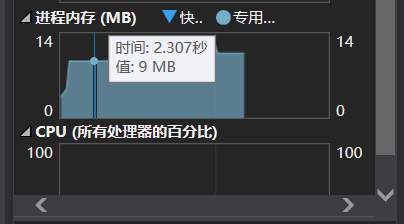
峰值为添加新的监听路径



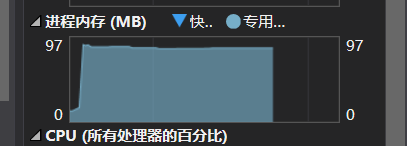


1. vs诊断工具分析结果如下

文件多线程监听/日志查询/任务编辑(不包括路径修改)：



添加新的监听路径：



1. 结果分析

占据cpu时间的事件多为io事件，在本项目中表现为添加新的监听路径。

文件IO是不带缓冲的，所谓不带缓冲，即每调用一次IO(如write)，就进行一次用户态与内核态的切换，调用一次write，就需要立即把数据从用户态拷贝到内核态。在本项目中表现为占据了较大cpu时间。

监听线程也占据了部分cpu时间，考虑到在进行添加监听路径操作后，整体cpu占用率为0.42%，已经具备了较高的性能，性能优化考虑从IO事件的优化入手。

# 优化建议

实际IO的速度有存储设备本身物理特性决定的，无法优化；但我们可以优化寻道和等待的时间（最理想的情况下是硬盘不需要寻道，也不需要等待），我们可以通过以下操作降低寻道和等待消耗的时间：

1.按顺序读取数据；这样就不需要寻道和等待了。

2.数据尽可能一次性读写，减少文件碎片；文件碎片就是一个文件在存储设备上存放时不是连续存放的，假设一个文件2K大小，可能它的前512字节存放在扇区1，第二个512字节存放在扇区10，最后1K存放在扇区5。这样虽然在逻辑上读取文件时没有seek操作，但实际上IO时还是需要等待的。

大部分的文件系统是先到先得的原则，因此一次性写入可以极大的降低文件碎片的情况。

3.字节对齐；硬盘IO的最小单位为扇区，即使需要IO的数据小于一个扇区，实际IO时也是对一个扇区进行操作的。eg：假设一个扇区为512字节，现在需要分两次一共写入1K的数据，这1K数据存放在两个连续的扇区，扇区1和扇区2中。

（1）如果每次写512字节，整个IO消耗的时间就是两次真实IO的时间；

（2）而如果不对齐写，第一次写511个字节，第二次写513个字节；由于硬盘一次必须写入512字节，因此第一次IO时扇区1中写入了512字节，最后一个字节是无效数据，第二次IO时磁头已经偏移到扇区2了，但控制器发现第512个字节是在扇区1里面的，所以控制器会等到扇区1再次出现在磁头下面后才开始第二次IO，并且这次IO实际上只有一个字节是有效的；写完后还需要对扇区2再进行一次IO操作。因此整个IO消耗的时间等于三次真实IO的时间加上等待盘片转完一遍的时间（U盘的话还存在一个写惩罚，即每次IO时都要先将原来的数据读取出来，然后擦除数据，最后再将新数据写入进去，这也需要时间）

IO过程中，数据在这些缓冲区内进行复制也要消耗时间和性能的，字节对齐后，这些复制操作可视情况去掉（通过open时传入的参数进行修改），这里又可以节省点时间

4.减少不必要的IO同步操作；一般IO时，操作系统、磁盘控制器等都会对传下来的数据进行优化，比如将多次write合并成一次，调整IO顺序，数据对齐等操作；而IO同步时不经过这些优化操作，一旦传下来的数据诸如字节未对齐，多次IO等会极大影响IO的性能。

5.减少不必要的文件开关操作；大部分的文件系统中，文件的属性数据和文件的真实数据是分开存放的，open一个文件时，操作系统要先找到文件的属性数据将这些数据载入到内存，然后根据属性数据再偏移到文件真实数据存放的扇区对数据进行操作；close时一般操作系统会强制刷新数据到存储设备，然后再偏移到文件的属性数据存放的扇区更新属性信息，同样也会强制刷新数据；

原文链接：https://blog.csdn.net/xypzwl/article/details/51416883