Meowlab实验报告

任务3

1.为什么将缓冲区对齐到系统的内存可能提高性能？你的实验结果支持这个猜想吗？为什么？

文件系统一般以页为基本内存单位，加载缓存时如果缓冲区在同一页，可以一次将整个页加载到缓存中，避免在不同页的多次缓存加载。现代处理器主要使用DMA进行加载，将同一页的内存加载到同一页大小的缓存中时，也可以简化地址映射的计算。

实际实验结果和没有进行缓冲区对齐的效果几乎一致，甚至稍慢，并不支持这个猜想。

原因在于cat的缓冲区即使不存在在同一页，IO操作也可以做到并行加载，且地址计算所需时间非常短。现有的malloc的优化非常好，即使缓冲区没有完全对齐，malloc也可以做到尽可能的对齐减少开销。使用posix_memalign强行分配一个对齐的内存可能会导致malloc花费更多的时间查找符合条件的内存，且相对于malloc的内存管理，posix_memalign的分配逻辑更加复杂，反而导致cat时间更长。

2.为什么我们直接使用 *`malloc`* 函数分配的内存不能对齐到内存页，即使我们分配的内存大小已经是内存页大小的整数倍了。

malloc的目的在于尽可能分配满足需求的内存，如果系统中占用的内存较少则可能分配到对齐的内存页，但在系统空闲内存较少的情况下，强行分配对齐到内存页的内存可能造成巨大的内存碎片，导致malloc不好管理。

3.你是怎么在不知道原始的 `malloc` 返回的指针的情况下正确释放内存的?

代码中内存分配的函数为posix_memalign，这个函数的抽象能够直接返回对齐后的块的指针，且块的头部与尾部与正常的malloc分配的块没有区别，因此可以直接使用free函数进行释放。

任务4

1. 为什么在设置缓冲区大小的时候需要考虑到文件系统块的大小?

文件系统的块大小是系统进行IO操作的基本单位，如果缓存区的大小是文件系统块大小的整数倍，加载文件系统时能够更好的和文件系统的数据对齐，省去了复杂的地址计算过程。不对齐可能导致对磁盘多次的访问且每次访问只访问块的一小部分，导致IO效率较低。

2. 对于文件系统块大小可能不同和非2的整数次幂的注意事项，如何解决?

可以使用fstat动态获取不同文件的块大小，动态调整块的大小，避免固定大小。

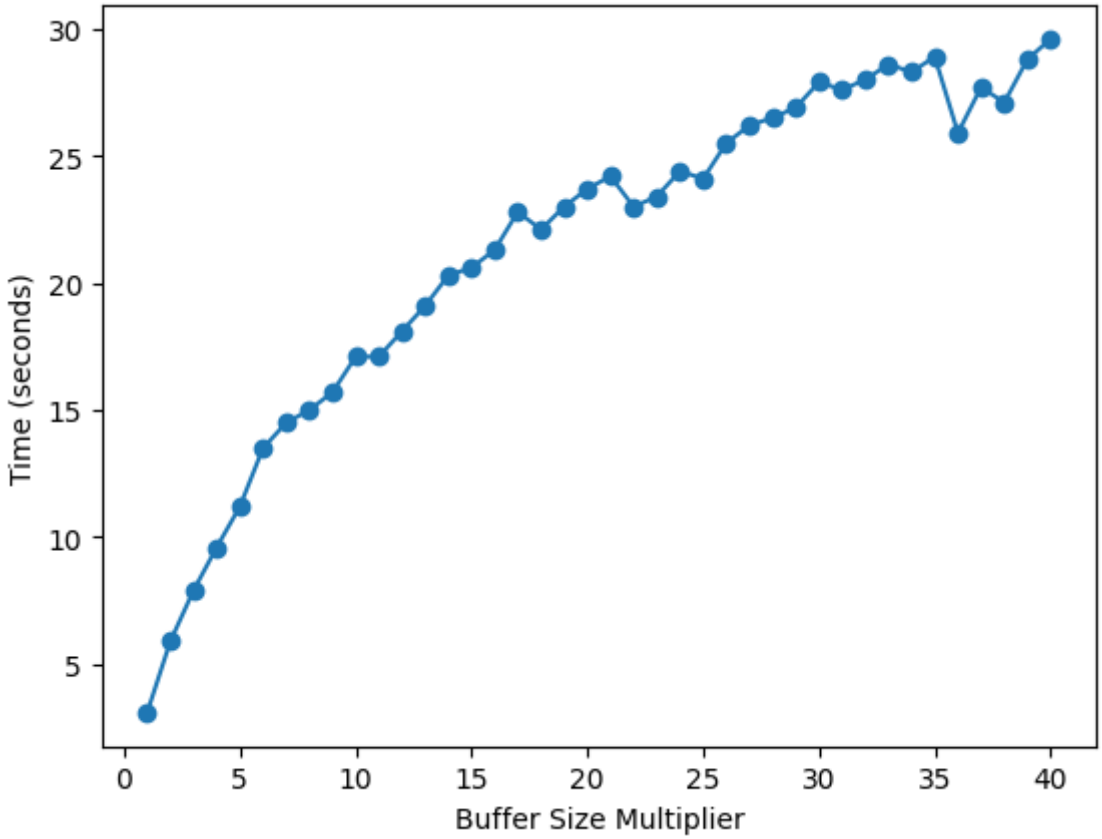
使用文件系统块大小和内存页大小的最小公倍数，可以保证缓存区大小是两者的整数倍，同时尽可能的减小内存占用。

任务5

1. 解释一下你的实验脚本是怎么设计的。你应该尝试了多种倍率，请将它们的读写速率画成图表包含在文档中。

实验脚本使用的测试环境为基础大小为4096（和系统的实际文件系统块大小、内存页大小相等），读取大小为1GB。实际测试为将1GB数据从/dev/zero读取到/dev/null测试系统性能，并用grep捕获输出dd指令输出的系统信息来展示缓存区大小对于系统读取速度的提升

实际测试范围为base size的1-40倍，测试得到的图表如下。



可以发现随着buffer的大小增大，读取速度上升。但是读取速度的提升并非线性，而是在20倍左右起增长放缓。为了平衡内存占用和读取速度，实际采用的倍数为20倍。

任务6

1. 你是如何设置 `fadvise` 的参数的?

参数设置为POSIX_FADV_SEQUENTIAL，告诉文件系统文件将要被顺序读取，以便于内核优化顺序读取情况。

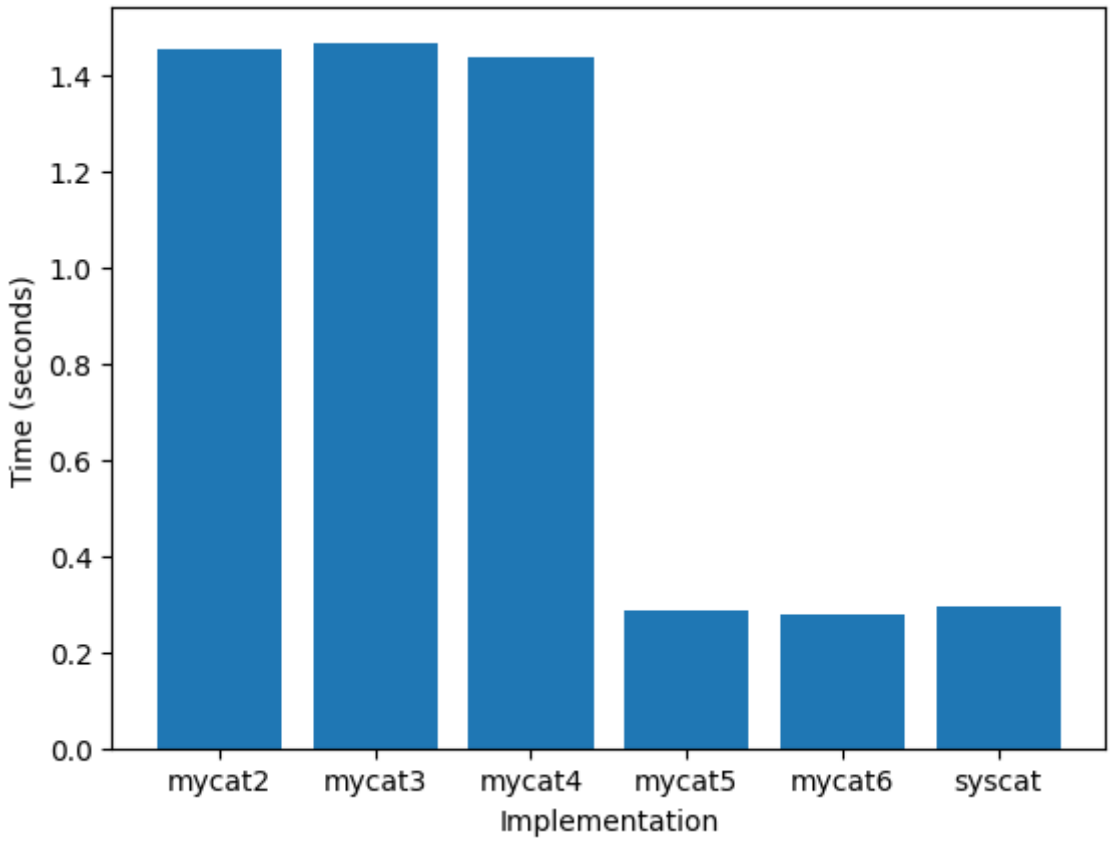
2. 对于顺序读写的情况，文件系统可以如何调整readahead？对于随机读写的情况呢?

对于顺序读写的情况下，文件系统可以异步的预读取当前正在读取的数据的下一个文件块。具体来说，内核会检测顺序读取的请求，在顺序请求较多时会预读下一个顺序块的数据，并且随着顺序预读取的命中率增多，内核会增大预读取的数据量。

在随机读取的情况下，文件系统也会检测随机读取的请求。在完全随机读取的时候会禁用readahead，或者随机预读取。

任务7

实际实验结果



mycat1占用时间远长于mycat2，因此没有表现在柱状图内。

mycat2在引入较大的缓冲区后，速度相比于mycat1快很多。

mycat3由于强行分配内存对齐的缓冲区块，导致malloc的管理难度增大，分配更加复杂，因此耗时略微多于mycat2。

mycat4采用了文件系统块和内存页的最小公倍数作为实际内存块，能够使文件在从磁盘读取的时候简化多次读取同一个数据块的次数。但是我的电脑中文件系统块和内存页大小均为4K，导致实际缓冲区大小没有变化，mycat4的表现和mycat3差距很小。

mycat5由于采用了更大的缓冲区大小，能够使得文件单次IO读取更多的数据，使得cat的表现更好。

mycat6采用了fadvice，但是考虑到现有文件系统的优化，文件系统能够自动使用readahead进行优化，实际fadvice的作用不大，同时fadvice也可能被内核选择性忽略，实际表现出和mycat5相似的性能。

系统标准cat可能采用的缓冲区大小不是很大（为了节省内存占用），实际的表现略微逊色于mycat5和6。