MeowLab 实验报告

刘曜嘉 2023202289

实验性能汇总

程序版本	平均运行时间 (ms)	说明
`system cat` (基准)	203.9	系统原生实现
`mycat1` (逐字节)	460408.0	性能极差，系统调用开销巨大
`mycat2` (页大小缓冲)	276.8	引入缓冲区，性能飞跃
`mycat3` (缓冲区对齐)	266.8	页对齐内存，避免额外拷贝
`mycat4` (考虑块大小)	268.3	效果不明显，因页/块大小通常一致
`mycat5` (优化缓冲大小)	203.0	使用128KB大缓冲区，充分摊销开销
`mycat6` (使用 fadvise)	**198.6**	提示内核优化预读，性能最佳

核心问题回答

任务 3: 缓冲区对齐

1. **问：为什么页对齐能提速？实验支持吗?**
   答： 页对齐能让内核使用 **DMA (直接内存访问)** 进行I/O，避免了CPU参与的额外内存拷贝。实验结果**支持**，`mycat3` (266.8ms) 比 `mycat2` (276.8ms) 有约 3.6% 的性能提升。
2. **问：`malloc` 为何不保证页对齐?**
   答： 因为 `malloc` 是通用分配器。若强制页对齐会造成巨大的**空间浪费**（内部碎片），牺牲大多数场景下的效率。
3. **问：如何 `free` 对齐后的内存?**
   答： 分配时多申请一些内存，将**原始 `malloc` 指针存放在对齐后指针的正前方**。释放时，先从对齐指针处取出原始指针，再对原始指针调用 `free`。

任务 4: 文件系统块大小

1. **问：为何要考虑块大小?**
   答： 为了让缓冲区大小成为**文件系统I/O基本单位（块）的整数倍**，从而避免底层发生低效的"读-修改-写"操作。
2. **问：如何处理"块大小不一"和"虚假块大小"?**
   答：
   - **块大小不一**：用 `stat()` 获取**当前文件自身**的块大小 `st_blksize`，不做全局假设。
   - **虚假块大小**：对获取的 `st_blksize` 做**健全性检查**，若数值不合理（如过小或过大），则回退到安全的默认值（如 4096）。

任务 5: 优化缓冲区大小

1. **问：实验脚本如何设计？结论是什么？**
   答：
   - **设计**：使用 `dd if=/dev/zero of=/dev/null bs=<size>` 测试不同缓冲区在纯内存I/O下的吞吐率，排除物理磁盘影响，找到摊平系统调用开销的性能"拐点"。
   - **结论**：实验数据表明，缓冲区在 **128KB** 左右性能趋于饱和。因此 128KB 是兼顾性能与内存占用的最优大小。

BufferSize	Throughput(MB/s)
1K	5734
2K	9830
4K	15053
8K	20070
16K	22938
32K	26931
64K	27955
128K	**29184**
256K	**29286**
512K	29286
1M	30003
2M	29798
4M	28365

任务 6: 使用 `fadvise`

1. **问：`fadvise` 参数如何设置？**
   答： 调用 `posix_fadvise(fd, 0, 0, POSIX_FADV_SEQUENTIAL)`。这告诉内核，我们将对**整个文件进行顺序读取。**
2. **问：`readahead` 如何调整？**
   答：
   - **顺序读**：内核会**增大**预读窗口，更激进地提前将文件内容加载到内存。
   - **随机读**：内核会**禁用或最小化**预读，避免用不会被访问的数据污染缓存。

任务 7: 总结

1. **问：结果符合预期吗？有什么启示？**
   答：
   - **符合预期**：性能提升最大的步骤是**引入大缓冲区**（`mycat1` -> `mycat2`，`mycat4` -> `mycat5`），这验证了减少系统调用是I/O优化的关键。

- **启示**：
    1. **抓主要矛盾**：优化的第一步永远是减少系统调用的数量。
    2. **理论结合实验**：理论（页/块大小）提供方向，但最优参数（128KB缓冲区）需要靠实验数据确定。
    3. **与内核协作**：用户空间优化有极限，通过 `fadvise` 等工具与内核协作可以进一步榨取性能。