AlphaFileSystem 开发文档

1. 概述

文件系统能够实现创建文件，读文件，写文件，复制文件，查看文件数据块内容。

本文件系统使用了两个 file-manager 及 3 个 block-manager，用户在进行操作时需要指定 file-manager，不同的 file-manager 之间无法拥有读写对方文件的权限。

索引使用 file-manager id 和文件名结合进行。

文件的存储是分块存储的模式，默认块大小可通过 DEFAULT_BLOCK_SIZE 常量调整且可指定块大小。每个 block 所从属的 block-manager 通过随机数分配。block 通过 block-manager id 及 block id 结合索引。block 有根据 block 数据内容计算的校验码来在读取数据时判断 block 中的内容是否损坏。

文件的 meta 数据中存储的内容有文件的大小，块大小及含有文件数据的 block 列表。block 的 meta 数据中存储 block 大小及 block 的校验码。

文件的读写支持从指定位置进行，File 接口中存储有代表位置的指针，可以从控制台读取用户输入并调整该指针来进行位置的调整。如果用户输入了大于文件 size 的指针来进行文件写，则中间的字节将用 0x00 填充。

文件可以进行 size 的调整，减小 size 则将减小部分的数据所对应的块从文件的 meta 数据中删除，增加 size 则用 0x00 填充增加的部分。

文件的复制通过 meta 文件的复制进行。

理论上，block 始终存在一致性，所有的操作都不对已经有过写操作的 block 进行更改，而通过对文件的 meta 数据的操作来调整文件内容。

系统支持 duplication，通过 LogicBlock 实现，每个 LogicBlock 中存储有三个内容相同的 block 副本。

系统支持 buffer，即系统中存储有最近使用的块的链表，当读或写文件时可以从链表中获取该块而无需进行文件 IO。

2. 实现

分块存储通过对文件数据的拆分进行。遍历数据的 byte 数组并以 blocksize 常量切分数据，分块写入磁盘，且写入时自动创建多个副本存储进同一个 LogicBlock。

对于从某个 Block 中部开始的写(包括 setsize 时大小从某个 block 中部截断)，先用当前游标与 blocksize 取余数来获取 offset，然后读取该 offset 长度的数据，将当前 block 从 LogicBlock 列表中删除并且重新将读取的部分数据写入文件系统。

读文件时先反序列化 meta 文件中的内容来初始化 LogicBlockList，之后读取每个 LogicBlock 并初始化 LogicBlock 中的 physicalblock，只要 physical block 校验成功则读取该 block 中的数据并跳出循环，否则继续校验下一个块，如果所有块都校验失败则抛出异常。

用户读和写文件时都可以设置游标来设置读写位置。若用户读文件时输入的长度与当前游标之和大于文件大小则抛出异常。

checksum 的实现参考了 lab 的文档中的方法，在创建 physical block 时创建 checknum，读取时重新计算并与 meta 文件中存取的 checksum 进行对比来确定块是否被损坏。

meta 数据的存储参考了 lab 文档中的示例数据并实现了对该数据的序列化及

反序列化。

buffer 通过简单的链表实现，使用某个块时将该块放到链表头部，当链表大小到达阈值时把链表尾部的块写入实际存储单元。用户调用 close 时将链表清空，将所有块进行写入。

3. ·打开文件且读数据的过程概述

首先从控制台读取文件管理器和文件名和读取位置，创建一个 AFSFile 对象 file。创建过程中根据 meta 文件初始化 File 的 size, blocksize, LogicBlockList。调用 move 方法移动 cursor，然后调用 file 的 read 方法。该方法根据游标定义到文件 LogicBlockList 的当前 LogicBlock，依次读取每个 LogicBlock，调用 LogicBlock 的 getBlockList 方法获取其中存储的 PhysicalBlock, 同理通过 meta 文件初始化 PhysicalBlock, 并调用 PhysicalBlock 的 read 方法读取有效块的数据，将有效块的数据依次写入结果，最后返回给主函数。

·写文件/创建文件的过程概述

调用 AlphaWrite 工具时首先从控制台读取文件管理器和文件名，判断该文件是否存在。若存在则进行下一步，不存在则调用 AFSFileManager 的 newFile 方法创建新的文件，对 meta 文件进行初始化。下一步是创建 AFSFile 对象，同样根据 meta 文件来初始化文件数据，根据游标找到需要写的位置。如果要写写的块内已有数据则将数据先读出，与即将放入新块的 byte 数组进行拼接，且不对之前的块进行更改。之后遍历数组，以块大小分割数组并创建新的 PhysicalBlock，更改 LogicBlockList 将新块放入。