作业 15

毕定钧 2021K8009906014

本次作业包含:

15.1 某个文件系统在磁盘上保存了一个大小为 20 KB 的文件 A，现有一个进程打开文件 A，并调用 $write$ 函数一次性向文件 A 的文件块 0 和文件块 1 写入新数据。假设该文件系统使用文件缓存，且宕机可能发生在任意时刻。请分析

(1) 如果文件系统采用数据日志，宕机恢复后，文件 A 的内容是什么？请分不同情况讨论 (即在什么样的宕机情况下，文件 A 的内容是什么);

(2) 如果文件系统采用元数据日志，并且采用先改数据再改元数据的方式，宕机恢复后，文件 A 的内容是什么？请分不同情况讨论 (即在什么样的宕机情况下，文件 A 的内容是什么)。

15.2 LFS 的 $imap$ 和 CR 都采用类似数组的结构，下标是 ino 或 $imap$ 块号，每一项保存对应 $i-node$ 或 $imap$ 块的磁盘地址。例如，$imap[k]$ 记录 ino 为 k 的 $i-node$ 的磁盘地址；$CR[n]$ 记录第 n 个 $imap$ 块的磁盘地址。假设一个 LFS 的块大小为 $4KB$，磁盘地址占 $4B$。如果已经分配了 200 万个 $i-node$，请问：

(1) 该 LFS 的 $imap$ 有多少个块？请给出计算过程;

(2) 该 LFS 的 CR 有多少个块？请给出计算过程;

(3) 如何查 $ino = 654321$ 的 $inode$ 的磁盘地址？请给出查找和计算过程。

15.3 一个 LFS 的块大小为 $4KB$，$segment$ 大小是 $4MB$。文件块采用多级索引，即包含 10 个直接指针，以及一、二、三级间接指针各 1 个。每个指向数据块的指针占 4 字节。该 LFS 中已经有一个 $10MB$ 的文件 foo，请分析：

(1) 给出文件 foo 的文件块索引结构，即文件 foo 使用了哪些指针？

(2) 写文件 foo 的第 2560 块 (假设它在磁盘块 Ai 中，Ai 为磁盘逻辑块号)，需要写哪些块？需要几次 I/O？请给出它们写在磁盘上的顺序;

(3) 如果是 $Fast\ FS$ (其块大小也为 $4KB$)，写文件 foo 的第 2560 块，需要写哪些块? 需要几次 I/O？

(4) 如果是日志文件系统，只记录元数据日志，且日志不采用批量提交，则写文件 foo 的第 2560 块，需要写哪些块？需要几次 I/O？

15.1

(1)

在提交日志 $commit$ 并向磁盘写入 TxE 之前发生宕机，会导致系统重启后由于检查到事件缺失 TxE 标志而根据事件日志进行回滚操作，从而使修改丢失，此时文件 A 的内容仍为修改前的数据，或是上一个 $Checkpoint$ 的数据。

在向磁盘写入 TxE 之后，即使是在磁盘修改完成并删除日志或在 $Checkpoint$ 前发生宕机，此时系统检测到完整的事件日志，会根据修改日志进行磁盘中文件的修改，由于日志文件记录的操作是幂等的，此时也视为修改有效，文件 A 内容已经修改完成，内容为修改完成后的内容。

(2)

在写数据块的过程中如果发生宕机，此时仍未写入 TxB，导致数据修改了而元数据没有更新，仍然指向修改前的数据，此时文件 A 的内容为修改前的内容，新写入的数据块被视为垃圾块。同样的，在提交日志 $commit$ 并向磁盘写入 TxE 前发生宕机，系统均会进行回滚操作，使得文件 A 的元数据仍为修改前的元数据，指向原来的数据，文件 A 的内容为修改前的内容。

而在磁盘中写入 TxE 后，此时开始写入元数据，此时宕机会导致元数据与数据不一致，也可能出现有一个块是正确的而另一个块的元数据异常。如果系统采取事务性的机制，可能会要求进行回滚，撤销写入操作。由于很有可能导致了文件系统不一致、元数据不一致、数据不完整等，系统有可能会需要进行恢复工作来保证文件系统的一致性，由于部分元数据已被覆写，有一定可能会发生数据损坏。系统也有可能会根据元数据日志执行操作，继续修改元数据，从而使文件 A 的内容为修改后的内容。

在清除日志后，此时数据与元数据均已写入磁盘，文件 A 的内容为写入完成后的内容。

15.2

(1)

一个 LFS 的块可以存储 $\frac{4KB}{4B} = 1024$ 个磁盘地址，故 200 万个 $i-node$ 需要 $\lceil \frac{2,000,000}{1024} \rceil = 1954$ 块 $imap$。

(2)

CR 用于记录每个 LFS 块的磁盘地址，故 1954 个 $imap$ 地址需要 $\lceil \frac{1954}{1024} \rceil = 2$ 块。

(3)

进行除法 $\frac{654321}{1024}$，得到商 638 和余数 1009，故该 ino 的 $inode$ 存储在第 639 块 $imap$ 块上，从而存储在第 1 块 CR 块上，因此先取第一块 CR 的第 639 个 $inode$，即 $CR[638]$，之后寻找该 $imap$ 块，并取 $imap[1008]$ 得到 $ino = 654321$ 对应的的磁盘地址。

15.3

(1)

文件 foo 使用了 10 个直接指针（$40KB$）、1 个一级间址指针（$4MB$）、1 个二级间址指针（$4GB$ 中的 $5MB + 984KB$）。

(2)

由于文件 foo 仅使用了 2560 个磁盘块，即写最后一个磁盘块，首先需要输入写数据，之后寻找空闲磁盘块并写入这一块数据，然后在这块数据后写入 $inode$，之前的第 2560 块及其后的 $inode$ 成为垃圾块，还需要更新 $imap$ 块。与此同时还可能需要写日志块，首先 $input$ 写数据、$input\ inode$、$output\ imap$，需要 3 次 I/O 如果需要日志，还会 $input\ TxB$、$input$ 写日志、$input\ bitmap$、$input\ TxE$，需要 7 次 I/O。

(3)

由于文件 foo 仅使用了 2560 个磁盘块，即写最后一个磁盘块，首先需要输入写数据，之后寻找空闲磁盘块并写入这一块数据，然后在这块数据后写入 $inode$，之前的第 2560 块及其后的 $inode$ 成为垃圾块，还需要更新 $bitmap$，共 3 次 I/O。

(4)

除了 (3) 中的三次以外，还需要 $input\ TxB$、$input$ 写日志、$input\ bitmap$、$input\ TxE$，需要 7 次 I/O。