操作系统课程作业5

PE20060014王晨HW5

1. Consider a RAID organization comprising five disks in total, how many blocks are accessed in order to perform the following operations for RAID-5 and RAID-6?

a. An update of one block of data

RAID-5: 2 blocks are accessed　　RAID-6: 2 blocks are accessed　(这题中算访问block的个数，不看次数)

b. An update of seven continuous blocks of data. Assume that the seven contiguous blocks begin at a boundary of a stripe.

RAID-5: 10 blocks are accessed　　RAID-6: 15 blocks are accessed

2. Suppose that a disk drive has 5,000 cylinders, numbered 0 to 4999. The drive is currently serving a request at cylinder 2150, and the previous request was at cylinder 1805. The queue of pending requests, in FIFO order, is: 2069, 1212, 2296, 2800, 544, 1618, 356, 1523, 4965, 3681 Starting from the current head position, what is the total distance (in cylinders) that the disk arm moves to satisfy all the pending requests for each of the following disk-scheduling algorithms?

The FCFS schedule is 2150, 2069, 1212, 2296, 2800, 544, 1618, 356, 1523, 4965, 3681. The total seek distance is 13,011.

The SSTF schedule is 2150, 2069, 2296, 2800, 3681, 4965, 1618, 1523, 1212, 544, 356. The total seek distance is 7586.

The SCAN schedule is 2150, 2296, 2800, 3681, 4965, 4999, 2069, 1618, 1523, 1212, 544, 356. The total seek distance is 7492.

SCAN和LOOK的区别在于SCAN需要扫到头/尾再转回。

The LOOK schedule is 2150, 2296, 2800, 3681, 4965, 2069, 1618, 1523, 1212, 544, 356. The total seek distance is 7424.

The C-SCAN schedule is 2150, 2296, 2800, 3681, 4965, 4999, 0 , 356, 544, 1212, 1523, 1618, 2069. The total seek distance is 9917.

The C-LOOK schedule is 2150, 2296, 2800, 3681, 4965, 356, 544, 1212, 1523, 1618, 2069. The total seek distance is 9137.

3. Explain what open-file table is and why we need it.

打开文件表是由操作系统维护的，位于内核空间的，用于记录打开文件信息的一张表，每个文件的条目包含了文件指针，文件打开计数，文件磁盘位置，访问权限等信息。由于许多文件操作需要搜索目录，以得到命名文件的相关条目。为了避免这种不断的搜索，OS采用打开文件表来维护所有打开文件的信息，当请求文件时，可以通过该表的索引指定文件，而不需要搜索。当文件不再使用时，进程关闭它，并从打开文件表中删除它的条目。

4. Explain the concept of file and directory, and what does “755” mean for file permission?

文件是操作体系提供的，用于信息存储的一种统一的逻辑视图。从OS的视角来说，OS对存储设备的物理属性加以抽象，从而定义逻辑存储单元，即文件。从用户角度看，文件是逻辑外存的最小分配单元。

目录可视为符号表，可将文件名称转化成目录条目，目录也是文件。一个目录文件记录着所有属于它的文件（包括属于它的子目录）。

755表示对于文件所有者，拥有rwx全部权限，对于文件的组其他用户，拥有rx读访问、执行权限，但没有写访问权限。

5. Explain the problems of using continuous allocation for file system layout and how to solve them.

对于文件的创建，会有外部碎片的问题导致空间浪费，也就是即使有足够的空闲空间，但因为是连续分配方式，由于这些空闲空间并不连续，导致无法写入新文件。对于外部碎片问题，可以通过移动原有文件，合并空闲空间解决，但是这样做的代价是非常大的。

另一个问题是文件分配空间大小的拓展，对于有些文件，可能在一开始无法确定它需要多少空间，但在连续分配的方法下，这个文件的两端空间都已被使用，这时候就无法再拓展空间了。解决方法采用一些修正，比如最初分配一块连续空间，当数量不够时，就在添加另一块连续空间（成为扩展extent）然后用指针记录下一块的位置。链接分配可以解决上述连续分配的所有问题。

6. What are the advantages of the variation of linked allocation that uses a FAT to chain together the blocks of a file? What is the major problem of FAT?

优点是当我们需要访问一个位于文件中间的block时，我们可以通过FAT表中的指针信息来确定它的具体位置，而不需要顺序遍历文件的每一个block来找到那个指向目标block的指针。也就是说，我们只需要搜索FAT表就可以找到目标block的位置，而FAT相对更小，可以存放在内存中。

FAT的缺点是，占用了更多的空间用于存放指针，同时FAT表必须缓存在内存，否则将导致大量的磁头寻道时间。另外也同样有可靠性问题，也就是指针的丢失或者错误可能会导致无法链接到相应文件或者错误链接到其他文件。

7. Consider a file system similar to the one used by UNIX with indexed allocation, and assume that every file uses only one block. How many disk I/O operations might be required to read the contents of a small local file at */a/b/c* in the following two cases? Should provide the detailed workflow.

a. Assume that none of the disk blocks and inodes is currently being cached.

4 disk I/O operations:

1. reading in the disk block containing the root directory */*
2. reading in the disk block containing the directories a
3. reading in the disk block containing the directories *b*
4. reading in the disk block containing the file *c*

b. Assume that none of the disk blocks is currently being cached but all inodes are in memory.

3 disk I/O operations:

The i-nodes are already cached in memory (in the file buffer), so they do not need to be read from disk. Only the actual data blocks need to be read from disk.

(1). We know that the root directory in Unix has i-node number 2. This i-node is in memory so this doesn't yet involve a disk read. The i-node contains the pointers to the data-blocks for the a directory. Given the assumption that each directory file is only one block long, we only have to read one data-block to read the contents of the directory. This is our first disk read. In this data, we search for directory "a" and get its i-node number, x.

(2). We look at i-node number x (in memory) and get the block number (i.e., the pointer) for the data. We read this data (the second read), search this directory for the directory "b" and get its i-node number, y.

(3). We look at i-node number y (in memory) and get the block number for the data. We read this data (the third read) and search this directory for the file "c"; we see its i-node number and are done.

8. Consider a file system that uses inodes to represent files. Disk blocks are 8-KB in size and a pointer to a disk block requires 4 bytes. This file system has 12 direct disk blocks, plus single, double, and triple indirect disk blocks. What is the maximum size of a file that can be stored in this file system?

Block size = 8KB, Address length = 4B ==> A block can contain 2^11=2048 address. Thus the maximum size = 12*8KB+2048*8KB+2048*2048*8KB+2048*2048*2048*8KB ≈ 64 TB

9. What is the 8+3 naming convention in FAT32 file system, and how to manage long filenames?

8+3 naming convention: 8 characters for name + 3 characters for file extension。长名文件采用LFN directory entry解决，也就是增加更多的entry来表示文件名。具体方式是有一个normal entry和若干个LFN entry来存储文件名（从尾到头的顺序）。其中normal entry仍然是8+3。LFN entry的1-10，14-25，28-31位分别存放5，6，2个文件名字符的unicode码，第11位为x0f表示这是LFN，第1位为0x40打头表示最后一个LFN。

10. How are directory entries managed in FAT and Ext file systems?

In FAT, directory entry is just a structure which includes filename, all file attributes, start cluster number and file size, etc.

In EXT, directory entries store the the inode number, file name, file type, record length, offset to the next directory entry.

11. What is the difference between hard link and symbolic link?

A hard link is a directory entry pointing to an existing file. Conceptually speaking, this creates a file with two pathnames. They point to the same inode (no new file content) .

In contrast, a symbolic link is a file which stores the pathname(shortcut). Unlike the hard link, a new inode is created for each symbolic link.

12. What are the initial link counts when a regular file or a directory is created? Why?

When a directory is created, the initial link count is always 2 because the hosting directory and the newly creating directory itself will both have its link.

When a regular file is created, the link count is always 1 because only the the hosting directory of the file has the link.

13. What is the difference between data journaling and metadata journaling? Explain the operation sequence for each of the two journaling methods.

The difference is that data journaling writes every DB to disk twice while Metadata journaling only logs metadata.

Data Journaling operation sequence:

1) Journal write ——Write the contents of the transaction (including TxB, metadata, and data)

2) Journal commit ——metadata, and data (including TxE)

3) Checkpoint—— Write the contents of the update to their on-disk locations

Metadata Journaling operation sequence: (Data write -> Journal metadata write can be issued in parallel)

Data write -> Journal metadata write -> Journal commit -> Checkpoint metadata

14. What are the three I/O control methods?

轮询（不断问设备有没有请求）、中断（有事再叫我处理）、DMA方式（设备通过DMA控制器直接访问内存）。

15. List at least three kinds of I/O devices and explain how to provide a standard and uniform application I/O interface?

In generally speaking, a standard and uniform I/O interface involves Abstraction, encapsulation, layering (抽象，封装，分层). For example,

Block devices include disk drives ——Commands include read, write, seek

Character devices include keyboards, mice, serial ports ——Commands include get( ), put( )

Network devices ——socket interface

16. What services are provided by the kernel I/O subsystem?

I/O scheduling

Buffering - store data in memory while transferring between devices

Caching - faster device holding copy of data

Spooling - hold output for a device

Error handling and I/O protection

Power management, etc.