Disk Tutorial and Revision

- this section is best read using the pdf version of the slides

Disk Tutorial Questions

- 1. A FAT32 based system has a cluster size of 32768 (32 * 2^10) bytes. What is the theoretical maximum capacity of the disk? Note that only 28 bits of each 32-bit FAT entry can be used for file allocation. Express your answer in TBytes (use 1 TByte = 2^40 Bytes). Check your answer at ⟨http://support.microsoft.com/kb/184006/en-us⟩

Disk Tutorial Questions

- 2. A file system uses linked allocation with a cluster size of 512 bytes and a pointer size of 4 bytes. How many clusters are required to store the following files?

- ```
  MEMO.TXT-4502 bytes
  PROJ.EXE-281750 bytes
  ```

- If the disk allocation method was changed to indexed allocation, with each index entry requiring 4 bytes, how many clusters are now required to store these files?
  ```

Disk Tutorial Questions

- 3. Using appropriate Web references, outline the major characteristics of NTFS and discuss its improvements over the older FAT based system. Is FAT still used? Where and why?

Disk Tutorial Questions

- 4. Outline the differences in approach taken by the DOS/Windows FAT system and the Unix inode with respect to the way in which both operating systems allocate disk space. Your answer should include a description of the operation of the DOS FAT, the structure of a DOS directory entry and their relationship. You should then describe the Unix inode data structure, the structure of a Unix directory entry and their relationship.

Disk Tutorial Questions

- 5. Assuming a block size of 2048 bytes and an address pointer size of 4 bytes, what would be the size of a Unix file which completely utilises the capacity of:

- ```
  a. the first ten inode pointers?
  b. the first eleven inode pointers?
  c. the first twelve inode pointers?
  d. the first thirteen inode pointers?
  ```
  ```

Disk Tutorial Questions

- 6. Demonstrate the concept of disk mounting in a Unix file system with a suitable diagram and relavent discussion. How does this compare with the concept of a Window network drive. Outline the similarities and differences.

Some tutorial answers

- Answer to question 1.
  - this section is best read using the pdf version of the slides

- 2^{28} possible clusters and each cluster is 32768 bytes

$$32768 = 32 \times 2^{10}$$

$$32 = 2^{5}$$

$$2^{28} \times 32 \times 2^{10} = 2^{28} \times 2^{5} \times 2^{10} = 2^{43}$$

Some tutorial answers

- 1 TB = 2^{40}
- max FAT32 size in TB = $\frac{2^{43}}{2^{40}} = 2^3 = 8$ TB

Some tutorial answers

- Answer to question 2
- linked allocation (last 4 bytes in each cluster are a pointer)
- each cluster is 512 bytes, so it can contain 512 – 4 = 508 bytes and a 4 byte pointer
- 4502 byte file requires ($\frac{4502}{508}$ = 8 remainder 438 clusters), so 9 clusters
- 281750 byte file requires ($\frac{281750}{508}$ = 554 remainder 318), so 555 clusters

Some tutorial answers

- Indexed allocation
- a single cluster can contain $\frac{512}{4}$ pointers (128 pointers)
  - one will be used to chain to the next indexed cluster (if required), so we only have 127 data pointers
- a single cluster can therefore point to the contents of a 128×512 byte file = 64 K bytes
  - a file exceeding 64KB will require another cluster to contain the next 64KB worth of data
- 4502 file requires $\frac{4502}{512} + 1 \approx 8.79 + 1$

Some tutorial answers

- therefore will require 10 clusters

Some tutorial answers

- a file of size 281750 bytes using indexed allocation will require
- multiple indexed clusters and multiple data blocks
- no of index clusters: $\frac{281750}{64 \times 1024} = 4.29 \approx 5$
- no of data clusters: $\frac{281750}{512} = 550.2 \approx 551$
- total = $5 + 551 = 556$ clusters

Some tutorial answers

- Answer to question 5a

- Assuming a block size of 2048 bytes and an address pointer size of 4 bytes, what would be the size of a Unix file which completely utilises the capacity of:
  - a. the first ten inode pointers?

- examine the inode diagram in the filesystem notes
  - we see that the first ten blocks use direct pointers
  - $10 \times 2048 = 20480$ byte file

Answer to question 5b

- first eleven inode pointers
- first ten = 20480 as calculated before
  - the eleventh is a single indirect inode, which means that this block contains $\frac{2048}{4} = 512$ pointers to data blocks
    - a single indirect block can point to a total of 512×2048 bytes
    - 1048576 bytes
- therefore first eleven inode pointer can contain a maximum file size of: $1048576 + 20480$ byte
  - 1069056 bytes

Answer to question 5c

- c. the first twelve inode pointers?
  - again we can use the prior calculation of the first eleven and add the twelfth
  - 1069056 + size of twelfth

- the twelfth uses a double indirect pointer
  - so this means that has 512 pointers which point to indirect pointer (as before)
  - $512 \times 1048576 = 536870912$

- so the maximum file size is $1069056 + 536870912 = 537919488$ bytes

Answer to question 5d

- d. the first thirteen inode pointers?
  - the thirteenth pointer is a triple indirect, which means that it points to a block containing 512 pointers which point to double indirect inodes
  - therefore $512 \times 536870912 = 274877906944$ bytes for a tripple indirect inode

- so the total is: size of the first twelve + size of the thirteenth
  - $537919488 + 274877906944 = 275415826432$ bytes

Revision

- revise **all** topics
- for the spring term material pay special attention to:
  - processes, process scheduling, file systems.