A file system uses 256-byte physical blocks. Each file has a directory entry giving the file name, location of the first block, length of file, and last block position. Assuming the last physical block read is 100, block 100 and the directory entry are already in main memory.

For the following two file allocation algorithms, how many physical blocks must be read to access the specific block 600 (including the reading of block 600 itself), and why?

(1) contiguous allocation

(2) linked allocation

(1) 1

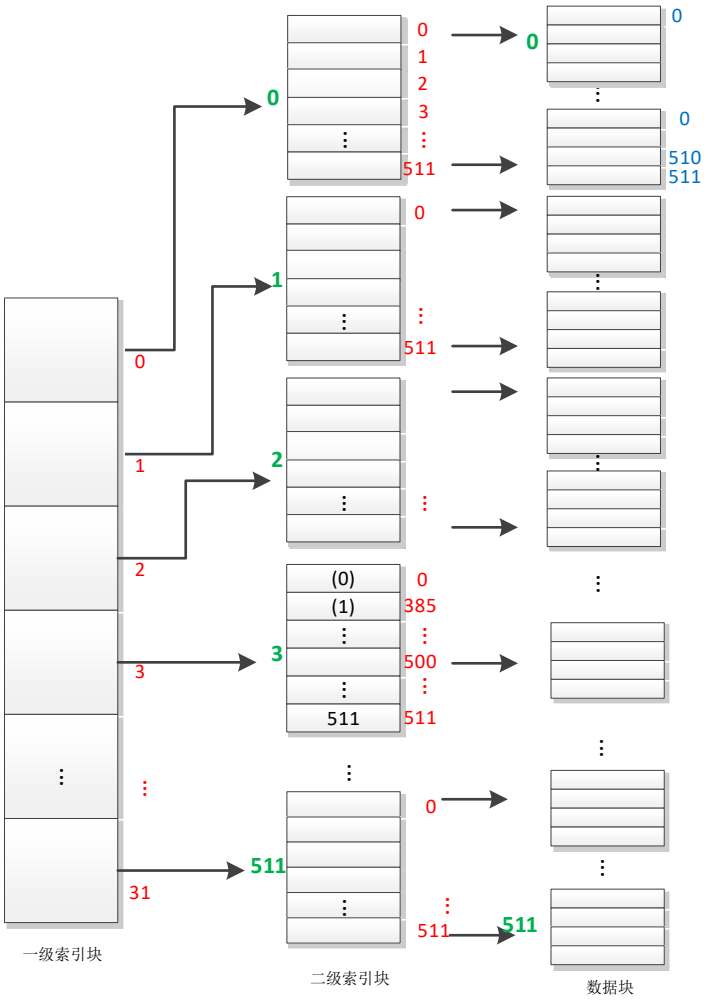
Contiguous allocation supports direct access on arbitrtary physical blocks in a file. To access physical block 600, the bolck 600 can be directory read into memory.

(2) 500

Linked allocation is used only for sequential-access files, file system can only read blocks one by one, directed by file pointers.

To find the phyical block 600 in the file, file system locates on the ***101th*** block following the ***100th*** block just read, and reads block 101, block 102, …, until block 600.

Consider a file system in which a directory entry can store up to 32 disk block addresses. For the files that is not larger than 32 blocks, the 32 addresses in the entry serves as the file's index table. For the files that is larger than 32 blocks, each of the 32 addresses points to an indirect block that in turn points to 512 file blocks on the disk, and the size of a block is 512-bytes. What is the largest size of a file?


0
1
2
3
31
一级索引块
0
1
2
3
511
二级索引块
(0)
(1)
511
0
385
500
511
0
511
0
510
511
数据块

32 x 512 =16384 file blocks

16384 x 512 = 8388608 bytes

or

2^5 x 2^9 x 2^9 = 2^{23} bytes

In the file system on a disk with physical block sizes of 512 bytes, a file is made up of 128-byte logical records, and each logical record cannot be separately stored in two different blocks. The disk space of the file is organized on the basis of indexed allocation, and a block address is stored in 4 bytes. Suppose that 2-level index blocks be used to manage the data blocks of the file, answer the following questions:

1) What is the largest size of the file?
2) Given 2000, the number of a logical record in the file, how to find out the physical address of the record 2000 in accordance with the 2-level index blocks.

1)

512/4=128

128*128*512=128*64 KB=8192KB

or $=2^{23}$ bytes = 8 MB

or = 8388680

2)

512/128=4

2000/4=500

in 3 block in the first –level index block

in 116 block in the second-level index block

A file is made up of 128-byte fix-sized logical records and stored on the disk in the unit of the block that is of 1024 bytes. The size of the file is 10240 bytes. Physical I/O operations transfer data on the disk into an OS buffer in main memory, in terms of 1024-byte block. If a process issues *read* requests to read the file's records in the sequential access manner, what is the percentage of the *read* requests that will result in I/O operations?

1024/128=8 record

10240/8=80 record

10240/1024=10 block

10/80=1/8=0.125

or 12.5%

Consider a paging system with the page table stored in memory.

a. If a memory reference takes 300 time unit, how long does it take to access an instruction or data in a page that has been paged into memory?

b. If we add TLB (*translation look-aside buffers*), and 80 percent of all page-table entries can be found in the TLB, what is the effective memory access time?

(Assume that finding a page-table entry in the associative registers takes 20 time unit, if the entry is there.)

a. 300 x 2=600

b. 0.8 x (300+20) + 0.2 x (600+20)=256+124=380