Chapter 8 Main Memory

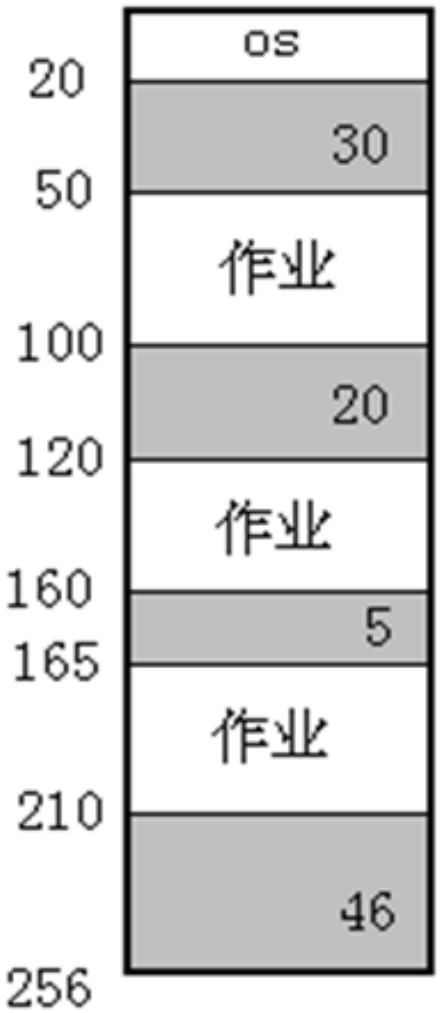
LI Wensheng, SCS, BUPT

* Exercise 1

某系统，内存状态如右所示。

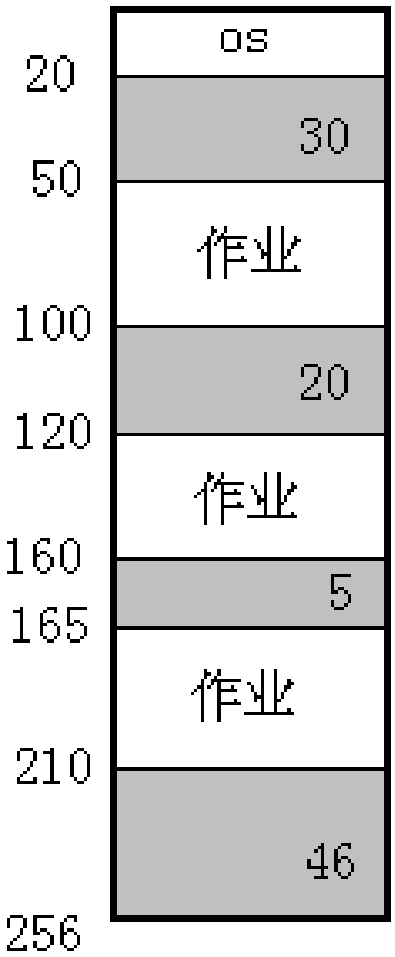
有如下作业序列：

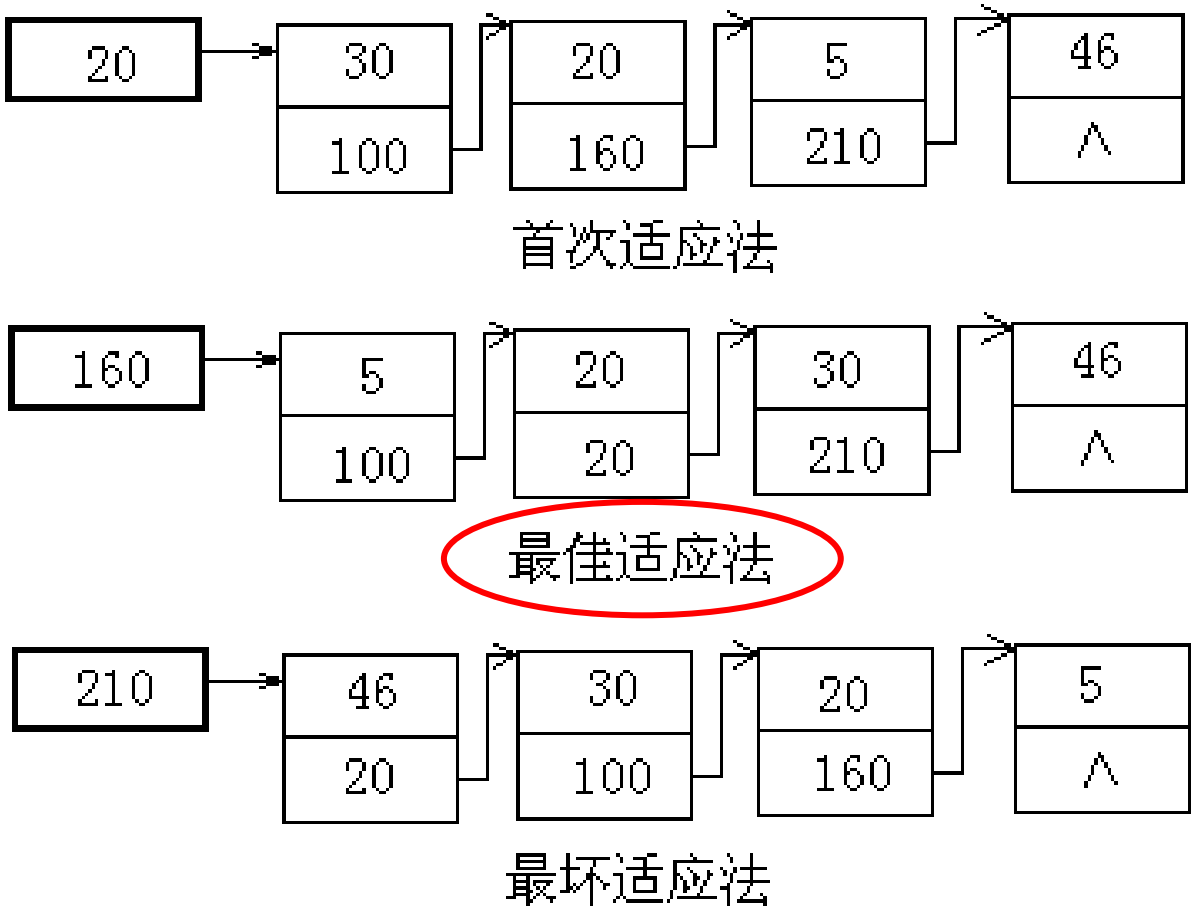
- A需要内存18M，B需要25M，C需要30M。
- 根据分析结果回答：哪种分配算法对此作业序列是合适的？



参考答案

- **A需要18M，B需要25M，C需要30M。**



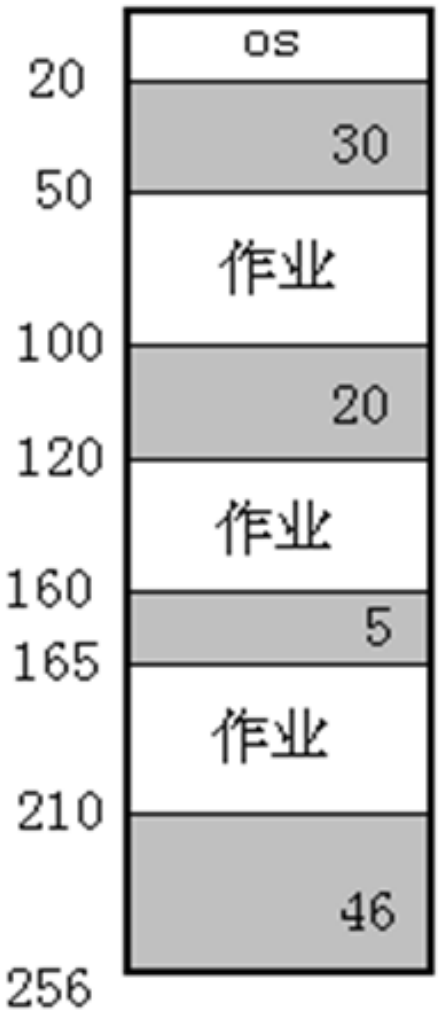


* Exercise 2

某系统，内存状态如右所示。

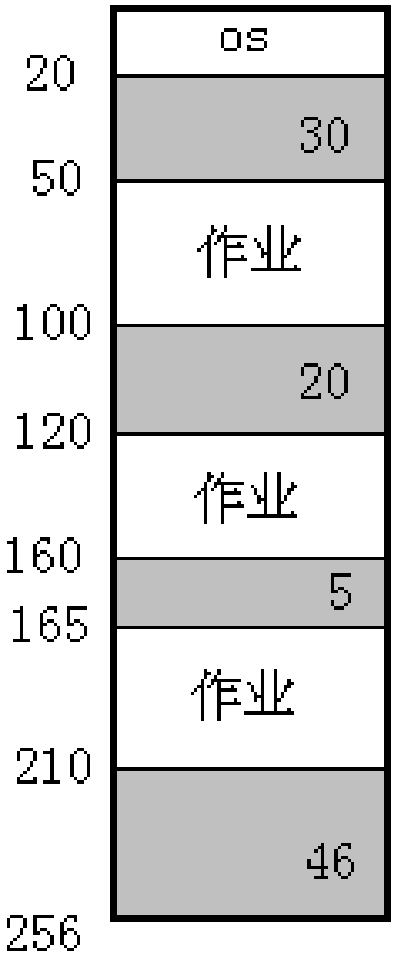
有如下作业序列：

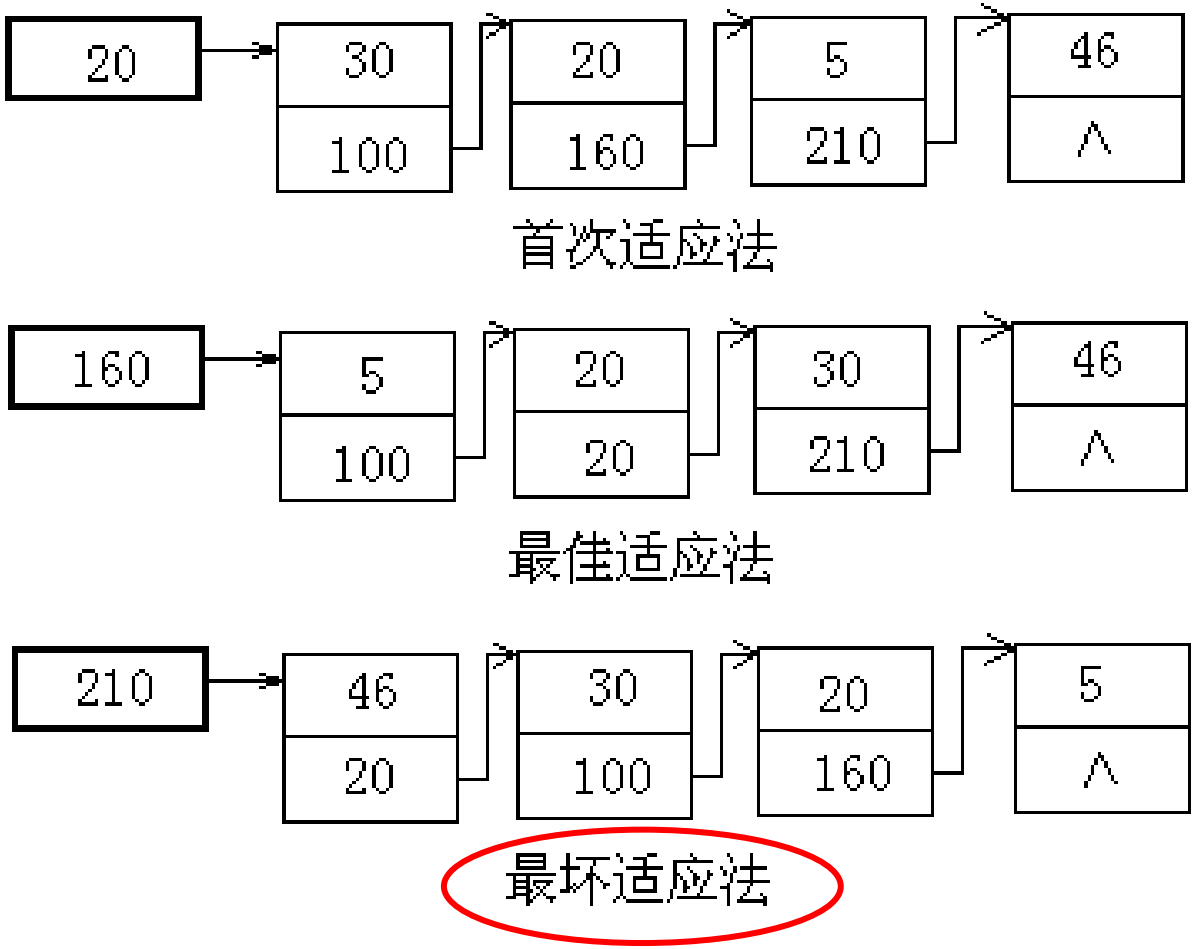
- A需要内存21M，B需要30M，C需要25M。
- 根据分析结果回答：哪种分配算法对此作业序列是合适的？



参考答案

- **A需要21M，B需要30M，C需要25M。**





* Exercise 3

- 用可变分区(动态重定位)方式管理内存时，假定内存中按地址顺序依次有5个空闲区，空闲区的大小依次为32M、10M、5M、228M和100M。

  现在有5个作业A、B、C、D、E。它们各需主存大小为：1M、10M、108M、28M和115M。

  请问：

  （1）若采用首次适应算法，能把这5个作业按顺序全部装入内存吗?

  （2）按怎样的次序装入这5个作业可使内存的利用率最高？

* Answer to Exercise 3

A: 1M
B: 10M
C: 108M
D: 28M
E: 115M

- **First fit:**
  - **Hole list: 32M、 10M、5M、228M、100M**
  - **Allocate: A(1) C(108)**
    B(10) D(28)
    21M 92M
  - **E(115M)?**
- **Best fit:**
  - **Hole list: 5M、10M、32M、100M、228M**
  - **Allocate: A(1) B(10) C(108)**
    4 120
    D(28) E(115)
    4 5
  - **Hole list: 4M、4M、5M、100M**

* Answer to Exercise 3 (Cont.)

A: 1M
B: 10M
C: 108M
D: 28M
E: 115M

- **Worst fit:**
  - **Hole list: 228M、 100M、32M、10M、5M**
  - **Allocate： A(1)**
    B(10)
    C(108)
    D(28)
  - **Hole list: 100M、81M、32M、10M、5M**
  - **E(115M)?**

Exercise 4:

- **Consider a system with physical memory of 2048 frames, the logical address space of a process is up to 32 pages, the page size is 2048 bytes.**
  - **(1) How many bits are there in the logical address?** 16
  - **(2) How many bits in the logical address refer to virtual page number?** 5
  - **(3) How many bits are there in the physical address?** 22
  - **(4) How many bits in the physical address refer to frame number?** 11
  - **(5) How many bits in the physical address refer to offset in a frame?** 11

Exercise 5:

- **A certain computer provides its users with a virtual-memory space of 2^{32} bytes. The computer has 2^{18} bytes of physical memory. The virtual memory is implemented by paging, and the page size is 4,096 bytes.**
  (1) How many bits are there in the logical address? 32
  (2) How many bits in the logical address refer to virtual page number? 20
  (3) How many bits are there in the physical address? 18
  (4) How many bits in the physical address refer to frame number? 6
  (5) How many bits in the physical address refer to offset in a frame? 12
  (6) A user process generates the virtual address 11123456, figure out its page number and page offset. 2715 2816
  Explain how the system establishes the corresponding physical location.

Exercise 6

- **Consider a paging system with the page table stored in memory.**
  - **if a memory reference takes 200 nanoseconds, how long does a paged memory reference take?**

400

  - **If we add TLBs, and 75 percent of all page-table references are found in the TLBs, what is the effective memory reference time? (Assume that finding a page-table entry in the TLBs takes 4 time, if the entry is there.)**

(4+200)*0.75+(4+200+200)*0.25=254

Exercise 7

- **Segment table：**

Segment no.	Limit	base
0	660	2219
1	140	3300
2	100	90
3	580	1237
4	960	1959

- **mapping logical address to physical address:**
  [0, 432]，[1, 10]，[2, 500]，[3, 400]

  2651　　3310　　invalid　　1637

- **Summarize the procedure that mapping a logical address to physical address.**

Chapter 9 Virtual Memory

LI Wensheng, SCS, BUPT

Exercise 1

- **On a system using paging, references to a swapped-in locations accessible through an entry in an associative table take 150ns. If the main memory page table must be used, the reference takes 400ns. References that result in page faults require 8ms if the page to be replaced has been modified, 3ms otherwise.**
  If the page fault rate is 2%, the associative table(TLB) hit rate is 70%, and 50% of replaced pages have been modified, what is the effective access time?
  Assume the system is running only a single process and the CPU is idle during page swaps.

Answer for exercise 1

1s=1000ms
1ms=1000μs
1μs=1000ns
1ns=1000ps

EAT=

(150ns*70% + 400*30%)* 98% +
(8ms*50% + 3ms*50%)*2%

= 100220.5ns

Exercise 2

- **Assume that we have a demand-paged memory. The page table is held in registers.**
  It takes 8 milliseconds to service a page fault if an empty frame is available or if the replaced page is not modified and 20 milliseconds if the replaced page is modified.
  Memory-access time is 100 nanoseconds.
  Assume that the page to be replaced is modified 30 percent of the time.
  What is the maximum acceptable page-fault rate for an effective access time of no more than 200 nanoseconds?

Answer for exercise 2

1s=1000ms
1ms=1000 μs
1 μs=1000ns
1ns=1000ps

200ns = (1-p)*100ns + (0.7p)*8*10^6 ns +(0.3p)*20*10^6 ns

100= -100p+5600000p+6000000p

100 = 11599900p

P ≈ 8.62*10^{-6}

Exercise 3

Consider the following page reference string:

1, 2, 4, 5, 3, 4, 1, 6, 8, 7, 8, 9, 7, 8, 9, 5, 4, 5, 4, 2.

How many page faults would occur for the following replacement algorithms, assuming there are three frames available.

Remember all frames are initially empty, so your first unique pages will all cost one fault each.

(1) LRU replacement

(2) FIFO replacement

(3) Optimal replacement

Answer for exercise 3

LRU:

	1	2	4	5	3	4	1	6	8	7	8	9	7	8	9	5	4	5	4	2
-	1	1	1	5	5		1	1	1	7		7				5	5			5
-	-	2	2	2	3		3	6	6	6		9				9	9			2
-	-	-	4	4	4		4	4	8	8		8				8	4			4
	F	F	F	F	F		F	F	F	F		F				F	F			F

Page fault: 13 times, page fault rate: 13/20

FIFO

	1	2	4	5	3	4	1	6	8	7	8	9	7	8	9	5	4	5	4	2
-	1	1	1	5	5		5	6	6	6		9				9	9			2
-	-	2	2	2	3		3	3	8	8		8				5	5			5
-	-	-	4	4	4		1	1	1	7		7				7	4			4
	F	F	F	F	F		F	F	F	F		F				F	F			F

Page fault: 13 times, page fault rate: 13/20

Answer for exercise 3

OPT

	1	**2**	**4**	**5**	**3**	**4**	**1**	**6**	**8**	**7**	**8**	**9**	**7**	**8**	**9**	**5**	**4**	**5**	**4**	**2**
-	1	1	1	1	1			6	6	7		7				7	4			4
-	-	2	2	5	3			3	8	8		8				5	5			5
-	-	-	4	4	4			4	4	4		9				9	9			2
	F	F	F	F	F			F	F	F		F				F	F			F

Page fault: 12 times, page fault rate: 12/20

Exercise 4

Page	Frame number	Valid/ invalid	Page in time	Reference time	Reference bit	Modify bit
0	60	1	225	326	1	1
1	35	0	100	105	1	
2	50	0	138	245	1	1
3	35	1	289	321	1	1
4		0				
5	50	1	312	387	1	0
6		0				
7	35	0	268	280	1	0

- A process has 8 pages, its page table is shown as above, assume three frames are allocated to this process.
- Now page 6 is needed,
- FIFO replacement algorithm is used, which frame will the page be paged into? **60 (now used by page 0)**
- LRU replacement algorithm is used, which frame will the page be paged into? **35 (now used by page 3)**

Exercise 5

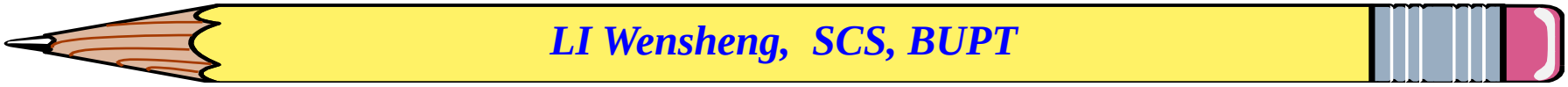
- **Consider a machine in which all memory-reference instructions have two memory addresses, and one-level indirect addressing is allowed; if an instruction is assumed to be stored in only one frame, then the minimum number of frames per process is _____. And briefly why?**

Answer to exercise 5

- **The minimum number of frames per process is 5.**
  - **1 page for instruction.**
  - **1 page for source address, which is an indirect reference to the source operand.**
  - **1 page for destination address, which is an indirect reference to the destination operand.**
  - **1 page for source operand.**
  - **1 page for destination operand.**

北京郵電大學

Chapter 11 File-System Implementation



LI Wensheng, SCS, BUPT

连续文件举例

LA=1248

1248/512 = 2 … 224

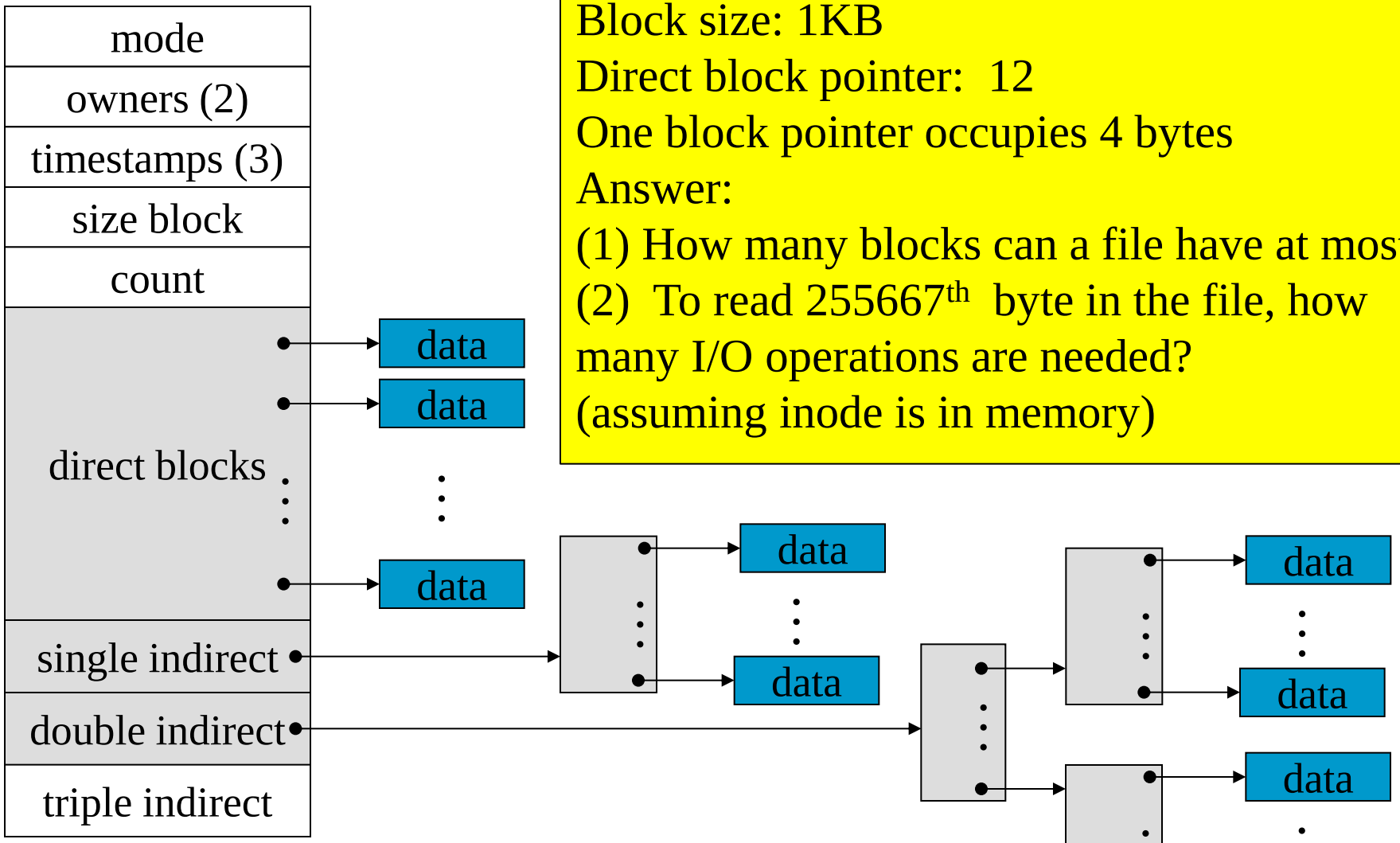
PA=32*512+224
=16608

例如：某文件系统，磁盘块大小为 512 B

- 字符流文件 A，长度为 1980 B
  - 文件A需要占用 4 个物理块。
  - 假如分配到30、31 、32和33四个相邻的物理块中。
  - 第33块中实际使用了444字节，剩余的68字节形成“内部碎片”。
- 记录文件 B，逻辑记录长 100 B，有23个记录
  - 假设：逻辑记录不能跨物理块存放。
  - 每个物理块中可以存放5个逻辑记录，需要5个物理块。
  - 假设分配到第 6、7、8、9、10 块
  - 前4块各有内部碎片12B，最后一块有212B没有使用。

LA=19
19/5=3 … 4
PA=9*512
+4*100
=5008

Combined Scheme: UNIX (4K bytes per block)



Block size: 1KB
Direct block pointer: 12
One block pointer occupies 4 bytes
Answer:
(1) How many blocks can a file have at most?
(2) To read 255667th byte in the file, how many I/O operations are needed?
(assuming inode is in memory)

Answer

每个索引块可以存放1024/4=256个块号。

(1) How many blocks can a file have at most?

12+256+256*256+256*256*256=16843020

(2) To read 255667th byte in the file, how many I/O operations are needed?

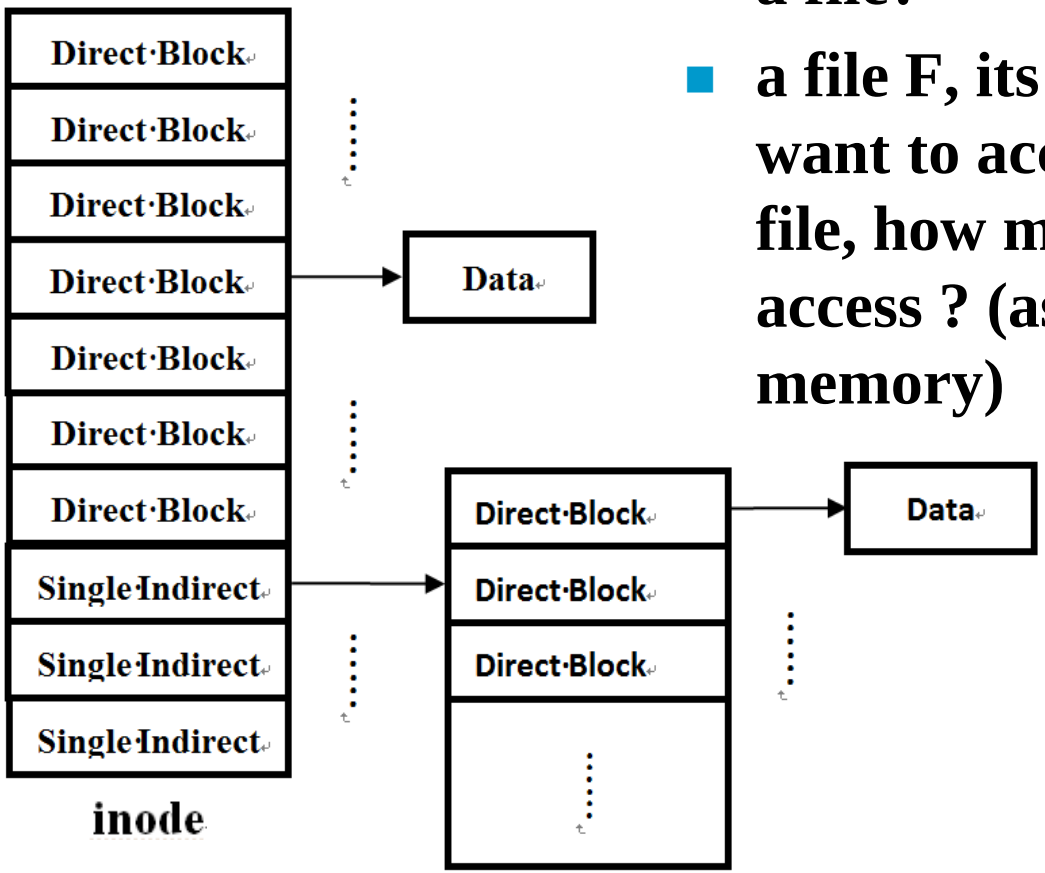
255667/1024=249 … 691

文件的第249块的块号存放在一级简介指针只想的索引块中。

所以，需要2次I/O操作，第一次I/O将索引块读进内存，找到相应的块号，第二次I/O，将字节255667所在的物理块读入内存。

exercise

- A file system, its allocation scheme is as follows :
- each file, only one block can be used as inode, in which, 10 block numbers can be hold at most, 7 point to data blocks and 3 Single indirect blocks.
  - How many blocks is the largest size of a file?
  - a file F, its size is 30 Blocks. If we want to access the third Block in this file, how many Blocks we need to access ? (assuming all Blocks is not in memory)



(1) 7+3*10=37
(2) 2

3. Consider a file currently consisting of 100 blocks. Assume that the file control block (and the index block, in the case of indexed allocation) is already in memory. Calculate how many disk I/O operations are required for contiguous, linked, and indexed (single-level) allocation strategies, if, for one block, the following conditions hold. In the contiguous-allocation case, assume that there is no room to grow in the beginning, but there is room to grow in the end. Assume that the block information to be added is stored in memory.

a. The block is added at the beginning.

b. The block is added in the middle.

c. The block is added at the end.

d. The block is removed from the beginning.

e. The block is removed from the middle.

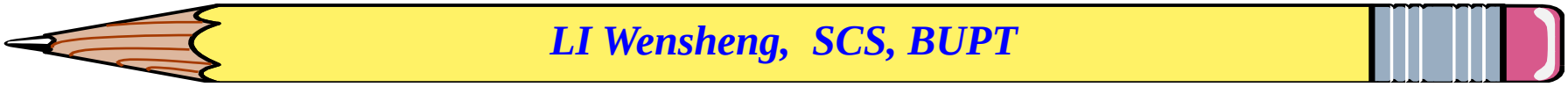
f. The block is removed from the end.

3. Answer

	contiguous	Linked	indexed
a	201	1	1
B	101	50+1+1=52	1
C	1	1+1+1=3	1
D	99*2=198	1	0
E	49*2=98	50+1+1=52	0
f	0	99+1=100	0

北京郵電大學

Chapter 12 Mass-Storage Structure



LI Wensheng, SCS, BUPT

exercise

- **Suppose that a disk drive has 5000 cylinders, number 0 to 4999. The drive is current serving a request at cylinder 143, and the previous request was at cylinder 94. The queue of pending requests, in FIFO order, is:**

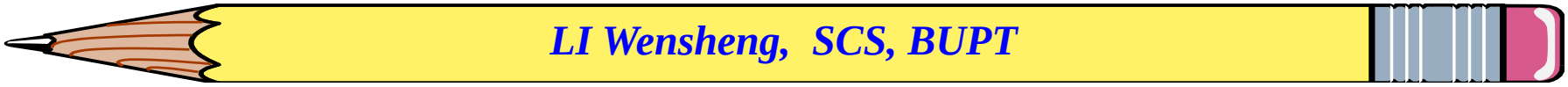
  86, 1470, 913, 1774, 948, 1509, 1022, 1750, 130

- **Starting from current position. If the total distance (in cylinders) that the disk arm moves to satisfy all the pending request is 9769, so what disk-scheduling algorithm may the system use? And why?**

Answer

- **FCFS：143, 86, 1470, 913, 1774, 948, 1509, 1022, 1750, 130.**
  寻道距离：7081
- **SSTF：143, 130, 86, 913, 948, 1022, 1470, 1509, 1750, 1774.**
  寻道距离：1745
- **SCAN：143, 913, 948, 1022, 1470, 1509, 1750, 1774, 130, 86.**
  寻道距离：9769
- **C-SCAN：143, 913, 948, 1022, 1470, 1509, 1750, 1774, 86, 130.**
  寻道距离：9985
- **LOOK：143, 913, 948, 1022, 1470, 1509, 1750, 1774, 130, 86.**
  寻道距离：3319

Chapter 13 I/O Systems



LI Wensheng, SCS, BUPT

Thinking the following question

- **The example of handshaking in polling used 2 bits: a busy bit and a command-ready bit.**
  Is it possible to implement this handshaking with only 1 bit? If it is, describe the protocol. If it is not, explain why 1 bit is insufficient.

Answer

It is possible, using the following algorithm. Let's assume we simply use the busy-bit (or the command-ready bit; this answer is the same regardless). When the bit is off, the controller is idle. The host writes to data-out and sets the bit to signal that an operation is ready (the equivalent of setting the command-ready bit). When the controller is finished, it clears the busy bit. The host then initiates the next operation.

This solution requires that both the host and the controller have read and write access to the same bit, which can complicate circuitry and increase the cost of the controller.

Thinking the following question

- **How does DMA increase system concurrency? How does it complicate hardware design?**
- **Answer:**

  DMA increases system concurrency by allowing the CPU to perform tasks while the DMA system transfers data via the system and memory busses.

  Hardware design is complicated because the DMA controller must be integrated into the system, and the system must allow the DMA controller to be a bus master.

  Cycle stealing may also be necessary to allow the CPU and DMA controller to share use of the memory bus.