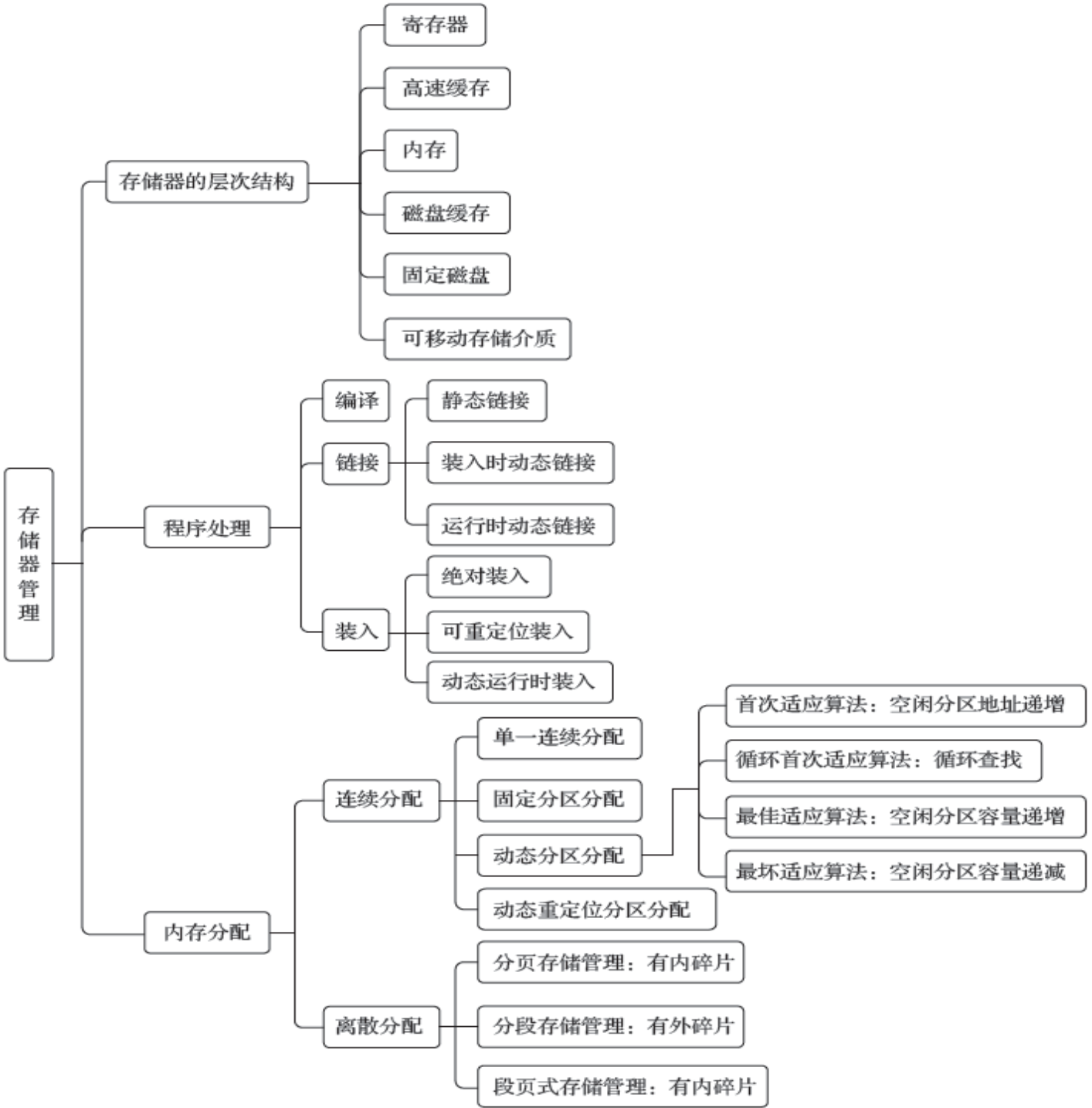

存储器管理
存储器的层次结构
寄存器
高速缓存
内存
磁盘缓存
固定磁盘
可移动存储介质
程序处理
编译
链接
静态链接
装入时动态链接
运行时动态链接
装入
绝对装入
可重定位装入
动态运行时装入
内存分配
连续分配
单一连续分配
固定分区分配
动态分区分配
首次适应算法：空闲分区地址递增
循环首次适应算法：循环查找
最佳适应算法：空闲分区容量递增
最坏适应算法：空闲分区容量递减
动态重定位分区分配
离散分配
分页存储管理：有内碎片
分段存储管理：有外碎片
段页式存储管理：有内碎片

海南大学
HAINAN UNIVERSITY

《操作系统原理及安全》

第6章 虚拟存储器

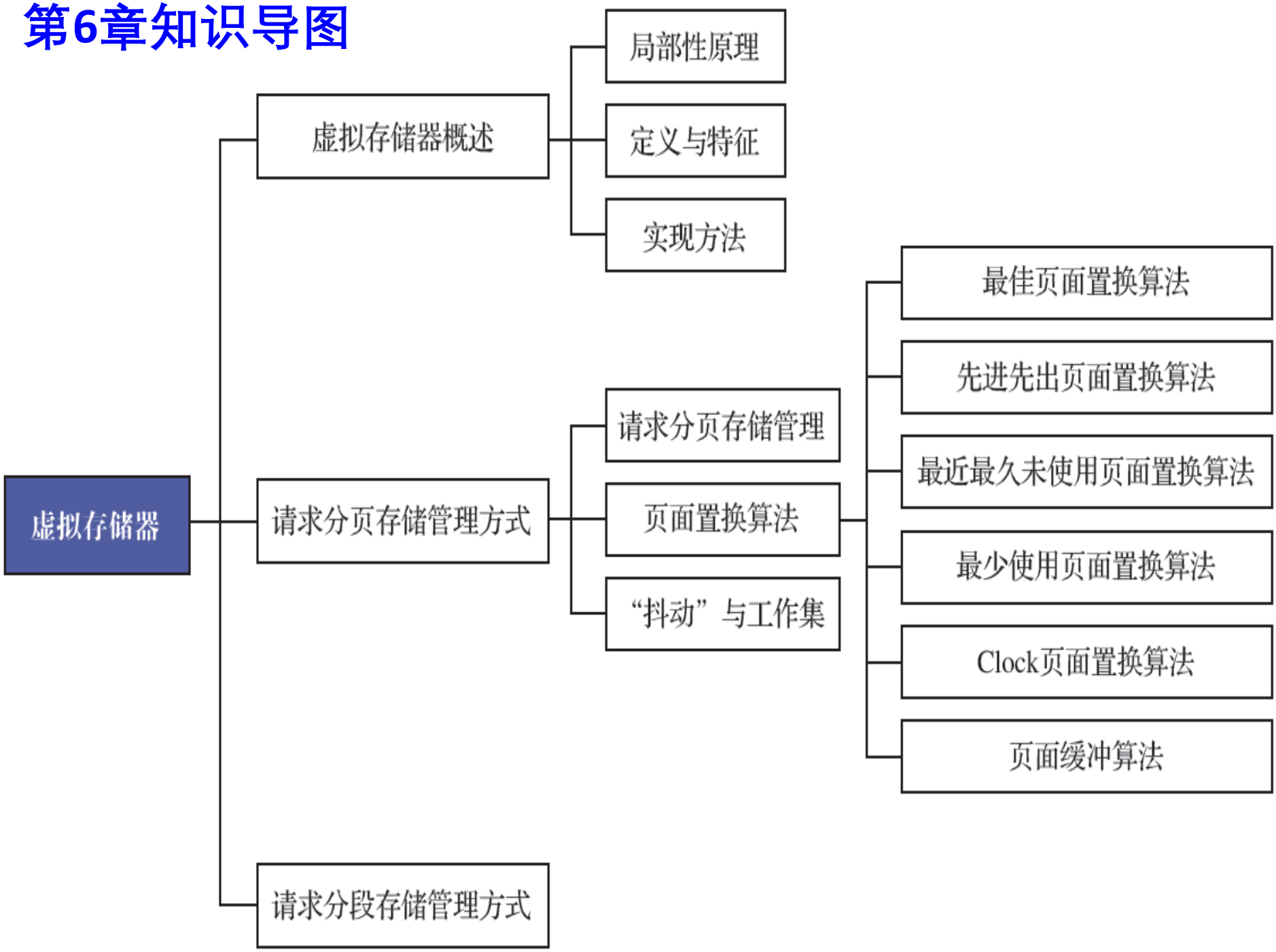
教师：秦小立
学院：网络空间安全学院
邮箱：xlqin@hainanu.edu.cn
办公地点：学院309

课程知识导图

第1章	操作系统引论
第2章	进程的描述与控制
第3章	处理机调度与死锁
第4章	进程同步
第5章	存储器管理
第6章	虚拟存储器
第7章	输入/输出系统
第8章	文件管理
第9章	磁盘存储器管理
第10章	多处理机操作系统
第11章	虚拟化和云计算
第12章	保护和安全

OS功能与虚拟存储器

第6章知识导图


虚拟存储器
虚拟存储器概述
局部性原理
定义与特征
实现方法
请求分页存储管理方式
请求分页存储管理
页面置换算法
最佳页面置换算法
先进先出页面置换算法
最近最久未使用页面置换算法
最少使用页面置换算法
Clock页面置换算法
页面缓冲算法
“抖动”与工作集
请求分段存储管理方式

第6章虚拟存储器

6.1.1常规存储器管理方式的特征和局部性原理

前面所介绍的各种存储器管理方式，有一个共同特点：作业全部装入内存后方能运行。于是出现了两种情况：

- 大作业所要求的内存空间超过内存总容量，作业不能被全部装入内存，致使该作业无法运行。
- 有大量作业要求运行时，内存容量不足以容纳所有作业，只能将少数作业装入内存使其运行，大量作业留在外存上等待。

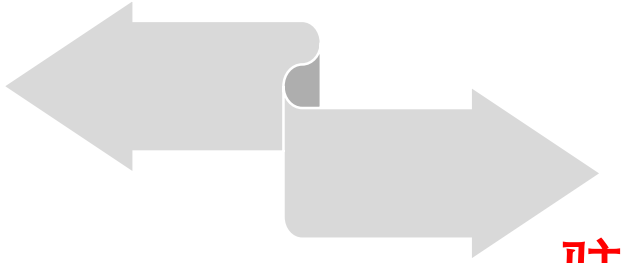
解决方法：

从物理上增加内存容量

从逻辑上扩充内存容量：对换和虚拟存储器

6.1.1常规存储器管理方式的特征和局部性原理

1、常规存储器管理方式的特征



一次性：

作业被一次性全部装入内存

驻留性：作业被装入内存后一直驻留在内存，直到运行结束

一次性和驻留性使许多在程序运行中不用或暂不用的程序（数据）占据了大量的内存空间，使得一些需要运行的作业无法装入运行。

6.1.1常规存储器管理方式的特征和局部性原理

2、局部性原理(虚拟存储器的理论基石)

- 1968年，P. denning 提出:程序在执行时将呈现出局部性规律,即在一较短的时间内，程序的执行仅局限于某个部分;相应地，它所访问的存储空间也局限于某个区域。

表现:

- 时间局限性:循环执行造成（一段时间后再次被访问）
- 空间局限性:顺序执行造成（附近的存储单元也被执行）

6.1.1常规存储器管理方式的特征和局部性原理

3、虚拟存储器的基本工作情况

- 基于局部性原理，程序在运行之前，没有必要全部装入内存，仅须将当前要运行的页(段)装入内存即可。
- 运行时，如访问的页(段)在内存中，则继续执行，如访问的页未在内存中(缺页或缺段)，则利用 OS 的请求调页(段)功能，将该页(段)调入内存。
- 如内存已满，则利用 OS 的页(段)置换功能（对换），按某种置换算法将内存中的某页(段)调至外存，从而调入需访问的页。

6.1.2虚拟存储器的定义与特征

1、虚拟存储器的定义

- 虚拟存储器：是指具有请求调入功能和置换功能，能从逻辑上对内存容量加以扩充的一种存储器系统。
- 虚拟存储器逻辑容量：由内存容量和外存容量之和所决定，其运行速度接近于内存速度，而成本却接近于外存。

- 实质：以时间换空间，但时间牺牲不大。

6.1.2虚拟存储器的定义与特征

2、虚拟存储器的特征-3个

- 多次性(虚拟存储器最重要的特征)：一个作业被分成多次调入内存运行
- 对换性：允许在作业的运行过程中进行换进、换出
- 虚拟性：能够从逻辑上扩充内存容量，用户可使用的内存容量远大于实际内存容量。

- 虚拟性是以多次性和对换性为基础的。
- 多次性和对换性是建立在离散分配方式基础上的。

6.1.3虚拟存储器的实现方法

实现是建立在离散分配的存储管理方式的基础上的

1、请求分页系统

在分页系统的基础上，**增加了请求调页功能和页面置换功能**所形成的**页式虚拟存储系统**。

- **基本原理：**它允许只装入部分页面的程序(及数据)，便启动运行。以后再通过调页功能及页面置换功能，陆续将即将要运行的页面调入内存，同时把暂不运行的页面换出到外存上。置换时以页面为单位。

- **硬件支持：**①请求分页的页表机制 ②缺页中断机构 ③地址变换机构

- **软件支持：**①实现请求调页的软件②实现页面置换的软件

6.1.3虚拟存储器的实现方法

2、请求分段系统

在分段系统的基础上，增加了请求调段功能和分段置换功能所形成的段式虚拟存储系统。

基本原理：它允许只装入若干段的程序(及数据)，便启动运行。以后再通过调段功能及段的置换功能，把暂不运行的段调出，同时调入即将要运行的段。置换时以段为单位。

为实现请求调段和置换功能，系统必须提供必要的支持：

- 硬件支持：①请求分段的段表机制 ②缺段中断机构 ③地址变换机构
- 软件支持：①实现请求调段的软件 ②实现段置换的软件

第6章虚拟存储器

6.2.1请求分页中的硬件支持

请求分页系统实现简单，是最常用一种虚拟存储器的实现方式。除了要内存外，还需要请求页表机制、缺页中断机构及地址变换机构。

1、请求页表机制

页号	物理块号	状态位P	访问字段A	修改位M	外存地址

- 状态位P：指示该页是否在内存
- 访问字段A：记录该页在一段时间内被访问的次数
- 修改位M：也称脏位，标志该页是否被修改过
- 外存地址：指示该页在外存中的地址（物理块号）

6.2.1请求分页中的硬件支持

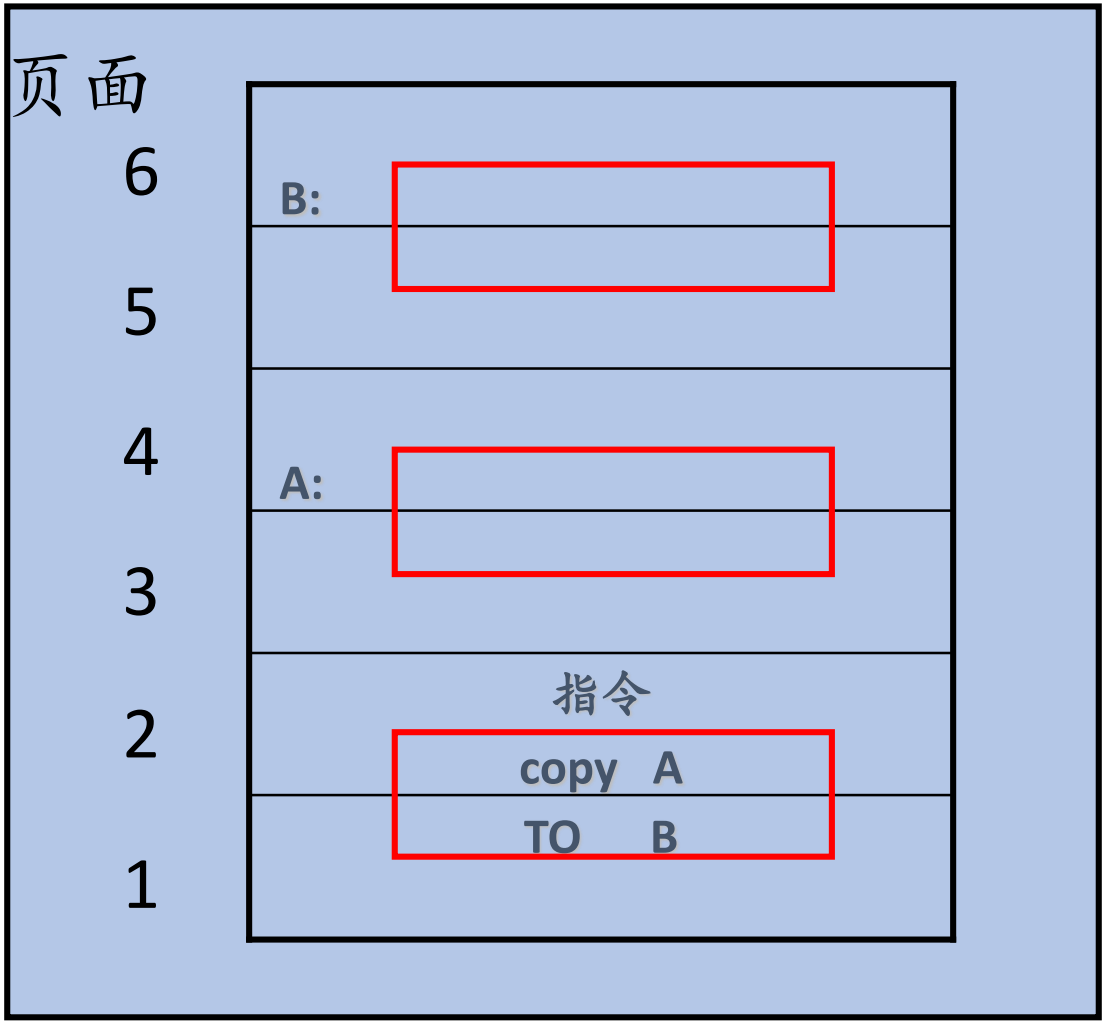
2、缺页中断机构

- 缺页中断：在请求分页系统中，每当所要访问的页面不在内存时，便产生一缺页中断，请求OS将所缺页面调入内存。
- 缺页中断的特点(缺页中断和一般中断的区别)
- 1）缺页中断在指令执行期间产生和处理，而一般中断要等到一条指令执行完才处理。
- 2）一条指令在执行期间，可能产生多次缺页中断,系统中的硬件机构应能保存多次中断时的状态，并保证最后能返回到中断前产生缺页中断的指令处继续执行。

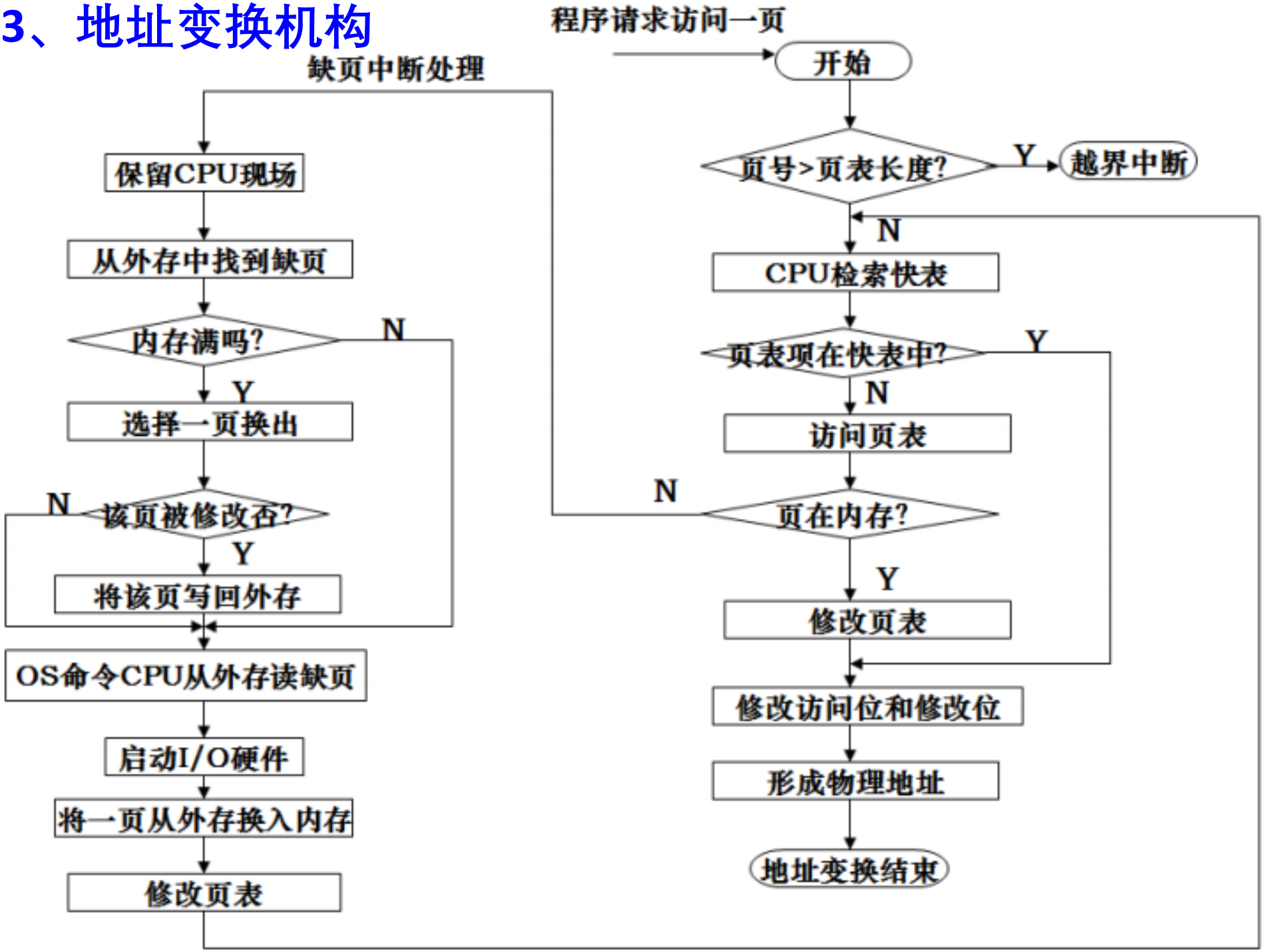
6.2.1请求分页中的硬件支持

2、缺页中断机构

涉及6次缺页中断的指令举例



3、地址变换机构


程序请求访问一页
开始
页号>页表长度?
Y
越界中断
N
CPU检索快表
页表项在快表中?
Y
N
访问页表
页在内存?
N
Y
修改页表
修改访问位和修改位
形成物理地址
地址变换结束
缺页中断处理
保留CPU现场
从外存中找到缺页
内存满吗?
N
Y
选择一页换出
该页被修改否?
N
Y
将该页写回外存
OS命令CPU从外存读缺页
启动I/O硬件
将一页从外存换入内存
修改页表

6.2.1请求分页中的硬件支持

3、地址变换机构

例题：某虚拟存储器的用户空间共有32个页面，每页1KB，主存16KB。假定某时刻系统为用户的第 0、1、2、3 页分别分配的物理块号为 5、10、4、7，试将虚拟地址 0A5C 变换为物理地址。

解：

逻辑地址结构为：页号（2^5=32）5位 页内位移（2^10=1024）10位

物理地址结构为：物理块号（2^4=16）4位 块内位移（2^10=1024）10位

虚拟地址0A5C对应的二进制为： 00010 1001011100

查表即虚拟地址 0A5C 的页号为 2，页内位移为 1001011100，由题意知，**对应的物理地址为：0100 1001011100 即125C**。

6.2.2请求分页中的内存分配

1. 最小物理块数的确定

- **最小物理块数，**指能保证进程正常运行所需的最小物理块数。当系统为进程分配的物理块数少于此值时，进程将无法运行。
- **进程应获得的最小物理块数**与计算机的硬件结构有关，取决于**指令的格式、功能和寻址方式**。

直接寻址方式，最少需2个物理块
指令、数据至少各需1个物理块
例: MOV AX, [1234H]
间接寻址方式，最少需3个物理块
例: MOV AX, [BX]

6.2.2请求分页中的内存分配

2. 内存分配策略

请求分析系统中，分配策略分为固定与可变分配策略。

1）固定分配策略：分配给进程的物理块数是固定的。并在最初装入时(即进程创建时)确定块数。当进程执行过程中出现缺页时，只能从分给该进程的物理块中进行页面置换。

2）可变分配策略：允许分给进程的物理块数随进程的活动而改变。如果一个进程在运行过程中持续缺页率太高，这就表明该进程的局部化行为不好，需要给它分配另外的物理块，以减少它的缺页率。如果一个进程的缺页率特别低，就可以减少分配的物理块，但不要显著增加缺页率。

6.2.2请求分页中的内存分配

2. 内存分配策略

置换策略:全局置换和局部置换策略

1）**局部置换策略**:每个进程只能从分给它的一组物理块中选择置换块。

2）**全局置换策略**:允许一个进程从全体物理块(包括分配给别的进程的块)的集合中选取置换块，尽管该块当前已分给其他进程，但还是能强行剥夺。

6.2.2请求分页中的内存分配

2. 内存分配策略

由上述两类分配策略可得：

（1）固定分配局部置换：

为每个进程分配固定数目n的物理块，在整个运行中都不改变。如出现缺页，只能从该进程的页面中选中一页换出，再调入所需页。

（2）可变分配全局置换：

分配固定数目的物理块，但OS自留一空闲块队列，若发现缺页，则从空闲块队列中分配一空闲块与该进程，并调入缺页。当空闲块队列用完时，OS才从内存中任选一页置换。

6.2.2请求分页中的内存分配

2. 内存分配策略

(3) 可变分配局部置换：

分配一定数目的物理块，若发现缺页，则从该进程的页面中置换一页，根据该进程缺页率高低，则可增加或减少分配给该进程的物理块。

6.2.2请求分页中的内存分配

3. 物理块分配算法

在采用固定分配策略时，将系统中可供分配的所有物理块分配给各个进程，可采用以下几种算法：

（1）平均分配算法：平均分配给各个进程。

（2）按比例分配算法：根据进程的大小按比例分配给各个进程。

（3）考虑优先权的分配算法：将系统提供的物理块一部分根据进程大小先按比例分配给各个进程，另一部分再根据各进程的优先权适当增加物理块数。

6.2.3页面调入策略

要执行程序进程，必须将该部分程序和数据所在页面调入内存。那么

1. 何时调入所需页面?
2. 系统从何处调入这些页面?
3. 如何进行调入?
4. 与调入次数相关的缺页率如何定义的

6.2.3页面调入策略

1、何时调入页面

1）预调页策略

基本思想：将那些预计在不久之后便会被访问的页面预先调入内存。

缺点：成功率约50%，适于进程首次调入时使用

2）请求调页策略

基本思想：根据请求需要将需要访问的缺页由OS将其调入内存。

缺点：每次仅调入一页，系统开销大

注：目前的虚拟存储系统大多采用此种策略。

6.2.3页面调入策略

2、确定从何处调入页面

对换区：采用连续分配方式，速度快，一般修改过、运行过的页面被换出时应放入对换区，需要时再从对换区换入。

文件区：采用离散分配方式，速度稍慢，如果对换区空间不够用，则将不会被修改的页面、未运行过的页面放在文件区。

对共享页面，应判断其是否在内存区，如在则无需调入。

6.2.3页面调入策略

3、如何调入页面

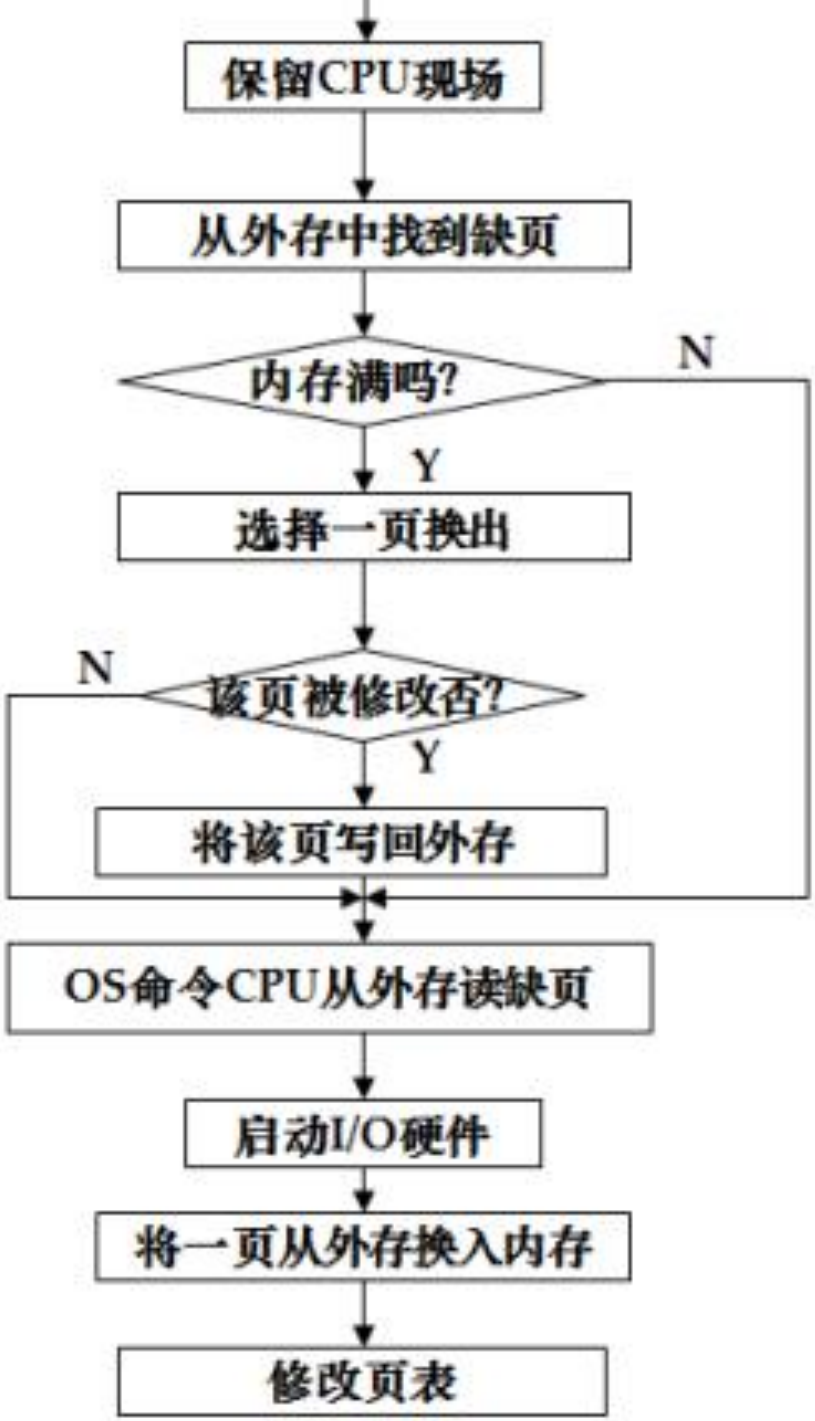
对换区:采用连续分配方式，速度快，一般修改过、运行过的页面被换出时应放入对换区，需要时再从对换区换入。

文件区:采用离散分配方式，速度稍慢，如果对换区空间不够用，则将不会被修改的页面、未运行过的页面放在文件区。

对共享页面，应判断其是否在内存区，如在则无需调入。

UNIX方式：凡未运行过的页面均从文件区调页，运行过的页面和换出的页面均从对换区调页。

3、如何调入页面


保留CPU现场
从外存中找到缺页
内存满吗?
N
Y
选择一页换出
该页被修改否?
N
Y
将该页写回外存
OS命令CPU从外存读缺页
启动I/O硬件
将一页从外存换入内存
修改页表

6.2.3页面调入策略

4、缺页率

缺页率受影响因素：

(1) 页面大小页面越大，缺页率越低

(2)进程所分配物理块的数目

一般而言，分配的物理块越多，，缺页率越低

(3) 页面置换算法

(4)程序固有特性-程序的局部化程度越高，缺页率越低

6.2.3页面调入策略

4、缺页率

$$f = \frac{F}{A}$$

其中:F为缺页次数，A为页面总访问数

缺页中断处理时间： $t = \beta \times t_a + (1-\beta) \times t_b$

其中： β 为缺页时被置换出的页面的修改概率，其缺页中断处理时间为 t_a；t_b为缺页时被置换出的页面的没有修改的中断处理时间

6.2.3页面调入策略

4、缺页率

缺页率受影响因素：

(1) 页面大小页面越大，缺页率越低

(2)进程所分配物理块的数目

一般而言，分配的物理块越多，，缺页率越低

(3) 页面置换算法

(4)程序固有特性-程序的局部化程度越高，缺页率越低

缺页中断处理时间的例子

- 存取内存的时间= 200 nanoseconds (ns)
- 平均缺页处理时间 = 8 milliseconds (ms)
- $t = (1 - p) \times 200ns + p \times 8ms$

  $= (1 - p) \times 200ns + p \times 8{,}000{,}000ns$

  $= 200ns + p \times 7{,}999{,}800ns$
- 如果每1,000次访问中有一个缺页中断，那么：

  $t = 8.2$ ms

这是导致计算机速度放慢40倍的影响因子！

第6章虚拟存储器

6.3 页面置换算法

页面置换算法也称为页面淘汰算法，是用来选择换出页面的算法。

目的：降低页面置换频率，也就是需要最小的缺页率

注：不当的页面置换算法可能引起页面“抖动”。

1、最佳置换算法

2、先进先出(FIFO) 置换算法

3、最近最久未使用(LRU)置换算法

4、最少使用(LFU)置换算法

5、Clock置换算法

6、页面缓冲(PBA) 置换算法

6.3.1最佳置换算法和先进先出置换算法

1、最佳置换算法（OPT）

思想：其所选择的被淘汰页面，将是以后永不再用的，或许是在最长(未来)时间内不再被访问的页面。

优点：保证获得最低的缺页率

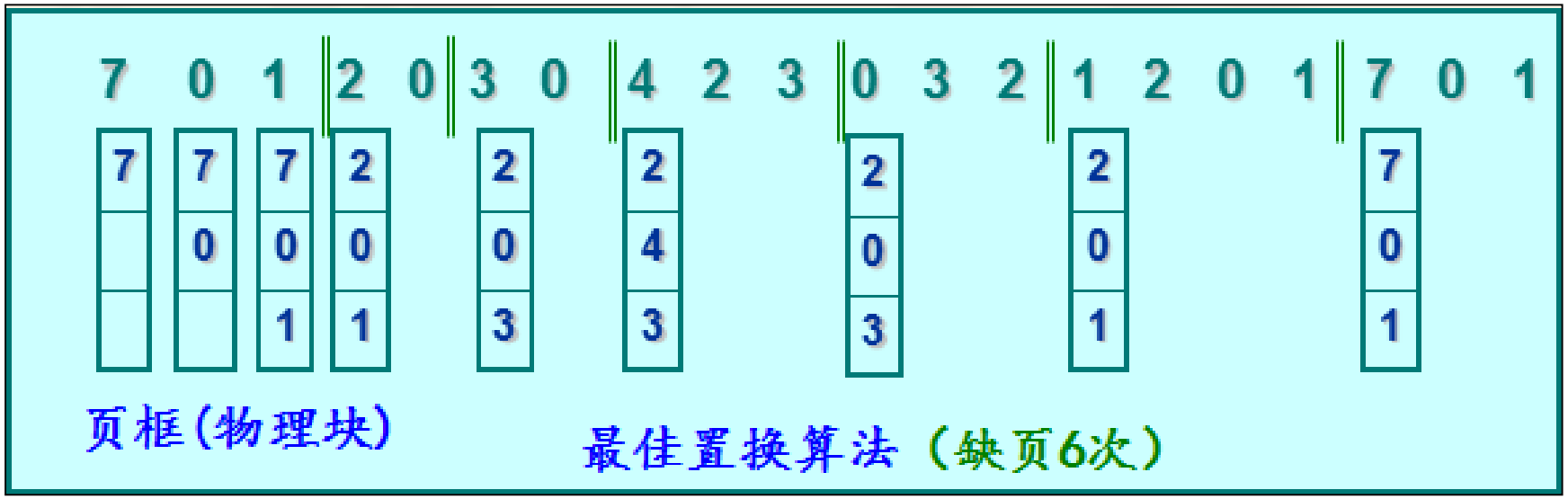
缺点：无法预知一个进程在内存的若干个页面，哪个在未来最长时间内不再被访问。

算法无法实现，但可评价其他算法。

6.3.1最佳置换算法和先进先出置换算法

1、最佳置换算法（OPT）

例如：设作业分配3个物理块，开始3页不算缺页（后面算法同）。采用最佳置换算法如何置换?



缺页次数: 9次(包括前3个),缺页率: 9/20=45%置换次数: 6次

6.3.1最佳置换算法和先进先出置换算法

2、先进先出(FIFO) 置换算法

算法思想：总是淘汰最先进入内存的页面，即选择在内存驻留时间最长的页面予以淘汰。

算法实现：将进程在内存中页面按先后次序链接成一个队列，并设置一个指针，称为替换指针，使它总是指向最老的页面。

算法特点：简单、易实现；貌似公平，实际上不公平，不切实际，有些经常被访问的页面可能先被淘汰，因此性能较差。

6.3.1最佳置换算法和先进先出置换算法

2、先进先出(FIFO) 置换算法

例题

缺页次数: 15次(包括前3个)，缺页率: 15/20=75% 置换次数: 12次

6.3.1最佳置换算法和先进先出置换算法

2、先进先出(FIFO) 置换算法

- 引用串：1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5
  - 3 个页：9次缺页
  - 4 个页框：10次缺页

1	1	4	5	
2	2	1	3	9 次缺页
3	3	2	4	

1	1	5	4	
2	2	1	5	10 次缺页
3	3	2		
4	4	3		

Belady现象

如果对一个进程未分配它所要求的全部页面，有时就会出现分配的页面数增多但缺页率反而提高的异常现象。发生在FIFO（先进先出）置换算法。

6.3.2最近最久未使用和最少使用置换算法

2、先进先出(FIFO) 置换算法

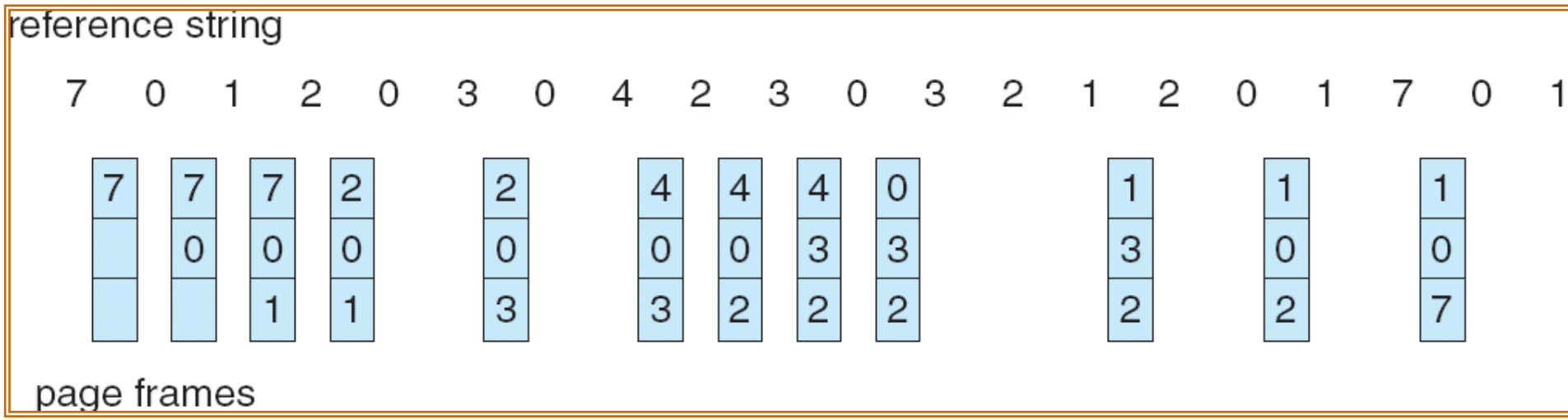
Belady现象

如果对一个进程未分配它所要求的全部页面，有时就会出现分配的页面数增多但缺页率反而提高的异常现象。发生在FIFO（先进先出）置换算法。

6.3.2最近最久未使用和最少使用置换算法

1、最近最少使用算法(LRU)

选择最近最久未使用的页面予以淘汰



6.3.2最近最久未使用和最少使用置换算法

2、LRU算法的硬件支持：

需要两者之一的支持

寄存器：为内存中的每个页面设置一个移位寄存器

$$R=R_{n-1}R_{n-2}R_{n-3}\ \ldots\ R_2R_1R_0$$

- 被访问的页面对应寄存器的Rn-1位置为1，定时右移
- 具有最小数值的寄存器所对应的页面为淘汰页

栈：保存当前使用的各个页面的页面号

- 被访问的页，移到栈顶
- 栈底是最近最久未使用页面的页面号

注：这个是特殊的栈，一般栈是先进后出，这里是先进先出的栈

6.3.2最近最久未使用和最少使用置换算法

1、最近最少使用算法(LRU)

图6-7 某进程具有8个访问页面时的LRU访问情况

例如：下表中，第 3 页是最近最久未被访问的页。

实页 \ R	R_7	R_6	R_5	R_4	R_3	R_2	R_1	R_0
1	0	1	0	1	0	0	1	0
2	1	0	1	0	1	1	0	0
3	0	0	0	0	0	1	0	0
4	0	1	1	0	1	0	1	1
5	1	1	0	1	0	1	1	0
6	0	0	1	0	1	0	1	1
7	0	0	0	0	0	1	1	1
8	0	1	1	0	1	1	0	1

最近访问

6.3.2最近最久未使用和最少使用置换算法

3、最少使用置换算法LFU

为内存中的每个页面设置一个移位寄存器，用来记录该页面的被访问频率

LFU 选择在最近时期使用最少的页面作为淘汰页

6.3.2最近最久未使用和最少使用置换算法

3、最少使用置换算法LFU

为内存中的每个页面设置一个移位寄存器，用来记录该页面的被访问频率

LFU 选择在最近时期使用最少的页面作为淘汰页

6.3.3 Clock页面置换算法

1、简单CLock置换算法

LRU的近似算法，又称最近未用(NRU)或二次机会页面置换算法

基本思想：

为每页设置一位访问位，再将内存中所有页面通过链接指针链成一个循环队列。

当某页被访问时，其访问位被置1，表示该页最近使用过。

置换算法在选择一页淘汰时，只需循环检查各页的访问位：如果为1，则将该访问位置0，暂不换出;如果为0,则将该页换出，算法终止。

简单型Clock算法每次选择的淘汰页面均是最近未使用的页面

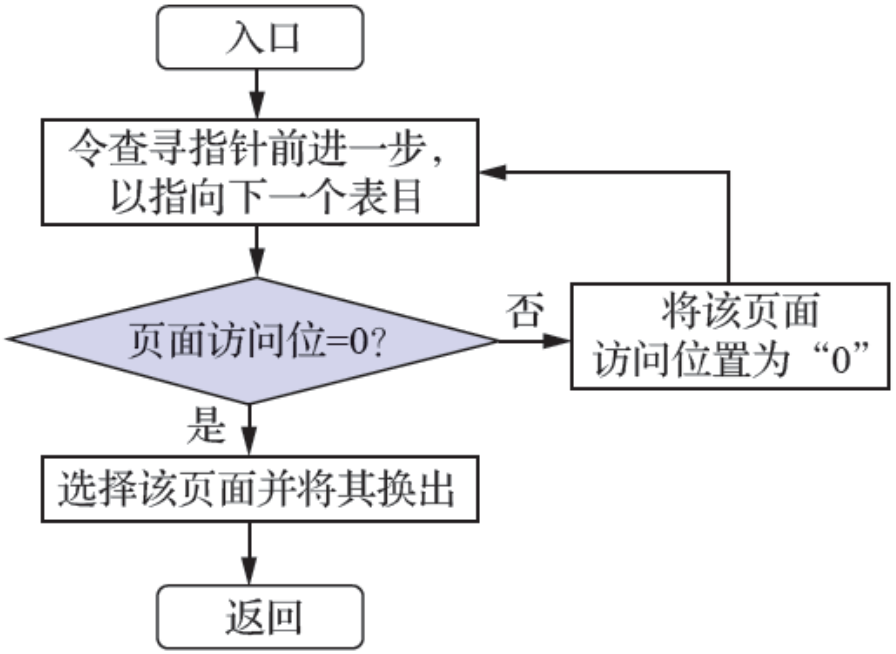
6.3.3 Clock页面置换算法

1、简单CLock置换算法

- 每个页都与一个访问位相关联，初始值位0
- 当页被访问时置访问位为1
- 置换时选择访问位为0的页；若为1，重新置为0

6.3.3 Clock页面置换算法

1、简单CLock置换算法



简单CLock置换算法流程

块号	页号	访问位	指针
0			
1			
2	4	0	
3			
4	2	1	
5			
6	5	0	
7	1	1	

替指

示意图

6.3.3 Clock页面置换算法

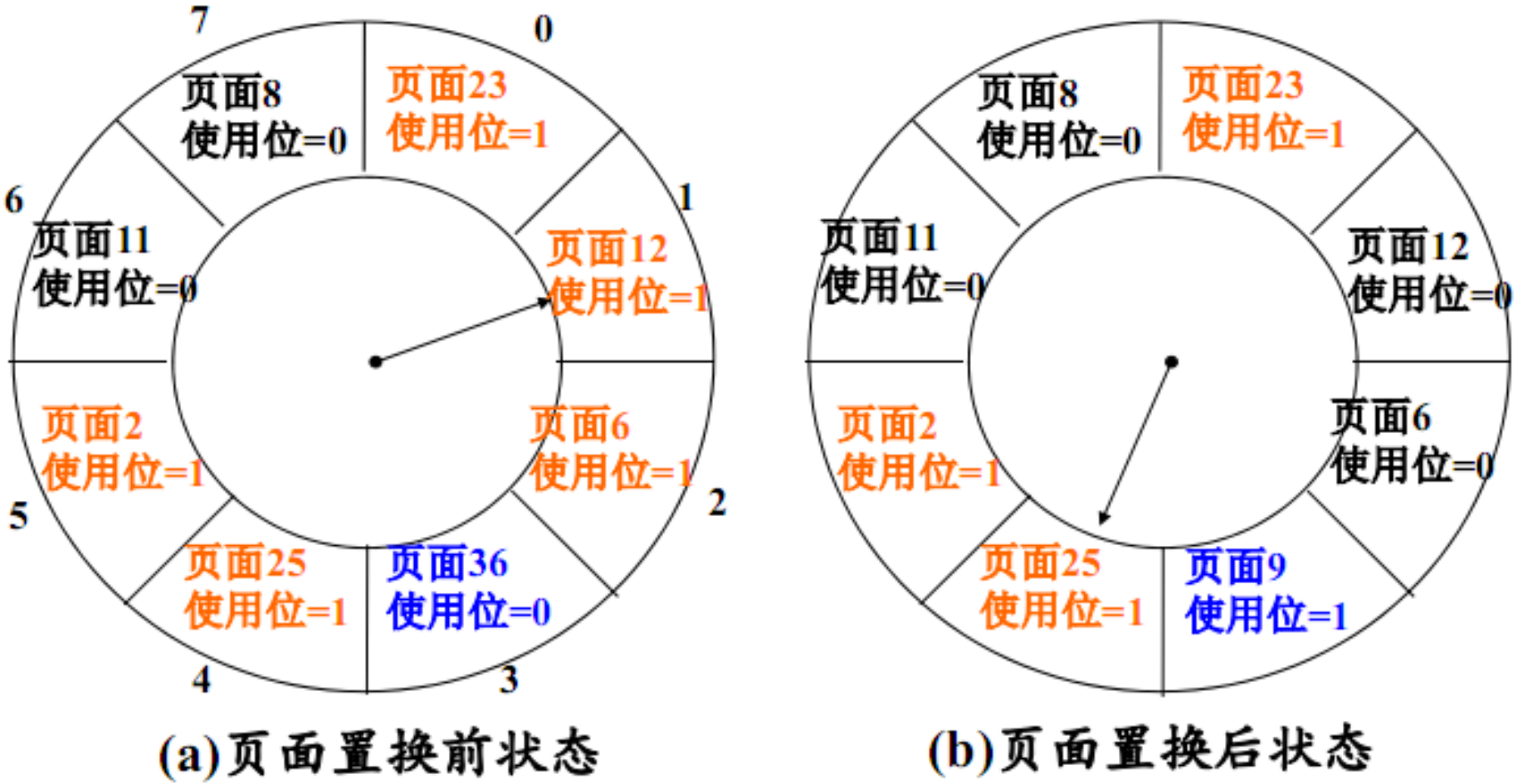


图　简单型Clock置换算法示例

6.3.3 Clock页面置换算法

2、改进型Clock算法

除须考虑页面的使用情况外，还须增加置换代价

淘汰时，同时检查访问位A与修改位M

- 第1类（A=0，M=0）：表示该页最近既未被访问、又未被修改，是最佳淘汰页。
- 第2类（A=0，M=1）：表示该页最近未被访问，但已被修改，并不是很好的淘汰页。
- 第3类（A=1，M=0）：表示该页最近已被访问，但未被修改，该页有可能再被访问。
- 第4类（A=1，M=1）：表示该页最近已被访问且被修改，该页有可能再被访问。

置换时，循环依次查找第1类、第2类页面，找到为止

6.3.4页面缓冲算法PBA

影响效率的因素：

- 页面置换算法、写回磁盘的频率、读入内存的频率

目的：

- 显著降低页面换进、换出的频率，减少了开销
- 可采用较简单的置换策略，如不需要硬件支持

6.3.4页面缓冲算法PBA

算法思想

- PBA采用可变分配和局部置换方式，置换算法采用FIFO。
- 该算法在内存中设置1个空闲物理块链表和1个已修改页面链表，置换时如果被淘汰页面未被修改，就将它直接挂在空闲链表末尾，并从空闲链表表首取出一个空闲块用来装入缺页；否则，将其挂在已修改页面链表末尾。

6.3.4页面缓冲算法PBA

具体做法：

- 设置两个链表：

①空闲页面链表：保存空闲物理块

②修改页面链表：保存已修改且需要被换出的页面等被换出的页面数目达到一定值时，再一起换出外存。

6.3.5 请求分页系统的内存有效访问时间

1、访问页在内存，且其对应页表项在快表中

- EAT= λ + t　　λ 快表访问时间，t是一次内存访问时间

2、访问页在内存，且其对应页表项不在快表中

- EAT= λ + t + λ + t = 2(λ + t)

3、访问页不在内存中

- 需进行缺页中断处理，有效时间可分为查找快表的时间、查找页表的时间、处理缺页的时间、更新快表的时间和访问实际物理地址的时间
- 假设缺页中断处理时间为ε

  EAT = λ + t + ε + λ + t = ε + 2(λ + t)
- f为缺页率，ε为缺页中断处理时间

  EAT = t + f ×(ε + t) + (1- f) × t （含缺页率的计算式）

第6章虚拟存储器

6.4 抖动与工作集

1、抖动

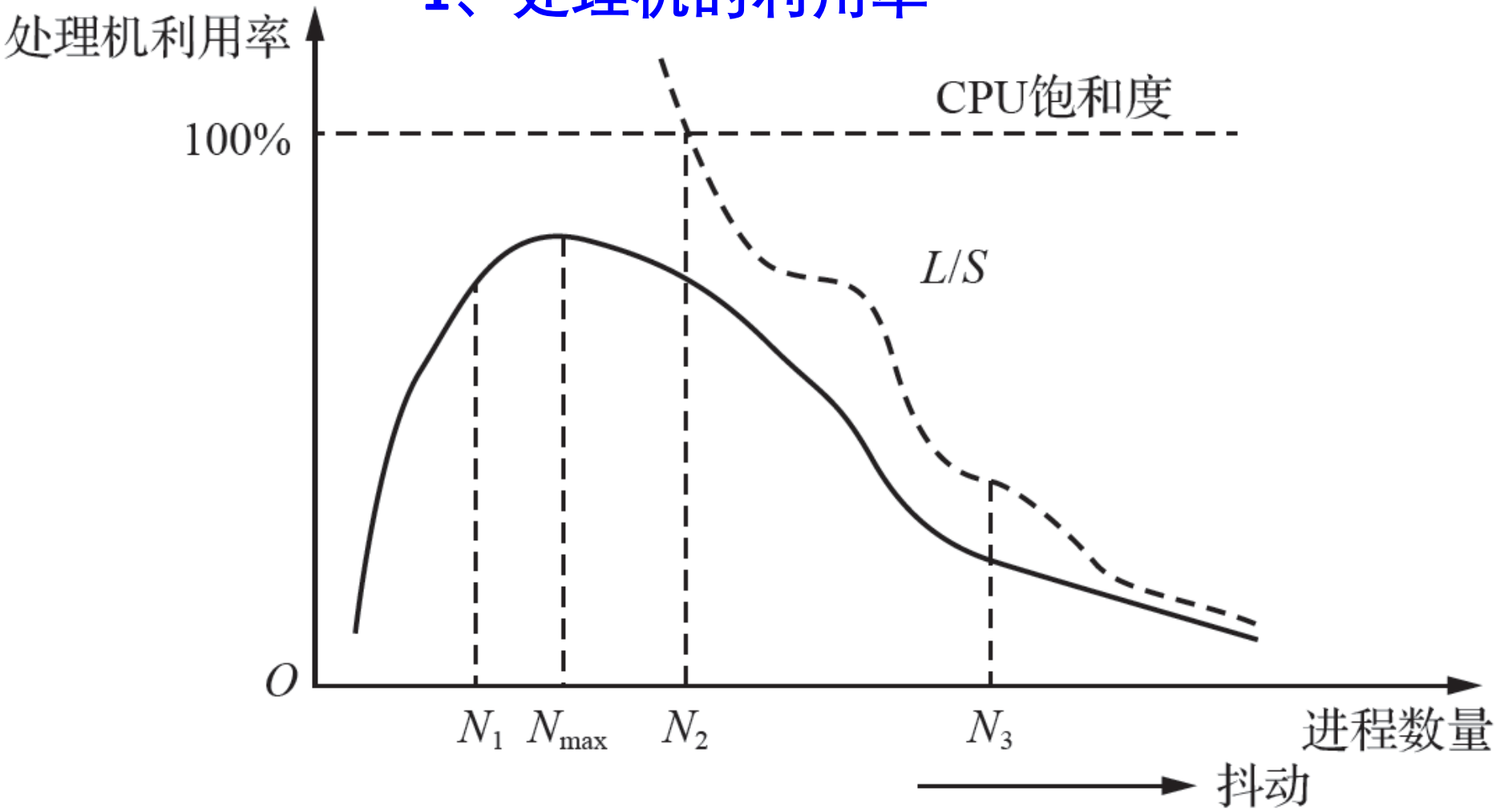
如果一个进程没有足够的页，那么缺页率将很高，这将导致:

- CPU利用率低下.
- 操作系统认为需要增加多道程序设计的道数
- 系统中将加入一个新的进程

抖动(Thrashing) ：一个进程的页面经常换入换出

6.4.1 多道程序度与抖动

1、处理机的利用率



6.4.1 多道程序度与抖动

2、产生“抖动”的原因

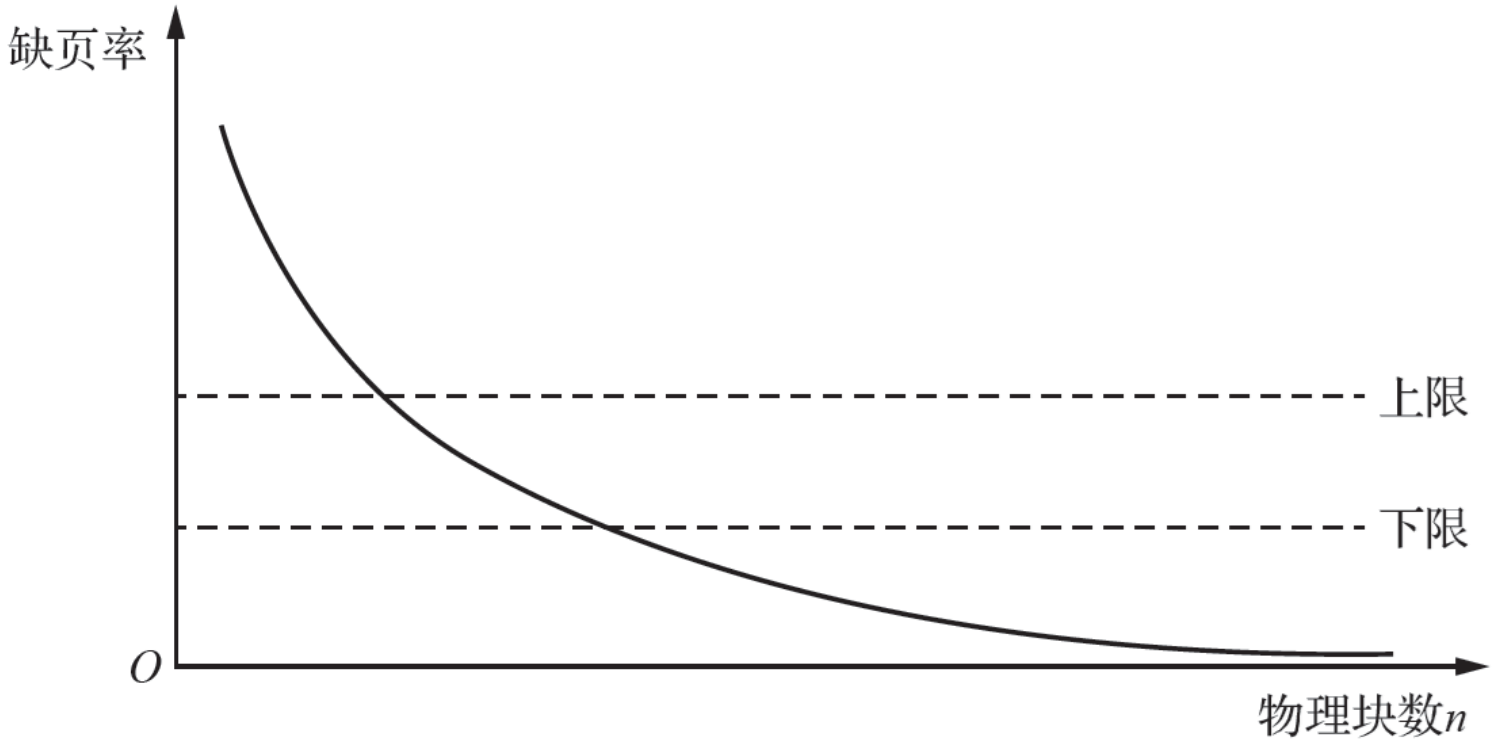
根本原因：

- 同时在系统中运行的进程太多；
- 因此分配给每一个进程的物理块太少，不能满足进程运行的基本要求，致使进程在运行时，频繁缺页，必须请求系统将所缺页面调入内存。

抖动的发生与系统为进程分配物理块的多少有关。

缺页率与物理块数之间的关系

- 进程发生缺页的时间间隔与所获得的物理块数有关。
- 根据程序运行的局部性原理，如果能够预知某段时间内程序要访问的页面，并将它们预先调入内存，将会大大降低缺页率。



工作集定义

I. 所谓工作集，指在某段时间间隔Δ里进程实际要访问页面的集合。

II. 把某进程在时间t的工作集记为w(t，Δ),其中的变量Δ称为工作集的“窗口尺寸”。

◆ 工作集w(t，Δ)是二元函数，即在不同时间t的工作集大小不同，所含的页面数也不同；工作集与窗口尺寸Δ有关，是Δ的非降函数，即：

$$w(t,\ \Delta) \subseteq w(t,\ \Delta+1)$$

工作集举例

访问页面序列	窗口大小 3	窗口大小 4	窗口大小 5
24	24	24	24
15	15 24	15 24	15 24
18	18 15 24	18 15 24	18 15 24
23	23 18 15	23 18 15 24	23 18 15 24
24	24 23 18	—	—
17	17 24 23	17 24 23 18	17 24 23 18 15
18	18 17 24	—	—
24	—	—	—
18	—	—	—
17	—	—	—
17	—	—	—
15	15 17 18	15 17 18 24	—
24	24 15 17	—	—
17	—	—	—
24	—	—	—
18	18 24 17	—	—

抖动的预防方法

01 采取局部置换策略：只能在分配给自己的内存空间内进行置换；

02 把工作集算法融入到处理机调度中；

03 利用“L=S”准则调节缺页率：

- L是缺页之间的平均时间
- S是平均缺页服务时间，即用于置换一个页面的时间
- L>S，说明很少发生缺页
- L<S，说明频繁缺页
- L=S，磁盘和处理机都可达到最大利用率

04 选择暂停进程。

第6章虚拟存储器

请求分段中的硬件支持（1）

请求段表机制

- 存取方式：表示段存取属性为只执行、只读或允许读/写
- 访问字段A：记录该段在一段时间内被访问的次数
- 修改位M：标志该段调入内存后是否被修改过
- 存在位P：指示该段是否在内存
- 增补位：表示该段在运行过程中是否做过动态增长
- 外存始址：指示该段在外存中的起始地址（盘块号）

段名	段长	段的始址	存取方式	访问字段A	修改位M	存在位P	增补位	外存始址

请求分段中的硬件支持（2）

缺段中断机构

- 在指令执行期间产生和处理中断信号
- 一条指令在执行期间，可能产生多次缺段中断
- 由于段不是定长的，对缺段中断的处理要比对缺页中断的处理复杂

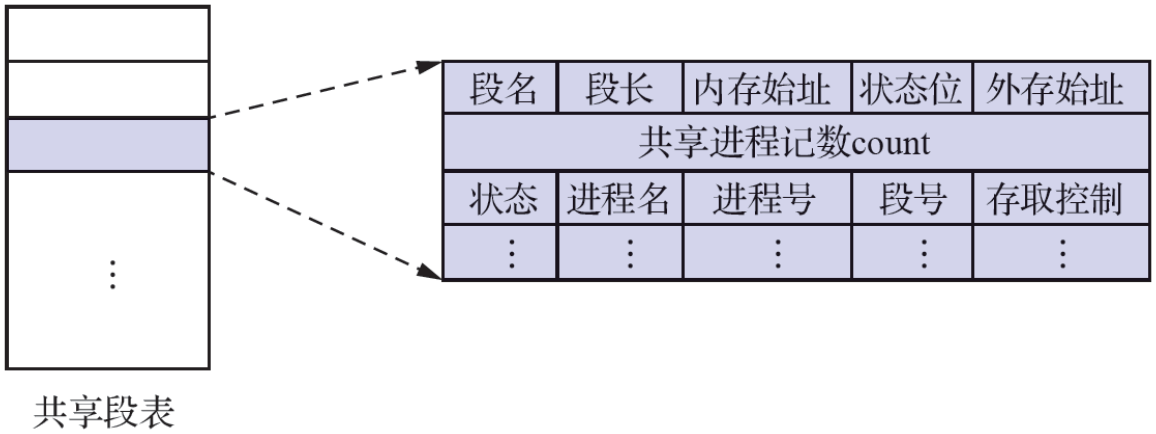
地址变换机构

- 若段不在内存中，则必须先将所缺的段调入内存，并修改段表，然后利用段表进行地址变换。

分段的共享

共享段表：保存所有的共享段

- 共享进程计数count
- 存取控制字段
  - 段号



共享段的分配

- 对首次请求使用共享段的用户，分配内存，调入共享段，修改该进程段表相应项，再为共享段表增加一项，count=1
- 对其他使用共享段的用户，修改该进程段表相应项，再为共享段表增加一项，count=count+1

分段的共享

共享段的回收

- 撤销在该进程段表中共享段所对应的表项，并执行count=count-1操作
- 若为0，回收该共享段的内存，并取消共享段表中对应的表项
- 若不为0，只取消调用者进程在共享段表中的有关记录

分段保护

越界检查：

- 由地址变换机构来完成；
- 比较段号与段表长度；段内地址与段表长度。

存取控制检查：以段为基本单位进行。

- 通过“存取控制”字段决定段的访问方式；
- 基于硬件实现。

环保护机构：

- 低编号的环具有高优先权；
- 一个程序可以访问驻留在相同环或较低特权环（外环）中的数据；
- 一个程序可以调用驻留在相同环或较高特权环（内环）中的服务。

第6章虚拟存储器

实例1：Windows XP系统

- ✓采用请求页面调度以及簇来实现虚拟存储器
- ✓使用簇在处理缺页中断时，不但会调入不在内存中的页（出错页），还会调入出错页周围的页
- ✓创建进程时，系统会为其分配工作集的最小值和最大值
  - 最小值：进程在内存中时所保证页面数的最小值
  - 若内存足够，可分配更多的页面，直到达到最大值
  - 通过维护空闲块链表（与一个阈值关联）来实现
  - 采用局部置换方式
- ✓置换算法与处理器类型有关
  - 如80x86系统，采用改进型Clock算法

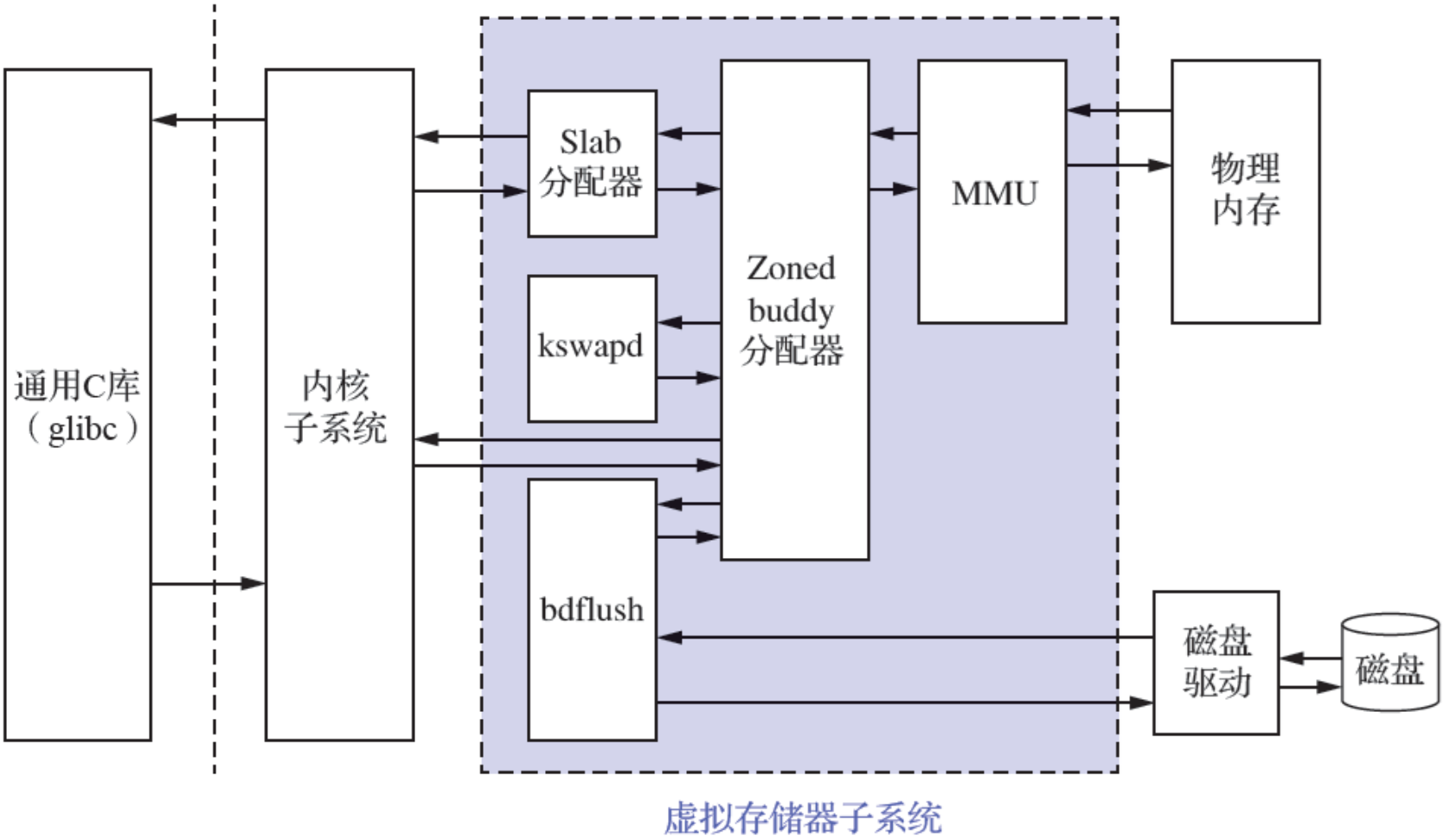
实例2：Linux系统（以32位为例）

虚拟存储器是大小为4GB的线性虚拟空间。

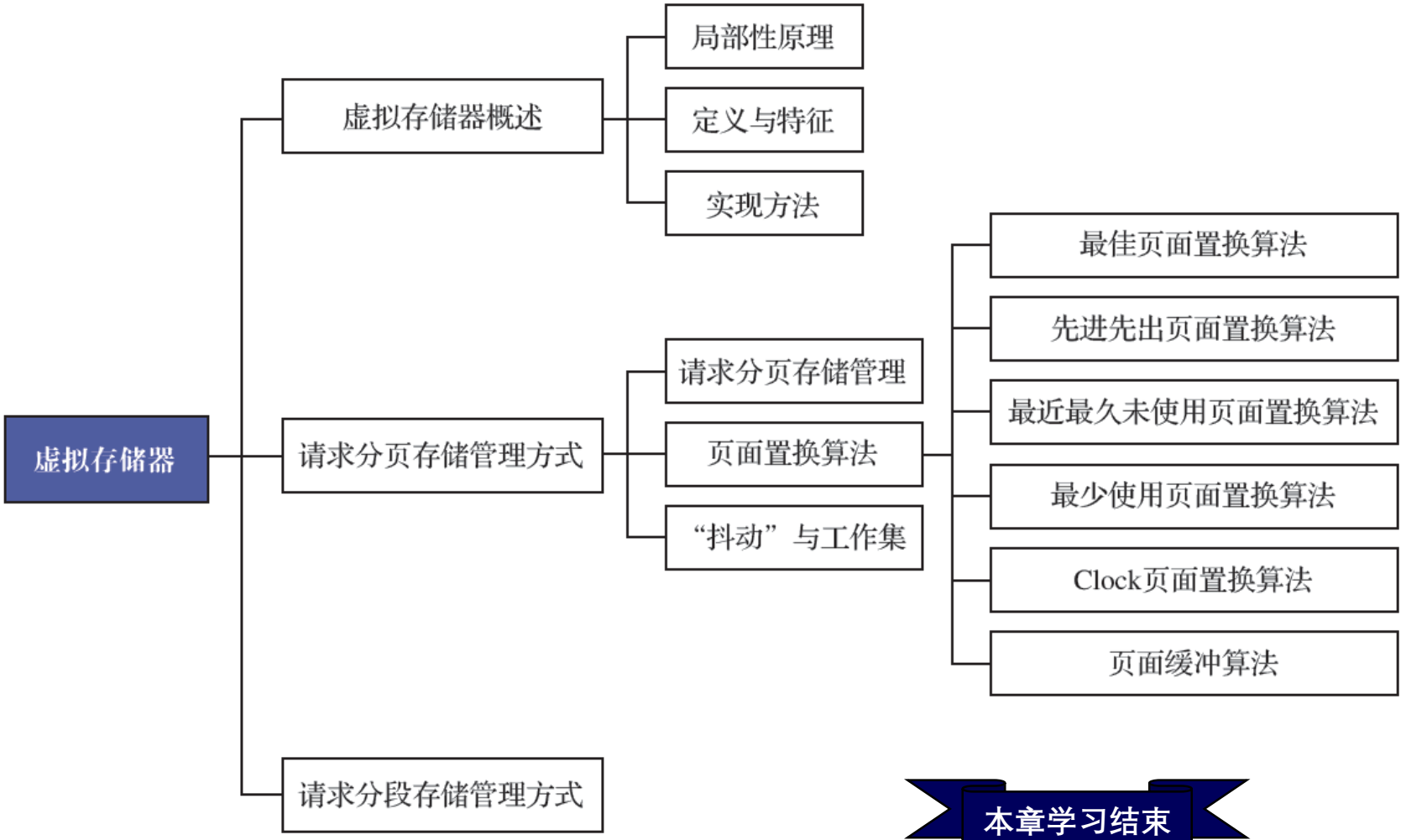
4GB的地址空间分为两个部分：

- 用户空间占据0~3GB（0xC0000000）
  - 由用户进程使用（MMU）
  - 使用请求页式存储管理
- 内核空间占据3GB~4GB
  - 由内核负责
  - 使用buddy和slab内存管理
    （zoned buddy 分配器和slab分配器）

实例2：Linux系统（以32位为例）



（第6章总结）


虚拟存储器
虚拟存储器概述
局部性原理
定义与特征
实现方法
请求分页存储管理方式
请求分页存储管理
页面置换算法
最佳页面置换算法
先进先出页面置换算法
最近最久未使用页面置换算法
最少使用页面置换算法
Clock页面置换算法
页面缓冲算法
“抖动”与工作集
请求分段存储管理方式
本章学习结束

第6章 总结

- 虚拟存储器理论*
  - 局部性原理**
  - 虚拟存储器定义**
- 请求分页存储管理方式*
  - 请求页表机制、缺页中断机构、地址变换机构**
- 六种页面置换算法**
  - OPT、FIFO、LRU、CLOCK
- 抖动与工作集*
  - 抖动的原因**
  - 预防方法*
- 请求分段存储管理
  - 请求段表机制、缺段中断机构、地址变换机构*
  - 共享段表**

第6章 总结

- **重要概念：**

虚拟存储器、请求分页存储管理、请求分段存储管理、缺页率、缺页中断、缺段中断、共享段表、可重入码、工作集、抖动等

第6章 虚拟存储器

作业

1、课后作业

- 简答：2、4、10
- 计算题：16、17
- 综合21

2、课后练习

- 简答：2
- 计算题：13
- 综合：20
- 3、预习下一章

海南大学
HAINAN UNIVERSITY

学 习 进 步 ！