深圳大学
Shenzhen University

操作系统

第六讲 虚拟存储器

谭舜泉
计算机与软件学院

第6章　虚拟存储器

第一节　基本概念

一、实存管理

- 连续内存分配，基本分页、分段内存分配具有以下共性：

1、一次性

2、驻留性

- 具有一次性、驻留性的存储器管理方法称为实存管理

第6章　虚拟存储器

第一节　基本概念

一、实存管理

- **实存管理的缺点：**

1、作业（进程）逻辑空间不能超过实际内存大小

2、同时存在的进程数目受限较大

第6章　虚拟存储器

第一节　基本概念

二、虚拟存储器

- **虚拟存储器是指对内存的虚拟，是一种实际并不存在的内存空间，它包括两个方面的概念：**

1、一级存储器概念：将大容量的外存作为内存的直接延伸，对内存、外存（交换区）实施统一管理

2、作业（进程）地址空间概念：虚存统一编址，按需要将作业（进程）放在内存或外存上

第6章　虚拟存储器

第一节　基本概念

三、虚拟存储器的特征

- **多次性**：一个进程被分成多次调入内存运行
- **对换性**：允许进程在运行过程中进行换进、换出
- **虚拟性**：从逻辑上扩充内存容量

（容量上接近于外存，运行速度上接近于内存）

第6章　虚拟存储器

第一节　基本概念

四、虚拟存储器系统

- 仅把作业（进程）的一部分装入内存便可运行作业（进程）的存储器系统
- 虚拟存储器系统必须具有请求调入功能和置换功能，从而从逻辑上对内存容量进行扩充

第6章　虚拟存储器

第一节　基本概念

五、虚拟存储器的容量

- **虚拟存储器的容量主要取决于：**

1、计算机的地址结构

2、外存储器的空间

第6章　虚拟存储器

第一节　基本概念

六、局部性原理

- **程序执行具有局部性特征，即在一较短时间内，程序的执行仅限于某个部分；访问的存储空间也局限于某个区域。**

第6章　虚拟存储器

第一节　基本概念

六、局部性原理

1、空间局部性：

- **一旦某个位置被访问了，那么它附近的位置很可能也要立即被访问**

例如：

- **程序代码的顺序执行**
- **对线性数据结构的访问或处理**

第6章　虚拟存储器

第一节　基本概念

六、局部性原理

2、时间局部性：

- 一旦某个位置被访问了，那么它往往很快会被再次访问

例如：

- 循环程序段的执行
- 子程序
- 堆栈、累积变量、计数器等

第6章　虚拟存储器

第一节　基本概念

七、虚拟存储器管理策略

- 调入策略：什么时候把所需要的那部分进程实体从外存调入内存，主要有请求调入算法，先行调入算法
- 分配策略：把调入的进程实体如何放入内存，主要有页式和段式内存分配
- 淘汰策略：当内存空间不够时，决定换出哪些内存空间

第6章　虚拟存储器

第二节　请求分页存储管理方式

一、页表机制

- 页表是请求分页存储管理的主要数据结构
- 页表项结构(p157)：

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：指示该页是否已调入内存
- 访问字段A：记录本页在一段时间内被访问的次数
- 修改位M：该页在调入内存后是否被修改过
- 外存地址：该页在外存中的位置，通常是物理块号

第6章　虚拟存储器

第二节　请求分页存储管理方式

二、缺页中断机构

- 当需要访问的页面不在内存时，便产生一**缺页中断**
- 缺页中断与一般中断**共同点**：

1、保护CPU环境（中断现场）

2、分析中断原因

3、缺页中断处理

4、恢复CPU环境（中断现场）

第6章　虚拟存储器

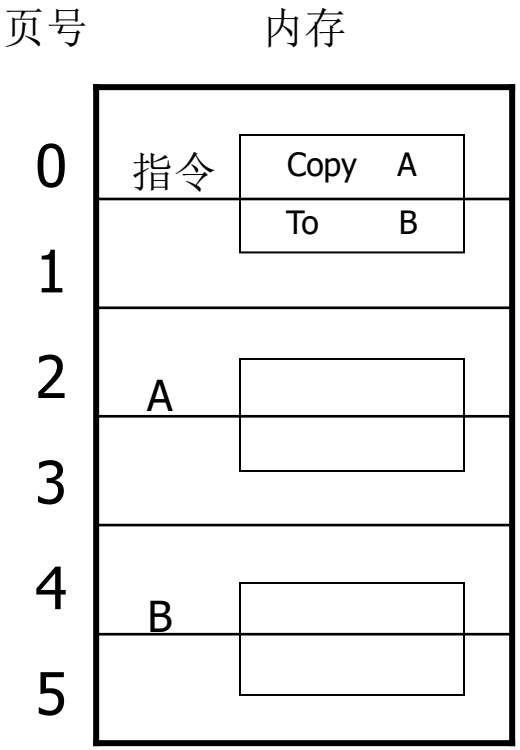
第二节　请求分页存储管理方式

二、缺页中断机构

- **缺页中断与一般中断的区别：**

1、缺页中断可能发生在指令执行过程中间

2、一条指令执行期间，可能发生多次缺页中断



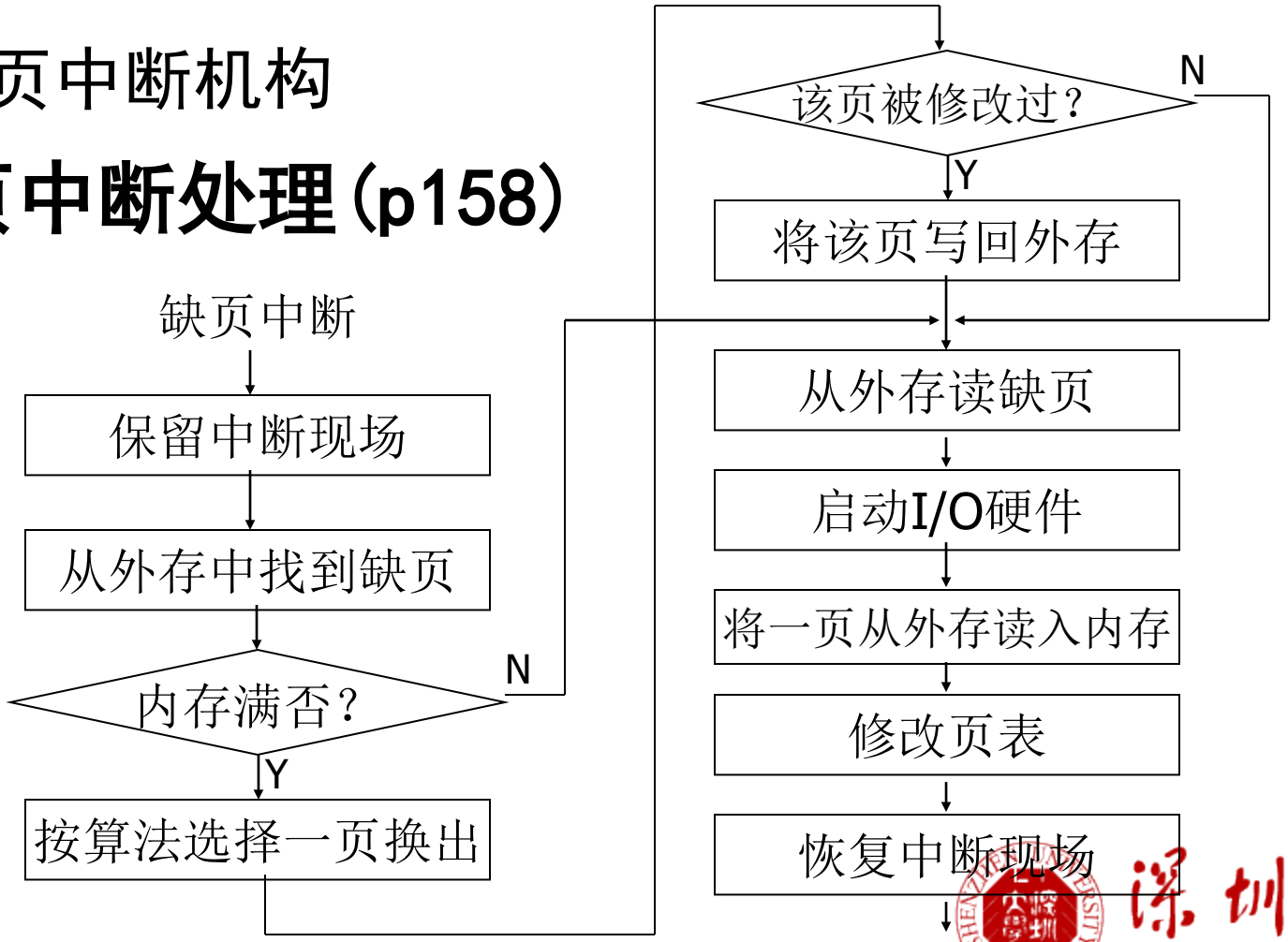
涉及6次缺页中断的指令

第6章　虚拟存储器

第二节　请求分页存储管理方式

二、缺页中断机构

- **缺页中断处理(p158)**

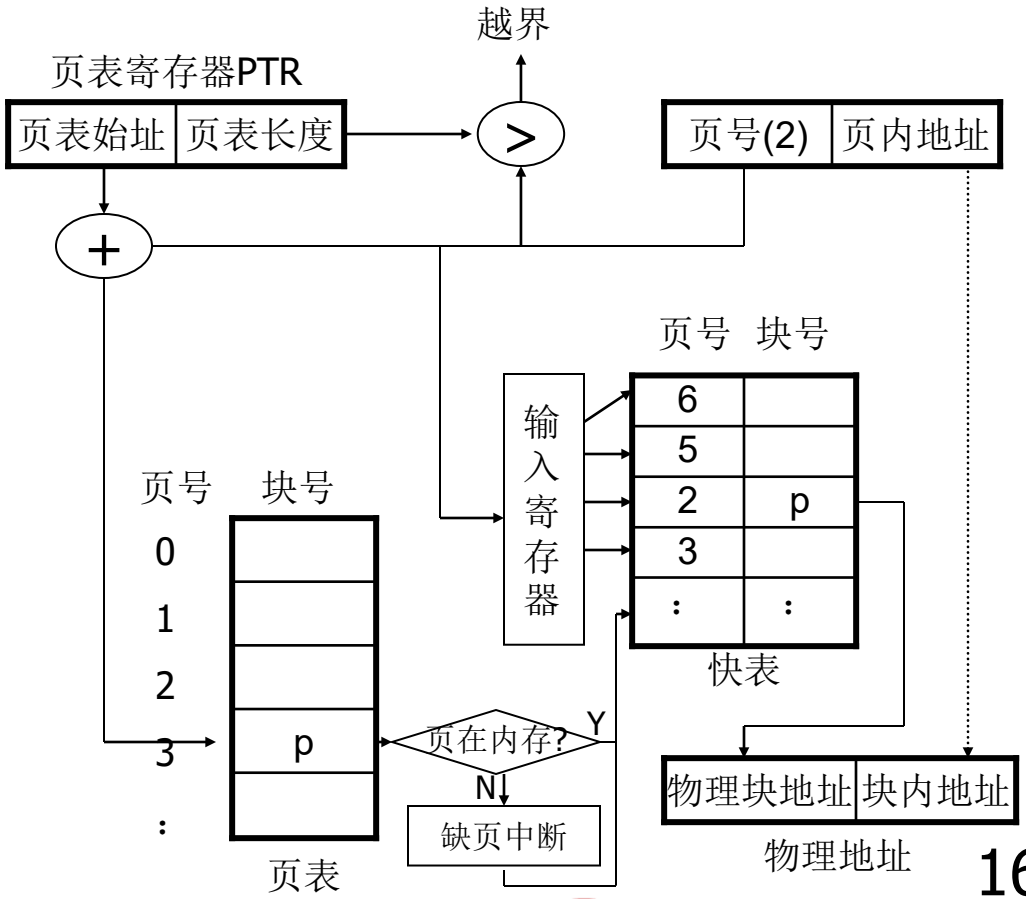


第6章　虚拟存储器

第二节　请求分页存储管理方式

三、地址变换机构

- 分页地址机构自动将逻辑地址分为页号与页内地址
- 越界检查
- 将页号送入快表中进行并行查找，如果从快表中找到所需页号，则将快表中对应的物理块号送物理地址寄存器，并修改访问位和修改位
- 否则将页表始址与页号和页表项长度的乘积相加，得页表中位置
- 如果页面已在内存，修改快表；否则，调用缺页中断，从外存调入页面，并修改快表
- 从快表中得到内存块号，送入物理地址寄存器，并修改访问位和修改位
- 页内地址送入物理地址寄存器



第6章　虚拟存储器

第二节　请求分页存储管理方式

四、页面分配

1、**最小物理块数**

- 保证进程正常运行所需要的最小物理块数
- 最小物理块数取决于指令格式、寻址方式
- 单字节、单地址指令且采用直接寻址，最少需2个物理块（指令和数据各一页）
- 2字节、2地址指令，**最小需6个物理块**

第6章　虚拟存储器

第二节　请求分页存储管理方式

四、页面分配策略

2、物理块分配算法（p160）

- 为进程分配固定个数的内存物理块，在运行期间不再改变。
  - 平均分配算法
    - 将系统中所有可供分配的物理块，平均分配给各个进程(不公平)

第6章　虚拟存储器

第二节　请求分页存储管理方式

四、页面分配策略

2、物理块分配算法

- **按比例分配算法**

  将系统中所有可供分配的物理块，按进程大小（占页面数目）按比例分配物理块

- **考虑优先权的分配算法**

  将系统中所有可供分配的物理块分成两部分，一部分按比例分配给各进程；另一部分根据优先权，为优先权高的进程多分配物理块

第6章　虚拟存储器

第二节　请求分页存储管理方式

四、页面分配策略

3、**可变页面分配**

- 为进程分配一定个数的内存物理块，在进程运行过程中，**发生缺页中断时，再分配新的物理块**

第6章　虚拟存储器

第二节　请求分页存储管理方式

四、页面置换策略

4、局部页面置换策略

- **缺页时，从本进程中，按一定算法，选择一页调出**

第6章　虚拟存储器

第二节　请求分页存储管理方式

四、页面置换策略

4、全局页面置换策略

- 缺页时，系统从空闲物理块中分配一物理块
- 如果没有新的空闲块，则从整个内存中按一定算法选择一页调出

第6章　虚拟存储器

第二节　请求分页存储管理方式

四、页面分配和置换策略

- 固定分配局部置换
- 可变分配局部置换

  先在本进程页面中置换，缺页频繁时，系统增加分配内存块数；缺页很少时，系统减少分配内存块数
- 可变分配全局置换

第6章　虚拟存储器

第二节　请求分页存储管理方式

五、页面调入策略

1、何时调入页面

- **预调页策略　（p161）**
  当发生缺页时，一次调入多个相邻页

- **请求调页策略**
  当发生缺页时，将缺页（1页）调入内存

第6章　虚拟存储器

第二节　请求分页存储管理方式

五、页面调入策略

2、从何处调入页面

- 全部从对换区调入页面
- 先从文件区调入页面，如果该页面被修改，则换出时，放到对换区；再调入时，则从对换区调入；未被修改的页面仍从文件区调入
- 未运行过的页面，从文件区调入；运行过且被换出的页面，从对换区调入

第6章　虚拟存储器

第三节　页面置换算法

一、最佳页面置换算法

- 选择被置换的页面，将是永不使用的页面，或最长时间不使用的页面
- 优点：可保证最低的缺页率
- 缺点：不可能真正实现，只可作为其它算法的评价参考

第6章　虚拟存储器

第三节　页面置换算法

一、最佳页面置换算法

- **举例：**

假定系统为进程固定分配三个物理块，且页面被访问顺序为（p163）：

7,	0,	1,	2,	0,	3,	0,	4,	2,	3,	0,	3,	2,	1,	2,	0,	1,	7,	0,	1
7	7	7	2	2	2	2	2	2	2	2	2	2	2	2	2	2	7	7	7
	0	0	0	0	0	0	4	4	4	0	0	0	0	0	0	0	0	0	0
		1	1	1	3	3	3	3	3	3	3	3	1	1	1	1	1	1	1

需要6次页面置换

第6章　虚拟存储器

第三节　页面置换算法

二、先进先出（FIFO）页面置换算法

- 选择最先进入内存的页面，即选择在内存中驻留时间最长的页面予以淘汰
- 优点：算法简单
- 缺点：性能不佳(p163)

第6章　虚拟存储器

第三节　页面置换算法

二、先进先出（FIFO）页面置换算法

- **举例：**

假定系统为进程固定分配三个物理块，且页面被访问顺序为：

7,	0,	1,	2,	0,	3,	0,	4,	2,	3,	0,	3,	2,	1,	2,	0,	1,	7,	0,	1
7	7	7	2	2	2	2	4	4	4	0	0	0	0	0	0	0	7	7	7
	0	0	0	0	3	3	3	2	2	2	2	2	1	1	1	1	1	0	0
		1	1	1	1	0	0	0	3	3	3	3	3	2	2	2	2	2	1

需要12次页面置换

第6章　虚拟存储器

第三节　页面置换算法

三、最近最久未使用（LRU）页面置换算法

1、算法

- 选择最近最久未被使用的页面淘汰
- 优点：性能较好
- 缺点：需要较多的硬件支持

第6章　虚拟存储器

第三节　页面置换算法

三、最近最久未使用（LRU）页面置换算法

- **2、举例：**

假定系统为进程固定分配三个物理块，且页面被访问顺序为：

7,	0,	1,	2,	0,	3,	0,	4,	2,	3,	0,	3,	2,	1,	2,	0,	1,	7,	0,	1
7	7	7	2	2	2	2	4	4	4	0	0	0	1	1	1	1	1	1	1
	0	0	0	0	0	0	0	0	3	3	3	3	3	3	0	0	0	0	0
		1	1	1	3	3	3	2	2	2	2	2	2	2	2	2	7	7	7

需要9次页面置换

第6章　虚拟存储器

第三节　页面置换算法

三、最近最久未使用（LRU）页面置换算法

3、硬件支持

- LRU算法需要了解：

(1)、进程的各页面各有多久时间未被进程访问

(2)、如何快速找到最近最久未使用的页面

第6章　虚拟存储器

第三节　页面置换算法

三、最近最久未使用（LRU）页面置换算法

3、硬件支持

- 移位寄存器（p165）

(1)、采用移位寄存器记录每页的使用情况

(2)、数据格式：$R=R_{n-1}R_{n-2}\ldots R_2R_1R_0$

(3)、页面被访问一次，最高位R_{n-1}置1，每隔一定时间，寄存器右移一位

(4)、R值最小的页面，是将被调出的页面

第6章　虚拟存储器

第三节　页面置换算法

三、最近最久未使用（LRU）页面置换算法

3、硬件支持

- 移位寄存器举例(p165)

R / 实页	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

第6章　虚拟存储器

第三节　页面置换算法

三、最近最久未使用（LRU）页面置换算法

3、硬件支持

- 栈

(1)、采用特殊的栈来保存当前使用的各个页面的页面号

(2)、**栈顶保留**的是最新被访问的页

(3)、**栈底保留**的是最久未被访问的页

第6章　虚拟存储器

第三节　页面置换算法

四、Clock置换算法

- Clock置换算法是LRU算法的近似算法
- Clock算法也称最近未用算法（NRU）(p166)
- Clock算法中，设置1位访问位，记录页面是否被访问过（访问过为1，未访问为0）

第6章　虚拟存储器

第三节　页面置换算法

四、Clock置换算法

- Clock置换算法实现原理

(1)、将内存所有页面用指针链成一个循环队列

(2)、Clock算法沿链查找页面

(3)、如果访问位为0，则换出

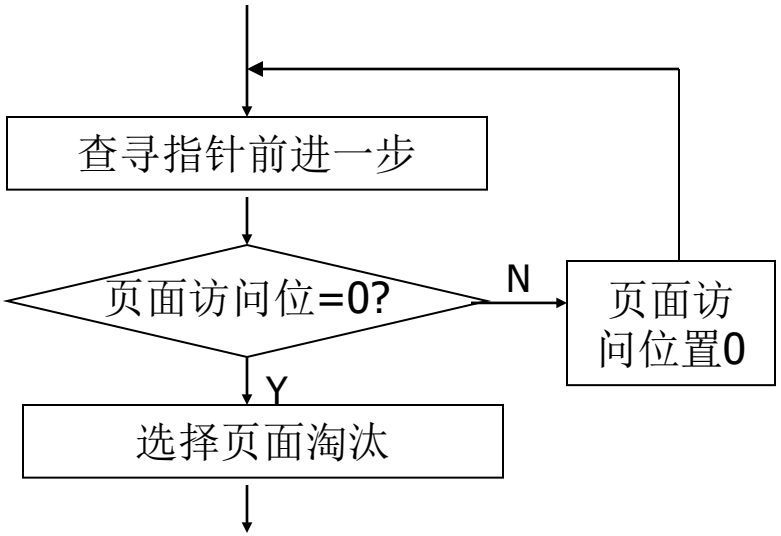
(4)、如果访问位为1，则将访问位复0；并继续沿链查找下一个页面，直到找到一个访问位为0的页面换出

第6章　虚拟存储器

第三节　页面置换算法

四、Clock置换算法

- Clock置换算法举例



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

替换指针

第6章　虚拟存储器

第三节　页面置换算法

五、改进型Clock置换算法

- 改进型Clock置换算法除设置1位访问位（A），还设置1位修改位（M）
- 将页面分成四类：

(1)、1类：A=0，M=0，最佳淘汰页

(2)、2类：A=0，M=1

(3)、3类：A=1，M=0

(4)、4类：A=1，M=1，最不应淘汰页

第6章　虚拟存储器

第三节　页面置换算法

五、改进型Clock置换算法

- 改进型Clock置换算法实现原理(p167)

(1). 查找指针前进一步，判断当前页是否为第1类页（A=0，M=0），是则选择该页淘汰

(2). 否则，继续沿链向下查找

(3). 如果沿链查找一周，没有第1类页面，则查找第2类页（A=0，M=1），找到后淘汰，并将访问过的页面访问位置为0（A=0）

(4). 如果沿链查找一周后，未找到第2类页面，则将所有页面的访问位置0，重新从第(1)步开始

第6章　虚拟存储器

第三节　页面置换算法

五、改进型Clock置换算法

- **改进型Clock置换算法举例**

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

替换指针

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

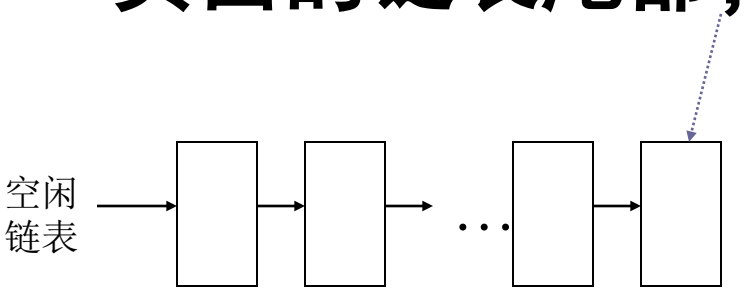
替换指针

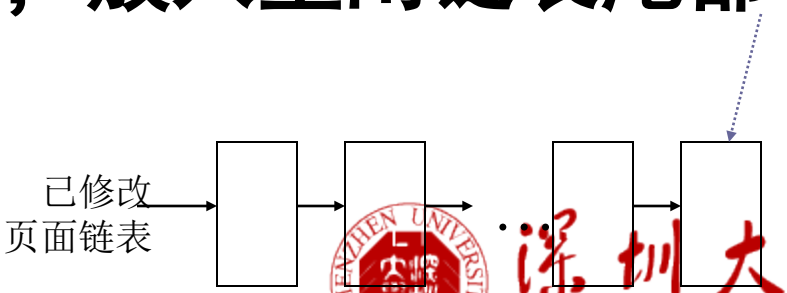
第6章　虚拟存储器

第三节　页面置换算法

六、页面缓冲（PBA）置换算法(p167)

- 采用可变分配和局部置换方式
- 当一个进程换进换出频率很低时，选择页面淘汰，以备其它进程使用
- 被淘汰的页面，如果已被修改，则放入已修改页面的链表尾部；否则，放入空闲链表尾部





第6章　虚拟存储器

第三节　页面置换算法

六、页面缓冲（PBA）置换算法

- 放在空闲链表和修改链表中的页面，不一定立即被分配出去，因此，如果进程需要，还可以从空闲链表或修改链表中，将页面恢复使用

- 如果一个进程需要新的内存块，则从空闲链表中的前面，分配一个内存块；如果空闲链表中内存块分配完毕，则从已修改链表中，分配内存块

第6章　虚拟存储器

第五节　请求分段存储管理方式

一、段表机制

- 段表是请求分段存储管理的重要数据结构
- 段表项结构：

段名	段长	基址	存取方式	访问字段A	修改字段M	存在位P	增补位	外存地址

- 存取方式：执行、只读、读/写
- 访问字段A：记录本段在一段时间内被访问的次数
- 修改位M：本段在调入内存后是否被修改过
- 存在位P：指示本段是否已调入内存
- 外存地址：本段在外存中的位置，通常是起始盘块号
- 增补位：指示运行过程中是否进行过动态增长

第6章　虚拟存储器

第五节　请求分段存储管理方式

二、缺段中断机构

- 当需要访问的段不在内存时，便产生一缺段中断
- 与缺页中断的相同点：

(1). 缺段中断同样可能发生在指令执行过程中

(2). 一条指令执行期间，可能发生多次缺段中断

- 与缺页中断的不同点：

(1). 一条指令只可能存在于一个段中

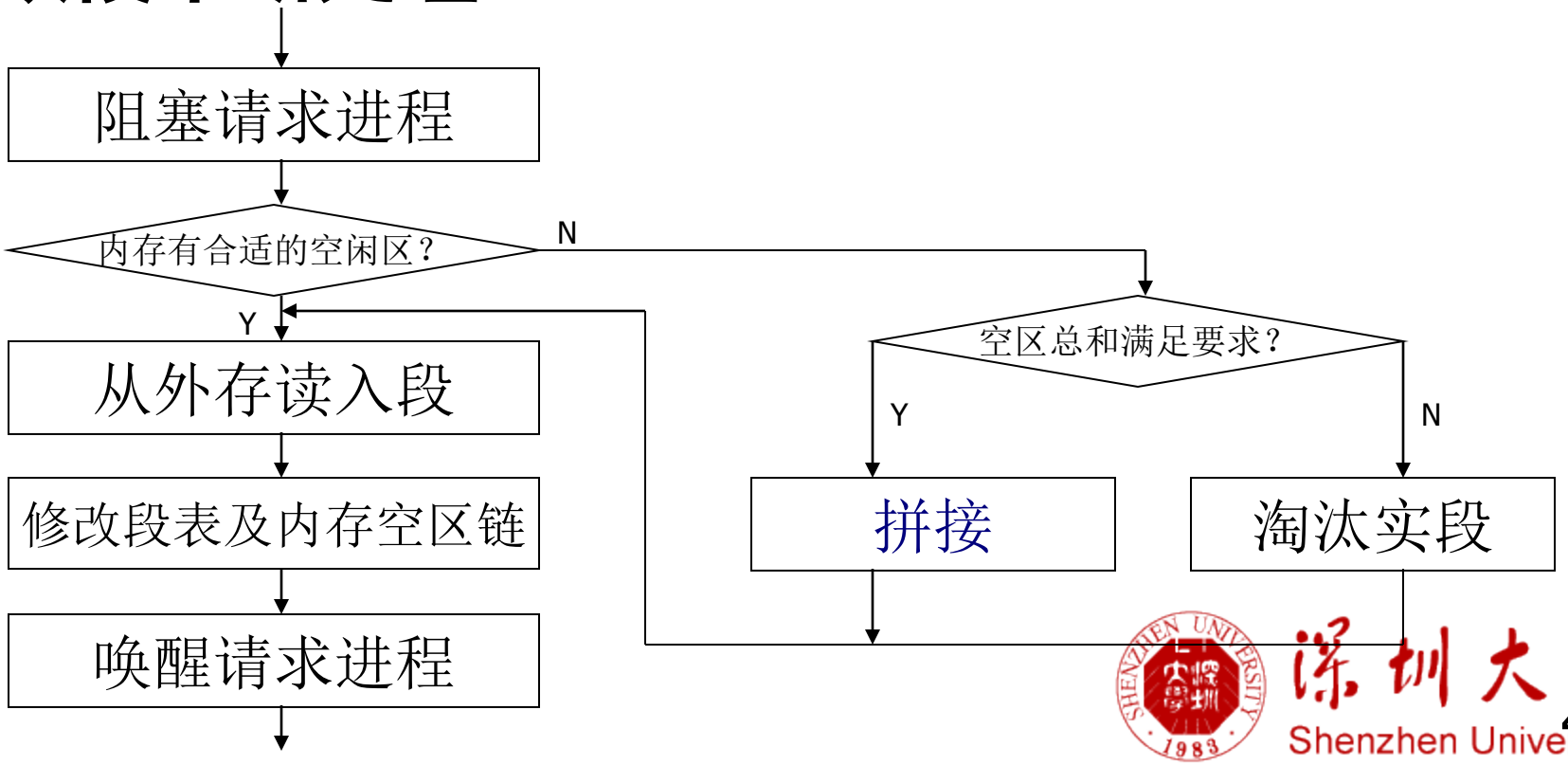
(2). 一个（组）数据只可能存在于一个段中

第6章　虚拟存储器

第五节　请求分段存储管理方式

二、缺段中断机构

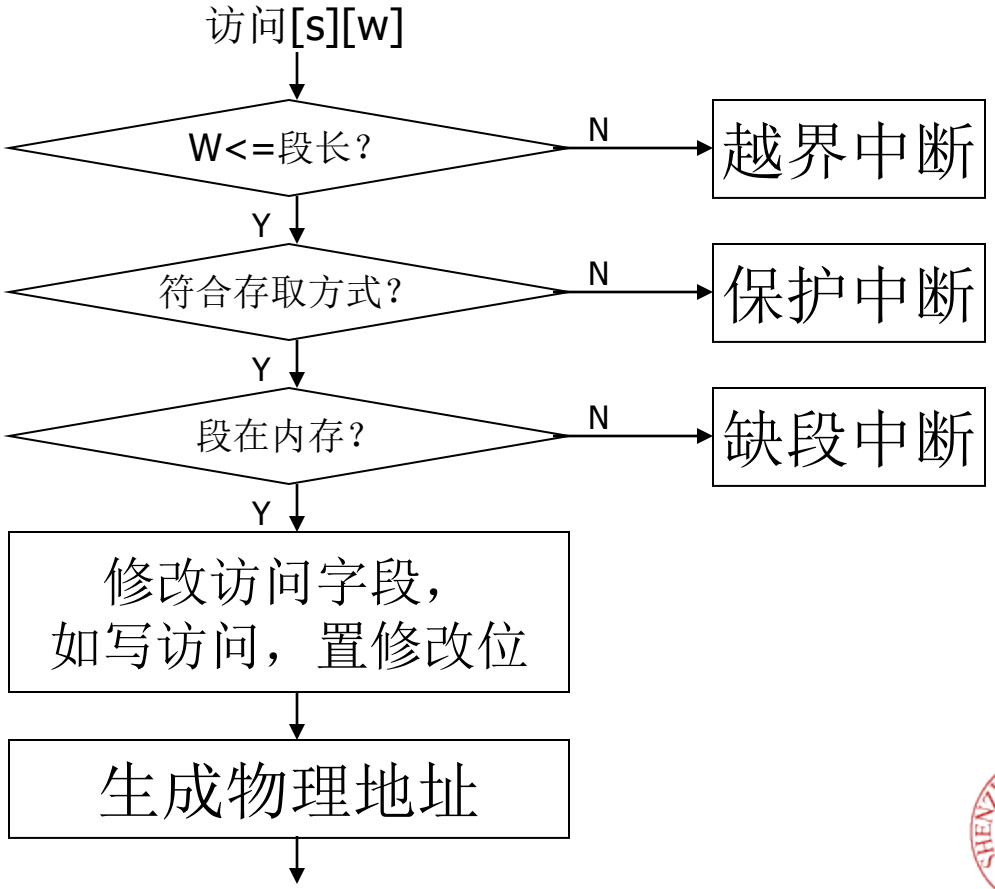
- **缺段中断处理**



第6章　虚拟存储器

第五节　请求分段存储管理方式

三、地址变换机构



第6章　虚拟存储器

第五节　请求分段存储管理方式

三、分段的共享与保护

1、共享段表

- **共享进程计数：** *记录共享的进程数*
- **存取控制字段：** *进程不同，存取权限不同*
- **段号**

第6章　虚拟存储器

第五节　请求分段存储管理方式

三、分段的共享与保护

2、共享段的分配与回收

- **分配**：*由第一个调用该共享段的进程请求系统分配*
- **回收**：*由最后一个使用该共享段的进程请求系统回收*

第6章　虚拟存储器

第五节　请求分段存储管理方式

三、分段的共享与保护

3、分段保护

- **越界检查**
- **存取控制检查**：*只读、只执行、读/写*
- **环保护机构**：

  数据（由高向低）

  调用（由低到高）