第四、五章作业

22121630 汪江豪

1. 已知某分页系统,主存容量为 64K 字节,页面大小为 1K,对一个 4 页大的作业,其 0、1、2、3 页分别被分配到主存的 2、4、6、7 块 中,试:
   (1)将十进制的逻辑地址 1023, 2500, 3500、4500 转换成物 理地址;
   (2)以十进制的逻辑地址 1023 为例画出地址变换过程图。
2. 页式存储管理允许用户的编程空间为 32 个页面(每页 1KB),主 存为 16KB。如有一用户程序为 10 页长，且某时刻该用户程序页 表见下表。若分别遇到三个逻辑地址 0AC5H,1AC5H,3AC5H 处的操作,计算并说明存储管理系统将如何处理。

逻辑页号	物理块号
0	8
1	7
2	4
3	10

3. 某操作系统采用段式管理,用户区主存为 512KB,空闲块链入空 块表,分配时截取空块的前半部分(小地址部分),初始时全部空闲 。执行申请、释放操作序列 reg(300KB),reg(100KB), release(300KB), reg(150B), reg(50KB), reg(90kB) :
   1)采用首次适应,空块表中有哪些空块?(指出大小及始址)
   2)采用最佳适应,空块表中有哪些空块?(指出大小及始址)
   3)若随后又要申请 80KB,针对上述两种情况会产生什么后果?这 说明了什么问题?

1 (1) 逻辑地址 1023

计算页号P和页内偏移量W

P = 1023/1024 = 0

W = 1023 % 1024 = 1023

0 → 2号主存块

物理地址：2×1024 + 1023 = 3071

逻辑地址 2500

P = 2500/1024 = 2

W = 2500 % 1024 = 452

2 → 6号主存块

物理地址：6×1024 + 452 = 6596

逻辑地址：3500

P = 3500/1024 = 3

W = 3500 % 1024 = 428

3 → 7号主存块

物理地址 7×1024 + 428 = 7596

逻辑地址：4500

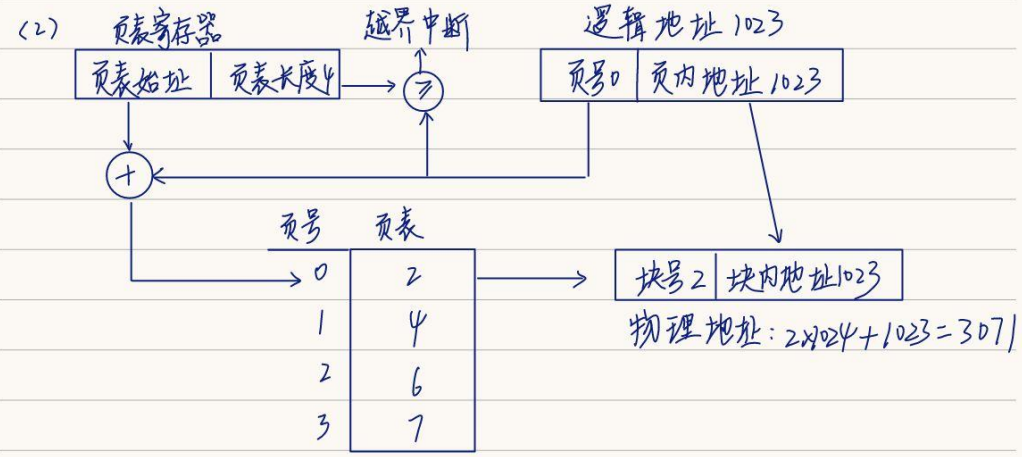
P = 4500/1024 = 4

W = 4500 % 1024 = 404

页号 4 ≥ 页表长度 4

越界中断

(2)



物理地址：2×1024 + 1023 = 3071

2. 逻辑地址中：页内地址 1KB = 2^{10}B，10位

32 = 2^5，逻辑页号 5位

物理地址中：16KB ÷ 1KB = 16 = 2^4，块号4位

逻辑地址转为二进制：

0A C5H = 000 1010 1100 0101 B

↓

0100 10 1100 0101 B 也即 12C5H

所以访问物理地址 12C5H

1A C5H = 0001 1010 1100 0101 B

↓

逻辑页号为 6，页号 < 页表长度，但不在页表中，

进行缺页中断处理

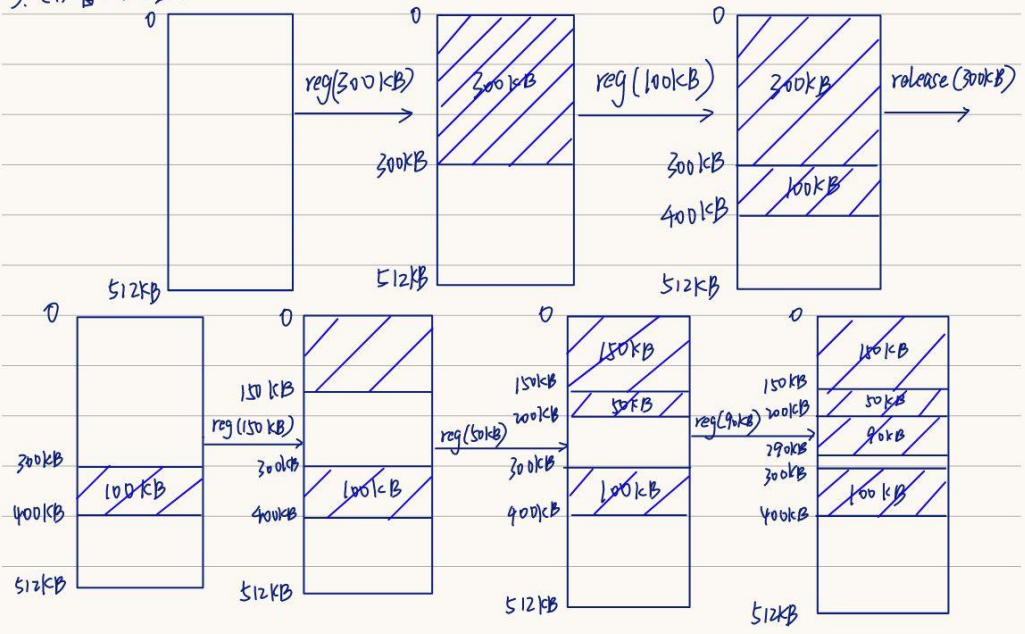
3A C5H = 0011 1010 1100 0101 B

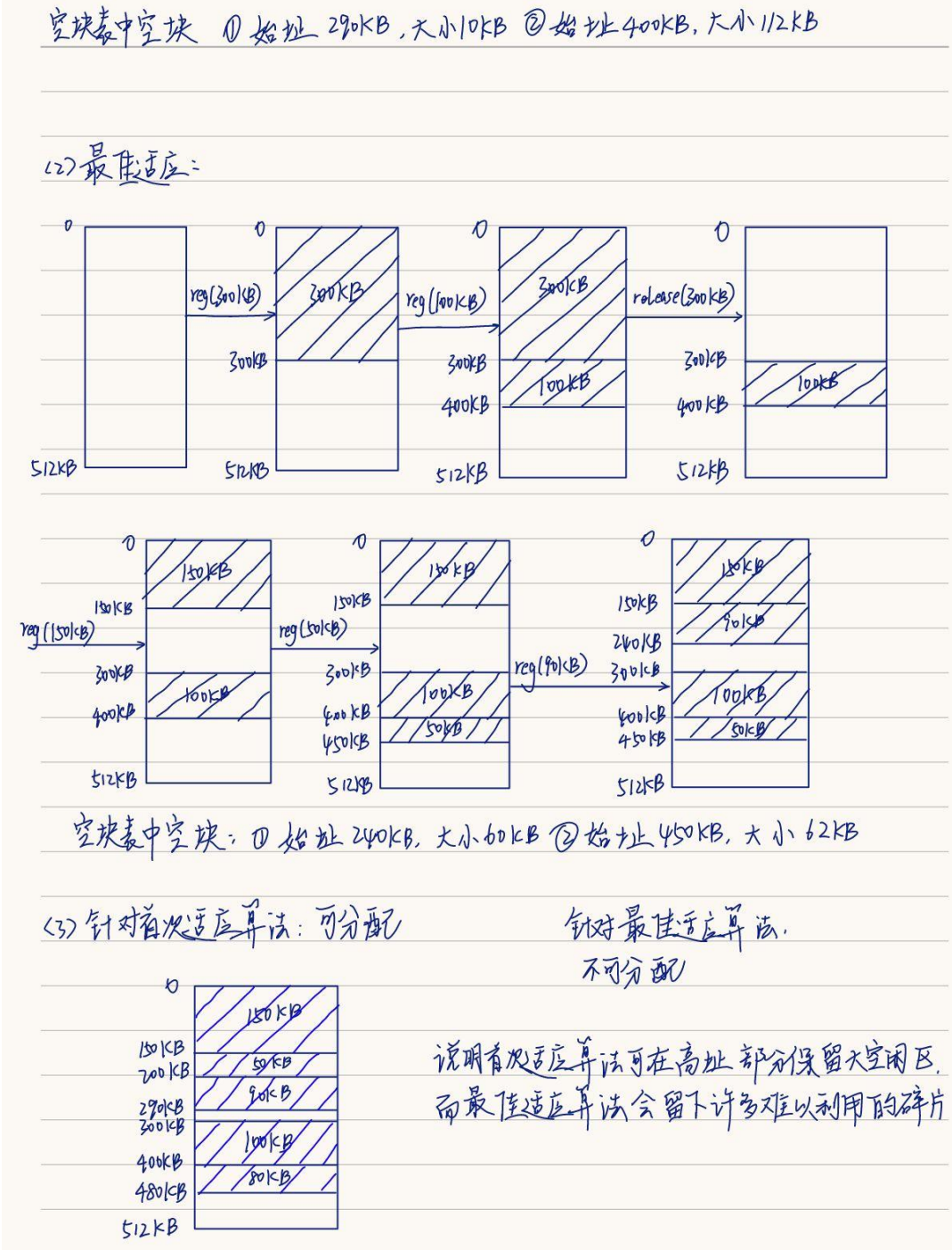
↓

逻辑页号 14，用户程序太长，页号 ≥ 页表长度

进行越界中断处理

3. (1) 首次适应：





4.1 简述动态分区分配的基本数据结构和三个算法（首次适应 算法、最佳适应算法、最差适应算法）

4.1.1 基本数据结构：

空闲分区表：在系统中设置一张空闲分区表，用于记录每个空闲分区的情况。每个空闲分区占一个表目，表目中包括分区号、分区大小和分区始址等数据项。

空闲分区链：在每个分区的其实部分设置一些用于控制分区分配的信息，以及用于链接各

分区所用的前向指针，在分区尾部则设置一后向指针，通过前后链接指针，将所有的空闲分区链接成一个双向链。

4.1.2 算法：

a. 首次适应算法（first fit）：每次从低地址按序寻找，找到首个能满足大小的空闲分区；若从头到尾找不到满足要求的分区，则内存分配失败。

优点：优先利用内存中低址部分的空闲分区，保留了高址部分的大空闲分区。

缺点：低址部分被不断划分，会留下许多难以利用的、很小的空闲分区碎片，且每次查找从低址部分开始，增加了查找的开销。

b. 最佳适应算法（best fit）：每次分配内存时，总是把能满足要求的、最小的空闲分区分配给作业。该算法要求将所有的空闲分区按其容量以从小到大的顺序形成一个空闲分区链。

缺点：分配后所切割下的剩余部分总是最小的，会留下许多难以利用的碎片。

c. 最差适应算法（worst fit）：分配内存时，总是挑选一个最大的空闲分区分割给作业使用。该算法要求将所有的空闲分区，按其容量以从大到小的顺序形成一空闲分区链，查找时，只看第一个分区能否满足即可。

优点：可使剩下的空闲区不至于太小，产生碎片的可能性最小，对中小作业有利，同时查找效率很高。

缺点：缺乏大的空闲分区。

4.2 某系统采用页式地址映射，页表中存储的内容如下图所示

物理块号	状态位P	访问位A	修改位M	其它信息

4.2.1 请写出：物理块号、状态位、访问字段和修改位的作用。

物理块号：用于指示该页在内存中对应的内存块号。

状态位 P：用于指示该页是否已调入内存，供程序访问时参考。

访问字段 A：用于记录本页在一段时间内被访问的次数，或记录本页最近已有多长时间未被访问，供置换算法或程序在选择换出页面时参考。

修改位 M：标识该页在调入内存后是否被修改过。

4.2.2 若系统采用 LRU 算法实现页面置换，需要计算页面的最近访问时间，但是上 面页表中并没有页面最近访问时间，请基于上述页表中的信息，给出一种实 现 LRU 算法的方案。

可以使用哈希表+双向链表的方式。

哈希表中，键为页号，值为指向双向链表节点的指针。

双向链表中的节点按页面的访问时间排序，最近访问的页面在表头，最久未访问的页面在表尾。

当访问一个页面时，如果它在哈希表中，从双向链表中将该节点移动到链表的表头。如果页面不在哈希表中，且若链表满了，删除表尾节点（即最久未使用的页面），在表头插入新节点，更新哈希表。

5. 一进程已分配到 4 个页帧,见下表(编号为十进制,从 0 开始)。当 进程访问第 4 页时,产生缺页中断,请分别用 FIFO、 LRU、NRU 算 法，决定缺页中断服务程序选择换出的页面。

虚拟页号	页帧	装入时间	最近访问时间	访问位	修改位
2	0	60	161	0	1
1	1	130	160	0	0
0	2	26	162	1	0
3	3	20	163	1	1

5.1 FIFO 算法：FIFO 算法总是换出最早进入内存的页面，因此换出虚拟页号为 3 的页面。最近最久未使用算法（LRU）：LRU 算法总是换出最近访问时间最早的页面，因此换出虚拟页号为 1 的页面。

5.2 Clock 置换算法（NRU）：

简单型 NRU 算法：从低地址开始检索，将访问位为 0 的页面换出，因此换出虚拟页号为 2 的页面。

5.3 改进型 NRU 算法：第一轮寻找 00，找到，因此换出虚拟页号为 1 的页面。

6. 在一个请求分页存储管理系统中，一个作业的页面走向为 4，3，2，1，4，3，5，4，3，2，1，5，当分配给作业的物理块数分别为 3 和 4 时，试计算采用下述页面淘汰算法时的缺页率（假设，开始执行时主存中没有页面），并比较结果。1）最佳置换算法。2）先进先出置换算法。3）最近最久未使用算法。（给出置换和计算过程）

6.1 最佳置换算法：

6.1.1 分配给作业的物理块数为 3：

访问页面	4	3	2	1	4	3	5	4	3	2	1	5
内存块 1	4	4	4	4			4			2	1	
内存块 2		3	3	3			3			3	3	
内存块 3			2	1			5			5	5	
是否缺页	✓	✓	✓	✓			✓			✓	✓	

$$缺页率 = \frac{7}{12} = 58.3\%$$

6.1.2 分配给作业的物理块数为 4：

访问页面	4	3	2	1	4	3	5	4	3	2	1	5
内存块 1	4	4	4	4			4				1	
内存块 2		3	3	3			3				3	
内存块 3			2	2			2				2	
内存块 4				1			5				5	
是否缺页	✓	✓	✓	✓			✓				✓	

$$缺页率 = \frac{6}{12} = 50\%$$

6.2 先进先出置换算法

6.2.1 分配给作业的物理块数为 3

访问页面	4	3	2	1	4	3	5	4	3	2	1	5
内存块 1	4	4	4	1	1	1	5			5	5	
内存块 2		3	3	3	4	4	4			2	2	
内存块 3			2	2	2	3	3			3	1	
是否缺页	✓	✓	✓	✓	✓	✓	✓			✓	✓	

$$缺页率 = \frac{9}{12} = 75\%$$

6.2.2 分配给作业的物理块数为 4

访问页面	4	3	2	1	4	3	5	4	3	2	1	5
内存块 1	4	4	4	4			5	5	5	5	1	1
内存块 2		3	3	3			3	4	4	4	4	5
内存块 3			2	2			2	2	3	3	3	3
内存块 4				1			1	1	1	2	2	2
是否缺页	✓	✓	✓	✓			✓	✓	✓	✓	✓	✓

$$缺页率 = \frac{10}{12} = 83.3\%$$

6.3 最近最久未使用算法

6.3.1 分配给作业的物理块数为 3

访问页面	4	3	2	1	4	3	5	4	3	2	1	5
内存块 1	4	4	4	1	1	1	5			2	2	2
内存块 2		3	3	3	4	4	4			4	1	1
内存块 3			2	2	2	3	3			3	3	5
是否缺页	✓	✓	✓	✓	✓	✓	✓			✓	✓	✓

$$缺页率 = \frac{10}{12} = 83.3\%$$

6.3.2 分配给作业的物理块数为 4

访问页面	4	3	2	1	4	3	5	4	3	2	1	5
内存块 1	4	4	4	4			4			4	4	5
内存块 2		3	3	3			3			3	3	3
内存块 3			2	2			5			5	1	1
内存块 4				1			1			2	2	2
是否缺页	✓	✓	✓	✓			✓			✓	✓	✓

$$缺页率 = \frac{8}{12} = 66.7\%$$

(选做)7. Gribble 公司正在开发一款 64 位的计算机体系结构，也就是说，在访问内存时，最多可以使用 64 位的地址。假设采用的是虚拟页式存储管理，现在要为这款机器 设计相应的地址映射机制。

1) 假设页面的大小是 4KB，每个页表项的长度是 4B，而且必须采用三级页表结构，每级页表 结构中的每个页表都必须正好存放在一个物理页面中，请问在这种情形下，如何实现地址 的映射?具体来说，对于给定的一个虚拟地址，应该把它划分为几部分，每部分的长度分别 是多少，功能是什么?另外，采用这种地址映射机制后，可以访问的虚拟地

址空间有多大?（ 提示：64 位地址并不一定全部用上。）

页面大小（也是物理页面大小）4KB=2^{12}B，设置 12 位页内地址。

每个页表项长 4B，一个页表填满一个物理页面，因此，一共有 4KB / 4B = 1K = 1024 个页表项。所以每级页表索引 10 位。有三级页表。格式如下：

空闲位 22 位	一级页号 10 位	二级页号 10 位	三级页号 10 位	页内偏移量 12 位
63 42	41 32	31 22	21 12	11 0

每个物理地址可指向一个字节，可访问的虚拟地址空间：2^{42} B = 4TB。

2) 假设每个页表项的长度变成了 8B,而且必须采用四级页表结构,每级页表结构中的页表都必须正好存放在一个物理页面中,请问在这种情形下,系统能够支持的最大的页面大小 是多少?此时,虚拟地址应该如何划分?

设页面大小（也是物理页面大小）为 X 字节（X 为整数），有$\frac{x}{8}$个页表项，4 级页表。所以

$$\log_2 X + 4\log_2 \frac{X}{8} \leq 64$$

求得

$$X \leq 2^{15}$$

所以 X = 32K，页面大小 32KB，页内偏移量为 15 位，一个物理页面有 32KB / 8B 个页表项即 4K = 2^{12} 个，所以页表索引 12 位，格式如下

空闲位 1 位	一级页号 12 位	二级页号 12 位	三级页号 12 位	四级页号 12 位	页内偏移量 15 位
63 62	51 50	39 38	27 26	15 14	0