

2. 一个具有两道作业的批处理系统，作业调度采用短作业优先调度算法，进程调度采用基于优先数的抢占式调度算法（作业优先数即为对应的进程优先数，优先数越小，优先级越高）。

| 作业名 | 到达时间 | 估计运行时间（分钟） | 优先数 |
|------|------|------------|-----|
| Job1 | 9:00 | 30 | 4 |
| Job2 | 9:10 | 35 | 2 |
| Job3 | 9:35 | 15 | 3 |
| Job4 | 9:40 | 10 | 1 |

试回答如下问题（6分）：

- （一） 列出各作业进入主存时间与结束时间，并计算作业的平均周转时间；
- （二） 请列出其他几种常见的进程调度算法（不少于2种）；
- （三） 进程调度算法往往需要平衡 IO 密集型和计算密集型进程占用 CPU 的时间，设计此类算法最关键的问题是什么（请结合课上介绍的某个具体的算法进行讨论）？
- （四） 请简要讨论抢占式调度算法与非抢占式调度算法的优缺点。

答：（一）

| | 进入主存 | 结束 |
|------|------|-------|
| Job1 | 9:00 | 10:30 |
| Job2 | 9:10 | 9:45 |
| Job3 | 9:55 | 10:10 |
| Job4 | 9:45 | 9:55 |

平均周转时间为 $(90+35+35+15)/4=43.75\text{min}$

- （二） 时间片轮转、先来先服务、最高响应比等

（三） 最关键是预测进程的行为。例如老化算法，可根据进程历史行为来预测其后续行为；

（四） 抢占式：优点：不会使得一个低优先级进程长时间占用 CPU，更好响应优先级高的进程；缺点：进行进程调度浪费太多时间

非抢占式：优点：进程调度少，CPU 利用率高；缺点：可能会使一个低优先级进程霸占 CPU，紧急的进程不能响应

3. 若某个系统中有 5 个并发进程分别是 P0、P1、P2、P3、P4，四类资源，分别标记为 A、B、C、D，系统目前各进程的资源分配和申请情况如下表所示：

| Process | Allocation | | | | Claim | | | | Available | | | |
|---------|------------|---|---|---|-------|---|---|---|-----------|---|---|---|
| | A | B | C | D | A | B | C | D | A | B | C | D |
| P0 | 0 | 0 | 2 | 1 | 2 | 0 | 3 | 2 | 1 | 2 | 1 | 2 |
| P1 | 1 | 0 | 0 | 0 | 3 | 1 | 2 | 1 | | | | |
| P2 | 1 | 2 | 0 | 0 | 2 | 3 | 1 | 2 | | | | |
| P3 | 0 | 1 | 1 | 0 | 0 | 2 | 1 | 1 | | | | |
| P4 | 1 | 0 | 1 | 2 | 2 | 1 | 3 | 4 | | | | |

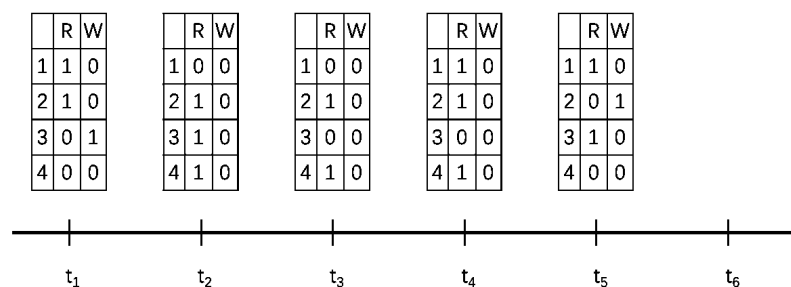
试回答如下问题（6 分）：

- （一）该系统目前是否安全，如果安全，请给出一个安全序列？
- （二）若此时 P1 发起资源申请(1, 0, 1, 0)，能否满足 P1 申请（简要说明理由）？
- （三）若此时 P4 发起资源申请(1, 1, 1, 1)，能否满足 P4 申请（简要说明理由）？

答：

- （一）p2->p0->x->x->x （x 可取 1,3,4 中任意值且不重复，下同）
 p2->p3->x->x->x
 p3->p2->x->x->x
 p3->p4->x->x->x 皆可。
- （二）不能，系统不存在安全序列。
- （三）可以，存在安全序列：p3->p4->x->x->x

4. 一个 32 位系统的计算机，具有 1GB 物理内存，其上的操作系统采用请求分页存储管理技术，页面大小为 4KB，页表项大小为 4B。系统中某进程固定分配了四个页框，已驻留 1~4 号页面的访问情况如下图所示（ $t_1 \sim t_6$ 为等间隔时刻），试回答如下问题（9 分）：



- （一）如果采用二级页表，则一个逻辑地址如何划分（页目录号、页号和页内偏移）？
- （二）在支持请求分页的系统中，一个页表项除了页框号还需要包括哪些主要信息（简要描述其作用）？
- （三）什么原因导致图中 3 号页面的 W 位从 t_1 时刻的 1 变为 t_2 时刻的 0？
- （四）若在 t_5 和 t_6 时刻之间需要执行机器指令 `mov [2000], REG`，将一个 32 位的 REG 值存放到逻辑地址 2000 开始存储器中，会引发缺页异常吗？若发生缺页，则试着分别采用 LFU（最不常用）和 Aging（老化算法，采用 4 位寄存器 存放页面引用情况）给出被淘汰的页面（结合 W 位考虑），并说明理由。如果后续没有再对逻辑地址 2000 对应的页面进行任何处理，则该页的 R 位和 W 位在 t_6 时刻分别为多少？
- （五）Aging 算法要接近 LRU（最近最少使用）算法的效果，需要采取什么措施？

答：

（一）逻辑地址前 10 位为页目录号，中间 10 位为页号，后 12 位为页内偏移。

（二）缺页（驻留）标志：指出对应页是否已经装入主存，访问一个页面时，如果某页所对应栏的驻留标志位为 1，则表示该页已经在主存。

辅存地址，

脏位：判断此页是否被修改，淘汰时写回内存

引用位：在该页被引用时（无论是读或写）设置，其值被用来帮助操作系统进行页面淘汰

访问标志：表示页面是否被访问

（三）3 号页面被写回了磁盘

（四）逻辑地址为 2000，对应第 0 号页面，引发缺页异常

LFU: t_5 时页面 2 和 4 的 $R=0$ ，最近未被访问。而页面 2 的 $w=1$ ，最近被修改，因此淘汰页面 4

Aging: 此时页面 1 为 1100，页面 2 为 0111，页面 3 为 1001，页面 4 为 0111，页面 2 和 4 一样小，但页面 2 的 $w=1$ ，最近被修改，所以淘汰页面 4

$R=0$ ， $W=1$

（五）①增加寄存器位数，使其存更多历史信息 ②减小时间间隔

5. 设某 UNIX 系统, 文件系统的每个 inode 包含直接索引项 10 个和一、二、三级间接索引项各 1 个, 物理块大小为 512B, 每个索引项占 4B, 每个目录项占 16B (包含文件名和 inode 号), 每个 inode 占 128B. test 目录下除了. 和.. 仅存在两个文件 demoA. dat 和 demoB. dat, 且互为硬链接文件, 文件大小为 2000B. 试阅读如下代码并回答问题(9 分):

```
1  #include <stdio.h>
2  #include <unistd.h>
3  #include <fcntl.h>
4
5  int main(void)
6  {
7      int fd;
8      char buf[1600];
9
10     fd = open("/test/demoA.dat", O_RDONLY);
11     if (fork() == 0)
12     {
13         sleep(1);
14         printf("childA: %ld\n", read(fd, buf, 1000));
15     } else {
16         printf("father: %ld\n", read(fd, buf, 1600));
17         if (fork() == 0)
18         {
19             sleep(2);
20             fd = open("/test/demoB.dat", O_RDONLY);
21             printf("childB: %ld\n", read(fd, buf, 1000));
22         } else sleep(3); // wait for all children to end
23     }
24     printf("Done.\n");
25 }
```

- (一) 该文件系统中, 一个普通文件的理论最大尺寸是多少字节? (给出计算过程)
- (二) test 目录文件的大小? 占了多少磁盘空间? (给出计算过程)
- (三) 程序运行过程中, OS 内核共创建了几个用户已打开文件表项, 几个系统已打开文件表项? 几个活动 inode (不考虑目录文件的 inode)? 并画出程序运行过程中用户已打开文件表项、系统已打开文件表项、活动 inode 之间的引用关系示意图;
- (四) 第 14 行代码中, 请描述 read 系统调用的大致工作过程? 共读入几个物理块?
- (五) 上述代码若能正常执行完成, 则输出的内容是什么?

答: (一) 每个物理块中的索引个数为 $512\text{B}/4\text{B}=128$ 个

直接索引项: $10*512\text{B}$

一级间接索引项: $128*512\text{B}$

二级间接索引项: $128*128*512\text{B}$

三级间接索引项: $128*128*128*512\text{B}$

最大尺寸为

$10*512\text{B}+128*512\text{B}+128*128*512\text{B}+128*128*128*512\text{B}=1082201088\text{B}=1056837\text{KB}$

(二) test 目录文件共包含 4 个目录项, 大小为 $16\text{B}*4=64\text{B}$ 。占据一个物理块, 所占磁盘空间为 512B。

(三) OS 内核共创建了 4 个用户已打开文件表项, 2 个系统已打开文件表项, 1 个活动 inode。

画图见下页

(四) 根据 fd 在 inode 表中找到对应表项, 由 inode 表中的索引项得到数据块的开始地址和

指针位置，按顺序读取 1000B 数据到缓冲区。父进程先读 1600，所以第十四行只读了 400B，读入 1 个物理块。

(五)

father: 1600

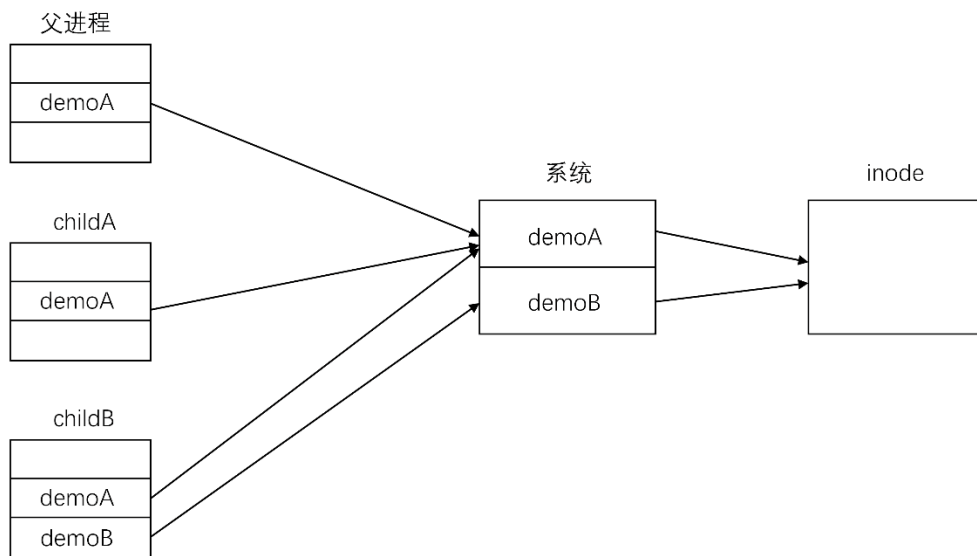
childA: 400

Done.

childB: 1000

Done.

Done.



6. 设有一个包含了 16 个磁头(编号 0-15)和 120 个柱面(编号 0-119)的磁盘, 每磁道扇区数 200 个(编号 0-199), 每个扇区 512B, 磁盘的转速为 7200rpm (转每分钟), 相邻柱面间的平均寻道时间为 1ms。试回答如下问题 (4 分):

(一) 该磁盘容量是多少字节? 最大理论传输速率是多少字节每秒? 访问某个扇区的平均循环等待时间是多少?

(二) 现磁盘移动臂刚处理完 10 号柱面的请求, 目前正在 25 号柱面读数据。接着依次到来磁盘访问请求(柱面号): 76、7、56、5、90、47、32、106、21、115, 试分别用先来先服务、电梯调度、扫描和最短寻找时间优先算法, 给出完成访问请求的顺序, 并计算各算法中移动臂经历的总柱面数。

答:

(一) 容量为 $16 \times 120 \times 200 \times 512 = 196608000 \text{B} = 192000 \text{KB}$

最大理论传输速率为 $7200 \times 200 \times 512 / 60 = 12000 \text{ KB/s}$

平均循环等待时间为 $1 / 120 \times (1/2) = 1/240 \text{ s}$

(二) 先来先服务: 25->76->7->56->5->90->47->32->106->21->115

经历的总柱面数为: 616

电梯调度: 25->32->47->56->76->90->106->115->21->7->5

经历的总柱面数为: 200

扫描: 25->32->47->56->76->90->106->115->119->21->7->5

经历的总柱面数为: 208

最短寻找时间优先: 25->21->32->47->56->76->90->106->115->7->5

经历的总柱面数: 208

| | |
|----|--|
| 得分 | |
|----|--|

二、编程题（8）

- Hoare 类型的管程实现如下图所示，为保证管程内各进程执行的串行性，其基本思想是执行 signal 操作的进程阻塞(next queue)，被唤醒的进程执行，请使用信号量 PV 操作模仿 Hoare 类管程的实现，给出一种新的管程，其基本思想是执行 signal 操作的进程继续执行，被唤醒的进程仍需等待直到执行 signal 操作的进程退出或者再次进入等待状态。（需要定义条件变量、管程体、给出 signal 和 wait 实现代码、管程方法体的封装代码等）

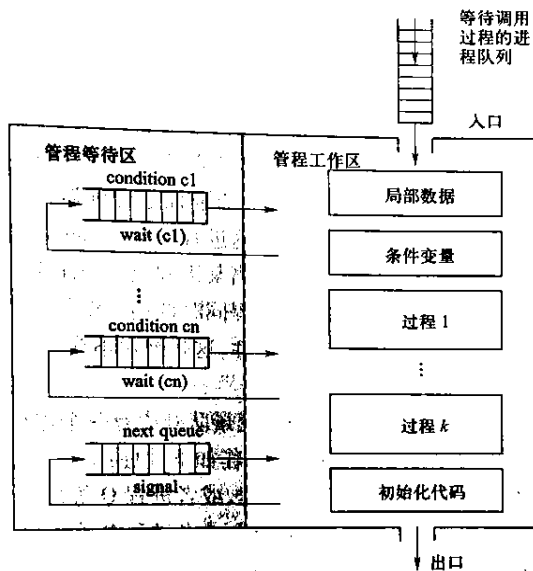


图 3-3 管程结构示意图

答：（思路）

真题解决思路与《操作系统教程》提供的思路一致，即每个条件变量引入一个信号量，每个管程另外需要一个互斥信号量和一个 next 信号量用于阻塞就绪状态的管程内进程。主要实现区别在于 signal 操作需要将满足条件的进程转移到 next 信号量上。

