



操作系统内容要点

期末事宜

- 期末考试：闭卷
- 作业和实验报告提交：
 - 截止时间：2023-06-15 23:00
 - 作业提交到网络学堂
 - 两份实验报告提交到OSTEC系统：
 - 实验1、2、3合为一份
 - 实验4、5、6合为一份

题型

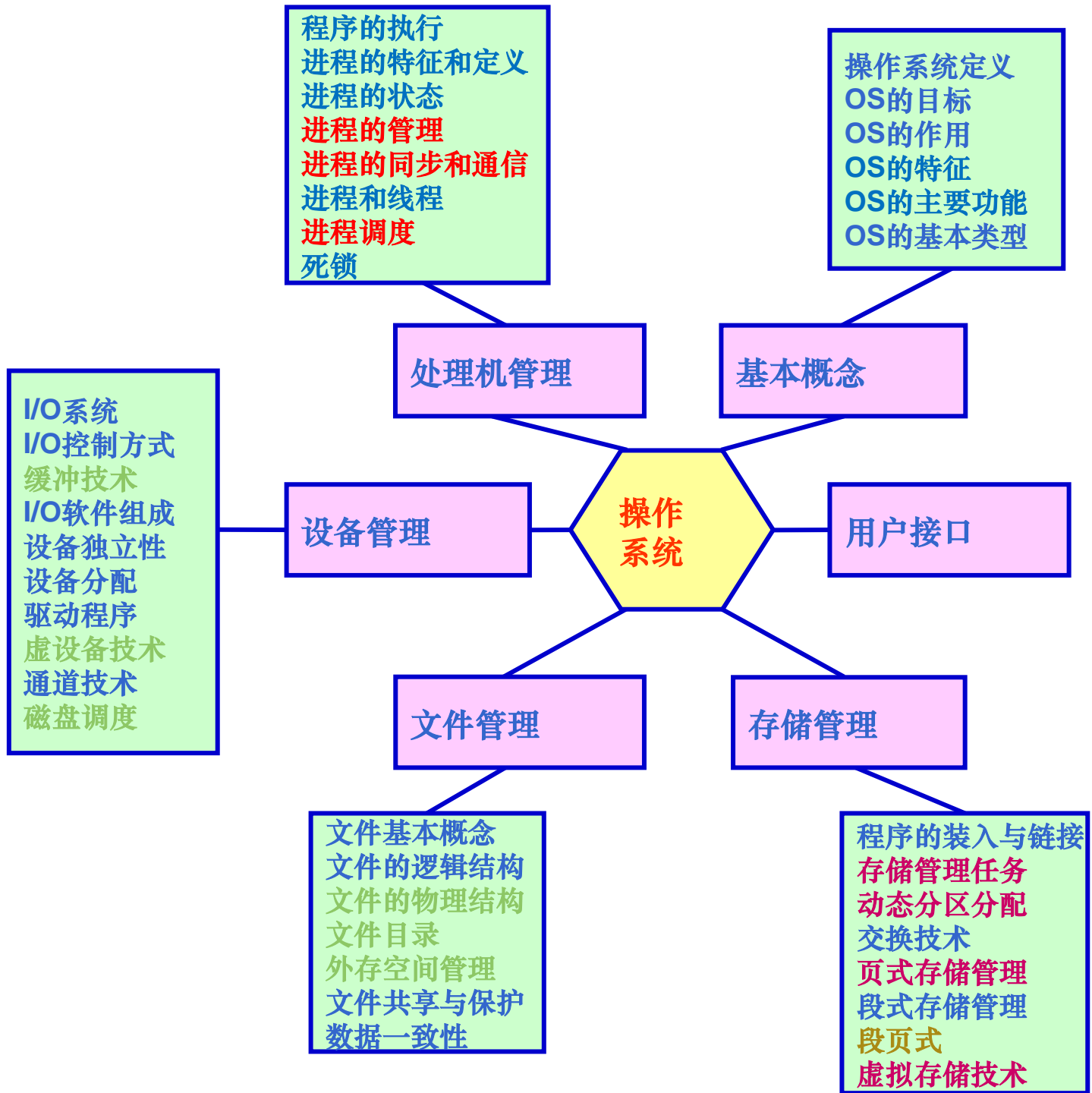
- 简答（6道，30分）
- 应用分析（5道，50分）
- 程序阅读（1道，5分）
- 程序设计（1道，15分）

知识点分布

- 引论 5分
- 进程 25分
- 存储 25分
- 设备 10分
- 文件 15分
- Shell 10分
- 信号量编程 15分

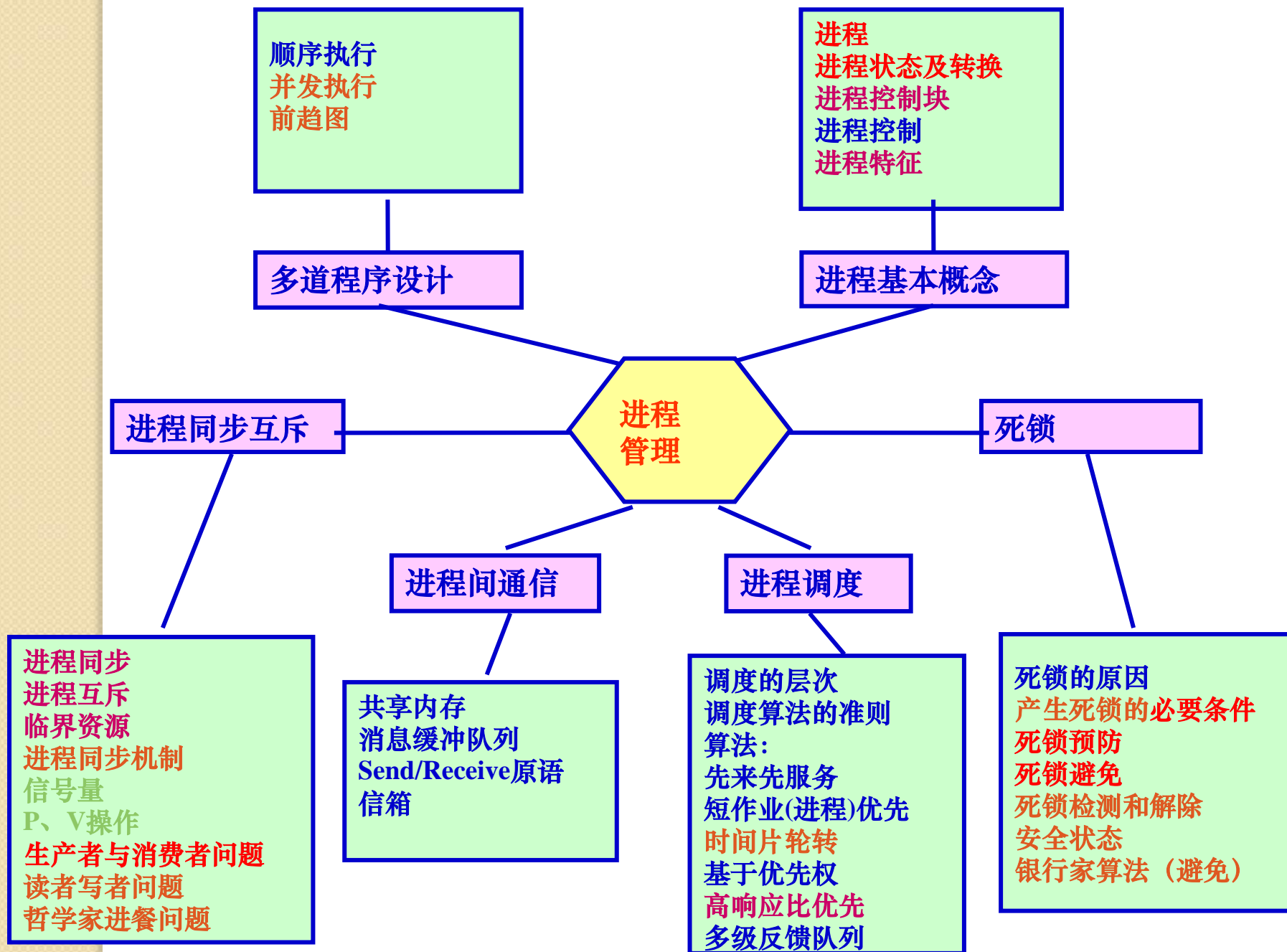


第一部分 操作系统基础



第一章 绪论（考概念）

- 什么是OS★
- 批处理、分时、实时系统（比较）
- OS的作用
- OS的四个基本特性★
- OS的五大功能★
- OS的基本类型（了解）



• 第二章 进程管理

- 1、**进程和线程**的概念、比较
- 2、**进程的基本状态及状态转换的原因**
- 3、**PCB的作用**
- 4、**进程控制的原语操作**
- 5、**进程互斥、临界区、进程同步的基本概念、同步准则（四条）**
- 6、**信号量机制（四种）**
- 7、**信号量应用实现（进程互斥、前趋关系）**
- ★ 8、**经典进程同步问题(三类);**
- 9、**进程间通信的原理和实现方法(四种)**



第二章 进程管理的典型问题

- 进程的三种基本状态及其转变原因。
- 进程互斥、临界资源
- 三种经典同步问题及其变型
 - 同步约束条件的分析，信号量的初值的设定
 - 单缓冲区的一个生产者一个消费者同步问题
 - 单缓冲区的一个生产者多个消费者同步问题
 - 多个生产者多个消费者多个缓冲区的同步问题

第三章 处理机调度的重点、难点

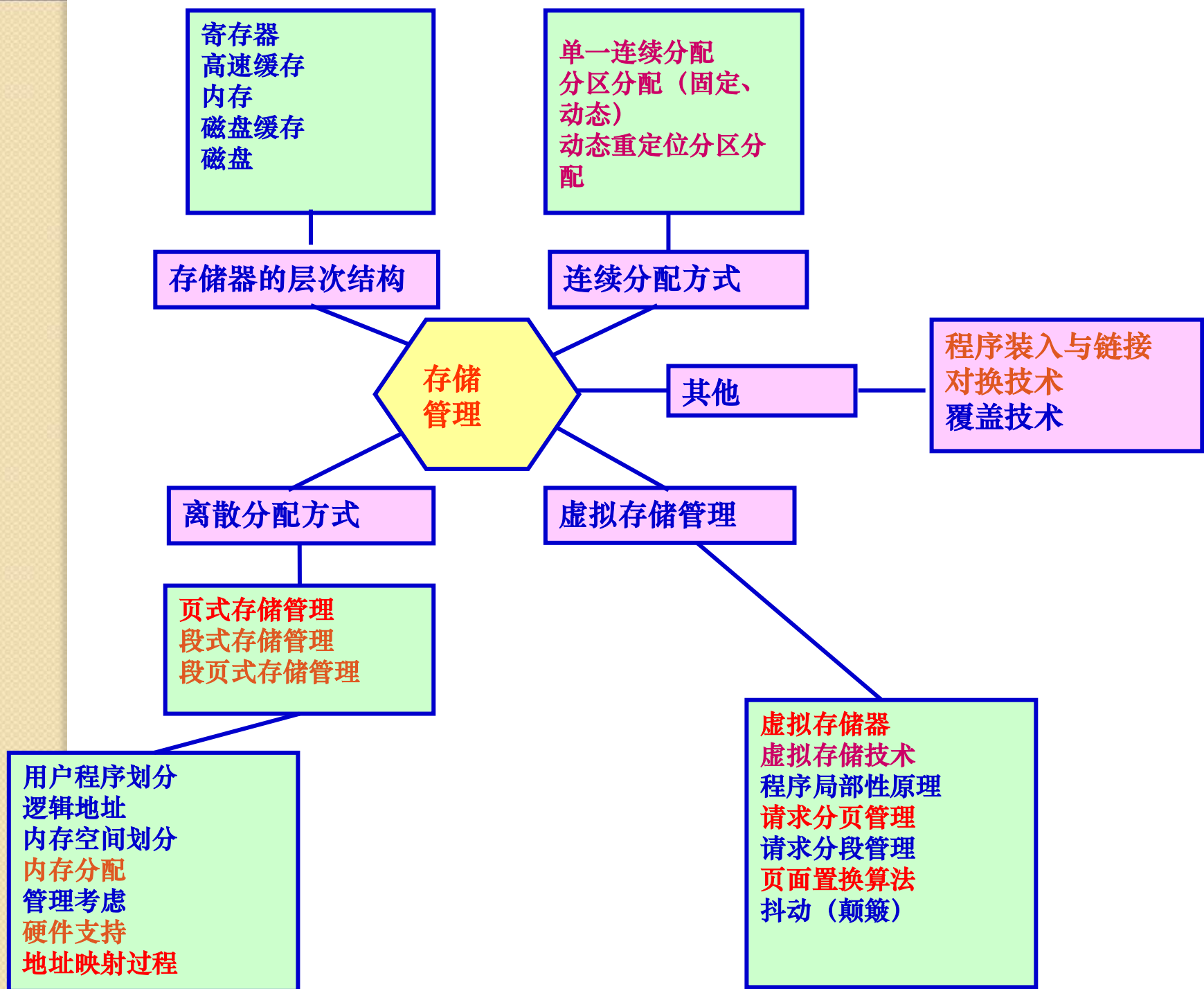
- 调度的层次（高级、中级、低级）
- 调度算法的准则（周转时间、带权周转时间、响应时间）
- 算法：（抢占/非抢占、调度时机）
 - 先来先服务
 - 短作业(进程)优先
 - 时间片轮转
 - 基于优先权
 - 高响应比优先
 - 多级反馈队列
- 实时调度（了解）

第三章 处理机调度

- 调度的层次 (高级、中级、低级)
- 调度算法的准则 (周转时间、带权周转时间、响应时间)
- 算法: (抢占/非抢占、调度时机)
 - 先来先服务
 - 短作业(进程)优先
 - 时间片轮转
 - 基于优先权
 - 高响应比优先
 - 多级反馈队列
- 实时调度 (了解)

第三章 处理机调度的典型问题

- 死锁的概念、原因
- 产生死锁的必要条件
- 死锁预防
- 死锁避免
- 死锁检测和解除
- 安全状态
- 银行家算法（避免）

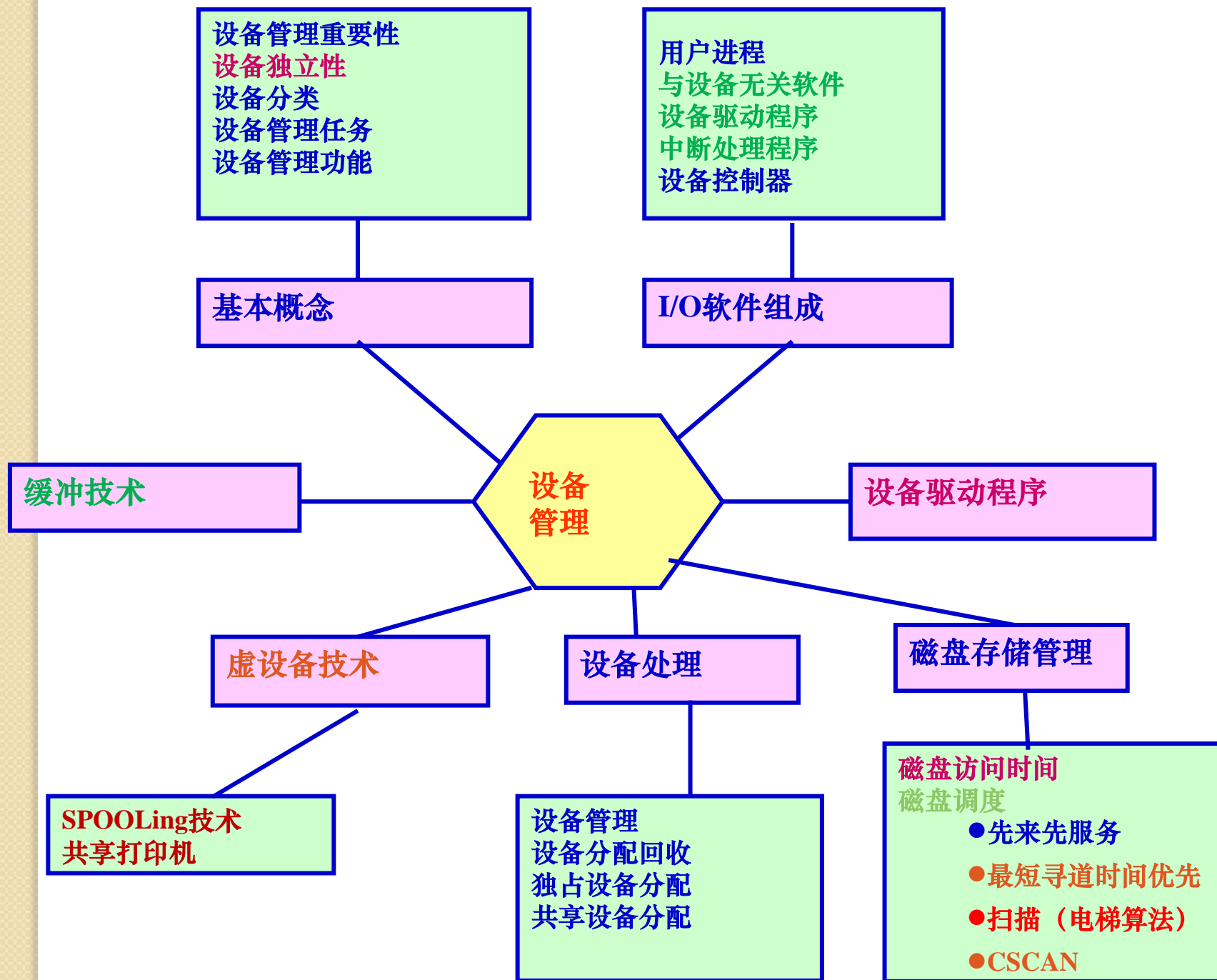


•第四章 存储管理的重点、难点

- **重定位的基本概念**：为什么要引入
- **如何提高内存利用率**：离散分配、对换机制、动态链接、虚拟存储器、存储器共享
- **动态分区分配方式**：分配、回收算法
- **基本分页存储管理方式**：为什么引入；地址变换机构和过程（含具有快表的情况）
- **基本分段存储管理方式**：为什么引入；地址变换机构和过程（含具有快表的情况）；信息的共享和保护
- **虚拟存储器的基本概念**：为什么要引入；特征；实现虚拟存储的关键技术
- **请求分页系统的基本原理**：页表机制；地址变换过程；页面置换算法
- **访问内存的有效时间**

第四章的典型问题

- **存储器管理的基本任务**
- **动态重定位的概念、实现方式，什么情况下需要重定位**
- **比较连续分配与离散分配**
- **基于空闲分区链的内存分配与回收算法的应用实例：首次适应法，循环首次适应法，最佳适应法、最坏适应法**
- **在某分页系统中，给定内存容量和物理块大小，计算物理块的数量；对给定的进程页表，将给定的逻辑地址，计算出其对应的物理地址并画出地址变换流程图。**
- **在某分段系统中对给定的进程段表，将给定的逻辑地址，计算出其对应的物理地址并画出地址变换流程图。**
- **请求分页系统过程的各种问题，并用流程图的方式表示地址变换过程**
- **对给定的问题，按各种页面置换算法，写页面调入过程，计算和分析缺页率，并对多种算法的性能作比较分析**



•第五章设备管理的重点、难点

- **I/O 控制方式：四种I/O 方式的基本原理；四种I/O 方式由低效到高效的演变**

缓冲管理

- **缓冲的概念，为什么引入缓冲**
- **单缓冲如何提高I/O 速度，它存在哪些不足，双缓冲、循环缓冲又如何提高CPU 与I/O 设备的并行性**
- **缓冲池是为了解决什么问题而引入，引入缓冲池后系统将如何处理I/O 设备和CPU 间的数据输送**
- **缓冲池的工作方式及Getbuf和Putbuf过程**

设备独立性（了解）

- **什么是设备独立性**
- **如何实现设备独立性**

设备驱动程序（了解）

第五章设备管理的重点、难点

虚拟设备和SPOOLing 技术（了解）

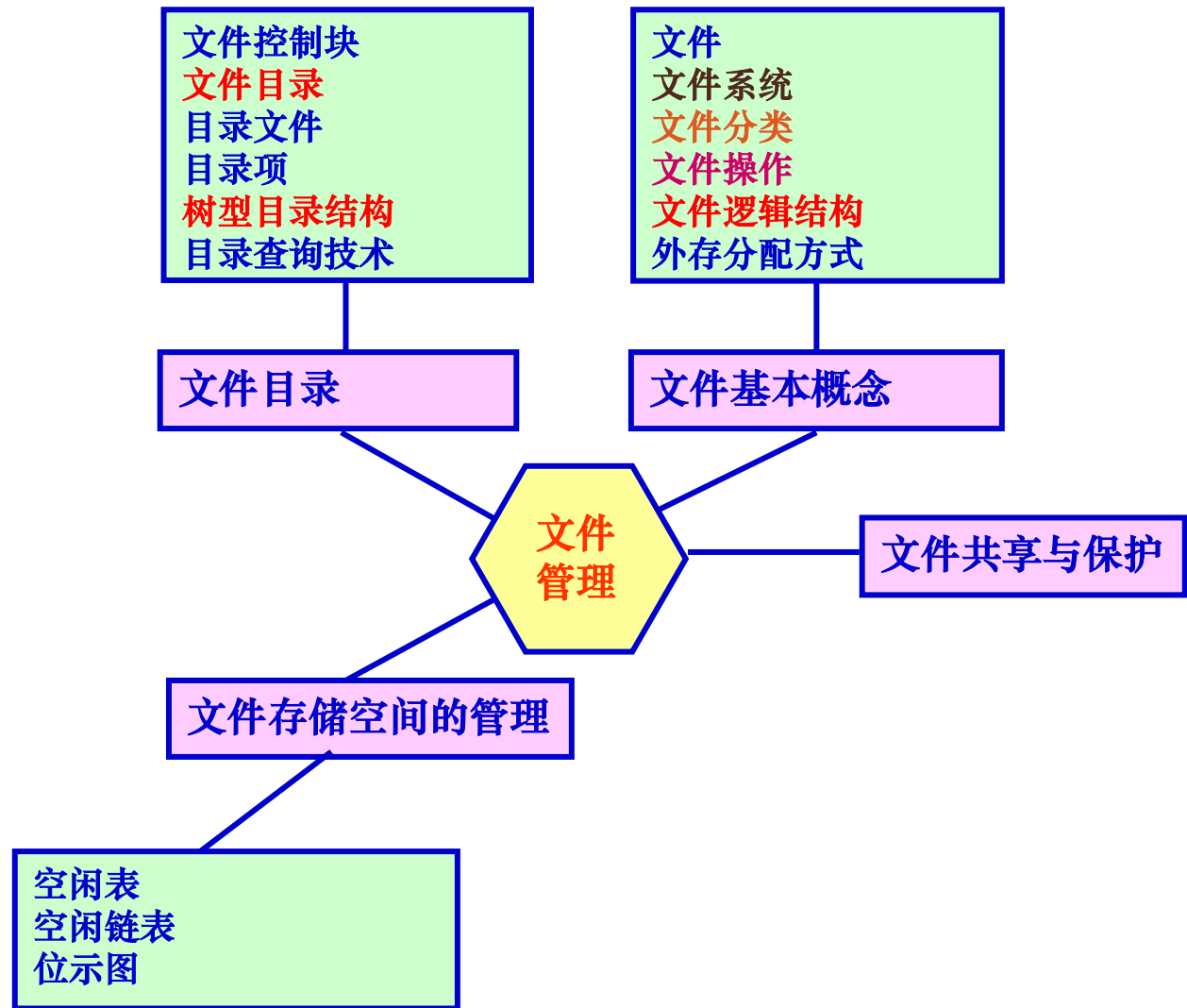
- 什么是虚拟设备
- 什么是假脱机（SPOOLing）技术，SPOOLing系统的组成
- 如何利用SPOOLing技术实现共享打印机

磁盘调度

- 磁盘调度的目标
- 磁盘访问时间的计算
- FCFS、SSTF、SCAN、CSCAN 等算法的应用及这些调度算法的演变过程，分别解决了哪些问题；各算法的性能比较（寻道长度的计算）

第五章设备管理的典型问题

- 各种I/O 控制方式的比较
- 为什么引入缓冲区
- 缓冲如何提高I/O 速度
- 为什么引入设备独立性，如何实现
- 什么是虚拟设备，实现虚拟设备的关键技术
- SPOOLing技术的组成，如何利用SPOOLing 技术实现共享打印机
- 对各种磁盘调度算法，计算访问次序和平均寻道时间，性能
- 磁盘访问时间的组成和计算



•第六章文件管理的重点、难点

文件的逻辑结构：顺序文件、索引文件和索引顺序文件

- 原理和特征

- 组织方式、访问方法及各种文件形式的比较

外存分配方式：连续分配、链接分配和索引分配原理、优缺点

- 显示链接FAT、增量式索引分配

目录管理：目录管理的要求

- 文件控制块（FCB）

- 索引结点

- 目录结构：单级、两级和多级

文件磁盘空间管理

- 空闲表法和空闲链法

- 位示图法：分配和回收的具体计算

第六章 文件管理的典型问题

- 画出链接分配方式的链接情况和FAT 的链接情况、FAT长度计算等。
- 增量式索引分配的的寻址方式、地址转换的计算和索引结点的地址映射图（书P278）
- 对给定的位示图和文件的分配和回收需求，具体写出分配过程和回收过程。
- 目录管理的要求；目前广泛采用的目录结构及其优点



第二部分 Linux编程

第七章 Unix/Linux入门的重点、难点

● 什么是SHELL

- Shell是系统的用户界面，提供了用户与内核进行交互操作的一种接口。它接收用户输入的命令并把它送入内核去执行。
- ①Shell是一个命令解释器，它解释由用户输入的命令并且把它们送到内核。
- ②Shell是一个解释型的程序设计语言。

● 简单命令

- cd,pwd,time,ls,
cat,vi, sleep等

- (1) date: 输出或设置系统日期和时间。
- (2) who: 查看系统中所有已登录用户的状况。
- (3) passwd: 修改用户密码。
- (4) logout、login: 登录 Shell 的登录和注销命令。
- (5) pwd: 输出工作路径。
- (6) more、less、head、tail: 显示或部分显示文件内容。
- (7) lp/lpstat/cancel、lpr/lpq/lprm: 输出文件。
- (8) chmod u+x: 更改文件权限。
- (9) rm -fr dir: 删除非空目录。
- (10) cp -R dir: 复制目录。
- (11) fg jobid: 可以将一个后台进程放到前台。
- (12) kill: 删除执行中的程序或工作。
- (13) ps: 列出当前运行的进程。

第八章 shell程序设计的重点、难点

- 输入/输出重定向：
例如：\$ (ls ; pwd ; ps) > run_log
 - `command > filename command >> filename`
 - `command < filename command << filename`
- 管道 `command1 | command2`
 - 例如：\$ ps -e | grep student2
- shell环境变量
 - HOME=/usr/computer/student6 用户主目录, 注册时的初始目录
 - PATH=/bin:/usr/bin:\$HOME/bin:./ 键盘命令的搜索路径
 - SHELL=/bin/sh 用户的初始shell的路径名称
 - TERM=vt100 当前所用的终端类型
- echo 命令的使用
- 系统变量
 - PS1=\$ shell的主提示符
 - IFS= 域分隔符, 通常为空白符, 用来分隔命令行各个域
- 系统变量只能引用不能修改!
- 单引号、双引号、反撇
 - 单引号禁止变量替换, 元字符替换.
 - 反撇号中的字符串作为命令
 - 花括号将变量名和后面的
- shell变量
 - \$0 当前shell程序的名字
 - \$1 ~ \$9 命令行上的第一到第九个参数
 - \$# 命令行上的参数个数, 不包含\$0
 - \$* 命令行上的所有参数
 - \$@ 分别用双引号引用命令行上的所有参数
 - \$\$ 当前进程的进程标识号(PID)
 - \$? 上一条命令的退出状态
 - \$! 最后一个后台进程的进程标识号

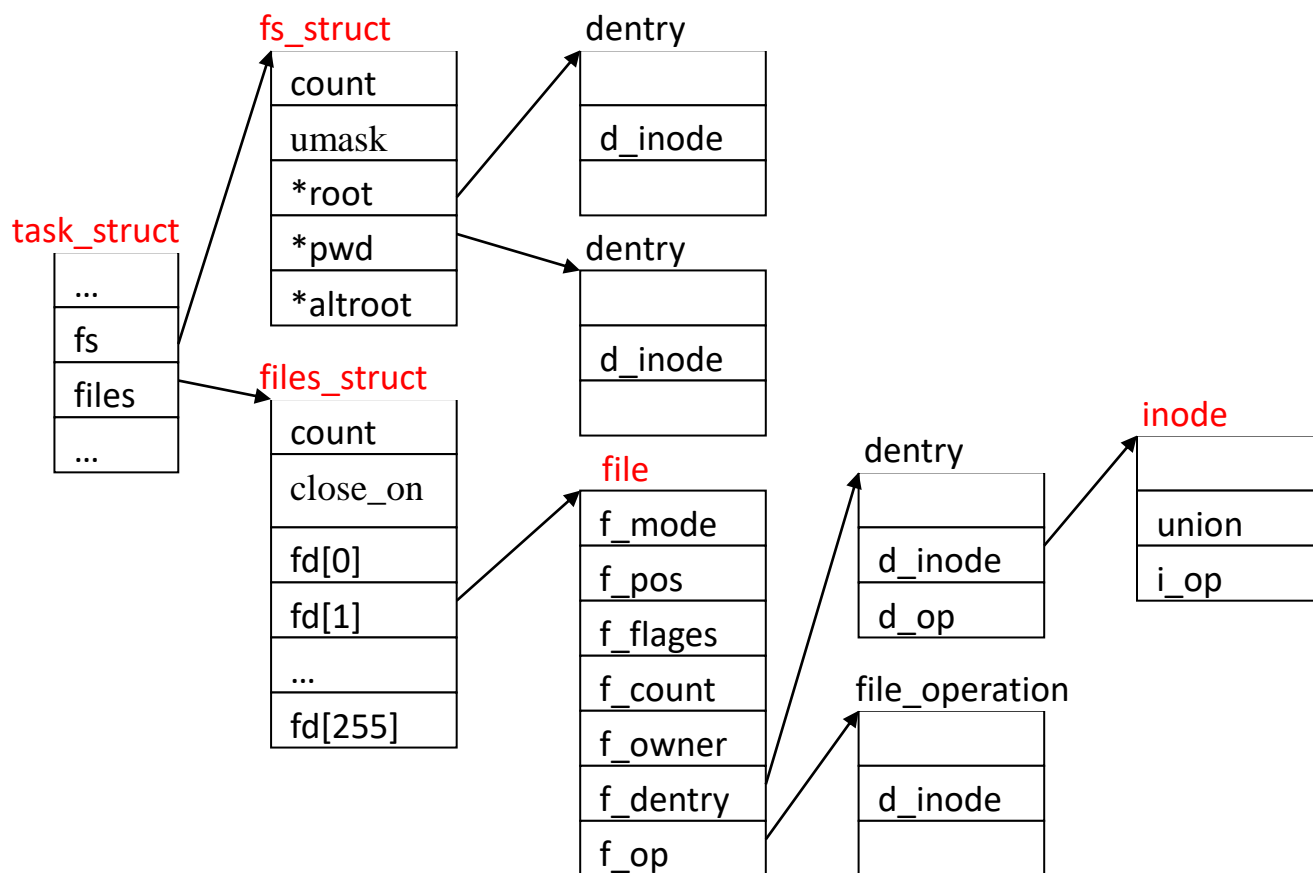
第八章 shell程序设计的典型问题

- shell编程: 功能性语句、结构性语句和说明性语句.
 - read, expr, test 等

```
#!/bin/sh
# Put line numbers on all lines of a file
if [ $# -ne 1 ]
then
    echo "Usage: $0 filename " >&2
    exit 1
fi
count=1          # Initialize count
cat $1 | while read line
# Input is coming from file on command line
do
    echo $count $line
    count=`expr $count + 1`
done > tmp$$      # Output is going to a temporary file
mv tmp$$ $1
```

第九章 文件操作与权限管理的重点、难点

- 目录文件内容：一系列目录项（dirent）的列表，每个目录项由两部分组成：所包含文件的文件名；文件名对应的索引节点（inode）号





文件在内核中的表现形式

文件在内核中的表现形式：文件描述符表，文件表，索引节点

文件描述符表

[0]
[1]
[2]
[3]
[4]

文件表

文件状态标志
文件当前位置
索引节点指针
引用数

...

文件状态标志
文件当前位置
索引节点指针
引用数

索引节点表

文件属性
引用数
数据块位置
当前文件长度

...

文件属性
引用数
数据块位置
当前文件长度



文件类型与文件访问权限

- 文件类型

标识	文件类型
-	普通文件
d	目录文件
c	字符设备文件
b	块设备文件
p	管道或FIFO
l	符号链接
s	套接字

- 文件访问权限

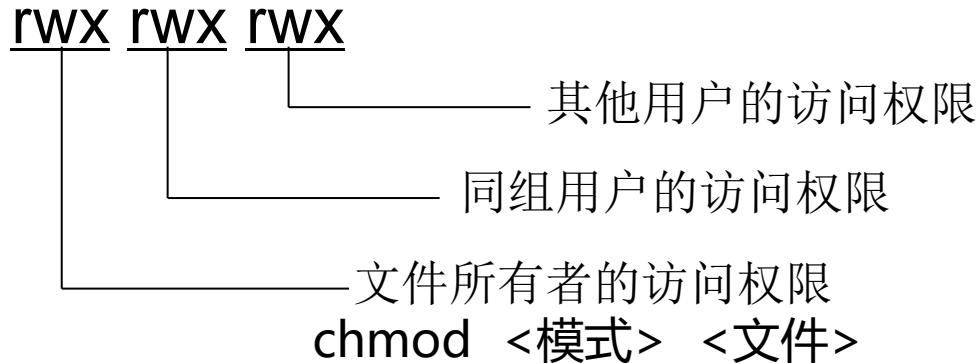
标识	文件访问权限
r	读权限
w	写权限
x	执行权限



基于用户的文件权限管理

```
root@ubuntu:/usr/bin# ls -l passwd
-rwsr-xr-x 1 root root 45420 Feb 16 2014 passwd
root@ubuntu:/usr/bin#
```

字母表示法



数字表示法

---	--X	-W-	-WX	r--	r-X	rw-	rwX
000	001	010	011	100	101	110	111

chmod <模式> <文件>



ls -l功能分析



文件属性

所有者
权限

其他用户
权限

文件
所有者

文件大小
(字节)

文件名

drwxr-xr-x	2	root	root	4096	2012-02-23	10:08	Documents
drwxr-xr-x	2	root	root	4096	2012-02-23	10:08	Downloads
drwxr-xr-x	3	root	root	4096	2014-01-01	03:18	etc
-rwxr-xr-x	1	root	root	7759	2014-01-01	02:33	listdirectory
-rw-r--r--	1	root	root	1985	2014-01-01	03:42	listdirectory.c



文件
类型

组用户
权限

文件硬链接
数或目录子
目录数

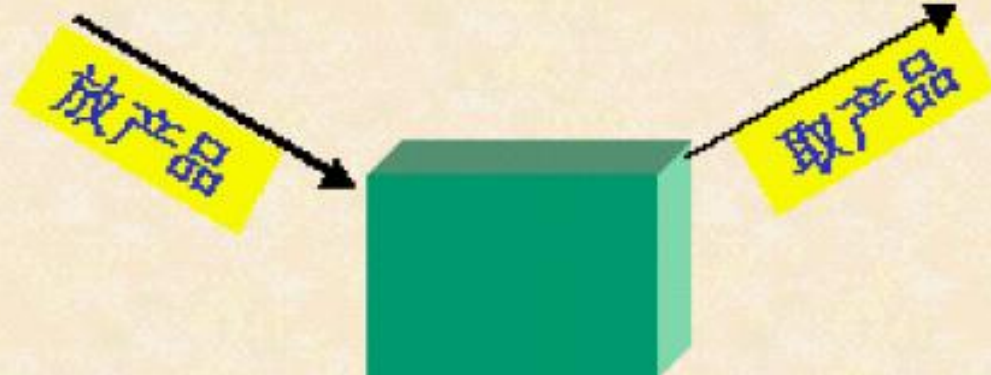
文件所有者
所在组

文件最后修改时间

经典的生产者—消费者问题

生产者

消费者



一次只可放一个产品



经典的生产者—消费者问题（续1）

同步问题：

P进程不能往“满”的缓冲区中放产品，设置信号量为 S_1

Q进程不能从“空”的缓冲区中取产品，设置信号量 S_2

经典的生产者—消费者问题（续2）

P:

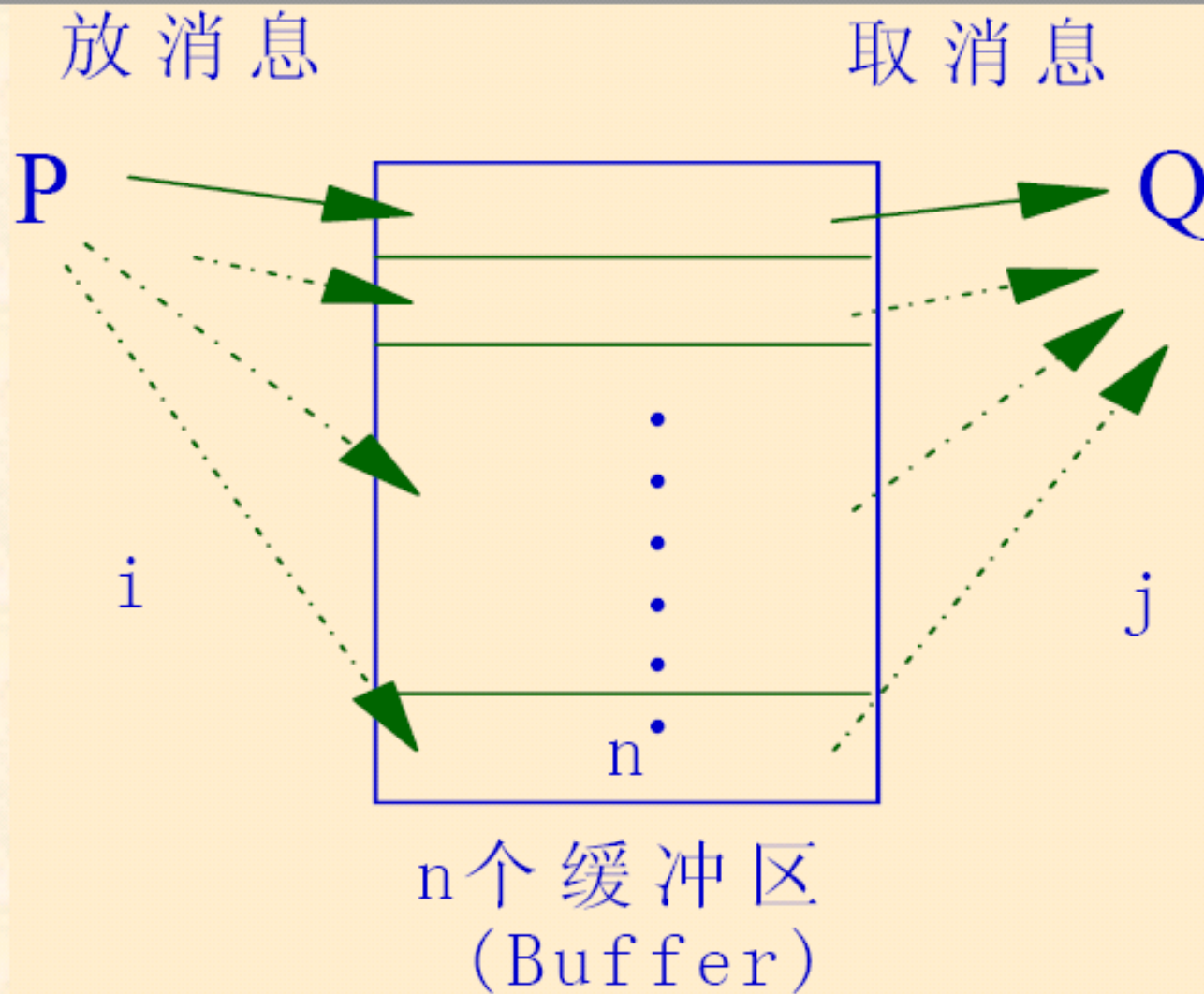
```
while (true) {  
    生产一个产品;  
    P(s1);  
    送产品到缓冲区;  
    V(s2);  
};
```

Q:

```
while (true) {  
    P(s2);  
    从缓冲区取产品;  
    V(s1);  
    消费产品;  
};
```

S1初值为1， S2初值为0

一个生产者一个消费者n个缓冲区



多个缓冲区的生产者和消费者

P:

```
i = 0;  
while (true) {  
    生产产品;  
    P(S1);  
    往Buffer [i]放产品;  
    V(S2);  
    i = (i+1) % n;  
};
```

Q:

```
j = 0;  
while (true) {  
    P(S2);  
    从Buffer[j]取产品;  
    V(S1);  
    消费产品;  
    j = (j+1) % n;  
};
```

有错误?



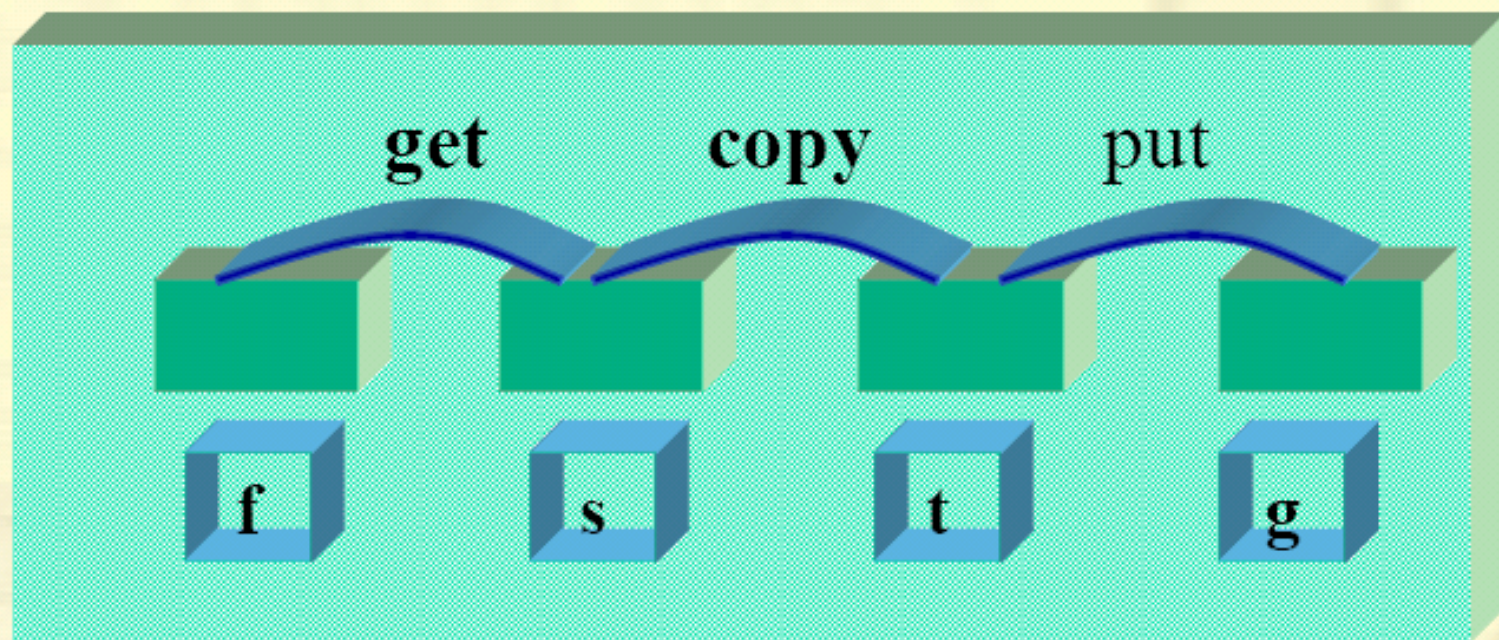
多个缓冲区的生产者和消费者（续）

【思考题】要不要对缓冲区（临界资源）进行互斥操作？

由于只有一个生产者和一个消费者，不会发生几个生产者和消费者同时存取同一缓冲单元的情况，故无须设置互斥信号量。

【思考题】

1.用P.V操作解决下图之同步问题:



假定系统有3个并发进程get、copy和put共享缓冲器B1和B2。进程get负责从输入设备上读信息，每读出一条记录后放到B1中。进程copy从缓冲器B1中取出一条记录拷贝后存入B2。进程put取出B2中的记录打印输出。B1和B2每次只能存放一条记录。要求3个进程协调完成任务，使打印出来的与读入的记录个数、次序完全一样。请用记录型信号量写出并发程序。

解：

设置4个信号量，其中empty1对应空闲的缓冲区1，其初值为1；full1对应缓冲区1中的记录，其初值为0；empty2对应空闲的缓冲区2，其初值为1；full2对应缓冲区2中的记录，其初值为0。相应进程描述为：

```
get( ){  
    while(ture){  
        从输入设备读入一条记录;  
        P(empty1);  
        将记录存入缓冲区1;  
        V(full1);  
    }  
}
```

```
copy( ){  
    while(true){  
        P(full1);  
        从缓冲区1中取出一条记录;  
        V(empty1);  
        P(empty2);  
        将取出的记录存入缓冲区2 ;  
        V(full2);  
    }  
}
```

```
put( ){  
    while(1){  
        P(full2);  
        从缓冲区2中取出一条记录;  
        V(empty2);  
        将取出的记录打印出来;  
    }  
}
```

- Main(){
 parbegin(get,copy,put);
}

例

一台计算机有10台磁带机被n个进程竞争，每个进程最多需要3台磁带机，那么n最多为_____时，系统没有死锁的危险？

解：n最大为4。

补充：关于死锁的公式：

当一个系统有N个并发进程，每个进程都需要M个同类资源，那么最少需要多少资源才能避免死锁的出现？

$(M-1) * N + 1$ 注：每个进程分配M-1个资源，然后再加上一个资源，该资源无论给哪个进程都可以保证当前系统不会出现死锁。

例

在银行家算法中，若出现下述的资源分配情况：

Process	Max	Allocation	Available
P0	0 0 4 4	0 0 3 2	1 6 2 2
P1	2 7 5 0	1 0 0 0	
P2	3 6 10 10	1 3 5 4	
P3	0 9 8 4	0 3 3 2	
P4	0 6 6 10	0 0 1 4	

试问：

- 1) 该状态是否安全？
- 2) 若进程P2提出请求Request (1, 2, 2, 2) 后，系统能否将资源分配给它？
- 3) 如果系统立即满足P2的上述请求，系统是否立即进入死锁状态？

解：

- 1) 利用安全性算法对上面的状态进行分析（如下表所示），找到了一个安全序列{P0, P3, P4, P1, P2}或{P0, P3, P1, P4, P2}，故系统是安全的。

资源情况 进程	Work				Need				Allocation				Work+Allocation				Finish
	A	B	C	D	A	B	C	D	A	B	C	D	A	B	C	D	
P0	1	6	2	2	0	0	1	2	0	0	3	2	1	6	5	4	True
P3	1	6	5	4	0	6	5	2	0	3	3	2	1	9	8	6	True
P4	1	9	8	6	0	6	5	6	0	0	1	4	1	9	9	10	True
P1	1	9	9	10	1	7	5	0	1	0	0	0	2	9	9	10	True
P2	2	9	9	10	2	3	5	6	1	3	5	4	3	12	14	14	True

2) P2发出请求向量Request (1, 2, 2, 2) 后, 系统按照银行家算法进行检查:

$\text{Request}_2 (1, 2, 2, 2) \leq \text{Need}_2 (2, 3, 5, 6) ;$

$\text{Request}_2 (1, 2, 2, 2) \leq \text{Available} (1, 6, 2, 2) ;$

系统先假定可为P2分配资源, 并修改Available, Allocation₂和Need₂向量:

Availabe = (0, 4, 0, 0) Allocation₂ = (2, 5, 7, 6)

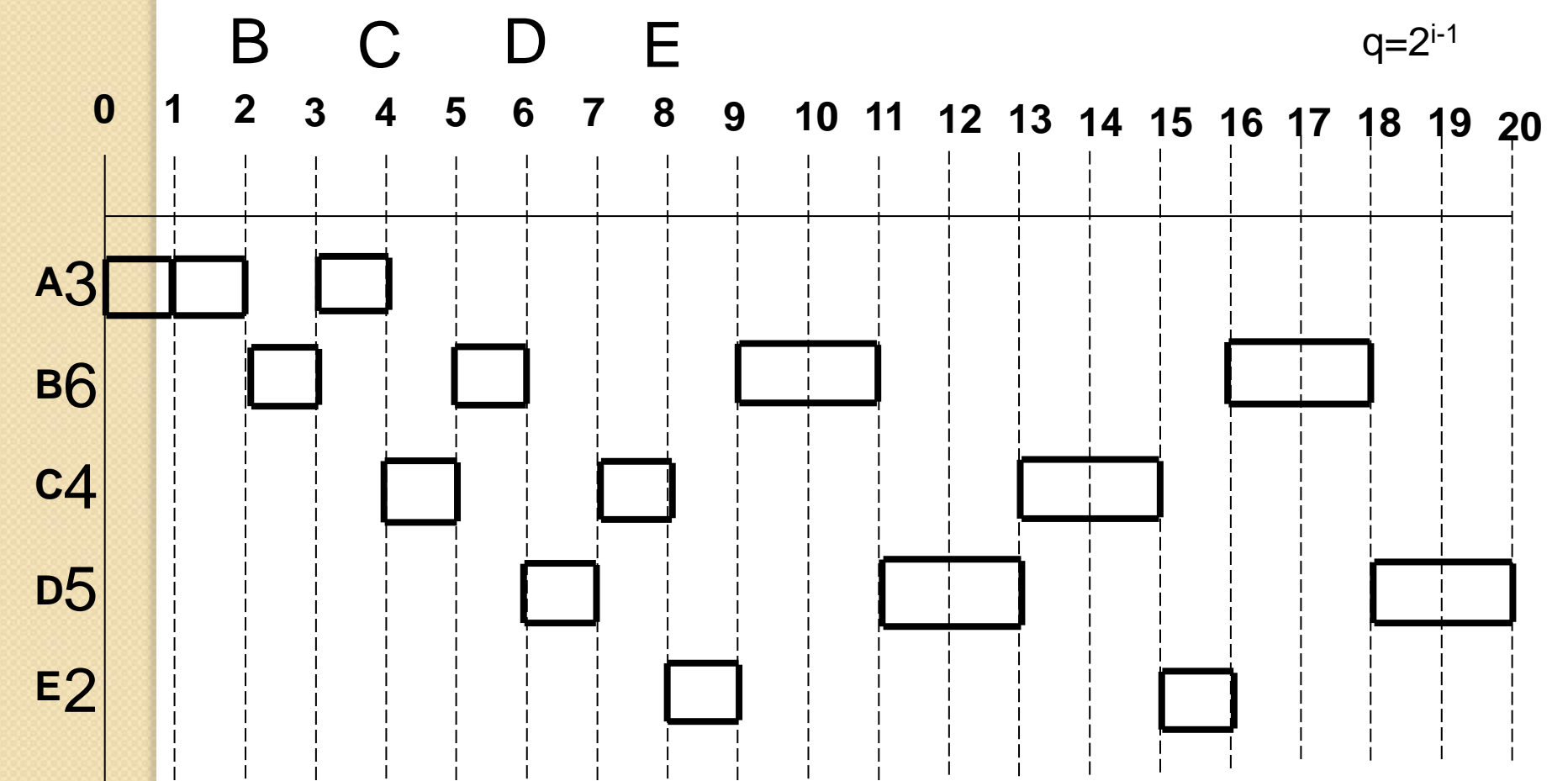
Need₂ = (1, 1, 3, 4)

进行安全性检查: 此时对所有进程, 条件 $\text{Need}_i \leq \text{Available}$ (0, 4, 0, 0) 都不成立, 即Available不能满足任何进程的请求, 故系统进入不安全状态。因此, 当进程P2提出请求Request (1, 2, 2, 2) 后, 系统不能将资源分配给它。

3) 系统立即满足进程P2的请求 (1, 2, 2, 2) 后, 并没有马上进入死锁状态。因为, 此时上述进程并没有申请新的资源, 并未因得不到资源而进入阻塞状态。只有当上述进程提出新的请求, 并导致所有没执行完的多个进程因得不到资源而阻塞时, 系统才进入死锁状态。

例：多级反馈队列调度算法

- 注意：统一为抢占式
- 高优先级队列中有进程进入时，会抢占低优先级队列中进程的CPU。被抢占的进程不降级，回到原级队列中，下次仍然执行该级队列的时间片。



t	0	1	2	3	4	5	6	7	8	9	11	13	15	16	18	20
一级 q=1	A		B		C		D		E							
二级 q=2		A	A	BA	B	CB	BC	DBC	CDB	ECDB	ECD	EC	E			
三级 q=4											B	DB	DB	DB	D	

例2：已知某分页系统，用户空间有64个页面，主存容量为32KB，页面大小为1KB，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。

(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？

逻辑地址位数	$64 * 1K = 2^{16}$	16位
物理地址位数	$32K = 2^{15}$	15位
每个块大小：	1KB	

答：

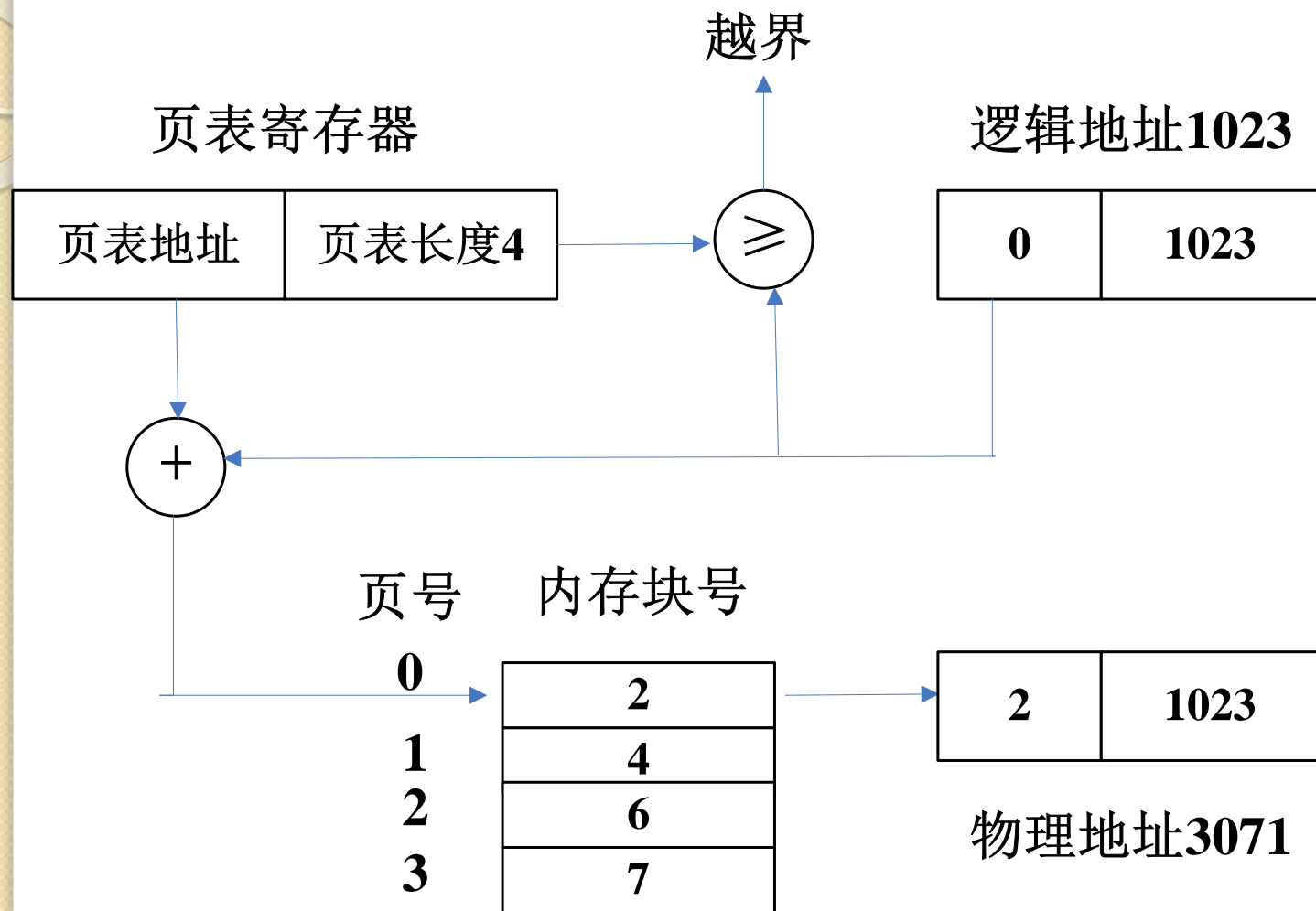
①逻辑地址1023：1023/1K，得页号为0，页内地址为1023，查页表找到对应的物理块号为2，故物理地址为 $2 \times 1K + 1023 = 3071$

②逻辑地址2500：2500/1K，得页号为2，页内地址为452，查页表找到对应的物理块号为6，故物理地址为 $6 \times 1K + 452 = 6596$

③逻辑地址3500：3500/1K，得页号为3，页内地址为428，查页表找到对应的物理块号为7，故物理地址为 $7 \times 1K + 428 = 7596$

④逻辑地址4500：4500/1K，得页号为4，页内地址为404，因页号大于页表长度，故产生越界中断。

(2) 地址变换过程图



例题

对访问串1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5, 指出在驻留集大小分别为3、4时, 使用FIFO替换算法的缺页次数和缺页率。结果说明了什么?

先进先出 (FIFO) 页面置换算法 (续)

- Reference string: 1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5
- 3 frames (3 pages can be in memory at a time per process)

1	1	4	5	
2	2	1	3	9 page faults
3	3	2	4	

- 4 frames

1	1	5	4	
2	2	1	5	10 page faults
3	3	2		
4	4	3		

FIFO Replacement

- more frames \Rightarrow less page faults ?

例

一台计算机有四个页框，装入时间、上次引用时间、它们的R（读）与M（修改）位如下表所示（时间单位：嘀嗒），请问NRU、FIFO、LRU算法将替换哪一页？

页	装入时间	上次引用时间	R	M
0	126	279	0	0
1	230	260	1	0
2	120	272	1	1
3	160	280	1	1

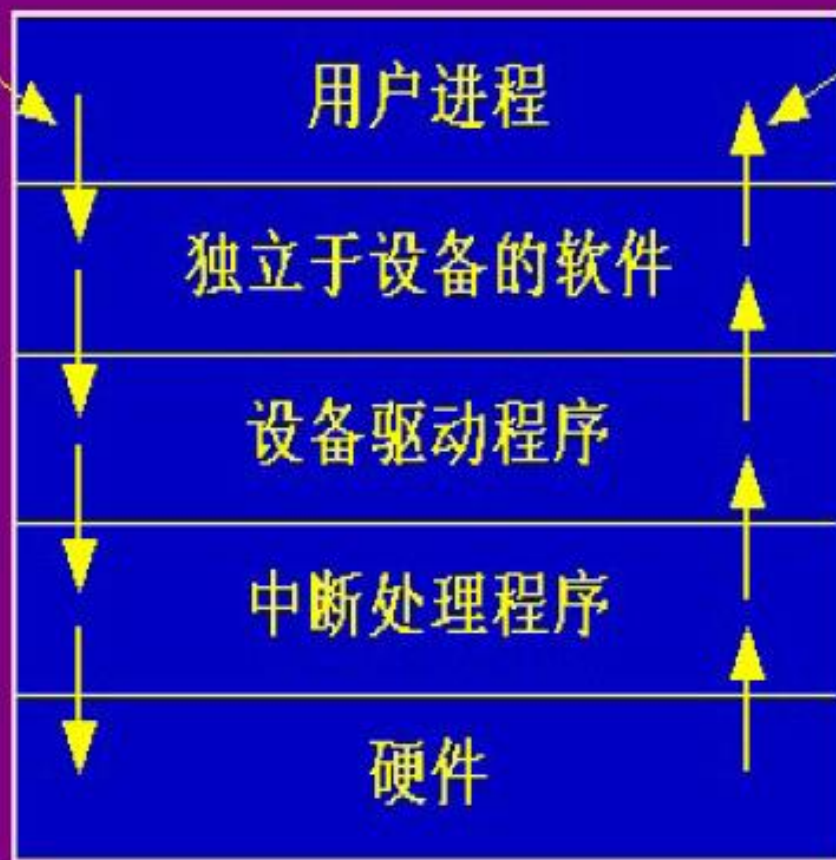
解答

- FIFO算法在需要淘汰某一页时，淘汰最先进入内存的页。在题述条件下，第2页是最先进入内存的页。故FIFO算法将淘汰第2页。
- LRU算法在需要淘汰某一页时，淘汰最近最久未使用的页面。在题述条件下，第1页是最近最久未使用的页面。故LRU算法将淘汰第1页。
- NRU算法在需要淘汰某一页时，从那些最近一个时期内未被访问的页中任选一页淘汰。在题述条件下，只有第0页是最近一个时期内未被访问的页。故NRU算法将淘汰第0页。

1. I / O软件的层次

I/O请求

I/O完成后的回答



系统的分量及各层的主要功能

磁盘调度

目标：减少寻道时间

1、FCFS (First Come First Served) 先来先服务

- 特点：公平、简单，寻道时间长，相当于随机访问模式。
- 仅适用于请求磁盘I/O的进程数目较少的场合。

2、SSTF (Shortest Seek Time First) 最短寻道时间优先

- SSTF比FCFS有更好的寻道性能
- 饥饿现象
- 不能保证平均寻道时间最短？

(从 100 号磁道开始)	
被访问的下一个磁道号	移动距离 (磁道数)
55	45
58	3
39	19
18	21
90	72
160	70
150	10
38	112
184	146
平均寻道长度: 55.3	

图 5-25 FCFS调度算法

(从 100 号磁道开始)	
被访问的下一个磁道号	移动距离 (磁道数)
90	10
58	32
55	3
39	16
38	1
18	20
150	132
160	10
184	24
平均寻道长度: 27.5	

图 5-26 SSTF调度算法

3、SCAN 扫描算法（也称为电梯算法）。

- SSTF存在进程“饥饿现象”

- SCAN算法：

- 在移动方向固定的情况下采用了SSTF，以避免饥饿现象
- 存在请求进程等待延迟现象

4、循环扫描CSCAN

- 磁头单向移动

- 一个方向读完，不是象SCAN那样回头，而是循环扫描。

(从 100# 磁道开始, 向磁道号增加方向访问)	
被访问的下一个磁道号	移动距离 (磁道数)
150	50
160	10
184	24
90	94
58	32
55	3
39	16
38	1
18	20
平均寻道长度: 27.8	

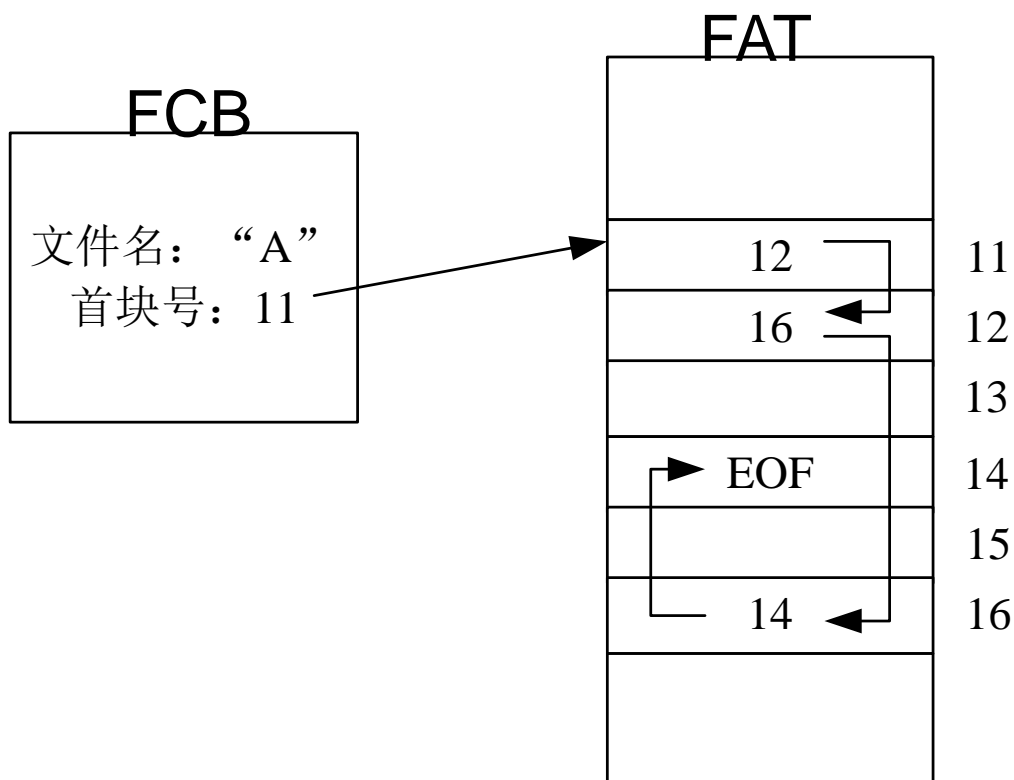
(从 100# 磁道开始, 向磁道号增加方向访问)	
被访问的下一个磁道号	移动距离 (磁道数)
150	50
160	10
184	24
18	166
38	20
39	1
55	16
58	3
90	32
平均寻道长度: 35.8	

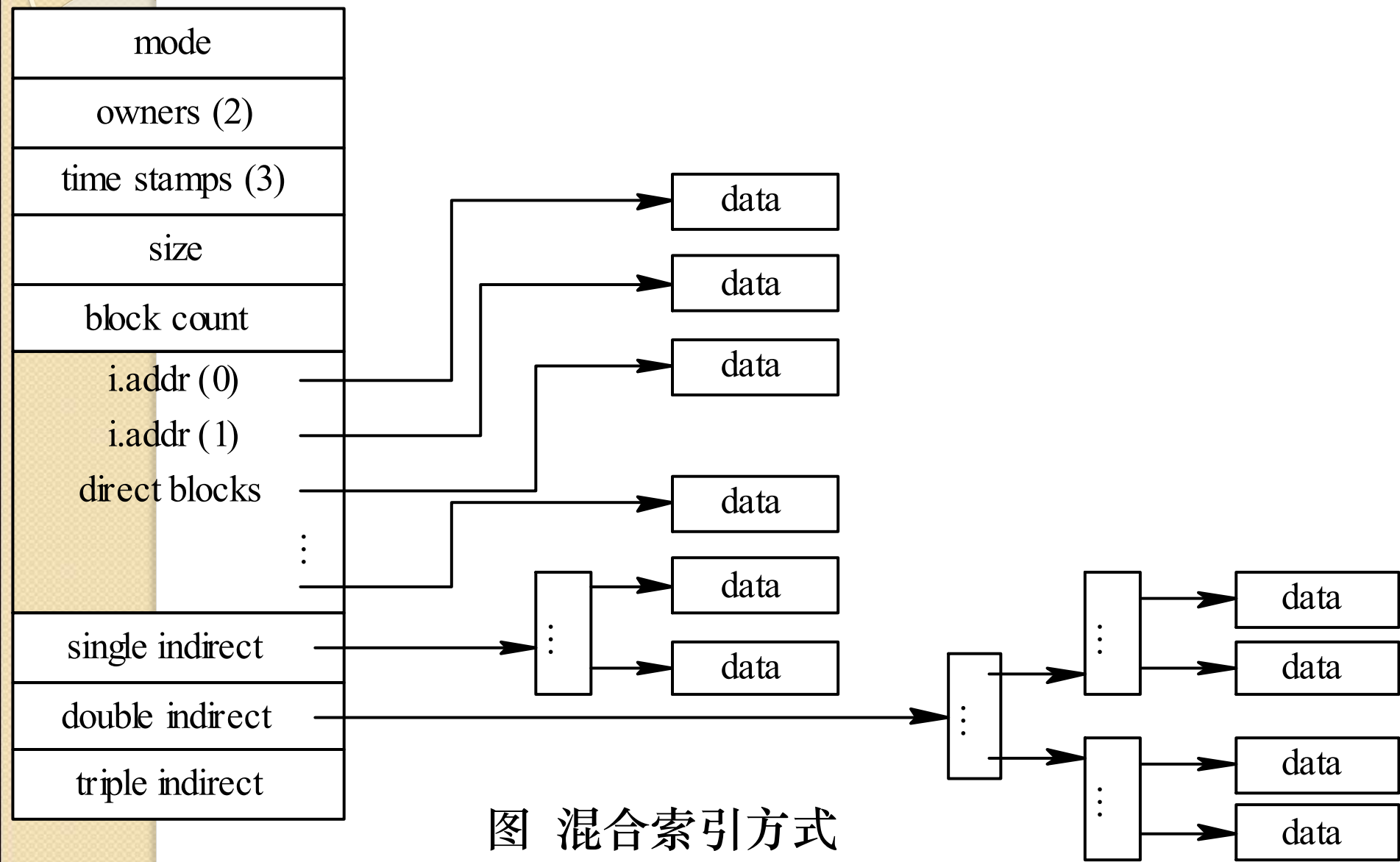
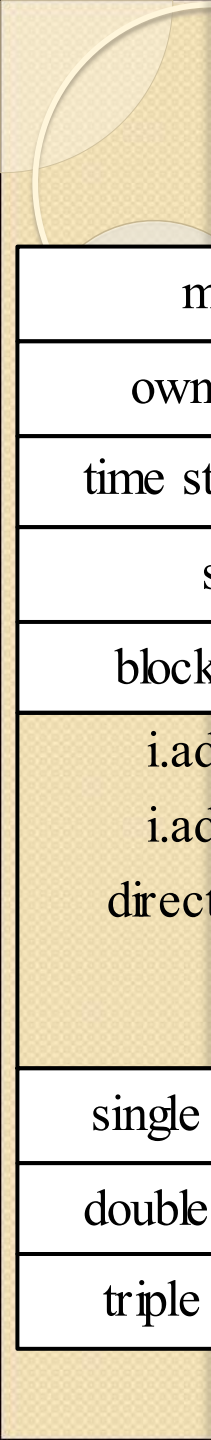
图 5-27 SCAN 调度算法示例 图 5-28 CSCAN 调度算法示例

例

假定盘块的大小为1KB,硬盘的大小为500MB,采用显示链接分配方式时,其FAT需占用多少存储空间 (FAT表项占2.5个字节)?如果文件A占用硬盘的11, 12, 16, 14四个盘块,试画出文件A中各盘块在FAT表中的链接情况.

解：此时硬盘共有 $500\text{M}/1\text{K}=500\text{K}$ 个盘块，
FAT表项共有 $500\text{K} * 2.5\text{B}=1250\text{KB}$





例

- 存放在某个磁盘上的文件系统，对于采用混合索引分配方式，其FCB中共有13项地址项，第0~9个地址项为直接地址，第10个地址项为一次间接地址，第11个地址项为二次间接地址，第12个地址项为三次间接地址。如果每个盘块的大小为512字节，盘块号需要3个字节来描述，则每个盘块最多存放170个盘块地址：
 - (1) 该文件系统允许的最大长度是多少？
 - (2) 将文件的字节偏移量5000、15000、150000转换为物理块号和块内偏移量。
 - (3) 假设某文件的索引结点已在内存中，但其他信息均在外存，为了访问该文件中某个位置的内容，最多需要几次访问磁盘？


(1) 文件的最大长度为:

$$10 + 170 + 170^2 + 170^3 = 4942080 \text{ 块} = 2471040 \text{ KB}$$

(2) 5000/512得商9, 余数为392。即逻辑块号为9, 块内偏移为392。故可直接从该文件的FCB的第9个地址处得到物理盘块号, 块内偏移为392。

15000/512得商为29, 余数为152。即逻辑块号为29, 块内偏移为152。由于 $10 \leq 29 < 10 + 170$, 而 $29 - 10 = 19$, 故可从FCB的第10个地址项, 即一次间址项中得到一次间址块; 并从一次间址块的19项中获得对应的物理盘块号, 块内偏移为152。

150000/512得商为292, 余数为496。即逻辑块号为292, 块内偏移为496。由于 $10 + 170 \leq 292$, 故可从FCB的第11个地址项, 即二次间址项中获得第1个一次间址块; 并从该一次间址块的112项中获得对应的物理盘块号, 块内偏移为496。



(3) 由于文件的索引结点已在内存，为了访问文件中的某个位置的内容，最少需要1次访问磁盘（即通过直接地址直接读文件盘块），最多需要4次访问磁盘（第一次是读三次间址块，第二次读二次间址块，第三次读一次间址块，第四次是读文件盘块）

有一个计算机系统利用下图所示的位示图（行号、列号都从0开始编号）来管理空闲盘块。如果盘块从1开始编号，每个盘块的大小为1KB。

(1) 现要从文件分配两盘块，试具体说明分配过程。

(2) 若要释放磁盘的第300块，应如何处理？

[illegible]

解

(1) 分配过程

- 线形检索得： $i1=2, j1=2; i2=3, j2=6$ 。
- 计算空闲盘块号：
 - $b1=i1 \times 16+j1+1=2 \times 16+2+1=35$
 - $b2=i2 \times 16+j2+1=3 \times 16+6+1=55$
- 修改位示图：
 - 令 $map[2, 2]=map[3, 6]=1$ ，并将对应块35， 55分配出去。

解

(2) 释放过程

- 计算出第300块所对应的二进制行号i和j
 - $i = (300-1) / 16 = 18$
 - $j = (300-1) \% 16 = 11$
- 修改位示图：
 - 令 $\text{map}[18, 11] = 0$ 。