2015年(408)计算机考研之操作系统

主讲：杨航空

考查目标

- 掌握操作系统的基本概念、基本原理和基本功能，理解操作系统的整体运行过程。
- 掌握操作系统进程、内存、文件和I/O管理的策略、算法、机制以及相互关系。
- 能够运用所学的操作系统原理、方法与技术分析问题和解决问题，并能利用C语言描述相关算法。

第一章 计算机系统概述

操作系统概述这一章出现大题的可能性微乎其微。选择题中常出现的点主要是这些：操作系统的定义，引入单道批处理系统、多道批处理、分时系统、实时系统的原因，这些不同阶段的操作系统有何特征，相互之间的差别在什么地方；操作系统的基本特征和功能；操作系统的运行环境。

难点精讲之多道批处理系统

多道批处理操作系统允许多个程序同时装入主存储器，使一个中央处理器轮流地执行多个作业，各个作业同时使用各自的外围设备，提高了计算机系统的资源使用效率。

- **多道批处理系统具有以下特征：**

  多道性、无序性和调度性。

- **多道批处理系统主要优点：**

  资源利用率高、系统吞吐量大。

- **多道批处理系统主要缺点：**

  平均周转时间长、无交互能力。

- **多道批处理系统需要解决的主要问题：**

  多道批处理系统是一种有效又十分复杂的系统，为使系统中的多道程序之间能协调地运行，必须解决如下几个问题：处理机管理问题、内存管理问题、设备管理问题、文件管理问题和作业问题。

难点精讲之多道批处理系统

分时系统

分时操作系统是多个用户通过终端机器同时使用一台主机，这些终端机器连接在主机上，用户可以同时与主机进行交互操作而互不干扰。

分时操作系统的特征：

- 交互性：用户可通过终端与系统进行广泛的人机对话。其广泛性表现在：用户可以请求系统提供多方面的服务，如文件编辑、数据处理和资源共享等。
- 及时性：用户的请求能在很短的时间内获得响应。此时间间隔是以人们所能接受的等待时间来确定的。
- 独立性：每个用户各占一个终端，彼此独立操作，互不干扰。因此，用户所感觉到的，就像是他一人独占主机。
- 多路性：允许在一台主机上同时连接多台联机终端，系统按分时原则为每个用户服务。

难点精讲之多道批处理系统

实时系统

实时系统是指系统能及时(或即时)响应外部事件的请求，在规定的时间内完成对该事件的处理，并控制所有实时任务协调一致地运行。在实时操作系统的控制下，计算机系统接收到外部信号后及时进行处理，并且要在严格的时限内处理完接收的事件。

难点精讲之实时与分时系统比较

- 多路性。实时信息处理系统也按分时原则为多个终端用户服务。实时控制系统的多路性则主要表现在系统周期性地对多路现场信息进行采集，以及对多个对象或多个执行机构进行控制。而分时系统中的多路性则与用户情况有关，时多时少。
- 独立性。实时信息处理系统中的多个终端用户在向实时系统提出服务请求时，是彼此独立地操作，互不干扰；而实时控制系统中，对信息的采集和对对象的控制也都是彼此互不干扰。
- 及时性。实时信息处理系统对实时性的要求与分时系统类似，都是以人所能接受的等待时间来确定的；而实时控制系统的及时性，则是以控制对象所要求的开始截止时间或完成截止时间来确定的，一般为秒级到毫秒级。
- 交互性。实时信息处理系统虽然也具有交互性，但这里人与系统的交互仅限于访问系统中某些特定的专用服务程序。不像分时系统那样能向终端用户提供数据处理和资源共享等服务。
- 可靠性。分时系统虽然也要求系统可靠，但相比之下，实时系统则要求系统具有高度的可靠性。因为任何差错都可能带来巨大的经济损失，甚至是无法预料的灾难性后果，所以在实时系统中，往往都采取了多级容错措施来保障系统的安全性及数据的安全性。

难点精讲之多道批处理系统

分布式计算机系统

- 分布式计算机系统是由多台计算机组成并满足下列条件的系统：
- 系统中任意两台计算机通过通信方式交换信息；
- 系统中的每台计算机都具有同等的地位，即没有主机也没有从机；
- 每台计算机上的资源为所有用户共享；
- 系统中的任意若干台计算机都可以构成一个子系统，并且还能重构；
- 任何工作都可以分布在几台计算机上，由它们并行工作协同完成；
- 用于管理分布式计算机系统的操作系统称为分布式操作系统。
- 具有分布性和并行性。

第二章 进程管理

进程管理是重点和难点之所在。考点既可以出现在选择题中，又可以出在综合应用题中。按照大纲考点的顺序，诸如进程的概念、基本特征、组成结构，进程与程序的区别与联系，进程的状态及其相互转换的条件及过程，进程间的通信方式，线程的定义以及和进程的区别与联系，调度的基本概念、时机、切换过程和各种调度算法，进程同步相关的概念，实现同步与互斥的机制，信号量和PV操作，管程的基本组成结构和运行过程，死锁的基本概念，死锁产生的四个必要条件，预防、避免、检测和解除死锁的原理与方法，这些点都可以出现在选择题中进行考查。对于综合应用题，重点应该放在PV操作，调度算法和银行家算法。

第二章 进程管理

其中，用PV操作实现经典同步问题及其变形是整个操作系统考试的最难点，也是最大的热点。要注意收集往年各校考过的PV操作应用题，把常见的经典题型做会做熟，力求看到题目就能想到相关的解题套路。调度算法的难点在于计算不同调度算法下调度的效率，建议使用时间轴的方法解决相关的调度时间计算问题。银行家算法是系统做资源分配的时候防止发生死锁的一种方法，该算法的难点在于搞清楚各种不同表格的含义，能够看懂并且会做出相关的表格，由表格推出结果。

难点精讲之进程

进程是支持程序执行的机制。进程可以理解为程序对数据或请求的处理过程。具体来说，进程由一些方面组成。

- 至少一个可执行程序，包括代码和初始数据，一般在进程创建时说明。注意可执行程序可以被多进程共享，换句话说多个进程可能允许同一个可执行程序。
- 一个独立的进程空间，在进程创建时由操作系统分配。
- 系统资源，指在进程创建时及执行过程中，由操作系统分配给进程的系统资源，包括I/O设备、文件等。
- 一个执行栈区，包含运行现场信息，如子程序调用时所压的栈帧，系统调用时所压的栈帧等。这是经常运行及进程调度进行处理机切换时所要涉及的数据结构。

难点精讲之进程状态变化

空→创建状态：

创建状态→就绪状态：

就绪状态→运行状态：

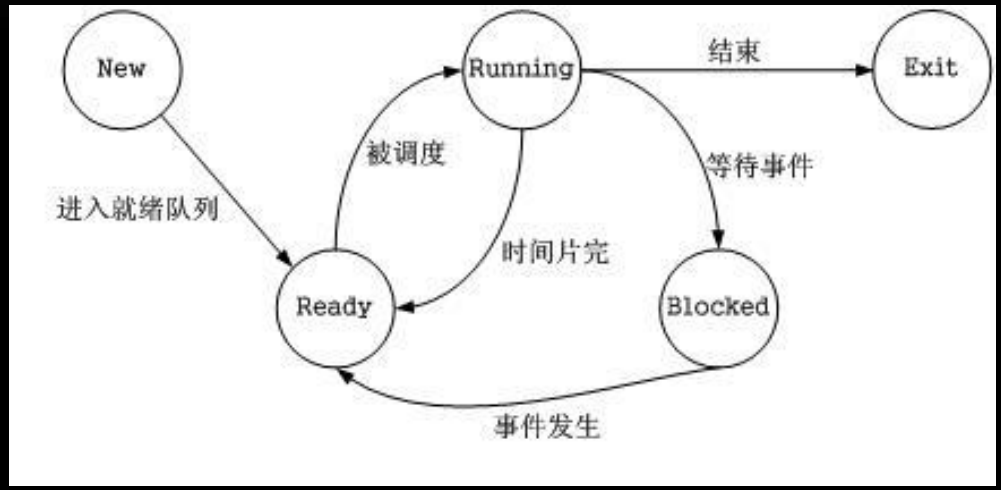
运行状态→结束状态：

运行状态→就绪状态：

运行状态→等待状态：

等待状态→就绪状态：

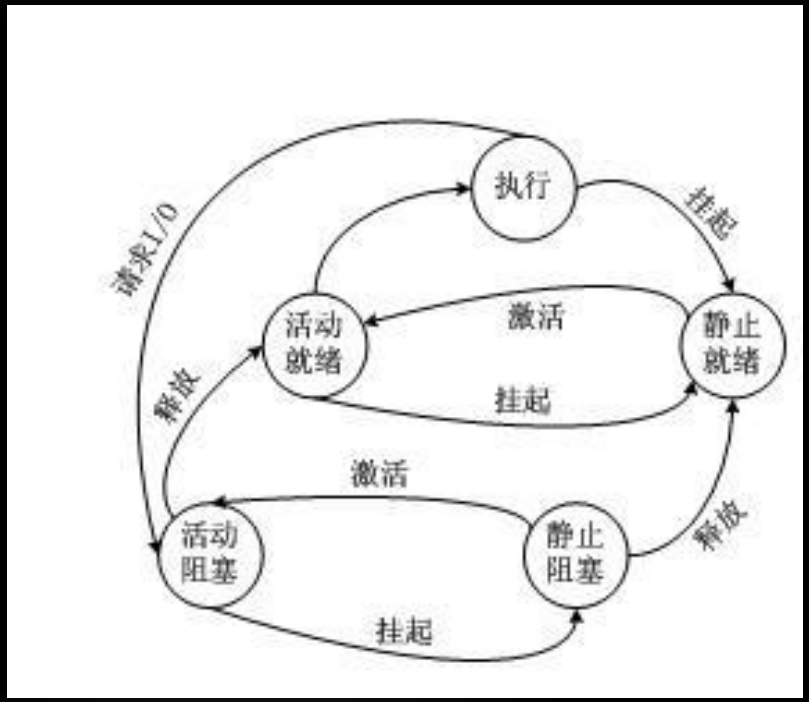
就绪(等待)状态→结束状态：



难点精讲之进程挂起状态

引入挂起状态的原因：

- 终端用户的请求。
- 父进程请求。
- 负荷调节的需要。
- 操作系统的需要。



难点精讲之进程挂起状态

在引入挂起状态后，需要增加从挂起状态(静止状态)到非挂起状态(活动状态)的转换。可有以下几种情况：

- 活动就绪+静止就绪。当进程处于未被挂起的就绪状态时，称此为活动就绪状态。当用挂起原语挂起该进程后，该进程便转变为静止就绪状态，处于静止就绪状态的进程不再被调度执行。
- 活动阻塞+静止阻塞。当进程处于未被挂起的阻塞状态时，称它是处于活动阻塞状态。当用挂起原语将它挂起后，进程便转变为静止阻塞状态。处于该状态的进程在其所期待的事件出现后，将从静止阻塞变为静止就绪。
- 静止就绪+活动就绪。处于就绪状态的进程，若用激活原语激活后，该进程将转变为活动就绪状态。
- 静止阻塞+活动阻塞。处于阻塞状态的进程，若用激活原语激活后，该进程将转变为活动阻塞状态。

难点精讲之进程控制块作用

- 进程控制块PCB是进程实体的一部分，是操作系统中最重要的记录型数据结构。
- PCB中记录了操作系统所需的、用于描述进程的当前情况以及控制进程运行的全部信息。
- 使一个在多道程序环境下不能独立运行的程序(含数据)，成为一个能独立运行的基本单位，一个能与其他进程并发执行的进程。
- OS是根据PCB来对并发执行的进程进行控制和管理的。

难点精讲之进程创建过程

- 接收新建进程运行初始值、初始优先级、初始执行程序描述等由父进程传来的参数。
- 请求分配进程描述块PCB空间，得到一个内部数组进程标识。用从父进程传来的参数初始化PCB表。
- 产生描述进程空间的数据结构，如页表，用初始参数指定的执行文件初始化进程空间，如建立城乡区、数据区、栈区等。
- 用进程运行初始值初始化处理机现场保护区。建立一个现场栈帧，等该进程第一次被调度后会从该栈帧恢复现场，从而能够进入用户程序的入口点运行。
- 置好父进程等关系域。
- 将进程置成就绪状态。将PCB挂入就绪队列，等待被调度运行。

难点精讲之进程终止过程

- 根据被终止进程的标识符，从PCB集合中检索出该进程的PCB，从中读出该进程的状态。
- 若被终止进程正处于执行状态，应立即终止该进程的执行，并置调度标志为真，用于指示该进程被终止后应重新进行调度。
- 若该进程还有子孙进程，还应将其所有子孙进程予以终止，以防它们成为不可控的进程。
- 将被终止进程所拥有的全部资源，或者归还给其父进程，或者归还给系统。
- 将被终止进程(它的PCB)从所在队列(或链表)中移出，等待其他程序来搜集信息。

难点精讲之进程阻塞过程

- 正在执行的进程，当发现上述某事件时，由于无法继续执行，于是进程便通过调用阻塞原语block把自己阻塞。可见，进程的阻塞是进程自身的一种主动行为。
- 进入block过程后，由于此时该进程还处于执行状态，所以应先立即停止执行，把进程控制块中的现行状态由“执行”改为阻塞，并将PCB插入阻塞队列。
- 转调度程序进行重新调度，将处理机分配给另一就绪进程，并进行切换，即保留被阻塞进程的处理机状态(在PCB中)，再按新进程的PCB中的处理机状态设置CPU的环境。

难点精讲之进程唤醒过程

当被阻塞进程所期待的事件出现时，如I/O完成或其所期待的数据已经到达，则由有关进程(比如，用完并释放了该I/O设备的进程)调用唤醒原语wakeup()，将等待该事件的进程唤醒。

- 唤醒原语执行的过程是：
- 把被阻塞的进程从等待该事件的阻塞队列中移出，将其PCB中的现行状态由阻塞改为就绪；
- 将该PCB插入到就绪队列。

难点精讲之进程通信

高级通信机制可归结为三大类：

共享存储器系统、消息传递系统以及管道通信系统。

- **共享存储器系统**

① 基于共享数据结构的通信方式

公用数据结构的设置及对进程间同步的处理，都是程序员的职责。这种通信方式是低效的，只适于传递相对少量的数据。

② 基于共享存储区的通信方式

为了传输大量数据，在存储器中划出了一块共享存储区，诸进程可通过对共享存储区中数据的读或写来实现通信。

难点精讲之进程通信

高级通信机制可归结为三大类：

共享存储器系统、消息传递系统以及管道通信系统。

- **共享存储器系统**

① 基于共享数据结构的通信方式

公用数据结构的设置及对进程间同步的处理，都是程序员的职责。这种通信方式是低效的，只适于传递相对少量的数据。

② 基于共享存储区的通信方式

为了传输大量数据，在存储器中划出了一块共享存储区，诸进程可通过对共享存储区中数据的读或写来实现通信。

难点精讲之进程通信

高级通信机制可归结为三大类：

共享存储器系统、消息传递系统以及管道通信系统。

- **共享存储器系统**

① 基于共享数据结构的通信方式

公用数据结构的设置及对进程间同步的处理，都是程序员的职责。这种通信方式是低效的，只适于传递相对少量的数据。

② 基于共享存储区的通信方式

为了传输大量数据，在存储器中划出了一块共享存储区，诸进程可通过对共享存储区中数据的读或写来实现通信。

难点精讲之进程通信

- **消息传递系统**

① 消息传递机制都是用得最广泛的一种进程间通信的机制。

② 在消息传递系统中，进程间的数据交换，是以格式化的消息为单位的。

③ 消息传递系统的通信方式属于高级通信方式。

a) 直接通信方式

b) 间接通信方式

难点精讲之进程通信

- **消息传递系统**

① 消息传递机制都是用得最广泛的一种进程间通信的机制。

② 在消息传递系统中，进程间的数据交换，是以格式化的消息为单位的。

③ 消息传递系统的通信方式属于高级通信方式。

a) 直接通信方式

b) 间接通信方式

难点精讲之进程通信

- **消息传递系统**

① 消息传递机制都是用得最广泛的一种进程间通信的机制。

② 在消息传递系统中，进程间的数据交换，是以格式化的消息为单位的。

③ 消息传递系统的通信方式属于高级通信方式。

a) 直接通信方式

b) 间接通信方式

难点精讲之进程通信

- **管道通信系统**

① 所谓“管道”，是指用于连接一个读进程和一个写进程以实现它们之间通信的一个共享文件，又名pipe文件。

② 向管道(共享文件)提供输入的发送进程(即写进程)，以字符流形式将大量的数据送入管道。

③ 接受管道输出的接收进程(即读进程)，则从管道中接收(读)数据。

④ 发送进程和接收进程是利用管道进行通信的，故又称为管道通信。

⑤ 为了协调双方的通信，管道机制必须提供以下三方面的协调能力：

⑥ 互斥，即当一个进程正在对pipe执行读/写操作时，其他(另一)进程必须等待。

⑦ 同步，指当写(输入)进程把一定数量(如4kB)的数据写入pipe，便去睡眠等待，直到读(输出)进程取走数据后，再把它唤醒。当读进程读一空pipe时，也应睡眠等待，直至写进程将数据写入管道后，才将之唤醒。

⑧ 确定对方是否存在，只有确定了对方已存在时，才能进行通信。

难点精讲之线程与进程比较

进程	线程
• **调度** 资源拥有的基本单位 • **并发性** 进程之间可以并发执行 • **拥有资源** 进程可以拥有资源，是系统中拥有资源的一个基本单位。 • **系统开销** 在创建或撤销进程时，系统都要为之创建和回收进程控制块，分配或回收资源，付出的开销明显大于线程创建或撤销时的开销。	•**调度** 调度和分派的基本单位 •**并发性** 不仅进程之间可以并发执行，而且在一个进程中的多个线程之间亦可并发执行。 •**拥有资源** 线程自己不拥有系统资源(也有一点必不可少的资源)，但它可以访问其隶属进程的资源。 •**系统开销** 在创建或撤销线程时，付出的开销明显小于进程创建或撤销时的开销。 线程的切换、同步和通信都无须操作系统内核的干预。

难点精讲之同步机制遵循规则

- 空闲让进：当无进程处于临界区时，表明临界资源处于空闲状态，应允许一个请求进入临界区的进程立即进入自己的临界区，以有效地利用临界资源。
- 忙则等待：当已有进程进入临界区时，表明临界资源正在被访问。因而其他试图进入临界区的进程必须等待，以保证对临界资源的互斥访问。
- 有限等待：对要求访问临界资源的进程，应保证在有限时间内能进入自己的临界区，以免陷入“死等”状态。
- 让权等待：当进程不能进入自己的临界区时，应立即释放处理机。以免进程陷入“忙等”状态。

难点精讲之进程与管程的区别

	进程	管程
1	进程定义的是私有数据结构PCB	管程定义的是公共数据结构，如消息队列等
2	由顺序程序执行有关的操作	主要是进行同步操作和初始化操作
3	进程的目的在于实现系统的并发性	管程的设置是解决共享资源的互斥使用问题
4	进程通过调用管程中的过程对共享数据结构实行操作，该过程就如通常的子程序一样被调用，是主动工作方式	管程为被动工作方式
5	进程之间能并发执行	管程则不能与其调用者并发
6	进程具有动态性，由“创建”而诞生，由“撤销”而消亡	管程是操作系统中的一个资源管理模块，供进程调用

难点精讲之生产者—消费者问题

① 首先，在每个程序中用于实现互斥的wait(mutex)和signal(mutex)必须成对地出现；
② 其次，对资源信号量empty和full的wait和signal操作，同样需要成对地出现，但它们分别处于不同的程序中；
③ 最后，在每个程序中的多个wait操作顺序不能颠倒。应先执行对资源信号量的wait操作，然后再执行对互斥信号量的wait操作，否则可能引起进程死锁。

- **管程解决方法**

在利用管程方法来解决生产者—消费者问题时，首先便是为它们建立一个管程，其中包括两个过程：

① (1) put(item)过程。生产者利用该过程将自己生产的产品投放到缓冲池中，并用整型变量count来表示在缓冲池中已有的产品数目，当count≥n时，表示缓冲池已满，生产者须等待。
② (2)get(item)过程。消费者利用该过程从缓冲池中取出一个产品，当count≤0时，表示缓冲池中已无可取用的产品，消费者应等待。

难点精讲之读者—写者问题

- **问题描述**

一组读者(指对共享数据对象只要求读的进程)与一组写者(指对共享数据对象只要求写的进程)循环访问共享的同一个数据对象，规定多个读者可以同时读这个数据对象，但不允许多个写者对这个数据对象进行写操作。也不允许读者，写者同时访问这个数据对象。

- **问题解析**

为实现Reader与Writer进程间在读或写时的互斥而设置了一个互斥信号量wmutex。另外，再设置一个整型变量Readcount表示正在读的进程数目。由于只要有一个Reader进程在读，便不允许Writer进程去写。因此，仅当Readcount=0，表示尚无Reader进程在读时，Reader进程才需要执行wait(wmutex)操作。若wait(wmutex)操作成功，Reader进程便可去读，相应地，做Readcountll操作。同理，仅当Reader进程在执行丁Readcount减1操作后其值为0时，才须执行signal(wmutex)操作，以便让Writer进程写。又因为Readeount是一个可被多个Reader进程访问的临界资源，因此，应该为它设置一个互斥信号量rmutex。

难点精讲之哲学家进餐问题

- **问题描述**

5个哲学家围绕一张圆桌而坐，桌子上放着5支筷子，每两个哲学家之间放一支；哲学家的动作包括思考和进餐，进餐时需要同时拿起他左边和右边的两支筷子，思考时则同时将两支筷子放回原处。如何保证哲学家们的动作有序进行?

如：不出现相邻者同时要求进餐；不出现有人永远拿不到筷子。

- **问题解析**

经分析可知，放在桌子上的筷子是临界资源，在一段时间内只允许一位哲学家使用。为了实现对筷子的互斥使用，可以用一个信号量表示一支筷子，由这五个信号量构成信号量数组。

- **死锁问题解决方法：**

① 至多只允许有四位哲学家同时去拿左边的筷子，最终能保证至少有一位哲学家能够进餐，并在用毕时能释放出他用过的两支筷子，从而使更多的哲学家能够进餐。

② 仅当哲学家的左、右两支筷子均可用时，才允许他拿起筷子进餐。

③ 规定奇数号哲学家先拿他左边的筷子，然后再去拿右边的筷子，而偶数号哲学家则相反。

难点精讲之调度的基本准则

不同的调度算法可满足不同的要求，一般来说批处理系统调度算法要求系统进程平均周转时间尽量短，分时系统要求进程能够轮流运行。要想得到一个满足所有用户和系统要求的算法集合是不可能的。设计调度程序，一方面要满足特定系统用户的要求(如某些实时和交互进程快速响应要求)，另一方面要考虑系统整体效率(如减少整个系统进程平均周转时间)，同时还要考虑调度算法的开销。

调度方式：

- 抢占方式
- 非抢占方式

难点精讲之典型调度算法

- 先来先服务
- 短作业优先
- 时间片轮转
- 优先级调度
- 多级反馈队列
- 高响应比优先

难点精讲之产生死锁的原因

- 竞争资源。当系统中供多个进程共享的资源数目不足以满足诸进程的需要时，会引起诸进程对资源的竞争而产生死锁。
- 进程间推进顺序非法。进程在运行过程中，请求和释放资源的顺序不当，也同样会导致产生进程死锁。

死锁的必要条件：

- 互斥条件：指进程对所分配到的资源进行排他性使用，即在一段时间内某资源只由一个进程占用。如果此时还有其OS进程请求该资源，则请求者只能等待，直至占有该资源的进程用毕释放。
- 不可剥夺条件：指进程已获得的资源，在未使用完之前，不能被剥夺，只能在使用完时由自己释放。
- 请求和保持条件：指进程已经保持了至少一个资源，但又提出了新的资源请求，而该资源又已被其他进程占有，此时请求进程阻塞，但又对自己已获得的其他资源保持不放。
- 环路条件：指在发生死锁时，必然存在一个进程和资源的环形链。

难点精讲之死锁预防

预防死锁的方法是使四个必要条件中的第2、3、4个条件之一不能成立，来避免发生。

- 破坏条件1(互斥条件)：难以否定，但可采用相应的技术，如利用假脱机技术，即用可共享使用的设备模拟非共享的设备。
- 破坏条件2(不可剥夺条件)：容易不定，可制定相应的规则即可，例如，当一个进程(程序)申请某资源被拒绝，则必须释放已占用的资源，如需要再与其他所需资源一起申请。对CPU还可进行可剥夺分配。
- 破坏条件3(请求和保持条件)：也是很容易否定的，只要分配策略上规定一个进程(或程序)一次将所需资源一次申请到位。用完后释放。可以全部用完后，统一释放，也可使用完后立即释放，只要是一次申请到的，系统就不会出现死锁。
- 破坏条件4(环路条件)：实际上系统不采用部分分配，也就破坏了环路条件。

第三章 存储管理

内存管理可考的点也很多，同样也可以有灵活的考查方法。但是相比进程管理来说，这一部分理解起来要相对简单，各种存储管理的算法的思想都是比较直接的，难点在于要记住解决某一个问题的算法有那几个，每一个算法的运行过程是怎么样的。这一章典型的综合应用题出现在：内存的连续分配算法，比如给出内存的申请和释放序列，要求解空闲块列表；非连续分配管理方式下虚拟地址和物理地址的转换，这一点可以和组成原理中的虚拟存储器结合来看；各种页面置换算法产生的缺页数的统计，经典的解法是表格法。至于选择题的点，这一章有比较多，除了上述综合题点都可以简化后出现在选择题中外，还需要注意内存管理的基本概念，如装入、链接、逻辑地址、物理地址、交换、覆盖等等，各种主存分配方式的工作过程以及优缺点对比，虚存的基本概念，抖动、工作集、程序局部性原理以及请求分段请求分页的基本原理。

难点精讲之存储器层次结构

- 主存储器：主存储器(简称内存或主存)是计算机系统中一个主要部件，用于保存进程运行时的程序和数据，也称可执行存储器。CPU的控制部件只能从主存储器中取得指令和数据。CPU与外围设备交换的信息一般也依托于主存储器地址空间。
- 寄存器：寄存器访问速度最快，完全能与CPU协调工作，但价格十分昂贵，因此容量不可能做得很大。寄存器用于加速存储器的访问速度，如用寄存器存放操作数，或用作地址寄存器加快地址转换速度等。
- 高速缓存：根据程序执行的局部性原理(即程序在执行时将呈现出局部性规律，在较短的时间内，程序的执行仅局限于某个部分)，将主存中一些经常访问的信息存放在高速缓存中，减少访问主存储器的次数，可大幅度提高程序执行速度。
- 磁盘缓存：由于目前磁盘的I/O速度远低于对主存的访问速度，因此将频繁使用的一部分磁盘数据和信息，暂时存放在磁盘缓存中，可减少访问磁盘的次数。磁盘缓存本身并不是一种实际存在的存储介质，它依托于固定磁盘，提供对主存储器存储空间的扩充。

难点精讲之交换的基本思想

交换的基本思想是，把处于等待状态(或在CPU调度原则下被剥夺运行权利)的作业从内存移到辅存，把内存空间腾出来，这一过程又叫做换出；把准备好竞争CPU运行的作业从辅存移到内存，这一过程又称为换入。应特别注意的是，当作业处于等待状态而准备出交换时，应注意该作业是否满足出交换条件。如果一个作业正在进行I/O活动则不能出交换，否则该作业的I/O数据区将被新换入的作业占用，导致错误。因此，应该等作业的VO活动结束后才能出交换。当然可以在操作系统区中开辟I/O缓冲区，将数据从外设输入或将数据输出到外设的VO活动在系统缓冲区中进行，这时在系统缓冲区与外设I/O时作业交换不受限制。交换技术可以作为通用的技术，与后面所述的各种存储方法相结合，借助辅存空间在逻辑上实现主存空间的扩展，从而有效地改善存储利用率。

难点精讲之连续分配管理方式

① 单一连续分配

② 固定分区分配

③ 动态分区分配

- 首次适应算法
- 循环首次适应算法
- 最佳适应算法
- 最坏适应算法
- 快速适应算法

④ 可重定位分区分配

难点精讲之页式虚拟存储基本思想

实现页式虚存空间的基本方法是，在页式存储管理的基础上，仅把进程的一部分页放在主存中。页表项中会注明对应的页是在主存还是在辅存。程序执行时，当访问的页不在主存时，根据页表项的指引，从辅存将其调入主存。如果这时已无可用的物理空间，则从主存淘汰若干页。

由此可知，要实现虚存，首先要有辅存的支持，没有辅存，进程虚空间的程序和数据就没有地方保存，主存可以看成是进程虚空间的缓)中区。主存空间管理必须以非连续的存储管理法为基础。如果进程虚空间的页面中有有效数据，那么这些页面数据可以来源于存在辅存空间中的文件。在进程执行时会新生成数据页面或其他数据页面，通常这些页面在释放主有时会被保存到辅存的交换区。

难点精讲之页面置换算法

- 最佳置换算法
- 先进先出置换算法
- 最近最少使用置换算法
- 时钟置换算法

难点精讲之抖动

“抖动”：也称“颠簸”，刚被调出的页面又立即要装入，而装入不久又被选中调出，调出不久又被装入。

页面替换中的一种最糟糕情形是；刚刚换出主存的页面马上需要换入主存，刚刚换入主存的页面马上就要换出主存。这种情形就称为内存抖动(Trashing)，又称为颠簸或者乒乓效应。

抖动时系统频繁发生缺页中断，系统的任何一次访存指令都伴随着一次磁盘访问，系统访存的开销甚至高于磁盘访问的开销。究其原因，主要是因为某个进程频繁访问的页面数量高于可用的物理。

一种解决抖动问题的方法是设定一个空闲物理页帧的数量下限，当系统中空闲物理页帧数量不足以支持一个进程运行时，将系统中一些进程占用的物理页帧换出主存，直到空闲物理页帧数量高于这个阈值。

这样的处理方式通常称为负载均衡(Loading Balance)。

当然，负载均衡方法也不能完全解决抖动问题，此时就只能采用其他的方法。或者终止引起抖动的进程执行，或者为计算机扩充更丈的内存。

第四章 文件管理

文件管理的重点在于文件的顺序和索引结构。这一部分最重要的应用题点在于索引文件的目录结构，要熟练掌握计算给定目录树结构下单文件的最大文件大小；其次是Unix系统的文件系统空闲块的组织方法——成组链接法，要能说清楚空闲块是怎么分配给申请空闲块的文件的，以及释放的空闲块如何加入到空闲块组里；最后是磁盘的调度算法，要熟练掌握不同调度算法寻道数的计算。文件系统其它需要了解的知识点包括：文件的相关概念，文件的逻辑结构和物理结构，目录结构以及目录管理，文件共享与保护机制，隐式链接和显式链接，空闲块的三种不同组织方法，磁盘的相关概念和参数，磁盘的结构以及调度算法的特点和优缺点对比等。

难点精讲之文件分类

•逻辑结构分类：

流式文件——有序相关信息项的集合，其基本单位是字节(B)。

记录式文件——数据记录的集合，其基本单位是逻辑记录(记录以序编号：记录1，记录2，…，记录n)，记录本身又有等长和变长之分。

•物理结构分类：

①顺序结构

a)磁盘存储空间的利用率不高。

b)对输出文件很难估计需多少磁盘块。

c)影响文件的扩展。

②链接结构

排在后面的各条记录只能在把前面各条记录读出之后才能得到，即只能按顺序读取记录。

索引结构

难点精讲之目录管理要求

- 实现“按名存取”，即用户只须向系统提供所需访问文件的名字，便能快速准确地找到指定文件在外存上的存储位置。
- 提高对目录的检索速度。通过合理地组织目录结构的方法，可加快对目录的检索速度，从而提高对文件的存取速度。
- 文件共享。在多用户系统中，应允许多个用户共享一个文件。这样就须在外存中只保留一份该文件的副本，供不同用户使用，以节省大量的存储空间，并方便用户和提高文件利用率。
- 允许文件重名。系统应允许不同用户对不同文件采用相同的名字。

难点精讲之文件控制块(FCB)

文件控制块中通常包含三类信息：

- 基本信息：文件名，文件物理位置，文件逻辑结构，文件物理结构。
- 存取控制信息类：文件拥有者的存取权限、核准用户的存取权限、一般的存取权限。
- 使用信息类：文件建立的日期与时间，文件上一次修改的日期与时间的使用信息。

难点精讲之文件存储空间管理方法

(1) 空闲表法

空闲表法属于连续分配方式，为何个文件分配一块连续的存储空间，即系统也为外存上的所有空闲区建立一张空闲表，每个空闲区对应于一个空闲表项，其中包括表项序号、该空闲区的第一个盘块号、该区的空闲盘块数等信息。再将所有空闲区按其起始盘块号递增的次序排列。

(2) 空闲链表法

空闲链表法是将所有空闲盘区拉成一条空闲链。根据构成链所用基本元素的不同，可把链表分成两种形式：空闲盘块链和空闲盘区链。

(3)位示图法

位示图是利用二进制的一位来表示磁盘中一个盘块的使用情况。
当其值为“0”时，表示对应的盘块空闲；为“1”时，表示已分配。
磁盘上的所有盘块都有一个二进制位与之对应，这样，由所有盘块所对应的位构成一个集合，称为位示图。

(4)成组链接法

1) 空闲盘块的组织：空闲盘块号栈用来存放当前可用的一组空闲盘块的盘块号及栈中尚有的空闲盘块号数N。
2) 空闲盘块的分配与回收
　当系统要为用户分配文件所需的盘块时，须调用盘块分配过程来完成。
在系统回收空闲盘块时，须调用盘块回收过程进行回收。

难点精讲之文件系统层次结构

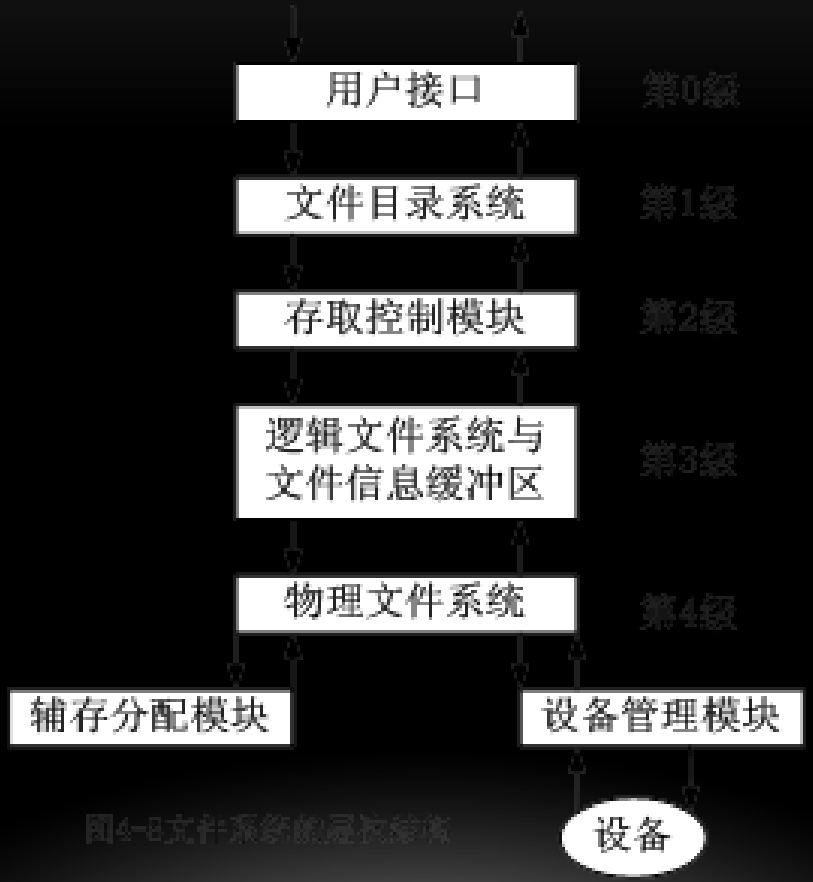


图4-8文件系统的层次结构

难点精讲之磁盘调度算法

- **减少寻找时间的方法**

① 先来先服务

② 最短寻找时间优先

③ 电梯调度

④ 单向扫描电梯调度

⑤ N-STEP算法

⑥ FSCAN算法

第五章 输入输出管理

输入输出管理这一章是非重点。需要注意的地方是五种I/O控制方式以及它们之间的对比，I/O软件的层次结构，Spooling技术以及缓冲策略，I/O调度的相关概念，设备独立性相关的概念和原理。

难点精讲之I/O系统层次

通常把I/O软件组织成四个层次：

- 用户层软件：实现与用户交互的接口，用户可直接调用在用户层提供的、与I/O操作有关的库函数，对设备进行操作。
- 设备独立性软件：负责实现与设备驱动器的统一接口、设备命名、设备的保护以及设备的分配与释放等，同时为设备管理和数据传送提供必要的存储空间。
- 设备驱动程序：与硬件直接相关，负责具体实现系统对设备发出的操作指令，驱动I/O设备工作的驱动程序。
- 中断处理程序：用于保存被中断进程的CPU环境，转入相应的中断处理程序进行处理，处理完后再恢复被中断进程的现场后返回到被中断进程。

难点精讲之DMA

当大量的数据在外部设备与主存之间移动时，一种有效的技术就是DMA(Direct Memory Access)。DMA方式比中断驱动方式要减少许多中断，减少许多CPU的I/O启动操作。

DMA的特点是：

- 数据传输的基本单位是数据块，即在CPU与I/O设备之间，每次传送至少一个数据块；
- 所传送的数据是从设备直接送入内存的，或者相反；
- 仅在传送一个或多个数据块的开始和结束时，才需CPU干预，整块数据的传送是在控制器的控制下完成的。

可见，DMA方式较之中断驱动方式，又是成百倍地减少了CPU对I/O的干预，进一步提高了CPU与I/O设备的并行操作程度。

难点精讲之I/O层次结构

层次式结构的基本思想是，将系统输入/输出的功能组织成一系列的层次，每一层次完成整个输入/输出系统功能的一个子集，每一层次都依赖其下层完成更原始的功能，并屏蔽这些功能的实现细节，从而为其上层提供各种服务。理想情况下，层次的定义应能达到这样的目标，对某一层次的修改不会引起其下层或上层代码的修改。

整个输入/输出系统可以看成具有3个层次的系统结构(如下图所示)，其中，用户层I/O是提供给用户进程使用输入/输出设备进行I/O操作的接口，运行在用户态。设备无关的操作系统软件、设备驱动及中断处理则在核心态运行，属于操作系统内核程序。当然，特定的操作系统在实现时并非严格遵守这种结构，但这种层次的划分是合理的。