操作系统课后总结

第一章

操作系统是计算机系统必要的存在，没有操作系统就很难控制计算机的各个功能，操作系统有几个目标：1）**方便性**。有了操作系统就能很方便地操纵自己的计算机；2）**有效性**。有了操作系统之后，计算机的很多功能都能用到，不浪费计算机的资源；3）**可扩充性**。操作系统能增加用户对计算机的操作，让用户有更多的选择去使用计算机的资源。4）**开放性。**操作系统能兼容到很多计算机机器，让很多用户能够用到这么方便的操作系统，实现与其他系统互连。

计算机一开始没有操作系统来操作，于是程序员只能用人工操作来操作计算机，这样的操作方式实在很浪费计算机的等待时间和资源，计算机的资源使用率很低。之后就用到了一个办法，能让用户和计算机节省时间，这个办法是“脱机”，脱机是接受和输出数据的外设备，这样减少了cpu的使用时间，让输入和输出给另外一些设备来处理。这个办法比之前的人工操作好，但是也有缺点，缺点就是在**等下一个作业的时候要等一会儿，不是自动准备的，所以出现了单道批处理系统。**

单道批处理系统可以说是最早的一种操作系统，一批作业被读入到磁带，在**监督程序**控制下，按队列方式每次调入内存**一个/道**作业，直到所有作业运行完成。虽然它解决了作业的顺序和装入的时间，但是还是有缺点，内存利用率很低。因为作业只会按顺序的来处理，导致其他作业都在等待。

多道批处理系统是能够处理多道作业，它处理了单道批处理系统做不了的事情，这个操作系统能够同时处理用户的多个作业，用户所提交的作业都先存放在外存上并排成一个队列，称为“**后备队列**”；然后，由作业调度程序按一定的算法从后备队列中选择若干个（不再是单道）作业调入内存，使它们共享CPU和系统中的各种资源。提高了cpu与I/O设备的利用率，但是这个操作系统把用户和计算机操作员给分开了，导致用户无法跟自己的作业互动，一旦运行了就无法停止，知道全部作业都运行完，这不只是唯一一个缺点，还有就是其他用户也要排队，导致用户与作业的互动时间比较长。

分时系统解决了上述遇到的问题，这个系统能够把cpu的时间分成好几个时间片，这样能给多个终端的设备有机会使用计算机的资源，减少了用户在排队等待的时间，也解决了用户不能够跟自己的作业互动。

实时系统是一个能够按实时地运行，实现，和完成几个任务地操作系统，这种系统用在于好多方面，比如工厂的产品生产，计算机或者手机上的天气预报，新闻软件，等等。

我们还看了操作系统的特性，分别是**并发，共享，虚拟和异步**。并发又分为并行和并发，并行性指的是多道作业在一时刻同时运行，并发是一个或多个事件在一时间间隔内发生。

共享，是每一个终端都能够使用cpu的资源，比如打印机，如果一位用户正在使用打印机，另一位用户也想使用，只能等待前用户用完才能占用此资源。当某进程A要使用该资源时，A必须先提出请求。如果此时该资源空闲，Os就可以将打印机分配给进程。

在打印过程中，如果其它进程也提出使用请求**，**则必须等待打印任务完成。

当A进程完成打印后，释放该资源，才能再次接受其它进程提出的请求。

这种资源共享方式就叫**互斥式共享**。

系统中还有另一类资源，允许在一段时间内由多个进程“同时”对它们进行访问。

这里的“同时”往往是宏观上的，而微观上这些进程在交替地使用这些资源。比如对磁盘设备的访问。

**并发和共享**是操作系统的两个最基本的特征。

**一方面，并发是共享的前提。没有并发，自然也不需要资源共享。**

**虚拟系统，所谓“虚拟”，是指通过某种技术把一个物理实体变为若干个逻辑上的对应物。**

**逻辑上的对应物只是一种用户的感觉。这种技术，称为虚拟技术。**

在多道程序环境下，由于进程执行上的并发方式，由于对资源使用上的共享方式，使得进程的执行“停停走走” 。每个进程在何时能获得资源**：处理机/内存/外设**，何时又会提出新的资源请求，请求会否被满足，资源要用多久、用多少，进程需要运行多久等等，这些问题都是不可预知的。

即：**进程是以不可预知的速度在向前推进，这就是进程的异步性**。

操作系统的主要功能有：**处理机管理、存储器管理、设备管理、文件管理、用户管理。**

**A）处理器管理**

在传统的多道程序环境下，要使作业运行，必须先为它创建一个或几个进程，并为之分配必要的资源。当进程运行结束时，立即撤消该进程，以便能及时回收该进程所占用的各类资源。

\***进程控制：的主要功能是为作业创建进程、撤消已结束的进程，以及控制进程在运行过程中的状态转换。** 在现代OS中，进程控制还应具有为一个进程创建若干个线程的功能和撤消(终止)已完成任务的线程的功能。

\***进程的同步：为使多个进程能有条不紊地运行，系统中必须设置进程同步机制**。进程同步的主要任务是为多个进程(含线程)的运行进行协调。有两种协调方式： ① 进程互斥方式， 这是指诸进程(线程)在对临界资源进行访问时， 应采用互斥方式； ② 进程同步方式，指在相互合作去完成共同任务的各进程(线程)间，由同步机构对它们的执行次序加以协调。

**\*进程通信：在多道程序环境下，进程(线程)相互合作去完成一个共同的任务。**而在这些进程(线程)之间，又往往需要交换信息。例如，有三个相互合作的进程， 它们是输入进程、计算进程和打印进程。输入进程负责将所输入的数据传送给计算进程；计算进程利用输入数据进行计算， 并把计算结果传送给打印进程；最后，由打印进程把计算结果打印出来。进程通信的任务就是用来实现在相互合作的进程之间的信息交换。

当相互合作的进程(线程)处于同一计算机系统时，通常在它们之前是采用直接通信方式，即由源进程利用发送命令直接将消息(message)挂到目标进程的消息队列上，以后由目标进程利用接收命令从其消息队列中取出消息。

\***调度：在后备队列上等待的每个作业，通常都要经过调度才能执行。**在传统的操作系统中，包括**作业调度和进程调度**两步。作业调度的基本任务，是从**后备队列**中按照一定的算法，选择出若干个作业，为它们**分配资源**。 在将它们调入内存后，便分别为它们建立进程，并按照一定的算法将它们插入就绪队列。而**进程调度的任务，则是从进程的就绪队列中选出一新进程，把处理机分配给它**。在多线程OS中，通常是把线程作为独立运行和分配处理机的基本单位，把就绪线程排成一个队列，每次调度时，是从就绪线程队列中选出一个线程，把处理机分配给它。

B）存储器管理功能

OS在实现内存分配时，可采取**静态和动态两种方式**。在静态分配方式中，每个作业的内存空间是在作业装入时确定的；在作业装入后的整个运行期间， 不允许该作业再申请新的内存空间，也不允许作业在内存中“移动”； 在动态分配方式中，每个作业所要求的基本内存空间， 也是在装入时确定的，但允许作业在运行过程中，继续申请新的附加内存空间，以适应程序和数据的动态增涨，也允许作业在内存中“移动”。

主要功能如下：

**\*内存分配**

**\*内存保护**

**\*地址映射**

**\*内存扩充**

C）设备管理功能

设备管理用于管理计算机系统中所有的外围设备， 而设备管理的主要任务是，**完成用户进程提出的I/O请求； 为用户进程分配其所需的I/O设备；提高CPU和I/O设备的利用率；提高I/O速度；方便用户使用I/O设备。**为实现上述任务，设备管理应具有缓冲管理、设备分配和设备处理，以及虚拟设备等功能。

主要功能如下：

**\*缓冲管理**

**\*设备管理**

**\*设备处理**

D）文件管理功能

主要功能如下：

\***文件存储空间的管理**

**\*目录管理**

**\*文件的读/写管理和保护**

**E）操作系统与用户之间的接口**

1. 用户接口

(1) **联机用户接口。**这是为联机用户提供的，它由一组键盘操作命令及命令解释程序所组成。当用户在终端或控制台上每键入一条命令后，系统便立即转入命令解释程序，对该命令加以解释并执行该命令。在完成指定功能后，控制又返回到终端或控制台上，等待用户键入下一条命令。这样，用户可通过先后键入不同命令的方式，来实现对作业的控制，直至作业完成。

(2) **脱机用户接口。**该接口是为批处理作业的用户提供的，故也称为批处理用户接口。该接口由一组作业控制语言JCL组成。批处理作业的用户不能直接与自己的作业交互作用，只能委托系统代替用户对作业进行控制和干预。这里的作业控制语言JCL便是提供给批处理作业用户的、为实现所需功能而委托系统代为控制的一种语言。作业一直在作业说明书的控制下运行，直至遇到作业结束语句时，系统才停止该作业的运行。

(3) 图形接口

用户虽然可以通过联机用户接口来取得OS的服务，但这时要求用户能熟记各种命令的名字和格式，并严格按照规定的格式输入命令，这既不方便又花时间，于是，图形用户接口便应运而生。图形用户接口采用了图形化的操作界面， 用非常容易识别的各种图标(icon)来将系统的各项功能、各种应用程序和文件，直观、逼真地表示出来。用户可用鼠标或通过菜单和对话框，来完成对应用程序和文件的操作。此时用户已完全不必像使用命令接口那样去记住命令名及格式，从而把用户从繁琐且单调的操作中解脱出来。

1. 程序接口

该接口是为用户程序在执行中访问系统资源而设置的，是用户程序取得操作系统服务的惟一途径。它是由一组系统调用组成，每一个系统调用都是一个能完成特定功能的子程序，每当应用程序要求OS提供某种服务(功能)时，便调用具有相应功能的系统调用。在高级语言以及C语言中，往往提供了与各系统调用一一对应的库函数，这样，应用程序便可通过调用对应的库函数来使用系统调用。但在近几年所推出的操作系统中，如UNIX、OS/2版本中，其系统调用本身已经采用C语言编写，并以函数形式提供，故在用C语言编制的程序中， 可直接使用系统调用。

第二章

1、程序顺序执行特征：顺序性、封闭性、可再现性

2、程序并发执行特征：间断性、失去封闭性、不可再现性

3、前趋图：有向无循环图，用于描述进程之间执行的前后关系

表示方式：

（1）p1--->p2

（2）--->={(p1,p2)| p1 必须在p2开始前完成}

节点表示：一条语句，一个程序段，一进程。

4、进程的定义：

(1)是程序的一次执行过程，由程序段、数据段、程序控制块（PBC）三部分构成，总称“进程映像”

(2)是一个程序及其数据在处理机上顺序执行时所发生的活动

(3)是程序在一个数据集合上的运行过程

(4)进程是进程实体的运行过程，是系统进行资源分配和调度的一个独立单位

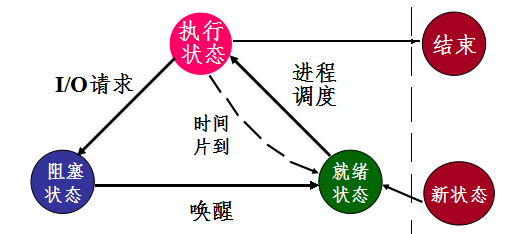
进程特征：动态性、并发性、独立性、异步性

由“创建”而产生，由“调度”而执行；由得不到资源而“阻塞”，由“撤消”而消亡

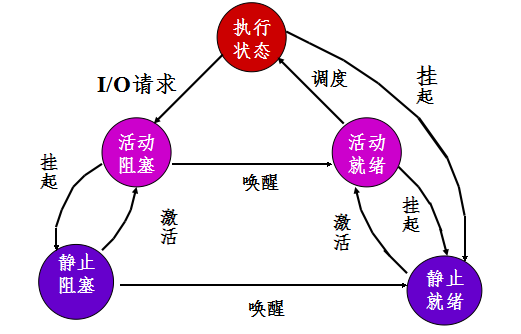
5、进程与程序关系

|  |  |  |
| --- | --- | --- |
|  | 进程 | 程序 |
| 概念 | 动态实体，强调执行过程 | 静态实体，是指令的有序集合 |
| 特征 | 并发性、独立性、异步性, 是竞争计算机系统资源的基本单位 | 无并行特性，是静止的 |
| 二者联系 | 不同进程可以共享同一程序，只要对应数据集不同 | |

6、进程的三种状态：就绪、阻塞、执行

 转换：

增加挂起：



1. **进程控制块（PCB）的作用：进程存在的唯一标志。**
2. 进程创建：

（1）申请空白PCB（一个系统的PCB是有限的）

（2）为新进程分配资源

（3）初始化PCB

（4）将新进程插入就绪队列。

进程终止：

（1）检查进程状态；

（2）执行态――>中止，且置调度标志为真。

（3）有无子孙需终止。

（4）归还资源给其父进程或系统。

（5）从PCB队列中移出PCB.

9、原语：是由若干条指令组成的，用于完成一定功能的一个过程，原语在执行中不允许被中断，**原语的作用是实现进程的通信和控制。**

10、进程同步：并发进程在执行次序上的协调，以达到有效的资源共享和相互合作，使程序执行有可再现性。

11、两种制约关系：

资源共享关系：（进程间接制约）

**需互斥地访问临界资源。**

相互合作关系：（进程直接制约）

12、**临界资源：一次仅允许一个进程访问的资源**

引起不可再现性是因为临界资源没有互斥访问。

临界区：**进程访问临界资源的那段代码。**

13、同步资源应遵循的准则：**空闲让进、忙则等待、有限等待、让权等待**

14、整形信号量、记录型信号量、AND型信号量

1. 生产者消费者问题
2. 操作系统中引入进程的目的是为了使多个程序能并发执行，以提高资源利用率和系统吞吐量，在操作系统中再次引入县城，则是为了减少程序在并发执行时所付出的时空开销。
3. 管程的定义：**定义了一个数据结构和能为并发进程在该数据结构上执行的一组操作，这祖操作能同步进程和个i变管程中的数据。**
4. 线程的概念：**线程是进程中的一个实体，是被系统独立调度和分派的基本单位。**

一个线程可以创建和撤销另一个线程；**同一个进程中的多个线程之间可以并发执行**。

1. 进程与线程比较

|  |  |  |
| --- | --- | --- |
|  | 进程 | 线程 |
| 引入目的 | 能并发执行，提高资源的利用率和系统吞吐量 | 提高并发执行的程度，减小开销，进一步提高资源的利用率和系统吞吐量 |
| 并发性 | 较低 | 较高 |
| 基本属性（调度） | 资源拥有的基本单位 | 独立调度/分派的基本单位 |
| 基本状态 | 就绪；执行；等待 | 就绪；执行；等待 |
| 拥有资源 | 资源拥有的基本单位 | 无资源分配 |
| 系统开销 | 创建/撤销/切换时空开销大 | 创建/撤销/切换时空开销较小 |
| 系统操作 | 创建，撤销，切换 | 创建，撤销，切换 |
| 存在标志 | 进程控制块（PCB） | 线程控制块（TCB） |
| 关系 | 单进程单线程；单进程多线程；多进程单线程；多进程多线程 | |

第三章

※1、处理机调度：高级调度（作业调度、长程调度）：将外存上的作业插入就绪队列。

中级调度（中程调度）：为提高系统吞吐量和内存利用率而引入的一内外存对换功能。

低级调度（进程调度、短程调度）：把处理器分配给程序，对象是进程。

高级调度与低级调度的区别

高级调度又称为作业调度或长程调度，调度对象是作业，作业调度往往发生于一个（批）作业运行完毕，退出系统，而需要重新调入一个（批）作业进入内存时，故作业调度的周期长；低级调度又称为进程调度和短程调度，调度物件为进程（或内核级线程），进程调度的运行频率最高，是最基本的一种调度，**多道批处理、分时、实时**三类OS中必须配置这种调度。

引入中级调度的主要目的：**是为了提高系统资源的利用率和系统吞吐量。**

2、选择调度方式和调度算法的准则:

面向用户准则： （1）周转时间短

1. 响应时间快（对交互性作业）
2. 截止时间保证（特别对于实时系统）
3. 优先权准则（需要抢占调度）

3、面向系统准则:

1. 系统吞吐量高
2. 处理机利用率好
3. 各类资源平衡利用

4、调度算法：在ＯＳ中调度实质是一种资源的分配。先来先服务（FCFS）和短作业（SJF）优先调度算法（SJF/SPF）、高优先权优先调度算法、基于时间片的轮转调度算法

实时调度算法：

最早截止时间优先（EDF）：

根据任务的截止时间来确定任务的优先级

截止时间越早，优先级越高

可以是抢占式或非抢占式

最低松弛度优先（LLF）：

主要用于可抢占的调度方式中

松弛度 = 必须完成时间 - 本身运行时间 - 当前时间

5、**何谓死锁？产生死锁的原因和必要条件是什么？**

a.死锁是指多个进程因竞争资源而造成的一种僵局，若无外力作用，这些进程都将永远不能再向前推进；

b.产生死锁的原因有二，一是竞争资源，二是进程推进顺序非法；

c.必要条件是: 互斥条件，请求和保持条件，不剥夺条件和环路等待条件

6、**处理死锁的基本方法**：

(1)预防死锁—破坏产生死锁的四个必要条件中的一个或几个条件

(2)避免死锁—破坏产生死锁的四个必要条件

(3)检测死锁—通过系统设置的检测机构，及时检测出死锁的发生

(4)解除死锁—撤销或挂起一些进程

**预防死锁的方法:**

a.摒弃"请求和保持"条

b.摒弃"不剥夺"条件

c.摒弃"环路等待"条件

7、（大题）**银行家算法、安全性算法**

1. 如果Request<=Need，去（2），否则错误
2. 如果Request<=Available，去（3），否则等待
3. 系统尝试分配资源给进程Pi，并修改值

Available = Available – Request

Allocation= Allocations+ Request

Need= Need – Request

(4)安全检查，即所有进程都顺利完成

第四章

1. **程序的装入**：

**绝对装入**：编译后，装入前已产生了绝对地址（内存地址），装入时不再作地址重定位。

**可重定位装入**：

静态重定位：装入时完成，主要工作是对相对地址中的指令和数据地址的调整过程。

动态运行时装入（动态重定位）：程序执行过程中，当访问指令或数据时，才进行的地址变换方法

1. **程序的链接**：静态链接、装入时动态链接、运行时动态链接。
2. **连续分配方式**：

单一连续分配：用于单用户单任务

分区式分配：

固定分区分配：分区大小不相等的利用率更高；

动态分区分配；

可重定位分区分配。

1. **分配算法**：

首次适应算法

要求：分区按低址――高址链接

特点：找到第一个大小满足的分区，划分。有外零头，低址内存使用频繁。

循环首次适应算法

从上次查找的位置的下一个空闲空闲分区开始查找。

特点：空闲分区分布均匀，提高了查找速度；缺乏大的空闲分区。

最佳适应算法

分区按大小递增排序

分区释放时需插入到适当位置。

1. **对换**：

概念：将阻塞的进程，暂时不用的程序、数据换出，将具备运行条件的数据换入。

类型：整体对换、部分对换（页面对换、分段对换）

空间管理：

分为对换区和文件区

对换区比文件区侧重于对换速度

因此，对换区一般采用连续分配。采用数据结构和分配回收类似于可变化分区分配。

1. **页面与页表**

分页存储管理是将一个进程的逻辑地址空间分成若干个大小相等的片，称为页面或页。

由于进程的最后一页经常装不满一块而形成不可利用的碎片，称为“页内碎片”。

系统为每个进程建立一张页面映像表，简称页表。页表的作用是实现从页号到物理块号的地址映射。

1. **基本分页存储管理方式** ：

连续分配引起碎片

碎片问题的解决：紧凑方式消耗系统开销

解决方式用离散分配：分页、分段、段页

页面和物理块：由机器的地址结构决定，将一个进程的逻辑地址空间分成若干个大小相等的片，称为页面或页，并为各页加以编号，从0开始。相应的把内存空间分成与页面相同大小的若干个存储块，称为物理块或页框。

页面大小：通常在512B~8KB

页太大，页内碎片大。

页太小：页表可能很长，换入/出效率低

※地址结构（逻辑地址换物理地址基础）（参照ppt第四章和书P139）

※地址变换机构：页表完成逻辑页号到物理块号的映射

有快表访问内存速度会提高，但是快表不能太多

※※题型：

**已知一次存取时间和快表命中率求访问时间**

**已知逻辑地址求物理地址**

1. **基本分段储存管理**：即多重定位分区管理

**原因：**引入分段存储管理方式， 主要是为了满足用户和程序员的下述一系列需要：

（1）方便编程（2）分段共享（3）分段保护（4）动态链接（5）动态增长

对用户而言分段是二维的：段号+段内地址

**分段基本原理**：每个段定义了一组逻辑信息，主程序段、子程序段、数据段等

分页与分段的区别：

（1）页是信息的物理单位，段是逻辑单位

（2）页长度固定，段长度不固定（由用户指定）

（3）一维与二维

段式系统易于共享

9、虚拟存储器

常规存储器特征：**一次性、驻留性**

局部性原理：时间局部性（循环执行）、空间局部性（顺序执行）

虚拟存储器定义：具有请求调入功能和置换功能，能从逻辑上对内存容量进行扩充的一种存储系统

虚拟存储器实质：以时间换空间，但时间牺牲不大

虚拟存储器特征：离散性（部分装入）、多次性（局部多次装入）、对换性、虚拟性

10、页面置换算法：FCFS、LRU、CLOCK。

第五章

* + - 1. 常规存储器管理方式的特征：

A) 一次性，作业被一次性地全部装入内存，因此才会出现前述两个问题。事实上，作业运行时，一段时间内只用到部分的程序与数据，没必要全部装入。

B）驻留性，作业装入内存后，会一直驻留在内存，直到结束，就算一段时间不运行**IO**，也不让出内存。

* + - 1. 局部性原理

程序在执行时将呈现出局部性规律，即在一较短的时间内，程序的执行仅局限于某个部分，相应地，它所访问的存储空间也局限于某个区域。

局限性又表现在下述两个方面：  
　　(1) 时间局限性。

某指令或数据刚被执行，可能会在短时间内被再次执行。例如，程序中使用了while、for这种循环语句。  
　　(2) 空间局限性。

某存储单元被访问后，可能会接着访问其附近的存储单元。一方面程序是顺序执行的，另一方面存在象数组这样的数据结构

* + - 1. 虚拟存储器的特征：

(1) 多次性。分多次调入，每次只调入部分运行；当需要运行其它部分时，再将它调入。程序道数（并发度）、存储器利用率都会因此提高。  
　　(2) 对换性。将暂时不用的程序/数据/进程换出到外存，需要时再换入内存+。  
　　(3) 虚拟性。感觉到：容量大了->就可以放入大的作业、多道作业-> 增加了并发度、提高了内存利用率、提高了系统吞吐量**。**有了对换性，才有了虚拟性。

* + - 1. 虚拟存储器的实现方法

A）请求分页系统

分页管理+请求调页+页面置换  
　　1) 硬件支持  
　　 (1) 请求分页的页表机制（增加了几个页表项）。  
　　 (2) 缺页中断机构

因为部分调入，产生缺页中断，请求OS调入所缺的页。  
　　 (3) 地址变换机构（逻辑地址->物理地址）  
　　2) 实现请求分页的软件（解决：请求调页+页面置换）

B）请求分段系统

1) 硬件支持  
　　主要的硬件支持有：  
　　(1) 请求分段的段表机制。  
　　(2) 缺页中断机构。  
　　(3) 地址变换机构。  
　　2) 软件支持

5. 请求分页存储管理方式

1）请求页表机制

主要数据结构是：请求页表，其基本作用仍然是将用户地址空间中的逻辑地址映射为内存空间中的物理地址。为了满足页面换进换出的需要，在请求页表中又增加了四个字段。这样，在请求分页系统中的每个页表应含以下诸项：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 页号 | 物理块号 | **状态位P** | **访问字段A** | **修改位M** | **外存地址** |

2）缺页中断机构

缺页中断：因访问的页不在内存，而引起的中断。

与一般中断不同之处：  
　　(1) 在指令执行期间产生和处理中断信号。  
　　(2) 一条指令在执行期间可能产生多次缺页中断。

3）地址变换机构

请求分页系统中的地址变换机构：是在分页系统地址变换机构的基础上，为实现虚拟存储器，再增加了某些功能所形成的，如产生和处理缺页中断，以及从内存中换出一页的功能等等。

* + - 1. 请求分页中的内存分配

1）最小物理块数的确定

一个显而易见的事实是，随着为每个进程所分配的物理块的减少，将使进程在执行中的缺页率上升，从而会降低进程的执行速度。

最小物理块数：是指能保证进程正常运行所需的最小物理块数。当系统为进程分配的物理块数少于此值时，进程将无法运行。

2）内存分配策略

在请求分页系统中，可采取两种内存分配策略：**固定和可变分配策略。**

在进行置换时， 也可采取两种置换策略：即**全局置换和局部置换**。于是可组合出以下三种适用的策略。

a) 固定分配局部置换

固定分配：为每个进程分配固定数目的物理块。局部置换：只置换本进程的物理块。问题：固定多大？大、小问题

b) 可变分配全局置换

可变分配：开始时分配若干个物理块，以后根据需要动态改变。全局置换：置换对象是所有进程的物理块。

c) 可变分配局部置换

3）物理块分配算法

a) **平均分配**算法

这是将系统中所有可供分配的物理块，平均分配给各个进程。

2) **按比例分配**算法

这是根据进程的大小按比例分配物理块的算法。如果系统中共有*n*个进程，每个进程的页面数为*S*i，则系统中各进程页面数的总和为：



又假定系统中可用的物理块总数为*m*，则每个进程所能分到的物理块数为*bi*，将有：



*b*应该取整，它必须大于最小物理块数。

* + - 1. 页面调入策略

1） 何时调入页面  
　　(1) 预调页策略。

预先调入可能在不久后要用到的页。后面：调入工作集中的所有页。  
　　(2) 请求调页策略。

页不在内存时，进程再发出中断请求。

2）从何处调入页面

外存分为文件区+对换区。

文件区：离散分配+访问速度慢；

对换区：连续分配+访问速度快；

(1) 系统拥有足够的对换区空间，这时可以全部从对换区调入所需页面，以提高调页速度。  
　　(2) 系统缺少足够的对换区空间，这时凡是不会被修改的文件，都直接从文件区调入；而当换出这些页面时，由于它们未被修改，则不必再将它们重写到磁盘(换出)，以后再调入时，仍从文件区直接调入。

(3)  UNIX方式。未运行过的文件，从文件区调入；运行过的文件，因被换出到了对换区，故从对换区调入。

* + - 1. 抖动与工作集

由于请求分页式虚拟存储器系统的性能优越，在正常运行情况下，它能有效地减少内存碎片，提高处理机的利用率和吞吐量，故是目前最常用的一种系统。

但如果在系统中运行的进程太多，进程在运行中会频繁地发生缺页情况，这又会对系统的性能产生很大的影响，故还须对请求分页系统的性能做简单的分析。

1. 产生抖动的原因

系统中运行的进程太多，由此

分配给每一个进程的物理块太少，不能满足进程正常运行的基本要求，结果使得：

每个进程在运行时，频繁地出现缺页。导致

系统中需要换进/调出页面的进程数目增加。

造成每个进程的大部分时间都用于页面的换进/换出，几乎不能再去做任何有效的工作，从而导致发生处理机的利用率急剧下降并趋于0的情况。我们称此时的进程是处于“抖动”状态。

* + - 1. 工作集

进程发生缺页率与进程所获得的物理块数有关。

所谓工作集，是指在某段时间间隔Δ里，进程实际所要访问页面的集合。

* + - 1. 抖动的解决方法

1. 采取局部置换策略  
 在页面分配和置换策略中，如果采取的是可变分配方式，则为了预防发生“抖动”，可采取局部置换策略。

2. 把工作集算法融入到处理机调度中  
　　当调度程序发现处理机利用率低下时，它将试图从外存调入一个新作业进入内存，来改善处理机的利用率。

3. 利用“L=S”准则调节缺页率

　 4. 选择暂停的进程

第六章

1）I/O系统的功能、模型和接口

\*I/O系统管理的主要对象：

\*I/O设备和相应的设备控制器。

\*其最主要的任务：

\*完成用户提出的I/O请求，

\*提高I/O速率，--速度上要快

\*提高设备的利用率，--时间上要充分利用

\*方便高层的进程使用这些设备。

有六个基本功能：

隐藏物理设备的细节，与设备的无关性，提高处理机和I/O设备的利用率，提高处理机和I/O设备的利用率，对I/O设备进行控制，确保对设备的正确共享和错误处理。

2)  I/O系统的上、下接口  
　　(1)  I/O系统接口。

向上层提供I/O命令，方便用户。

用户可以通过库函数、系统调用使用。  
(2) 软件/硬件(RW/HW)接口（下）。

　 其上：I/O系统（重点）；其下：设备控制器。

3)  I/O系统的分层  
　　三个层次：  
　　(1) 中断处理程序。（§6.3 ）

与硬件交互

中断请求🡪中断处理🡪返回断点

(2) 设备驱动程序。（§6.4 ）

进程与设备控制器之间的通信程序。

功能：将上层的I/O命令置换为设备I/O命令与参数。

来源：由厂家提供。  
(3) 设备独立性/无关性软件。 （§6.5 ）

I/O软件独立于具体的物理设备----可扩展、可更新

内容：设备命名、设备分配等

4）I/O系统接口

在I/O系统与高层之间的接口中，根据设备类型的不同，又进一步分为若干个接口。包括：块设备接口、流设备接口和网络接口。

5）I/O设备和设备控制器

I/O设备：一般是由执行I/O操作的机械部分1和执行控制I/O的电子部件2组成。通常将这两部分分开：

执行I/O操作的机械部分，叫I/O设备。

而执行控制I/O操作的电子部件，叫设备控制器或适配器。

1. I/O设备的类型

块设备、字符设备；

独占、共享；

还可以：

1. 按使用特性分类

2) 按传输速率分类

第七章

1. 程序和数据以文件的形式保留在外存中
2. **文件的定义属性：**文件是指由创建者所定义的、具有文件名的一组相关信息的集合，可分为有机构文件和无结构文件。

文件的属性包括：文件类型、文件长度、文件的物理位置、文件的建立时间。

1. **文件、记录和数据项**：

数据项

基本数据项：可命名的最小逻辑单位/字段

组合数据项：由若干基本数据项组成

记录

一组相关数据项的集合

关键字：能唯一地标识出记录的基本/组合数据项

文件

具有文件名的一组相关信息的集合，可分为有结构文件和无结构文件（流式文件）

1. **文件类型**：

用途：系统文件、用户文件、库文件

文件中的数据形式：源文件、目标文件、可执行文件

存取控制：只执行文件、只读文件、读写文件

物理安排：顺序文件、链接文件、索引文件

1. 文件系统模型

概念：文件和对文件进行操纵和管理的软件集合

三个层：文件(对象及属性)文件操作文件访问

对象及属性：文件、目录、磁盘存储空间

文件系统接口：命令接口、程序接口

文件操作基本操作：创建、删除、读、写、截断、设置文件的读写位置、打开关闭操作

1. **文件逻辑结构**：

概念：用户所能观察和访问到的文件的数据结构组织，独立于物理特性，容易检索和修改。

无论是逻辑还是物理结构，都会影响到文件的检索速度

逻辑结构类型：

有结构文件：定长记录、变长记录

※记录的组织形式：顺序文件、索引文件、顺序索引文件

无结构文件：流式文件，以字节为单位，利用读/写指针进行访问

记录式文件、流式文件

1. **顺序文件**

逻辑记录排序：按记录时间排：串结构

按关键字排：顺序结构

后一种情况更有利于提高查询速度。如可用折半查找法等

对顺序文件的读/写操作

定长记录：易于定位，甚至可随机读取。

变长记录：不易定位，只能顺序读取。

最佳适用场合是在对诸记录进行批量存取时。

批量存取时对顺序文件的存取速率是所有逻辑文件中最高的；只有顺序文件能存储在磁带上，并能有效地工作。

在交互应用场合，顺序文件表现出来的性能很差；如果想增加或删除一个记录都比较困难。

1. **索引文件**

由变长记录组成的顺序文件不容易直接存取，因此，为其建立一有序的索引表，对索引采用折半查找，速度更快

特点：提高了速度，增加了存储开销——放索引文件

增、删记录时，对索引表作相应的修改

1. **索引顺序文件**

将顺序文件中若干记录分为一组，每组的第一项在索引表中占一项

1. 外存分配方法：

连续分配：每个文件分配一组相邻盘块

特点：简单

链接分配：文件离散地分配于各盘块中，以提高外存利用率，文件长度可变，易于增删，只能顺序存取

隐式链接：文件目录表中有start块号，每块中有下一块号。

特点：只适合于顺序访问，对随机访问效率低，可靠性差。

显式链接：把用于链接的指针显式存放在内存的一张表中，查找在内存中进行

索引分配

单级索引：为每个文件分配一个索引块

特点：

（1）文件较大时有利。文件较小时浪费外存空间（还需为小文件建索引块）

（2）当文件较大时，索引块太多，查找速度减慢

解决：当索引太大时，则需建立多级索引

多级索引

1. **文件目录**：文件控制块的有序集合

**功能/要求**：

（1）按名存取；

（2）提高检索速度；

（3）文件共享；

（4）允许文件重名。

单级目录结构

两级目录结构

树型目录结构

1. **文件存储空间管理**

空闲表法

空闲链表法

空闲盘块链

空闲盘区链

位示图法

成组链接法

第八章

* + - 1. 外存的组织结构

常用的外存组织方式有：

1. 连续组织方式，分配连续磁盘空间，产生顺序物理文件结构。
2. 链接组织方式，分配不连续磁盘空间（链式分配），产生链接式物理文件结构。

3） 索引组织方式，通过索引的方式分配磁盘空间，产生索引物理文件结构。

连续组织方式的主要优缺点：

1. 优点：

\*易实现

从目录中找到第一个盘块，顺序读写即可。

额外的好处：对定长记录的文件可随机存取(不同链式文件)。

即：访问方式 顺序+随机

\*访问速度快

连续的盘块 盘块位于相同或相邻的磁道上，磁头移动距离（最）短，访问速度最快。

1. 缺点：

\*要求分配连续的存储空间（碎片🡪利用率低🡪紧缩🡪费时）

\*要求事先知道文件的长度（此工作交给了应用程序或用户，不能确保准确的文件长度）。

\*不便插入、删除记录：连续记录 顺序存储（第4点）

顺序存储的通病（数据结构）

\*不便文件的动态增长：同上点，还是顺序存储的问题

结论：空间换时间



* + - 1. 链接组织方式

优点：

(1) 消除了磁盘的外部碎片，提高了外存的利用率。  
(2) 对插入、删除和修改记录都非常容易（链表特点）。  
(3) 能适应文件的动态增长，无需事先知道文件的大小。

链接方式又可以分为隐式链接和显示链接

隐式链接：

仅适合顺序访问🡪*随机访问*速度低。

可靠性差（某盘块出问题🡪指针出问题🡪链表断开🡪 文件无法访问）

指针占用了存储空间。

解决检索速度（及指针占用空间）问题

（i）分配盘块以簇为单位(几个盘块)

（ii）每个簇作为链表的一个结点 减少指针数目

（iii）检索以簇为单位 提高检索速度



\*隐式链接分配

每个链表的头指针在FCB中，指出文件的第一个盘块号。

链表中每个结点的值是各物理盘块号，指针指向包含下一物理盘块号的结点。指针显式地存放在该链表中。



* + - 1. 索引组织方式

1. 单级索引分配

链接分配方式虽然解决了连续分配方式所存在的问题， 但又出现了另外两个问题， 即：

(1) 不能支持高效的直接存取（速度慢，查找时间长）。

　　对于大文件，盘块多、链表长。对文件存取时，须在FAT中顺序地查找许多盘块号；

(2) FAT需占用较大的内存空间，（占用空间大）。

　　查找文件时，因为在FAT表中，盘块号是离散分布的，所以只有将整个FAT调入内存，才能在FAT中找到一个文件的所有盘块号，但整个FAT会占用过多内存。

　　所以，最好不要将整个FAT调入内存。事实上，要打开某个文件，只需要把该文件的盘块号调入内存即可，这就引出了文件的索引组织方式。

索引分配方法：建立文件时，

　　１. 为文件分配一个索引块/表，块中记录了文件的所有盘块号。

　　２. 在文件的目录项(FCB)中，填入指向该索引块的指针。

优点：

支持直接访问。可以直接在索引块中找到第i个盘块的盘块号。

也不会产生外部碎片。因为文件分配的盘块，在整个磁盘中，仍然是离散的(可变分区管理中，连续分配存储，当两个分区之间的空隙太小时，就会用不着----碎片)。

缺点：

对于小文件（≦数十个盘块），索引表占用了较多的存储空间🡪浪费了磁盘空间。

****

* + - 1. 文件存储空间的管理

1) 空闲表法  
　　是文件的连续分配方式，它为每个文件分配一块连续的存储空间。它与内存的动态分配方式类似。

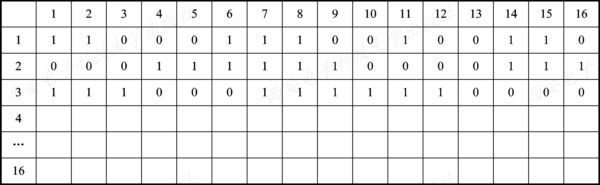
2）空闲链表发

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，可以把链表分为两种形式：空闲盘块链和空闲盘区链。

* + - 1. 位示图法

1）位示图

1. 盘块的表示：用二进制的一位来表示盘中一个盘块的使用情况。
2. 0：盘块空闲；1：已分配(相反也可)。
3. 位示图：磁盘上的所有盘块都有一个二进制位与之对应，由所有盘块所对应的位构成一个集合，称为位示图。



2）盘块的分配

根据位示图进行盘块分配时，可分三步进行：  
(1) 顺序扫描位示图，找出一个或一组值为“0”的二进制位。  
(2) 将这些位转换成与之相应的盘块号b。

b = n(i - 1) + j b = i\*n + j (行列号从0开始)  
式中，n代表每行的位数。  
(3) 修改位示图，令map[i, j] = 1。

3. 盘块的回收

(1) 将回收盘块号转换成位示图中的行号和列号

i = (b - 1)/ n + 1 i=b/n  
 j = (b - 1) MOD n + 1 j=b MOD n   
(2) 修改位示图。令map[i, j] = 0