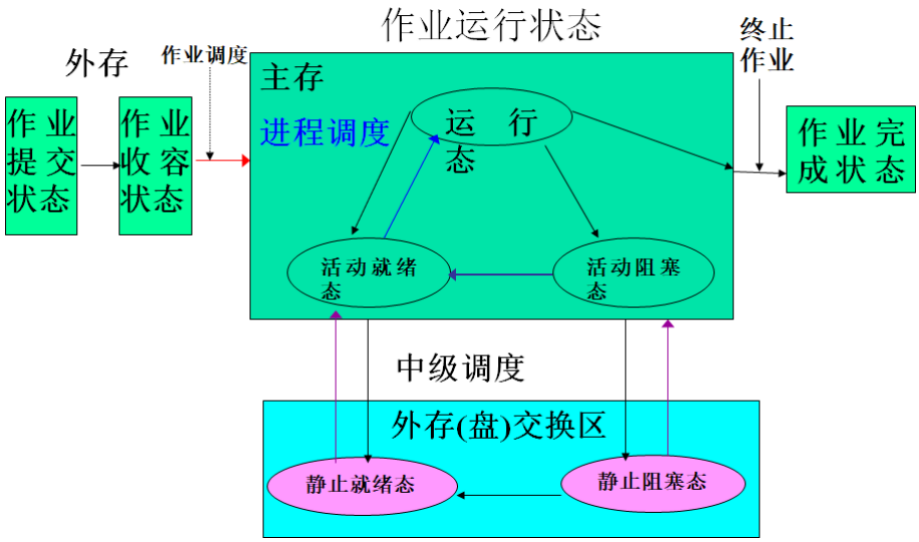
3. CPU调度

3.1. CPU的三级调度



3.1.1. 作业调度/高级调度/宏观调度/长程调度

1. 内容：选取外存中后备状态的作业→装入内存/IO后建立进程→进程就绪
2. 特点：效率低几分钟一次；仅在通用/批处理OS上才有作业调度，决定了有多少程序在多道运行
3. 核心问题：多少作业进内存？(由规模速度决定)；哪些作业进内存？(先进外存的先进内存/短作业优先等等)

3.1.2. 中级调度/交换调度

1. 目的：提高内存利用率/系统吞吐
2. 内容：把内存中阻塞进程交换到外存对换区(挂起)，必要时再调入内存

3.1.3. 低级调度/进程调度/微观调度/短程调度

从就绪队列→CPU执行，特点是高频

3.2. 调度性能评价指标(调度准则)

1. CPU利用率大：CPU被工作占用时间/总时间
2. 系统吞吐大：单位时间CPU完成作业数(但长作业会降低吞吐)
3. 响应时间短：**用户提交请求到系统首次做出响应的时间**
4. 周转时间：作业从提交到完成耗时
    1. 平均周转时间：是指多个作业(如n个作业)周转时间的平均值
    2. 带权周转时间：作业周转时间与作业实际运行时间(服务时间)的比
5. 等待时间短：**进程在就绪队列中等待被调度的时间**

3.3. 进程调度概述

3.3.1. 进程调度的功能

1️⃣记录所有进程的状态，进程管理模块会把每个进程的状态记录在其PCB中，组织PCB队列

2️⃣选取就绪进程获得CPU资源，开始执行(先来先服务，时间片)

3️⃣处理器分配：在将程序从执行转为就绪/阻塞前保护CPU现场，然后从转为执行时恢复CPU现场

3.3.2. 进程调度的诱因(时机)

1️⃣运行→等待：运行进程因为IO/阻塞原语等阻塞

2️⃣运行→就绪：(抢占调度)有更高优先级程序要用CPU，(分时系统)时间片用完

3️⃣等待→就绪：(抢占调度)执行完系统调用后返回用户进程

4️⃣终止运行：进程运行结束(正常/出错异常)

3.3.3. 不能调度的情况

1️⃣在处理中断：逻辑上中断处理属于OS，不属于任一进程，时刻不能被剥夺CPU资源

2️⃣程序进入OS kernel程序临界区：此时程序需要独占共享数据，所以要加锁防止其他程序进入，也不可切换

3️⃣需要完全屏蔽中断的原子操作过程：加锁/解锁/中断现场保护(恢复)

3.3.4. 进程调度的方式

突然某个更紧迫的进程需要处理，CPU应该如何分配

1️⃣非抢占方式/不可剥夺方式：优先级高的进程进入就绪队列，也要排队(等目前进程结束/阻塞后再执行)，实现简单开销小但是实时性差

2️⃣抢占方式/剥夺方式：进程进入就绪队列，可以插队(立即暂现在进程去执行优先级高的进程)

3️⃣抢占原则：什么进程可以插队？

1. 时间片原则(分时系统)：用完一个时间片后，停止目前程序运行并重新调度
2. 优先权原则：优先级高的进入队列，停止目前的进程，去执行优先级更高的进程
3. 短作业优先：新到达作业比执行作业明显短时，停止目前的进程，去执行优先级更高的进程

3.3.5. 分派程序

1️⃣定义：OS的一部分，负责按照某种策略(优先级/轮转法)选一个就绪进程给CPU，是就绪到执行的最后一步

2️⃣工作原理：调度器选好进程→分派程序上下文切换(保存前一进程状态/加载下个进程上下文到CPU)

3️⃣分派延迟：分派程序终结上个进程—[分派延迟]—→启动另一个进程

4️⃣如何降低延迟：系统调用可抢占(确保高优先级进程快速响应)+

3.4. 常见调度算法(如何把CPU分配给进程)

调度算法影响的是等待时间，而不能影响进程真正使用CPU的时间和I/O时间

3.4.1. 先来先服务(作业/进程调度)FCFS

1. 概述：按进程进入就绪队列的先后来分配CPU，非抢占方式(一旦一个进程占据CPU就会一直执行)
2. 特点：有利于长作业不利于短作业；有利于CPU繁忙型不利于I/O繁忙型
3. 适用范围：结合其它调度策略使用，例如优先级调度策略中，用一优先级的进程就采用FCFS

3.4.2. 短作业优先调度算法(作业/进程调度)SJF

1. 概述：从后备作业/进程队列选估计运行时间最短的几个调入内存，非抢占
2. 特点：全部作业同时到达时SJF算法最佳(平均周转时间最短)，对长作业不利(长作业容易等到饿死)，同时也难以实现
3. 最短剩余时间优先调度：SJF的抢占调度版本。当某一进程到达，其时间片比当前执行进程剩余时间片更少时，抢占调度版会强行执行新进程/非抢占调度版本会保持原有进程执行

3.4.3. 优先级调度算法(作业/进程调度)

用整数小/大区分优先级高/低，优先级高的优先分配CPU，所以优先级如何确定?

3.4.3.1. 静态优先级：进程创建时确定后不变

1. 按进程类型：系统进程>用户进程，前台作业>后台作业，I/O繁忙的进程>CPU繁忙进程
2. 按作业需要的资源：进程占据资源(CPU时间/内存大小/IO类型)越多优先级越低
3. 按用户类型和要求：用户收费越高优先级越高(如服务器租用)

3.4.3.2. 动态优先级：优先级随进程推进而改变

1. 进程使用CPU情况：使用时间越长优先级越低
2. 进程就绪等待情况：等的越久优先级越高
3. 进程占用资源情况：占用资源越多优先级高还是低不好说

PS: 一些概念

1. 优先级倒置：低优先级进程占据内核数据，高优先级进程必须等
2. 优先级继承：低优先进程用高优先资源时提升其优先级，但资源回收后其优先级又会回归原样

3.4.4. 时间片轮转调度(进程调度)RR

1. 概述：进程调度程序选择就绪队列中的一个进程，执行一个时间片后将其送入队尾，去执行下一个时间片，以此类推
2. 时间片多长(核心问题)：太长(所有进程一个时间片内完成)则算法就退化为FCFS，如果时间片太短则切换频繁CPU利用率不高，通常为10-100ms

3️⃣时间片大小确定因素

1. 系统响应时间T=N×q=就绪队列中进程数×时间片大小，分时系统对时间片有要求
2. 就绪队列进程数
3. 系统处理能力：计算机速度越快，单位时间处理命令就越多，时间片越小

4️⃣特点：

1. 平均周转时长长于SJF，但是当大多进程在一个时间片内完成，周转时间就会减少
2. 响应时间短于SJF

3.4.5. 高响应比(作业调度)

FCFS+短作业优先

0️⃣

$$响应比=\frac{响应时间}{预估运行时间}=\frac{作业等待时间+作业运行时间}{预估运行时间}\approx\frac{作业等待时间}{作业运行时间}+1$$

1️⃣概述：每次调度时先计算就绪队列中每个作业响应比，响应比高的优先级高

2️⃣特点：对短作业有利(预估运行时间短)+兼顾长作业(等足够长就优先级高了)，但计算响应比增加了开销

3️⃣与其他调度类型相比

1. 若等待时间相同，则作业越短，运行时间越短，响应比越大，优先级越高，等于SJF
2. 若运行时间相同，则先来的进程等的时间长，优先级高，等于FCFS

3.4.6. 多级队列(进程调度)

1️⃣含义：将进程按照类型/优先级/占用资源分类，每类进程弄一个就绪队列(每个进程固定属于一个)，每个队列调度算法不一样

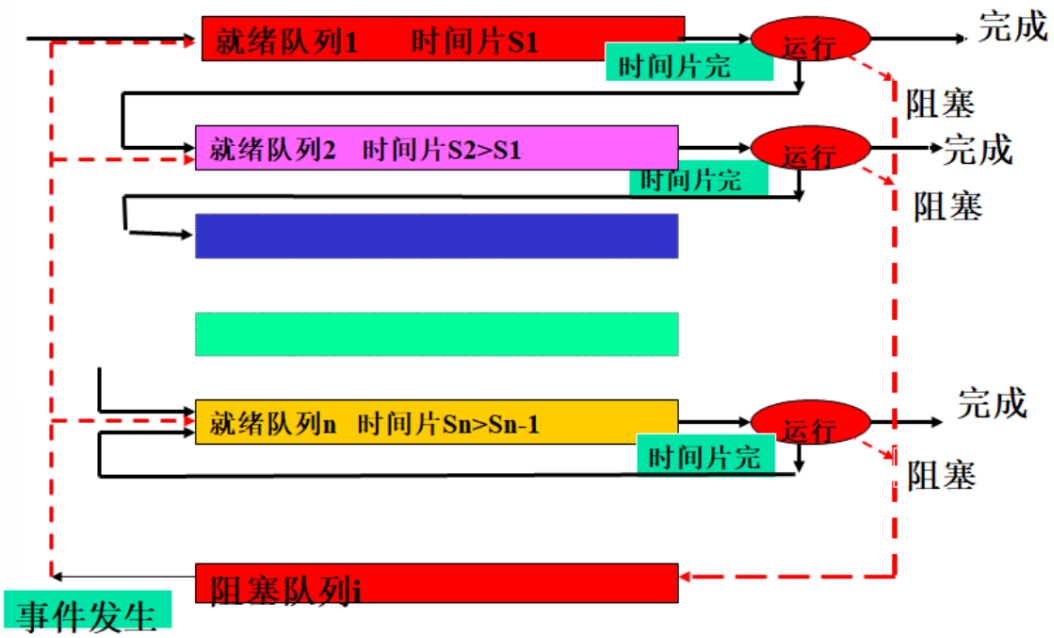
2️⃣实例：就绪队列分为，前台/交互式/时间片调度，后台/批处理/FCFS

3️⃣队列层级的调度

1. 队列优先级：比如，前台运行完后再运行后台
2. 给队列时间片：给每个队列一个时间片，80%时间执行前台，20%时间执行后台

3.4.7. 多级反馈队列(进程调度)

时间片轮转调度+优先级调度



1️⃣概述：有多个就绪队列，每个队列有优先级，各自按时间片轮转，调度允许进程跨队列移动

2️⃣关于时间片的长度：**优先级越高的队列时间片越小**，通常第i+1队列的时间片是第i队列时间片的两倍

3️⃣一个进程要放在什么优先级的队列中？(优先级从上到下减小)

1. 新进程先进入第一队列末尾(FCFS调度)，随之被执行一个时间片，若执行完就退出
2. 若一时间内未执行完，就把他丢到第二个队列尾(FCFS调度)，以此类推
3. 若进程到了优先级最低队列都没执行完，就只有重新塞回本队列尾了

PS: 阻塞进程的优先级低于以上一切队列

4️⃣按什么顺序执行优先级不同的队列?

1. 优先执行优先级高的队列：只当第一队列空才执行第二队列进程
2. 抢占：执行第二队列进程时，有一进程插入第一队列，则转而执行插入进程(第二队列原来的进程丢到队尾)

3.5. 多处理器调度

1️⃣对称多CPU(SMP)：每个CPU都有自己调度方案，他们互斥访问公共就绪队列，领取进程执行

2️⃣非对称多CPU(AMP)：只有一个CPU能管理OS资源，其余执行用户级任务，数据共享更容易

PS：进程从一个CPU到另一个CPU需要更新Cache所以开销大，SMP不允许进程迁移到另一个CPU

3.6. 实时调度：基于优先级+抢占

1️⃣目的：完成实时任务()而分配CPU的调度方法

2️⃣实时任务：硬实时(规定时间内必须执行完)+软实时(允许偶尔的延迟)

3️⃣实现：基于优先级调度，任何时候实时进程优先级最高，调度延时小