调度的基本概念

调度层次

- 高级调度：
  - 作业调度、宏观调度或长程调度
  - 主要任务
    ① 按一定的原则**从外存上**处于后备状态的作业中选择一个或多个作业，**决定哪个进程进入系统，或者说**选择哪些进程可以进入就绪队列。
    ② 为选中作业分配内存、输入/输出设备等必要的资源，并建立相应的进程，以使该作业具有获得竞争处理机的权利。
  - 作业调度的运行频率较低，通常为几分钟一次。
  - 高级调度决定着多道程序的道数问题

低级调度

- 低级调度
  - 进程调度、微观调度或短程调度
  - 主要任务
    ① 按照某种原则决定就绪队列中的哪个进程/内核级线程能获得处理器，并将处理机出让给它进行工作。
  - 进程调度程序是操作系统最为核心的部分，进程调度策略的优劣直接影响到整个系统的性能。
  - 包括：剥夺式(preemptive)/非剥夺式，或抢占/非抢占
  - 进程调度的运行频率很高，一般几十毫秒要运行一次。

中级调度

- 中级调度
  - 负载均衡调度，中程调度，交换调度
  - 主要任务
    - ① 将内存中暂时不用的信息移到外存，以腾出空间给内存中的进程使用，
    - ② 将需要的信息从外存读入内存。
  - 中级调度决定了**主存储器中所能容纳的进程数**，这些进程将允许参与竞争处理器资源，提高内存利用率和系统吞吐量
  - 中级调度根据存储资源量和进程的当前状态来决定辅存和主存中进程的对换，运行频率介于两种调度之间。

调度性能评价标准

- 资源利用率

  ① **资源利用率**

  - CPU利用率=CPU有效工作时间/CPU总的运行时间
- 响应时间
- 系统吞吐率
- 公平性
- 周转时间

  周转时间=作业完成时间-作业提交时间

  带权周转时间=作业周转时间/作业实际运行时间，这个作业实际运行时间就是CPU时间

③ 周转时间

- **从作业提交**给系统开始，到**作业完成为**止的时间间隔称**作业周转时间**，应使作业周转时间或平均作业周转时间尽可能短。

- 这是衡量批处理系统的一个重要指标

- **周转时间**
  - 作业的**周转时间**是指从作业提交到作业完成之间的时间间隔。周转时间是所有时间段之和，包括等待进入内存、在就绪队列中等待、在CPU上执行和I/O执行的时间。这段时间间隔包括了，作业等待、挂起等。
    - $T_i = Te_i - Ts_i$

- **平均周转时间**
  - 是指多个作业的周转时间的平均值。 n 个作业的平均周转时间：
    - $T = (T_1 + T_2 + \ldots + T_n) / n$ （T_i为作业 i 的周转时间）

- **带权周转时间**
  - **带权周转时间**是指作业周转时间与作业实际运行时间的**比值**，W_i。
  - **注意**：这里作业实际运行时间是指在CPU中的时间，不包括阻塞、挂起等时间
- **平均带权周转时间**
  - 是指多个作业的带权周转时间的平均值。 n 个作业的平均带权周转时间：
  - $W = (W_1 + W_2 + \ldots + W_n) / n$ （W_i为作业 i 的带权周转时间）

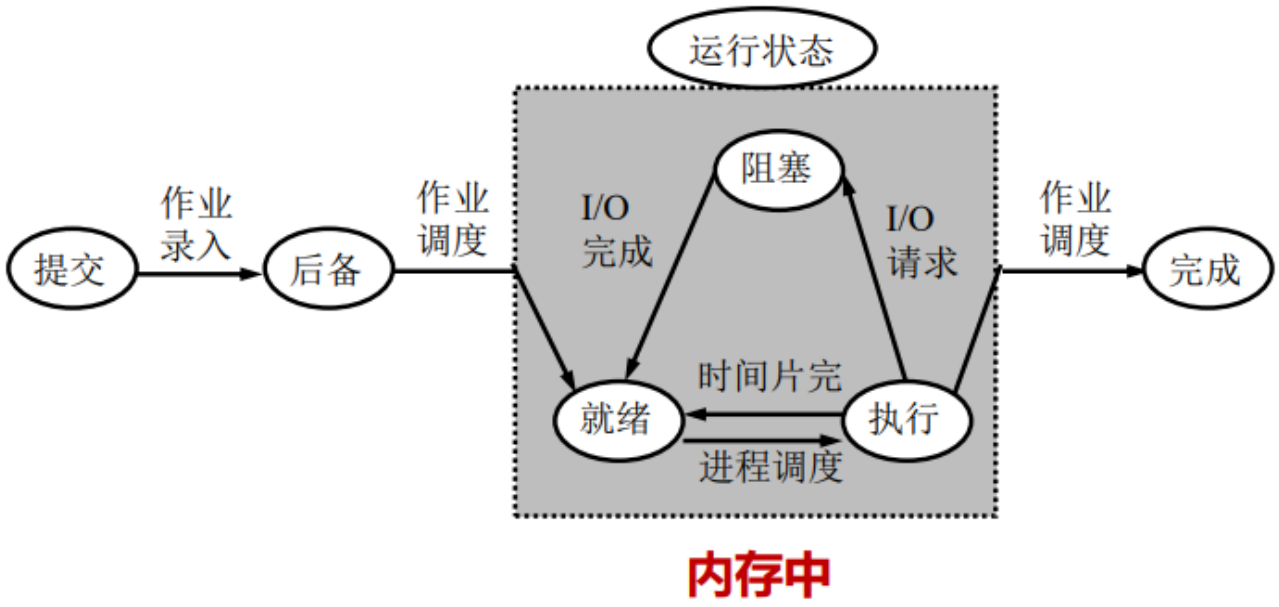
作业、进程和线程的调度

一篇文章搞懂作业与进程的区别

作业

- 作业状态转化

- 作业从提交到完成要经历四种状态：
  ① 提交状态：用户作业由输入设备，向**系统外存**输入时作业所处的状态。
  ② 后备状态：作业输入到外存后，系统为其建立了作业控制块，并把它插入到后备作业队列中等待调度运行。
  ③ 运行状态：作业在**内存中，包含占有CPU&没占有CPU**
  ④ 完成状态：作业正常或异常结束，但作业占有的资源还未被系统全部回收。



- 作业控制块

- 为管理作业，系统设置了作业控制块。系统通过JCB感知作业的存在，JCB是作业存在的唯一标志。
- 通常作业控制块中包括的主要内容有：
  - 资源要求。
  - 资源使用情况。
  - 作业的控制方式、类型和优先级等。
  - 作业名、作业状态。

进程

- 抢占与非抢占

抢占方式

- 抢占方式：又称剥夺方式、可剥夺方式。这种调度方式是指允许调度程序根据某种原则去停止正在执行的进程，将已分配给该进程的处理机重新分配给其他进程。
- 抢占原则有：
  - 优先权：高优先级抢占/剥夺低优先级
  - 时间片：时间片用完，剥夺CPU使用权

非抢占方式

- 非抢占方式：又称非剥夺方式、不可剥夺方式、不可抢占方式。
  - 指一旦将处理机分配给某进程后，便让该进程**一直执行**，直到该进程完成或发生某事件而进入阻塞状态，才把处理机分配给其他进程。
- 非抢占方式中引起进程调度的因素有：
  - 进程结束
  - 因某种原因而阻塞
  - 执行同步原语等。
- 特点：简单，系统开销小，但无法处理紧急任务。

- 进程调度的原因

引起进程调度的原因

当一个进程：

- 从运行态切换到等待状态
  - I/O请求，调用wait等
- 从运行状态切换到就绪状态
  - 如发生中断
- 从等待状态切换到就绪状态
  - 如系统调用或者中断返回
- 进程终止

线程

没什么东西，看看PPT就行了

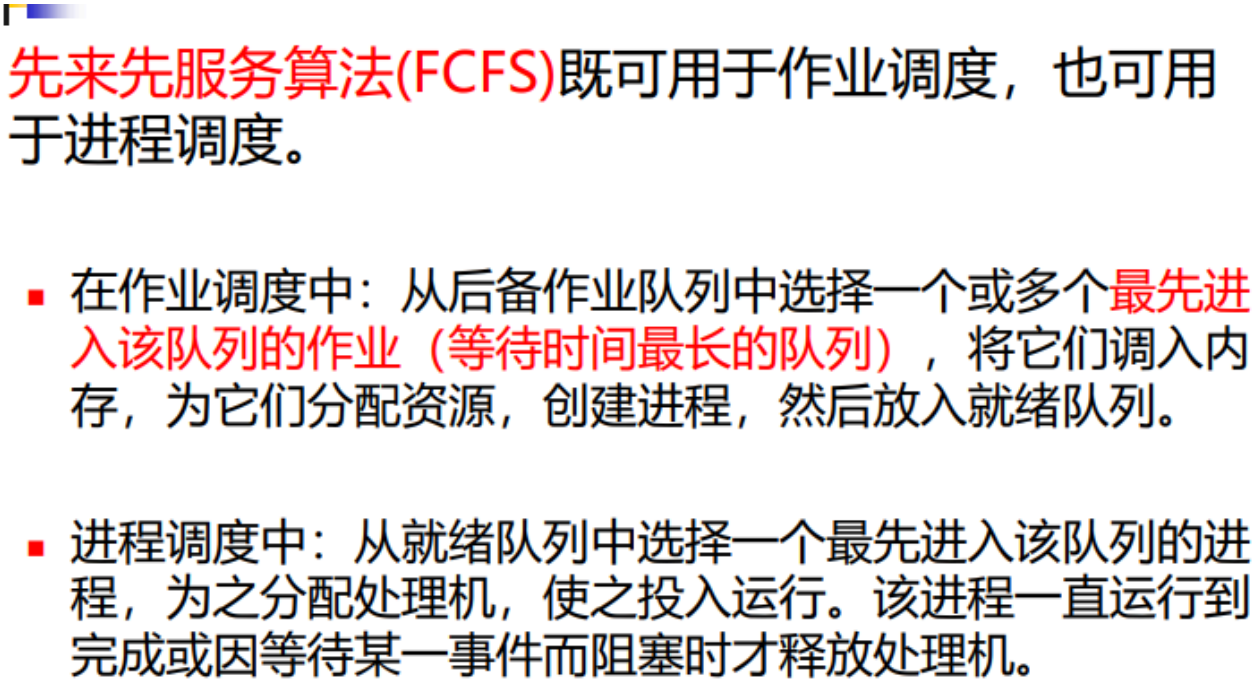
调度算法

这篇文章讲的很详细
具体例题看PPT中即可

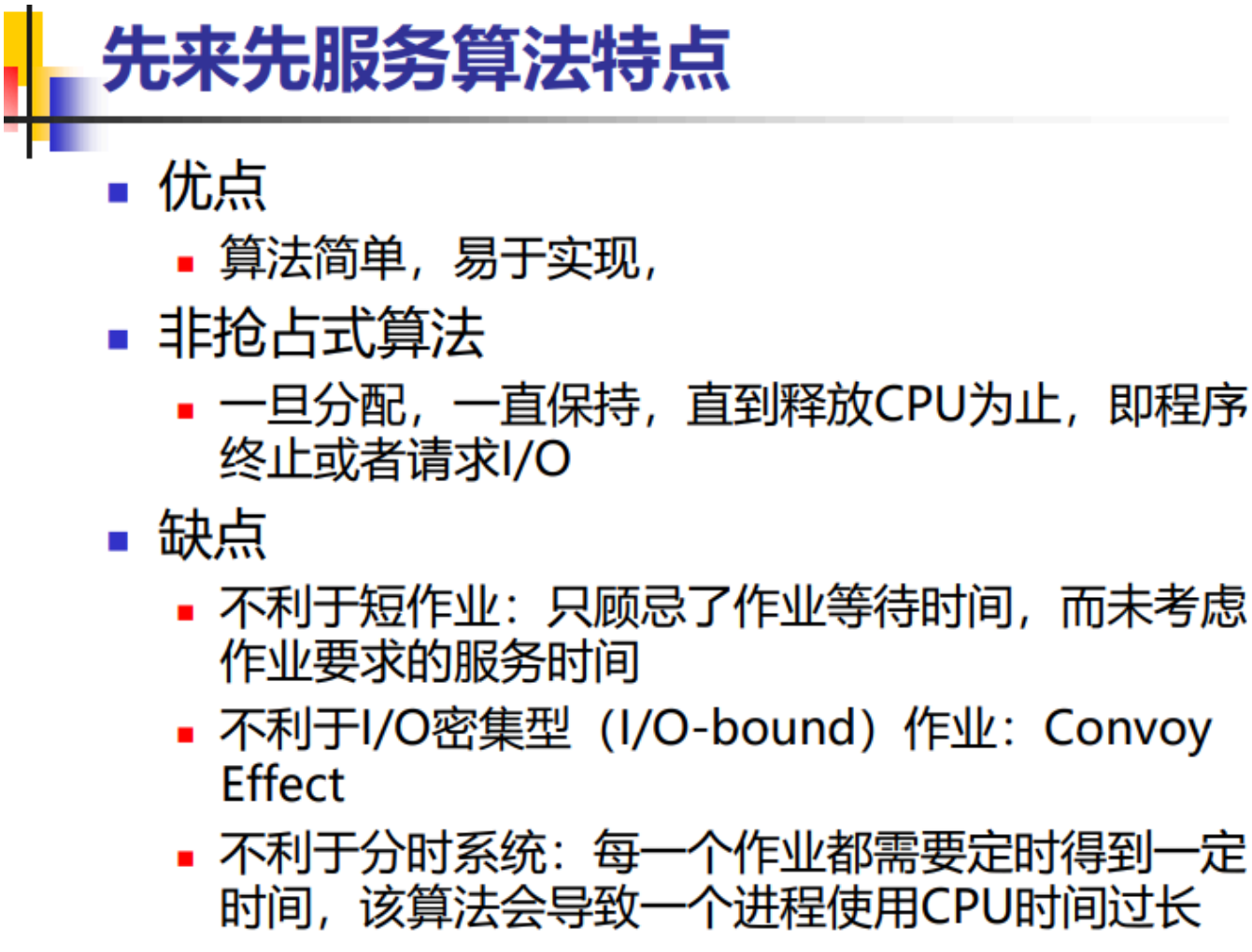
先来先服务算法

这个算法就是字面意思

- 具体内容



- 特点



短作业优先调度算法

实际运行时间（CPU时间）短的先运行

- 具体内容

3.3.2 短作业优先调度算法

- 同时适合作业和进程调度
- 在作业调度中，从后备队列中选择一个或多个估计运行时间最短的作业，将它们调入内存运行。
- 在进程调度中，从就绪队列中选择一个估计运行时间最短的进程，为之分配处理机，使之投入运行。该进程一直运行到完成或因等待某一事件而阻塞时才释放处理机。

- 特点

短作业优先调度算法的特点

- 算法调度性能较好，

如上例中，	先来先服务	短作业优先
平均周转时间	2.8	2.45
平均带权周转时间	5.25	3.85

- 但对长作业不利，未考虑作业的紧迫程度，运行时间为估计。
- 当SFJ用于进程调度时：
  - 需要计算的是进程/线程在下一个CPU周期长度，而不是整个进程/线程的用时长度。
  - 实则为“最短下次CPU用时优先”算法(shortest next CPU burst)

最短剩余时间优先算法

当一个新进程进入就绪队列时，若其需要的运行时间比当前运行进程的剩余时间短，则它将抢占CPU。注意这里抢占了CPU

- 具体内容

优先级调度算法

依据优先级来调度

- 具体内容

- 在作业调度中，从后备作业队列中选择若干优先级高的作业调入内存。
- 在进程调度中，将处理机分配给就绪队列中优先级最高的进程。
  - 具有相同优先级的进程按照FCFS调度
  - SJF是一种简单的优先级算法，优先级为下次CPU区间的倒数
- 优先级表示进程的重要性及运行优先性，通常用优先数来衡量。
  - 在某些系统中，优先数越大优先级越高；而在另一些系统中，优先数越大优先级越小。

- 按调度方式对优先级调度算法分类

按调度方式对优先级调度算法分类

- 非抢占式优先级调度算法：
  - 系统一旦将处理机分配给就绪队列中优先级最高的进程后，该进程便一直运行下去，直到完成或因发生某事件使该进程放弃处理机时，系统才将处理机分配给另一个更高优先级的进程。
  - 优先级体现：非抢占算法只是将新的进程，按照优先级，加到就绪队列的头部
- 抢占式优先级调度算法：
  - 将处理机分配给优先级最高的进程，使之运行。在进程运行过程中，一旦出现了另一个优先级更高的进程时，进程调度程序就停止原运行进程，而将处理机分配给新出现的高优先级进程。
  - 优先级体现：优先级高的进程只要就绪，就抢占当前进程

- 优先级的类型

- 静态优先级是在创建进程时确定的，确定之后在整个进程运行期间不再改变。
- 确定依据有：
  - 进程类型：系统，用户
  - 进程对资源的需求：执行时间，资源数量
  - 用户要求：紧迫程度
- 特点：简单易行，系统开销小，但不精确

- 动态优先级是指在创建进程时，根据进程的特点及相关情况确定一个优先级，在进程运行过程中再根据情况的变化调整优先级。

- 确定原则有：占用CPU时间，等待时间。
  - 例：优先数=CPU使用时间/2+基本优先数

- 缺陷

优先级调度算法的问题

- 无穷阻塞/饥饿
  - 超负载计算机系统中，低优先级进程会无穷等待CPU
  - 例：1973年关闭的MIT IBM7094，发现了1967年提交的低优先级进程
- 解决方案：老化aging
  - 对低优先级进程，随着等待时间，逐渐增加其优先级
  - 例：以15分钟递减优先值1（值越低优先级越高），从127开始，不超过32小时，就可以老化为优先级值为0

时间片轮转调度算法

时间片下众生平等，时间片用完了排到队列尾部

- 具体内容

  - 时间片轮转法（Round-robin）：
    - 专门为分时系统设计。
    - 系统将所有就绪进程按到达时间的先后次序排成一个队列，每次调度时把CPU分配给队首进程，并令其执行一个时间片。
    - 当时间片用完时，停止该进程的执行，将它送至就绪队列末尾等待下一次执行，然后再把处理机分配给就绪队列中的新队首进程。
    - 如此不断循环，直至完成为止。

最高响应比优先调度算法

响应比=1+作业等待时间/估计运行时间（CPU时间）

- 具体内容

  - 先来先服务：片面考虑作业等待时间，忽视计算时间
  - 短作业优先：片面考虑计算时间，忽视作业的等待时间

  - 高响应比优先调度算法是对短作业优先调度算法和先来先服务调度算法的一种综合。（非抢占式）

  - 此处响应比的定义：
    响应比 = 作业响应时间/估计运行时间
  - 由于响应时间为作业进入系统后的等待时间加上估计运行时间。因此

    响应比 = 1 + 作业等待时间/估计运行时间

- 特点

  - 在每次调度作业运行时，先计算后备作业队列中每个作业的响应比，然后挑选响应比最高者投入运行。
  - 特点：
    - 有利于短作业——等待时间相同，短作业优先，
    - 考虑等待时间——运行时间相同，等待时间长的作业优先运行。
    - 主要用于作业调度

多级反馈队列调度算法

设计的很巧妙

- 主要内容

- 改进多级队列调度算法的灵活性问题
- 设置多个就绪队列，并为每个队列赋予不同的优先级。第1个队列的优先级最高，第2队列次之，其余队列的优先级逐次降低。
- 每个队列中进程执行的时间片大小也各不相同，进程所在队列的**优先级越高**，其相应的**时间片就越短**。
- 当一个新进程进入系统时，首先将它放入第1个队列的末尾，按**先来先服务**的原则排队等待调度。
- 当轮到该进程执行时，如能在此时间片内完成，便可准备撤离系统；
- 如果它在一个时间片结束时尚未完成，调度程序便**将该进程转入第2队列的末尾**，再同样地按FCFS原则等待调度执行。
- 如此下去，最后一个队列中使用某种调度算法。

- 仅当第1个队列为空时，调度程序才调度第2队列中的进程运行；仅当第1个至第（i - 1）个队列均为空时，才会调度第i个队列中的进程运行。

- 当处理机正在为第i个队列中的某进程服务时，若又有**新进程**进入优先级较高的队列中，则此时新进程将**抢占**正在运行进程的处理机，即由调度程序把正在执行进程放回**第i个队列末尾**，重新将处理机分配给新进程。

- 特点

  - 特点
    - 允许进程在队列之间移动
    - 根据CPU区间的特点区分进程，如果进程使用过多的CPU时间，就会被移动到低优先级队列
  - 多级反馈队列是非常复杂调度程序，通过配置参数来定义最佳的调度程序
    - 队列数量
    - 每个队列的调度算法
    - 用以确定何时升级到最高优先级队列的方法
    - 用以确定何时降级到最低优先级的方法
    - 用以确定进程在需要服务时将会进入哪个队列的方法

公平分享调度算法

- 具体内容

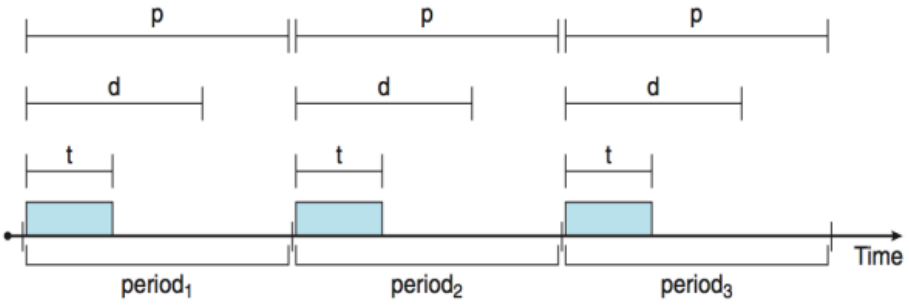
- 公平分享调度算法
  - 动机：
    - 采用时间片轮转(Round-Robin)的情况下，**多个用户，如果所拥有的进程数不同，则该用户拥有CPU的比率不同。**这样造成某些用户的响应比过低，不利于公平原则。
  - 解决
    - 对于不同用户，**基于进程组来分配CPU时间**，其实现思想是对系统中的每个用户赋予某种权值，根据用户权值大小，按比例分配处理机时间。

实时调度

优先级调度

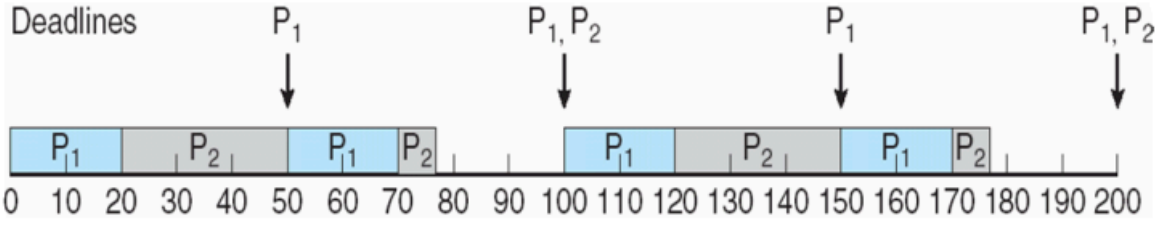
- 具体内容

- 对于实时调度，调度器必须支持抢占的基于优先权的调度
  - 但是仅保证软实时要求。
  - 硬实时要提供截止期限内得到服务的要求
- 这些进程的新特点：周期性的
  - 固定的处理时间 t_i，截止期限 d_i，周期 p
  - $0 \le t \le d \le p$
  - 任务速率 $1/p$
  - 根据进程的截止期限、任务速率来进行分配优先级



单调速率调度

- 具体内容
  - 单调速率调度，Rate Monotonic Scheduling
    - 基于任务速率来分配优先级
    - 更短的周期=更高的优先级
    - 更长的周期=更低的优先级
  - 例：
    - P_1：周期50，处理时间：20，截止期限：下一个周期开始前
    - P_2：周期100，处理时间：35，截止期限：下一个周期开始前
    - P_1被赋予比P_2更高的优先级

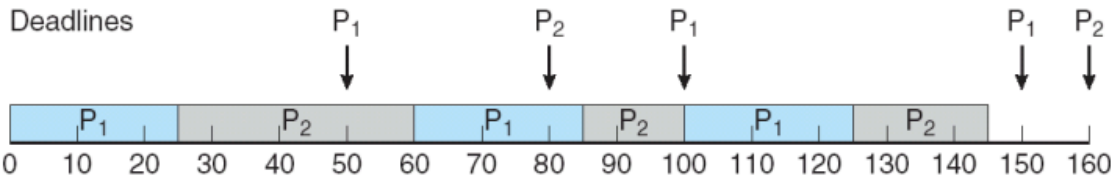


最早截止期限优先调度

因为在该任务下一个周期来之前，应当执行完当任务。为了达到这个目的，把当前任务的截止时间作为优先级的评判，这也就是最早截止期限优先调度

- 具体内容
  比如在时间节点50时，P1的截止时间为100，P2的截止时间为80，因此要继续执行P2
  而在时间节点100时，P1的截至时间为150，P2的截止时间为160，因此直接抢占P2执行P1

- 最早截止期限优先调度：Earliest Deadline First Scheduling (EDF)
  - 根据截至期限动态分配优先级
  - 期限越早：优先级越高
  - P1：周期50，处理时间：25，截止期限：下一个周期开始前
  - P2：周期80，处理时间：35，截止期限：下一个周期开始前



多处理器的调度

见PPT