在设计短程调度程序时使用了各种各样的准则。一些准则与单个用户觉察到的系统行为有关（面向用户），而其他准则查看系统在满足所有用户的需求时的总效率（面向系统）。一些准则与性能的定量度量有关，另一些在本质上是定性的。从用户的角度看，响应时间通常是系统最重要的一个特性；从系统的角度看，吞吐量或处理器利用率是最重要的。

为所有就绪进程的短程调度决策已经开发许多种算法：

- **先来先服务**：选择等待服务时间最长的进程。
- **轮转**：使用时间片限制任何正在运行的进程只能使用一段处理器时间，并在所有就绪进程中轮转。
- **最短进程优先**：选择预期处理时间最短的进程，并且不抢占该进程。
- **最短剩余时间**：选择预期的剩余处理时间最短的进程。当另一个进程就绪时，这个进程可能会被抢占。
- **最高响应比优先**：调度决策基于对归一化周转时间的估计。
- **反馈**：建立一组调度队列，基于每个进程的执行历史和其他一些准则，把它们分配到各个队列中。

调度算法的选择取决于预期的性能和实现的复杂度。

9.5　推荐读物

基本上所有的操作系统教材都涉及了调度。在[KLEI04]和[CONW67]中有对各种调度策略的严格的排队分析。[DOWD93]对各种调度算法有指导性的性能分析。

CONW67 Conway, R.; Maxwell, W.; and Miller, L. *Theroy of Scheduling*. Reading, MA:Addison-Wesley, 1967.Reprinted by Dover Publicatios,2003.

DOWD93 Dowdy, L., and Lowery, C. *P.S to Operating Systems*. Upper Saddle River, NJ: Prentice Hall, 1993.

KLEI04 Kleinrock, L. *Queuing System, Volume Three: Computer Applications*. New York: Wiley, 2004.

9.6　关键术语、复习题和习题

关键术语

到达速率	长程调度程序	服务时间	分派程序
中程调度程序	短程调度程序	指数平均	多级反馈队列
吞吐量	公平共享调度	可预测性	时间片
公平性	驻留时间	周转时间（TAT）	先来先服务（FCFS）
响应时间	利用率	先进先出（FIFO）	轮转
等待时间	调度优先级		

复习题

9.1　简要描述三种类型的处理器调度。

9.2　在交互式操作系统中，通常最重要的性能要求是什么？

9.3　周转时间和响应时间有什么区别？

9.4　对于进程调度，较小的优先级值表示较低的优先级还是较高的优先级？

9.5　抢占式和非抢占式调度有什么区别？

9.6　简单定义 FCFS 调度。

9.7　简单定义轮转调度。

9.8　简单定义最短进程优先调度。

9.9 简单定义最短剩余时间调度。

9.10 简单定义最高响应比优先调度。

9.11 简单定义反馈调度。

习题

9.1 考虑下面的进程集合：

进 程 名	到达时间	处理时间
A	0	3
B	1	5
C	3	2
D	9	5
E	12	5

对这个集合，给出类似于表 9.5 和图 9.5 的分析。

9.2 按习题 9.1 的要求完成下面的集合：

进 程 名	到达时间	处理时间
A	0	1
B	1	9
C	2	1
D	3	9

9.3 证明在非抢占式调度算法中，对于同时到达的批处理作业，SPN 提供了最小平均等待时间。假设调度程序只要有任务就必须执行。

9.4 假设一个进程的每一次瞬间的执行时间（burst-time 模式）为 6，4，6，4，13，13，13，假设最初的猜测值为 10。请画出类似于图 9.9 的图表。

9.5 考虑下面的公式，它可以替代公式（9.3）：

$$S_{n+1} = \alpha T_n + (1-\alpha)S_n$$
$$X_{n+1} = \min[Ubound, \max[Lbound, (\beta S_{n+1})]]$$

其中 *Ubound* 和 *Lbound* 是预先选择的估计值 T 的上限和下限。X_{n+1} 的值用于最短进程优先的算法，并且可以代替 S_{n+1}。α 和 β 有什么功能，它们每个取最大和最小值会产生什么影响？

9.6 在图 9.5 下面的例子中，在控制权限转移到 B 之前，进程 A 运行 2 个时间单元，另外一个场景是在控制权限转移到 B 之前，进程 A 运行 3 个时间单元。在反馈调度算法中，这两种场景的策略有什么不同？

9.7 在一个非抢占的单处理器系统中，在刚刚完成一个作业后的时刻 t，就绪队列中包含三个作业。这些作业分别在时刻 t_1、t_2 和 t_3 处到达，估计执行时间分别为 r_1、r_2 和 r_3。图 9.18 显示了它们的响应比随着时间线性增加。使用这个例子，设计一种响应比调度的变体，称为极小极大响应比调度算法，它可以使给定的一批作业（忽略后来到达的）的最大响应比最小。（提示：首先确定最后调度哪一个作业。）

9.8 证明，对给定的一批作业，上一习题中的最大响应比调度算法能够使它们的最大响应时间最小。（提示：重点研究达到最高响应比的作业，以及在它之前执行的所有作业。考虑同样的作业子集，按任何其他顺序调度，观察最后一个执行的作业的响应比。注意，这个子集现在可能混合了全集中的其他作业。）

9.9 驻留时间 T_r 被定义成一个进程花费在等待和被服务上的总的平均时间。说明对 FIFO，若平均服务时间为 T_s，则有 $T_r=T_s/(1-\rho)$，其中 ρ 为利用率。

9.10 假设一个处理器被就绪队列中的所有进程以无限的速度多路复用，且没有任何额外的开销。（这是一个在就绪进程中使用轮转调度的理想模型，时间片相对于平均服务时间非常小。）说明对来自一个指数服务时间的无限源的泊松输入，一个进程的平均响应时间 R_x 和服务时间 x 由式 $R_x=x/(1-\rho)$ 给出（提示：复习在 WilliamStallings.com/StudentSupport.html 上排队分析文档中的基本排队公式。然后考虑当一个给定进程到达时，系统中等待项的数目 w）。

9.11　对某一系统的度量结果表明：在一个进程由于 I/O 而阻塞前，其平均运行时间为 T。在时间段 T 中，进程切换的开销为 S。换句话说，T 包括了在 CPU 上进程切换的开销，且 S<T。如果采用轮转调度策略，时间配额为 Q，针对以下三种状况，给出相应的 CPU 效率计算公式。CPU 效率是指 CPU 执行用户代码（不是进程切换时间）的部分时间与总时间的之比。

a）Q>T

b）Q=S

c）Q 接近 0

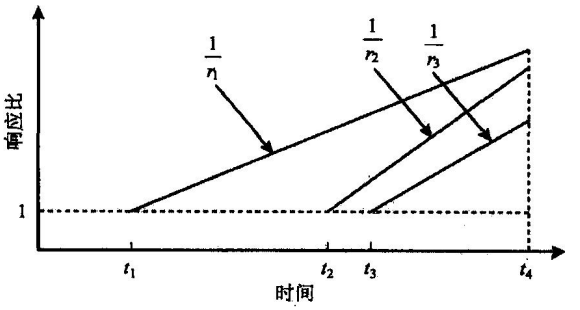

图 9.18　响应比关于时间的函数

9.12　在排队系统中，新作业在得到服务之前必须等待一小段时间。当一个作业等待时，它的优先级从 0 开始以速度 α 线性增加。一个作业一直等待到它的优先级达到正在接收服务的作业的优先级，然后它开始与其他正在接收服务的作业使用轮转法平等地共享处理器，与此同时，它的优先级继续以比较慢的速度 β 增长。这个算法称做自私轮转法，因为在接收服务的作业试图（徒然）通过不断地增加它的优先级来垄断处理器。使用图 9.19 说明服务时间为 x 的一个作业的平均响应时间 R_x 为

$$R_x = \frac{s}{1-\rho} + \frac{x-s}{1-\rho'}$$

$$\rho = \lambda s \qquad \rho' = \rho\left(1-\frac{\beta}{\alpha}\right) \qquad 0 \leqslant \beta < \alpha$$

假设到达时间和服务时间分别以均值 $1/\lambda$ 和 s 呈指数分布（提示：分别考虑整个系统和两个子系统）。

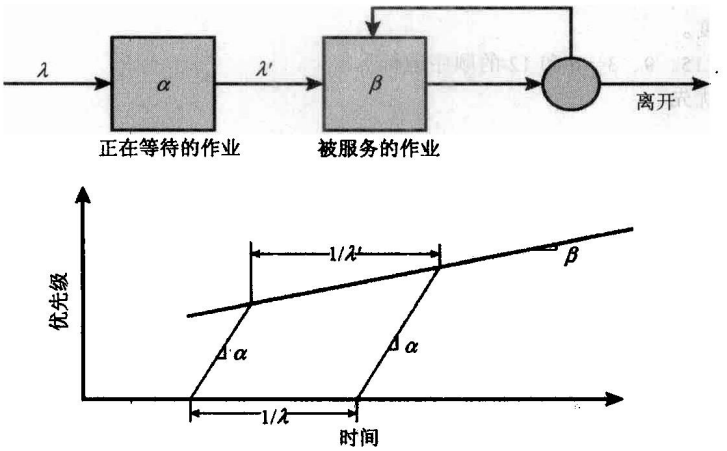

图 9.19　自私轮转法

9.13　四个进程的需求特征如下表所示。这些进程准备在一个仅有简单 I/O 处理设备的单处理机上执行。在该系统中，分派一个进程到 CPU 上执行的开销是 1 个时间单位，将进程从 CPU 上移除的开销为 0。在 I/O 设备上操作一个进程的开销为 0，该系统采用轮转调度策略，时间配额为 5 个单位。

a）两个时间关系图（一个关于 CPU，一个关于 I/O 设备），描述了执行顺序

b）TR = 平均进程周转时间

c）CPU-UTIL = CPU 利用率（CPU 处于繁忙的时间/总时间）

d）I/O-UTIL = I/O 设备利用率（I/O 设备处于繁忙的时间/总时间）

假设每一个 I/O 请求（如果有 I/O 请求）都在 CPU 运行之前执行，并假设 I/O 操作从不被抢占。

进 程 号	到达时间	CPU 进发时间 1	I/O 进发时间	CPU 进发时间 2
P1	0	5	5	2
P2	$3-\varepsilon$	2	22	2
P3	$7-\varepsilon$	8	0	0
P4	$25-\varepsilon$	9	2	1

9.14 考虑一个有 5 个优先级队列的多级反馈调度策略，每一个优先级队列采用时间片轮转策略，时间片大小为 3 个单位，每一优先级的最大执行时间为 2 个时间片（或 6 个单位）。每一个需要 CPU 的作业从最高优先级开始，随着 CPU 时间的增长，优先级降低。系统总是把 CPU 分配给最高优先级的作业。

a）对 CPU 密集型进程，这一调度策略是否能很好地工作？请解释理由。

b）这一调度算法对 I/O 密集型进程是否有好处？请解释理由。

c）会出现饥饿现象吗？如果会，怎样对该策略进行修改以避免饥饿现象？

9.15 在基于优先级的进程调度中，如果当前没有其他优先级更高的进程处于就绪状态，调度程序将把控制权交给一个特定的进程。假设在进行进程调度决策时没有使用其他信息，还假设进程的优先级是在进程被创建时建立的，并且不会改变。在这样的系统中，为什么使用 Dekker 方法（见 A.1 节）解决互斥问题是非常“危险”的？通过说明会发生什么不希望发生的事件和如何发生这种事件来解释该问题。

9.16 5 个批作业，从 A 到 E，同时到达计算机中心。它们的估计运行时间分别为 15、9、3、6 和 12 分钟，它们的优先级（外部定义）分别为 6、3、7、9 和 4（值越小，表示的优先级越高）。对下面的每种调度算法，确定每个进程的周转时间和所有作业的平均周转时间（忽略进程切换的开销），并解释是如何得到这个结果的。对于最后三种情况，假设一次只有一个作业运行直到它结束，并且所有作业都完全是处理器密集型的。

a）时间片为 1 分钟的轮转法。

b）优先级调度。

c）FCFS（按 15、9、3、6 和 12 的顺序运行）。

d）最短作业优先。

附录 9A　响应时间

响应时间是系统对某个给定的输入做出反应所需要的时间。在一个交互事务中，响应时间可以定义为从用户最后一次击键到计算机开始显示结果之间的时间。对不同类型的应用程序，定义有略微的不同。一般而言，它是指系统用于响应执行某个特定任务的请求所花费的时间。

理想情况下，用户希望对任何应用程序的响应时间都非常短。但是，更短的响应时间通常总是需要更大的花费，这个花费来自两方面：

- **计算机处理能力**：处理器速度越快，响应时间越短。当然，处理能力的增加意味着成本的增加。
- **竞争要求**：对一些进程提供快速的响应时间必然不利于另外一些进程。

因此，一个给定级别的响应时间值必须通过达到这个响应时间的花费来评估。

基于［MART88］，表 9.7 列出了 6 种常用的响应时间范围。当要求响应时间小于 1 秒时，就会面临设计上的困难。响应时间小于 1 秒的要求是由控制正在运行的外部活动或以某种方式与其进行交互的系统产生的，如一个汇编行；这时的要求是很直接的。当考虑人机交互时，例如数据输入应用，就处于会话响应时间的范围。在这种情况下，仍然存在对短响应时间的要求，但是可能很难估计可以接受的时间长度。

表 9.7　响应时间的范围

范　围	说　明
大于 15 秒	这排除了会话交互。对某些类型的应用程序，某些类型的用户可能愿意坐在终端前等待 15 秒以上，以得到一个简单查询的回答。但是，对比较忙的用户而言，等待 15 秒是无法忍受的。如果发生这类延迟，系统应该设计成主用户转去做其他的活动，过一段时间再来请求该响应