能够强制执行该策略。

另一个共同的目标是保持系统的所有部分尽可能忙碌。如果CPU和所有I/O设备能够始终运行，那么相对于让某些部件空转而言，每秒钟就可以完成更多的工作。例如，在批处理系统中，调度程序控制哪个作业调入内存运行。在内存中既有一些CPU密集型进程又有一些I/O密集型进程是一个较好的想法，好于先调入和运行所有的CPU密集型作业，然后在它们完成之后再调入和运行所有I/O密集型作业的做法。如果使用后面一种策略，在CPU密集型进程运行时，它们就要竞争CPU，而磁盘却在空转。稍后，当I/O密集型作业来了之后，它们要为磁盘而竞争，而CPU又空转了。显然，通过对进程的仔细组合，可以保持整个系统运行得更好一些。

运行大量批处理作业的大型计算中心的管理者们为了掌握其系统的工作状态，通常检查三个指标：吞吐量、周转时间以及CPU利用率。吞吐量（throughout）是系统每小时完成的作业数量。把所有的因素考虑进去之后，每小时完成50个作业好于每小时完成40个作业。周转时间（turnaround time）是指从一个批处理作业提交时刻开始直到该作业完成时刻为止的统计平均时间。该数据度量了用户要得到输出所需的平均等待时间。其规则是：小就是好的。

能够使吞吐量最大化的调度算法不一定就有最小的周转时间。例如，对于确定的短作业和长作业的一个组合，总是运行短作业而不运行长作业的调度程序，可能会获得出色的吞吐性能（每小时大量的短作业），但是其代价是对于长的作业周转时间很差。如果短作业以一个稳定的速率不断到达，长作业可能根本运行不了，这样平均周转时间是无限长，但是得到了高的吞吐量。

CPU利用率常常用于对批处理系统的度量。尽管这样，CPU利用率并不是一个好的度量参数。真正有价值的是，系统每小时可完成多少作业（吞吐量），以及完成作业需要多长时间（周转时间）。把CPU利用率作为度量依据，就像用引擎每小时转动了多少次来比较汽车的好坏一样。另一方面，知道什么时候CPU利用率接近100%比知道什么时候要求得到更多的计算能力要有用。

对于交互式系统，则有不同的指标。最重要的是最小响应时间，即从发出命令到得到响应之间的时间。在有后台进程运行（例如，从网络上读取和存储电子邮件）的个人计算机上，用户请求启动一个程序或打开一个文件应该优先于后台的工作。能够让所有的交互式请求首先运行的则是好服务。

一个相关的问题是均衡性。用户对做一件事情需要多长时间总是有一种固有的（不过通常不正确）看法。当认为一个请求很复杂需要较多的时间时，用户会接受这个看法，但是当认为一个请求很简单，但也需要较多的时间时，用户就会急躁。例如，如果点击一个图标花费了60秒钟发送完成一份传真，用户大概会接受这个事实，因为他没有期望花5秒钟得到传真。

另一方面，当传真发送完成，用户点击断开电话连接的图标时，该用户就有不一样的期待了。如果30秒之后还没有完成断开操作，用户就可能会抱怨，而60秒之后，他就要气得要命了。之所以有这种行为，其原因是：一般用户认为拿起听筒并建立通话连接所需的时间要比挂掉电话所需的时间长。在有些情形下（如本例），调度程序对响应时间指标起不了作用；但是在另外一些情形下，调度程序还是能够做一些事的，特别是在出现差的进程顺序选择时。

实时系统有着与交互式系统不一样的特性，所以有不同的调度目标。实时系统的特点是或多或少必须满足截止时间。例如，如果计算机正在控制一个以正常速率产生数据的设备，若一个按时运行的数据收集进程出现失败，会导致数据丢失。所以，实时系统最主要的要求是满足所有的（或大多数）截止时间要求。

在多数实时系统中，特别是那些涉及多媒体的实时系统中，可预测性是很重要的。偶尔不能满足截止时间要求的问题并不严重，但是如果音频进程运行的错误太多，那么音质就会下降得很快。视频品质也是一个问题，但是人的耳朵比眼睛对抖动要敏感得多。为了避免这些问题，进程调度程序必须是高度可预测和有规律的。在本章中我们将研究批处理和交互式调度算法，而把有关实时调度处理的研究放到第7章多媒体操作系统中。

2.4.2 批处理系统中的调度

现在我们从一般的调度处理问题转向特定的调度算法。在这一节中，我们将考察在批处理系统中使用的算法，随后将讨论交互式和实时系统中的调度算法。有必要指出，某些算法既可以用在批处理系统