柱面，然后继续沿向上的方向移动。实际上，这相当于将最低编号的柱面看作是最高编号的柱面之上的相邻柱面。

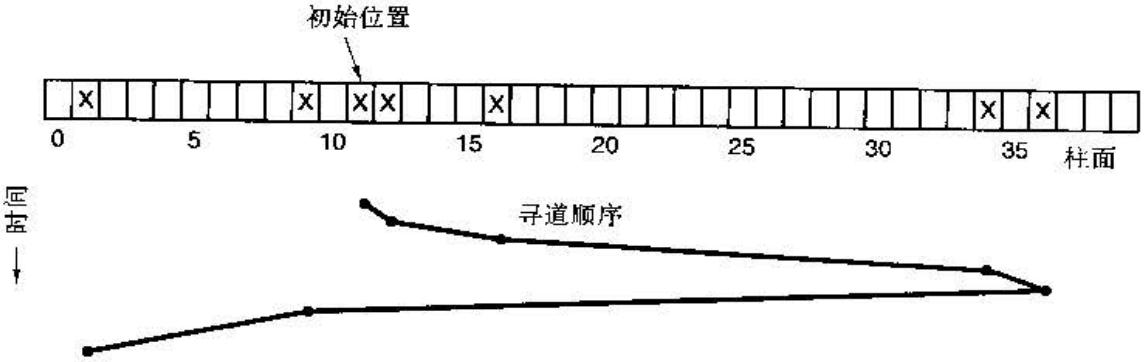

图5-29 调度磁盘请求的电梯算法

某些磁盘控制器提供了一种方法供软件检查磁头下方的当前扇区号。对于这种磁盘控制器，还可以进行另一种优化。如果针对同一柱面有两个或多个请求正等待处理，驱动程序可以发出请求读写下一次要通过磁头的扇区。注意，当一个柱面有多条磁道时，相继的请求可能针对不同的磁道，故没有任何代价。因为选择磁头既不需要移动磁盘臂也没有旋转延迟，所以控制器几乎可以立即选择任意磁头。

如果磁盘具有寻道时间比旋转延迟快很多的特性，那么应该使用不同的优化策略。未完成的请求应该按扇区号排序，并且当下一个扇区就要通过磁头的时候，磁盘臂应该飞快地移动到正确的磁道上对其进行读或者写。

对于现代硬盘，寻道和旋转延迟是如此影响性能，所以一次只读取一个或两个扇区的效率是非常低下的。由于这个原因，许多磁盘控制器总是读出多个扇区并对其进行高速缓存，即使只请求一个扇区时也是如此。典型地，读一个扇区的任何请求将导致该扇区和当前磁道的多个或者所有剩余的扇区被读出，读出的扇区数取决于控制器的高速缓存中有多少可用的空间。例如，在图5-18所描述的磁盘中有4MB的高速缓存。高速缓存的使用是由控制器动态地决定的。在最简单的模式下，高速缓存被分成两个区段，一个用于读，一个用于写。如果后来的读操作可以用控制器的高速缓存来满足，那么就可以立即返回被请求的数据。

值得注意的是，磁盘控制器的高速缓存完全独立于操作系统的高速缓存。控制器的高速缓存通常保存还没有实际被请求的块，但是这对于读操作是很便利的，因为它们只是作为某些其他读操作的附带效应而恰巧要在磁头下通过。与之相反，操作系统所维护的任何高速缓存由显式地读出的块组成，并且操作系统认为它们在较近的将来可能再次需要（例如，保存目录块的一个磁盘块）。

当同一个控制器上有多个驱动器时，操作系统应该为每个驱动器都单独地维护一个未完成的请求表。一旦任何一个驱动器空闲下来，就应该发出一个寻道请求将磁盘臂移到下一个将被请求的柱面处（假设控制器允许重叠寻道）。当前传输结束时，将检查是否有驱动器的磁盘臂位于正确的柱面上。如果存在一个或多个这样的驱动器，则在磁盘臂已经位于正确柱面处的驱动器上开始下一次传输。如果没有驱动器的磁盘臂处于正确的位置，则驱动程序在刚刚完成传输的驱动器上发出一个新的寻道命令并且等待，直到下一次中断到来时检查哪一个磁盘臂首先到达了目标位置。

上面所有的磁盘调度算法都是默认地假设实际磁盘的几何规格与虚拟几何规格相同，认识到这一点十分重要。如果不是这样，那么调度磁盘请求就毫无意义，因为操作系统实际上不能断定柱面40与柱面200哪一个与柱面39更接近。另一方面，如果磁盘控制器能够接收多个未完成的请求，它就可以在内部使用这些调度算法。在这样的情况下，算法仍然是有效的，但是低了一个层次，局限在控制器内部。

5.4.4 错误处理

磁盘制造商通过不断地加大线性位密度而持续地推进技术的极限。在一块5.25英寸的磁盘上，处于中间位置的一个磁道大约有300mm的周长。如果该磁道存放300个512字节的扇区，考虑到由于前导码、ECC和扇区间隙而损失了部分空间这样的实际情况，线性记录密度大约是5000b/mm。记录5000b/mm需