都有初始的大小和随后依需要可以增长到的最大空间，但是在系统安装时就创建这些文件达到它的最大值是最好的。如果当文件系统非常满却需要增长页面文件时，页面文件的新空间可能会由多个碎片所组成，这会降低系统的性能。

操作系统通过为进程的私有页写入映射信息到页表入口，或与原页表入口相对应的共享页的内存区对象，来跟踪虚拟页与页面文件的映射关系。除了被页面文件保留的页面外，进程中的许多页面也被映射到文件系统中的普通文件。

程序文件中的可执行代码和只读数据（例如EXE或DLL）可以映射到任何进程正在使用的地址空间。因为这些页面无法被修改，它们从来不需要换出内存，然而在页表映射全部被标记为无效后，可以立即重用物理页面。当一个页面在今后再次需要时，内存管理器将从程序文件中将其读入。

有时候页面开始时为只读但最终被修改。例如，当调试进程时在代码中设定中断点，或将代码重定向为进程中不同的地址，或对于开始时为共享的数据页面进行修改。在这些情况下，像大多数现代操作系统一样，Windows支持写时复制（copy-on-write）类型的页面。这些页面开始时像普通的被映射页面一样，但如果试图修改任何部分页面，内存管理器将会建立一份私有的、可写的副本。然后它更新虚拟页面的页表，使之指向那个私有副本，并且使线程重新进行写操作——这一次将会成功。如果这个副本之后需要被换出内存，那么它将被写回到页面文件而不是原始文件中。

除了从EXE和DLL文件映射程序代码和数据，一般的文件都可以映射到内存中，使得程序不需要进行显式的读写操作就可以从文件引用数据。I/O操作仍然是必要的，但它们由内存管理器通过使用内存区对象隐式提供，来表示内存中的页面和磁盘中的文件块的映射。

内存区对象并不一定和文件相关。它们可以和匿名内存区域相关。通过映射匿名内存区对象到多个进程，内存可以在不分配磁盘文件的前提下共享。既然内存区可以在NT名字空间给予名字，进程可以通过用名字打开内存区对象、或者复制进程间的内存区对象句柄的方式来进行通信。

3. 大物理内存寻址

多年前，当16位（或20位）的地址空间还作为标准的时候，机器已有兆字节的物理内存，人们努力想出各种技术使得程序可以使用更多的物理内存、而不是去适应有限的地址空间。这些技术通常基于*存储器组转换*（bank switching），使得一个程序可以突破16或者20位的限制，替换掉自己的一些内存块。在刚引入32位计算机时，大多数桌面计算机只有几个兆的物理内存。然而随着内存在集成电路上变得更加密集，可用内存开始迅速增长。这推动了服务器的发展，因为服务器上的应用程序往往需要更多的内存。英特尔的Xeon芯片支持物理地址扩展（PAE），物理内存寻址空间从32位变为36位，意味着一个单一的系统可以支持高达64GB的物理内存。这远远大于2G或者3G——单个进程可以在32位的用户模式寻址的虚拟地址空间，然而一些像SQL数据库这样的大型应用软件恰恰被设计为运行在一个单个进程的寻址空间中，因此存储器组转换已经过时了，取代它的是*地址窗口扩展*（Address Windowing Extensions，AWE）。这种机制允许程序（以正确的特权级运行）去请求物理内存的分配。进程可以保留所需的虚拟地址，并请求操作系统进行虚拟地址与物理地址间的映射。在所有的服务器应用64位寻址方式前，AWE一直充当权宜之计的角色。

11.5.2 内存管理系统调用

Win32 API包含了大量的函数来支持一个进程显式地管理它自己的虚拟内存，其中最重要的函数如图11-31所示。它们都是在包含一个单独的页或一个由两个或多个在虚拟地址空间中连续页的序列的区域上进行操作的。

前四个API函数是用来分配、释放、保护和查询虚拟地址空间中的区域的。被分配的区域总是从64KB的边界开始，以尽量减少移植到将来的体系结构的问题（因为将来的体系结构可能使用比当前使用的页更大的页）。实际分配的地址空间可以小于64KB，但是必须是一个页大小的整数倍。接下来的两个API给一个进程把页面固定到内存中以防止它们被替换到外存以及撤销这一性质的功能。举例来说，一个实时程序可能需要它的页面具有这样的性质以防止在关键操作上发生页面失效。操作系统强加了一个限制来防止一个进程过于“贪婪”：这些页面能够移出内存，但是仅仅在整个进程被替换出内存的时候才能这么做。当该进程被重新装入内存时，所有之前被指定固定到内存中的页面会在任何线程开始运