3.2.2 交换技术

如果计算机物理内存足够大，可以保存所有进程，那么之前提及的所有方案都或多或少是可行的。但实际上，所有进程所需的RAM数量总和通常要远远超出存储器能够支持的范围。在一个典型的Windows或Linux系统中，在计算机完成引导后，会启动40~60个，甚至更多的进程。例如，当一个Windows应用程序安装后，通常会发出一系列命令，使得在此后的系统引导中会启动一个仅仅用于查看该应用程序更新的进程。这样一个进程会轻易地占据5~10MB的内存。其他后台进程还会查看所收到的邮件和进来的网络连接，以及其他很多诸如此类的任务。并且，这一切都发生在第一个用户程序启动之前。当前重要的应用程序能轻易地占据50~200MB甚至更多的空间。因此，把所有进程一直保存在内存中需要巨大的内存，如果内存不够，就做不到这一点。

有两种处理内存超载的通用方法。最简单的策略是交换（swapping）技术，即把一个进程完整调入内存，使该进程运行一段时间，然后把它存回磁盘。空闲进程主要存储在磁盘上，所以当它们不运行时就不会占用内存（尽管它们的一些进程会周期性地被唤醒以完成相关工作，然后就又进入睡眠状态）。另一种策略是虚拟内存（virtual memory），该策略甚至能使程序在只有一部分被调入内存的情况下运行。下面我们先讨论交换技术，3.3节我们将考察虚拟内存。

交换系统的操作如图3-4所示。开始时内存中只有进程A。之后创建进程B和C或者从磁盘将它们换入内存。图3-4d显示A被交换到磁盘。然后D被调入，B被调出，最后A再次被调入。由于A的位置发生变化，所以在它换入的时候通过软件或者在程序运行期间（多数是这种情况）通过硬件对其地址进行重定位。例如，在这里可以很好地使用基址寄存器和界限寄存器。

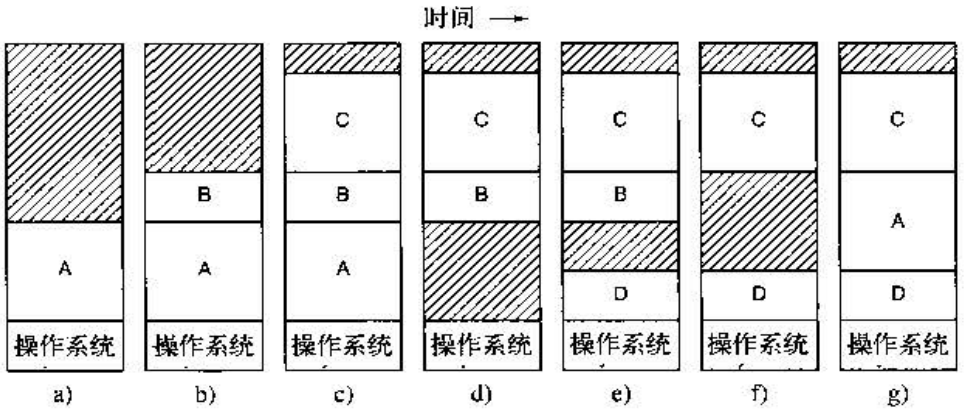

图3-4 内存分配情况随着进程进出而变化，阴影区域表示未使用的内存

交换在内存中产生了多个空闲区（hole，也称为空洞），通过把所有的进程尽可能向下移动，有可能将这些小的空闲区合成一大块。该技术称为内存紧缩（memory compaction）。这个操作通常不进行，因为它要耗费大量的CPU时间。例如，一台有1GB内存的计算机可以每20ns复制4个字节，它紧缩全部内存大约要花费5s。

有一个问题值得注意，即当进程被创建或换入时应该为它分配多大的内存。若进程创建时其大小是固定的并且不再改变，则分配很简单，操作系统准确地按其需要的大小进行分配，不多也不少。

但是如果进程的数据段可以增长，例如，很多程序设计语言都允许从堆中动态地分配内存，那么当进程空间试图增长时，就会出现问题。若该进程与一个空闲区相邻，那么可把该空闲区分配给该进程让它在这个空闲区增大。另一方面，若进程相邻的是另一个进程，那么要么把需要增长的进程移到内存中一个足够大的区域中去，要么把一个或多个进程交换出去，以便生成一个足够大的空闲区。若一个进程在内存中不能增长，而且磁盘上的交换区也已满了，那么这个进程只有挂起直到一些空间空闲（或者可以结束该进程）。