对许多压缩算法而言，备份磁带上的单个坏点就能破坏解压缩算法，并导致整个文件甚至整个磁带无法阅读。所以是否要对备份文件流进行压缩必须慎重考虑。

第四，对活动文件系统做备份是很难的。因为在转储过程中添加、删除或修改文件和目录可能会导致文件系统的不一致性。不过，既然转储一次需要几个小时，那么在晚上大部分时间让文件系统脱机是很有必要的，虽然这种做法有时会令人难以接受。正因如此，人们修改了转储算法，记下文件系统的瞬时状态，即复制关键的数据结构，然后需要把将来对文件和目录所做的修改复制到块中，而不是处处更新它们（Hutchinson等人，1999）。这样，文件系统在抓取快照的时候就被有效地冻结了，留待以后空闲时再备份。

第五，即最后一个问题，做备份会给一个单位引入许多非技术性问题。如果当系统管理员下楼去取打印文件，而毫无防备地把备份磁带搁置在办公室里的时候，就是世界上最棒的在线保安系统也会失去作用。这时，一个间谍所要做的只是潜入办公室、将一个小磁带放入口袋，然后绅士般地离开。再见吧保安系统。即使每天都做备份，如果碰上一场大火烧光了计算机和所有的备份磁带，那做备份又有什么意义呢？由于这个原因，所以备份磁带应该远离现场存放，不过这又带来了更多的安全风险（因为，现在必须保护两个地点了）。关于此问题和管理中的其他实际问题，请参考（Nemeth等人，2000）。接下来我们只讨论文件系统备份所涉及的技术问题。

转储磁盘到磁带上有两种方案：物理转储和逻辑转储。*物理转储*是从磁盘的第0块开始，将全部的磁盘块按序输出到磁带上，直到最后一块复制完毕。此程序很简单，可以确保万无一失，这是其他任何实用程序所不能比的。

不过有几点关于物理转储的评价还是值得一提的。首先，未使用的磁盘块无须备份。如果转储程序能够得到访问空闲块的数据结构，就可以避免该程序备份未使用的磁盘块。但是，既然磁带上的第k块并不代表磁盘上的第k块，那么要想略过未使用的磁盘块就需要在每个磁盘块前边写下该磁盘块的号码（或其他等效数据）。

第二个需要关注的是坏块的转储。制造大型磁盘而没有任何瑕疵几乎是不可能的，总是有一些坏块存在。有时进行低级格式化后，坏块会被检测出来，标记为坏的，并被应对这种紧急状况的在每个轨道末端的一些空闲块所替换。在很多情况下，磁盘控制器处理坏块的替换过程是透明的，甚至操作系统也不知道。

然而，有时格式化后块也会变坏，在这种情况下操作系统可以检测到它们。通常，可以通过建立一个包含所有坏块的“文件”来解决这个问题——只要确保它们不会出现在空闲块池中并且决不会被分配。不用说，这个文件是完全不能够读取的。

如果磁盘控制器将所有坏块重新映射，并对操作系统隐藏的话，物理转储工作还是能够顺利进行的。另一方面，如果这些坏块对操作系统可见并映射到在一个或几个坏块文件或者位图中，那么在转储过程中，物理转储程序绝对有必要能访问这些信息，并避免转储之，从而防止在对坏块文件备份时的无止境磁盘读错误发生。

物理转储的主要优点是简单、极为快速（基本上是以磁盘的速度运行）。主要缺点是，既不能跳过选定的目录，也无法增量转储，还不能满足恢复个人文件的请求。正因如此，绝大多数配置都使用逻辑转储。

*逻辑转储*从一个或几个指定的目录开始，递归地转储其自给定基准日期（例如，最近一次增量转储或全面系统转储的日期）后有所更改的全部文件和目录。所以，在逻辑转储中，转储磁带上会有一连串精心标识的目录和文件，这样就很容易满足恢复特定文件或目录的请求。

既然逻辑转储是最为普遍的形式，就让我们以图4-25为例来仔细研究一个通用算法。该算法在UNIX系统上广为使用。在图中可以看到一棵由目录（方框）和文件（圆圈）组成的文件树。被阴影覆盖的项目代表自基准日期以来修改过，因此需要转储，无阴影的则不需要转储。

该算法还转储通向修改过的文件或目录的路径上的所有目录（甚至包括未修改的目录），原因有二。其一是为了将这些转储的文件和目录恢复到另一台计算机的新文件系统中。这样，转储程序和恢复程序就可以在计算机之间进行文件系统的整体转移。