**UNIX的快速文件系统**

**陈双-P17206032-中电十五所**

**介绍：**

本文描述了UNIX文件系统的重新实现。重新实现通过使用更灵活的分配策略提供了更高的吞吐量速率，该策略允许更好的参考位置，并且可以适用于广泛的外围设备和处理器特性。新的文件系统集群数据被顺序访问，并提供两个块大小，以允许快速访问大文件，同时不浪费大量的空间用于小文件。文件访问率比传统UNIX文件系统快十倍。讨论了程序员接口的长期增强需求。这些机制包括在文件上放置咨询锁的机制、跨文件系统的名称空间的扩展、使用长文件名的能力、以及对资源使用的管理控制的规定。

文件系统的输入/输出是由内核缓冲的；没有数据传输的对齐约束，所有操作都会出现同步。磁盘的所有传输都是512字节的块，可以任意地放置在文件系统的数据区域内。实际上，除了可用的磁盘空间之外，没有任何限制被放置在文件增长这一方面。

对UNIX文件系统进行了改进，以提高其性能。由于UNIX文件系统接口是很好理解的，并不是固有地慢，所以这种开发保留了抽象，并简单地改变了底层实现以增加其吞吐量。因此，系统的用户还没有面临大规模的软件转换。

**旧文件系统：**

每个磁盘驱动器被划分为一个或多个分区。这些磁盘分区中的每一个可以包含一个文件系统。文件系统从不跨越多个分区。文件系统用超级块来描述，它包含文件系统的基本参数。这些包括文件系统中的数据块的数量、最大文件数的计数和指向空闲列表的指针，这是文件系统中所有空闲块的链表。

文件系统内是文件。某些文件被区分为目录，并包含指向本身可能是目录的文件的指针。每个文件都有与其相关联的描述符，称为索引节点。iNoDE包含描述文件所有权的信息、标记最后修改的时间戳和文件的访问时间，以及指向文件的数据块的索引数组。

**新文件系统：**

在新的文件系统中，每个磁盘驱动器包含一个或多个文件系统。文件系统由其超级块描述，位于文件系统的磁盘分区的开头。因为超级块包含关键数据，所以它被复制以防止灾难性损失。这是在创建文件系统时完成的，因为超级块数据没有改变，除非头部崩溃或其他硬盘错误导致默认超级块不可用，否则不需要引用这些副本。

为了能够使用大的块而不需要太多的浪费，小文件必须以更有效的方式存储。新的文件系统通过允许将单个文件系统块分割成一个或多个片段来实现这一目标。在创建文件系统时指定文件系统片段大小；每个文件系统块可以可选地分成2, 4个或8个片段，每个片段都是可寻址的。这些碎片大小的下限受到磁盘扇区大小的限制，通常为512字节。与每个气缸组相关联的块映射记录在碎片级别的气缸组中可用的空间；为了确定是否有块可用，检查对齐的片段。

在一段时间内扩展一个文件的问题是，由于碎片化的块扩展到一个完整的块，数据可以被复制很多次。如果用户程序一次写入完整的块，除了文件结尾处的部分块，碎片再分配可以最小化。由于具有不同块大小的文件系统可能驻留在同一系统上，所以文件系统接口已被扩展以提供应用程序最佳的读或写大小。对于文件，最佳大小是文件正在访问的文件系统的块大小。对于其他对象，例如管道和套接字，最佳大小是底层缓冲区大小。这个特性由标准输入/输出库使用，这是大多数用户程序使用的包。此功能还被某些系统实用程序使用，例如归档器和加载程序，它们自己进行输入和输出管理，并需要最高的文件系统带宽。

**功能增强的文件系统：**

1）长文件名，文件名现在可以是几乎任意长度。只有读取目录的程序才会受到此更改的影响。为了提高不运行新文件系统的UNIX系统的可移植性，引入了一组目录访问例程，以提供对旧系统和新系统的目录的一致接口。目录以512字节单位分配，称为块。这样的大小被选择，使得每个分配可以在一次操作中被传送到磁盘。块被分成可变长度记录，称为目录条目。目录条目包含将文件名映射到其关联的内联所必需的信息。不允许目录条目跨越多个块

2）文件锁定，旧文件系统没有锁定文件的规定。需要同步文件更新的进程必须使用单独的“锁定”文件。一个进程将尝试创建一个“锁定”文件。如果创建成功，则该过程可以继续进行更新；如果创建失败，则该进程将等待并重试。这种机制有三个缺点。进程通过循环来创建锁来消耗CPU时间。由于系统崩溃而留下的锁必须手动删除（通常在系统启动命令脚本中）。最后，始终允许系统管理员运行的进程创建文件，因此被迫使用不同的机制。虽然有可能绕过所有这些问题，但解决方案不是直接的，因此已经添加了锁定文件的机制。

锁定方案分为两类，那些使用硬锁和那些使用咨询锁。咨询锁和硬锁的主要区别在于执行的程度。当程序试图访问文件时，通常会执行硬锁；只有在程序请求时才应用咨询锁。

3）符号链接，传统的UNIX文件系统允许同一文件系统中的多个目录条目引用单个文件。每个目录条目“链接”文件名称到索引节点及其内容。符号链接实现为包含路径名的文件。当系统在解释路径名的组件时遇到符号链接时，符号链接的内容被添加到路径名的其余部分，并且该名称被解释为产生结果路径名。在UNIX中，路径名相对于文件系统层次结构的根，或者相对于进程的当前工作目录来指定。相对于根指定的路径名称为绝对路径名。相对于当前工作目录指定的路径名称为相对路径名。如果符号链接包含绝对路径名，则使用绝对路径名，否则符号链接的内容相对于文件层次结构中链接的位置进行评估。

4）重命名，创建现有文件的新版本的程序通常将新版本创建为临时文件，然后用目标文件的名称重命名临时文件。在旧的UNIX文件系统中，重命名需要对系统进行三次调用。如果程序中断或系统在这些调用之间崩溃，则目标文件只能保留其临时名称。为了消除这种可能性，已经添加了重命名系统调用。重命名调用以保证目标名称存在的方式进行重命名操作。

重命名对数据文件和目录都起作用。当重命名目录时，系统必须进行特殊的验证检查，以确保目录树结构不会因创建循环或不可访问目录而损坏。如果将父目录移动到其子孙中的一个，则会发生这种损坏。验证检查需要跟踪目标目录的后裔，以确保它不包括正在移动的目录。

5）配额

UNIX系统传统上试图尽可能最大程度地共享所有可用资源。因此，任何单个用户都可以分配文件系统中的所有可用空间。在某些环境下，这是不可接受的。因此，添加了配额机制来限制用户可以获得的文件系统资源的数量。