计算机研究与发展
Journal of Computer Research and Development
ISSN 1000-1239/CN 11-1777/TP
42(6): 1019~1027, 2005

DCFS2 的元数据一致性策略

熊 劲[1,2] 范志华[1,2] 马 捷[1] 唐荣锋[1,2] 李 晖[1,2] 孟 丹[1]
[1](中国科学院计算技术研究所国家智能计算机研究开发中心 北京 100080)
[2](中国科学院研究生院 北京 100039)
(xj@ncic.ac.cn)

Metadata Consistency in DCFS2

Xiong Jin[1,2], Fan Zhihua[1,2], Ma Jie[1], Tang Rongfeng[1,2], Li Hui[1,2], and Meng Dan[1]
[1](*National Research Center for Intelligent Computing Systems, Institute of Computing Technology, Chinese Academy of Sciences, Beijing* 100080)
[2](*Graduate School of the Chinese Academy of Sciences, Beijing* 100039)

Abstract With the increase of required performance, capacity and scalability for a cluster file system, the multi-metadata-server structure is the trend for future cluster file systems. Distributed metadata processing based on multiple metadata servers is an important but difficult issue. In order to get high metadata performance and scalability, DCFS2, a cluster file system, uses distributed metadata processing over multiple metadata servers. Moreover, DCFS2 solved the metadata consistency problem for distributed metadata processing through a distributed logging technology and a simplified two-phase commit protocol. And performance results show that DCFS2's metadata processing policy based on distributed logging can deliver high I/O performance, and the file system can quickly recover from a metadata server failure.

Key words cluster file system; file system consistency; metadata consistency; distributed metadata processing

摘 要 随着集群应用对机群文件系统的性能、容量和规模等需求的日益增长,采用多元数据服务器是机群文件系统发展的必然趋势.基于多元数据服务器的分布式元数据处理是文件系统研究的一个重要问题.机群文件系统 DCFS2 采用分布式日志技术和改进的两阶段提交协议解决了分布式元数据处理下元数据的一致性问题.性能测试结果表明,DCFS2 所采用的基于分布式日志的元数据处理策略能够提供高的 I/O 性能,并能够保证在元数据服务器失效后文件系统快速恢复.

关键词 机群文件系统;文件系统一致性;元数据一致性;分布式元数据处理

中图法分类号 TP316.4

1 引 言

集群体系结构已被广泛用于构建高性能计算机.集群上的应用以及集群的管理和使用都需要一个全局共享的机群文件系统.

应用的海量数据需求推动互连技术和存储系统的迅速发展,SAN 和 IP-SAN 成为满足海量数据存储需求的存储体系结构[1,2].从近年的发展趋势中我们不难看到,SAN 和 IP-SAN 将广泛应用于集群系统.

DCFS2 是为曙光 4000A 研制的、基于共享 IP-

收稿日期:2005-01-12
基金项目:国家"八六三"高技术研究发展计划基金项目(2002AA1Z2102,2002AA104410)

SAN 存储的机群文件系统. 它利用客户节点与 IP-SAN 存储之间直接数据通路. 为了获得高的元数据处理性能和好的扩展性,DCFS2 采用多元数据服务器结构和分布式元数据处理,有些元数据操作需要多个元数据服务器协同方可完成. 这样带来的问题是一旦某个元数据服务器在处理过程中崩溃,就可能导致各个元数据服务器上的元数据不一致.

其他的文件系统无论是传统的分布式文件系统如 NFS 和 AFS,还是近年推出的 SAN 文件系统如 GFS[3],CXFS[4],GPFS[5]和 StorageTank[6],包括还在研制中的基于对象存储的 Lustre 文件系统[7],都没有分布式元数据处理,因此,它们可以利用传统的元数据日志技术或直接利用服务器上的日志文件系统[8]来解决元数据的一致性问题.

DCFS2 借鉴分布式事务处理[9]中的分布式日志技术和两阶段提交协议,解决了分布式元数据处理中的元数据一致性问题. 任何一个元数据服务器崩溃后,DCFS2 都能够快速恢复元数据的一致性,保证文件系统的可用性.

本文的安排如下:第 2 节介绍文件系统一致性方面的相关研究;第 3 节介绍 DCFS2 的总体结构和分布式元数据处理机制;第 4 节详细介绍 DCFS2 的元数据处理协议、分布式日志技术和故障恢复协议;第 5 节分析分布式日志对文件系统性能的影响以及恢复性能;第 6 节给出本研究的结论和未来的研究方向.

2 背景及相关研究

文件系统是提供组织和管理保存在持久存储介质(如磁盘、RAID 等)上的用户数据的手段. 文件系统的内容包括用户数据(即文件内容)和元数据(即描述文件系统组织结构的数据). 一个文件操作通常是由一系列更基本的子操作构成,每个子操作只修改一个元数据. 构成一个文件操作的子操作序列是一个不可分割的整体. 因此,文件操作具有与事务类似的 ACID 特性[10].

如果一个文件系统的内容(包括数据和元数据)是而且仅是所有已成功完成的文件操作按顺序完成的结果,那么就说这个文件系统是一致的.

文件系统在使用过程中有许多中间状态. 在这些中间状态时文件系统不一定是一致的. 但是在稳态时(如 unmount 后)它必须是一致的. 如果在中间状态时系统突然崩溃,则可能导致文件系统的一致性被破坏,文件系统某些内容不可用,甚至整个文件系统不可用. 另一方面,文件系统广泛使用缓存来缓解磁盘与 CPU 速度上的巨大差距. 缓存引入后,使文件系统在机器崩溃时的不一致性问题更加严重. 由于文件系统的不一致只发生在系统出现故障时,因此维护文件系统一致性的方法称为故障恢复.

按一致性由弱到强,可将文件系统一致性分为 3 个级别:

(1) 1 级. 元数据一致,这是最低级别,仅保证文件系统的组织结构是正确的.

(2) 2 级. 元数据和数据都一致,但并不保证含所有已完成操作的结果,也不一定保证操作的顺序性.

(3) 3 级. 元数据和数据都一致,且含所有已完成操作的结果. 它保证文件操作的 ACID 性质,是最高级别.

元数据一致性是文件系统可用性的最基本要求. 对于很多应用环境来说这是可以接受的.

2.1 本地文件系统的故障恢复技术

本地文件系统广泛采用的故障恢复技术包括 fsck 和日志文件系统. fsck[11]工具是传统 UNIX 文件系统采用的故障恢复手段. 传统的 UNIX 文件系统为保证元数据的一致性,元数据缓存采用同步写回(write-through)方式,使性能损失较大. 在系统崩溃后,用 fsck 工具恢复时需要多遍扫描磁盘上的整个文件系统,恢复时间较长,而且与文件系统的大小成正比,它不适合用于大文件系统的恢复.

日志文件系统[3,8,12]是借鉴数据库中的事务处理和故障恢复技术[9,10,13],将文件操作作为事务对待,采用写前日志技术(write-ahead log). 每当需要修改元数据时,先记录一个关于修改的日志. 如果系统崩溃,恢复时只需要依据日志记录重做或撤消. 许多现代文件系统都采用元数据日志技术,包括 JFS,XFS,NTFS,EXT3 等.

此外,还有学者提出了软更新(soft updates)技术[14]和元数据快照[15]技术来快速恢复元数据一致性,它们都通过控制缓存写回策略来保证磁盘上的文件系统的一致性.

2.2 分布式文件系统的故障恢复技术

NFS,Sprite[16]和 Calypso[17]都是单服务器的分布式文件系统,实现在服务器的本地文件系统之上. CXFS[4],Lustre[7]和 SANergy[18]等都只有单个元数据服务器,元数据保存在服务器的本地文件系统中. 这些文件系统都利用服务器的本地文件系统故障恢

复技术来保证文件系统的一致性,一般都在服务器上采用日志文件系统.

GFS[3]和 GPFS[5]都是对称式 SAN 文件系统,没有服务器(但有锁服务器),它们都采用与日志文件系统类似的元数据日志技术来恢复元数据的一致性.

StorageTank[6]和 PVFS2[19]都是多元数据服务器结构.StorageTank 的元数据分布采用名字空间静态划分方式,各个元数据服务器无需交互,只需恢复崩溃元数据服务器上的元数据的一致性.因此,它采用与日志文件系统类似的日志技术.PVFS2 含分布式元数据操作,它试图通过规定子操作顺序和每个修改立即写回磁盘来保证文件系统层次结构的正确性.但元数据服务器崩溃后可能遗留垃圾元数据,例如有 inode,但没有加入任何目录,目前 PVFS2 对之不做处理.

综上所述,目前大部分分布式文件系统都直接利日志文件系统或采用与日志文件系统类似的日志技术来解决故障恢复问题.

在分布式数据库中,分布式事务是指一个事务需要修改多个节点上的数据资源.两阶段提交协议[9]是用来保证分布式事务完整性(ACID)的技术.分布式元数据操作类似于一种简单的分布式事务,因此 DCFS2 借鉴分布式事务处理和恢复技术来解决元数据一致性问题.

3　DCFS2 概述

DCFS2 是为曙光 4000A 超级服务器研制的、基于共享 IP-SAN 存储的机群文件系统.

3.1　总体结构

当部署在一个集群上时,DCFS2 由一个存储空间管理器(ssm)、若干元数据服务器(mgr)、若干 IP-SAN 存储设备、一个配置及状态服务器(csa)和众多的客户节点组成,如图 1 所示.ssm 负责管理所有 IP-AN 存储设备的存储空间,并分配给各个文件使用;每个 mgr 负责管理一部分名字空间的元数据(包括属性、目录内容、文件块的物理位置信息等);csa 负责维护 DCFS2 的配置和状态信息,辅助管理工具 *dcfs_adm* 完成文件系统的日常管理,包括配置、启动和停止 DCFS2 服务进程、创建、安装和卸载文件系统、状态查看和故障恢复等;客户节点是安装(mount)了 DCFS2 文件系统的节点.

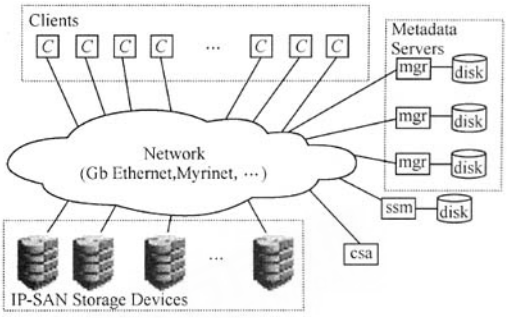

Fig. 1　The architecture of DCFS2.

图 1　DCFS2 的结构

为了支持大规模节点的共享,DCFS2 将元数据的存储和访问路径同文件数据的存储和访问路径分离.文件数据存储在客户节点可直接访问的 IP-SAN 存储设备上;元数据则存储在元数据服务器(mgr)上,通过 DCFS2 的元数据访问协议来访问.将元数据的存储和处理分离出来有 3 个好处:第 1,数据和元数据可采用不同的网络来进行传输,可进一步提高二者的访问性能;第 2,可根据元数据和数据不同的存取特点分别进行优化;第 3,元数据可利用本地文件来存储,大大简化实现的难度.

客户节点上账行的用户进程在访问 DCFS2 文件时,用户进程的操作首先被 VFS 层处理,并转换成对 DCFS2 客户端文件系统的请求.DCFS2 客户端文件系统在处理这些请求时,要么与 mgr 交互,获得相应的元数据;要么与 ssm 交互,申请物理存储空间;要么直接与 IP-SAN 存储设备交互,读写数据.在目前的实现中,客户端文件系统通过一个用户级进程 *dcfsd* 来与 mgr 和 ssm 交互.

3.2　分布式元数据处理

DCFS2 之所以采用多元数据服务器结构,是因为现今的高性能计算应用不仅对机群文件系统的 I/O 带宽和元数据处理性能提出了很高的需求,而且对文件系统的规模和扩展性也有很高的需求[20].单一的元数据服务器已不能满足应用对规模和性能日益增长的需求,正在研制中的 PVFS2[19]和新版本的 Lustre[7]都采用多元数据服务器结构.因此,采用多元数据服务器是机群文件系统发展的必然趋势.

DCFS2 中由存储元数据的 mgr 来处理对该元数据的访问请求.元数据分布是指名字空间中的每个文件和目录由哪个元数据服务器来存储和处理.DCFS2 采用动态目录分布策略,它以一个目录树为

分布粒度,其大小(宽度和深度)是预先定义的. 当在一个目录树中创建一个新对象(文件或目录)时,如果加入新对象后这个目录树没有超过分布粒度大小,则采用这个目录树所在的 mgr 来存储新建对象的元数据. 否则,按照一定的策略重新选择一个 mgr 来存储新建对象的元数据.

因此,在 DCFS2 文件系统的层次结构中,有些对象的父目录所在的 mgr 与该对象本身所在的 mgr 不同. 有些元数据操作(lookup, create, mkdir, rename 等)需要涉及父目录和对象本身,因此,如果对象与其父目录在同一 mgr 上,这些操作可在一个 mgr 上完成;否则,这些操作就需要多个元数据服务器协作才能完成. 前者称为普通元数据操作,后者称为分布式元数据操作. 一个分布式元数据操作所需要的时间远远比一个普通元数据操作长. DCFS2 的分布策略尽可能使有父子关系的对象在同一个 mgr 上,因此多数元数据操作都是普通元数据操作,只有小部分的分布式元数据操作.

4 DCFS2 的元数据一致性策略

DCFS2 的元数据一致性需要解决两个关键问题:第 1,单个 mgr 上的元数据一致性. 因为 mgr 有缓存,若 mgr 崩溃缓存中的数据将丢失. 第 2,多个 mgr 之间的元数据一致性. DCFS2 中含有分布式元数据操作,在某些情况下可能出现 mgr 之间的元数据不一致,例如,在一个分布式元数据操作的处理中,①某个 mgr 发现因为资源冲突或其他原因而不能完成相应的子操作;②某个 mgr 在完成其子操作之前崩溃. 这两种情况下,由于其他的 mgr 都已完成各自的子操作,各个 mgr 上的元数据将不一致. DCFS2 的故障恢复目标是保证在某个 mgr 崩溃或者某些客户端崩溃后,能够快速恢复元数据的一致性.

4.1 元数据处理协议

分布式元数据处理有两种机制:一种是以客户端为主导的处理机制,另一种是以 mgr 为主导的处理机制. 以客户端为主导的处理机制是由客户端分析用户进程发出的文件操作需要哪些 mgr 参与,然后依次向它们发出请求. 以 mgr 为主导的处理机制是客户端分析用户进程发出的文件操作,向一个 mgr 发出请求. 该 mgr 在处理来自客户端的请求时,若发现需要其他 mgr 参与,它便会依次向相关的 mgr 发出请求. 如果采用以客户端为主导的处理机制,那么客户节点崩溃会影响元数据的一致性,需要复杂的恢复流程. 而在大规模集群系统中,有几百甚至上千个客户节点,客户节点崩溃的概率是比较高的. 相反,采用 mgr 为主导的处理机制,客户节点崩溃不影响元数据一致性,从而大大减少了故障恢复所需的处理. 因此,DCFS2 采用以 mgr 为主导的处理机制.

mgr 的实现是非阻塞的. 若在处理请求的过程中需要向其他 mgr 发出请求,则将未完成的请求挂在等待队列上,等待其他 mgr 的处理结果,而该 mgr

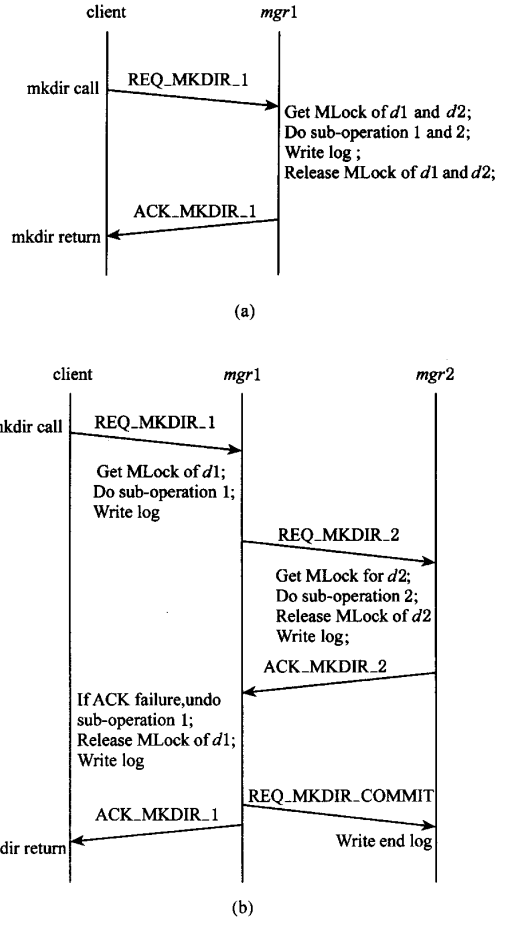

Fig. 2 DCFS2 metadata processing protocol, illustrated by a *mkdir* operation, which makes a directory "*d*2" in directory "*d*1". (a) Protocol for ordinary *mkdir* operation; (b) Protocol for distributed *mkdir* operation.

图 2. DCFS2 元数据处理协议. 以 *mkdir* 操作为例,假设在目录"*d*1"下面创建目录"*d*2". (a) 普通 *mkdir* 操作的处理协议;(b) 分布式的 *mkdir* 操作的处理协议

继续处理下一个元数据请求. 如果后续的请求依赖该未完成的请求的处理结果,那么可能会使得在第 2 子操作失败时,取消第 1 个子操作的结果需要取消该请求之后的所有相关操作. 这样的回滚不容易控制. 为解决此问题,DCFS2 在 inode 上增加了一个锁(称为 MLock),用于使有依赖关系的元数据操作依照一定的顺序进行. 每个需要修改元数据的操作都需要先获取相应的 MLock 后才能开始,从而使得依赖某个未完成操作(进行中的分布式操作)的结果的元数据操作必须等待那个操作完成后方可开始,这样就保证了任何一个分布式操作的中间结果是可以撤消的,不需回滚.

普通元数据操作和分布式元数据操作的处理协议分别如图 2(a),(b)所示. 对于分布式元数据操作,DCFS2 采用一种简化的两阶段提交协议. 接收来自客户节点的操作请求的 mgr 称为协调者(mgr1),而参与该请求服务的其他 mgr 称为参与者(mgr2). 在文献[21]中,我们给出了 DCFS2 的元数据处理协议同两阶段提交协议的比较,性能测试表明,DCFS2 的元数据处理协议有更好的性能.

4.2 分布式日志

为了能够在 mgr 崩溃后快速恢复元数据的一致性,mgr 采用分布式日志技术. 每个 mgr 都分别记录元数据修改日志,并保存在各个 mgr 的本地文件中,称之为日志文件. 一个分布式操作的各个子操作的日志记录分别位于各个 MDS 的日志文件中. 恢复时需要相关的元数据服务器根据各自的日志记录协商各个的子操作完成情况. 因此,关键问题是日志记录的匹配,即如何从多个日志文件中找出某个分布式操作的日志记录. 而且,在一个分布式操作中,协调者和参与者需要不同的恢复协议.

DCFS2 对每个日志记录进行如下标识:操作类型(普通操作/分布式操作)、本服务器的类型(协调者/参与者)、参与本操作的其他服务器标识、请求的序列号和请求类型. 通过这些信息就能够在恢复时将同一个分布式操作在各个 mgr 上的日志记录进行匹配,并确定所需要的恢复协议.

为恢复分布式元数据操作的结果,分布式操作的日志记录必须立即写回磁盘. 因此,对于分布式操作必须采用同步日志. 同步日志是每个操作的日志都立即写回磁盘,它能够在服务器崩溃时恢复出所有已完成的操作,但对文件系统的性能有较大影响. 异步日志将日志缓存在内存中,定期写回磁盘,而且要保证日志先于缓存元数据写回磁盘. 异步日志极大地改善了文件系统的性能,但可能丢失少量的操作结果. 为了减少记录日志的开销,DCFS2 中,普通操作的日志与分布式操作的日志采取不同的写回策略. 为了适应不同应用环境对性能和文件系统可靠性的不同需求,对于普通操作,DCFS2 支持 3 种日志方式,即不记录日志、异步日志和同步日志,这是在启动 mgr 时进行设置.

DCFS2 中,日志文件是固定大小(50MB). 当 mgr 将缓存中的元数据写回磁盘后,与这些元数据对应的日志记录便可以抛弃,释放出占用的日志文件的空间. 由于 mgr 定期写回缓存中的元数据,因此这种日志空间的整理也是定期进行的. 为了解决各个 mgr 上的进度差异,mgr 在日志整理时利用一个辅助文件来暂存未完成的分布式操作的日志记录.

4.3 故障恢复协议

DCFS2 的故障恢复机制的前提是当一个 mgr 失效后,它存储元数据的磁盘仍是好的. 失效 mgr 所在节点要么是好的,要么可以很快重启. 如果不能重启,可用另一好的节点替换它,并且新节点能够访问存储元数据的磁盘(双端口磁盘或者拆卸).

当一个 mgr 崩溃后,DCFS2 的故障恢复协议要求先重启该 mgr,并令其进入恢复流程. 它先向其他 mgr 广播一个通知,其他 mgr 收到这个通知后,便也进入恢复状态,此时,所有 mgr 都不响应来自客户端的新请求. 重启的 mgr 读日志文件,若是普通操作,则直接根据日志记录重做或撤消;若是分布式操作,则向相关的 mgr 发请求进行协商,根据协商结果重做或撤消. 当重启 mgr 完成恢复后,它向其他 mgr 广播一个通知,其他 mgr 收到这个通知后,对涉及崩溃 mgr 的分布式操作进行恢复. 当所有 mgr 恢复完成后,便回到正常状态,接收和处理客户节点来的请求.

5 性能评测

测试平台是一个 32 节点的集群. 每个节点有 2 个 CPU(AMD Opteron™ Processor 242),2GB 内存和一块 36.7GB 的 SCSI 硬盘(Seagate ST373307LC),运行 Turbolinux3.2.2(核心是 Linux-2.4.21-numa),节

点间通过百兆网和千兆网互连.

DCFS2 的配置为 2 个元数据服务器、1 个 IP-SAN 存储设备(通过 SuperNBD[22]软件实现)和 16 个客户端节点. 数据和元数据都使用千兆网传输.

5.1 异步日志对性能的影响

DCFS2 的读写性能使用 iozone[23]来测试,采用集群测试模式和 Stone wall 模式. 测试时,读写总量为 528MB,读写粒度为 1MB. 测试结果如图 3 和图 4 所示. 由于 DCFS2 并不实时修改文件的访问时间(atime),记录日志对读性能没有影响,我们的测试结果与此一致. 写文件时,由于 DCFS2 采用异步方式来修改文件的长度和修改时间(mtime),因此异步日志对写性能的影响也非常小,几乎与不记录日志时性能一样.

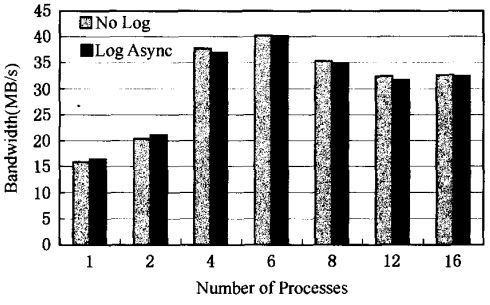

Fig. 3 Read performance of DCFS2.

图 3 DCFS2 的读性能

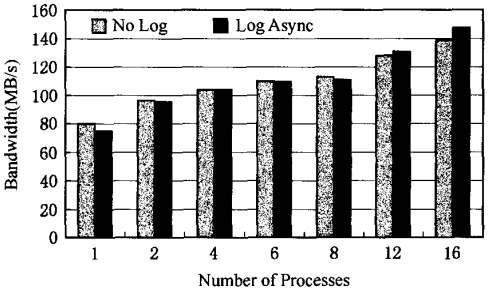

Fig. 4 Write performance of DCFS2.

图 4 DCFS2 的写性能

DCFS2 的元数据性能采用 postmark[24]来测试. 测试时每个节点运行一个 postmark 进程. 所有进程总共创建 48 个目录,初始化阶段在每个目录下生成 1000 个文件,事务总数为 48×2000 个. Postmark 的测试结果如图 5、图 6 和图 7 所示. 异步日志时,文件创建和删除吞吐率有所下降,性能损失在 10%左右,如表 1 所示. 虽然异步日志对元数据处理性能有影响,但是对事务吞吐率的影响不大,如图 7 所示.

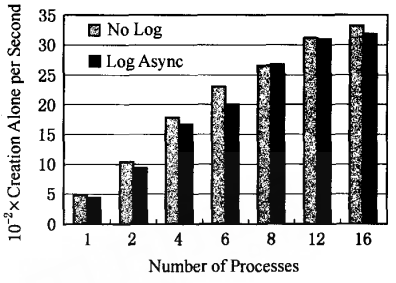

Fig. 5 Throughput of creation alone.

图 5 DCFS2 的文件创建吞吐率(Creation alone)

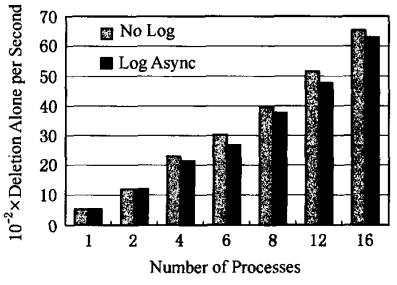

Fig. 6 Throughput of deletion alone.

图 6 DCFS2 的文件删除吞吐率(Deletion alone)

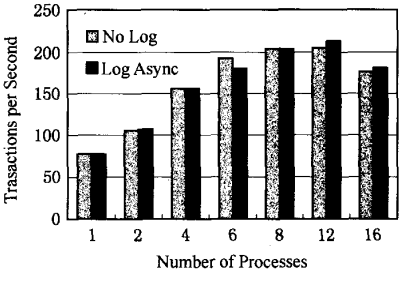

Fig. 7 Throughput of transaction.

图 7 DCFS2 的事务吞吐率

Table 1 The Percentage of File Creation/Deletion Throughput with Asynchronous Logging to that Without Logging

表 1 异步日志时性能占不记录日志时性能的百分比

Number of Client Nodes	File Creation(%)	File Deletion (%)
1	90.91	102.11
2	90.23	99.83
4	93.55	93.15
6	87.34	88.29
8	100.53	95.08
12	99.20	92.52
16	95.96	96.46

5.2　恢复性能

DCFS2 的恢复时间与文件系统的大小无关,而只与日志文件大小和日志文件中所含的未完成的分布式操作个数有关. 由于未完成的分布式操作需要与其他 mgr 协商才能完成. 因此日志文件中这样的操作越多所需的恢复时间就越长.

表 2 给出了不同日志文件大小所需要的恢复时间,这些日志文件中未完成分布式操作个数都相同,反映的是日志文件长度对恢复时间的影响. 恢复时间随日志文件长度的增长而同步增长. 表 3 给出的是未完成分布式操作所占比例不同时所需要的恢复时间,它反映的是未完成的分布式操作的恢复性能,及其对整个恢复效率的影响. 如表 3 所示,未完成分布式操作个数增长 10 倍,恢复时间只增加 1 倍. 这说明分布式操作的恢复效率较高.

Table 2　Recovery Time with Varied Log File Size
表 2　不同日志文件长度的恢复时间

Size of the Log File (MB)	Number of Records in the Log File	Number of Records for Distributed Operations	Recovery Time (s)
10	71998	721	4.144728
20	143607	721	7.871283
30	215332	721	11.649644
40	287087	721	15.732586
50	359044	721	19.258066

Table 3　Recovery Time with Varied Number of Distributed Operations
表 3　不同分布式操作个数时的恢复时间

Size of the Log File (MB)	Number of Records in the Log File	Number of Records for Distributed Operations	Percentage of Distributed Operations (%)	Recovery Time(s)
10	71763	0	0	4.124264
10	71998	720	1	4.144728
10	72262	1446	2	4.388831
10	72508	2178	3	4.637616
10	72760	2912	4	4.887085
10	73019	3655	5	5.330844
10	74338	7440	10	9.255075

总的来说,恢复时间主要取决于日志文件长度,一般为几秒至几十秒.

6　结　　论

为了适应集群应用对文件系统的性能和扩展性的需求,基于 IP-SAN 的共享存储机群文件系统 DCFS2 采用多元数据服务器结构,元数据存储在各个元数据服务器的本地文件系统中. DCFS2 采用固定粒度的基于目录的动态元数据分布策略,有些元数据操作需要多个元数据服务器协作来完成. 为解决分布式元数据处理下的元数据一致性问题,DCFS2 采用分布式日志技术和改进的两阶段提交协议保证了在单个元数据服务器崩溃或客户节点崩溃时,可快速恢复 DCFS2 文件系统的元数据一致性.

性能分析表明,在采用异步日志方式时,DCFS2 的读写带宽与不记录日志时的几乎无差别,文件创建和删除吞吐率可达到不记录日志时的 90% 左右. 而且文件系统的恢复时间与文件系统大小无关,只与日志文件大小和日志文件中所含的未完成的分布式操作个数有关,一般只需要几秒到几十秒. 因此,DCFS2 的基于分布式日志和两阶段提交的元数据处理策略能够提供高的 I/O 性能,并能够保证在元数据服务器失效后文件系统快速恢复.

我们未来的工作是研究如何更好地发挥多个元数据服务器的存储和处理能力,以及实现高可用的 DCFS2.

参　考　文　献

1　R. J. Morris, B. J. Truskowski. The evolution of storage systems. IBM Systems Journal, 2003, 42(2): 205~217

2　P. Sarkar, K. Voruganti, K. Meth, *et al*. Internet protocol storage area network. IBM Systems Journal, 2003, 42(2): 218~231

3　K. W. Preslan, A. Barry, J. Brassow, *et al*. Implementing journaling in a Linux shared disk file system. The 8th NASA Goddard Conf. Mass Storage Systemsand Technologies in Cooperation with the Seventeenth IEEE Symposium on Mass Storage Systems, Maryland, 2000

4　Laura Shepard, Eric Eppe. SGI InfiniteStorage Shared Filesystem CXFS™: A High-Performance, Multi-OS SAN File System from SGI. Mountain View, CA: Silicon Graphics, Inc. 2004

5　F. Schmuck, R. Haskin. GPFS: A shared-disk file system for large computing clusters. The First USENIX Conf. File and Storage(FAST2002), Monterey, 2002

6　J. Menon, D. A. Pease, R. Rees, *et al*. IBM storage tank—A heterogeneous scalable SAN file system. IBM Systems Journal, 2003, 42(2): 250~267

7　P. J. Braam. The Lustre Storage Architecture. Medford, MA: Cluster File Systems, Inc. 2004

8 Uresh Vahalia. UNIX Internals: The New Frontiers. Englewood Cliffs, NJ: Prentice-Hall, 1996

9 J. Gray. Notes on data base operating systems. In: R. Bayer, R. M. Graham, G. Seegmüller, eds, Operating Systems: An Advanced Course, Lecture Notes on Computer Science 60. New York: Springer-Verlag, 1978. 393~481

10 T. Haerder, A. Reuter. Principles of transaction-oriented database recovery. ACM Computing Surveys, 1983, 15(4): 287~317

11 M. K. McKusick, T. J. Kowalski. FSCK—The UNIX file system check program. In: 4.4 BSD System Manager's Manual. Sebastopol: O'Reilly, 1994

12 S. Tweedie. Journaling the Linux ext2fs file system. The 4th Annual LinuxExpo, Durham, 1998

13 J. Gray, A. Reuter. Trans. Processing: Concepts and Techniques. New York: Morgan Kaufman, 1993

14 G. Ganger, M. McKusick, C. Soules, *et al*. Soft updates: A solution to the metadata update problem in file systems. ACM Trans. Computer Systems, 2000, 18(2): 127~153

15 L. Soares, O. Krieger, D. Silva. Meta-data snapshotting: A simple mechanism for file system consistency. Int'l Workshop on Storage Network Architecture and Parallel I/O s held with 12th Int'l Conf. Parallel Architectures and Compilation Techniques, New Orleans, 2003

16 M. Baker. Fast crash recovery in distributed file systems: [Ph. D. dissertation]: Berkeley, CA: University of California Berkeley, 1994

17 M. Devarakonda, B. Kish, A. Mohindra. Recovery in the Calypso file system. ACM Trans. Computer Systems, 1996, 14(3): 287~310

18 M. Bancroft, N. Bear, *et al*. Functionality and performance evaluation of file systems for storage area networks (SAN). The 8th NASA Goddard Conf. Mass Storage Systems and Technologies in cooperation with the Seventeenth IEEE Symposium on Mass Storage Systems, Maryland, 2000

19 PVFS2 Development Team. Parallel Virtue File System, Version 2. http://www.pvfs.org/pvfs2/pvfs2-guide.html, 2003-09-23

20 Tom Ruwart. Storage on the lunatic fringe. The special invited talk. The 2nd USENIX Conf. File and Storage Technologies (FAST2003), San Francisco, 2003

21 Zhihua Fan, Jin Xiong, Jie Ma. A failure recovery mechanism for distributed metadata servers in DCFS2. The 7th Int'l Conf. High-Performance Computing and Grid in Asia Pacific Region (HPC-Asia 2004), Omiya Sonic City, 2004

22 Rongfeng Tang, Dan Meng, Jin Xiong. SuperNBD: An efficient network storage software for cluster. The IFIP Int'l Conf. Network and Parallel Computing, Wuhan, 2004

23 W. D. Norcott. Iozone File System Benchmark. http://www.iozone.org/docs/IOzone_msword_98.pdf, 2005

24 Jeffrey Katcher. Postmark: A new file system benchmark. Network Appliance, Inc. Tech Rep: TR3022, 1997

Xiong Jin, born in 1968. She is currently associate professor and also Ph. D. candidate. Her research interests include cluster file systems, networked storage systems and high performance I/O.

熊劲,1968 年生,副研究员,博士研究生,主要研究方向为机群文件系统、网络存储和高性能 I/O.

Fan Zhihua, born in 1978. He is currently Ph. D. candidate. His research interests include cluster file system high availability and performance evaluation.

范志华,1978 年生,博士研究生,主要研究方向为机群文件系统的高可用性和性能评价.

Ma Jie, born in 1975. He is currently associate professor and also master supervisor. His research interests include high performance computer architecture, operating system, high performance communication protocol and parallel computing.

马捷,1975 年生,副研究员,硕士生导师,主要研究方向为高性能计算机体系结构、操作系统、高性能通信协议和并行计算.

Tang Rongfeng, born in 1979. He is Ph. D candidate. His research interests include distributed file systems, network storage and operation systems.

唐荣锋,1979 年生,博士研究生,主要研究方向为分布式文件系统、网络存储、操作系统.

Li Hui, born in 1980. He is a M. S candidate. His research interests include cluster file system and cluster operation system.

李晖,1980 年生,硕士研究生,主要研究方向为机群文件系统、机群操作系统.

Meng Dan, born in 1965. He is currently professor and also Ph.D. supervisor. His research interests include high performance computer architecture, operating system, high performance communication protocol, distributed file system and storage system.

孟丹,1965 年生,研究员,博士生导师,主要研究方向为高性能计算机体系结构、操作系统、高效通信协议、分布式文件系统和存储.

Research Background

High-end computing applications require cluster file systems to provide higher and higher performance, larger and larger capacity and scalability. Traditional single-server and central-controlled cluster file systems cannot satisfy these requirements. Instead, cluster file systems based on distributed networked storages and multiple metadata servers are required. We developed a scalable cluster file system DCFS2 for the Dawning 4000A Super-server. It is based on multiple IP-SAN storage devices and multiple metadata servers. Metadata distribution and metadata consistency are two key issues in such distributed metadata processing. We use distributed logging and the two-phase commit protocol to solve the metadataconsistency problem in DCFS2. And the performance results show that distributed logging in DCFS2 ensures high I/O performance and fast recovery from a metadata server failure. Our work is supported by the National High Technology Development 863 Program of China under Grant No. 2002AA1Z2102 and grant No.2002AA104410.

DCFS2的元数据一致性策略

作者： 熊劲， 范志华， 马捷， 唐荣锋， 李晖， 孟丹， XIONG Jin， Fan Zhihua， Ma Jie， Tang Rongfeng， Li Hui， MENG Dan

作者单位： 熊劲,范志华,唐荣锋,李晖,XIONG Jin,Fan Zhihua,Tang Rongfeng,Li Hui(中国科学院计算技术研究所国家智能计算机研究开发中心,北京,100080;中国科学院研究生院,北京,100039)， 马捷,孟丹,Ma Jie,MENG Dan(中国科学院计算技术研究所国家智能计算机研究开发中心,北京,100080)

刊名： 计算机研究与发展 ISTIC EI PKU

英文刊名： JOURNAL OF COMPUTER RESEARCH AND DEVELOPMENT

年，卷(期)： 2005，42(6)

被引用次数： 7次

参考文献(24条)

1. R. J. Morris. B. J. Truskowski The evolution of storage systems 2003(02)

2. P Sarkar. K. Voruganti. K Meth Internet protocol storage area network 2003(02)

3. K. W. Preslan. A Barry. J Brassow Implementing journaling in a Linux shared disk file system 2000

4. Laura Shepard. Eric Eppe SGI InfiniteStorage Shared Filesystem CXFSTM: A High-Performance, Multi-OS SAN File System from SGI 2004

5. F Schmuck. R. Haskin GPFS: A shared-disk file system for large computing clusters 2002

6. J Menon. D. A. Pease. R Rees IBM storage tank-A heterogeneous scalable SAN file system 2003(02)

7. P. J. Braam The Lustre Storage Architecture 2004

8. Uresh Vahalia UNIX Internals: The New Frontiers 1996

9. J Gray Notes on data base operating systems 1978

10. T. Haerder. A. Reuter Principles of transaction-oriented database recovery 1983(04)

11. M.K. McKusick. T. J. Kowalski FSCK-The UNIX file system check program 1994

12. S. Tweedie Journaling the Linux ext2fs file system 1998

13. J Gray. A. Reuter Processing: Concepts and Techniques 1993

14. G. Ganger. M. McKusick. C Soules Soft updates: A solution to the metadata update problem in file systems 2000(02)

15. L. Soares. O. Krieger. D Silva Meta-data snapshotting: A simple mechanism for file system consistency 2003

16. M Baker Fast crash recovery in distributed file systems 1994

17. M. Devarakonda. B. Kish. A Mohindra Recovery in the Calypso file system 1996(03)

18. M. Bancroft. N. Bear Functionality and performance evaluation of file systems for storage area networks (SAN) 2000

19. PVFS2 Development Team Parallel Virtue File System, Version 2 2003

20. Tom Ruwart Storage on the lunatic fringe The special invited talk 2003

21. Zhihua Fan. Jin Xiong. Jie Ma A failure recovery mechanism for distributed metadata servers in DCFS2 2004

22. Rongfeng Tang. Dan Meng. Jin Xiong SuperNBD: An efficient network storage software for cluster 2004

23. W. D. Norcott Iozone File System Benchmark 2005

24. Jeffrey Katcher Postmark: A new file system benchmark 1997

相似文献(1条)

1. 学位论文 李晖 基于日志的机群文件系统高可用关键技术研究 2005

近年来机群系统以其低成本、高性能而逐渐成为高性能计算的主流平台、作为解决机群系统外存储瓶颈上的有效手段的机群文件系统因此得到了很大的发展。一个机群文件系统必须要满足机群计算环境的需要，为应用提供高性能、可扩展、高可用的文件服务。由于机群文件系统本身结构复杂，实现复杂而且整个系统规模很大，这些因素就决定了对高可用技术的依赖。本文将研究基于日志的机群文件系统高可用的关键问题以及解决策略，同时给出了一些评价方法以及具体的评测结果。具体内容以及研究成果如下：

(1)研究了基于日志的机群文件系统高可用技术的关键问题。本文分析了不同类型的机群文件系统的高可用需求以及高可用技术，对机群文件系统高可用相关的概念进行了介绍，描述了机群文件系统高可用领域的研究内容，并在分析的基础上提出了基于日志的机群文件系统高可用技术，分析了其中的关键问题，给出了相应的解决策略，并对其正确性和完备性给予了证明。

(2)实现了DCFS2机群文件系统高可用模块。作为文中策略的一个实际应用，本文给出了DCFS2机群文件系统高可用的设计与实现技术，给出了系统中利用日志来保证机群文件系统一致性的方法。主要内容包括：以DCFS2机群文件系统为原型系统，研究了单一以及多个元数据服务器下如何使用日志来保证文件系统的一致性；研究了机群文件系统日志对元数据操作的性能影响；研究了客户端的高可用问题。

(3)提出了机群文件系统高可用性的分级的定义。机群文件系统的高可用性的高低一直缺乏有效的定性或定量的分析方法，由于软件系统不能象硬件系统那样进行定量分析，我们根据机群文件系统的应用模式，将影响机群文件系统高可用性的因素进行分析，以机群文件系统的故障因素和恢复目标因素为线索，采用分级的方法对机群文件系统高可用性进行了定义，提出了机群文件系统高可用性的分级的定义。

(4)对基于日志的高可用技术进行了评价。目前在高可用技术的评价上尚没有完善的评价体系，本文从功能性，正确性，性能，恢复时间等多个方面对基于日志的高可用技术进行了评价，并给出了各种情况下的具体的测试结果。 文中还讨论了下一步的研究方向，包括多节点故障恢复等方面。

引证文献(7条)

1. 海深. 周燕艳 嵌入式文件系统故障恢复机制设计[期刊论文]-计算机工程与设计 2009(9)

2. 张军伟. 贾瑞勇. 贾亚军. 张建刚. 许鲁 蓝鲸集群文件系统中资源交互一致性协议[期刊论文]-计算机工程 2008(11)

3. 田俊峰. 于洪芬. 宋玮玮 小规模集群文件系统中两级元数据服务器的设计与实现[期刊论文]-小型微型计算机系统 2007(6)

4. LIU Yuling. YU Hongfen. SONG Weiwei Design and Implementation of Two-Level Metadata Server in Small-Scale Cluster File System[期刊论文]-武汉大学学报（英文版） 2006(6)

5. 海深. 周燕艳 嵌入式文件系统一致性设计[期刊论文]-铜陵学院学报 2006(4)

6. 海深. 陆阳. 袁菲 嵌入式存储系统恢复机制的设计与实现[期刊论文]-计算机工程 2006(24)

7. 海深 嵌入式系统的存储卡接口技术研究[学位论文]硕士 2006