4.7 缓存有效性的影响评估

为说明元数据分布信息缓存的有效性，本节从两个方面进行评估。首先评估不同缓存命中率对系统聚合吞吐率、BS 负载的影响。然后，评估典型应用模式下，应用能够获得的缓存命中率情况。

4.7.1 缓存命中率的影响

通过调整不活跃元数据分布信息的释放参数，获得 0%、50%、90%、95%、98%和100%的缓存命中率，统计客户聚合请求吞吐率、需要与 BS 的通信数目比例。0%缓存命中率和 100%命中率分别模拟最坏和最好缓存命中情况。

为避免服务器数量庞大带来的竞争影响，测试环境由 1 个客户端、1 个元数据服务器、1 个具有 120GB 的 SATA 硬盘的存储设备和 1 个绑定服务器构成。每个服务器的 CPU 是 Intel® Xeon® 3.4GHz, 3GB 内存，操作系统是 RedHat® Linux® 8.0。

测试由 postmark 驱动。Postmark 参数是 10000 个 4KB 至 16KB 的文件，10 个子目录，50000 个事务，块读写大小为 4KB。Postmark 首先需要生成 10000 个测试文件。在创建测试文件时，MS 需要与 BS 进行通信，完成 MS 元数据分布信息缓存的填充。

图 4.9 是不同缓存命中率下，postmark 聚合事务吞吐率，单位是每秒完成的事务数量。从图中可以看出，随着缓存命中率的提高，请求吞吐率明显提高。当命中率达到 95%后，缓存对系统的事务吞吐率的影响比较弱，事务吞吐率逐渐平稳。

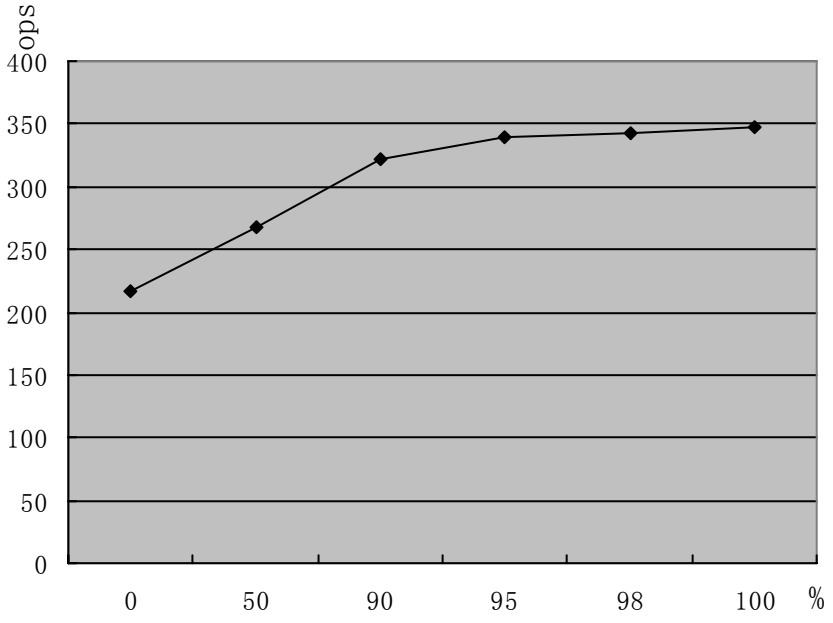

图 4.9 聚合事务吞吐率随缓存命中率变化

缓存中有效的元数据分布信息可以直接用来决定客户端请求的处理。在缓存信息无效时，通过请求 BS，刷新缓存信息。

BS 的负载压力可以通过请求比率来描述：

$$ratio = REQUEST(ms - bs) / REQUEST(as - ms)。$$

其中 REQUEST（as-ms）是 AS 发送给 MS 的请求数量，REQUEST(ms-bs)是 MS 与 BS 的通信数量。

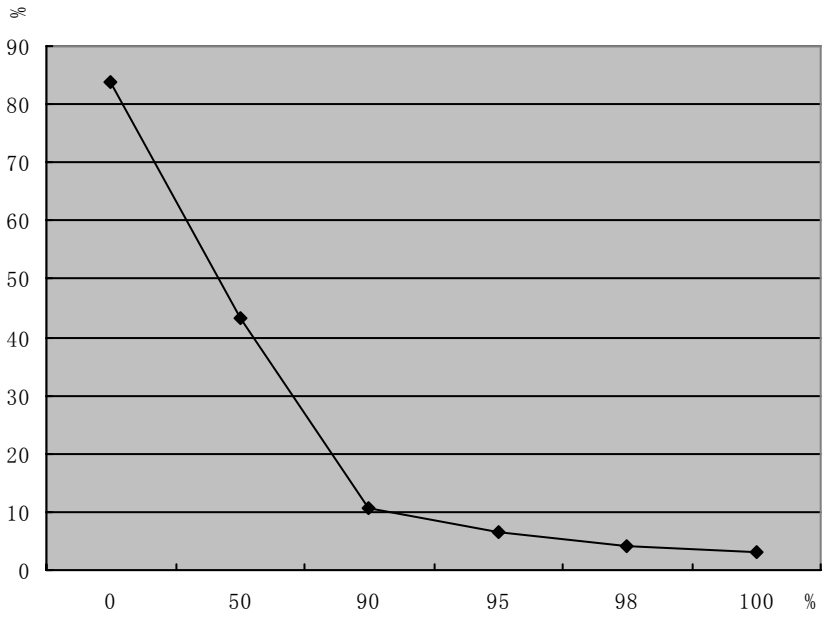

图 4.10 BS 通信负载随缓存命中率变化

图 4.10 是根据上述公式计算的各种缓存命中率下的请求比率，每个计算值包含创建测试文件集的 10000 次通信。从图可以看出，缓存命中率对 BS 的负载压力的影响非常明显。

从上述两个图还可以看出，当缓存命中率达到一定值后，进一步的提高对系统的促进作用将不太明显，为缓存管理的优化提供参考。

4.7.2 不同应用的缓存命中率

为评估不同应用可能获得的缓存命中率，本部分评估创建/删除空文件、应用以共享模式访问同一测试文件集、以及应用以私有模式访问不同测试文件集三种应用下，各个服务器的缓存命中率。测试采用 7 个客户端、6 个元数据服务器的配置，服务器的硬件配置跟上述评估相同。结果如表 4.8 所示。

表 4.8 不同应用的缓存命中率

	MS1	MS2	MS3	MS4	MS5	MS6	*平均值*
创建/删除空文件	87.49	87.49	87.52	87.49	87.49	47.06	80.76
共享模式	97.81	97.24	97.06	96.99	97.6	93.49	96.70
私有模式	96.21	96.19	96.22	96.22	96.22	96.18	96.21

由于创建空文件需要 BS 决定新创建文件的元数据请求的分布，每一个创建请求都

需要与 BS 通信。创建完成的文件并没有被反复访问，所以，缓存的作用不明显，其缓存命中率非常低，约为 81%。

共享模式是 7 个客户端同时通过“find . -exec stat {} \;”命令访问同一个目录下的文件。元数据请求以单个活跃文件为单位，并且同一个文件被反复访问的次数较多。在一个客户端驱动了请求分布并获得结果后，其他的客户端可以利用缓存信息，处理元数据请求。所以，其缓存命中率较高，达到 96.70%。

私有模式是各个客户端在各自目录下进行文件的创建和读写请求。每个元数据的请求分布结果仅对本客户端有利。所以，其缓存命中率相对于共享模式低，但由于缓存信息能够重复利用，其缓存命中率比文件创建/删除高得多。

综上所述，BWMMS 元数据分布信息缓存管理能够为具体应用提供有效的支持。

4.8 本章小结

因为其简单性和易扩展性，集中方式的元数据请求分布决策机制将成为重要的元数据请求分布管理机制。

用户元数据访问表现出的局部活跃性特征决定了缓存机制的有效性。本章结合 BWMMS 讨论集中元数据请求分布管理机制中有效的元数据分布信息缓存管理。

在元数据分布信息缓存管理中，通过哈希加速查找等传统的缓存管理技术仍然适用。在此基础上，BWMMS 还需要根据元数据的活跃性管理元数据的分布信息缓存。它需要结合元数据请求的语义，描述、验证元数据分布信息状态转换的活跃性。

BWMMS 的不活跃元数据分布信息项替换策略，结合内存开销压力、元数据的活跃性和用户访问的局部性特征，优先替换不活跃的非宿主权限信息，以支持宿主权限信息的有效访问。

元数据分布信息缓存的命中率将直接影响BS的负载和元数据请求处理的时间延迟。在本章的最后，通过调整不活跃元数据分布信息的释放参数，不同缓存命中率对系统聚合吞吐率和 BS 压力的影响得到评估。结果表明，元数据缓存管理有效性对 BS 的压力和系统元数据请求处理吞吐率的影响明显，达到一定值后，缓存命中率对系统的促进作用将不明显。同时，BWMMS 当前提供的缓存管理策略能够很好地支持应用的需要。

第五章 基于宿主改变的请求原子性保证协议

更改文件系统的元数据请求要求其结果具有原子性。在集群环境中，涉及多个元数据的元数据请求可能需要跨服务器处理，元数据请求的原子性保证问题更加突出。尽管跨服务器请求的比例极小，但两阶段提交等传统的请求原子性保证协议对系统整体性能和错误恢复能力的影响比较大。探讨轻量级协议、降低比例极小的跨服务器请求对系统的影响，对提高系统扩展能力有着重要作用。

本章讨论基于集中共享存储结构、对称服务器结构的跨服务器元数据请求的原子性保证问题。通过改变活跃元数据的宿主，完成元数据在服务器间的迁移，跨服务器请求涉及的活跃元数据集中到单个服务器。在本地文件系统技术支持下，元数据请求由单个元数据服务器集中处理，有效解决其原子性保证问题。

5.1 问题描述

在集群环境中，元数据请求分布算法可能将访问上下文相关的元数据分布到不同的服务器，导致涉及多个元数据的请求需要跨元数据服务器协同处理。系统故障可能导致文件系统的不完整更新。如何通过保证单个请求的原子性，实现文件系统的原子更新，是分布式文件系统研究的重要问题。

为减少文件系统的不一致，出现故障的文件系统需要尽可能地修复。FSCK [McKusick1994]是本地文件系统常用的修复方法。它通过对比资源使用情况，完成文件系统元数据的修复。FSCK 需要搜索整个文件系统，其开销随文件系统规模线性增长。

针对 FSCK 的不足，日志技术[Tweedie1998][Best2002][Chinner2006][ReiserFS]使用额外的存储空间记录文件系统的更改，以缩小故障恢复的检测范围。它以单个请求的事务性保证为单位，将文件请求结果首先写到日志中，然后在更新存储设备后清除日志内容。由于日志技术需要两次设备写操作，且日志本身的内容需要同步写，其性能相对较低。为解决日志技术的性能较低问题，软更新技术[Ganger2000]通过将请求更改的元数据排序，按照请求处理的时间顺序更新设备，保证请求结果的事务性。

在分布式系统中，两阶段提交及其变种协议[Bernstein1987][Mullender1990][Samaras1993][Luckham1995]常用来保证分布式请求的原子性。尽管两阶段协议本身时延可以接受。但其协议过程的锁机制、复杂的错误恢复协议，导致它对系统整体性能和错误恢复能力的影响较大[Liu1994]。在新的存储架构中，探讨简单高效的请求原子性保证协议，显得非常必要。