CEPH动态元数据管理方法分析与改进

冯幼乐[1] 朱六璋[2]
(1. 中国科技大学信息科学技术学院自动化系 2. 安徽电力继远软件公司)

摘 要：分布式文件系统(CEPH)的动态元数据管理方法极大地提高了元数据服务器的性能和扩展性。本文首先分析了CEPH元数据服务器集群中的负载均衡策略，针对其在异构元数据服务器和网络延迟较大时存在的问题提出了改进方案。实验证明，改进后的方法不仅提高了系统的性能，扩大了系统的使用范围。

关键词：分布式文件系统；元数据管理；负载均衡

Analysis and Improvement of CEPH Dynamic Metadata Management

Feng Youle[1] Zhu Liuzhang[2]
(1.Dept. of Automation, University of Science and Technology of China 2.Anhui Electric Power Jiyuan Software Co. Ltd)

Abstract: The dynamic metadata management of CEPH has significantly increased the performance and scalability of metadata server cluster. We first introduce the migration algorithm of directory subtrees, and then propose an improved algorithm to resolve the problem existed in larger heterogeneous MDS and network latency. With the improved migration algorithm, not only the migration process is less called and unnecessary network cost is saved, but also the performance in large latency network is improved, which makes CEPH can be used in more environments.

Key words: distributed file system; metadata management; load balance

0 引言

分布式文件系统(CEPH)通过将多台机器的资源组织起来，对外提供统一的、大容量、高性能、高可靠的文件服务，满足了大规模应用的要求，是目前存储领域研究的重点和难点。CEPH[1-3]通常由元数据服务器(MetaData Server,MDS)集群和存储服务器集群构成。统计表明：在文件系统的访问中，对元数据的访问次数占全部访问次数的50-80%[4]。为应对大量的元数据操作请求，保障良好的性能和扩展性，元数据的管理方式与MDS集群的负载均衡策略极其重要。

现有的元数据的管理方式主要为集中存储分布式处理[2,5]和分布式存储分布式处理[1]两种方式。第一种方式元数据保存在共享的存储设备中，元数据和MDS的对应关系是动态划分的，每台MDS负责缓存一部分目录子树并处理相应的元数据操作；动态划分在出现热点数据或者MDS负载过高时可以很方便地进行目录子树的复制和迁移，易于扩展MDS和负载均衡，但实现较为复杂。第二种方式元数据按一定的方式分布到MDS上，每台MDS负责处理存储在其上的元数据请求，元数据与MDS的对应关系一旦确立就不会改变；这种方式实现较为简单，元数据的划分一般采用静态子树划分或者hash方法。由于目录子树并不是均衡增长，静态子树划分很容易出现负载不均衡；hash方法在初始的时候，可以通过良好的设计使得元数据在MDS间均匀分布，整体负载比较均衡，但是在增加或减少MDS的时候，需要重新调整hash函数，这两种方法在负载均衡时都会导致大量的数据迁移。

CEPH[5]采用集中存储分布式处理的动态元数据管理方法，通过对目录子树的复制和迁移实现了MDS的负载均衡，具有良好的性能和扩展性。本文首先分析了CEPH元数据服务器集群中的负载均衡策略，针对其在异构元数据服务器和网络延迟较大时存在的问题提出了改进方案。实验证明，改进后的方法提高了系统的性能，而且扩大了系统的使用范围。本文组织如下：第1节介绍CEPH文件系统的系统架构及其元数据管理器集群的负载均衡策略；第2节针对其在异构和高网络延迟环境下存在的问题提出改进方案，第3节对其进行实验分析，最后给出总结。

1 CEPH文件系统架构及其元数据服务器集群负载均衡策略

1.1 CEPH文件系统体系架构

CEPH采用元数据和文件数据分开处理的体系结构，由三个子系统组成：客户端(CLIENT)，元数据服务器集群(Metadata Cluster)和对象存储集群(Object Storage Cluster)。MDS维护全局的名字空间，负责处理元数据相关的请求以及相应的权限管理；对象存储设备负责文件数据和元数据的存储，为客户端和MDS提供统一的数据读写服务。在CEPH中，元数据保存在对象存储设备中，MDS利用缓存的数据对外提供服务。由于MDS本身并不存储数据，所以可以很方便地进行目录子树的复制迁移以实现负载均衡。

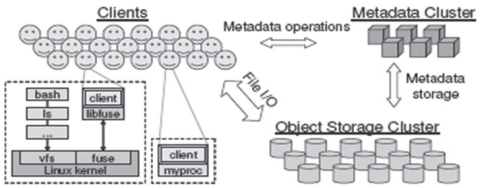

图1 CEPH文件系统架构[5]

1.2 元数据的负载均衡策略

1.2.1 基本定义

节点负载(1)：采用一段时间内的CPU占用率的平均值作为节点负载的度量。

元数据热度(p)：每当有来自用户的元数据请求时，对应的元数据热度增一。考虑到不同时间内的元数据请求对热度的影响应该是不同的，元数据的热度会随时间衰减。元数据访问的时间间隔越长，对热度的影响应该越小。对某一元数据访问时，其在目录树上的祖先节点的热度亦会受到影响，即热度的更新会向上传播，传播的量逐层衰减。综合以上考虑，得到元数据节点的热度

计算公式如下：

$$p_{new} = p_{old} * f(\Delta t) + 1$$

$$p_{ancestor\_new} = p_{ancestor\_old} * f(\Delta t) + 1/2^n$$

其中 Δt 为当前时间与上次计算热度的时间差，f 为衰减函数，n 为被访问元数据与祖先节点的目录层次差。

1.2.2 子树复制

当元数据的热度超出阈值后，将会启动子树复制流程，系统在其他的MDS上创建缓存副本。子树复制可以解决热点数据的访问问题

1.2.3 子树迁移

MDS定期上报自己的负载信息，当单台MDS的负载持续一段时间高于平均负载一定范围时，将会启动子树迁移流程：

(1)根据收集到的全局负载信息，选择一负载较轻的MDS作为子树迁移的目标MDS，根据过载MDS和轻载MDS的负载与平均负载的差值决定迁移负载的数量；

(2)根据过载MDS的整体热度与负载的关系，决定迁移目录子树的热度；

(3)从过载MDS中选择相应热度的目录子树进行迁移。

子树迁移策略解决了负载在MDS间分布不均的问题。

2 CEPH元数据管理方法的改进

CEPH的元数据管理策略可以很好地应对热点数据访问以及负载分布不均的问题，但其实现的子树迁移算法却有如下问题：

(1)迁移算法默认了各MDS能力相同。实际上同样的元数据访问在不同MDS上造成的负载是不同的。目录子树从高负载、能力强的MDS上迁移到低负载、能力弱的机器上时，能力弱的MDS可能接受过多的元数据负载，很快成为新的系统瓶颈；虽然在新一轮的迁移活动中，目录子树会从能力弱的MDS迁移出去，但仍然浪费了大量的网络流量，延长了负载均衡的时间。当目录子树从高负载、能力弱的MDS上迁移到低负载、能力强的机器上时，虽然不会造成网络流量的浪费，但实际上可以迁移更多的热度过去，没有充分利用迁移的机会。

(2)子树迁移算法的目标是达到全局负载的平均，在网络延迟较大的情况下，这一目标的实现需要花费较大代价。由于网络延迟的影响，收集全局负载信息可能需要较长的时间，而且在选择迁移目标时，并没有考虑网络延迟的状况，最后选出的目标对象间可能传输代价较大，极大地影响了负载均衡算法的效率。

本文将针对以上两个问题对子树迁移算法进行改进，在迁移时综合机器能力与网络状况合理选择迁移目标和热度，提高了迁移性能，扩大了CEPH的使用范围。

2.1 基本定义

节点负载的重新定义(L)：节点负载除了与CPU有关外，还和内存占用、带宽和I/O等有关。考虑到CEPH元数据服务器的特性，我们采用CPU，内存和带宽占用情况的加权和重新定义负载。其中 l_{cpu}、l_{mem}，l_{bw} 分别为CPU、内存和带宽的占用情况，α、β、γ 为相应的加权系数，根据应用类型的不同通过反复试验得出。

$$L = \alpha * l_{cpu} + \beta * l_{mem} + \gamma * l_{bw}$$

节点能力的评估(s)：节点能力一般用其处理器能力、内存容量等参数的加权和来表示，但这种方法需要针对每台MDS做实验才能得出具体的加权系数，不同应用系数还有改变。本文采用MDS的元数据总热度(P)与负载(L)的比值来定义节点能力。容易看出，处理的元数据请求越多，产生的负载越小，节点能力越大，这是与实际状况相符的。

$$s = P/L$$

传输代价(c)：MDS节点间传输单位数据需要花费的代价。大部分节点间c的计算在两台服务器有数据交换的时候捎带完成，考虑到网络结构的稳定性，c的更新并不频繁。c还可以通过手动配置直接指定，对于新加入的MDS，通过指定其到某些MDS的传输代价，可以加快其融入MDS集群的速度。

MDS的分区：通过对c进行聚类，将MDS分成不同的区域，区域内MDS间的传输代价明显小于同区域外MDS通信的传输代价。区域内距离聚类中心最近的MDS作为决策MDS，负责收集局域内MDS的信息和迁移决策。

2.2 子树迁移

MDS定期向区域内的决策MDS上报自身的负载和热度，当单台MDS的负载持续一段时间高于区域内平均负载一定范围时，将会启动子树迁移流程：

(1)根据过载MDS负载L_i与区域平均负载 $\overline{L}$ 的差值计算需要迁移出的负载L_{out}。

(2)根据负载和热度信息，计算第j台轻载MDS能接受的最大负载 ΔL_j，$\Delta L_j = (\overline{L} - L_j)\frac{P_j}{L_j}\frac{L_i}{P_i}$ ，其中P_i、P_j分别为过载MDS和轻载MDS的热度信息。

(3)选择能接受负载最多的MDS作为目标MDS，迁移的负载量为相应的 ΔL_j与L_{out}中较小者。

(4)通知过载MDS迁移目标MDS和迁移的负载，过载MDS中选择相应热度的目录子树进行迁移。

3 实验及结果分析

在实验中，元数据管理系统有4台异构的MDS组成，其中MDS0和MDS1性能相同，MDS2和MDS3性能相同，MDS0的性能是MDS2的十倍。分别在两种网络环境下进行实验：首先4台MDS连接在同一个局域网内，比较两种负载均衡方法的性能；接下来MDS0和MDS2在同一局域网，MDS1和MDS3在同一局域网，两个局域网通过Internet连接起来，比较两种负载均衡方法的性能。通过向元数据管理系统连续发送6个小时的数据，记录负载均衡过程中各MDS的实时负载，迁移次数与通讯量等数据进行比较。

表1 算法性能比较

网络环境	迁移算法	迁移次数	迁移通讯总量	迁移花费时间	操作延时
局域网	改进前	12	50M	15min	小
	改进后	10	40M	12min	小
Internet 连接的局域网	改进前	12	50M	40min	大部分操作延时较大
	改进后	22	35M	15min	小部分操作延迟较大

在表1可以看出，在局域网环境下，改进后的算法起到减少迁移次数，降低网络通讯的作用。在广域网环境下，原有的算法在广域环境下迁移元数据，迁移流程耗时较长，而且元数据的跨区域分布，使得许多元数据操作需要跨区执行，延迟较大。改进后的算法使得元数据仅在区域内迁移，仅有少部分操作涉及区域外的数据，大部分操作在区域内即可完成，提高了用户体验。在广

域环境下，迁移次数较多是因为两个区域内同时进行了数据迁移。

4 总结

本文对CEPH元数据的负载均衡算法进行了研究，并提出一种改进方法。改进的算法通过对元数据服务器进行区域划分，在子树迁移时综合考虑负载和机器性能，优化了子树迁移目标的选择策略。实验证明，改进后的方法在异构元数据服务器和网络延迟较大的情况下，仍能发挥较好的性能，扩大了CEPH的适用范围。

参考文献：

[1] Braam P J.A Scalable,High-Performance File System [M]. Lustre Whitepaper Version 1.0,2002.
[2] 黄华. 蓝鲸分布式文件系统的资源管理[D]. 博士学位论文，中国科学院计算技术研究所, 2005. 5.
[3] Ghemawat S,Gobioff H,Shun-Tak L.The Google file system[C]//Proceedings of the Nineteenth ACM Symposium on Operating Systems Principles,2003 Oct 19-22,New York. New York:ACM Press,2003:29-43.
[4] Roselli D,Lorch J,Anderson T.A comparison of file system workloads[C]//Proceedings of the 2000 USENIX Annual Technical Conference,San Diego,CA,June 2000. USENIX Association:41-54.
[5] Sage A.Weil.Ceph.Reliable,scalable,and high-performance distributed storage[D].Santa Cruz:University of California, December,2007.

作者简介：

冯幼乐，男，1986年生，硕士研究生，研究方向为网络多媒体
手机：13645512517
电子信箱：fengyl03@163.com
联系地址：安徽省合肥市中国科学技术大学西区7号楼621（230027）
朱六璋，男，1969年生，安徽舒城人，高级工程师，从事电力应用软件研发技术管理、企业信息化项目相关技术应用研究工作

基金项目：

国家863课题"新一代业务运行管控协同支撑环境的开发(2008AA01A317)"资助

（上接第8页）

在考虑时间因素调整权重后，最相似案例为X2007X15。为简单起见，对得到的最相似案例使用空调整。调整前得经济损失为18540万元，与实际损失误差50.73%；调整后得经济损失为14340万元，与实际损失误差16.59%。时序调整后的权重在计算相似度上考虑较全面，得到的结果较未考虑时间因素的更接近实际值。

3 结论

本文针对案例推理灾害救助系统提出了一种基于时序的权重调整算法，考虑了自然灾害各特征属性对时间的敏感性，在寻找相似案例时，检索出的案例更为合理。但该算法本质上没有摆脱k-NN方法运行时间复杂度较高的缺点，随着案例库中案例数目的增加，算法复杂度将成指数增长。这时应该考虑对案例库建立归纳索引，这将在以后的工作中进行探讨。

参考文献：

[1] 杨 健, 杨晓光, 等. 一种基于k-NN的案例相似度权重调整算法[J].计算机工程与应用, 2007(23):12-15.
[2] 武民民, 宋良图. 基于替代算法的案例推理灾害救助系统[J]. 计算机系统应用, 2009, 18(4):13-15.
[3] 常春光, 崔建江, 等. 案例推理中案例调整技术的研究[J]. 系统仿真学报, 2004(6):149-154, 172.
[4] 蹇 明, 黄定轩. 无决策属性的多属性决策权重融合方法[J]. 西南交通大学学报, 2005(2):134-138.

作者简介：

姜枫(1984－)，男(汉族)，江苏常州市人，中国科学院物质科学研究院合肥智能机械研究所在读硕士研究生，研究方向为模式识别与智能系统。
电话：13637098541
电子信箱:raksasa@live.cn或jfjf@mail.ustc.edu.cn
通信地址：安徽省合肥市1130信箱8楼（230031）

基金项目：

1. 国家科技支撑计划：中国重大自然灾害风险防范技术，项目编号2008BAK50B08.
2. 国家科技支撑计划：灾害应急决策支持与远程会商协同技术研究，项目编号2008BAK49B05.