

一种多层次 Hadoop 平台设计

李兆兴, 马自堂

(解放军信息工程大学 密码工程学院, 河南 郑州 450000)

摘 要: 大数据和大数据中心规模的进一步扩大为 Hadoop 平台的可扩展性提出了新的要求. 在系统地分析了制约 Hadoop 平台性能和可扩展性因素的基础上, 提出了一种多层次 Hadoop 平台. 通过将 Hadoop 平台划分为若干区域, 形成全局-区域-节点的多层次结构. 由 Master 节点负责系统级的元数据管理和任务分发, 区域管理节点负责区域级的元数据管理和任务分发, 从而提高 Hadoop 平台可扩展性和容错性. 实验表明, 该平台可有效提高 Hadoop 的可扩展性, 具有一定的可行性.

关键词: Hadoop; NameNode; SeMNode; 多层次

中图分类号: TP391

文献标识码: A

文章编号: 1000-7180(2016)02-0027-04

A Multi-Layer Hadoop Platform

LI Zhao-xing, MA Zi-tang

(College of Cryptography Engineering, PLA Information Engineering University, Zhengzhou 450000, China)

Abstract: With the further expansion of Big Data and Big Data Centre, new extensibility requirements are proposed on the Hadoop Platform. In the base of systematically analysis constraints on performance and extensibility of Hadoop Platform, this paper presents a multi-level Hadoop Platform which has a structure of Global-Region-Node after being divided. In this platform, Master Node is responsible for metadata management and task development on the global level, and regional management node takes charge of metadata management and task development on the regional level, which improve both scalability and fault tolerance of Hadoop Platform. Experiments show that this platform can effectively promote the scalability of Hadoop and has certain feasibility.

Key words: Hadoop; NameNode; SeMNode; multi-layer

1 引言

随着数据规模的爆炸式增长, 基于 GFS 和 MapReduce 创建的 Hadoop^[1] 以其简单易用, 运行在廉价的 PC 集群上, 且具备一定容错性等特点, 已经在大数据分析领域中得到了日益广泛的应用. 如今, 包括 Facebook, IBM, Amazon, Yahoo! 等在内的世界著名互联网公司和部分政府部门已经广泛使用 Hadoop, 实现包括创建搜索引擎索引、超大规模图表分析等在内的多种功能.

Hadoop 的核心组件是 HDFS^[2] 和 Hadoop-MapReduce^[3], 分别可视为 GFS 和 MapReduce 的开源实现.

HDFS 是 GFS 的开源版本, 采用 Master-Slave 架构, 是一种高度容错性的分布式文件系统, 它以流式数据访问模式来存储超大文件 (通常达到数百 MB, GB 甚至 PB 级), 运行于廉价计算机集群上. HDFS 集群中的节点分为两类, 一个 namenode 作为管理者, 多个 datanode 作为工作者^[4-5].

Hadoop-MapReduce 是 Hadoop 的最核心组件, 是 MapReduce 模型的一种开源实现, 通过将问题转换为分布式集群上若干台设备的并行计算达到分布式处理的目的^[6].

随着大数据和大数据中心规模的扩大, Hadoop 的集群规模也亟需扩大, 由于 Hadoop 采取 1:n 的 Master-Slave 架构, Master 节点的内存空间、网络带

宽和处理性能已经成为 Hadoop 集群规模和计算能力的瓶颈,如果 Master 节点宕机,也将导致进行中的任务全部失效,为此,设计更好的平台架构,使 Hadoop 获得更好的可扩展性和容错性是本文的研究重点.

当前这一问题的思路主要有以下几点:

- (1) 使用大型机或超级计算机作为 Master;
- (2) 建立多个 NameNode;
- (3) 对 Hadoop 进行层次化管理,降低管理节点压力

由于 Hadoop 的 Master-Slave 架构是 1: n 关系,管理任务全部集中于 namenode 和 Jobtracker 上^[7],高度集中的管理机制简化了 Hadoop 的管理控制流和数据流,带来了更好的负载均衡,但也带来了性能进一步提高的瓶颈^[8]. 面向这一问题,本文提出了一种层次化 Hadoop 模型,通过引入层次化概念,在 datanode 中建立由 SeMNode (semi-management node) 组成的次级管理网络对元数据和细粒度负载均衡提供管理,使 Master 专注于更粗粒度的负载均衡和节点情况,从而降低 Master 负担,提高 Hadoop 的可扩展性.

2 多层次 Hadoop

2.1 相关定义

多层次 Hadoop 是指在 Hadoop 现有的 Master-Slave 架构中将部分 Slave 节点升级为 Semi-Master-Node 形成 Semi 层并替代 Master 实现对区域内 Slave 的直接管理,作为连接 Master 和 Slave 的枢纽,其体系结构如图 1 所示.

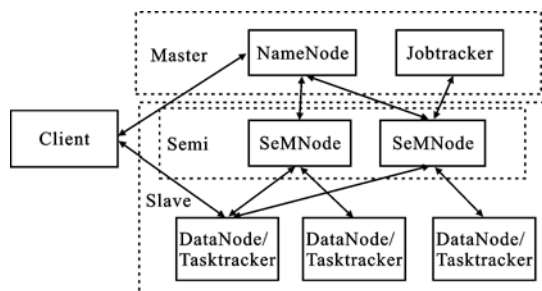


图 1 多层次 Hadoop 体系结构

2.1.1 Semi 层

在 Slave 层内,选举部分节点成为 SeMNode,形成 Semi 层. SeMNode 拥有传统 HDFS 中 NameNode 的部分功能. 但需要注意的是,传统 Hadoop 架构中 NameNode 节点将整个文件系统的元数据信息全部加载到内存中,而 SeMNode 则仅仅加载它所负责的

部分元数据信息. 从而将 Hadoop 上元数据的操作分散到若干个 SeMNode 节点处理.

在每个 SeMNode 上,设计了一个通信模块 C2M. C2M 负责向 NameNode 以心跳形式定期汇报本节点的状态信息,同时接收 Slave 节点的状态信息,若发现管辖区域内某节点失效则利用这些信息进行相应的任务转移等操作.

在数据发生变化时,SeMNode 负责决定数据添加或删除的节点位置,并修改对应索引文件,同时与 NameNode 通信,并将变化的数据元文件发送给 NameNode 用于修改全局索引.

2.1.2 Master 层

Master 层是 Hadoop 的管理层和控制中心,负责管理元数据和作业. 在多层次 Hadoop 架构中,NameNode 是一个全局服务器,负责管理 SeMNode 传来的元文件,构成全局命名空间及全局索引. 当需要检索某个文件时,NameNode 只需确定其位于某个 SeMNode 区域内,并将相应检索请求发送给 SeMNode 由它进行细粒度检索操作,进而完成检索操作.

Job 是 Hadoop 的任务单元,JobTracker 负责 job 的分发和负载均衡. 与传统 Hadoop 架构中的 JobTracker 将任务分配到 Slave 节点级不同,多层次 Hadoop 架构中,JobTracker 仅仅需要将 job 分配到 SeMNode 中,完成这一级别的负载均衡,再由 SeMNode 完成下一步任务分配工作.

2.1.3 Slave 层

Slave 层包括数据节点 DataNode 和作业处理节点 TaskTracker (事实上由于 Hadoop 遵循本地处理原则,TaskTracker 通常部署于相应的 DataNode 上),数据以块 (block) 的形式存储在 DataNode 上. 一个数据块在 DataNode 以文件存储在磁盘上,包括两个文件,一个是数据本身,一个是元数据,包括数据块的长度、块数据的校验和,以及时间戳.

DataNode 启动后向 SeMNode 注册,通过后,周期性 (1 小时) 地向 SeMNode 上报所有的块信息. 同时,DataNode 周期性地向 SeMNode 发送心跳包,心跳返回结果带有 SeMNode 给该 DataNode 的命令如复制块数据到另一台机器,或删除某个数据块. 如果超过 10 分钟没有收到某个 DataNode 的心跳,则认为该节点不可用. 集群运行中可以安全加入和退出一些机器.

2.1.4 Client

Client 即客户端,是用户与 Hadoop 通信的枢纽. 多层次 Hadoop 系统中的 Client 并未做出改动,

在此不做赘述。

2.2 选举机制

依据一定的规则, SeMNode 从 Slave 节点中产生, 作为“领导”与 Master 层通信, 并负责它所负责范围内 Slave 节点的负载均衡和数据管理。SeMNode 需要兼顾通讯效率和可用性容错性, 因此其选举机制非常重要。本文以文件为单位, 令文件集为 $F\{f_1, f_2, \dots, f_n\}$, Hadoop 集群 H , 由 x 个 DN (DataNode) 和 1 个 NN (NameNode), 1 个 JT (JobTracker) 组成, 则集群可以表示为 $H = \{NN, JT, DN_i | i \in (0, x)\}$ 。对于某个文件 f_k , 假设被分为 a 个块 (block), 则 f_k 可以表示为 $B = \{b_1, b_2, \dots, b_a\}$ 。算法的基本流程如图 2 所示。

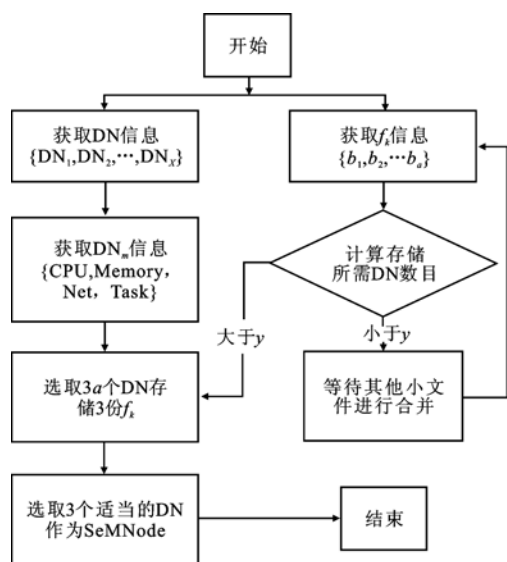


图2 SeMNode 选举算法基本流程图

选举算法基本步骤如下所示。

(1) NameNode 通过心跳包获取集群中全部 DataNode 信息, 并检视需要新存储的文件 f_k 结构, 计算其需要的分块规模信息。

(2) 通过计算出的文件分块规模信息确定需要选举为 SeMNode 的 DataNode 数量。

(3) 若所需 DataNode 数量大于预设的特定阈值 y , NameNode 与活动的 DataNode 进行通信, 获取包括其 CPU 使用、内存占用、网络使用和任务执行情况等信息, 并进行负载评估, 选出负载较小的 $3a$ 个 DataNode 并存储相应文件 f_k 的 3 份副本。

(4) 在这 $3a$ 个 DataNode 中进一步选取 3 个适当的节点 (实现时考虑性能和故障恢复情况, 实际选取 CPU 使用率最低, 网络带宽使用最小和任务执行量最小的三个节点) 成为 SeMNode, 由这三个节

点获取并维护该文件的分块和索引信息。

(5) 若 f_k 的规模未能达到预设的阈值 y , 即文件的规模较小时, NameNode 将该任务暂时挂起, 等待下一个任务, 并将二者合并, 重新进行其规模与阈值 y 的判定。

3 实验与性能分析

实验在本地机房中搭建的 Hadoop 平台下进行, 为保证实验结果尽量不受其他因素及偶然情况所影响, 在对 Hadoop 进行修改前后作为对比的实验均在相同机群上进行, 每组实验均进行 10 次, 最终结果在去除极值后取均值得到。

具体实验配置如下所示。

Master: 浪潮 NF5240M3 服务器 (CPU: Xeon E5-2407, 内存: 4 GB DDR3, 硬盘: 1 TB 7 200 转) 数量: 1 台。

Slave: Lenovo B4360 (CPU: 赛扬 G1620, 内存: 2 GB DDR3 1 333 MHz, 硬盘: 500 GB 7 200 转), 数量: 4/8 台。SeMNode 数量: 0/3 台。

OS: Ubuntu 12.04。

Hadoop: 2.2.0 (为在实验室中模拟大量数据集环境, 将 Hadoop 数据块大小由默认的 64 Mb 修改为 64 kb, 从而获得千倍于正常 Slave 节点容量的 block 数量)。

在实验时, 我们首先产生大量随机数作为文件存入 Hadoop 平台中, 当 block 数量达到 10 000, 20 000, 30 000, 40 000 后, 分别在 Hadoop 平台上运行 Sort 和 Grep 两个平台自带任务作为性能考量标准, 从而尽量避免由于算法等其他因素的干扰, 保证实验结果的有效性。

在 Master 节点数量为 1, Slave 节点数量为 8, Hadoop 多层次化改造前后 SeMNode 节点数量分别为 0/3 时, Hadoop 完成 Sort 和 Grep 所需时间如图 3 所示。

图 4 描述了 Hadoop 平台多层次化前后完成全部数据 Sort 和 Grep 任务所需时间。为确定系统瓶颈为 NameNode, 我们将 Master 节点内存容量数量增倍为 8 Gb, 并相应地将 block 数量峰值增倍到 8×10^4 , 其余参数不变, 进行了下一组实验, 其结果如图 4 所示。

对比两组实验可见, NameNode 的内存容量制约着传统 Hadoop 正常运行时所能容纳的数据规模, 在对 Hadoop 进行多层次改造后, NameNode 的任务负担得到减轻, Hadoop 获得了更大规模数据的存储

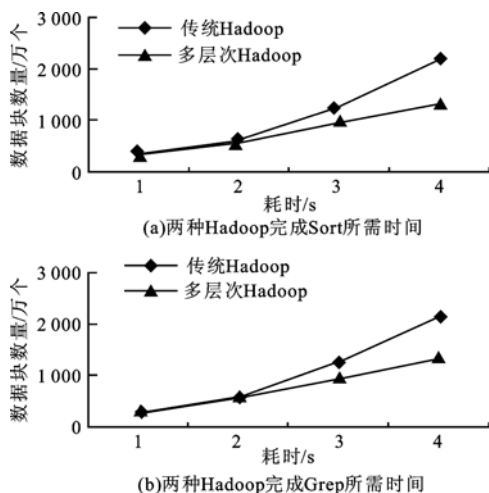


图3 两种 Hadoop 完成任务所需时间

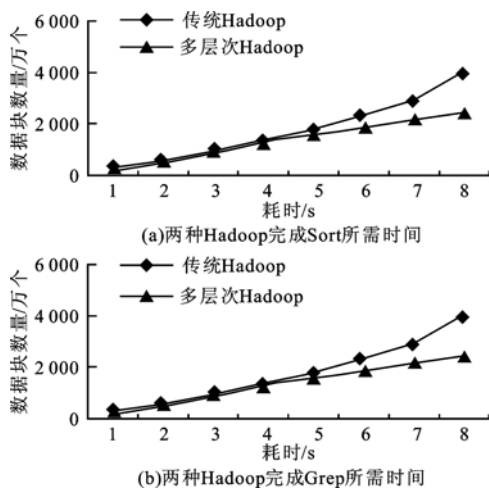


图4 两种 Hadoop 完成任务所需时间

和处理性能;当存储的数据规模较小时,多层次 Hadoop 的性能并未产生明显下降。

4 结束语

为提高 Hadoop 平台的可扩展性及 Master 节

点的故障恢复性能,对 Hadoop 平台进行多层次改造,同时分散了 Master 节点的任务负担,使其仅承担粗粒度的元数据存储和任务分发工作。由于多层次结构将带来集群网络上更多的通信任务和更复杂的负载均衡任务,通信数据的压缩及负载均衡是有待深入解决的问题。

参考文献:

- [1] Tom White T. Hadoop: the definitive guide[M]. O'Reilly: Yahoo Press, 2011:35-36.
- [2] Shvachko K, Kuang H, Radia S, et al. The hadoop distributed file system[C] // 2010. Mass Storage Systems and Technologies (MSST), 2010 IEEE 26th Symposium on. Nevada, Indine Village, IEEE, 2010:1-10.
- [3] 覃雄派. 大数据分析:RDBMS 与 MapReduce 的竞争与共生[J]. 软件学报, 2012,23(1):23.
- [4] 邓鹏,李枚毅,何诚. Namenode 单点故障解决方案研究[J]. 计算机工程, 2012,38(21):40-44.
- [5] 龙柏. 并行计算平台上的数据索引技术研究[D]. 合肥:中国科学技术大学, 2011:25-33.
- [6] Borthakur D, Sarma J. Apache hadoop goes realtime at facebook[C] // Proceedings of the 2011 ACM SIGMOD International Conference on Management of data. New York, ACM, 2011:1071.
- [7] Tom White. The small files problem[EB/OL]. (2009-02-02). [2015-04-04]. <http://www.cloudear.com/blog/2009/02/02/the-small-files-problem/>.
- [8] 王凯. MapReduce 集群多用户作业调度方法的研究与实现[D]. 长沙:国防科技大学, 2010:15-24.

作者简介:

李兆兴 男, (1990-), 博士研究生. 研究为大数据、数据检索、数据挖掘. E-mail: hoops@whu.edu.cn

马自堂 男, (1962-), 教授. 研究方向为大数据、云计算、信息.