

基于异构 Hadoop 集群的负载均衡策略研究

秦 军¹, 冯亮亮², 孙 蒙²

(1. 南京邮电大学 教育科学与技术学院, 江苏 南京 210003;
2. 南京邮电大学 计算机学院, 江苏 南京 210003)

摘 要:异构 Hadoop 环境中, 每个节点的处理能力各不相同, 且集群中的节点会不断增加和删除, 随着作业量的增大, 负载倾斜会越来越明显。显然, 负载均衡也成为影响 Hadoop 集群性能的重要因素之一。针对异构 Hadoop 环境中 MapReduce 任务调度, 提出了一种新的负载均衡算法。该算法充分利用节点性能和当前的计算资源, 根据集群负载平衡度量值进行任务分配, 将任务分配给适合的节点, 使集群负载逐渐趋于平衡, 以提高集群节点利用率。由于 Hadoop 集群中各节点通过网络连接, 以节省网络传输代价, 因此在负载均衡调度时, 根据数据分布特点, 优先考虑数据的本地性, 以缩短任务执行时间。仿真实验结果表明, 所提出的负载均衡算法能明显改善系统性能, 有效缩短 MapReduce 作业执行时间。

关键词:Hadoop 集群; MapReduce; 节点性能; 任务调度; 负载均衡

中图分类号: TP301.6

文献标识码: A

文章编号: 1673-629X(2017)06-0110-04

doi: 10.3969/j.issn.1673-629X.2017.06.023

Research on Load Balancing Strategy with Heterogeneous Hadoop Clustering

QIN Jun¹, FENG Liang-liang², SUN Meng²

(1. College of Education Science & Technology, Nanjing University of Posts and Telecommunications,
Nanjing 210003, China;
2. College of Computer, Nanjing University of Posts and Telecommunications, Nanjing 210003, China)

Abstract: In the heterogeneous Hadoop environment, processing capabilities of the nodes are diverse and various, among which each node may be continuously added or removed in the clustering and its load slope may increase obviously with tasks. Apparently, load balancing is one of the most important factors that affect the performance of Hadoop clustering. Thus a new load balancing algorithm has been proposed and employed in MapReduce task scheduling of the heterogeneous environment, which makes full use of the node performance and current computation resources according to the cluster load balancing measurement value to allocate task to suitable node for balancing the cluster load gradually and promoting coefficient of utilization of cluster nodes. Since the nodes in the Hadoop clustering are connected with network to save the costs of network transmission and the data locality should be considered with priority to decrease execution time for each task according to the characteristics of data distribution during load balancing scheduling. Simulation results show that the proposed load balancing algorithm has improved performances of whole system significantly and shorten the execution time of the MapReduce task.

Key words: Hadoop clustering; MapReduce; node performance; task scheduling; load balancing

0 引 言

Hadoop 是由 Apache 基金开发的分布式系统基础架构^[1]。该架构充分利用集群进行高速运算和存储。作为开源的云计算平台, 已有大量学者加入到 Hadoop 集群的研究中, 并且有许多著名的互联网公司, 例如

Facebook、Yahoo、百度、阿里巴巴等利用 Hadoop 集群进行海量数据的存储和处理^[2]。在大规模的并行计算中, 如何充分利用集群的计算资源, 提高系统性能, 成为负载均衡技术研究的重点^[3-4]。

作为 Hadoop 的核心框架, MapReduce 利用分而治

收稿日期: 2016-07-15

修回日期: 2016-10-20

网络出版时间: 2017-04-28

基金项目: 江苏省自然科学基金项目 (BK20130882)

作者简介: 秦 军 (1955-), 女, 教授, 研究方向为计算机网络技术、多媒体技术、数据库技术; 冯亮亮 (1992-), 女, 硕士研究生, 研究方向为分布式计算机技术与应用。

网络出版地址: <http://cnki.net/kcms/detail/61.1450.TP.20170428.1703.070.html>

之的思想将主节点 JobTracker 的作业执行分成很多个子任务 task^[5],然后交给集群中 TaskTracker 节点去执行。MapReduce 作业执行主要分为 map 任务和 reduce 任务。在 map 阶段,接收<key₁, value₁>的输入,产生<key₂, value₂>的中间结果,将所有键值对中 key 相同的 value 组成 value list 传递给 reduce,reduce 阶段接收<key₂, (value₂ list)>作为输入,对 value 集处理后得到<key₃, value₃>,并输出最终结果^[6]。整个作业的完成时间取决于 map 任务和 reduce 任务的完成时间之和。在集群节点很多且负载较大时,任务调度会起到关键的作用^[7]。同构环境下,集群节点之间差异小,任务调度在各节点之间的负载能够达到较好的平衡^[8]。但是实际的 Hadoop 集群是由大量普通廉价计算机组成^[9],节点处理能力各不相同,随着负载的增加,集群中负载不平衡会越来越严重,导致集群整体吞吐量不高、资源利用率低等问题。从而导致任务处理时间的增加,进而延长了整个作业的执行时间。

因此,为了保持异构 Hadoop 集群处于负载平衡状态,在对集群节点进行任务分配时,应该充分考虑节点之间的性能差异和集群当前的负载情况,将任务分配给与其计算能力相应的节点,从而不因局部节点负载过重导致集群负载失衡。仿真实验结果表明,所提出的负载均衡算法能明显改善系统性能,有效降低 MapReduce 的运行时间。

1 相关研究及改进模型

负载均衡问题一直是影响 Hadoop 集群性能^[10]的重要因素之一。通过负载均衡可以使得集群中的节点处于相对的相等忙碌状态。一个好的负载均衡算法能通过均衡任务来重新分配负载、优化系统的资源利用率和任务的响应时间。

文献[11]提出了 LBVP 算法:利用虚拟分区 Partition,将 map 阶段处理完成后的输出数据进行分区,保证各个 reduce 获取的数据量基本一致,从而达到负载均衡的目的。该算法对于同构环境下的集群负载均衡效果较好,但对于异构环境下的集群,每个节点处理能力各不相同,即使对于输入同样的数据量,其处理效果也会有较大差异。文献[12]提出了基于节点性能的 LBNP 负载均衡算法:异构环境下,根据节点性能,解决 map 阶段执行完后的数据预分配问题,从而均衡 reduce 阶段任务的执行时间。该算法考虑了异构环境下节点的性能差异,从局部 reduce 阶段任务调度进行均衡,不能解决在全局 MapReduce 执行过程中的负载均衡问题。同样地,文献[13]结合节点计算能力和数据本地性感知的 MapReduce 负载均衡也只是针对 reduce 阶段任务调度提出的。

针对上述集群负载均衡中存在的问题,提出了一种改进的任务调度负载均衡模型,如图 1 所示。

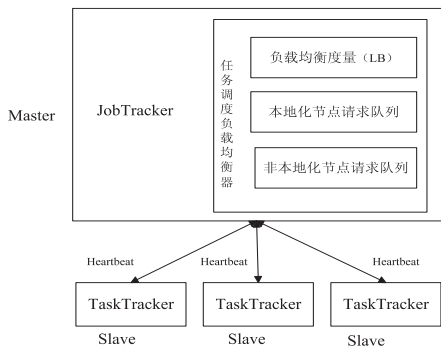


图 1 任务调度负载均衡模型

图 1 中的负载均衡器包括三个模块:负载均衡度量(Load Balancing, LB)、本地化节点请求队列和非本地化节点请求队列。通过对异构环境中 TaskTracker 节点进行合理的任务调度,保持集群中节点间的负载平衡。其中 LB 是通过周期地收集 TaskTracker 节点信息得到的集群负载均衡估量值。按照数据本地性要求,将请求 map 或 reduce 任务的 TaskTracker 节点分为本地化和非本地化请求队列,并根据当前集群负载均衡度量值 LB 优先对本地化节点请求队列中的节点分配任务。由于在每一个心跳周期内,集群性能和请求分配任务的节点是动态变化的,所以负载均衡器所维护的两个节点请求队列是基于抢占式优先的。这样负载均衡器在进行任务调度时是基于当前集群的实时信息进行的,保证了调度的可靠性。

2 算法实现过程

图 1 的任务调度负载均衡模型是为了在进行任务调度时,根据当前节点性能和负载均衡度量值将任务分配给与其能力相匹配的节点上,避免超载或饥饿现象,从而使集群的资源得到充分利用。该算法所需要的计算资源和实现步骤如下所述。

2.1 节点性能计算

为了充分反映节点当前性能,应该从多个变量因素进行分析,如 CPU 利用率、内存利用率、磁盘 I/O 占用率、网络带宽占用率以及执行 task 任务的响应等。综合这些因素来反映节点性能。对于节点 i ($i = 1, 2, \dots, n$),其性能计算见式(1):

$$p_i = [r_1 \quad r_2 \quad r_3 \quad r_4 \quad r_5] \begin{bmatrix} p_{cpu} \\ p_{io} \\ p_{memory} \\ p_{bandwidth} \\ p_{response} \end{bmatrix} \quad (1)$$

其中, p_i 表示节点 i 的性能; p_{cpu} 、 p_{io} 、 p_{memory} 、 $p_{bandwidth}$ 和 $p_{response}$ 分别表示 CPU 利用率、磁盘 I/O 占用率、内存

利用率、网络带宽占用率和单位时间内对 task 任务的响应速率; r_1 、 r_2 、 r_3 、 r_4 和 r_5 分别表示各变量在影响节点性能方面所占的比重,根据实际环境来确定,并且 $r_1 + r_2 + r_3 + r_4 + r_5 = 1$ 。

2.2 集群整体性能估算

JobTracker 周期性地收集 TaskTracker 通过式(1)计算得到的节点性能,从而得到集群整体的负载状况。假设集群中含有 n 个计算节点,每个节点性能为 p_i ($i = 1, 2, \dots, n$),计算集群整体平均性能为:

$$p_{\text{average}} = \sum_{i=1}^n p_i \quad (2)$$

2.3 节点请求队列的维护

TaskTracker 节点向 JobTracker 请求分配新的任务时,不仅要保证当前节点运行良好,具有较高的性能,还要保证有足够的计算资源用于执行 task 任务。节点 i ($i = 1, 2, \dots, n$) 当前可以用的计算资源式表示为:

$$q_i = [k_1 \quad k_2 \quad k_3 \quad k_4] \begin{bmatrix} q_{\text{cpu}} \\ q_{\text{io}} \\ q_{\text{memory}} \\ q_{\text{bandwidth}} \end{bmatrix} \quad (3)$$

其中, q_i 表示当前节点剩余计算资源; q_{cpu} 、 q_{io} 、 q_{memory} 、 $q_{\text{bandwidth}}$ 分别表示 CPU、磁盘 I/O、内存和网络带宽的剩余量; k_1 、 k_2 、 k_3 和 k_4 分别表示各变量所占的比重,并且 $k_1 + k_2 + k_3 + k_4 = 1$ 。

当有多个节点请求分配任务时,考虑到异构环境中的节点差异,为同样性能表现的节点,分配相等任务量时,执行任务时间会受到剩余内存和 CPU 等可用计算资源的限制。因此任务调度器需要综合当前节点性能和剩余可用计算资源。可用两者的比值来表示:

$$\text{Node}_j = \frac{q_j}{p_j} \quad (4)$$

其中, j 表示请求分配任务的节点; Node_j 表示作为任务调度器中节点请求队列排序的标准; p_j 和 q_j 可分别由式(1)、式(3)计算得到。

考虑到数据本地性,任务调度器会维护两个节点请求队列:本地化节点请求队列和非本地化节点请求队列,分别用 LocalityQueue 和 nonLocalityQueue 来表示。当有多个节点同时请求分配任务时,通过判断该节点是否含有本地化数据,将其加入到相应的队列。加入到队列的节点按照式(4)从大到小进行排序。而且队列 LocalityQueue 具有比队列 nonLocalityQueue 更高的优先级,即在进行任务调度时,会优先从 LocalityQueue 队列中选择节点来分配任务。

2.4 负载均衡度量

在 MapReduce 集群大规模的数据计算时,集群负载应该是逐渐趋于平衡的,即大部分节点的计算资源

能得到充分利用,节点性能均表现良好,出现节点负载超载或者长期空闲的概率很低。根据概率论中的大数定律和中心极限定理可知,节点性能表现值分布应该近似数学中的正态分布。用 X 表示集群节点性能值,则 $X \sim N(\mu, \sigma^2)$ 。其中 X 即为式(1)中的 p_i ,期望 μ 为式(2)计算得到的均值, σ^2 为方差,见式(5):

$$\sigma^2 = \frac{\sum_{i=1}^n (X_i - \mu)^2}{n} \quad (5)$$

其中, X_i 表示节点 i ($i = 1, 2, \dots, n$) 的性能。

由正态分布 3σ 准则可知,数据的取值几乎全部集中在 $(\mu - 3\sigma, \mu + 3\sigma)$ 范围内。对于负载均衡的集群来说,其节点性能值的分布也应遵守 3σ 准则。这里设定集群负载均衡的范围为 $\mu - 2\sigma < X_i < \mu + 2\sigma$,因为数值落在 $(\mu - 2\sigma, \mu + 2\sigma)$ 的概率为 95%。对于集群中负载均衡的节点来说,其节点性能值应该分布在该范围内,分布在该范围外的节点是极少数的。性能值 $X_i < \mu - 2\sigma$ 的节点,表示节点当前性能较差,负载已超载,应该避免分配新的任务。性能值 $X_i > \mu + 2\sigma$ 的节点,表示节点当前性能很好,处于空闲状态,可以分配新的任务。假设负载均衡度量为 $\text{LB} = \mu - 2\sigma$,当节点性能值 $X_i > \text{LB}$ 时均可以分配新的任务。

2.5 任务调度负载均衡算法的实现步骤

在异构环境下,基于集群节点性能和数据本地性的任务调度负载均衡算法的实现步骤如下:

(1) 每个心跳周期内,JobTracker 收集 TaskTracker 节点的信息。节点的性能和可用资源的计算根据式(1)、式(3)完成。

(2) 当有多个 TaskTracker 节点发送请求分配新任务的请求时,任务调度器会将含有本地化数据的节点放入到 LocalityQueue 队列中,而不含本地化数据的节点放到 nonLocalityQueue 队列。并按照式(4)将队列中的节点从大到小进行排序。

(3) 当 LocalityQueue 队列非空时,依次取队列中节点,当节点性能值大于 LB 时,可以为该节点分配新任务,否则判断队列中的下一个节点。

(4) 当 LocalityQueue 为空或者已经遍历结束时,依次取 nonLocalityQueue 队列中的节点,同样当节点性能值大于 LB 时,可以为该节点分配新任务,否则判断队列中的下一个节点。

(5) 重复步骤(1)~(4)。

需要说明的是,在心跳周期内如果有新的节点请求分配任务,则将其按照步骤(2)中的规则加入到相应队列,可见负载均衡器所维护的节点请求队列是实时更新并且是基于抢占式优先的。这样保证了任务分配时,选择性能和计算资源最好的节点,缩短任务执行

时间,并且满足负载均衡的要求。

3 仿真实验及结果分析

通过仿真实验评估改进后的任务调度算法和 Hadoop 默认的 FIFO 调度算法^[14]的性能表现。实验中选取 10 台在 CPU 频率、内存大小等各不相同的物理机器用来模拟异构的集群环境。MapReduce 应用作业选择 sort,即对输入字典中的数据进行排序。作业的大小依次选择 2 G、4 G、6 G、8 G 和 10 G。在相同输入数据规模的情况下,分析原有 Hadoop 框架中 FIFO 任务调度算法和改进后的任务调度算法在作业完成时间、集群性能和系统吞吐量方面的对比,结果见图 2~4。

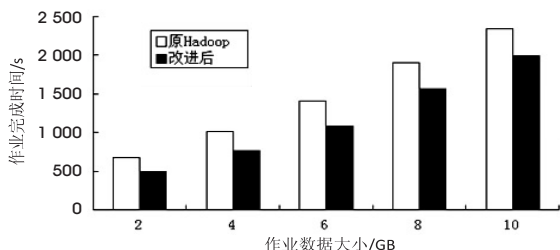


图2 基于作业完成时间的比较

从图2可以看到,改进后的任务调度执行时间通过更多的执行本地化数据,缩短了任务执行时间,从而加快了整个作业完成时间。

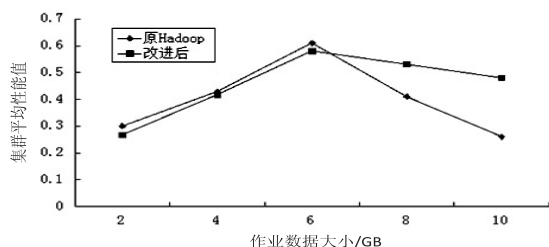


图3 基于集群性能表现的比较

从图3可以看到,作业在 6 GB 左右时,集群平均性能达到了最大,随着作业规模的增加,负载加大,改进后的集群性能相对于原 Hadoop 仍然能保持在一个较高水平,可见负载均衡效果得到了明显的改善。

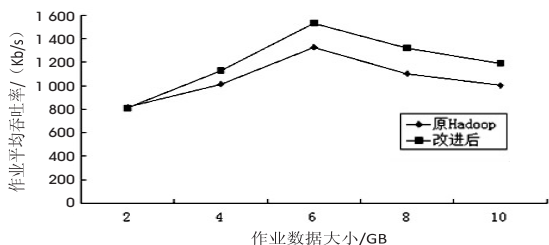


图4 基于集群系统吞吐量的对比

从图4中可以看到,系统的吞吐量增大,这是由于改进后的任务调度算法充分考虑异构环境下的节点差异,通过利用集群中节点计算资源增大了系统吞吐量。

万方数据

4 结束语

MapReduce 中的调度相关问题是影响其执行性能的重要因素之一,负载均衡则是其在调度中需要考虑的因素。在分析节点计算性能的基础上,提出了一种负载均衡度量新的计算方式,通过与 LB 的比较,将任务分配给与节点计算能力最匹配的节点,使计算资源得以充分利用,使得集群负载逐渐趋于均衡,增加了系统吞吐量。且在进行负载均衡调度的同时,通过优先调度本地化节点,以减少数据迁移代价和网络带宽,缩短了任务的执行时间。仿真实验结果表明,提出算法显著提高了系统的综合性能。

参考文献:

- [1] Apache Hadoop[EB/OL]. 2016-02-13. <http://hadoop.apache.org>.
- [2] 应毅,刘亚军. MapReduce 并行计算技术发展综述[J]. 计算机系统应用,2014,23(4):1-6.
- [3] 柳香,李瑞台,李俊红,等. Hadoop 性能优化研究[J]. 河北师范大学学报:自然科学版,2011,35(6):567-570.
- [4] 周一可. 云计算下 MapReduce 编程模型可用性的研究与优化[D]. 上海:上海交通大学,2011.
- [5] 孙彦超,王兴芬. 基于 Hadoop 框架的 MapReduce 计算模式的优化设计[J]. 计算机科学,2014,41(11A):333-336.
- [6] Dean J, Ghemawat S. MapReduce: simplified data processing on large clusters[J]. Communications of the ACM, 2008, 51(1):107-113.
- [7] 许丞,刘洪,谭良. Hadoop 云平台的一种新的任务调度和监控机制[J]. 计算机科学,2013,40(1):112-117.
- [8] Fan Y, Wu W, Cao H, et al. A heterogeneity-aware data distribution and rebalance method in Hadoop cluster[C]//ChinaGrid annual conference. [s. l.]: IEEE, 2012:176-181.
- [9] 顾涛. 集群 MapReduce 环境中任务和作业调度若干关键问题的研究[D]. 天津:南开大学,2014.
- [10] 荀亚玲,张继福,秦啸. MapReduce 集群环境下的数据放置策略[J]. 软件学报,2015,26(8):2056-2073.
- [11] Fan Y, Wu W, Cao H, et al. LBVP: a load balance algorithm based on virtual partition in Hadoop cluster[C]//Asia Pacific cloud computing congress. [s. l.]: IEEE, 2012:37-41.
- [12] Gao Z, Liu D, Yang Y, et al. A load balance algorithm based on nodes performance in Hadoop cluster[C]//16th Asia-Pacific network operations and management symposium. [s. l.]: IEEE, 2014:1-4.
- [13] 李航晨,秦小麟,沈尧. 数据本地性感知的 MapReduce 负载均衡策略[J]. 计算机科学,2015,42(10):50-56.
- [14] Tao Y, Zhang Q, Shi L, et al. Job scheduling optimization for multi-user MapReduce clusters[C]//2011 fourth international symposium on parallel architectures, algorithms and programming. [s. l.]: IEEE, 2011:213-217.