基于性能测量对设计一种横向，纵向扩展混合的 Hadoop 架构的探究

论文发表在 IEEE Transactions on Parallel and Distributed Systems 分布并行系统期刊上。作者：Zhuozhao Li, Haiying Shen，Walter Ligon, and Jeffrey Denton。发表年份：No. 02 - Feb. (2017 vol. 28)。团队介绍：第一作者 Zhuozhao Li 现在 Clemson University 攻读博士学位，研究兴趣包括分布式计算机系统，云计算，云网络上的资源管理，和大数据处理。

主要问题：

纵向扩展的系统在小型或中型（有 KB,MB 大小的数据量）作业上表现更好，而横向扩展的系统在大型（有 GB、TB 大小的数据量）作业上表现更出色。而工作中通常涉及到不同数据大小级别的业务场景，导致传统的 Hadoop 在混合任务场景中发挥不出全部资源优势。研究团队设法构建一种同时包含纵向集群和横向集群的集群架构。纵向扩展为 scale-up 类型，指通过给单机配置更强大的 CPU 和更大的内存，但单机数量相对较少。横向扩展为 scale-out 类型，指通过给单机配置较弱的 CPU 和较小的内存，但单机数量相对更多。

研究背景及相关工作：

MapReduce 是一个用于在集群的计算节点上用并行方式处理大量数据的框架。Hadoop 是一个受欢迎的 MapReduce 的实现项目，已经在许多公司比如 Facebook，Google 和雅虎被采用。在过去几十年间，数据的数量和计算量以指数级别增长[9]，这个趋势带来了一个对 MapReduce 高性能要求的难题并且激励许多研究者从各方面探寻改进其效率（包括任务调度，小任务表现的最优化以及中间数据的重组）。有研究表明生产工作中的最多的任务无论是输入、中间数据还是输出通常都在 MB 到 GB 的范围。现在的 Hadoop 集群一开始并不是为了小数据量，延迟敏感型的工作而设计的[19]，故而生产集群都受到许多小型任务的性能低下的影响。研究团队计算了 Facebook 综合任务中超过 6000 个任务的累计分布函数，40%的任务处理不到 1MB 的小数据集，49%的任务处理 1MB 到 30GB 的中等数据集，剩下 11%的任务处理超过 30GB 的大数据集。这样的实际工作场景同时需要纵向扩展和横向扩展的集群来处理不同大小级别的数据。

另一方面，为了更好地了解实际 MapReduce 作业的特征，并针对特征来对 MapReduce 提出改进方案。已经出现了许多分析 MapReduce 工作任务特征的研究文章，这些文章致力于将 MapReduce 工作任务类型化以使得它们能更好地被分配。Chen 的团队分析了雅虎和 Facebook 的生产工作来建立描述 Mapreduce 工作的特征。他们的另一个工作尝试预测新的 MapReduce（非重复执行的 MapReduce）的工作的特征：通过交互式分析以及查询语言（比如构建在 MapReduce 上的 Hive）的使用来预测特征。Ren 等人尝试描述 Taobao Hadoop 集群的生产任务来增进对它们效率情况的理解以及 Taobao Hadoop 中任务在生产环境中的特征。Kavulya 等人分析了 M45 超算集群中的 MapReduce 日志。Appuswamy 等人进行了代表性的 Hadoop 任务在横向扩展和纵向扩展集群上的性能测算。团队发现纵向高配机器处理 MB 到 GB 大小的数据时能够有更好的表现。Camdoop 在数据转发期间对 shuffle 数据执行了网络内的聚合，以降低网络流量。Wang 等人提出了对于 Hadoop 绕过 JVM 执行 shuffle 的想法来避免 JVM 带来的瓶颈和限制。

研究动机、难点、主要解决方法：

研究动机：

真实生产中通常有许多不同类型的任务和计算，仅仅使用横向或者纵向集群来驾驭整个生产显然不能达到理想的表现，使得研究团队希望建立混合的 Hadoop 架构，并配备一个任务调度器，将任务分配给适合执行该任务的集群。致力于改进集群性能而不花费更多的金额。在设想的混合集群中，利用横向扩展和纵向扩展两种方式的特性。纵向扩展有更强大的 CPU 和内存都更少的 CPU 核心数，因此更擅长小任务。另一方面，纵向有更多的 CPU 核心数，因此它更适合处理大数据量的数据。

难点主要来自两方面：

1.需要恰当的数据存储方式来使得纵向机器和横向机器都能够有效地得到需要的数据。数以千计的小数据量任务可能会使得有限的纵向扩展机器的本地磁盘超出负荷，故而不能将任务所需数据完全放在负责处理它的节点上。无论是纵向和横向机器中的任务都需要相同大小的数据集，这会导致机器间的数据传输。Facebook 档案显示数千计的工作（85%）处理小型或中型任务。工作的大部分能更高效地运行在纵向扩展的机器上，但从横向集群到纵向集群网络间的通信可能会降低应用的性能。

2.需要通过比较哪种集群更适合任务，自适应地将任务调度给纵向或横向集群。简单地通过输入数据集的大小来指派并不足够有效，因为还存在其他的决定任务在纵向、横向不同集群间表现差异的影响因素。在这篇论文中，研究团队致力于调查设计这样一种混合结构的可行性并开始一些初步探索。

主要解决方法：

HDFS 是设计用来密切与 Hadoop MapReduce 计算框架结合的文件系统[2]。为了应对第一个挑战，研究团队一开始将纵向和横向节点的数量设为一样，但是这种方式使得纵向节点需要时常在横向机器中获取数据，不经降低了工作效率同时也消耗了带宽。真正的解决方案是通过使用专用的远程存储系统（OrangeFS[6]）。专用存储卸下了计算节点的 I/O 负担并且使得纵向和横向节点的数据共享更加方便。

为应对第二个挑战，混合架构需要有能够决定不同作业的交叉点的模型。交叉点是指当纵向和横向集群表现出近似的效率时，输入数据的大小。当真实数据大小高于或低于交叉点时，其中一个集群可以提供更好的服务。团队发现一个任务的交叉点取决于多种因素。具体会在研究内容中详细说明，大体来说当输入数据量大于临界点时，任务在横向集群上效率更高，相反在纵向集群上表现更优。

研究内容：

文章先介绍了基于 HPC Hadoop 的横向和纵向扩展机器的基本配置。之后在**第三部分**展示了对于四种不同架构在不同类型的应用上所做的性能分析。第四部分在之前的性能分析后，发现由于不同类型的任务需要在不同的集群类型上才能取得最佳表现，从而提出了设想的纵向、横向混合的 Hadoop 架构。**第五部分**展现了构建的新架构与传统的 Hadoop 架构在试验性任务上的结果比较。第六部分介绍了其他研究人员做的相关工作。第七章总结了全文，并对未来的改进工作提出了一些方向。

这里重点介绍文章的第三部分和第五部分，即不同类型任务的性能分析和新架构与传统 Hadoop 架构的比较。第三部分中比较了四种 Hadoop 架构运行典型任务时的性能，分别是 up-OFS, up-HDFS, out-OFS, and out-HDFS 四种架构，up 和 out 分别表示纵向，横向；HDFS 和 OFS 表示不配备和配备 OFS。典型任务分别有 Wordcount、Grep、Terasort 以及 TestDFSIO（大数据量文件的读写）。前三项任务可归类为 Shuffle-intensive 任务，大数据文件读写可归类为 Map-intensive，区别在于前三项任务的中间数据规模较大，故而 shuffle 阶段较为耗时。第三部分的比较分析主要是为更精确地确定临界点服务，找出影响临界点的相关因素。

而第五部分的主要工作是在确定好临界点以及分配工作的模型后，利用建立的混合 Hadoop 架构与传统 Hadoop 架构运行实际生产任务（Facebook 提供），比较两者在执行时间，吞吐率，耗电量等指标上的差异。

实验方法及实验结果：

第三部分

实验结果中将任务执行时间，Map 阶段耗时，Shuffle 阶段耗时，Reduce 阶段耗时这些数据根据 up-OFS 组的对应值进行标准化。由于实验仅关注四组架构的性能比较，故而具体的执行时间的值是可以去除的。

当输入数据很大（>16GB）时，Wordcount 和 Grep 两个任务的性能高低可表示为：out-OFS>out-HDFS>up-OFS>up-HDFS，而 Terasort 的可表示为：out-OFS>up-OFS>out-HDFS>upHDFS。造成大数据量时 Terasort 性能测量结果与另外两种任务稍有差异的原因是排序程序不仅有相对大规模的中间数据，也有大数据量的输出，而 Wordcount 和 Grep 的输出数据量几乎可忽略。

当输入数据小的时候（0.5-8GB）三种任务的性能高低均为：up-HDFS>up-OFS>outHDFS>out-OFS。

不同作业临界点的差异来自于中间数据与输入数据的比率的不同（shuffle/input ratio），给定相同的输入数据大小，如果一个任务的中间数据更大，那么它更能够在 Shuffle 阶段获益于纵向机器的大内存和快速的虚拟内存，因此也会减少 Shuffle 阶段的执行时间。综上推

导可知，shuffle/input ratio 更大，临界点也会更大。但这并不意味着 shuffle/input radio 是唯一决定临界值的因素，实验当中的 Terasort 和 Grep 的 shuffle/input radio 均为 0.4，但 Terasort 的临界点却大于 Grep 的，输出数据大小同样也会影响到临界点。

根据四组架构的比较分析可知，输入数据大小，中间数据与输入数据比率，以及输出数据大小都会影响到在纵向和横向两种集群中的具体表现。论文也提出了针对论文中集群利用数据大小，中间数据与输入数据比率，输出数据大小等信息计算临界值的线性模型。

第五部分

实验结果显示，无论相较于传统的 HDFS，还是配备有 OFS 的 Hadoop 纵向集群，混合架构 Hybrid 的执行时间分布都增长更快，这表明混合架构在改善小任务执行效率上的有效性。混合架构不仅在纵向任务（小任务）上表现更突出，在横向任务上也如此。造成这一现象的原因有两个。一、混合架构集成了远程文件系统，这给大作业提供了更好的 I/O 效率。二、虽然传统的 Hadoop 架构内有更多的 Map 和 Reduce 的插槽，但大量的纵向任务占据了许多插槽，使得小任务有较差的表现。另一方面，实验观察到混合架构不仅改善了任务的执行效率，而且也会有更低地任务失败率。具体来讲，混合架构上有 98%的任务成功率，而 THadoop 和 RHadoop 分别是 82.3%，89.2%。混合架构集群在耗电量上也比传统的 Hadoop 架构更低。

在第五部分的最后，文章还对决定临界点数值的模型进行了敏感性分析，因为混合架构的优越性取决于两点

一、任务特性（比如输入，中间，输出数据的大小）可以被准确地预测。

二、决定临界点模型的可靠性。

敏感性分析也发现了临界点数值准确率对集群吞吐率具有重大影响，当使任务的 shuffle/input ratio 上升时，混合架构的吞吐量会急剧降低。因为当它上升时，临界点也会随之上升，这会使得许多横向任务跑在纵向高配的机器上。

可能存在的问题、局限性、可改进的思路和方法：

分配调度模型不够全面，没有考虑到纵向集群、横向集群之间的负载均衡，HDFS 的数据块的大小设置也存在随意性，未考虑 Map、Reduce 任务的数量。比如，如果许多小任务几乎同时发起资源请求，所有的任务都将会被分配给纵向集群，这会导致在纵向集群，横向集群之间的资源分配的不平衡。

另一个改进方向是：混合集群的算法中可以融入耗电量模型来指导调度程序来进一步地减少耗电量。项目 GreenMR[28]的调度算法就是根据期望耗电量最小来运行调度程序。

参考文献：

[2] Apache Hadoop. http://hadoop.apache.org/. [Accessed in Oct. 2015].

[6] OrangeFS. http://www.orangefs.org. [Accessed in Oct. 2015].

[9] S. Agrawal. The Next Generation of Apache Hadoop MapReduce. Apache Hadoop Summit India, 2011.

[19] K. Elmeleegy. Piranha: Optimizing Short Jobs in Hadoop. In Proc. of VLDB Endow, 2013.

[28] Z. Niu, B. He, and F. Liu. Not all joules are equal: Towards energyefficient and green-aware data processing frameworks. In Proc. of IC2E, 2016.