大数据负载的体系结构特征分析

罗建平[1,2] 谢梦瑶[1,3] 王华锋[2]
(中国科学院计算技术研究所先进计算机系统研究中心 北京 100190)[1]
(北京航空航天大学软件学院 北京 10091)[2] (郑州大学信息工程学院 郑州 450001)[3]

摘 要 针对大数据离线分析类和交互式查询类负载,首先对这些负载的一些共性进行分析,提取出公共操作集,并对它们进行分组整理;然后在大数据平台上测试这些负载运行过程中的微体系结构特征,采用PCA和SimpleKMeans算法对这些体系结构特征参数进行降维和聚类处理。实验分析结果表明负载之间有公共的操作集,如Join和Cross Production;有些负载有相似的属性,如Difference和Projection共享相同的微体系结构特征。实验结果对于处理器等硬件平台的设计以及应用程序的优化具有指导性的意义,并且为大数据基准测试平台的设计提供了参考。

关键词 大数据,大数据负载,体系结构特征
中图法分类号 TP391 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2015.11.008

Analysis of Architecture Characteristics of Big Data Workloads

LUO Jian-ping[1,2] XIE Meng-yao[1,3] WANG Hua-feng[2]
(Advanced Computer Systems Laboratory, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China)[1]
(School of Software, Beihang University, Beijing 100191, China)[2]
(School of Information Engineering, Zhengzhou University, Zhengzhou 450001, China)[3]

Abstract Aiming at the big data workloads with off-line analysis and interactive queries, first we analyzed some common features of these workloads, extracted the common set of operations and arranged the workloads in groups. Then, we tested the big data workloads on the BigDataBench platform and got the micro-architecture characteristics using PCA and SimpleKMeans algorithm for dimensionality reduction and clustering analysis. Our study revealed that big data workloads share a common set of operations such as the Join and Cross Production. We also observed that some of the big data workloads have many similar features. For example, the Difference and Projection operations share micro-architectural characteristics. The result of our experiment has a guiding significance for the design of hardware platforms like processors and the optimization of applications. Meanwhile, it also provides valuable insights into the implementation of the big data benchmark platform.

Keywords Big data, Big data workloads, Architecture characteristic

1 引言

人们发现在大数据基准测试平台ICTBench[7]、HiBench以及真实世界典型应用(深圳交通运输时空数据分析系统)中[1]负载之间存在冗余。作为一个大数据Benchmark,为使得人工合成的数据保留真实数据的关键特征,有两个关键的特性需要考虑:语义性Semantic和本地性[2]。语义性的特征反应了真实生活中深刻见解且具生活意义的数据。本地性反应了数据接近的模型。Benchmark的设计需要考虑以下几点:大数据负载应该代表大范围的应用领域;Benchmark中的负载应该具有数据多样性的特征;Benchmark中的负载本身不应该有冗余;在大规模分布式集群中,只有真实世界的数据才能反映实际系统的行为和负载的特征,因此真实世界的数据负载在大数据Benchmarks中被提出。然而获得真实世界的数据面临巨大的挑战:由于真实世界的数据涉及到商业秘密和用户的隐私,数据拥有者不愿意分享出他们的大数据。即使真实世界互联网上的数据是可利用的,在当前的互联网网速流量的情况下,下载兆兆字节的数据对于研究者是不可接受的。因此,发现大数据负载中的公共操作集,提取出负载之间公共的体系结构特征,对大数据基准测试平台的设计具有现实的指导意义。

总体来说,本文做了如下工作:对一些离线分析和交互式查询负载进行对比分析和描述,分析负载之间的一些共性,提取出共同的数据操作集,并将它们进行了分组整理;同时,对计算机体系结构的特征参数进行了详细的描述和分类。在BigDataBench[3]上测试了这些负载运行时的体系结构特征,

到稿日期:2014-10-17 返修日期:2014-12-01 本文受国家重点基础研究发展规划项目(2014CB340402),国家自然科学基金(61303054)资助。
罗建平(1989-),男,硕士生,CCF学会生员,主要研究方向为分布式系统、机器学习、大数据处理,E-mail:luojianping@ict.ac.cn;**谢梦瑶**(1992-),女,主要研究方向为数据挖掘、内存管理、异构存储,E-mail:xiemy1992@163.com;**王华锋** 男,博士后,副教授,主要研究方向为图像识别及基于图像的测量,E-mail:wanghufengbuaa@gmail.com。

包括指令级缓存相关的、数据级缓存相关的、旁路转换缓冲器相关的和内存访问相关的特征等总共 45 个特征参数。采用 PCA 和 SimpleKMeans 算法对负载的体系结构特征参数实验数据进行降维和聚类分析，并就一些重要的体系结构特征参数对各负载的影响进行了分析和描述。

本文组织结构如下：

1. 大数据负载的实验配置环境，包括硬件环境、软件环境配置、大数据负载类型(第 2 节)。

2. 微体系结构特征参数度量，对一些重要特征参数对各负载的影响进行了分析和描述(第 3 节)。

3. 对大数据负载运行过程中的体系结构特征参数进行降维和聚类分析(第 4 章)。

4. 实验结果分析和未来工作展望(第 5 节)。

2 实验配置

2.1 实验环境

实验环境的配置采用 Intel(R) Xeon E5645 处理器；核心为 6 核心 6 线程；CPU 主频为 2.4GHz；内存 32GB，DDR3；Linux 采用 CentOS 6.4-64 位，Linux 内核版本为 3.11.10。配置细节如表 1 所列。

表 1 硬件环境配置细节

处理器类型	Intel (R)Xeon E5645
内存	32GB,DDR3
Sockets	2
核数	6 核 2.4GHz
每个核的线程数	6
ITLB	4 路组相连，64 entries
DTLB	4 路组相连，64 entries
二级缓存共享 TLB	4 路组相连，512 entries
一级数据缓存	32kB,8 路组相连，64byte/line
一级指令缓存	32kB,4 路组相连，64byte/line
二级缓存	256kB,8 路组相连，64byte/line
三级缓存	12MB,16 路组相连，64byte/line

软件环境：JDK 版本为 1.7.0；Hadoop 版本为 1.0.2；Shark[8] 版本为 0.8.0。负载测试的软件平台为 Spark-0.8.1[9] 和 Hive-0.9.0[10]。首先，Spark 是为集群计算中的特定类型的工作负载而设计，即那些在并行操作之间重用工作数据集(比如机器学习算法)的工作负载。为了优化这些类型的工作负载，Spark 引进了内存集群计算的概念，可在内存集群计算中将数据集缓存在内存中，以缩短访问延迟。Spark 启用了内存分布数据集，使得 Spark 在某些工作负载方面表现得更加优越。除了能够提供交互式查询外，它还可以优化迭代工作负载。本实验选取了交互式查询相关的负载，故采用了 Spark 平台。

Hive 是基于 Hadoop 的一个数据仓库，与传统数据库相比，它可以将结构化的数据文件映射转换为一张数据库表。Hive 中定义了类 SQL 查询功能，包括对存储在 Hadoop 中的大量数据进行存储、查询和分析。这使得能在 MapReduce 中自定义 mapper 和 reducer 来处理原来 Hadoop 无法完成的复杂分析工作，不必开发专门的 MapReduce。本实验选取了离线分析相关的负载，故采用 Hive 平台[10]。

2.2 负载对比分析

本文选取离线分析和交互查询两大类负载进行对比，分析各个负载运用的算法或查询操作，根据公共操作集进行分类。基于排序操作的 Sort 和 Order by 归为一组(如表 2 a 栏所列)，它们之间的共性是将文件的每一行或者表格的某些字段作为单位进行比较。在 Hadoop 中，Sort 算法加载数据后首先进行抽样并产生标尺，然后 mapper 对输入的每条数据计算其处于哪两个标尺之间，将数据发给对应区间的 reducer，最后 reducer 分别对数据进行排序进而输出。基于 Spark 平台实现 Sort 算法的基本思想与 Hadoop 一样，但由于 Spark 支持数据内存驻留，操作过程中缓存 RDD links[5]，因此运行效率有所提高。在 Hadoop 平台实现 Order by[11] 操作，加载所有数据到一台服务器，然后执行 reduce 操作来对所有数据进行排序。Order by 只用一个 reducer 产生完全排序的结果，因此对于大规模数据集，它的效率十分低下[5]。

基于正则匹配或过滤的 Grep、Select 和 Filter 归为一组(如表 2 b 栏所列)。Grep 指令用于查找内容中包含指定范本样式的文件，该指令对载入的数据进行遍历，搜索到匹配项后将数据输出给用户。Hive 定义了简单的类 SQL 查询语言(HQL)、解释器、编译器、优化器来完成 HQL 查询语句的词法分析、语法分析、编译、优化以及查询计划的生成。生成的查询计划存储在 HDFS 中，并随后由 MapReduce 调用执行。在 Spark 中，Filter[12] 操作将数据中不满足条件的属性过滤掉，返回一个文件子集的新 RDD(弹性数据集)，最终生成 HDFS 文件。

WordCount 对输入的数据进行统计计数，对相同的属性进行累加求和。在 Hadoop 平台执行 WordCount 时，将载入的文件按行拆分为〈key，value〉对，交给 map 方法进行处理，将 key 值相同的 value 值做累加操作；然后 reducer 对数据进行排序，最终得到输出的结果。Aggregation 聚集算法是针对立即执行的连续查询而提出的，数据流到一个元组后立即进行一次聚集操作。利用第 N 次的聚集值计算第 $N+1$ 次的聚集值，可以提高查询的效率。AggQuery 的操作过程与 Aggregation 类似，故本文将具有聚集操作的 WordCount、Aggregation 和 AggQuery 归为一组(如表 2 c 所列)。

在表 2 d 栏中，Join 和 Cross Product 都是将两个关系模型拼接成为一个更宽的关系模型。在 MapReduce 中，Join[16] 在 map 阶段，map 函数同时读取两个文件 File1 和 File2，为了区分两种来源的 key/value 数据对，对每条数据打一个标签(tag)，比如：tag=0 表示来自文件 File1，tag=2 表示来自文件 File2，即 map 阶段的主要任务是对不同文件中的数据打标签。在 reduce 阶段，reduce 函数获取 key 相同的来自 File1 和 File2 文件的 value list，然后对于同一个 key，对 File1 和 File2 中的数据进行 Join，即 reduce 阶段进行实际的连接操作。在 Hive 中执行 Join 操作的过程如下：先载入数据集，然后按指定规则进行匹配，输入表之间的每次匹配都会在输出表里生成一行。在 reduce 阶段，位于 Join 操作符左边的表的内容会被加载进内存，再通过最后一个表将结果序列化到文件系统，所以把数据量最大的表放最后能减少内存的使用量。Hive 对 Cross Product[12](笛卡尔积)支持较弱，由于找不到 Join key，Hive 只能使用一个 reducer 完成笛卡尔积，不能通过 Join key 划分数据，因此当 Hive 设定为严格模式(hive.mapred.mode=strict)时，不允许在 HQL 语句中出现笛卡尔积[13]。

如表 2 e 栏所列，Hive 将 Union 封装在子查询中，Union

对多个 select 语句的结果集执行连接操作，合并为一个独立的结果集。Hive 将多表 Union all 优化成一个 job，减少了 Map/Reduce 个数，因此有更好的性能。相当于集合论中的差运算，Difference[13]可以将一个数据集中属于另一个数据集的元素过滤掉，形成一个新的数据集，然后输出 HDFS 文件。Projection(投影)操作是对一个给定的结果集去掉若干属性，并可按照特定的顺序重新排列数据。

在表 2 f 栏中，K-Means 算法是常用的聚类算法。基于 Spark 平台实现 K-Means 算法，RDD 抽象划分大量的数据样本分布至 P 个计算节点，聚类中心通过 SparkContext 的 broadcast 方法在各个计算节点间共享，使用 map()函数完成分配步骤，再使用 reduce()函数完成更新均值中心的步骤。将上述过程进行迭代至分配步骤不再发生改变时，K-Means 算法收敛。NaiveBayes[14](朴素贝叶斯)是一种常用的分类算法，其先计算训练数据和待分类数据的先验概率、条件概率以及后验概率，然后对后验概率进行比较，选择具有最大后验概率的类作为对象所属的类。MapReduce 实现 PageRank[6]算法所做的操作如下：在 map 阶段，map 操作的每一行对所有出链发射当前网页概率值的 $1/k$(k 为当前网页的出链数)。在 reduce 阶段，reduce 操作收集网页 id 相同的值，并做累加操作。将上述过程进行迭代，由公式即可获得当前各个网页的 PR 值。基于 Spark 平台实现 PageRank 算法时，将当前的各个网页节点作为对象组成 RDD 抽象[15]，通过 RDD 对象的 groupByKey 方法组织成邻接表结构，该邻接表由于会在计算中反复用到，因此可以缓存在内存中。然后调用 RDD 对象的 Join 方法，使得同一节点的 rank 值和出链节点列表组织在一起，更新 rank 值后输出，重复迭代以上步骤至最后的结果收敛[14]。

表 2　负载对比

编号	离线分析	交互式查询
a	Sort	Order by
b	Grep	Select;Filter
c	WordCount	Aggregation;AggQuery
d		Join;Cross Product
e		Union;Difference;Projection
f	PageRank;K-Means;NaiveBayes	

3　参数度量

对计算机体系结构的特征参数进行了分析，包括 L1、L2、L3 缓存相关，页表缓存 TLB 和内存访问相关，分支执行，非内核请求 Offcore Request，监听请求，Parallelism 参数，操作强度特征等总共 45 个特征参数；并提取了一些重要的特征参数进行归类整理，如表 3 所列，其整体上分为与缓存相关和缓存非相关两大类，并对每个特征参数的物理意义及对运行程序的影响进行了描述。

3.1　重要参数度量

有些参数相互之间有冗余部分，须去掉冗余部分。如用户模式和内核模式下，运行的程序在用户模式下运行的比例越高说明程序的性能越好，安全性更高。指令缓存和 TLB 是两个重要的部分，它们提供了容量预取单元及加速指令的提取。在数据中心第三方库和高级编程语言的使用使得负载的 L1 指令缓存和 TLB 指令的性能恶化[4]。所以当用第三方库和高级语言编写程序时，工程师应该注意代码量的大小。相关研究结果表明[3,4]：由于 LLC 在片上芯片作为最大的元件，因此产生的每一代处理器的容量会相应地增加。在现在的处理器中用于高速缓存的面积占了差不多一半。在基于 Map-Reduce应用的试验中，LLC 是主要的限制者。从节约芯片的面积、提高处理器的效率考虑，有必要优化 LLC。在现在的故障处理器中，为了预测下一个分支以避免流水停顿，在处理器设计中使用了函数单元。流水线能去除错误的指令，提取正确的指令。采用流水线技术后，并没有加速单条指令的执行，每条指令的操作步骤一个也不能少，只是多条指令的不同操作步骤同时执行，因而从总体上看加快了指令流速度，缩短了程序执行时间。当分支预测未命中率发生时，它将导致大量的周期浪费。对于分析类负载，预测错误率比大多数服务类负载要低。这预示着现代处理器的分支预测有不错的效果。一个简单的分支预测器在节约能量和芯片面积时更加有优势。

同时，L2 缓存在各平台上是低效的，而对于数据分析类负载，L2 缓存是有效的，它们有较低的 L2 缓存未命中率，服务类负载高于 HPCC 负载[3,4]。随着数据容量的增加，对于不同的应用，Cache 和 TLB 行为有不同的趋势。例如，每一千条指令中 L1 指令未命中的数量随着输入数据的容量的增加，Sort 增加，而 Grep 下降[1,2]。然而，对于不同的应用，当数据容量增加到一个确切的值时，未命中率达到稳定值。操作强度是测量指令总数与内存访问总字节数量的比值，整型操作强度指的是整型指令总数与内存访问总字节数量的比值，浮点型操作强度指的是浮点型指令总数与内存访问总字节数量的比值。大数据负载中，整型操作强度与浮点型操作强度的比值非常高。SPECFP 有相对高的操作强度，因为它是以浮点型操作为导向的。在 Xeon E5310 和 Xeon E5645 中，与传统的负载相比，大数据负载的浮点操作强度低两个数量级[3]。在 E5645 中，BigDataBench 的浮点操作强度高于 E5310，原因是 L3 级缓存在降低内存访问量上是有效的。大数据负载的平均整型操作强度与其他 Benchmark 数量级相同。大数据负载在 BigDataBench 中其计算指令与内存访问的比例比较低，这主要可以从两个方面解释：首先大数据处理严重地依赖于内存访问；其次大数据负载必须处理大容量的数据，因此大部分大数据负载使用低计算复杂度的简单算法。我们也可以得出结论：大数据负载数据移动操作比指令执行操作要多。

总之，对运行程序的性能分析绝大多数是由未命中率、分支预测率以及停顿相关的特征参数决定的，其中，未命中率的特征参数在分析实验性能和缓存设计中是比较重要的度量参数。例如，在缓存相关的特征参数中，缓存命中率越高说明程序在该部分运行的性能越好。当缓存未命中率比较高时，未命中率的特征参数为改进和分析运行程序的性能提供了依据。CPU 流水线技术将指令分解为多步，当指令之间产生数据冲突时，会产生停顿，直到获得所需的资源。当预测失效时，先把流水排空，再用正确的指令填充。流水线技术就是为了解决指令之间产生的数据冲突、资源争夺[17]，它使得 CPU 的处理能力大大提高。如表 3 所列，对缓存相关的和缓存非相关的重要特征参数进行了总结，包括一级缓存未命中、二级缓存未命中、三级缓存未命中、载入未命中、页表缓存未命中行为、分支执行未命中行为、操作强度等特征参数。

表 3　重要体系结构特征参数度量

	类型	编号	度量名称	物理意义及对运行程序的影响
缓存相关特征参数	一级缓存未命中行为	1	L1I MISS	一级指令缓存未命中：每 K 条指令中，L1I 缓存未命中。在数据中心第三方库和高级编程语言的使用使得负载的 L1 指令缓存和 TLB 指令的性能恶化[4]
	二级缓存未命中行为	2	L2 MISS	二级缓存未命中：每 K 条指令中，L2 Cache 未命中。二级缓存未命中比一级缓存未命中对性能的影响更大，而且二级缓存问题通常比一级缓存问题更容易解决。L2 缓存在各平台上是低效的，而对数据分析类负载，L2 缓存是有效的
	三级缓存未命中行为	3	L3 MISS	三级缓存未命中：每 K 条指令中，L3 Cache 未命中
	页表缓存未命中行为	4	ITLB MISS	指令转换后备缓冲器：每 K 条指令中，在 TLB 各级指令未命中
		5	DTLB MISS	每 K 条指令中，在 DTLB 各级未命中指令
非缓存相关特征参数	用户模式	6	USER MODE	在用户模式下指令运行的比例
	载入未命中行为	7	LOAD HIT LFB	载入(loads)命中行填充缓冲区但未命中一级数据缓存：每 K 条指令中，载入(loads)命中行填充缓冲区但是未命中 L1D
		8	LOAD LLC MISS	每 K 条指令中，载入未命中 L3 Cache，MapReduce 应用的实验中 LLC 是关键的限制者
	分支执行未命中行为	9	BR MISS	分支预测未命中的比例，分支预测未命中率发生，它将导致大量的周期浪费
	流水线行为 Pipeline Behavior	10	FETCH STALL	指令提取停顿的周期与总周期的比
		11	ILD STALL	指令长度解码器停顿的周期与总周期的比
		12	DECODER STALL	解码器停顿的周期与总周期的比
		13	RAT STALL	寄存器分配表停顿的周期与总周期的比
		14	RESOURCE STALL	资源相关停顿的周期与总周期的比，这包括加载存储缓冲区完全阻塞，保留站(Reservation Station)的完全阻塞，reoder 缓冲区完全阻塞等
	操作强度 Operation Intensity	15	INT TO MEM	整型计算指令数与内存访存指令数的比例，L3 级缓存在降低内存访问量上是有效的。大数据负载平均整型操作强度与其他 Benchmark 相比有相同数量级
		16	FP TO MEM	浮点计算指令数与内存访存指令数的比例，SPECFP 有相对高的操作强度由于它是以浮点型操作为导向

4　实验结果与分析

基于 Hive 平台和 Spark 平台，本文对表 1 中两大类 16 种负载进行实验，实验中的参数利用 perf 工具抓取，抓取的参数包括 L1、L2、L3 缓存相关，页表缓存 TLB 相关，分支执行，非内核请求 Offcore Request，监听请求，Parallelism 参数，操作强度特征参数等总共 45 个微体系结构特征参数，将抓取后的所有特征参数进行降维或聚类处理，将降维后的特征作为聚类处理的输入。聚类处理后的结果通过命令行的交互式绘图工具 gnuplot 显示，算法用类似 shell 的语言实现。其中降维用到了 Principal Component Analysis (PCA)算法，聚类用到了 SimpleKMeans 算法，SimpleKMeans 算法有不同的产生初始聚类中心点的方式，本实验采用了 Random 方式和 Kmeans++方式。以下是对实验结果的分析。

基于 Hive 平台，降维后使用 Random 方式聚类，k 值(聚类数量值)分别为 6 和 8 时，产生的差异是 AggQuery 从 cluster0 中分离，单独成为一类 cluster6；Grep 从 cluster3 中分离，单独成为一类 cluster7；Aggregation 从 cluster5 中分离进入 cluster0，如图 1 所示。

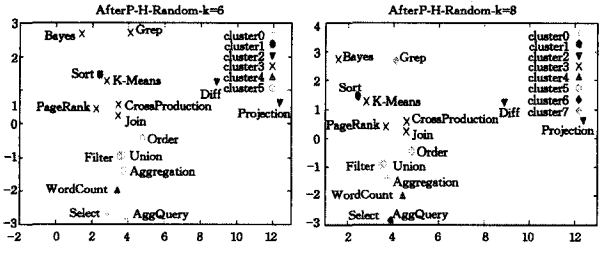

图 1　AfterP-H-Random-k=6 和 AfterP-H-Random-k=8

基于 Hive 平台，降维后使用 Kmeans++方式聚类，k 值分别为 6 和 8 时，产生的差异是 Filter 和 Union 从 cluster0 中分离，单独成为一类 cluster2；Kmeans 从 cluster5 中分离，单独成为一类 cluster6；Bayes 从 cluster2 中分离，单独成为一类 cluster7，如图 2 所示。

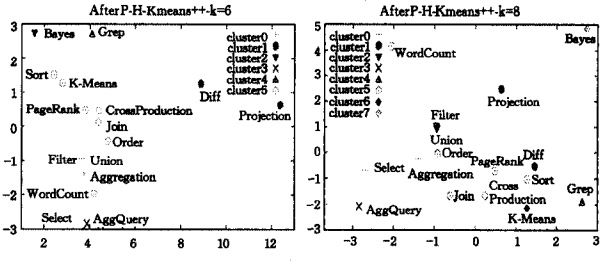

图 2　AfterP-H-Kmeans++-k=6 和 AfterP-H-Kmeans++-k=8

上述基于同一组数据的聚类之所以产生不同的结果是因为使用了不同的产生初始聚类中心点的方式。Random 方式是将数据集随机划分为指定的 k 个部分，进而产生初始聚类中心点，是一种最常用的经典的产生方式；Kmeans++方式思路如下：首先从数据集中随机选取一个点作为中心点并将其加入中心点集合 centers；对于数据集中每个点 i，都分别与集合 centers 中的点计算距离，并选出最小距离 $d[i]$，一轮计算完成后得到 $sum(d[i])$；取一个随机值 $random$ 使其落在 $sum(d[i])$ 内，然后 $random$ 不断减去 $d[i]$，直至 $random$ 小于 0 时将 $d[i]$ 对应的点加入中心点集合 centers；重复上述过程直到完成中心点的选取。

基于 Spark 平台，使用 Random 方式聚类，k 值取 6 时，降维前后的聚类结果差别如下：降维后 Kmeans 从 cluster0 分离进入 cluster5，如图 3 所示。

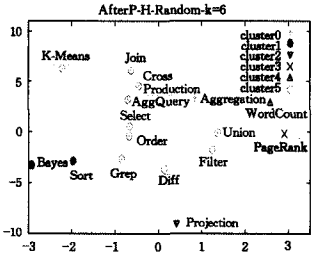

图 3　AfterP-S-Random-k=6

基于 Spark 平台，使用 Kmeans++方式聚类，k 值取 6

时，降维前后的聚类结果差别如下：Sort 从 cluster1 分离进入 cluster3，Kmeans 从 cluster3 分离进入 cluster5，Bayes 从 cluster4 分离进入 cluster5，Filter 从 cluster5 分离进入 cluster1，如图 4 中左图所示。

基于 Spark 平台，使用 Kmeans＋＋方式聚类，k 值取 8 时，降维前后的聚类结果差别同上。

基于 Spark 平台，降维后使用 Kmeans＋＋方式聚类，k 值取 6 和 8 时，产生的差别如下：PageRank 从 cluster3 分离，单独成为一类 cluster7；Kmeans 从 cluster5 分离，单独成为一类 cluster6，如图 4 中右图所示。

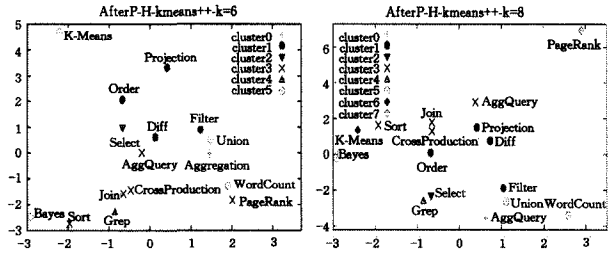

图 4　AfterP-S-Kmeans＋＋-k=6 和 AfterP-S-Kmeans＋＋-k=8

基于 Spark 平台，降维前使用 Kmeans＋＋方式聚类，k 值取 6 和 8 时，产生的差别有：Kmeans 和 PageRank 从 cluster3 中分离出来，分别单独成为一类，Kmeans 成为一类 cluster6，PageRank 成为一类 cluster7。

结束语　本文采用 PCA 和 SimpleKMeans 算法，对各个负载的体系结构特征参数实验结果进行降维和聚类处理，其中 SimpleKMeans 算法采用 Random 和 Kmeans＋＋两种初始聚类中心的方式进行处理。实验结果表明有些负载之间有公共的操作集，如基于 Hive 平台，在 Random 聚类处理方式和 Kmeans＋＋聚类处理方式中，$k=6$ 和 $k=8$ 前后，Join 和 CrossProduction 都在一个 cluster 中。基于 Spark 平台，在 Kmeans＋＋聚类处理方式中，$k=6$ 和 $k=8$ 前后，Join 和 CrossProduction 也都在一个 cluster 中。同时，实验结果表明有些负载之间有相似的属性，共享许多相似的微体系结构特征。如图 1、图 2、图 4 中，对于降维后再聚类的交互式查询类负载 Difference 和 Projection，在 $k=6$ 和 $k=8$ 前后，其都在一个 cluster 中，不会随着聚类 k 值的不同而变化。交互式查询类负载不会随着聚类处理方式 k 值的不同而变化，部分原因可以归纳为：交互式查询类负载里面有些数据操作集缓存在内存中，以便于下次执行相同操作时使用。同时实验结果表明，某些单个负载中存在多个操作集属性，如离线分析类负载中的 K-means 和 PageRank。总之，通过研究 16 种大数据负载，发现它们之间有公共的操作集，负载之间有共享的微体系结构特征，某些单个负载中存在多个操作集属性。这对于应用程序的优化以及对大数据基准测试平台的设计具有重要的指导意义。

我们未来的研究工作如下：1）继续从各离线分析类以及交互查询类大数据负载方面对它们的相似特征进行分析。2）设计适应时空数据的大数据基准测试平台。3）使得平台适合测试现实世界的数据，为研究者提供更加准确的测试标准。4）研究适合测试混合负载形式的 MixBenchmark 大数据基准测试平台，以测试分布式系统的计算能力。

参 考 文 献

[1] Wen Xiong, Yu Zhi-bin, Bei Zhen-dong, et al. A characterization of big data benchmarks[C]// 2013 IEEE International Conference on Big Data. 2013: 118-125

[2] Gao Wan-ling, Zhu Yu-qing, Jia Zhen, et al. BigDataBench: a Big Data Benchmark Suite fromWeb Search Engines[C]// The Third Workshop on Architectures and Systems for Big Data (ASBD 2013) in Conjunction with The 40th International Symposium on Computer Architecture. 2013

[3] Wang Lei, Zhan Jian-feng, Luo Chun-jie, et al. BigDataBench: A big data benchmark suite from internet services[C]// 2014 IEEE 20th International Symposium on High Performance Computer Architecture (HPCA). 2014: 488-499

[4] Jia Zhen, Wang Lei, Zhan Jian-feng, et al. Characterizing data analysis workloads in data centers[C]// 2013 IEEE International Symposium on Workload Characterization (IISWC). IEEE, 2013: 66-76

[5] White T. Hadoop: The Definitive Guide(Second Edition)[M]. San Francisco: O'Reilly Media, 2011

[6] Han Jia-wei, Kamber M, Pei Jian. Data Mining: Concepts and Techniques(Third Edition)[M]. San Francisco: Elsevier, 2012

[7] ICTBench[OL]. http://prof.ict.ac.cn/ICTBench/

[8] Shark[OL]. http://shark.cs.berkeley.edu/

[9] Spark[OL]. http://spark.apache.org/

[10] Hive[OL]. http://hive.apache.org/

[11] 冯琳. 集群计算引擎 Spark 中的内存优化研究与实现[D]. 北京：清华大学，2013
Feng Lin. Research and Implementation ofMemory Optimization Based on Parallel Computing Engine Spark[D]. Beijing: Tsinghua University, 2013

[12] 赵龙，江荣安. 基于 Hive 的海量搜索日志分析系统研究[J]. 计算机应用研究，2013，30(11)：3343-3345
Zhao Long, Jiang Rong-an. Research of massive searching logs analysis system based on Hive[J]. Application Research of Computers, 2013, 30(11): 3343-3345

[13] 叶文宸. 基于 hive 的性能优化方法的研究与实践[D]. 南京：南京大学，2011
Ye Wen-chen. The Research and Practice of Performance Optimization Based on Hive[D]. Nanjing: Nanjing University, 2011

[14] 唐振坤. 基于 Spark 的机器学习平台设计与实现[D]. 厦门：厦门大学，2014
Tang Zheng-kun. Design and Implementation of Machine Learning Platform Based on Spark[D]. Xiamen: Xiamen University, 2014

[15] 刘记云. 基于 MapReduce 的个性化 PageRank 算法研究[D]. 哈尔滨：哈尔滨工程大学，2013
Liu Ji-yun. A Research on Personalized PageRank Based on MapReduce[D]. Harbin: Harbin Engineering University, 2013

[16] 李林. 基于 Hadoop 平台的视觉数据聚类研究与实现[D]. 西安：西安电子科技大学，2013
Li Lin. Research and Implementation of Clustering on Visual Data Based on Hadoop[D]. Xi'an: XiDian University, 2013

[17] 黄永兵，陈明宇. 移动设备应用程序的体系结构特征分析[J]. 计算机学报，2015：38(2)：386-396
Huang Yong-bing, Chen Ming-yu. Architecture Characteristics and Analysis of Mobile Device Applications[J]. Chinese Journal of Computers, 2015, 38(2): 386-396