

# 基于 Spark 的层次聚类算法的并行化研究

余胜辉, 李玲娟

(南京邮电大学 计算机学院, 江苏 南京 210023)

**摘要:**随着大数据时代的来临,传统的计算模式已经不足以支撑如此大量的数据。基于内存计算的大数据并行化计算框架 Spark 的出现很好地解决了这一问题。CURE 是一种基于取样和代表点的层次聚类算法,它采用迭代的方式,自底向上地合并两个距离最近的簇。与传统的聚类算法相比,CURE 算法对异常点的敏感度更小。但是在处理大量数据的情况下,CURE 算法存在着因反复迭代而消耗大量时间的问题。文中利用了 Spark 的 RDD 编程模型的可伸缩性和分布式等特点,实现了对 CURE 算法计算过程的并行化,提升了该算法对数据的处理速度,使算法能够适应数据规模的扩展,并且提高了聚类的性能。在 Spark 上运用 CURE 算法对公开数据集的并行化处理结果表明,基于 Spark 的 CURE 算法并行化既保证了聚类准确率又提高了算法的时效性。

**关键词:** Spark; 层次聚类; CURE; RDD; 并行化

**中图分类号:** TP301.6

**文献标识码:** A

**文章编号:** 1673-629X(2020)06-0019-04

**doi:** 10.3969/j.issn.1673-629X.2020.06.004

## Research on Parallelization of Hierarchical Clustering Algorithm Based on Spark

YU Sheng-hui, LI Ling-juan

(School of Computer Science, Nanjing University of Posts and Telecommunications, Nanjing 210023, China)

**Abstract:** With the advent of the era of big data, traditional computing models are not enough to support such a large amount of data. The emergence of Spark, a big data parallel computing framework based on in-memory computing, solves this problem well. CURE is a hierarchical clustering algorithm based on sampling and representative points, and uses an iterative method to merge two closest clusters from the bottom up. Compared with traditional clustering algorithm, CURE algorithm is less sensitive to outliers. However, in the case of processing large amounts of data, the CURE algorithm has the problem of consuming a lot of time due to repeated iterations. We utilize the scalability and distributed characteristics of Spark's RDD programming model to realize the parallelization of the computing process of CURE algorithm, which improves the speed of data processing, makes the algorithm adapt to the expansion of data scale, and improves the performance of clustering. The parallelization of the public dataset using CURE algorithm on Spark shows that the parallelization of Spark-based CURE algorithm not only ensures the clustering accuracy but also improves the timeliness of the algorithm.

**Key words:** Spark; hierarchical clustering; CURE; RDD; parallelization

## 0 引言

大数据时代的来临导致行业领域产生的信息数据呈爆炸式的增长。对海量数据的挖掘以及利用成为了当下最热门的重点研究课题之一。聚类方法是数据挖掘领域中最为重要的方法之一,是一种无监督的机器学习算法,能够将数据对象中具有相同属性的数据划分为若干个子集,每个子集形成一个簇,同一个簇中的数据具有相似的特征,不同簇中的数据相异<sup>[1]</sup>。层次聚类算法是聚类算法中最常见的一类方法,“类内的点都足够

近,类间的点都足够远”<sup>[2]</sup>是其最重要的判断标准。CURE 算法是一种典型的层次聚类算法,该算法简单、速度快,对大数据集有较高的聚类效率和可伸缩性,时间复杂度近于线性,适合对大规模数据集进行挖掘<sup>[3-4]</sup>。Spark<sup>[5-6]</sup>是专门为大规模数据处理设计的快速通用计算引擎,是以 Hadoop MapReduce 为基础框架的一个新的开源平台。Spark 支持交互式计算和复杂算法,速度很快;具有很高的通用性;支持多种资源管理器。为了进一步提升 CURE 层次聚类算法的效率和可伸缩性,文

收稿日期:2019-07-05

修回日期:2019-11-07

网络出版时间:2020-01-10

基金项目:国家重点研发计划专项(2017YFB1401302,2017YFB0202200);国家自然科学基金(61572260,61872196)

作者简介:余胜辉(1995-),男,硕士研究生,研究方向为数据挖掘;李玲娟,教授,通讯作者,研究方向为数据挖掘、分布式计算。

网络出版地址: <http://kns.cnki.net/kcms/detail/61.1450.TP.20200110.1123.046.html>

中基于 Spark 平台对其进行并行化,并通过对不同的数据集的聚类实验来证明该并行化方案的有效性。

## 1 CURE 算法的原理

CURE 是一种基于取样和代表点的层次聚类算法,它采用迭代的方式,自底向上地合并两个距离最近的簇<sup>[7]</sup>。CURE 算法在传统的划分聚类算法受异常数据比较敏感这一特点上能够得到很好的解决。CURE 算法在计算过程中将每一类当成一个点,用这个点代表其中一个类型的数据,然后计算一个类与另一个类之间的距离,最相似的两个点进行合并,不需要计算类内每个数据点之间的距离,这样计算有利于减少运算量,提高效率。但此算法的复杂度较高,运行时所消耗的资源较多。CURE 算法的具体特征及思想主要体现在如下几个方面<sup>[8-9]</sup>:

(1)最初,每一个对象就是一个独立的类,然后从最相似的对象开始进行合并。

(2)为处理大数据集,采用了随机抽样和分割手段。采用抽样的方法可降低数据量,提高算法效率。一般能得到比较好的聚类结果。分割手段,即样本分割为几个部分,然后针对各个部分中的对象分别进行局部聚类,形成子类。再针对子类进行聚类,形成新的类。

(3)CURE 算法由分散的若干对象按收缩因子移向其所在类的中心之后来代表该类。由于 CURE 算法采用多个对象代表一个类,并通过收缩因子类调节类的形状,因此能够处理非球形的对象分布。

(4)消除异常值可以分两个阶段进行,第一个阶段的工作,是将聚类过程中增长非常缓慢的类作为异常值除去,第二个阶段的工作(聚类基本结束的时候)是将数目明显少的类作为异常值除去。

CURE 的具体操作流程如下<sup>[10]</sup>:

- (1)从原始数据中随机抽取样本,得数据集  $S$ 。
- (2)对  $S$  进行分区,对每个分区分别进行聚类。
- (3)如果一个类增长很慢,说明它是噪声要去除。
- (4)将局部的类进行聚类。
- (5)对原始数据进行标记。

## 2 Spark

Spark 是为了提高计算速度、易用性和复杂分析而构建的大数据处理框架。为了提高 Spark 处理大量数据的实时性,因此计算都是基于内存计算的,Spark 的集群一般都是部署在很多廉价的硬件上的,这种部署对容错性和可伸缩性都能得到保障。

(1)Spark 与传统大数据平台的对比。

在 Spark 之前已有 Hadoop 和 Storm 等大数据平台,这些大数据平台都是基于 MapReduce<sup>[11]</sup> 技术,与

之相比 Spark 具有十分明显的优势<sup>[12]</sup>:

①Spark 底层通过开辟了线程池,利用了线程池能够复用线程,不必反复创建销毁线程的优势,能够减少资源,大大减少了任务调度的开销。

②Spark 中的 RDD 可以理解为一个集合,可以在上面进行计算,传统数据模型的容错,可伸缩性等特点 RDD 都具有,也可将数据缓存,方便后续的重复使用。

③Spark 在每一轮计算过程中都会得到一个中间结果,因此可以将每一轮的输出结果在内存缓存,在后续的运算需要时,可以直接从缓存中对数据进行读取,代替了从 HDFS 中读取数据,从而避免了大量耗时的磁盘 I/O 操作,直接从内存里进行读取,对运行速度有很大的提升<sup>[13]</sup>。

(2)Spark 的组成。

Spark 主要包含以下几个部分<sup>[5]</sup>:

①Spark Core:这是 Spark 最为核心的一个部分,是一个通用分布式数据引擎,它自成一个体系,可适应在任何商用服务器集群上。对 Spark 一些基础功能进行了定义,尤其是 RDD 的 API,操作及这两者的动作。

②Spark SQL:是提供在大数据上的 SQL 查询语句,这一功能扩充了 SQL 算子,丰富了 Spark 的算子和功能。

③Spark Streaming:利用 Spark Core 的快速调度能力来执行流数据。它以最小批次获取数据,并对批次上的数据执行 RDD 转化,其核心是采用微批处理引擎,支持多种数据源的导入,可根据程序配置的时间,将数据打成一个 RDD,发送给 Spark Core 进行处理。

④MLlib:一个常用算法库,包含了几种常见的机器学习算法。

(3)Spark 的架构。

与传统经典的分布式计算架构模型类似,Spark 沿用了传统的架构模式:Master-Slave 模型<sup>[14]</sup>,如图 1 所示,Master 对应 ClusterManger 进程的节点,能够控制整个集群的运行,是集群中最不可缺少的部分。Slave 对应着 Worker 进程的节点,它的作用是对现在的形态向上汇报和接受特定的命令和向下进行分发具体的任务,Executor 是执行任务的具体部分进行运算,Client 是与用户进行交互的界面提供了可视化的功能,同时也负责任务的提交。Driver 相当于处理中心,进行对应用的执行。当新建一个任务被提交上来之后,Driver 节点会创建一个 SparkContext,由 SparkContext 向资源管理器(ClusterManger)申请资源,资源分配完毕后 Spark 会启动 Worker 上负责执行具体任务的进程 Executor,并把任务分给 Executor,计算完成后 Worker 会将结果发回 Driver,然后释放资源<sup>[15]</sup>。这就是一次具体的任务提交处理流程。

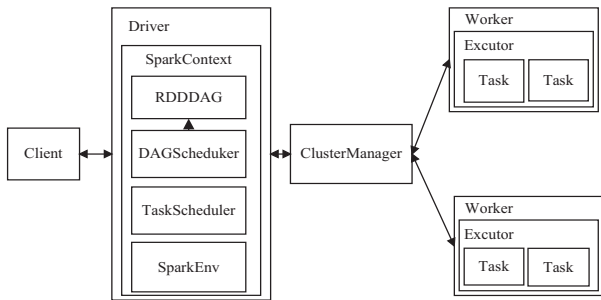


图1 Spark 架构

### 3 基于 Spark 的 CURE 算法的并行化方案

基于上述的分析,CURE 算法具有适合处理海量数据集,需要对数据进行分区处理,以及需要对数据进行反复迭代等特点,这与 Spark 大数据计算框架 RDD 的特性十分相契合。文中正好通过此特性,实现 CURE 算法基于 Spark 的并行化。

CURE 算法基于 Spark 的并行化步骤如下:

(1) Spark 的配置与数据源的读取。

构建 Spark Application 的运行环境后,配置文件会被 Spark 的驱动程序自行加载,首先会生成一个 SparkConf 对象,之后通过该对象又会自动生成一个新的 SparkContext,Executor 资源就是通过 SparkContext 向资源管理器注册并申请运行的。通过上述的分析,得知 Spark 计算流程实际上是将待处理的数据集读取后转换成为源 RDD,然后通过 Spark Core 提供的 transformation 算子,对该 RDD 进行转换来获取新的 RDD,最后调用 Action 操作求得结果值。使用 HDFS 文件创建 RDD 是最常用的方式,可以针对 HDFS 上存储的大数据,进行离线批处理操作。在 RDD 计算过程中,每个分区都会运行一个 task,所以 RDD 的分区数目决定了总的 task 数目,在 Spark 集群资源有限的条件下,需要对 RDD 进行合理的分区,分区太多或太少都会导致计算效率的降低。

(2) CURE 算法并行化执行的流程。

Step1:对原始数据进行分区,随机分成  $N$  个数据片,将数据读入 HDFS 转化成 DataFrame。相比于 RDD,DataFrame 能提供更为详细的结构信息,使得 Spark SQL 可以清楚地知道数据的详细信息。DataFrame 相比 RDD 不仅提供了更丰富的算子,最重要的是可提升执行效率。

Step2:采用收缩因子的方法为每个数据块选取代表点,通过 mapPartitions 对  $N$  个数据片进行 MapReduce 聚类计算,计算出中心点。为了提升聚类结果的准确度有两种删除异常点的操作:①如果出现两次及以上的聚类计算则会统计该类的增长速度,发现增长速度慢的类则当作噪声去除;②当类内的数目

低于某一个阈值时也会进行删除,因为有可能恰好较近的几个异常点在同一个类中。

Step3:对上一步聚类产生的类,通过拉格朗日距离来计算它们之间的中心点距离,并将距离最小的两个类合并,合并之后得到一个新的类。

Step4:对新的类重新进行 MapReduce 计算,得到新类的中心点以及代表点。

Step5:判断聚类后的个数是否满足设定的聚类数,如果不满足则继续重复 Step2,如果满足则得到最终聚类结果。

CURE 算法基于 Spark 的并行化流程如图2所示。

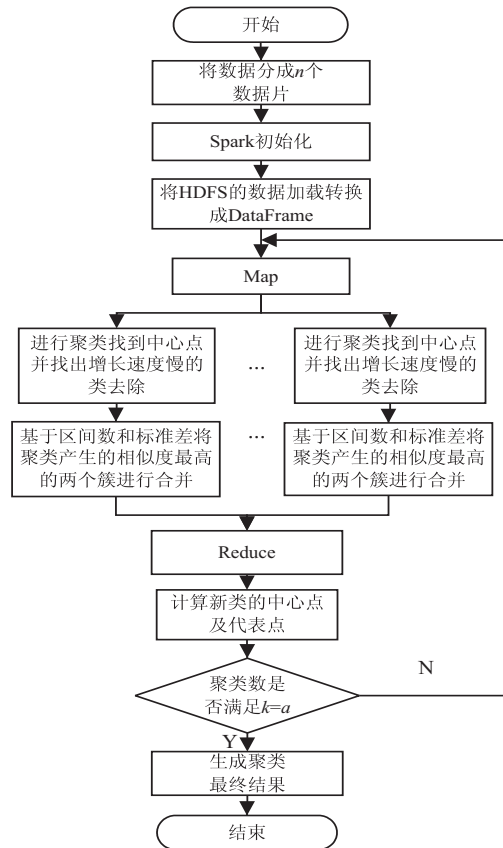


图2 CURE 算法并行化流程

CURE 算法本身在不确定的数据上的计算时间消耗可能会非常大,在上述执行过程中通过 Spark 平台 MapReduce 以及缓存特性,执行器会将需要缓存的数据缓存到内存中,调用驱动器节点对需要处理的数据进行再次计算,当达到终止条件时会将计算好的结果存储在 HDFS 中。Spark 的核心就是 RDD 是分发到不同 executor 节点上进行并行计算,并将中间结果缓存到内存中,对需要大量迭代重复计算的 CURE 算法在时间效率上有很大的提升。

### 4 实验与结果分析

为了验证和分析基于 Spark 的并行化 CURE 算法聚类结果的准确性与时间效率,设计了以下实验,分别



在单机和 Spark 上对 CURE 算法对相同大小的实验数据集进行聚类操作,比较对多种数据集在不同运行模式下的聚类准确率和时间效率。

### (1) 实验环境和实验数据。

实验环境如下:共搭建了包括 1 台驱动器节点,2 台执行器节点的 Spark。节点的 CPU 是 Intel CORE i5-7440H,每个节点配有 2 个处理器,硬盘数据读写速度为 600.00 MB/s,驱动器节点的内存大小为 6 G,执行器节点的内存大小为 4 G。操作系统是 centos6.5;Java 版本是 JDK1.8.0\_144;Spark 版本为 1.6.0;Scala 版本为 2.11.0。实验分别采用了 UCI 实验室提供的 Bag of Words 数据集,该数据集包含五个文字集合,八百万个实例数;YouTube MuLtiView Video Games Dataset 数据集,该数据集包含大约 120 000 个实例,每个实例由 13 种要素类型描述;Plants 数据集,该数据集包含了 70 个属性,22 632 个实例数。

### (2) 聚类准确率对比实验。

通过 CURE 算法对上述的数据集进行聚类,聚类准确率如表 1 表示。

表 1 不同数据集上 CURE 算法的聚类准确率

数据集	属性数	运行方式	聚类准确率/%
Bag of Words	5	单机	80.28
		Spark	81.02
Plants	70	单机	78.96
		Spark	79.77
Youtube MuLtiView Video Games DataSet	13	单机	79.83
		Spark	80.28

由表 1 可以看出,对于三个数据集,基于 Spark 的并行 CURE 算法相对于传统单机运行模式的 CURE 算法的聚类准确率略微有所提高。可以看出分布式数据处理对结果的精准度和稳定性有较好的保障。

### (3) 时间效率对比实验的聚类时耗。

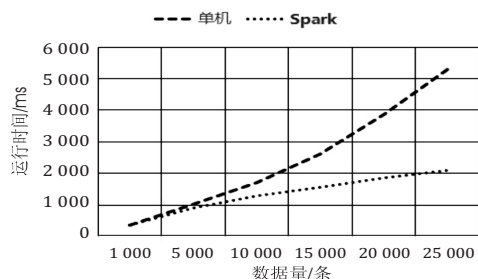


图 3 单机和基于 Spark 平台的 CURE 算法聚类时间对比

由图 3 可知,在开始阶段数据量比较少,两种不同环境的 CURE 聚类运行所消耗的时间相差不大,因为 Spark 启动消耗了大量的时间,在数据量较少的情况下无法体现其在时间方面的优势。伴随着数据量增大,基于 Spark 的并行化计算所消耗的时间明显少于传统的单机计算方式。因此,在处理大量数据的情况

下 Spark 的并行化计算具有良好的时效性。

## 5 结束语

主要研究了 CURE 算法基于 Spark 平台的并行化实现方案,给出了通过将数据集分区后分发给多个子节点和进行 RDD 转换来实现 CURE 算法的并行化步骤;通过在两种运行模式和三种公开数据集上的聚类实验,验证了 CURE 算法基于 Spark 的并行化运行比传统的单机运行的聚类准确率略有提升,且随着聚类数据量的增加,基于 Spark 平台的并行化 CURE 算法展示了更好的时效性。

### 参考文献:

- [1] 裴康. 数据挖掘中的聚类算法研究[D]. 北京:北京邮电大学,2014.
- [2] 贺玲,吴玲达,蔡益朝. 数据挖掘中的聚类算法综述[J]. 计算机应用研究,2007,24(1):10-13.
- [3] 曹泽文,周姚. 基于 MapReduce 的 JP 算法设计与实现[J]. 计算机工程,2012,38(24):14-16.
- [4] 魏桂英,郑玄轩. 层次聚类方法的 CURE 算法研究[J]. 科技和产业,2005,5(11):22-24.
- [5] 温馨,罗侃,陈荣国. 基于 Shark/Spark 的分布式空间数据分析框架[J]. 地球信息科学学报,2015,17(4):401-407.
- [6] 黎文阳. 大数据处理模型 Apache Spark 研究[J]. 现代计算机,2015(3):55-60.
- [7] GUHA S, RASTOGI R, SHIM K. Cure: an efficient clustering algorithm for large databases[J]. Information Systems, 2001,26(1):35-58.
- [8] AGRAWAL R, SRIKANT R. Fast algorithms for mining association rules[C]//International conference on very large data bases. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1994:487-499.
- [9] 董健康. 数据挖掘中 CURE 聚类算法研究[J]. 电脑与电信,2007(4):14-15.
- [10] 张磊,黄建华. CURE 算法在入侵检测系统中的应用研究[J]. 计算机安全,2007(11):14-16.
- [11] MO Y, CHEN J, XIE X, et al. Cloud-based mobile multimedia recommendation system with user behavior information[J]. IEEE Systems Journal, 2014,8(1):184-193.
- [12] 冯兴杰,王文超. Hadoop 与 Spark 应用场景研究[J]. 计算机应用研究,2018,35(9):2561-2566.
- [13] 邱荣财. 基于 Spark 平台的 CURE 算法并行化设计与应用[D]. 广州:华南理工大学,2014.
- [14] 胡俊,胡贤德,程家兴. 基于 Spark 的大数据混合计算模型[J]. 计算机系统应用,2015,24(4):214-218.
- [15] HAN Jiawei, PEI Jian, YIN Yiwen. Mining frequent patterns without candidate generation[C]//Proceedings of the ACM SIGMOD international conference on management of data. Dallas, Texas, USA: ACM, 2000:1-12.