Computer Science and Application 计算机科学与应用, 2020, 10(5), 824-831
Published Online May 2020 in Hans. http://www.hanspub.org/journal/csa
https://doi.org/10.12677/csa.2020.105085

Hans汉斯

Based on the Hierarchical Clustering Algorithm Research and Application of Spark

Weihua Liu[1], Tingting Shi[2*], Xuetian Xu[1]

[1]Guangdong Vocational College of Judicial Police, Guangzhou Guangdong
[2]College of Information Science and Technology, Zhongkai College of Agricultural Engineering, Guangzhou Guangdong
Email: to_shitingting@126.com

Received: Apr. 7th, 2020; accepted: Apr. 22nd, 2020; published: Apr. 29th, 2020

Abstract

In the era of rapid development of information technology, a large number of information data are generated. If they are not properly sorted and classified, they cannot meet the requirements of fast, convenient and accurate data search and use. With the development of information security science and technology, the demand for sorting and sorting of these data is increasing, but the traditional clustering algorithm can no longer meet the needs of current information data processing. Therefore, the optimization and improvement of the original algorithm or the reconstruction of a new algorithm has become the most urgent thing now. At the same time, on huge amounts of data processing, a single computer hardware facility cannot meet the demand of classification of data processing. According to the above situation, this article is based on the Spark in a distributed computing framework, on the basis of the clustering algorithm is optimized to improve. The use of Apache Spark's big data processing framework extends the use of the computing model, and provides a parallel computing framework in memory. By caching intermediate results in memory, the number of repeated disk I/O operations can be reduced, so as to better serve the needs of iterative computing, interactive query and other computing requirements. Through the optimization of clustering algorithm to improve the computational efficiency of data analysis, processing and classification, the significance of this study is realized.

Keywords

Spark, Clustering Algorithm, Optimization and Improvement, Big Data Processing

基于Spark的层次聚类算法的研究与应用

刘卫华[1]，史婷婷[2*]，许学添[1]

[*]通讯作者。

文章引用: 刘卫华, 史婷婷, 许学添. 基于 Spark 的层次聚类算法的研究与应用[J]. 计算机科学与应用, 2020, 10(5): 824-831. DOI: 10.12677/csa.2020.105085

[1]广东司法警官职业学院，广东 广州
[2]仲恺农业工程学院信息科学与技术学院，广东 广州
Email: to_shitingting@126.com

收稿日期：2020年4月7日；录用日期：2020年4月22日；发布日期：2020年4月29日

摘 要

信息化高速发展的时代，信息数据大量产生，如没得到较好的整理归类，就无法满足对数据查找和使用上的快捷便利与准确性。随着信息安全科学技术的发展，这些数据在整理分类上的需求日益增长，但是在传统的聚类算法上，已经不能满足现在信息数据处理的需要。因此，对原算法的优化改进或重建新的算法成为现在最为迫切的事情。同时，在海量的数据处理上，单台计算机的硬件设施也无法满足对数据处理分类的需求。针对上述情况，基于Spark在分布式计算框架的基础上，本文对聚类算法进行了优化改进。利用Apache Spark的大数据处理框架，扩展了对计算模型的使用，并在内存上提供可以并行的计算框架，利用借着中间结果缓存在内存中，减少磁盘I/O的重复操作次数，从而可以更好地为迭代式计算、交互式查询等多种计算需求服务。通过对聚类算法的优化提高对数据分析处理归类的计算效率，实现本文研究的意义。

关键词

Spark，聚类算法，优化改进，大数据处理

1. 引言

随着科学技术尤其是信息化技术的蓬勃发展，信息大数据如雨后春笋的增长。近十年以来，互联网一直处于社会发展的风口浪尖上，各大电商网络平台也在政府的大力支持帮助下，得到了快速成长。产业发展，不但拉动了经济的增长，也促使信息数据大量的产生。这些海量的数据在短时间内涌现，产生了数据混乱无序、杂乱无章的结果。因此，数据处理分析与分类管理就显的尤为重要，就目前的数据管理技术，对一般的基础数据尚能管理，但是无法满足海量大数据的管理分类要求，达到快速、高时效性的要求。

通过对数据基本分类划分，可分为结构化数据和非结构化数据。结构化数据处理分析技术已经比较成熟，但非结构化数据因其信息种类繁多、结构复杂以及数量巨大等原因，以现在的技术无法满足数据管理分析的要求，因此在技术开发处理上也亟待解决。通过利用对大数据处理工具的层次聚类算法的研究，将其应用于对海量大数据的信息处理分析，高效快速地分析整理出结果，实现数据的时效价值。

现在主流的大数据处理框架，比如 Spark 与 Hadoop 等工具。这些可扩展、分布式以及并行化的大数据处理工具渐渐进入政府、学校与企业中，取代之前相对落后的技术计算框架。基于内存计算的 Spark 框架是一款性能较好的数据处理工具。最早于 2009 年 Spark 大数据并行化处理框架被提出，并于下一年宣布开源。通过数个大数据公司的开发研究，Spark 的生态系统越来越完善，在性能提高的同时也简化其代码可读性和开发的难度。

聚类算法面对海量大数据，其在数据处理方面就无法满足信息化价值的要求，通过引进分布式计算框架技术，完美解决了聚类算法在处理海量大数据的不足。

Spark 拥有比较好的抽象化编程模型，单独处理除算法逻辑以外的其他所有问题，用户操作简单方便。同时，Spark 编程模型是将输入的大数据转化为自定义的 RDD，而且 RDD 拥有较多的操作算子，在满足一般的计算分析操作同时，也算对 RDD 的操作过程，所以所有操作计算过程都是 RDD 的迭代过程。Spark 通过合理充分利用计算机内存，减少计算过程中产生中间结果对磁盘 I/O 读写，提高计算效率。

2. 相关知识

2.1. Spark 分布式工具

2.1.1. Spark 简介

Spark 作为一个开源大数据计算框架，是以 Scala 为开发语言，利用 Scala 独特的函数式编程思想，提供较好的编程模型[1]；Spark 的运行模式是将自定义的程序，各自发送给集群的从节点，然后让 Worker 各自进行运算[2]。让 Spark 分布式编程成为真正简单化的开源软件。

Spark 在拥有 Map Reduce 的线性扩展与容错性同时，也进行了较多的本质扩展，大大提高了对内存利用的效率[3]。Spark 首先对 Map 进行操作，然后对 Reduce 进行编程模式；并优化 Spark 引擎，使其可以运行通用的有向无环图算子[4]。Spark 将运行的中间结果，通过内存传递给下一个程序，减少运行结果的多次写入次数。而且 Spark 还有一大特色，就是将弹性分布式数据收集 RDD 中，将代码中的任意数据都映射到集群节点中，让后续计算分析步骤相当于初始数据，无需对磁盘重新读取，减少运行时间[5]。

2.1.2. Spark 的构架

Spark 的核心机制 RDD 是基于分布式内存的并行数据结构，它将数据存储在内存中，通过控制分区划分，优化数据分布[6]。

Spark 支持八十几种高级的算法，也支持很多种计算机语言，其中就包括了 Java 和 Scala。Spark 兼容的文件系统有：Cassandra、Amazon S3、HDFS 与 HBase 等[7]。Spark 的解决方案用于机器学习、图计算、交互式查询、处理离线数据和实时数据流，并且这些处理方式可以在同一个程序中无缝对接[8]。

Spark 在集群计算中，将这些数据集缓存在各个节点的内存中，减少对磁盘 I/O 的操作，节省时间，提高计算效率[9]。

Spark 拥有较好的兼容性，与其它开源软件可以很方便进行融合。因其自身内部调度框架是 Standalone 模式，所以 Spark 不会依赖第三方资源管理和调度系统[10]。

Spark 的运行构架见图 1 所示。

2.2. 聚类算法的类型

聚类算法的划分，可以分为六类：层次聚类，密度聚类，网格聚类，模型聚类，图聚类和划分聚类等。这些聚类算法各有各自的优势与特点，它们所对应的模式也各不相同，一般根据实际状况、数据的类型、结构和分析目的要求等来选取对应的聚类算法[11]。

3. 聚类算法的优化使用

聚类算法是众多数据挖掘中的一种比较高级的算法，它也算无监督学习算法[12]。聚类算法是将数据对象集划分为数个组或者多个簇的算法过程，它使划分后的各组或各簇内的集拥有较高的相似性，而各组或各簇之间亦不相似[13]。

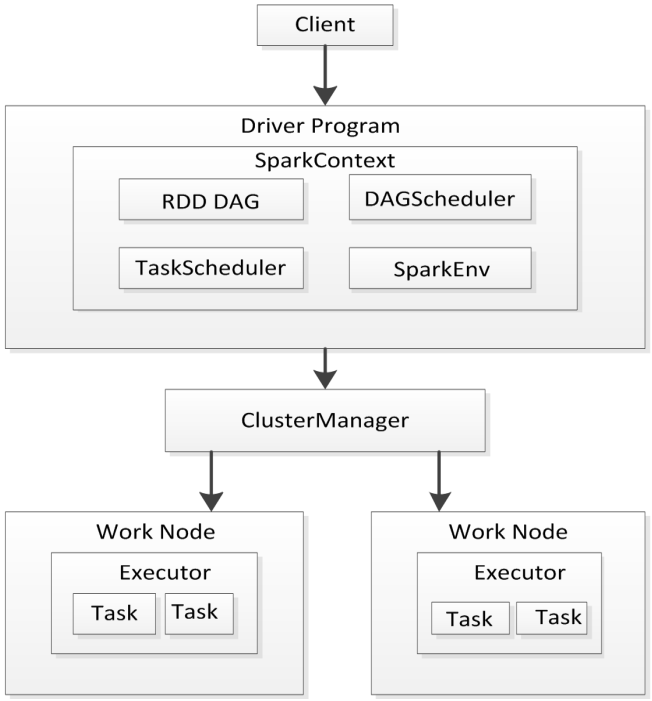

Figure 1. Shows the running architecture of Spark
图 1. Spark 的运行构架图

3.1. 聚类算法中的距离

3.1.1. 切比雪夫距离

以两个点 p 与 q 作为例，其对应的坐标就为 p_i，q_i。这两个点之间的切比雪夫距离为公式 1：

$$d(p,q)=\max\left(\left|p_i-q_i\right|\right) \tag{1}$$

此距离相当于 $\mathcal{L}p$ 度量的极值：$\lim_{n\to\infty}\left(\sum_{i=1}^{n}\left|p_i-q_i\right|^k\right)^{1/k}$，因此切比雪距离又可称为 $\mathcal{L}\infty$ 度量。

a) 在平面中，如果两个坐标点是 $a(x_1,y_1)$，$b(x_2,y_2)$，则切比雪距离为：

$$d_{12}=\max\left(\left|x_2-x_1\right|,\left|y_2-y_1\right|\right) \tag{2}$$

b) 在 n 维空间中，两个坐标点为 $a(x_{11},x_{12},\cdots,x_{1n})$，$b(y_{11},y_{12},\cdots,y_{1n})$，则切比雪距离为：

$$d_{12}=\max\left(\left|x_{1i}-x_{2i}\right|\right) \tag{3}$$

根据上述的 $\mathcal{L}p$ 度量极值 $\mathcal{L}p$，因此该公式的等价公式是：

$$d_{12}=\lim_{k\to\infty}\left(\sum_{i=1}^{n}\left|p_i-q_i\right|^k\right)^{1/k} \tag{4}$$

3.1.2. 欧式距离

欧式距离的定义是在 n 维空间上，两个点中间的真实距离。比如以下两点 $x(x_1,x_2,\cdots,x_n)$ 和 $y(y_1,y_2,\cdots,y_n)$ 之间的距离为：

$$\begin{aligned}d(x,y)&=\sqrt{(x_1-y_1)^2+(x_2-y_2)^2+\cdots+(x_n-y_n)^2}\\&=\sqrt{\sum_{i=1}^{n}(x_n-y_n)^2}\end{aligned} \tag{5}$$

a) 两个二维向量的点 $a(x_1,y_1)$， $b(x_2,y_2)$ 之间的欧式距离：

$$d_{12}=\sqrt{(x_1-y_1)^2+(x_2-y_2)^2} \tag{6}$$

b) 两个三维向量的点 $a(x_1,y_1,z_1)$， $b(x_2,y_2,z_2)$ 之间的欧式距离：

$$d_{12}=\sqrt{(x_1-y_1)^2+(x_2-y_2)^2+(z_1-z_2)^2} \tag{7}$$

c) 两个 n 维向量的点 $a(x_{11},x_{12},\cdots,x_{1n})$， $b(x_{21},x_{22},\cdots,x_{2n})$，之间的欧式距离：

$$d_{12}=\sqrt{\sum_{k=1}^{n}(x_{1k}-x_{2k})^2} \tag{8}$$

3.2. BIRCH 聚类算法

BIRCH 全名是 Balanced Iterative Reducing and Clustering Using Hierarchies，利用利用层次方法的平衡迭代规约和聚类[14]。这种算法属层次聚类算法，在相同内存容量，它可以处理更多的数据；在相同数据量下，它的处理速度更快[15]。因其具有独特的结构逻辑，一次扫描数据就能完成对数据的分析，用最小化的 I/O 处理，换成高质量聚类数据[16]。

CF 向量是三元组的簇聚类特征[17]，依据给定聚类数据集，*N* 个 *d* 维数据点：

$$CF=\langle n,LS,SS\rangle \tag{9}$$

关于簇的形心 x_0，直径 *D* 和半径 *R*，分别做公式的推导，通过数据和的平均值：

$$x_0=\frac{\sum_{i=1}^{n}x_i}{n}=\frac{LS}{n} \tag{10}$$

直径 *D* 是簇中每两点之间的平均距离：

$$D=\sqrt{\frac{\sum_{i=1}^{n}\sum_{j=1}^{n}(x_i-x_j)^2}{n(n-1)}}=\sqrt{\frac{2nSS-2LS^2}{n(n-1)}} \tag{11}$$

半径是簇中数据点到形心的平均距离：

$$R=\sqrt{\frac{\sum_{i=1}^{n}(x_i-x_0)^2}{n}} \tag{12}$$

通过直径和半径来观察簇中数据的聚集紧密程度，用作判断聚类效果的标准，从中挑选出最优聚类簇[18]。

3.3. 聚类算法的优化

针对 BIRCH 算法的优化改进，输入 *n* 个聚类对象的数据集合 $D=\{x_1,x_2,\cdots,x_n\}$，结果聚类簇的个数为 *k*，输出 *k* 个聚类簇。

步骤如下：

a) 初定参数 μ 和 σ，经公式

$$x_i=\frac{u_i}{\sigma_i}\left[\left(\frac{x_i\sigma_i}{\mu_i}\right)+\frac{1}{2}\right],i=1,2,\cdots,d \tag{13}$$

转换成汇聚点，将这个汇聚点和初始点的进行个数的比较，对参数 μ 和 σ 进行数值的调整，让其比

值介于 0.01~1 中间，得到汇聚后的点；

b) 扫描汇聚后点，用聚类算法进行处理，结果得出 k 个簇；

c) 进行操作聚类簇的全排列；

d) 参考 k 值；如果 k 值小于 5，就用 BIRCH 算法对所有排列进行操作；如果大于 5，就用 BIRCH 算法对随机抽取十个进行操作。

3.4. 聚类算法并行编程模型

对于数据的并行模式，当环境相同时，其逻辑代码也是一样的，只有计算使用的数据不相同。主要是为了利用数据间不具联系数据的共同运算，数据先运行，再分类，并且多个程序并行执行，提高运行效率。

3.5. BIRCH 聚类算法并行化

通过 Spark 工具来对 BIRCH 算法进行并行化应用。解决海量数据的井喷，单台计算机无法满足数据整理分析归类的要求，因此通过分布式集群来解决这一问题。Spark 工具平台对 BIRCH 算法的运行流程图见图 2 所示。

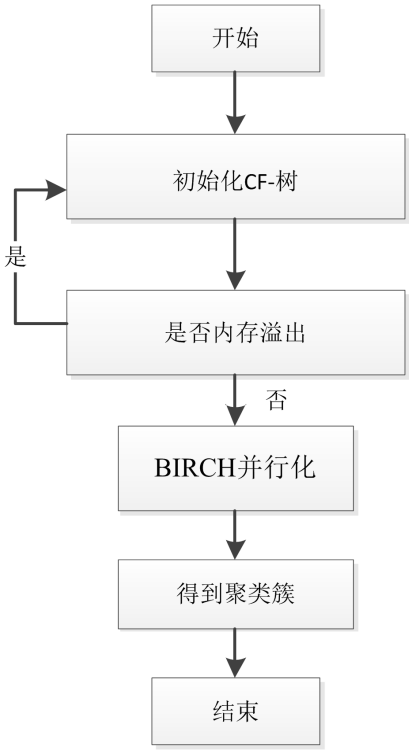

Figure 2. Operation flow chart of BIRCH algorithm
图 2. BIRCH 算法的运行流程图

4. 基于 Spark 的大数据聚类应用

通过在物理机上搭建 Spark 分布式集群来进行试验。首先将系统网络根据试验的需求来配置防火墙端口，然后对 Hadoop 集群进行搭建，根据需求选好相应的组件，再对物理机上搭建 Spark 集群，最后是安装部署 GeoMesa。

4.1. 实验数据分析

本文将改进的算法应用于实际的大数据分析中，分析的数据是来自在高校实验室管理中应用采集数据。大数据对高校实验室的管理，有利于将安全管理进行职能化，将各实验室的门锁用门禁系统所替代，实行刷卡进出，对刷卡数据进行采集，为提前做好实验室安全事故人员疏散作保障；并通过视频监控系统对进出实验室人员进行监督信息采集，利用大数据的分析提高仪器、设备使用率，为仪器、设备运转提供所需的保障；通过数据现象和统计分析，协助科技人员对技术的开发和升级，促进大数据实验室管理升级，提高高校实验室的实验教学质量。

4.1.1. 数据结构分析和提取

对门禁系统经过数据筛选，留下进出人员用户的身份 IP，门禁 IP 和数据上传时间，具体如表 1 所示。

Table 1. Data structure table
表 1. 数据结构表

名称	类型	数据信息
id_num	String	身份 IP
create_time	Date	上传时间
lng	Double	门禁 IP

将该实验楼的所有的门禁系统 IP 也输入系统，并提前清除该区域内需要特殊权限开启门禁的数据，减小计算量，提高分析效率。

4.1.2. 数据结构分析方法

数据结构分析方法有两种，其一就是对全体数据进行聚类操作，其二就是对独特的簇进行具体的关系分析。

a) 聚类数据

对所有的数据进行聚类分析，筛选出聚类簇中数量较多的簇，通过排序得到数量较多的簇，存储簇的中心点，再根据中心点在实验室所在的教学楼上呈现具体的物理 IP 位置。

b) 聚类簇内部分析

聚类数据分析后，就会将这些数据存储在 GeoMesa 数据库里，利用特殊的时间和事件来分析数据。

4.2. 实验结果

通过上面的聚类算法数据统计，排出前面几位实验室人次数比较多，相对集中的的区域，通过得知该区域实验室人员比较密集，并为学校建议该区域实验室作为防火防灾的重点区域，进行保护防范，同时对实验室里的仪器设备维护管理作为重点关注对象，进行有效管理。根据数据的分析，提前做好预测分析，为高校实验室进行规划管理提供参考，做到防患于未然，有效提高仪器设备使用效率，并做好仪器设备的维护保养工作。

5. 结束语

经过实验验证，对聚类算法的优化，在性能上有所提高。为高校实验室人员的区域密集度分析做出了分类统计功能，取得了一定的成果，分析出了特殊的区域，特定的时间会在什么区域出现人口密集度较高的情况等。

在改进 BIRCH 算法上，原来不能直接并行化运行在 Spark 分布式工具平台上，现在不仅可以直接运行，还简化了数据量，减少了运行时间，提高了效率。在后续的工作中将继续对算法进行优化，提高效率，通过实验，一步步提高精确度。

基金项目

教育部科技发展中心高校产学研创新基金——新一代信息技术创新项目(2018A02027)；教育部科技发展中心高校产学研创新基金——新一代信息技术创新项目(2018A01015)；共青团广东省委员 2019 年广东大学生科技创新培育专项资金项目《知识产权大数据分析与服务系统设计》(pdjh2019b0775)。

参考文献

[1] Manyika, J., Chui, M., Brown, B., *et al*. (2011) Big Data: The Next Frontier for Innovation, Competition and Productivity. *Analytics*, 3-17.

[2] Lohr, S. (2012) The Age of Big Data. *International Journal of Communications, Network and System Sciences*, **16**, 10-15.

[3] Yu, Q.L. (2015) Learning Analytics: The Next Frontier for Computer Assisted Language Learning in Big Data Age. 2015 *IEEE* 31*st International Conference on Data Engineering* (*ICDE*), Seoul, Korea, 13-16 April 2015, 1-8. https://doi.org/10.1051/shsconf/20151702013

[4] Khan, M., Jin, Y., Li, M., *et al.* (2016) Hadoop Performance Modeling for Job Estimation and Resource Provisioning. *IEEE Transactions on Parallel & Distributed Systems*, **27**, 441-454. https://doi.org/10.1109/TPDS.2015.2405552

[5] Guo, Y., Rao, J., Cheng, D., *et al*. (2017) iShuffle: Improving Hadoop Performance with Shuffle-on-Write. *IEEE Transactions on Parallel & Distributed Systems*, **28**, 11-20. https://doi.org/10.1109/TPDS.2016.2587645

[6] Li, Z., Yang, C., Liu, K., e*t al*. (2016) Automatic Scaling Hadoop in the Cloud for Efficient Process of Big Geospatial Data. *International Journal of Geo-Information*, **5**, 173. https://doi.org/10.3390/ijgi5100173

[7] 李璐明, 蒋新华, 廖律超. 基于弹性分布数据集的海量空间数据密度聚类[J]. 湖南大学学报(自科版), 2015(8): 116-124.

[8] 宋杰, 郭朝鹏, 张一川, 等. 增量式迭代计算模型研究与实现[J]. 计算机学报, 2016(1): 109-125.

[9] 侯丽利, 董书宝. 基于 NoSQL 数据库的大数据查询技术的研究与应用[J]. 无线互联科技, 2015(1): 147-154.

[10] 穆罕默德扎基(Mohammed J. Zaki),小瓦格纳梅拉. 数据挖掘与分析概念与算法[M]. 北京: 人民邮电出版社, 2017: 155-167.

[11] 牛新征, 佘堃. 面向大规模数据的快速并行聚类划分算法研究[J]. 计算机科学, 2012, 39(1): 134-137.

[12] 金相郁. 中国区域划分的层次聚类分析[J]. 城市规划学刊, 2004(2): 23-28.

[13] 闫安, 刘琪林. 一种基于参考点的快速密度聚类算法[J]. 微电子学与计算机, 2017, 34(10): 32-35.

[14] 赵慧, 刘希玉, 崔海青. 网格聚类算法[J]. 计算机技术与发展, 2010, 20(9): 83-85.

[15] 张忠林, 曹志宇, 李元韬. 基于加权欧式距离的 k-means 算法研究[J]. 郑州大学学报(工学版), 2010, 31(1): 89-92.

[16] Hartigan, J.A. (1979) A K-Means Clustering Algorithm. *Applied Statistics*, **2**, 100-108. https://doi.org/10.2307/2346830

[17] 张蓉, 钟艳. 基于 BIRCH 算法的模糊集数据库挖掘算法[J]. 科技通报, 2014(4): 47-49.

[18] 宋雨, 焦谱, 李刚. 大数据预处理中属性约简的特性保持分析[J]. 计算机测量与控制, 2015, 23(12): 4191-4194.