云环境下基于Canopy聚类的FCM算法研究

余长俊 张 燃
(武汉理工大学计算机学院 武汉 430063)

摘 要 FCM算法是目前广泛使用的算法之一。针对FCM聚类质量和收敛速度依赖于初始聚类中心的问题,结合Canopy聚类算法能够粗略快速地对数据集进行聚类的优点,提出了一种基于Canopy聚类的FCM算法。该算法通过将Canopy算法快速获取到的聚类中心作为FCM算法的输入来加快FCM算法收敛速度。并在云环境下设计了其MapReduce化方案,实验结果表明,MapReduce化的基于Canopy聚类的FCM算法比MapReduce化的FCM聚类算法具有更好的聚类质量和运行速度。

关键词 FCM算法(模糊均值聚类算法),聚类,MapReduce,云环境
中图法分类号 TP305 **文献标识码** A

Research of FCM Algorithm Based on Canopy Clustering Algorithm under Cloud Environment

YU Chang-jun ZHANG Ran
(Department of Computer Science, Wuhan University of Technology, Wuhan 430063, China)

Abstract FCM algorithm is one of the widely used algorithms, but the quality and convergence speed of it depend on the quality of the initial cluster centers. Because Canopy algorithm can quickly cluster the data set and get the cluster centers, we proposed the FCM algorithm combining with Canopy cluster algorithm. The algorithm accelerates the convergence rate by making the clustering center obtained by canopy algorithm as the input of FCM. Then we designed its MapReduce scheme in a cloud environment. Experimental results show that the MapReduce of FCM clustering algorithm based on Canopy clustering algorithm has better clustering quality and speed than MapReduce of FCM clustering algorithm.

Keywords FCM algorithm(Fuzzy C Means algorithm), Clustering, MapReduce, Cloud environment

传统的对FCM算法(模糊均值聚类算法)的研究旨在在串行化的条件下提高FCM算法的聚类效果,以及提高FCM算法的运行速度。随着数据量的增加,计算机的性能就逐渐能成为FCM算法的瓶颈。

目前,国内外研究者提出了许多对FCM算法的改进方法,张建强等[1]针对模糊均值聚类算法在多核平台的性能问题,利用高性能工具Intel Parallel Amplifier提出了并行化设计方案。虞倩倩、戴月明[2]针对FCM聚类算法随着数据量的增大其时间复杂度会变得较高的缺点,提出了使用MapReduce编程框架对FCM聚类算法进行并行化的方法,其MapReduce化过程包括Map和Reduce两个过程。Esteves R. M等[3]使用开源的云计算平台Apache Mahout/Hadoop技术以及以维基百科中的最新文章作为数据源对K-means和FCM算法进行了对比实验。以上方法都是对FCM算法本身进行研究,并没有结合其它并行化方法对FCM算法进行改进来提高其运行效率。本文利用云计算对FCM算法进行进一步研究,将其与更多的算法进行结合,以提高MapReduce化的FCM算法的执行速度,同时提高聚类效率与效果,将具有重要的理论研究价值和实际应用意义。

1 Canopy-FCM算法思想及流程

Canopy-FCM算法的主要思想是使用Canopy算法产生聚类中心,然后利用该聚类中心作为FCM聚类算法的初始聚类中心进行聚类。

根据Canopy-FCM算法的思想得出,Canopy-FCM算法的基本流程分为两个阶段:第一个阶段是使用Canopy算法将数据集划分为若干个canopy中心,并将canopy中心集合中小于一定阈值的canopy中心删除,以起到剔除孤立点的作用。第二个阶段是根据第一阶段产生的初始聚类中心,使用FCM聚类算法进行聚类。因此需要首先介绍Canopy算法及Canopy算法。

1.1 Canopy算法

Canopy算法[4,5]的思想是使用一种代价低的相似性度量方法,快速粗略地将数据划分成若干个重叠子集,每个子集可以看成是一个簇。Canopy算法基本流程[6,7]如下:①设置阈值$T1$、$T2$,其中$T1>T2$;②从数据集中取得一个数据对象,构建初始canopy,并从该数据集中删除该数据对象;③从剩余的数据集中取出一个数据对象,计算其与所有的canopy中心之间的距离,如果该数据对象与某个canopy中心的距离在$T1$内,则将该对象加入到这个canopy中,如果该对象与canopy中的某个canopy中心的距离小于$T2$,则当该数据对象与所有canopy中心的距离计算完成后,将其从数据集中删除,如果该数据对象没有加入到任何canopy,则构建一个新的canopy;④重复步骤3,直到数据集为空。其中$T1$、$T2$的值可

本文受教育部:网络时代的科技论文快速共享研究(20111140004)资助。
余长俊(1989—),男,硕士生;张 燃(1990—),男,硕士生。

以由交叉检验来获取[3]。

1.2 FCM 算法

FCM 算法的具体流程如下[10,11]：① 设置聚类中心个数 c，随机获取 c 个聚类中心；②对所有的数据对象计算隶属度矩阵；③计算初始聚类中心；④判断聚类中心是否收敛，若收敛则结束，否则，返回步骤②。

2 串行化的 Canopy-FCM 算法思想

根据 Canopy-FCM 算法的串行思想以及 MapReduce 的编程模型，在 Hadoop 平台下对该算法实现 MapReduce 化。Canopy-FCM 算法的并行化分为两个阶段：Canopy 算法的 MapReduce 化及 FCM 算法的 MapReduce 化。Canopy 算法的作用是为 FCM 算法提供初始聚类中心。其具体框架如图 1 所示。

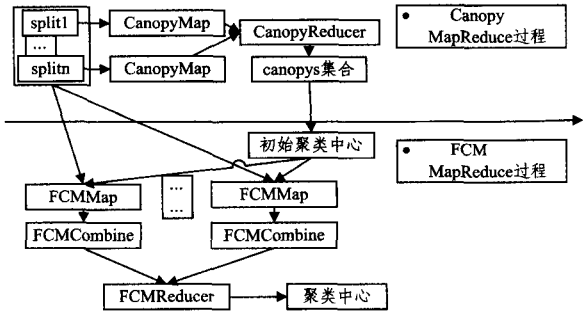

图 1 Canopy-FCM 算法的 MapReduce 框架图

2.1 Canopy 算法的 MapReduce 化

Canopy 算法的并行化的设计分为两个过程，即 Map 过程和 Reduce 过程。Map 过程的主要任务是对本节点中的数据对象按照串行化的 Canopy 算法思想进行处理，对各 Map 结点需要输入阈值 $T1$ 和 $T2$，形成 canopy 中心集合，由于 Map 过程所产生的 canopy 中心集合是局部的，其产生 canopy 中心集合的数量远小于该 Map 结点上的数据对象的数量，因此所有 Map 结点上 canopy 中心的数目也一定远小于原数据集合的数量[8]，需要使 Reduce 过程的阈值（$T3$，$T4$）较 Map 过程的（$T1$，$T2$）稍大。为了减少 Reduce 过程的次数，只需要一个 Reduce 结点对从所有 Map 结点上获取的 canopy 中心的集合进行合并。

另外，可以设置一个阈值 filter，该阈值的作用是删除包含数据对象数目少于 filter 的 canopy 中心。由于 Map 过程各 Map 结点处理的数据对象只是原数据对象的一部分，存在局限性，因此在该阶段不需要对 filter 的值进行设置，而在 Reduce 过程中是对全局 canopy 中心的集合进行合并，因此可以设置一个较为合理的 filter 的值，将包含数据对象小于 filter 的 canopy 中心删除，可以起到去除异常点的作用。对 Canopy 算法的 Map 和 Reduce 过程的具体的流程与其串行化的流程一致。

2.2 FCM 算法的 MapReduce 化

由图 1 中可以看出 FCM 的 MapReduce 化分为 FCMMap（Map 过程）、FCMCombine（Combine 过程）、FCMReduce（Reduce 过程），其中 Map 过程的任务是计算出 Map 结点内数据对象对初始聚类中心的隶属度，Combine 过程是根据 Map 过程的计算结果计算出 Map 结点上与 center 对应的所有隶属度 m 次方与隶属度对应数据对象乘积的和（Sum0）以及与 center 对应的所有隶属度 m 次方的和（Sum1），Reduce 过程的任务是根据 Combine 过程得到所有 Map 结点的 Sum0、Sum1，Sum0/Sum1 即为新的聚类中心。由于 FCM 聚类算法需要经过多次迭代才能够取得较好的聚类中心，因此需要对上述的 Map、Combine、Reduce 过程进行迭代。经过迭代过程得到了最终的聚类中心，但对数据集合中的数据对象并没有划分其归属的类，因此需要一个新的 MapReduce 过程对其进行划分。FCM 算法的 MapReduce 化分为 5 个过程，即 Map、Combine、Reduce、迭代过程及数据对象分类过程。

2.2.1 Map 过程设计

Map 过程的主要任务是计算出本结点中的数据对象对聚类中心的隶属度。Map 过程的输入是本结点上的数据对象和初始聚类中心（或上一次迭代过程更新的聚类中心），其输出是〈key，value〉键值对的形式，其中 key 是聚类中心的索引号，value 是 key 对应的聚类中心对 Map 结点内所有数据对象隶属度以及数据对象，〈key，value〉的数据结构可以表示为（center，（point，weight））。

2.2.2 Combine 过程设计

由于 Map 过程中产生的数据都是保存在本地磁盘，因此，为了减少数据在各节点之间的通信开销以及 Reduce 过程的计算量，在 Map 过程后使用了 Combine 过程。它在 Map 结点上进行，其作用是对本 Map 结点上的数据进行合并。

Combine 过程的主要任务是对每一个聚类中心计算出与之对应的所有隶属度 m 次方与隶属度对应数据对象乘积的和（以 Sum0 表示）以及所有隶属度 m（模糊因子）次方的和（以 Sum1 表示）。Combine 过程的输出是〈key，value〉的形式，其中 key 表示聚类中心的索引，value 的数据结构是（Sum0，Sum1）。

Map 过程输出的是数据中心的索引、数据对象以及隶属度组成的数据集合，该数据集合作为 Combine 过程的输入，其输入的形式是将 Map 阶段生成数据集合中具有相同 key 值的 value 值组成一个数据集合，以表示为〈key，values〉，因此一次 Combine 过程实质是对该〈key，values〉进行操作，每一个 key 值对应一个聚类中心的索引，因此每执行完一个 Combine 过程就可以得出一个 Sum0 和一个 Sum1。

2.2.3 Reduce 过程设计

由于在 Map 和 Combine 的过程中都只是针对本 Map 结点上的数据进行计算，因此设计了 Reduce 过程，Reduce 过程的主要任务是对所有 Map 结点上的数据进行合并，并计算出聚类中心。

经过 Combine 过程后，每一个 Map 结点的输出已经非常少了，其输出的数据集的形式为｛center，Sum0，Sum1｝，其中 center 是聚类中心的索引，Sum0 表示该 Map 结点上与 center 对应的所有隶属度 m 次方与隶属度对应数据对象乘积的和，Sum1 表示该 Map 结点上与 center 对应的所有隶属度 m 次方的和。

Reduce 过程在对 Map 过程进行收集的时候，将具有相同 key 值的所有 value 值组成一个集合，因此一次 Reduce 过

程实际上是对具有相同 center 值的{Sum0,Sum1}组成的集合进行操作,操作的步骤为:① 将{Sum0,Sum1}组成的集合分别对 Sum0 和 Sum1 求和;② Sum0/Sum1 即为索引号为 center 的聚类中心;③ 将聚类中心输出。当执行完所有的 reduce 过程后就得到了所有的聚类中心,该过程执行的次数等于聚类中心的个数。

由于 Reduce 过程执行的次数等于聚类中心的个数,并且一次 Reduce 过程能够获取一个聚类中心,该过程对其它的 Reduce 结果没有影响,因此可以设置多个 Reduce 结点执行 Reduce 过程。

2.2.4 迭代过程设计

FCM 聚类算法的一个重要步骤是判断生成的聚类中心是否收敛,因此 FCM 聚类算法的实现过程需要对上述的 Map、Combine、Reduce 过程进行多次迭代,直到聚类中心收敛。

该过程对聚类是否收敛的判断是通过比较上一次获取的聚类中心(或初始聚类中心)与本次获取的聚类中心进行比较,若所有聚类中心的变化都小于指定的阈值,则聚类中心收敛,算法停止;反之,使用本次获取的聚类中心替代上一次获取的聚类中心,并开始新一轮的 MapReduce 过程。为了避免算法执行的迭代次数过多而消耗大量的时间,需要为算法设置最大的迭代次数,当 MapReduce 过程的迭代次数大于该最大迭代次数的时候,算法停止。

2.2.5 数据对象分类设计

经过上述过程仅仅获取了最终的聚类中心,但是并没有将所有的数据对象划分到其所属的类,因此,要解决这个问题需要再执行一个 MapReduce 过程,该过程只需要 Map 过程就可以完成。

一般将数据对象归属于最大隶属度聚类中心对应的类。由于计算数据对象对聚类中心的隶属度是一个独立的过程,因此能够设计其 MapReduce 化的过程,该过程只包括 Map 阶段,流程如下:①获取最终的聚类中心;②对 Map 结点上的数据对象计算其与所有聚类中心的隶属度;③获取隶属度最大值对应的聚类中心的索引,则该索引即为该数据对象所对应的类;④执行步骤 2,直到对 Map 结点上所有的数据对象处理完毕。

3 实验结果的分析

3.1 实验环境

硬件环境:PC 数量 5 台,CPU 为 2.2GHz 以上,硬盘 160G 以上,内存 2G 以上。

软件环境:Eclipse,Ubuntu12.10,HBase 数据库,Hadoop 分布式处理环境。

3.2 FCM 聚类质量的分析

使用查准率和查全率[9,10]对聚类结果的质量进行评测。

使用数据堂获取的动物属性分类数据的 3 种动物类别的特征进行聚类算法的研究,每个类包含 1000 个特征向量,在 Hadoop 集群上,分别对 FCM 算法、Canopy-FCM 进行了测试。先给出各算法的具体参数:FCM 算法需要给出聚类数目,这里使用的数据集有 3 类,因此 $k=3$;Canopy-FCM 算法的阈值 $T1=0.15$,$T2=0.08$,$T3=1.5T1$,$T4=2T1$,$filter=10$。

为了能体现 Canopy-FCM 算法与传统的 FCM 算法在并行化的条件下的优越性,使用相同 Hadoop 集群、相同数据集进行聚类,聚类的结果如表 1 所列。

表 1 算法的聚类质量以及执行效率

算法	平均查准率	平均查全率	迭代次数	执行时间(s)
FCM	0.69	0.59	12	98
Canopy-FCM	0.74	0.68	9	76

由表 1 可以看出 Canopy-FCM 算法聚类效果较 FCM 算法好,且其具有较好的聚类速度。Canopy 算法能够快速地获取到较好的初始聚类中心,加快收敛速度。

3.3 FCM 聚类加速比分析

选取数据大小为 2G 的数据进行实验和分析。分别在节点数为 1、2、3、5 的 Hadoop 云计算平台上使用 Canopy-FCM 算法和 FCM 算法对应的加速比,如图 2 所示。

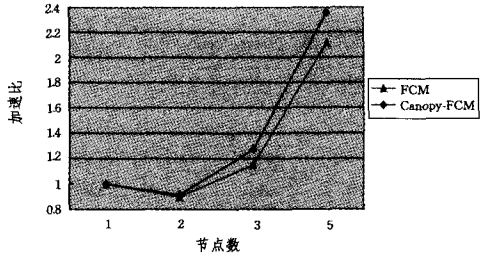

图 2 Hadoop 下不同节点数目的加速比

由图 2 可以看出,当 Hadoop 云平台的节点数相同的时候,Canopy-FCM 算法的执行效率较 FCM 算法高。节点数为 2 的时候两个算法的执行时间都高于单机的执行时间,这是因为在算法执行的过程中节点中的数据通信消耗了更多的时间。而随着节点数目的增加,算法执行的时间都逐渐缩小,说明数据节点间的通信时间比例减小,算法运行速度呈逐步上升趋势。

结束语 随着大数据时代的来临、数据量的急速增加,需要对云平台下 MapReduce 化 FCM 算法进行进一步的优化,以减少计算机集群中通信量,提高算法的查准率和查全率以及执行效率。本文结合 MapReduce 化的 Canopy 算法,对 FCM 算法进行了研究。实验结果表明,通过 Canopy 算法获取的聚类中心能够提高 FCM 算法的效率和聚类效果 。但是,Canopy 算法本身需要参数的输入,对 Canopy 参数有效性的确定在未来需要进一步探索。

参 考 文 献

[1] 张建强,郑晓薇,吴华平. 模糊 C 均值聚类算法的并行化研究[J]. 微型机与应用,2010,29(23):8-18

[2] 虞倩,戴月明. 基于 MapReduce 的并行模糊 C 均值算法[J]. 计算机工程与应用,2013,49(14):133-137

[3] 高新波,裴继红,谢维信. 模糊 C 均值聚类算法中加权指数 m 的研究[J]. 电子学报,2000,4:80-83

[4] 孙吉贵,刘杰,赵连宇. 聚类算法研究[J]. 软件学报,2008(1):48-61

[5] Esteves R M,Rong C. Using Mahout for clustering Wikipedia's latest articles: A comparison between k-means and fuzzy c-means in the cloud [C]// Proceedings of the 2011 Third IEEE International Conference on Cloud Computing Technology and Science. Washington,DC: IEEE Computer Society,2011:565-569

[6] McCallum A, Nigam K, Ungar L H. Efficient clustering of high-dimensional data sets with application to reference matching[C]// Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, 2000: 169-178

[7] 李应安. 基于 MapReduce 的聚类算法的并行化研究[D]. 广州: 中山大学, 2010

[8] Ruspini E H. Numerical methods for fuzzy clustering[J]. Information Sciences, 1970, 2(3): 319-350

[9] 赵洪昌. 云环境下的关联分析和模糊聚类研究[D]. 南京: 南京信息工程大学, 2013

[10] 陈爱平. 基于 Hadoop 的聚类算法并行化分析及应用研究[D]. 成都: 电子科技大学, 2012

[11] Ohmann T, Rahal I. Efficient clustering-based source code plagiarism detection using PIY[J]. Knowledge and Information Systems, 2014, 3: 1-28

[12] 余丹. 关于查全率和查准率的新认识[J]. 西南民族大学学报, 2009(2): 283-285

(上接第 306 页)

[3] Femminella M, Francescangeli R, Reali G, et al. An enabling platform for autonomic management of the future internet[J]. Network, IEEE, 2011, 25(6): 24-32

[4] Arthur C, Carlos K, Stênio F, et al. A Survey on Internet Traffic Identification and Classification [J]. Communications Surveys and Tutorials, IEEE, 2009, 11(3): 37-52

[5] Moore A, Zuev D, Crogan M. Discriminators for use in flow-based classification[M]. Queen Mary and Westfield College, Department of Computer Science, 2005

[6] Szabó G, Szabó I, Orincsay D. Accurate traffic classification [C]// IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks, 2007 (WoWMoM 2007). IEEE, 2007: 1-8

[7] Kumar S, Dharmapurikar S, Yu F, et al. Algorithms to accelerate multiple regular expressions matching for deep packet inspection [J]. ACM SIGCOMM Computer Communication Review, 2006, 36(4): 339-350

[8] Smith R, Estan C, Jha S, et al. Deflating the big bang: fast and scalable deep packet inspection with extended finite automata [J]. ACM SIGCOMM Computer Communication Review, 2008, 38(4): 207-218

[9] Haffner P, Sen S, Spatscheck O, et al. ACAS: automated construction of application signatures[C]//Proceedings of the 2005 ACM SIGCOMM workshop on Mining network data. ACM, 2005: 197-202

[10] Moore A W, Zuev D. Internet traffic classification using bayesian analysis techniques[J]. ACM SIGMETRICS Performance Evaluation Review, 2005, 33(1): 50-60

[11] Williams N, Zander S, Armitage G. A preliminary performance comparison of five machine learning algorithms for practical IP traffic flow classification[J]. ACM SIGCOMM Computer Communication Review, 2006, 36(5): 5-16

[12] Bernaille L, Teixeira R, Akodkenou I, et al. Traffic classification on the fly[J]. ACM SIGCOMM Computer Communication Review, 2006, 36(2): 23-26

[13] Erman J, Arlitt M, Mahanti A. Traffic classification using clustering algorithms [C] // Proceedings of the 2006 SIGCOMM workshop on Mining network data. ACM, 2006: 281-286

[14] Park B C, Won Y J, Kim M S, et al. Towards automated application signature generation for traffic identification [C] // IEEE Network Operations and Management Symposium, 2008(NOMS 2008). IEEE, 2008: 160-167

[15] Ye M, Xu K, Wu J, et al. Autosig-automatically generating signatures for applications[C] // Ninth IEEE International Conference on Computer and Information Technology, 2009 (CIT' 09). IEEE, 2009, 2: 104-109

[16] Szabó G, Turányi Z, Toka L, et al. Automatic protocol signature generation framework for deep packet inspection[C] // Proceedings of the 5th International ICST Conference on Performance Evaluation Methodologies and Tools. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2011: 291-299

[17] Karagiannis T, Papagiannaki K, Faloutsos M. BLINC: multilevel traffic classification in the dark[J]. ACM SIGCOMM Computer Communication Review, 2005, 35(4): 229-240

[18] Bujlow T, Riaz T, Pedersen J M. A method for classification of network traffic based on C5. 0 Machine Learning Algorithm[C]// 2012 International Conference on Computing, Networking and Communications(ICNC). IEEE, 2012: 237-241

[19] Parsons L, Haque E, Liu H. Subspace clustering for high dimensional data: a review[J]. ACM SIGKDD Explorations Newsletter, 2004, 6(1): 90-105

[20] Müller E, Günnemann S, Assent I, et al. Evaluating clustering in subspace projections of high dimensional data[J]. Proceedings of the VLDB Endowment, 2009, 2(1): 1270-1281

[21] Agrawal R, Gehrke J E, Gunopulos D, et al. Automatic subspace clustering of high dimensional data for data mining applications: U. S. Patent 6, 003, 029[P]. 1999-12-14

[22] Cheng C H, Fu A W, Zhang Y. Entropy-based subspace clustering for mining numerical data[C] // Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 1999: 84-93

[23] Goil S, Nagesh H, Choudhary A. MAFIA: Efficient and scalable subspace clustering for very large data sets[C]//Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 1999: 443-452

[24] Xie G, Iliofotou M, Keralapura R, et al. SubFlow: towards practical flow-level traffic classification[C] // INFOCOM, 2012 Proceedings IEEE. IEEE, 2012: 2541-2545