基于 MapReduce 框架下 K-means 的改进算法*

阴爱英[1], 吴运兵[2], 朱敏琛[1,2], 张 莹[1,2]
(1. 福州大学至诚学院 计算机工程系, 福州 350002;2. 福州大学 数学与计算机科学学院, 福州 350116)

摘 要: 针对海量数据背景下 K-means 聚类结果不稳定和收敛速度较慢的问题,提出了基于 MapReduce 框架下的 K-means 改进算法。首先,为了能获得 K-means 聚类的初始簇数,利用凝聚层次聚类法对数据集进行聚类,并用轮廓系数对聚类结果进行初步评价,将获得数据集的簇数作为 K-means 算法的初始簇中心进行聚类;其次,为了能适应于海量数据的聚类挖掘,将改进的 K-means 算法部署在 MapReduce 框架上进行运算。实验结果表明,在单机性能上,该方法具有较高的准确率和召回率,同时也具有较强的聚类稳定性;在集群性能上,也具有较好的加速比和运行速度。

关键词: MapReduce 框架; K-means 算法; 数据挖掘; 聚类分析
中图分类号: TP391 **文献标志码**: A **文章编号**: 1001-3695(2018)08-2295-04
doi:10.3969/j.issn.1001-3695.2018.08.014

Improved K-means algorithm based on MapReduce framework

Yin Aiying[1], Wu Yunbing[2], Zhu Minchen[1,2], Zhang Ying[1,2]
(1. *Dept. of Computer Engineering, Zhicheng College of Fuzhou University, Fuzhou* 350002, *China*; 2. *College of Mathematics & Computer Science, Fuzhou University, Fuzhou* 350116, *China*)

Abstract: Focusing on the unstable result and slow convergence of K-means clustering algorithm for huge amount of data, this paper proposed an improved K-means algorithm based on MapReduce framework. Firstly, in order to obtain the initial cluster number of K-means clustering, it used hierarchical clustering method to cluster the dataset, and evaluated the clustering result by silhouette coefficient. It clustered the cluster number of the acquired data set as the initial cluster center of the K-means algorithm. Secondly, in order to adapt to the clustering mining of massive data, it used the modified K-means algorithm to deploy in the MapReduce framework. The experimental results show that the proposed method has high precision and recall rate and strong clustering stability in single machine performance, and also has better speedup ratio and running speed in clustering performance.

Key words: MapReduce framework; K-means algorithm; data mining; clustering analysis

随着大数据时代的来临,如何从海量数据中快速而准确地挖掘出有用的信息,已成为学术界和工业界普遍关注的问题。在传统的数据挖掘算法中,聚类分析得到广泛应用,而 K-means 算法作为一种基于划分的聚类方法[1],由于其简单高效,成为了运用比较广泛的一种聚类算法。然而,传统的 K-means 算法在面对海量数据时,由于其算法聚类结果不稳定,同时针对大数据算法运行效率较低,所以难以适应于海量数据的挖掘。近年来,随着并行计算模型的提出,为解决大数据挖掘问题提供了新的思路,而 Google 提出的 MapReduce 模型[2]在处理海量数据方面具有巨大的优势,成为新的研究方向[3]。

在面对海量数据时,传统的 K-means 算法易产生内存溢出的现象,易出现局部最优、收敛速度过慢的问题,同时由于对 k 值随机选取,导致聚类结果不稳定的状况。针对这些问题,国内外学者进行了大量研究与改进。Cui 等人[1]提出利用抽样方法获得稳定的初始簇数,并在 MapReduce 框架下解决大数据的挖掘问题;Lin 等人[2]基于密度聚类来改善 K-means 初始聚簇数的选择,并通过 MapReduce 框架来降低运行时间;Debatty 等人[3]为了能自动选择 K-means 算法的初始 k 值,提出了 G-means 算法;Yuan 等人[4]基于密度聚类方法来获得初始 k 的值,并通过平均密度来排除噪声数据;Kettani 等人[5]提出了一种自动确定簇数 k 及初始簇中心等参数的 AK-means 算法;Ma 等人[6]提出了一种基于 MapReduce 框架和网格计算的 K-means 改进算法;王永贵等人[7]采用多次随机抽样,通过计算密度、距离与平方误差等方法,提出 MapReduce 框架的 K-means 算法改进;李兰英等人[8]利用模糊聚类的思想对 K-means 算法进行改进,并将改进算法在 MapReduce 框架下运行。

在大数据环境下的聚类挖掘,基于 MapReduce 框架下的 K-means 算法能有效解决内存不足的问题,但是其聚类结果仍存在着不稳定的问题。因此,本文提出基于 MapReduce 框架下的 K-means 改进算法。首先,对数据集采用凝聚层次聚类方法获得初始聚类簇,并对初始聚类簇采用轮廓系数对其进行评价,得到较为合理的簇数,避免了传统 K-means 聚类由于初始簇中心选择的随机性,而导致聚类结果不稳定的现象;其次,为了避免传统 K-means 算法在海量数据下收敛速度慢而导致内存不足的现象,设计了基于 MapReduce 框架下并行 K-means 算法,以此提高 K-means 算法在海量数据挖掘时的运行效率。实验结果表明,本文提出的算法不仅在聚类性能上优于传统的 K-means 聚类,而且在集群性能方面也有良好的效果。

1 相关知识

1.1 MapReduce 框架

MapReduce 是 Google 提出的一种并行编程框架,用于对海量数据集的并行分析与运算,能够实现程序的自动并行处理,

收稿日期: 2017-04-12; **修回日期**: 2017-05-22 **基金项目**: 福建省自然科学基金资助项目(2017J01755);福建省教育厅中青年教师教育科研项目(JAT160658,JAT160077);福建省科技计划项目(2016R0095)

作者简介: 阴爱英(1976-),女,山西芮城人,讲师,硕士,主要研究方向为数据挖掘和机器学习(ireneying@126.com);吴运兵(1976-),男,副教授,硕士,主要研究方向为数据挖掘、知识表示和知识学习;朱敏琛(1961-),女,教授,硕导,硕士,主要研究方向为智能技术、模式识别和图像处理;张莹(1964-),女,副教授,硕导,硕士,主要研究方向为数据挖掘、智能技术.

并提供数据分割、任务调度等细节，实现了程序高度并行性和可扩展性[9]。MapReduce 编程框架的核心是 map 和 reduce 两个函数。一个作业(job)通常会把输入的数据集切分为若干个独立的数据块，由 map 函数完成并行的方式处理它们，其结果作为 reduce 函数的输入，并最终生成结果[10]。MapReduce 框架的具体步骤如下：首先通过对数据源进行 split 分割成若干个数据块，将这些数据块读入到 map 函数进行处理，生成了键值组〈key，value〉的中间文件，然后将 map 函数的输出作为 reduce 函数的输入，对相同 key 的 value 进行计算等操作，将运算结果合并，最后输出相应的结果。具体 MapReduce 框架的流程模型如图 1 所示[11]。

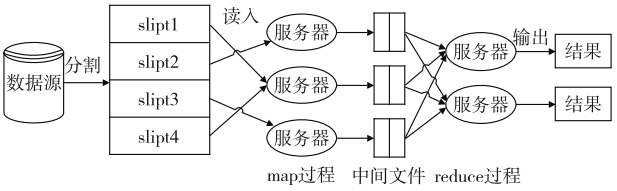

图1 MapReduce框架图

1.2 K-means算法

K-means 算法是最常用的基于划分的聚类算法，具有实现简单、计算复杂度不高、执行速度快等优点。K-means 聚类算法是针对给定 k 值，将数据样本分成 k 个簇类，在同一簇内的数据相似度较高，而不同簇的数据相似度较低。具体算法步骤如下[12]：

a)对于待分类的数据集中，随机地选择 k 个对象，每个对象代表一个簇的初始均值或中心。

b)对剩余的每个对象，根据它与簇均值的距离，将其指派到最相似的簇。

c)计算每个簇的新均值，重新修正聚类中心。

d)计算偏差并进行判断。如果聚类的误差平方收敛，则返回最佳的聚类中心，算法终止；否则，转到步骤 b)重新寻找。

2 基于 MapReduce 的 K-means算法改进

传统 K-means 聚类算法通过随机选取 k 个簇，导致聚类结果不稳定，本文提出了 K-means 算法的改进。首先通过凝聚层次聚类获取简单的分类，然后利用轮廓系数对其进行评价，最终得出聚类结果作为 K-means 算法初始 k 值的输入，避免了 K-means 算法随机选择 k 而导致结果不稳定的问题。同时，由于传统 K-means 聚类算法在大数据背景下，其运行速度和效率受到一定的制约，所以，本文采用了并行计算的方式，将改进后的 K-means 部署在 MapReduce 框架下运行，提高了运行效率。

2.1 K-means算法

层次聚类算法是对给定的数据集合按层次结构进行分解，形成一棵以簇为节点的树[12]。根据分解形式可以分为凝聚和分裂。凝聚层次聚类是采用了自底向上的策略，由于其聚类方式简单，因此成为广泛应用的聚类方法。凝聚层次聚类算法是先让每个对象自成一簇，然后将这些簇合并为更大的簇，直到所有对象都在一个簇中，或者满足某个终结条件。有关凝聚层次聚类的过程描述如下：

设数据集 $D=\{d_1,d_2,\cdots,d_n\}$，每个数据对象具有 p 个特征，即 $d_i=\{d_{i1},d_{i2},\cdots,d_{ip}\}$。首先通过欧氏距离来计算数据对象间的两两距离：

$$\mathrm{dist}(d_i,d_j)=\sqrt{\sum_{m=1}^{p}(d_{im}-d_{jm})^2} \tag{1}$$

通过式(1)来计算数据集中两两数据对象间的距离，找出距离最小的两个数据对象，将它们合并为一个类，同时重新计算这两个数据对象的平均值作为新类的中心点，并计算出所得新类和其他各类的相似度，接着再将相似度最小的两类合并。在计算各类间的相似度时，本文采用了平均链接方法(average linkage)，具体公式如下：

$$d_{\mathrm{avg}}(C_i,C_j)=\frac{1}{n_i n_j}\sum_{d\in C_i}\sum_{d'\in C_j}\|d-d'\| \tag{2}$$

其中：$d_{\mathrm{avg}}(C_i,C_j)$ 表示类 C_i、C_j 间的相似度；$\|d-d'\|$ 表示数据对象 d 和 d' 的距离；n_i 表示类 C_i 中数据的个数；n_j 表示类 C_j 中数据的个数。

不断重复上述迭代过程，直到将所有样本数据都合并成一类为止。因此，经过凝聚层次聚类后，本文可以获得该数据样本的聚类情况，并利用下面的轮廓系统对聚类情况进行初步评估，最终得到较为合理的类别数量作为 K-means 算法 k 的初始值。

2.2 轮廓系数

轮廓系数(silhouette coefficient)是聚类效果好坏的一种评价方式，它结合内聚度和分离度两种因素，可以在相同原始数据的基础上，用来评价不同算法或者算法不同运行方式对聚类结果所产生的影响[13]。对于某一个点 i，其轮廓系数计算过程如下：

$$S(i)=\frac{b(i)-a(i)}{\max(a(i),b(i))} \tag{3}$$

其中：$a(i)$ 表示 i 点向量到与它同簇的其他点的平均距离；$b(i)$ 表示 i 点向量到与它异簇的点的平均距离最小值。从式(3)可知，轮廓系数的值为[-1,1]，如果轮廓系数 $S(i)$ 值越大，则表明 i 点所在的簇就越紧密。因此，对于整个数据集来说，其轮廓系数的计算公式如下：

$$SC=\frac{\sum_{i=1}^{n}S(i)}{n} \tag{4}$$

其中：n 表示数据集中的数量；SC 值越大表示聚类效果越好，反之表示越差。

2.3 算法描述

本文将凝聚层次聚类引入到 K-means 算法中，是为了获取初始簇类的数目，避免 K-means 算法随机选择 k 值而导致分类结果误差较大的问题。其过程是：将待分类的数据集利用凝聚层次聚类方法获取该数据集的初步分类，然后通过轮廓系数对分类结果进行评价，得出较为合理的分类结果，最终将凝聚层次聚类的结果作为 K-means 算法的 k 值进行输入，避免了传统 K-means 随机选择 k 值而影响聚类结果。具体算法流程如图 2 所示。

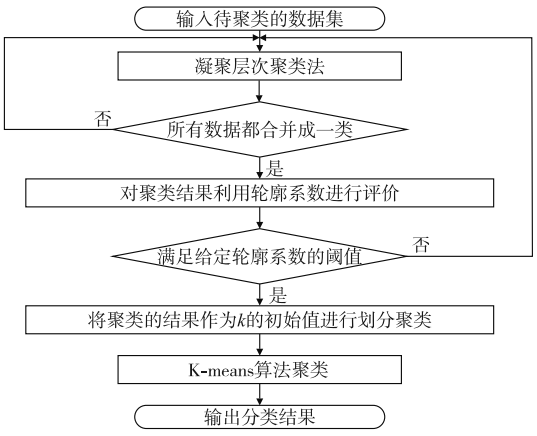

图2 K-means改进算法流程

2.4 Map 和 reduce 函数设计

针对海量数据的聚类挖掘，上述改进的 K-means 算法难以满足运行效率的要求，由于 K-means 聚类的核心是计算样本数据与聚类中心的距离，并分配样本数据到距离其最近的聚类中心，其操作是相互独立的[14]，所以，可以利用并行方式来执行。下面本文将改进的 K-means 算法部署在 MapReduce 框架下运

行,分别设计了 map 和 reduce 函数。其中 map 和 reduce 函数的每一步计算都可以独立分开处理,其各个元素在运行中是相互独立的。在本文中,对于 MapReduce 的过程实际上是由两大部分构成,一部分是由凝聚层次聚类和轮廓系数得到了初始聚类数 k;另一部分是将上述得到的聚类结果作为 K-means 算法的初始 k 值输入进行聚类,但由于设计 map 和 reduce 函数的过程基本相似,所以,将其统一用伪代码来表示。具体流程如图 3 所示。

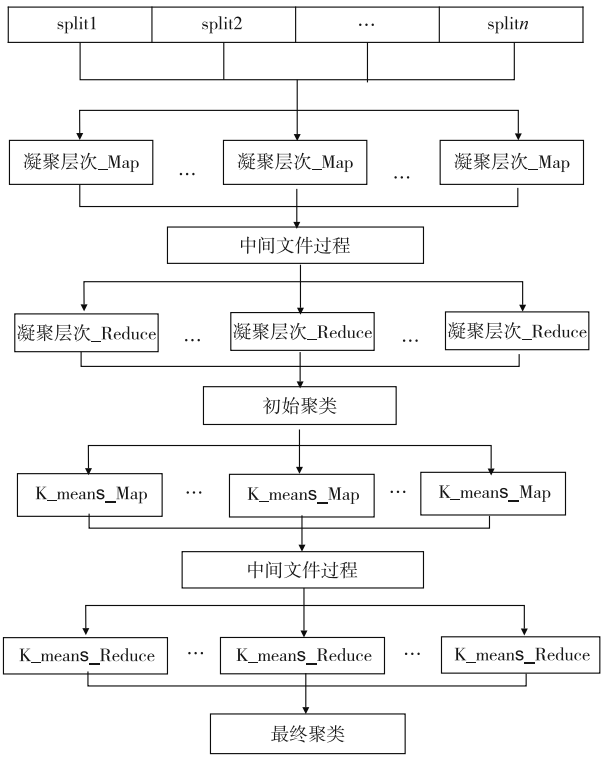

图3　改进K-means算法的MapReduce过程

1)Map 函数

Map 函数的任务是完成每个记录到中心点距离的计算,并重新标记其属于的新聚类类别,最后产生中间键值组〈key,value〉,并将结果输出给 reduce 函数。Map 函数的伪代码描述如下:

```
function map(object key,object value)
{计算样本数据点间的距离;
将这些距离与设定的阈值进行比较;
找出与样本数据点距离最近的那个类,将该数据点纳入该类,并标记类标签;
将所属类别和相应的值写入中间文件;
最后以〈key,value〉的形式输出;
}
```

2)Reduce 函数

Reduce 函数的任务是根据 map 函数得到的中间结果计算出新的聚类中心。即将 map 函数输出的中间文件对键值相同的进行合并,并依据设定的阈值进行计算,重新产生新的聚类。具体的 reduce 函数代码如下:

```
function reduce(object key,object value)
{对相同 key 进行合并;
while(对于 key 相同的 value)
  {计算其数据到质心的距离;
  对上述距离进行分类(凝聚层次聚类、K-means 聚类);/* 如果在凝聚层次聚类时,需用轮廓系数对聚类结果进行评价 */
  }
将〈数据类别,聚类值〉写入最后的聚类群集中;
}
```

3　实验结果与分析

3.1　实验环境和数据集

本实验是在由六台计算机组成的集群上运行,实验采用了 Hadoop 分布式框架,其中一台作为主节点(nameNode),另外五台作为从节点(dataNode)。每一台配置为 Intel Xeon CPU E5-2620 v2 2.10 GHz、64 GB 内存与 2 TB 的硬盘。操作系统是 Ubuntu14.04,JDK 版本是 JDK1.7,Hadoop 版本是 Hadoop2.5.2。本文使用 UCI Machine Learning Repository 所提供的六种数据集(http://archive.ics.uci.edu/ml/datasets.html),分别是 breast cancer、IRIS、GLASS、THYROID、WINE、YEAST,具体实验数据统计如表 1 所示。同时为了能体现该算法在处理大数据方面的性能,本文通过选取约 1.03 GB 的数据集进行加速比和运行速度比较实验。

表 1　本文实验数据集

比较项	breast cancer	IRIS	GLASS	THYROID	WINE	YEAST
类别数	2	3	6	3	3	10
样本数	699	150	214	215	178	1 484
属性数	9	4	9	5	13	8

3.2　实验设置

本文通过设置两个实验来说明提出 K-means 改进算法的有效性。第一个实验是单机性能的测试,通过对比一些已有的算法,来验证本文的 K-means 改进算法的准确率和召回率,以及聚类结果的稳定性;第二个实验是集群性能的测试,本文将改进的 K-means 算法部署在 MapReduce 框架下,验证在大数据环境下,该算法在不同节点数量下的加速比和运行速度。

3.3　实验评价指标

本次实验评价指标是常用的准确率、召回率、加速比。具体如下:

a)准确率和召回率:

$$\text{presicion} = \frac{A}{A+B} \tag{5}$$

$$\text{recall} = \frac{A}{A+C} \tag{6}$$

式(5)表示准确率,式(6)表示召回率。其中 A 表示实际值是正确的且检测也是正确的;B 表示实际值是错误的但检测为正确的;C 表示实际值是正确的但检测为错误的。

b)加速比。它是指同一个任务在单机系统和集群系统中运行消耗的时间比率,用来衡量并行系统的性能和效果。计算公式如下:$Sp = T_1/T_p$。其中 Sp 为加速比;T_1 表示单机性能下的运行时间;T_p 表示在 p 个数量节点上并行运行的时间。

3.4　结果及分析

3.4.1　单机性能比较实验

在单机性能方面的比较,本文分为两组实验进行比较:第一组实验是测试算法聚类结果的准确率和召回率;第二组实验是测试算法聚类结果的稳定性。文中选取传统的 K-means 算法、G-means 算法[3]和 AK-means 算法[5]作为基准对比实验,并应用在 UCI 提供的六种数据集上,其结果如下。

1)准确率和召回率实验

图 4 是几种算法在不同数据集上的聚类准确率比较。从结果来看,本文提出的算法在 UCI 的六个数据集上,其准确率均高于传统 K-means 算法的准确率。而在数据集 IRIS、GLASS、THYROID 及 YEAST 上,本文算法准确率也高于 G-means 和 AK-means 算法,而在 BREAST 数据集上,本文算法与 AK-means 基本相同,高于传统 K-means 和 G-means。这是由于本文算法是先通过凝聚层次聚类获得较为准确的簇数,并将簇数作为 K-means 的 k 初始值,所以具有较高的聚类准确率。

图 5 是几种算法在不同数据集的召回率比较。从结果来

看,本文改进算法的召回率均高于 G-means 和 AK-means,在数据集 BREAST、IRIS、WINE 和 YEAST 上也高于传统 K-means 算法,而在数据集 GLASS 和 THYROID 上低于 K-means。结合图 4 和 5,本文也发现当样本数据集的类别较多时,传统 K-means 及基于它的改进算法聚类的准确率和召回率相对较低。

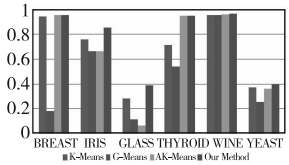

图4 算法准确率实验结果

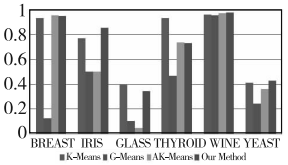

图5 算法召回率实验结果

综合图 4 和 5 来看,本文提出的改进算法在多个数据集中,均能获得较好的准确率和召回率,因此,本文的算法能适应于不同数据集的聚类挖掘。

2)稳定性实验

由于传统 K-means 算法在聚类时结果存在不稳定现象,所以,对提出改进算法的聚类结果进行了稳定性实验。本文选取其中一个数据集并进行 20 次运行实验,其结果如图 6 所示。从图 6 可以看出,传统的 K-means 算法稳定性较差,结果会随着运行次数的不同而呈现出不同的聚类结果,其原因是传统 K-means 算法的初始簇中心是随机选取,因此聚类结果也表现出不稳定的状态。而 G-means、AK-means 及本文提出的方法是通过对传统 K-means 的改进,避免初始簇中心随机选择的问题,因此,聚类结果就表现出较强的稳定性。

3.4.2 集群性能实验

集群性能实验本文是从算法在不同节点上的加速比和运行速度两方面来进行的。为了能体现本文改进算法在海量数据集上的性能,在其中 YEAST 的数据集基础上构造更大规模的数据集,数据规模分别为 369.5 MB 和 1.03 GB;集群节点数由 1 个节点逐步增加到 6 个,验证算法的加速比,具体实验结果如图 7 所示。

从图 7 结果可以看出,当算法应用于 1 个节点时,其运行速度不如单机的,这是因为当数据规模较小时,数据传输与集群通信带来的时间开销的增加超过了数据并行处理带来的时间开销的下降,此时,并行算法反而不如串行算法。但是,当数据规模上升到一定程度,后者将超过前者。因此,数据规模越大,加速效果越显著。这说明了该算法可高效地进行大规模数据的计算。

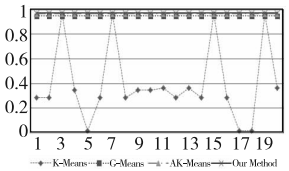

图6 算法准确率稳定性比较

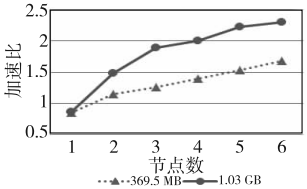

图7 加速比对比

为了能对比几种算法的运行速度,本文将传统的 K-means 算法进行并行运算,运用于数据集 1.03 GB 上进行实验,其结果如图 8 所示。从图 8 结果来看,当节点数为 1 个时,本文算法的运行时间与并行 K-means 算法相差不多,随着节点增加,本文算法运行收敛速度较快于并行 K-means 算法,因此,本文算法应用于大数据时具有较好的收敛速度。

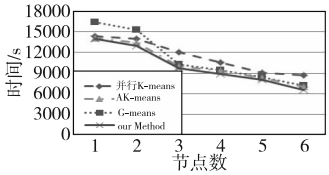

图8 算法运行速度

综上所述,本文算法在聚类的准确率、召回率和稳定性方面具有良好的效果,同时在应用于大数据方面的聚类也有较好的加速比和收敛速度,因此,本文算法就有较好的聚类效果和处理大数据性能。

4 结束语

本文针对大数据背景下的数据挖掘问题,提出了基于 MapReduce 框架下的 K-means 改进算法。首先通过对传统的 K-means 算法进行改进,利用凝聚层次聚类方法获得初始聚类,并通过轮廓系数对凝聚层次聚类进行评估,获得较为合理的聚类数量,接着将获得聚类的结果作为 K-means 算法初始簇数,避免了传统 K-means 的因随机选取初始簇而导致了聚类结果不稳定的现象,最后将改进的 K-means 算法部署在 MapReduce 框架下运行。实验结果表明,本文提出的改进算法在准确率、召回率和稳定性能方面都优于传统的 K-means 算法,同时在 MapReduce 框架下,该算法随着数据规模的增加,其加速比也就越显著。因此,本文提出的算法是有效可行的。在 K-means 聚类算法中,如何设置参数 k 一直都是一个开放性课题,下一步研究工作中,本文将继续寻求最优的参数 k,使得聚类效果和精度能达到一个最佳理想的结果。

参考文献:

[1] Cui Xiaoli, Zhu Pingfei, Yang Xin, *et al*. Optimized big data K-means clustering using MapReduce[J]. Journal of Supercomputing, 2014, 70(3): 1249-1259.

[2] Lin Kunhui, Li Xiang, Zhang Zhongnan, *et al*. A K-means clustering with optimized initial center based on Hadoop platform[C]// Proc of the 9th International Conference on Computer Science & Education. Piscataway, NJ: IEEE Press, 2014: 263-266.

[3] Debatty T, Michiardi P, Mees W, *et al*. Determining the k in K-means with MapReduce [C]//Proc of EDBT/ICDT Workshops. 2014: 19-28.

[4] Yuan Qilong, Shi Haibo, Zhou Xiaofeng. An optimized initialization center K-means clustering algorithm based on density[C]//Proc of IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems. Piscataway, NJ: IEEE Press, 2015: 790-794.

[5] Kettani O, Ramdani F, Tadili B. AK-means: an automatic clustering algorithm based on K-means[J]. Journal of Advanced Computer Science & Technology, 2015, 4(2): 231-236.

[6] Ma Li, Gu Lei, Li Bo, *et al*. An improved K-means algorithm based on MapReduce and grid[J]. International Journal of Grid & Distributed Computing, 2015, 8(1): 189-200.

[7] 王永贵, 武超, 戴伟. 基于 MapReduce 的随机抽样 K-means 算法[J]. 计算机工程与应用, 2016, 52(8): 74-79.

[8] 李兰英, 董义明, 孔银, 等. 改进 K-means 算法的 MapReduce 并行化研究[J]. 哈尔滨理工大学学报, 2016, 21(1): 31-35.

[9] 刘义, 景宁, 陈荦, 等. MapReduce 框架下基于 R-树的 K-近邻连接算法[J]. 软件学报, 2013, 24(8): 1836-1851.

[10] 梁俊杰, 李凤华, 刘琼妮, 等. MapReduce 框架下的优化高维索引与 KNN 查询[J]. 电子学报, 2016, 44(8): 1873-1880.

[11] 孙玉强, 李媛媛, 陆勇. 基于 MapReduce 的 K-means 聚类算法的优化[J]. 计算机测量与控制, 2016, 24(7): 272-275.

[12] 梁亚声, 徐欣, 成小菊, 等. 数据挖掘原理·算法与应用[M]. 北京: 机械工业出版社, 2015.

[13] Xia S, Li W, Zhou Y, *et al*. Improved K-means clustering algorithm [J]. Journal of Southeast University, 2007, 23(3): 435-438.

[14] 李钊, 李晓, 王春梅, 等. 一种基于 MapReduce 的文本聚类方法研究[J]. 计算机科学, 2016, 43(1): 246-250.

[15] 李建江, 崔健, 王聃, 等. MapReduce 并行编程模型研究综述[J]. 电子学报, 2011, 39(11): 2635-2642.

[16] 许丞, 刘洪, 谭良. Hadoop 云平台的一种新的任务调度和监控机制[J]. 计算机科学, 2013, 40(1): 112-117.