基于 Hadoop 平台 Canopy-Kmeans 聚类算法优化改进研究

周功建

（厦门大学 嘉庚学院，福建 漳州 363105）

摘要：在分析 Hadoop 平台架构和 Canopy-Kmeans 聚类算法的基础上，对 Canopy-Kmeans 算法进行了并行化优化改进，通过统计学思维对数据分组抽样后聚类以方便并行化和降低时间复杂度，利用最小最大原则优化 Canopy 初始中心点选取，用数据异度均值抽样法保证从原数据中均匀提取数据样本，并对 Kmeans 迭代计算过程进行优化。结合 Hadoop 平台下 MapReduce 框架将改进算法进行并行化设计实现。实验表明，对海量数值数据进行聚类时，改进的 Canopy-Kmeans 并行算法是有效的、收敛的，在聚类准确率和时效性上都有一定程度的提升。

关键词：Hadoop；MapReduce；聚类分析；Kmeans 算法；Canopy-Kmeans 算法；加速比

中图分类号：G621　　**文献标志码**：A　　**文章编号**：1008-6021(2018)04-0117-06

一、引言

大数据时代，随着互联网和信息技术的快速发展，数据量呈指数式增长。如何快速准确地从海量数据里挖掘出有价值的知识和信息是数据挖掘技术最重要的特征，而聚类分析是数据挖掘中一个非常重要的研究领域[1]。传统的聚类分析算法在空间复杂度和时间复杂度上都有一定的局限性，利用云计算平台对其进行并行化改进优化，可以有效降低聚类算法的时间复杂度和空间复杂度，节约聚类时间，从而更好地满足实际问题中数据挖掘的需要[2]。

Hadoop 是目前使用最广的云计算平台，利用其中的 MapReduce 模块来提高聚类算法的效率，已经成为大数据聚类分析研究的热点[3]。针对经典的基于划分的 Kmeans 聚类算法，人们根据不同的要求和使用环境，在传统 Kmeans 算法的基础上提出了许多改进思路，其中结合 Canopy 算法和 Hadoop 云平台改进生成的 Canopy-Kmeans 并行聚类算法及其优化就是很有代表性的一类。张石磊、武装等在 Hadoop 平台上实现了 Kmeans 和 Canopy-Kmeans 算法的并行化，并且验证了 Canopy-Kmeans 并行算法比 Kmeans 并行算法在不同的数据集下有更高的聚类准确率和更快的收敛速度[4]。李钊、王春梅等基于 Hadoop 平台设计了 Canopy-Kmeans 并行化算法，应用“最小最大原则”对 Canopy 中心点随机性选取问题进行改进，并用海量新闻数据做聚类测试验证改进算法的性能和有效性[5]。李兰英、董义明等采用“平均距离估值”方法对 Canopy 算法区域半径 $T1$ 和 $T2$ 取值具有盲目性问题进行改进，可以得到比较准确的区域半径取值，并在 Hadoop 云平台上对改进后的 Canopy-Kmeans 算法进行了实现和验证[6]。

在学习借鉴他人研究基础上，利用统计学思维对数据分组抽样后聚类，并结合数据异度均值抽样法和最小最大法原则对 Canopy-Kmeans 算法进行优化改进；并在 Hadoop 平台上对改进的 Canopy-Kmeans 算法进行并行化设计与实现；最后通过实验验证算法并行执行时的加速比和可扩展性。结果表明改进的 Canopy-Kmeans 并行算法是有效的、收敛的。

收稿日期：2018-08-29

基金项目：福建省教育科学“十三五”规划重点课题（项目编号：FJJKCGZ16-165）

作者简介：周功建（1976－），男，湖北襄阳人，副教授，硕士。研究方向：电子商务、信息管理与信息系统、大数据处理和挖掘等。

二、Hadoop 云计算平台

Hadoop 云平台是一个开源分布式系统框架，遵循了 Google 的 Bigtable、GFS、MapReduce 三大关键技术及其设计思想[2,7]；在 Hadoop 框架中，MapReduce 和 HDFS(Hadoop Distributed File System)是两个最核心模块，分别负责海量数据的存储与计算。HDFS 是一种运行在大量普通配置的机器节点上的分布式文件系统[7]。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，并提供数据分割、任务调度等细节，实现程序高度并行性和可扩展性[3,8]。

三、Canopy-Kmeans 算法研究

(一) Kmeans 算法原理

Kmeans 算法是一种基于划分的聚类算法，利用数据对象间的距离作为相似性的评价指标[4,7]。聚类过程是一个不断循环迭代的过程，对数据点进行簇划分和对簇中心点进行调整交替执行，直到簇中心点不再发生变化为止，最后生成的同一簇内数据相似度高，不同簇间数据相似度低，其算法执行流程如图 1 所示。

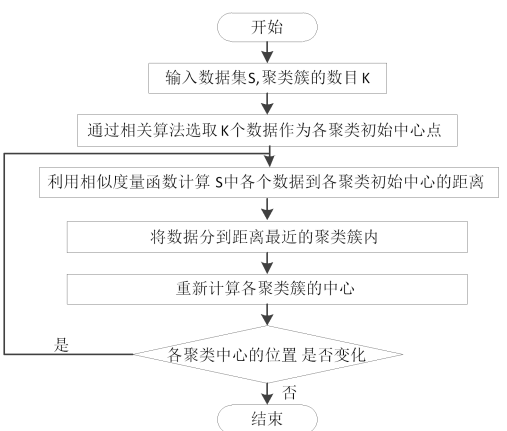

图 1　Kmeans 算法聚类流程

Kmeans 算法思想简单、理论可靠、处理大数据集时有较强的鲁棒性和可伸缩性，实际应用中得到广泛关注，但算法也存在如下不足之处：

(1)Kmeans 算法的聚类簇数量 k 值需人为指定，不同的 k 值对聚类结果会产生很大影响；并且每个簇的初始中心点是随机选取的，易导致聚类结果不稳定和聚类质量不高[9]。如何准确确定 k 值和选择初始中心点是使用 Kmeans 算法聚类时需思考的优化改进方向。

(2)串行执行时，当聚类数据量为 n，聚类簇个数为 k，迭代次数为 t 时，算法的时间复杂度为$O(nkt)$；可以看出当聚类的数据集较大时，算法的运行时间将大大增加，如果能将聚类过程改为并行化执行，则可有效降低时间复杂度，提升聚类效率[4,8]。

(二)Canopy-Kmeans 算法原理

Canopy-Kmeans 算法是一种借助 Canopy 算法进行改进优化后的 Kmeans 算法，在 Canopy-Kmeans 算法中，先用 Canopy 算法对数据集进行“粗”聚类预处理，快速算出 k 个 Canopy 中心点，再用 Kmeans 算法对数据集进行“细”聚类，生成聚类结果[5,9]。Canopy-Kmeans 算法执行步骤如下：

(1) 将待聚类的数据集向量化为一个 List 集合，并指定两个距离阈值 $T2$ 和 $T1$($T2 < T1$)；

(2)从集合 List 中随机删除一个数据对象 P，构成一个新的 Canopy；

(3)对于 List 中剩余数据对象，如果与 P 的间距小于 $T1$，就把它指派到 P 所在的 Canopy 中；如果与点 P 的间距小于 $T2$，则将它从 List 中删除；

(4)重复步骤(2)和步骤(3)，直到 List 为空，至此将形成多个 Canopy；

(5) 将得到的 Canopy 数目作为 k 值，每个 Canopy 的中心点作为初始聚类中心点代入 Kmeans 算法中进行再次聚类，生成最终聚类结果。

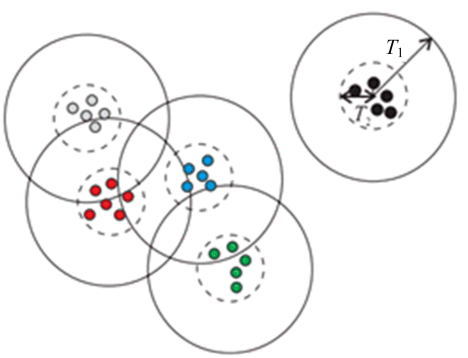

图 2　Canopy 聚类结果示意图

在 Canopy-Kmeans 算法中，无须事先指定 k 值和随机选取初始中心点，而是利用 Canopy 算法聚类求得，可有效解决 k 值人为指定和随机选取初始中心点的盲目性；但 Canopy-Kmeans 算法仍有以下不足之处：

(1)Canopy 算法在求解过程中各个 Canopy 的

初始中心点仍然是随机选取的，这种做法可能会造成聚类结果的不稳定[1,5]；

(2)串行执行时，Canopy-Kmeans 算法的时间复杂度为 $O(nkt\ f^2/c)$，其中 c 为 Canopy 的总个数，n 为数据的规模，t 为算法迭代的次数，k 为最终生成的聚类数目，f 为平均每个数据对应的 Canopy 数量[8]；可以看出，在处理海量数据时，仍然有较高的时间复杂度[2,6]。

四、改进的 Canopy-kmeans 算法

通过对 Canopy 算法聚类过程、Kmeans 算法迭代过程及时间复杂度等方面分别进行优化，改进的 Canopy-Kmeans 算法执行流程如图 3 所示。

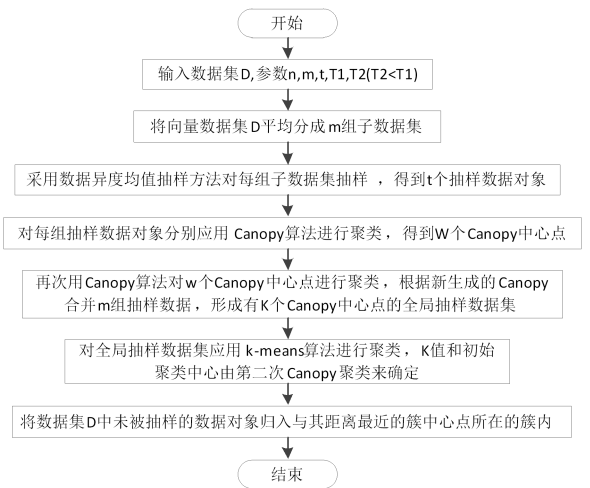

图 3　改进的 Canopy-Kmeans 算法执行流程

(一) Canopy 聚类过程优化

为避免聚类陷入局部最优解，针对 Canopy 聚类时初始中心点选取随机性问题，利用“最小最大原则”对 Canopy 算法进行优化，基本思想为：在将数据聚类划分为若干个 Canopy 过程中，任意两个 Canopy 中心点之间的距离应尽可能远；即如果已知 m 个 Canopy 中心点，则第 $m+1$ 个 Canopy 初始中心点应为所有候选数据对象与前 m 个 Canopy 中心点之间最小距离中的最大者[6]。具体优化流程如下：

(1)从数据集 List 中随机删去第一个点 $P1$，计算与其对应的 Canopy；

(2) 计算 List 中剩余的点到点 $P1$ 的距离，选取其中距离最大的点 $P2$ 作为第二个 Canopy 的初始中心点，并从 List 中删除点 $P2$，计算与其对应的Canopy；

(3)计算 List 中剩余的点到所有已生成 Canopy 中心点最小距离 d，选取 d 的数值最大的点作为下一个 Canopy 的初始中心点，计算与其对应的 Canopy；

(4)重复执行步骤(3)，直到数据集 List 为空。

另外，为了提高下一步 Kmeans 聚类的稳定性和准确率，还需检测每个 Canopy 簇中数据对象的个数，若一个 Canopy 簇只有一个或很少的数据对象，则认为该 Canopy 簇是孤立的，应将他们从总的 Canopy 簇中减掉，从而得到 Kmeans 聚类过程的 K 值和初始聚类中心。

(二) kmeans 迭代过程优化

Canopy 算法作为 Kmeans 聚类的初始化操作，在快速得到 k 个 Canopy 中心时，会对每个 Canopy 中的数据对象进行标注，每个数据对象至少属于一个 Canopy，也可能出现在多个 Canopy 中。可以断定数据在同一 Canopy 中相似度高，不同 Canopy 中相似度低，因此在 Kmeans 迭代时，不属于同一个 Canopy 中的数据之间可不再计算它们的相似性。当计算数据所属聚类簇时，只需计算它到所属 Canopy 中心点的距离，并将其归入到距离最近的 Canopy 中心点所属类簇中，形成相不重叠的簇，从而减少了迭代过程中的计算量。在 Canopy 较小，且相互间数据重叠不多的情况下可有效降低 Canopy-Kmeans 算法的时间复杂度。

(三)时间复杂度优化

为进一步降低算法的整体时间复杂度，利用统计学思维对数据集进行分组抽样后聚类，以增强其时效性。具体思路为：先将待聚类数据集进行分组，从每组抽样出一定量的数据，并对各组抽样数据分别进行 Canopy 聚类，然后对求得的全部 Canopy 中心点再次进行 Canopy 聚类，生成 k 个 Canopy 中心点，代入 Kmeans 算法中对全部抽样数据进行聚类划分，产生新的 Kmeans 聚类簇，最后将所有未被抽样到的数据对象归类到距离最近的 Kmeans 聚类簇中。

分组的目的是为了更方便在 Hadoop 平台上进行并行化实现。在对数据抽样时，利用数据异度均值抽样方法来确保数据样本尽量均匀分布在原始数据中，使其能够代表全体数据，从而保证算法的稳定性和正确性。数据异度均值抽样操作步骤如下：

(1)对于非空数据集 $S=\{c_i \mid c_i=(c_{i1},c_{i2},\cdots,c_{ip}),i=1,2,\cdots n\}$，用公式(4-1)找

出集合 S 中最小数据对象 $c_m=(c_{m1},c_{m2},\cdots,c_{mp})$，$1\leqslant m\leqslant n$；

$$S=\sqrt{\sum_{l=1}^{p}c_{il}^2}。\quad(1)$$

(2)用公式(2)计算所有其余数据对象到 c_m 的距离，形成一个距离值集合 DIS；

$$D(c_i,c_j)=\sqrt{\sum_{l=1}^{p}(c_{il}-c_{jl})^2}。\quad(2)$$

(3)根据 DIS 中的距离值，按从小到大的顺序对集合 S 中的全部数据对象进行升序排序；

(4)利用公式(3)从排好序的集合 S 中均匀地选取 t 个数据对象作为抽样数据集 $RIS=\{c_1,c_2,\cdots,c_t\}$。

$$h_i=\left[1+\frac{(i-1)(n-1)}{t-1}\right],(i=1,2,\cdots,t)。\quad(3)$$

五、基于 Hadoop 平台 Canopy-kmeans 改进算法的并行化设计实现

在改进的 Canopy-Kmeans 算法中，因对数据集采取了分组抽样策略，所有数据对象到各 Canopy 质心点距离的计算操作都可以独立执行，相互之间不存在任何依赖关系，因此改进算法完全可以应用 Hadoop 平台上的 Map-Reduce 编程框架进行并行化设计实现，将执行过程分成三个阶段，具体流程如图 4 所示：

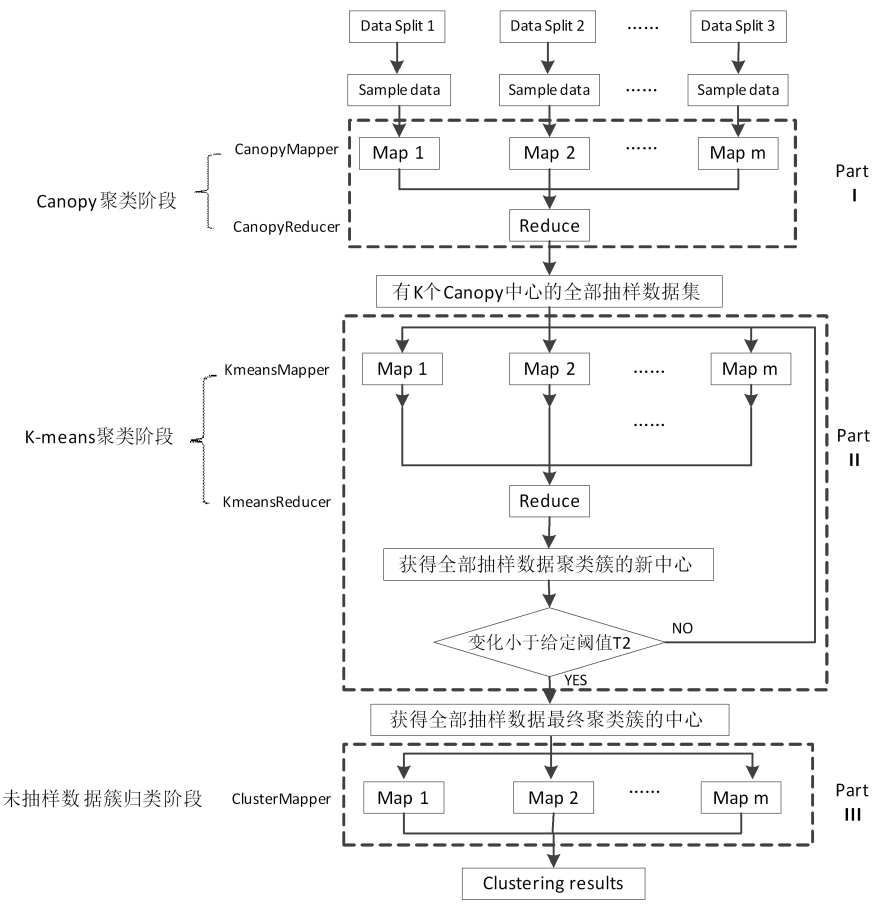

图 4　基于 Hadoop 平台改进 Canopy-Kmeans 算法并行化执行流程

(一) Canopy 聚类阶段

1.Map 过程

从分组数据集 split 中采用数据异度均值抽样方法抽取 t 个数据对象，然后对抽取的数据对象利用 Canopy 算法聚类，得到多个重叠的 Canopy，输出 Canopy 中心点集合。

2.Reduce 过程

使用改进的 Canopy 算法将 Map 过程中得到的多组 Canopy 中心点合并为一组，再将所有抽样的数据对象划分到距离最近的 Canopy 中，并重新计算各个 Canopy 的中心点，将结果保存至 HDFS 中。

```
Input: 从对应的分组子数据集S中应用数据异度均值抽样方法抽样的t个数据对象，记
作数据集D; 距离间隔阈值T2 < T1;
Output 抽样数据集D的Canopy中心点集合X
X=null;
While (D!=null){
if(集合X为空){
  从D中随机选取并删除一个数据对象作为第一个 Canopy中心点;
}else{
  采用最小 最大法原则从D中选取并删除一个数据对象作为下一个 Canopy的中心;
}
While(遍历D){
  计算D中剩余数据对象到新 Canopy中心点的距离 dis;
  if(dis<T2){
   从D中删除 该数据对象，并为其打上强标记;
  }else if(d<T1){
    为该数据对象打上弱标记;
      }
  }
将新Canopy中心点放入集合X中;
}
```

图 5　Map 函数执行伪代码

```
Input: 各组抽样数据集 Canopy中心点集合 X={X1,X2,...,Xn}，区域半径T2
Output: 有K个Canopy中心的全部抽样数据集 Q
1.先对集合X运用改进的 Canopy算法，得到K个Canopy，过程与map函数处理过程类似;
2.将所有抽样数据按照区域半径 T2划分到距离最近的canopy中心点所在的簇中，生成
canopyGroup
3.对每个canopy的中心点重新计算，得到新的中心点 canopyCenter
FOR i=0 to K-1 DO
  key =i
  canopyCenter=getCanopyCenter(canopyGroup.get(i))
  value = canopyCenter
  output(key, value)
END FOR
```

图 6　Reduce 函数执行伪代码

（二）Kmeans 聚类阶段

Kmeans 算法每一轮迭代都启动一次 Map-Reduce 任务，算法所需的 k 值及初始聚类中心由 Canopy 算法得到，因 Canopy 算法对每个数据做了所属 Canopy 标注，所以 Kmeans 聚类过程只需计算各数据对象与所属 Canopy 中心的距离，选择距离最短的 Canopy 中心对其进行簇归类。

```
  Input:  全部抽样数据集Canopy中心点集合U
  Output: 全部抽样数据集K-means聚类中心点集合U'
//Canopy中心点集合U作为K-means聚类初始K中心点集合
 1. U'=U
 2. while  U' 发生改变
 3.通过Map函数比较已标注的输入数据点与对应中心点的距离，
   输出当前数据点及对应最近距 离的中心点;
 4.通过Reduce函数将同一中心点下的数据点归并，
    计算均值并将结果输出作为新的中心点;
 5. end while
```

图 7　Kmeans 迭代执行伪代码

在 Kmeans 算法每一轮 Map-Reduce 过程结束后，将本次输出结果与上次进行比较，当两者差值小于给定的阈值 $T2$，就进入后续过程；否则，就将本次的输出结果作为新一轮的 Map-Reduce 输入传输到 HDFS 上，再次进入下一轮 Kmeans 迭代，直到满足前后两次输出结果差值小于阈值 $T2$ 后进入后续过程。

（三）未抽样数据簇归类阶段

根据 Kmeans 聚类结果，使用 Map 函数对未被抽样到的数据集进行归类操作，把数据对象归入到离其距离最近的簇中心点所在的聚类簇中，生成最终聚类结果。

```
Input:  含K个聚类簇中心点，未被抽样的 数据集G
Output: 全体数据的聚类结果 N
While (G不为空时) {
 //找出与数据对象 p最近的聚类簇中心
for(i=O; i<k; i++) {
  Cluster_ ID=0;
 min-distance = distance(p, cluster[i]);
if(distance( p, cluster[i])< min-distance){
          Cluster_ ID = i;
 //记录下来与数据对象 p距离最近中心点所在的簇的 ID
      }
}
 //将数据对象 point归入簇ID为Cluster_ID的簇中,并从G中删除数据对象 P
Emit_Intermediate(Cluster_ID, p);    }
```

图 8　Map 函数执行伪代码

六、实验验证与分析

（一）实验平台和实验数据

实验平台采用开源 Hadoop 分布式框架，由 4 台计算机组成集群，其中 1 台作为主节点（Namenode），另外 3 台作为从节点（Datanode）。每一台配置为 Intel Xeon CPU ES-2620 2GHz，128GB 内存与 2TB 的硬盘。操作系统为 Ubuntu14.04，JDK 版本是 jdk1.7，Hadoop 版本是 Hadoop2.5.2。

实验数据采用 UCI 数据集下 60 维的 Synthetic_Control 数据集，分别采集了 100 k、200 k、400 k、800 k、1 000 k 五种不同大小的数据集。分组抽样的样本数据占原始数据量的 60%，Canopy 聚类所需的距离阈值 $T1$ 和 $T2$ 的设定可以根据实际情况通过交叉验证给出。另外，为避免算法随机性的影响，每类数据集均取 20 次运行结果的平均值进行分析。

（二）加速比分析

加速比（speedup）可用来衡量一个算法并行化执行的有效性，是算法在并行系统和串行系统中运行所消耗时间的比率。其定义为：$S_p = T_l / T_p$，其中，T_l 代表串行时的处理时间，T_p 代表并行时的处理时间。当加速比指标 S_p 越大，算法效率越高。

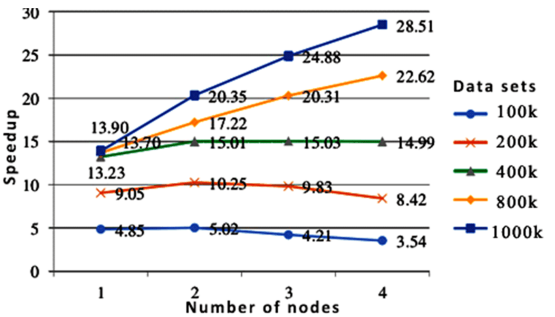

图 9　改进的 Canopy_Kmeans 算法加速比

可以看出，数据集在 100 k、200 k、400 k 规模时，随着节点数目的增加，加速比曲线趋向平缓。这是因为 100 k、200 k、400 k 数据集的数据量相对较小，处理时间相对较短，节点数的增加，使数据传输、任务调度、资源管理等方面的时间开销加大，从而降低了算法的执行效率。而在处理 800 k、1 000 k 规模的较大数据集合时，加速比曲线呈现线性增长，说明改进的 Canopy-Kmeans 算法在并行化执行时能够有效提升聚类效率，并且数据量越大时算法的效率越高。

（三）可扩展性分析

可扩展性（scalability）反映了集群计算节点数目对并行化效率的影响，主要用于检验并行化算法的实用性。其定义为：$E = S_P / P$，其中 P 表示集群计算节点数目，S_P 表示加速比；若 E 越大，则算法可扩展性越好。

可以看出，随着集群节点数的增加，算法的可扩展性会下降，这是因为节点数的增多会增加网络通信、任务调度、资源管理等方面的开销。但当数据的规模越来越大，节点越来越多时，算法的并行扩展比曲线下降的幅度越来越小，最后趋向维持平滑，说明随着节点数的增加，算法在处理大数据时的效率基本保持稳定。

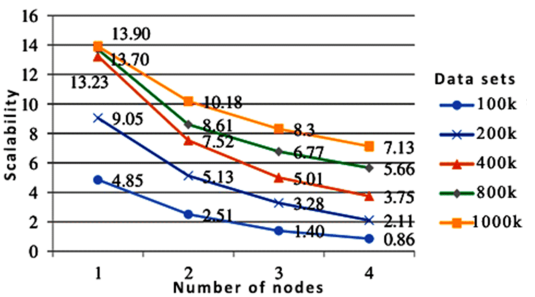

图 10　改进的 Canopy_Kmeans 算法可扩展性

七、结语

综上所述，改进的 Canopy-Kmeans 聚类算法通过将数据集进行分组抽样后聚类减少了整体计算量；最小最大原则解决了 Canopy 聚类初始中心点选取随机性问题；数据异度均值抽样法确保能均匀提取数据样本，使其能代表全局数据，避免聚类陷入局部最优解，保证了算法的正确性；对 Kmeans 迭代计算过程进行优化可进一步降低算法的时间复杂度；最后在分布式 Hadoop 集群系统中测试实验表明，对海量多维数值数据进行聚类处理时，改进的 Canopy-Kmeans 算法是有效的、收敛的，在聚类准确率和时效性上都有一定程度的提升。但另一方面，抽样数据规模如何确定以及 Canopy 过程中的两个距离阈值（$T1$ 和 $T2$）的取值如何准确给出都对聚类结果的精度和稳定性有较大影响，这些问题还有待后续更进一步深入研究。

参考文献：

[1]　KETTANI O，RAMDANI F，FADILI B. AKmeans：An Automatic Clustering Algorithm Based on Kmeans[J]. Journal of Advanced Computer Science & Technology，2015，4(2)：231-236.

[2]　XIA S，LI W，ZHOU Y，*et al*. Improved Kmeans Clustering Algorithm[J]. Journal of Southeast University，2007，23(3)：435-438.

[3]　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，2015：102-154.

[4]　万旭. 基于 Hadoop 平台的聚类算法研究[D]. 西安：西安电子科技大学，2016：24-36.

[5]　梁彦. 基于分布式平台 Spark 和 YARN 的数据挖掘算法的并行化研究[D]. 广州：中山大学，2014：32-57.　（下转第 128 页）

[7] 陈书申.经典土压力理论的局限与小变位土压力计算的建议[J].土工基础,1997,11(2):15-21.
[8] 马祥东.CFG 桩复合地基在石灰岩溶地区的应用[J].冶金丛刊,2017(3):45-46.
[9] 东营世纪城 1 号楼岩土工程勘察报告[R].潍坊:山东省潍坊基础工程公司,2012:9-10.

Application of CFG Pile Composite Foundation in the Yellow River Delt

CHEN Qinyuan
(Anhui Radio and TV University, Hefei 230022,China)

Abstract: The project of the foundation treatment of No.1 Building in Dongying Century City was selected to explore the optimal foundation treatment scheme suitable for the soil quality in the Yellow River Delta. Through the comparison of the schemes, the CFG pile composite foundation is selected and calculated and analyzed. The analysis results show that the foundation bearing capacity and settlement meet the requirements of the specification. Practice has proved that it is reasonable and feasible to use CFG pile composite foundation to reinforce soft soil foundation in the Yellow River Delta.

Keywords: CFG Pile; composite foundation; foundation design; pile testing; settlement observation; Yellow River Delta

[责任编辑 李潜生]

(上接第 122 页)

[6] 毛典辉.基于 MapReduce 的 Canopy-Kmeans 改进算法[J].计算机工程与应用,2012,48(27):22-26.
[7] 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.
[8] 王永贵,武超,戴伟.基于 MapReduce 的随机抽样 Kmeans 算法[J].计算机工程与应用,2016,52(8):74-79.
[9] 缪裕青,张锦杏.一种基于 Hadoop 平台的新聚类算法[J].计算机科学,2014,41(4):269-272.

Research on Optimization and Improvement of Canopy-Kmeans Clustering Algorithm Based on Hadoop Platform

ZHOU Gongjian
(Xiamen University Tan Kah Kee College, Zhangzhou Fujian 363105, China)

Abstract: Based on the analysis of Hadoop platform architecture and Canopy-kmeans clustering algorithm, the Canopy-kmeans algorithm is optimized for parallelization. The data is sampled and clustered by statistical thinking to facilitate parallelization and reduce time complexity. The minimum and maximum principle optimizes the Canopy initial center point selection, and the data heterogeneous mean sampling method is used to ensure uniform extraction of data samples from the original data, and the Kmeans iterative calculation process is optimized. Combined with the MapReduce framework under the Hadoop platform, the improved algorithm is designed and implemented in parallel. Experiments show that the improved Canopy-Kmeans parallel algorithm is effective and convergent when clustering massive numerical data, and has a certain degree of improvement in clustering accuracy and timeliness.

Keywords: Hadoop; MapReduce; cluster analysis; Kmeans algorithm; Canopy-Kmeans algorithm; speedup

[责任编辑 李潜生]