第 43 卷 第 6A 期
2016 年 6 月
计 算 机 科 学
Computer Science
Vol. 43 No. 6A
June 2016

一种基于聚类中心的快速聚类算法

周鹿扬[1] 程文杰[1,2] 徐建鹏[1,2] 徐 祥[1]
(安徽省农村综合经济信息中心 合肥 230001)[1] (安徽省农业气象中心 合肥 230001)[2]

摘 要 针对 k-means 算法采用单一的聚类中心描述一个类簇,一般不能有效适用于任意形状簇的缺陷,在研究 k-means 算法以及初始聚类中心优化算法的基础上,考虑将数据集中较大或延伸状的簇分割成若干球状簇,而后合并这些小簇。该算法首先选取一组分布于高密度区域的聚类中心,将聚类中心周围的对象划分到离其最近的聚类中心形成子簇,再根据子簇之间的连通性完成子簇合并。实验证明,该算法能有效适应任意形状簇,并保持了 k-means 算法简单的优点。

关键词 聚类算法,聚类中心,簇合并,快速
中图法分类号 TP301.6 文献标识码 A

Fast Clustering Algorithm Based on Cluster-centers

ZHOU Lu-yang[1] CHENG Wen-jie[1,2] XU Jian-peng[1,2] XU Xiang[1]
(Rural Comprehensive Economic Information Center of Anhui Province, Hefei 230001, China)[1]
(Anhui Agrometeorological Center, Hefei 230001, China)[2]

Abstract To deal with the problem that classical k-means algorithm inefficiently adapt to clustering for all kinds of clusters, in this paper an algorithm which is improved on k-means algorithm using optimization cluster-center was proposed. It divides large or extended-shaped cluster into a number of globular clusters, and then merges these small clusters. Firstly, a group of cluster centers located in the high-density region are selected, and the object around the cluster center is divided to its nearest cluster center forming the sub-cluster. Then the merger is completed in accordance with sub-cluster connectivity between sub-clusters. Experimental results show that the algorithm can adapt to irregular shape cluster and is simple.

Keywords Clustering algorithm, Cluster-center, Cluster consolidation, Fast

1 引言

聚类分析作为一种重要的数据处理技术,是机器学习和人工智能领域的重要课题之一,被广泛应用于数据挖掘、模式识别、计算机视觉、信息检索和生物信息学的研究中。聚类算法的目的是在多维空间中将给定的数据集划分为若干子集,使得具有相似性质的数据归于同一类,不相似性质的数据尽可能地分开[1]。作为统计学的一个分支,聚类分析已经被广泛研究了若干年,主要包括划分法、层次法、基于密度的方法等。k-means 算法作为经典的基于划分的聚类算法,具有理论可靠、算法简单、收敛速度快等特点,但它仅适合对数值型数据聚类且聚类结果为球形的数据集,对异常数据比较脆弱。针对采用单一聚类中心只能描述球状簇的缺陷,CURE、DBSCAN 等算法采用多个代表点(或中心点)描述一个任意形状的簇。DBSCAN 算法是基于密度的聚类方法中的代表方法,可以发现任意形状的类簇,但对人工设定的半径和最少数目参数敏感。CURE 是一种针对大型数据库的高效的层次聚类算法,采用多个点代表一个簇,可以较好地处理非球形数据集,并且在处理大数据的时候采用了随机取样及分区的方法来提高其效率[2]。

近年来已有研究人员采用局部聚类后根据簇之间的联系完成最终聚类,一方面克服了传统基于划分的聚类算法只适用于球状簇的缺陷,另一方面通过一些改进提高了算法速度。如雷小锋等人提出一种基于 k-means 局部最优性的高效聚类算法[3],该算法借助 k-means 算法的局部最优性,对同一个数据集进行多次不同初始聚类中心的 k-means 聚类,以产生数种不同的聚类结果,通过对这些聚类结果求交集,可以将整个数据集分割成若干个子簇并构造出关于子簇的加权连通图,根据子簇之间的连通性合并子簇;张丽等人提出基于划分与层次方法的混合聚类算法,该算法包括分裂和合并两个阶段,在分裂阶段反复采用 k-means 算法,将数据集划分为多个同质的子簇,在合并阶段采用凝聚的层次聚类算法[4,5];李晓翠等人提出一种基于代表点的快速聚类算法,通过领域内邻居结点的数量选取代表点,代表点代表一定数量被代表点,形成类簇,通过密度相关性合并类簇,最后将噪声点划分到最近的类簇[6];马儒宁等人提出一种多层核心集凝聚算法,该算法使用多层核心集表述聚类结构,使得每一层数据集向其核心集凝聚,同时上层的核心集自动成为下层的数据集,每层核心集规模按比例迅速减少,控制了凝聚过程的迭代次数[7]。

受上述文献启发,对划分子簇及子簇合并过程进行改进,

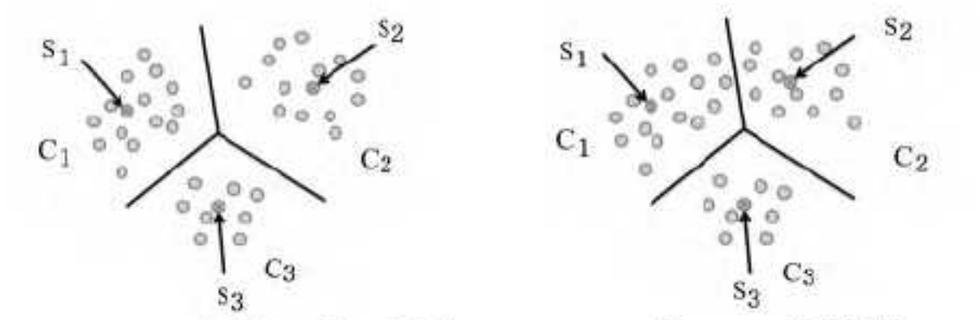
本文受国家科技计划项目(2014BAD10B05)资助。
周鹿扬(1982—),男,硕士生,主要研究方向为农村信息化、数据挖掘。

使算法速度更快，提出了基于聚类中心的快速聚类算法 IMC-CA。该算法是在初始聚类中心优化的 k-means 算法基础上的改进，首先选取一组聚类中心，将非聚类中心点划分到离其最近的聚类中心，形成子簇，再通过子簇之间的连通性合并子簇。

2 准备工作

2.1 算法思想

k-means 算法在聚类球状簇时是高效的，它通过迭代的方法寻找到合适的聚类中心，这种寻找聚类中心的方法虽然看似盲目，但十分有效，在没有陷入局部最优的情况下最后一次迭代的聚类中心往往是合理的。图 1(a)所示为 k-means 算法在最后一次迭代时得到的聚类结果及聚类中心，划分结果是正确的。针对图 1(b)的非球状簇(即簇 C_1、C_2 靠得过近的情形)，假设仍能找到合理聚类中心 $\{S_1, S_2, S_3\}$，类似于 k-means 算法，将周围的点划归到最近的聚类中心，形成簇 $\{C_1, C_2, C_3\}$，最后合并连通的簇(靠得过近)C_1、C_2，算法结束。



(a) k-means 算法的最后一次迭代　　(b) 簇 C_1、C_2 靠得过近

图 1

2.2 相关定义

给定数据集合 $S=\{x_1, x_2, \cdots, x_n\}$，任意的对象 x_i 有属性 $\{x_{i1}, x_{i2}, \cdots, x_{ik}\}$，给出以下定义。

定义 1(数据对象 i, j 之间的距离)

$$d_{ij}=\sqrt{(x_{i1}-x_{j1})^2+(x_{i2}-x_{j2})^2+\cdots+(x_{ik}-x_{jk})^2}$$

定义 2(数据对象 i 的局部密度)

$$\rho_i=\sum_{j=1}^{n}\chi(d_{ij}-d_c)$$

其中，d_c 为截断距离，$\chi(x)=\begin{cases}1, & x<0\\ 0, & x\geqslant 0\end{cases}$。

定义 3(数据集合的平均局部密度)

$$AD=\frac{1}{n}\sum_{i=1}^{n}\rho_i$$

定义 4(连通对象)

对于某个非聚类中心对象 x_i，距离其最近和次近的聚类中心分别为 x_n^* 和 x_m^*，若对象 x_i 满足：

$$\begin{cases}\rho_i\geqslant AD\\ d_{in}<\frac{1}{2}(1+\alpha)d_{nm}\\ d_{im}<\frac{1}{2}(1+\alpha)d_{nm}\end{cases}$$

则称节点 x 是以 x_n^* 为聚类中心的簇 n 和以 x_m^* 为聚类中心的簇 m 的连通对象。

本文认为，如果两个簇是连通的，那么这两个簇的边界处应该存在若干局部密度较高的数据对象，如图 2 所示，簇 C_1、C_2 是连通的，这两个簇的交界处存在若干数据对象，且局部密度较高。如图 2 中椭圆区域所示，通过比较两个簇交界区域内的高密度对象的数量可以评价两个簇的连通程度，具体如定义 3 所示。

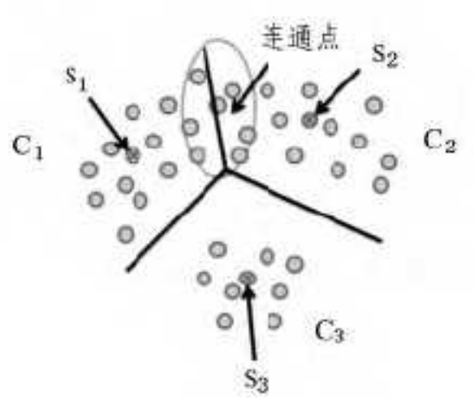


图 2　子簇的连通情况及连通对象

3 一种基于局部密度的聚类中心选取算法(ASCC)

算法对聚类中心提出的要求是：1)能够代表周围的一些对象；2)从整体上描述数据集的结构或者形状。分析第一个要求可知，聚类中心的局部密度应该是较高的，分析第二个要求可知这些聚类中心相互之间的距离较远。本文受文献[8]启发，提出一种基于局部密度的聚类中心选取算法，通过领域 d_c 内的邻居个数描述其局部密度，通过控制聚类中心之间的距离使其能反映数据集的全貌，算法的伪代码如下。

输入：S={$x_1, x_2, \cdots, x_n$}, dc
1)计算 ρ_i, i=1,2,…,n
2)降序排列 ρ_i, T_1=1
3)WHILE ρ_i>AD
4)　IF $d_{i,T_j}>2*d_c$, j=1,2,…,k−1
5)　　T_k=i
输出：T_k, k=1,2,…

通过定义可知，数据对象的局部密度就是其半径 dc 内的邻居个数，算法首先计算每个数据对象的局部密度，而后对其降序排序，将局部密度最大的一个对象作为第一个聚类中心，并随着局部密度降序从前往后搜索，第二个聚类中心是与第一个聚类中心的距离大于 $2dc$ 的对象，第三个聚类中心是与第一、第二个聚类中心的距离都大于 $2dc$ 的对象，以此类推，直至搜索到局部密度小于平均局部密度 AD。

分析上述算法可知，对局部密度降序排序是为了尽可能选局部密度大的点作为聚类中心，而控制聚类中心的距离是为了让这些聚类中心分散在数据集中，能尽可能地反映数据集的整体情况。之所以选取 $2dc$ 作为聚类中心之间距离的最低要求，是因为聚类中心已经能够代表其领域 dc 内的所有对象，$2dc$ 恰好相切。控制参数 AD 是为了控制最低局部密度要求，如果其局部密度小于 AD，那么它可能是噪声点或者不具有代表性，我们认为此时它不适合作为聚类中心。

4 一种基于聚类中心的快速聚类算法(IMCCA)

该算法将聚类划分两个阶段：第一阶段选取合理的聚类中心，以便于将较大的、延伸状的簇划分为若干小簇；第二阶段是将这些小簇通过小簇合并算法进行合并，得到最终结果。为了尽可能提高算法速度，采用了以下措施：1)在形成子簇时，直接将非聚类中心点划分到最近的聚类中心，这是因为聚类中心选取算法所取得的聚类中心已经十分接近 k-means 算法最后一次迭代的聚类中心，即正确的聚类中心；而且划分的这些簇并不是最终结果，通过小簇合并可以消除一些错误的划分。2)尽可能缩短小簇合并的时间，考虑在将非聚类中心划分到聚类中心的同时判断该非聚类中心是否为某两个簇的连通点，划分时，在寻找最近聚类中心的同时寻找次近的聚类中心，并判断它是否为这两个聚类中心代表的簇的连通对象。

在划分子簇结束时，子簇之间的连通点数量也已经计算完毕，这时只需将两两簇的连通点的数量合并成小簇，整个算法结束，因此小簇合并与划分小簇几乎是同时进行的。算法伪代码如下。

输入：$S=\{S_1,S_2,\cdots,S_n\}$，d_c，α，k

1) ASCC()；//调用 ASCC 算法，获取一组聚类中心 T_K，K=1，2…

2) FOR i=1：n

3) Calculation $\min1_{m\in T}(dist(i,m))$，$\min2_{n\in T}(dist(i,n))$

//计算离当前节点 i 最近、次近的聚类中心 m，n

4) $S_i.class=S_m.class$//划归到最近的聚类中心

5) IF $\alpha>=AD$ AND $d_{in}<1/2(1+\alpha)d_{nn}$ AND $d_{im}<1/2(1+\alpha)d_{nn}$

6) Similar(m，n)=+1

//以 m，n 为聚类中心的簇的连通对象数量加 1

7) sort Similar(i，j) //降序排列

8) WHILE K>k//根据连通对象的数量自大到小合并小簇

9) $T_i.class=T_j.class$

10) K=K−1

输出：$S_i.class$，i=1，2，…，n

5 参数分析与实验结果

实验环境：处理器为 Pentium® E2180 2.0GHz，内存大小为 2GB，硬盘 250GB，操作系统为 Windows Server2003，编程环境为 VC6.0，程序设计语言为 C++。

算法对参数 dc 并不敏感，这在文献[6，8]中都有讨论，这里不再讨论。图 3 所示的实验表面 dc 在很大调整范围内能够获得正确的聚类中心，dc 越大聚类中心数量越多，一般 dc 的取值能使得 AD 为数据点总数的 1%～2%。参数 α 控制连通区域大小，在[0.1，0.3]间取值，默认为 0.1。图 4 表明在不同程度重叠情况下，仅在 S_3 中误判簇 1、2 存在 1 个连通对象。

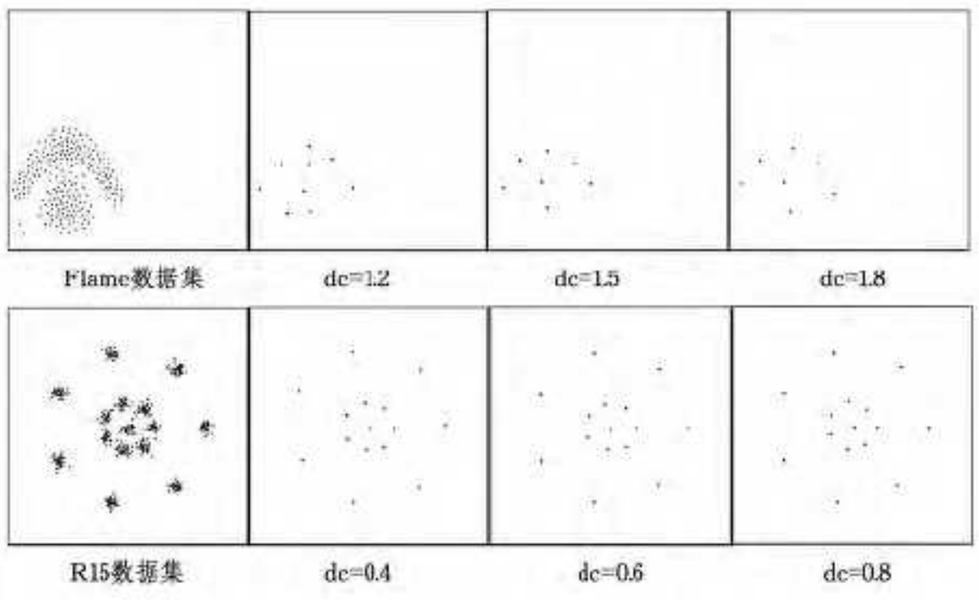


图 3 算法对参数 dc 的敏感性实验

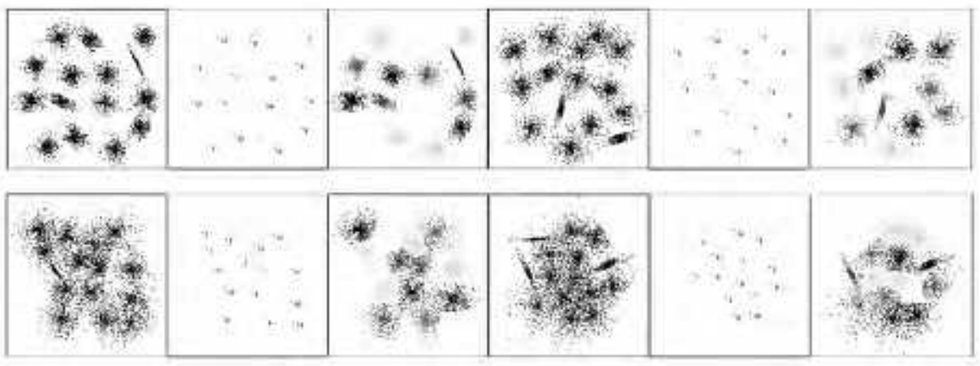


图 4 算法在 S_1、S_2、S_3、S_4 数据集上的聚类结果($dc=0.5$，$\alpha=0.1$)

为了对该算法的优异性做出评估，本文分别在球形数据集、非球形数据集以及标准数据集上进行实验，其中二维数据集均来自文献[9]。针对球形数据集在 R15 及 D31 数据集上进行实验，如图 5、图 6 所示。针对球形数据集，选取的聚类中心数量 K 等于最终聚类数 k，一次划分就得到最终聚类结果，未进行小簇合并。

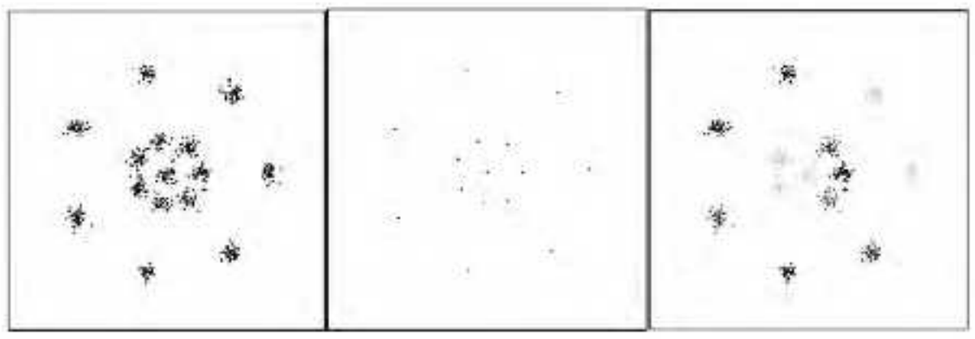


图 5 算法在 R15 数据集上的聚类结果($dc=0.5$)

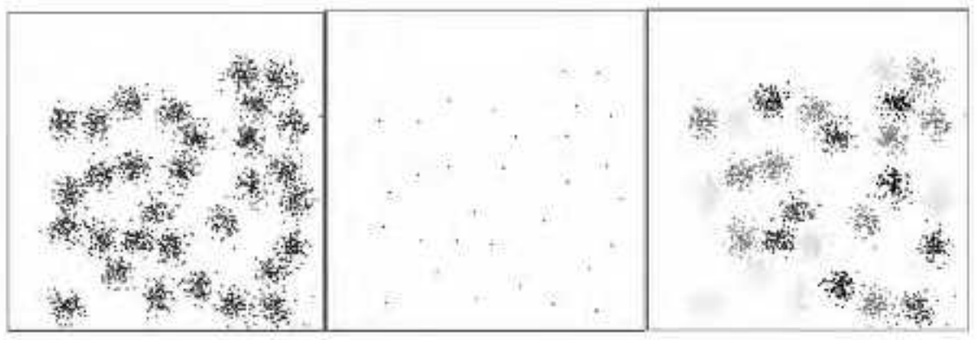


图 6 算法在 D31 数据集上的聚类结果($dc=1$)

针对非球形数据集，分别在 Aggregation、Flame 数据集上进行实验，结果如图 7、图 8 所示。实验结果表明，算法能有效聚类任意形状数据集，且能够适应具有噪声点、簇大小不均的情形。在小簇合并阶段，以 Aggregation 数据集为例，小簇之间的连通对象的数量分别为 Similar(0，1)=4，Similar(0，2)=4，Similar(0，12)=1，Similar(1，2)=2，Similar(1，12)=13，Similar(3，5)=12，Similar(4，9)=9，Similar(6，11)=9，其余簇之间的连通对象数量为 0。

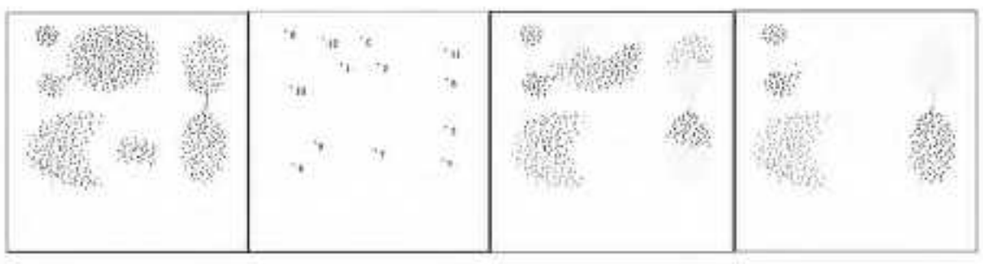


图 7 算法在 Aggregation 数据集上的聚类结果($dc=2.5$)

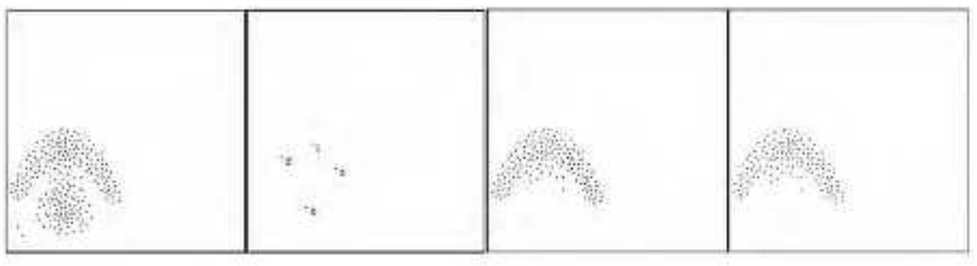


图 8 算法在 Flame 数据集上的聚类结果($dc=2$)

针对标准数据集，在 Iris 上进行聚类，从实验结果(见表 1)可以看出，该算法的精度高于 k-means 算法，聚类速度较快。

表 1 3 种算法在 Iris 数据集上的平均聚类精度及运行时间对比

算法	平均聚类精度(%)	平均运行时间(ms)
k-means	86.34	86
CURE	97.33	254
IMCCA	90.51	152

结束语 本文提出了一种基于聚类中心的快速聚类算法，其能较好地适应任意形状数据集；且算法是在初始聚类中心优化的 k-means 算法基础上的改进，保持了算法简单的特点，对噪声点、簇大小不均具有一定的鲁棒性。算法在获取聚类中心阶段花费的时间较长，针对大规模数据集，可以采用随机采样的方式获取聚类中心，即随机采集一个数据对象，判断局部密度是否达到要求且距离已有的聚类中心较远，判定是

(下转第 484 页)

体利用率是不一样的。当数据的负载不均匀时，会出现有的 CPU 处于空闲等待状态而有的 CPU 处于一直计算的状态。

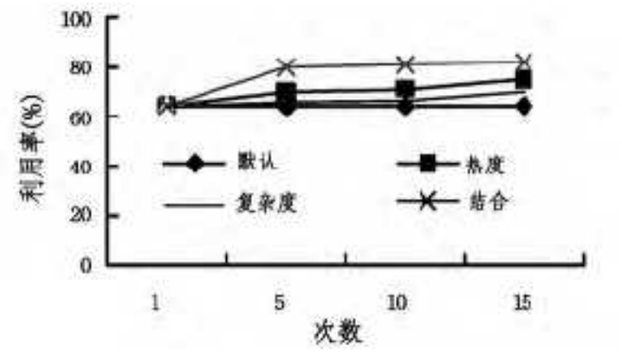


图 10 集群 CPU 利用率

图 10 显示了 4 种方式对相同的摄像头数据处理的情况下，集群 CPU 的利用率情况。本文实验通过周期性地取 ganglia 监控数据作为实验最终数据，如图所示，经过所设计的置换算法对数据进行置换置换，集群的整体 CPU 利用率提高了。因为数据更好地基于本地性的执行，不需要经过网络传输，所以能够更好地将任务及时分配给 CPU 计算，从而 CPU 的利用率提高了。因此上面 4 个部分的实验数据统计也正好验证了我们提出的置换算法可以更好地应用于大规模的实时视频流计算。

结束语 针对大规模的视频文件，我们一般利用 MapReduce 或者 Spark 来对其进行计算。为了提高数据的计算效率，使用 HDFS 来对数据进行存储，因为 HDFS 本身的数据存放机制能够让数据均衡地存放在集群中。但是视频文件自身较传统的文本文件、基于大小的块文件的计算消耗的资源往往不一样，甚至差异很大，这样上层分布式计算框架在对视频进行计算的时候，如果某个节点的视频数据负载度大，那么该节点的执行时间将比其他节点长，而系统的执行时间取决于最长任务的执行时间。采用有效的数据重分布算法能够很好地将热度高以及复杂度高的视频与热度低以及复杂度低的文件存储在一起，从而降低该节点的负载。实验结果表明，所提出的算法能够提高系统的计算效率，减少系统的计算总时间。

基于视频访问热度以及视频的复杂度能够在一定程度上解决大部分算法的负载均衡问题。但是本文所提算法是适用于需要进行物体的检测和提取的算法。而当有些算法不一定需要对物体进行检测以及提取的时候，视频的复杂度定义将不一样，未来将会继续探索这些算法及如何提高其计算效率。

参 考 文 献

[1] White T. Hadoop 权威指南[M]. 北京：清华大学出版社，2011：1-123

[2] Zaharia M, Chowdhury M, Das T, et al. Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing[C]// Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation. USENIX Association, 2012: 2-2

[3] Geng Chen-yao, et al. Distributed Video Processing Platform Based on Map Reduce[J]. Computer Engineering, 2012, 38(10): 280-283

[4] Zaharia M, Chowdhury M, Franklin M J, et al. Spark: cluster computing with working sets[C]// Proceedings of the 2nd USENIX Conference on Hot Topics in Cloud Computing. 2010: 1765-1773

[5] Zaharia M, Borthakur D, Sen Sarma J, et al. Delay scheduling: a simple technique for achieving locality and fairness in cluster scheduling[C]// Proceedings of the 5th European Conference on Computer Systems. ACM, 2010: 265-278

[6] Dean J, Ghemawat S. MapReduce: simplified data processing on large clusters[J]. Communications of the ACM, 2008, 51(1): 107-113

[7] Lin S H. An introduction to face recognition technology[J]. Informing Science, 2000, 3(1): 1-8

[8] Spark job scheduling[OL]. http://spark.apache.org/ docs/latest/job-scheduling.htm

[9] Borthakur D. HDFS architecture guide. HADOOP APACHE PROJECT[OL]. (2008). http://hadoop.apache.org/common/docs/current/hdfs design.pdf, 2008

[10] Kapil B S, Kamath S S. Resource aware scheduling in Hadoop for heterogeneous workloads based on load estimation[C]// 2013 Fourth International Conference on Computing, Communications and Networking Technologies (ICCCNT). IEEE, 2013: 1-5

[11] Orrite C, Bernues E, Gracia J J, et al. Face detection and recognition in a video sequence[C]// Defense and Security. International Society for Optics and Photonics, 2004: 94-105

[12] Bezerra A, Hernández P, Espinosa A, et al. Job scheduling for optimizing data locality in Hadoop clusters[C]// Proceedings of the 20th European MPI Users' Group Meeting. ACM, 2013: 271-276

(上接第 456 页)

否为聚类中心；小簇合并花费时间较短，但也存在一定问题，即小簇之间的边界并不一定在两个聚类中心的中点附近，下一步希望能够找到一种合理、简单的小簇合并方法。

参 考 文 献

[1] 于海涛，贾美娟，王慧强，等. 基于人工鱼群的优化 K-means 聚类算法[J]. 计算机科学，2012，39(12)：60-64

[2] 张建朋，陈福才，李邵梅，等. 基于密度与近邻传播的数据流聚类算法[J]. 自动化学报，2014，40(2)：277-288

[3] Lei Xiao-Feng，谢昆青，Lin Fan，等. 一种基于 K-Means 局部最优性的高效聚类算法[J]. 软件学报，2008，19(7)：1683-1692

[4] 张丽，崔卫东，邱保志，等. 基于划分与层次方法的混合聚类算法[J]. 计算机工程与应用，2010，46(16)：127-129

[5] 邱保志，陈本华，张真，等. 一种新的快速混合聚类算法[J]. 微电子学与计算机，2008，25(7)：78-80

[6] 李晓翠，孟凡荣，周勇，等. 一种基于代表点的快速聚类算法[J]. 南京大学学报(自然科学版)，2012，48(4)：504-512

[7] 马儒宁，王秀丽，丁军娣，等. 多层核心集凝聚算法[J]. 软件学报，2013(3)：490-506

[8] Rodriguez A, Laio A. Clustering by fast search and find of density peak[J]. Science, 2014(344): 1492-1496

[9] Clusteringdatasets [EB/OL]. (2008-04-15) [2015-03-05]. http://cs.joensuu.fi/sipu/datasets