计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.190900113

基于混合蛙跳算法的 K-mediods 聚类挖掘与并行优化

魏霖静[1]　宁璐璐[2]　郭　斌[3]　侯振兴[4]　甘诗润[1]
1 甘肃农业大学信息科学技术学院　兰州 730070
2 陕西科技大学轻工科学与工程学院　西安 710021
3 河海大学计算机与信息学院　南京 210094
4 南京大学信息管理学院　南京 210093

摘　要　为了降低 K-mediods 聚类算法的误差并提高并行优化的性能，将混合蛙跳算法运用于聚类和并行优化过程。在 K-mediods 聚类过程中，将 K-mediods 与聚类簇思想相结合，对各个聚类簇进行混合蛙跳算法优化，从而提高了大规模数据样本聚类的效率。针对多个聚类执行节点并行完成大规模样本 K-mediods 聚类时，混合蛙跳算法有效提高了加速比。实验证明，相比于普通的 K-mediods 聚类，基于混合蛙跳算法的 K-mediods 聚类优势明显，且处理大规模样本的加速比性能更优。

关键词：混合蛙跳算法；K-mediods 聚类；并行优化；聚类簇；加速比
中图法分类号　TP181

K-mediods Cluster Mining and Parallel Optimization Based on Shuffled Frog Leaping Algorithm

WEI Lin-jing[1], NING Lu-lu[2], GUO Bin[3], HOU Zhen-xing[4] and GAN Shi-run[1]
1 School of Information Science & Technology, Gansu Agriculture University, Lanzhou 730070, China
2 School of Light Industry Science & Engineering, Shaanxi University of Science and Technology, Xi'an 710021, China
3 College of Computer and Information, Hohai University, Nanjing 210094, China
4 School of Information Management, Nanjing University, Nanjing 210093, China

Abstract　In order to reduce the error of K-mediods clustering algorithm and improve the performance of parallel optimization, the shuffled frog leaping algorithm is applied to the clustering and parallel optimization process. In the K-mediods clustering process, K-mediods is combined with the clustering cluster idea to optimize the shuffled frog leaping algorithm for each cluster cluster, which improves the efficiency of large-scale data sample clustering, especially for multiple clusters. When the class execution nodes complete the large-scale sample K-mediods clustering in parallel, the shuffled frog leaping algorithm effectively improves the speedup ratio. It has been proved by experiments that the K-mediods clustering based on the shuffled frog leaping algorithm has obvious clustering advantages compared with the common K-mediods clustering, and the acceleration ratio performance of processing large-scale samples is better.

Keywords　Shuffled frog leaping algorithm, K-mediods clustering, Parallel optimization, Clustering cluster, Acceleration ratio

1　引言

聚类是数据挖掘的重要组成部分，广泛适用于各种数据平台下的非标签数据挖掘。各种差异化结构数据进入数据平台后，从这些大规模数据中提取有价值的数据成为了数据挖掘的关键。聚类挖掘正广泛用于各领域数据平台[1-6]：电商平台根据用户消费的历史数据进行聚类挖掘，为不同用户分类制定营销策略；旅游平台根据用户的历史搜索记录，为用户分类制定旅游套餐和线路推荐；在线社交平台根据用户访问情况，可为用户推荐更多的好友及定制服务。这些平台为不同用户提供的精准服务和推荐离不开聚类挖掘。

聚类挖掘较好地解决了非标签数据的挖掘问题；但是对于大规模数据挖掘而言，聚类算法的效率并不能满足数据平台实时推荐的需求。现在用于聚类分析的对象大部分是大数据平台数据或者云平台数据，其中大数据平台数据量大，而且种类差异较大。一般的聚类方法在数量不大且维度较小的情况下性能表现良好[7-13]，当数据量较大时则暴露出一些问题，特别是针对高维数据的聚类效果并不理想。K-medoids 作为

到稿日期：2019-09-16　返修日期：2019-10-21　本文已加入开放科学计划(OSID)，请扫描上方二维码获取补充信息。
基金项目：甘肃农业大学学科建设专项项目(GAU-XKJS-2018-257)；甘肃农业大学青年导师扶持基金项目(GAU-QDFC-2018-13)；兰州市科技计划项目(2019-1-31)；甘肃省教育厅创新基金(2020B-122)；国家自然科学基金(61063028,31560378)
This work was supported by the Discipline Construction Project of Gansu Agricultural University (GAU-XKJS-2018-257), Youth Tutor Support Fund Project of Gansu Agricultural University (GAU-QDFC-2018-13), Lanzhou Science and Technology Plan Project (2019-1-31), Gansu Provincial Department of Education Innovation Fund (2020B-122) and National Natural Science Foundation of China (61063028,31560378).
通信作者：魏霖静(wlj@gsau.edu.cn)

一种典型的基于划分方式的无监督聚类算法，有着聚类思想简单、聚类过程可行性高以及聚类时间复杂度接近线性等优点[14-17]。文献[18]提出了一种基于 PSO 的 K-Medoids 聚类检测的新方法。文献[19]提出了半径自适应的初始中心点选择 K-medoids 聚类算法。文献[20]使用 K-Medoids 聚类的近似最短距离计算来实现目标函数的优化。混合蛙跳算法是最近提出的较为先进的仿生优化算法，本文借助混合蛙跳算法并结合 K-mediods 的方法来完成数据聚类，并将其运用于大数据的数据挖掘过程中，以便在海量数据中更好地提取更有价值的数据。考虑到聚类效率，本文引入并行计算的思想，利用多节点来实现并行聚类，相比于单机聚类，其在大数据聚类效率上优势明显。

2 并行优化模型的数学模型

并行优化是数学优化方法的一种，一般用于多个多载体的大数据样本优化。在优化过程中，并行优化既要解决效率提升问题，又要解决多个载体进行协同优化的策略问题。

下面以常见的求解最大和最小化问题为目标函数，对并行优化模型进行数学描述[4-5]。

$$\max(\min)\ Z=f(x_1,x_2,\cdots,x_n)$$
$$\text{s.t.}\ \begin{cases} g_k(x_1,x_2,\cdots,x_n)\leqslant 0 \\ k=1,2,\cdots,m \end{cases} \tag{1}$$

下面以最小化问题 $\min f(x)$，$x\in[L,U]$为例进行说明，其中 L，U 分别为 x 的下界和上界[6]。

假设有可行解 $x^*\in[L,U]$，且有集合 $B(x^*)$，$B(x^*)$满足式(2)[7]。

$$B(x^*)=\{x\mid \|x-x^*\|<\delta\} \tag{2}$$

其中，δ 为逼近差，可以根据实际应用场景的误差精度进行参考设定。我们只需要采用算法从 $f(x)$函数的下端不断逼近，则可以求解 $f(x)$的最小值。那么，问题即转化为求解式(3)[8]。

$$f(x^*)\leqslant f(x),\forall x\in[L,U]\cap B(x^*) \tag{3}$$

当 $x^*\in[L,U]$满足式(3)时，即可获得最优解。最优解的求解方法需要通过智能优化算法完成。以最大和最小化问题为目标函数是针对多节点的聚类任务并行优化，其目的是最小化整个聚类运行的时间；而 K-mediods 聚类为具体采用的聚类挖掘。

3 基于混合蛙跳算法的 K-mediods 聚类

3.1 混合蛙跳算法

在聚类优化过程中，将除了中心点之外的样本点和中心点位置关系的和作为混合蛙跳算法的适应度函数，适应度函数值越大，表明适应度越好。在第 $t+1$ 次计算迭代过程中，运用了第 t 次迭代后的结果，即适应度函数最大值的青蛙 $X_b(t)$和最小值的青蛙 $X_w(t)$[9-10]。为了保证群组内的青蛙可以朝着适应度函数值最大的青蛙靠拢，从适应度函数值最小的青蛙开始不断移动，移动方法为[11]：

$$\Delta_w(t)=rand()(X_b(t)-x(t)) \tag{4}$$
$$X_w(t+1)=X_w(t)\Delta_w(t),R_{\min}\leqslant\Delta_w(t)\leqslant R_{\max} \tag{5}$$

若 $t+1$ 时刻求解的 $X_w(t+1)$值比 $X_w(t)$大，即具有更好的适应度，那么用 $X_w(t+1)$替换 $X_w(t)$；反之，继续执行式(4)和式(5)。关于青蛙移动步长的问题，可引入步长因子 C，那么第 k 只青蛙在第 i 次移动的距离为：

$$d_i=rand()\times(X_b^k-X_w^k)\times C \tag{6}$$

步长因子为[12]：

$$C=C_{\min}+i_{\text{now}}/G_{\text{global}}\times(C_{\max}-C_{\min}) \tag{7}$$

其中，$C_{\min}$和 $C_{\max}$分别为当前群组内青蛙的最小移动步长和最大移动步长，可以根据实际情况设定；G_{global}为群组内所有青蛙的适应度值之和；i_{now}为当前时刻青蛙移动的次数。

当群组内所有青蛙的适应度值更接近 $X_b(t)$，且误差在设定的阈值内时，算法停止迭代，输出当前时刻青蛙的分布图作为最优解。

3.2 K-mediods 聚类树的混合蛙跳优化

K-mediods 算法是一种常用的聚类方法，很多聚类模型都是基于此算法或其改进算法。K-mediods 聚类算法的主要流程如图 1 所示[14-15]。

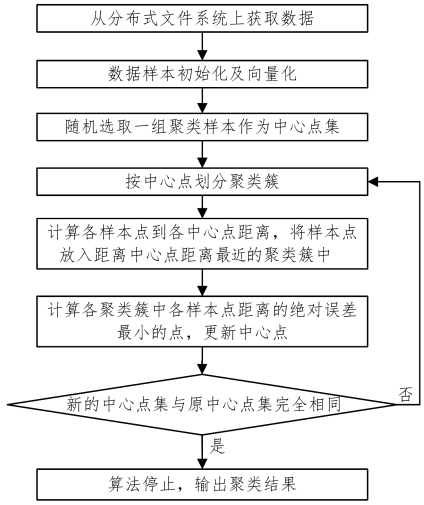

图 1 K-mediods 聚类算法的流程图

Fig. 1 Flow chart of K-mediods clustering algorithm

划分聚类簇之后，每个样本点都需要和所有聚类簇进行距离运算并比较，同时还需要对聚类簇内所有的点进行举例运算，因此该聚类算法的聚类时间较长。为了解决该问题，将混合蛙跳算法运用于 K-mediods 聚类中。

下面对聚类簇模型进行数学描述。设有 n 个 d 维数据样本 $x(x_1,x_2,\cdots x_d)$，则簇的聚类特征 CF 为[16]：

$$CF=\langle n,LS,SS\rangle \tag{8}$$

其中，LS 和 SS 的计算方法为：

$$LS=\sum_{i=1}^{n}x_i \tag{9}$$
$$SS=\sum_{i=1}^{n}x_i^2 \tag{10}$$

那么，聚类簇中心 x_0 的计算方法为：

$$x_0=\frac{\sum_{i=1}^{n}x_i}{n}=\frac{LS}{n} \tag{11}$$

根据式(12)，可以计算聚类簇中所有点到中心点 x_0 的均值距离。

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

除了簇中所有点相对于中心点的均值距离，簇中任意两点之间的均值距离也可以参照式(13)进行计算[16]。

$$D=\sqrt{\frac{\sum_{i=1}^{n}\sum_{j=1}^{n}(x_i-x_j)^2}{n(n-1)}}=\sqrt{\frac{2nSS-2LS^2}{n(n-1)}} \tag{13}$$

设参数 α 和 β 分别描述簇内样本点的归属程度和聚合度，那么混合蛙跳算法的适应度为：

$$g(x)=\sum_{i=1}^{n}\sum_{j=1}^{d}\frac{x_{id}}{\alpha R+\beta D} \tag{14}$$

其中，x_{id} 表示第 i 个样本的第 j 个属性。

启动混合蛙跳算法，求解 $g(x)$ 的最大值。

$$G(X_b)=Max(g(x)) \tag{15}$$

得到的 $G(X_b)$ 最大值即为蛙跳算法的最优解，从而得到聚类结果。

4 实例仿真

本节通过实例仿真来验证混合蛙跳算法在 K-mediods 中的聚类性能以及在并行优化过程中的优势。仿真样本数据集均来自于 UCI 网站公开数据集。为了充分验证混合蛙跳算法在聚类及并行优化中的性能，以传统的 K-mediods 算法作为参照对象，分别对聚类质量和并行优化的运行时间等相关指标进行对比。

4.1 聚类性能仿真

为了验证混合蛙跳算法在 K-mediods 中的聚类效果，分别采用 Matlab 对 UCI 的 5 个不同数据集进行 K-mediods 聚类和基于混合蛙跳的 K-mediods 聚类，并将两种算法的聚类准确率进行比较。选取聚类中心 $K=10$，统计结果如表 1 所列。

表 1 聚类准确率

Table 1 Clustering accuracy

算法	数据集	准确率/%
K-mediods	Flowers	74.238
	Wine	76.311
	iris	73.921
	seeds	81.007
	glass	78.329
基于混合蛙跳的 K-mediods	Flowers	75.173
	Wine	77.325
	iris	75.291
	seeds	81.467
	glass	82.536

从表 1 可知，相比于传统的 K-mediods，基于混合蛙跳的 K-mediods 聚类的准确率更高（虽然两者在同类数据集下的聚类准确率差异并不是很大，只有在 glass 数据集上两者的准确率差异达到了 4%，其他样本集合上的准确率差异均在 1%左右）。

4.2 并行优化性能仿真

从 UCI 数据集中选取不同大类的数据集合，其中包含了 Flowers，Wine，seeds 等不同大类的数据样本；然后将这些不同大类的样本进行混合，分别得到两个数据集合大小分别为 27.02 GB 和 102.11 GB 的样本文件。在设置并行优化实验环境时，选择 30 个节点（1 个控制节点和 29 个分布式节点）参与聚类运算，且 30 个节点的硬件配置相同，分别对聚类中心个数 K 为 20，30 和 50 进行实例仿真。

为了充分对比单机聚类和多节点并行聚类的差异，引入两个效率参数 λ_1 和 λ_2。效率参数的计算方法为：

$$\lambda_1=(1-\frac{t_2}{t_0})\times 100\% \tag{16}$$

$$\lambda_2=(1-\frac{t_2}{t_1})\times 100\% \tag{17}$$

其中，t_2 为基于混合蛙跳算法的多节点 mediods 聚类时间，t_1 为多节点并行的 K-mediods 聚类时间，t_0 为单机基于混合蛙跳算法的 K-mediods 聚类时间。

4.2.1 算法的运行时间

设定样本聚类准确率阈值为 75%，分别对样本容量为 37.02 GB 和 102.11 GB 的两个样本文件进行聚类，参与并行计算的节点数为 10 个。不同算法的运行时间统计结果如表 2 所列。从表 2 可以看出，随着样本容量的增加，聚类运算时间不断增加；而且随着聚类中心点数的增加，类别增多，聚类运算时间也增加。

表 2 不同算法的运行时间

Table 2 Running time of different algorithms

样本容量/GB	K 值	算法	运行时间/s
37.02	20	单机 K-mediods	1172
		多节点 K-mediods	527
		基于 PSO 的多节点 K-mediods	498
		基于混合蛙跳的多节点 K-mediods	473
	30	单机 K-mediods	1774
		多节点 K-mediods	632
		基于 PSO 的多节点 K-mediods	564
		基于混合蛙跳的多节点 K-mediods	531
	50	单机 K-mediods	2447
		多节点 K-mediods	728
		基于 PSO 的多节点 K-mediods	642
		基于混合蛙跳的多节点 K-mediods	593
102.11	20	单机 K-mediods	4374
		多节点 K-mediods	837
		基于 PSO 的多节点 K-mediods	748
		基于混合蛙跳的多节点 K-mediods	678
	30	单机 K-mediods	4922
		多节点 K-mediods	856
		基于 PSO 的多节点 K-mediods	762
		基于混合蛙跳的多节点 K-mediods	681
	50	单机 K-mediods	5122
		多节点 K-mediods	894
		基于 PSO 的多节点 K-mediods	740
		基于混合蛙跳的多节点 K-mediods	701

为了更直观地描述聚类过程采用多节点并行优化的效果，根据表 2 的统计结果和式(16)－式(17)，得到效率参数值，结果如表 3 所列。从表 3 可以看出，随着样本容量和 K 值的增加，效率参数 λ_1 快速增加，相比于单机基于混合蛙跳算法的 K-mediods 聚类，10 个参与聚类的分布式节点并行优化的优势更加明显，当样本容量为 102.11 GB 且聚类中心个数为 50 时，聚类效率提高了 86.31%。而随着 K 值和样本容量的增加，效率参数 λ_2 也在缓慢增加，即使在相同的多节点并行聚类的条件下，相比于普通的 K-mediods 聚类，采用蛙跳混合算法的 K-mediods 聚类在运行效率上也有所提升，因此本文算法更具优势。

表 3 并行优化效率参数表

Table 3 Parallel optimization efficiency parameters

样本容量/GB	K 值	λ_1/%	λ_2/%
37.02	20	59.64	10.24
	30	70.06	15.98
	50	75.76	18.54
102.11	20	84.49	18.99
	30	86.16	20.44
	50	86.31	21.58

4.2.2 扩展性能和加速性能

为了更充分地验证多节点并行聚类优化的性能，将参与并行聚类的节点数进行动态变化，以验证节点数对聚类时间的影响。在仿真过程中，首先对单机基于混合蛙跳的 K-mediods 聚类进行仿真，然后设置节点数分别为 10，20 和 30，进行基于混合蛙跳的 K-mediods 聚类仿真，设定聚类准确率阈值为 75%，聚类中心个数为 30。当运算迭代终止时，记录运行时间，结果如表 4 所列。

为了对基于混合蛙跳的 K-mediods 聚类与单机聚类的速度进行比较，求解加速比，其计算公式为：

$$S=\frac{T_a}{T_m} \tag{18}$$

其中，T_a 及 T_m 分别表示基于单机和多节点的混合蛙跳的 K-mediods 聚类的计算时间。

表 4 并行优化的运行时间和加速比

Table 4 Runtime and acceleration ratio for parallel optimization

样本容量/GB	节点数	算法	运行时间/s	加速比
37.02	10	单机	1 767	3.32
		多节点	532	
	20	单机	1 767	10.27
		多节点	172	
102.11	30	单机	1 767	16.51
		多节点	107	
	10	单机	4 918	7.26
		多节点	677	
	20	单机	4 918	26.02
		多节点	189	
	30	单机	4 918	43.52
		多节点	113	

从表 4 可以看出，当参与聚类运算的节点数不断增加时，基于混合蛙跳的多节点 K-mediods 算法的运行时间不断减少，表明并行聚类的扩展性能好。

结束语 本文采用基于混合蛙跳算法的 K-mediods 聚类算法实现数据聚类，结合多节点的并行优化运算，实现了大数据平台数据的高效聚类。相比于传统的 K-mediods 聚类，本文算法的准确率更高，运行时间更短；结合多节点并行优化聚类，运算效率更高，应用价值更广泛。

参 考 文 献

[1] QU J. Research on Intelligent Parallel Clustering Method for Large Data in Virtual Environment[J]. Computer Measurement and Control, 2017, 25(6): 257-260.

[2] HONG Y H. Parallel computation based on MPI bee swarm K-means clustering algorithm [J]. Computer Engineering and Design, 2017, 38(12): 3339-3343.

[3] ZHAO B W, XU H. Parallel MRACO-PAM clustering algorithm based on MapReduce [J]. Computer Engineering and Science, 2017, 39(10): 1801-1806.

[4] ZHU F L, ZENG B, CAO J. Parallel optimization and implementation of SLAM algorithm based on particle filter [J]. Journal of Guangdong University of Technology, 2017, 34(2): 92-96.

[5] OUYANG C J, LIU C X, MING L, et al. An OMP Steganographic Algorithm Optimized by SFLA[J]. International Journal of Pattern Recognition & Artificial Intelligence, 2017, 31(1): 496-505.

[6] TANG Z, LIU K, XIAO J, et al. A parallel k-means clustering algorithm based on redundance elimination and extreme points optimization employing MapReduce[J]. Concurrency & Computation Practice & Experience, 2017, 29: e4109.

[7] LAI Z, FENG X, YU H, et al. A Parallel Social Spider Optimization Algorithm Based on Emotional Learning[J]. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 2018, 12(9): 1-12.

[8] LIU P, TENG J Y, DING E J, et al. Large-scale text K-means parallel clustering algorithm based on Spark [J]. Chinese Journal of Information, 2017, 31(4): 145-153.

[9] SOODI H A, VURAL A M. STATCOM Estimation Using Back-Propagation, PSO, Shuffled Frog Leap Algorithm, and Genetic Algorithm Based Neural Networks[J]. Computational Intelligence & Neuroscience, 2018, 2018(1): 1-17.

[10] WANG N, GAO X J. A novel differential hybrid frog jump algorithm[J]. Computer System Applications, 2017, 26(1): 196-200.

[11] LU Y H, CHEN J H. Application of hybrid frog leaping algorithm in text classification feature selection optimization [J]. Modern Library and Information Technology, 2017, 1(1): 91-101.

[12] ZHAO H X, CHANG X G. An improved hybrid leaping frog algorithm [J]. Journal of Lanzhou Jiaotong University, 2017, 36(1): 51-56.

[13] TANG X Y, ZHANG X Z, ZHAO Y A. Big Data Clustering Mining Based on Swarm Intelligence Algorithm [J]. Journal of Chongqing University of Technology (Natural Science), 2019, 33(4): 128-133.

[14] ZHOU Z, SI G, CHEN J, et al. A novel method of transformer fault diagnosis based on k-mediods and decision tree algorithm [C]// International Conference on Electrical Materials & Power Equipment. 2017.

[15] WANG Q, YANG J, ZHANG S S. A K-mediods clustering algorithm based on improved Drosophila optimization[J]. Computer Technology and Development, 2018, 28(12): 23-28.

[16] LIU J P, ZHANG W X, TANG Z H, et al. Adaptive Network Intrusion Detection Based on Fuzzy Rough Set Attribute Reduction and GMM-LDA Optimal Cluster Feature Learning [J]. Control and Decision, 2019, 34(2): 22-30.

[17] WANG Y, WANG L F, RAO Q F, et al. Radius-Adaptive on Selection of Initial Centers in K-Medoids Clustering [J]. Journal of Chongqing University of Technology (Natural Science), 2017, 31(2): 95-101.

[18] KHATAMI A, MIRGHASEMI S, KHOSRAVI A, et al. A new PSO-based approach to fire flame detection using K-Medoids clustering[J]. Expert Systems with Applications, 2017, 68(C): 69-80.

[19] WANG Y, WANG L F, RAO Q F, et al. K-medoids clustering algorithm for initial center point selection based on radius adaptation[J]. Journal of Chongqing University of Technology (Natural Science Edition), 2017, 31(2): 95-101.

[20] AGARWAL S, MEHTA S. Approximate Shortest Distance Computing Using k-Medoids Clustering[J]. Annals of Data Science, 2017, 4(4): 547-564.

WEI Lin-jing, born in 1977, Ph.D, professor, master supervisor, is a member of China Computer Federation. Her main research interests include intelligent computing, algorithm application research and image analysis.