云计算环境下高复杂度动态数据的增量密度快速聚类算法研究

陈赣浪[1] 颜飞龙[2] 潘家辉[1]

(华南师范大学软件学院 广东 南海 528225)[1] (华南师范大学教育科学学院 广州 510631)[2]

摘 要 针对传统的聚类算法存在开销大、聚类质量差、聚类速度慢等问题,提出一种新的云计算环境下高复杂度动态数据的增量密度快速聚类算法。首先,依据密度对云计算环境下高复杂度动态数据进行聚类,从数据空间中找到部分子空间,使得数据映射至该空间后可产生高密度点集区域,将连通区域的集合看作聚类结果;其次,通过DBSCAN算法进行增量聚类,并对插入或删除数据导致的原聚类合并或分裂进行研究;最后,在更新的过程中通过改变核心状态数据的邻域中含有的全部核心数据进行处理,从插入或删除数据两方面进行增量聚类分析。实验结果表明,所提算法开销低、聚类速度快、聚类质量高。

关键词 云计算环境,高复杂度,动态数据,增量密度,快速聚类

中图法分类号 TP311 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2018.02.049

Study on Fast Incremental Clustering Algorithm for High Complexity Dynamic Data in Cloud Computing Environment

CHEN Gan-lang[1] YAN Fei-long[2] PAN Jia-hui[1]

(School of Software, South China Normal University, Nanhai, Guangdong 528225, China)[1]

(School of Education, South China Normal University, Guangzhou 510631, China)[2]

Abstract In order to solve the problems that the traditional clustering algorithm has the disadvantages of high cost, poor clustering quality and slow clustering speed, this paper proposed a new fast clustering algorithm based on incremental density of high complexity dynamic data in cloud computing environment. First of all, on the basis of density under the environment of high complexity of dynamic data clustering in cloud computing, this algorithm finds some subspace from the data space. The data mapped to the space area can produce high density point set, and the set of connected regions is regarded as the clustering results. Secondly, it executes incremental clustering by DBSCAN algorithm, and studies the original clustering merger or split caused by inserting or deleting data. Finally, by dealing with all the core data in the neighborhood of changing the core status in the process of updating, the incremental clustering is analyzed from two aspects of inserting or deleting data. The experimental results show that the proposed algorithm has the characteristics of low cost, fast clustering speed and high clustering quality.

Keywords Cloud computing environment, High complexity, Dynamic data, Incremental density, Fast clustering

1 引言

当前,云计算技术因具有处理速度快、实时性高的优势而被广泛应用[1-2]。人们对云计算环境下不同应用的需求越来越大,云计算环境下高复杂度的动态数据通常以快速数据流的形式被接收,导致海量递增数据的处理编程成为云计算技术的瓶颈[3-5]。聚类算法是一种有效的数据处理算法,研究云计算环境下高复杂度动态数据的聚类算法具有重要意义[6-7]。

文献[8]提出一种采用双层架构的数据聚类算法。该算法把数据聚类划分成在线微聚类与离线宏聚类两个层次,通过在线微聚类对在线数据特征进行采集,获取概要数据信息;通过离线宏聚类对用户请求进行处理,依据概要数据得到聚类结果。该方法针对数据量较少的情况有很好的聚类效果,但不适用于数据量较多的情况。文献[9]提出一种实时数据流聚类D-Stream算法,把数据空间分割成网格,将接收到的数据映射至网格中,利用既定阈值对网格进行划分,通过网格映射空间中的数据点数量对网格密度进行描述。该算法的计算复杂度较低,但容易出现数据丢失的情况。文献[10]通过EXCC算法(有优势且完整的聚类算法)进行聚类,首先求出密度阈值,同时把密集网格及其最近的数据置于网格池中,之

到稿日期:2017-01-23 返修日期:2017-02-10 本文受国家自然科学基金青年科学基金项目(61503143),广东省自然科学基金博士科研启动项目(2014A030310244)资助。

陈赣浪(1979-),女,硕士,讲师,主要研究方向为软件工程,E-mail:chenganlang1643@163.com(通信作者);**颜飞龙**(1975-),男,硕士,主要研究方向为电子商务;**潘家辉**(1982-),男,博士,副教授,主要研究方向为脑机接口、机器学习、数字医学。

后的每次聚类均从该网格池中取对象。该方法实现过程简单,但聚类质量不佳,且所需内存空间大。

针对上述算法的弊端,提出一种新的云计算环境下高复杂度动态数据的增量密度快速聚类算法,依据密度对云计算环境下高复杂度动态数据进行聚类,通过 DBSCAN 算法进行增量聚类。实验结果表明,所提算法开销低、聚类速度快、聚类质量高。

2 云计算环境下高复杂度动态数据的增量密度快速聚类算法

2.1 云计算环境下高复杂度动态数据密度聚类

本节首先通过密度对云计算环境下高复杂度的动态数据进行聚类。该算法的基本思想如下:从云计算的数据空间中找到部分子空间,该空间中点的密度高于其相邻空间,使得数据映射至该空间后,可产生高密度点集区域,此时连通区域的集合即为聚类结果[11-13]。数学描述如下。

假设 $D=\{A_1,A_2,\cdots,A_n\}$ 为云计算环境下的 n 个有界定义域,则 $S=A_1\times A_2\times\cdots\times A_n$ 即为 n 维空间,将 $A_1,A_2,\cdots,A_n$ 看作 S 的不同维。

算法的输入为云计算环境下 n 维空间中的点集,假设存在 m 个点,则 $V=\{v_1,v_2,\cdots,v_m\}$,其中 $v_i=\{v_{i1},v_{i2},\cdots,v_{in}\}$,$v_i$ 的第 j 个分量 $v_{ij}\in A_j$。

通过抽样分析将空间 S 的各维划分成几个区间,从而把整个云计算空间分割成有限个不相交的类矩形单元,用 $\{a_1,a_2,\cdots,a_n\}$ 进行描述,其中 $a_i=[l_i,h_i)$ 为前闭后开区间,是该维某区间的大小。

一个 $v_i=\{v_{i1},v_{i2},\cdots,v_{in}\}$ 落入一个区间 $A=\{a_1,a_2,\cdots,a_n\}$,如果针对所有 a_i,$l_i\leqslant v_i<h_i$ 均成立,则某单元格 u 的选择率 $select(u)$ 可通过式(1)求出。

$$select(u)=\frac{dd}{kd} \tag{1}$$

其中,dd 用于描述单元格中的点数;kd 用于描述整个云计算环境下的点数。

针对云计算环境下用户的输入参数 τ,在 $select(u)>\tau$ 的情况下称该单元 u 是密集的。为了便于分析,只要该单元格中的点数超过 M(M 为 τ 与总点数的乘积),则认为该单元是密集的。

若一个记录集 S 为云计算环境下 n 维空间中的一个聚类,则针对某 $n-1$ 维子空间,对记录集 S 进行映射后仍可构成一个聚类。依据该理论,本节从低维到高维寻找密集子空间,云计算环境下高复杂度动态数据的聚类过程如下。

(1)令 $n\rightarrow1$,找到云计算环境下所有的一维密集子空间数据,用 D_1 描述构成的集合,同时把不密集的子空间的数据属性标记成 0,以降低计算量。

(2)通过下述过程,使 n 维密集子空间数据集合 D_n 形成 $n+1$ 维的候选密集子空间数据集合 D_{n+1}。

1)针对第 $n+1$ 维数据空间,求出其所有空间的一维子空间密度。假设获取 m 个密集的一维数据空间,维数用 $b_1,b_2,\cdots,b_m$ 进行描述,$b_i\in A_{n+1}$;

2)针对 D_n 中的所有元素,假设 D_n 中某元素为 $(a_1,a_2,\cdots,a_n)$,依据前 $n+1$ 维求出的 m 个密集子空间 $b_1,b_2,\cdots,b_m$ 形成新的 $n+1$ 维候选密集数据子空间:$(a_1,a_2,\cdots,a_n,b_1)$,$(a_1,a_2,\cdots,a_n,b_2)$,$\cdots$,$(a_1,a_2,\cdots,a_n,b_m)$;

3)如果 D_{n+1} 是空集,则进行步骤 4);否则,求出候选单元格中的 $select(u)$ 值,删除非密集数据单元格,用 D_{n+1} 对其进行搜索,令 $n\rightarrow n+1$,重新进行上述过程;

4)算法结束,获取包含聚类的数据子空间。

依据香农定律,在本节聚类算法中对每个维的熵进行计算,以反映密度集中状态:

$$E_i=-\sum_{j=1}^{k}\frac{C_{ij}}{R_i}\log_2\frac{C_{ij}}{R_i} \tag{2}$$

其中,E_i 用于描述第 i 维的熵;R_i 用于描述第 i 维的总点数;C_{ij} 用于描述第 i 维第 j 个区间中的点数;k 用于描述第 i 维经分割后的区间个数。熵值越大,说明分布越均匀,反之则说明集中度越高。因此可基于熵的特性,依据熵值对维的次序进行选择。

最后将标记是密集的数据子空间进行连通性检测,所有连通集构成一个聚类。

2.2 增量聚类分析

本节通过 DBSCAN 算法进行增量聚类,因为 DBSCAN 算法依据的是密度的特性,而云计算环境下一个高复杂度动态数据的插入或删除仅可对其相邻空间的密度产生影响,所以对该算法从插入或删除数据两方面进行增量聚类分析。

依据对插入或删除数据导致的原聚类合并或分裂的研究,在更新的过程中通过改变核心状态数据的邻域中含有的全部核心数据进行处理。

根据式(2)描述的密度集中状态,用 D 描述云计算环境下的高复杂度动态数据集,用 p 描述插入或删除的数据,则有:

$$UpdSeeds_{Ins}(p)=\{q\mid q\in D\cup\{p\},q\in N_{Eps}(o)\} \tag{3}$$

$$UpdSeeds_{Del}(p)=\{q\mid q\in D-\{p\},q\in N_{Eps}(o)\} \tag{4}$$

因为插入一个数据 p 时,仅对其邻域的密度产生影响,所以改变核心状态的对象一定在 $N_{Eps}(p)$ 中。

2.2.1 数据的插入

在云计算环境下,当插入数据 p 时,对数据库中数据和插入表的数据进行检索,获取 $UpdSeeds_{Ins}(p)$。数据插入的处理过程如下:

针对云计算环境下所有插入数据 p,如果其没有被处理,且 $UpdSeeds_{Ins}(p)$ 为空同时符合噪声条件,则将 p 标记为噪声;

如果 $UpdSeeds_{Ins}(p)$ 中含有的核心数据无法与聚类的核心数据密度可达,则建立一个关于 p 的新聚类,同时继续对改变核心状态的对象 o 的 $N_{Eps}(o)$ 中没有处理的数据进行处理;

如果 $UpdSeeds_{Ins}(p)$ 中含有的核心数据与 p 的关系符合纳入聚类的情况,则将 p 划分至该聚类中;

如果 $UpdSeeds_{Ins}(p)$ 中含有的核心数据不属于相同的聚

类，同时互相密度可达，则将该聚类合并。

对于数据的处理顺序，算法并非按照插入顺序进行处理，而是按照插入数据与已有数据密度可达的关系进行处理的，所以可快速找到新聚类中的数据。针对合并的状况，算法可快速找到聚类之间的核心数据，提高了合并效率。因为分析的是更新数据的空间密度，所以该算法仅需对所有更新对象检索一次，避免了重复检索的现象。

2.2.2　数据的删除

在云计算环境下，除了对数据进行插入操作，还需删除高复杂度动态数据所含有的聚类标识号，降低标识号对增量密度聚类速度造成的干扰。针对某一聚类在很大程度上会分裂成两个或多个的情况，仅需检索一次高复杂度动态数据目标对象，就能获取最终结果，其基本思想如下：当某数据被删除后可能导致分裂时，不立即检查 $UpdSeeds_{Del}(p)$ 中数据的密度可达关系，首先对密度不可达的点进行记录，待找到全部可能导致分裂的核心数据后，再对其密度可达对象进行检索，获取该类高复杂度动态数据的密度可达关系。详细过程如下：

(1)从云计算环境下的删除列表中选择聚类标识一样的数据对象，用 D' 进行描述。

(2)针对所有在 D' 中的数据对象 p，如果 $UpdSeeds_{Del}(p)$ 非空，同时 $UpdSeeds_{Del}(p)$ 中的数据无法彼此直接密度可达，则针对所有无法密度可达的数据集，选择代表点添加至 d_link；如果数据 p 在 d_link 中，则通过 p 的一个直接密度可达核心数据对 p 进行替换，否则从 d_link 中删除 p，将与 p 直接密度可达的数据看作噪声，并在云计算环境下删除数据 p。

(3)当 d_link 非空，$p=pop(d\_link)$，$Q=\{p\}$ 时，优先对 p 的全部密度可达数据 q 进行检索，如果 q 在 d_link 中，则将 q 删除，令 $Q=Q+\{q\}$；如果 d_link 为空，则停止检索，Q 中含有的全部可能导致分裂的核心数据及与该数据密度可达的其他数据形成分裂后的一个聚类；如果 d_link 非空，继续对下一个数据进行处理，获取分裂后的另一个聚类。

(4)将不包含在分裂后子聚类中的原聚类数据看作噪声。

(5)转至步骤(1)，继续处理下一组数据。由于删除 p 后，与其相邻的空间中的数据无法直接密度可达的情况很少发生，因此 d_link 中的数据量 m 远小于云计算环境下数据库中的数据量 N，故该算法能够被有效地实现。

3　实验结果分析

为了验证本文算法的有效性，进行相关的实验分析。实验将 D-Stream 算法和 EXCC 算法作为对比，从性能开销、聚类速度和聚类精度 3 方面进行验证。本文通过 C 语言在 VC6.0 环境下进行。

下面定义本文算法相对于其他算法的性能“加速因数”，它为本文算法和其他算法性能开销的比值：

$$SpeedupFactor=\frac{Cost_M(n+num_{upd}(per_{ins}-per_{del}))}{Cost(num_{upd})} \quad (5)$$

其中，num_{upd} 用于描述数据更新数量；per_{ins} 用于描述数据插入数量与数据更新数量之比；per_{del} 用于描述数据删除数量与数据更新数量之比。图 1 和图 2 分别给出了随着处理数据量的逐渐增加，本文算法相对于 D-Stream 算法和 EXCC 算法的加速因数比较结果。

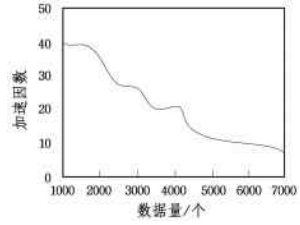


图 1　本文算法相对于 D-Stream 算法的加速因数比较结果

Fig. 1　Comparison results of the acceleration factor between this algorithm and D-Stream algorithm

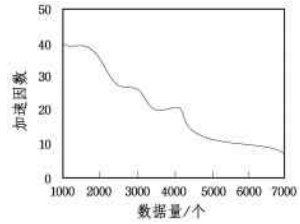
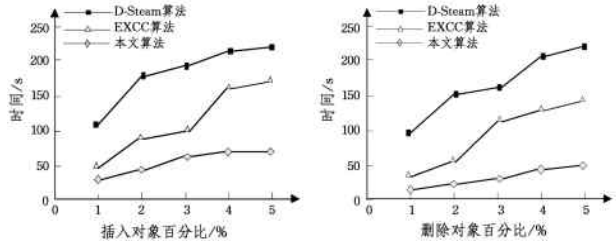


图 2　本文算法相对于 EXCC 算法的加速因数比较结果

Fig. 2　Comparison results of the acceleration factor between this algorithm and EXCC algorithm

分析图 1 和图 2 可以看出，随着处理数据量的逐渐增加，本文算法的加速因数相对于 D-Stream 算法和 EXCC 算法的加速因数均呈下降趋势，说明本文算法的开销明显低于 D-Stream 算法和 EXCC 算法。

随着插入数据量和删除数据量的逐渐增加，分别采用本文算法、D-Stream 算法和 EXCC 算法对数据进行聚类处理，3 种算法的处理时间的比较结果如图 3 所示。



(a)插入数据时 3 种方法聚类时间的比较结果　　(b)删除数据时 3 种方法聚类时间的比较结果

图 3　3 种方法聚类时间的比较结果

Fig. 3　Comparison results of clustering time of three methods

分析图 3 可知，当插入数据和删除数据的百分比相同时，删除数据比插入数据所需的时间更少。在更新数据量较少的情况下，本文算法的速度优势非常明显，且随着更新数据量的逐渐增加，相比于 D-Stream 算法和 EXCC 算法，本文算法的优势更加明显。

下面通过准确率对算法的聚类质量进行评价，聚类准确率的计算公式如下：

$$M_{Micro\text{-}precision}=\frac{1}{N}\sum_{i=1}^{k}a_i \quad (6)$$

其中，N 用于描述数据库中的数据总量；k 用于描述聚类簇的数量；a_i 用于描述簇 i 中被准确聚类成相应类别的样本数量。$M_{Micro\text{-}precision}$ 值越大，说明聚类质量越高。

表 1 列出了本文算法、D-Stream 算法和 EXCC 算法的聚类准确度比较结果。

表 1 3 种算法聚类准确度的比较结果

Table 3 Comparison results of clustering accuracy of three methods

数据量/个	本文算法/%	D-Stream 算法/%	EXCC 算法/%
1000	96.2	91.3	85.9
2000	98.3	92.5	82.2
3000	95.2	88.7	88.7
4000	94.7	89.6	91.5
5000	95.8	90.5	83.6
6000	97.2	92.6	82.5
7000	96.5	87.3	85.7
平均值	96.27	90.35	85.72

分析表 1 可知，本文算法对数据的聚类准确度均超过 94%，最高达 98.3%，平均聚类准确度为 96.27%，高于 D-Stream算法的 90.35%和 EXCC 算法的 85.72%。这说明本文算法的聚类精度高于 D-Stream 算法和 EXCC 算法。

结束语 本文提出一种新的云计算环境下高复杂度动态数据的增量密度快速聚类算法，依据密度对云计算环境下高复杂度动态数据进行聚类，通过 DBSCAN 算法，从插入或删除数据两方面进行增量聚类分析。实验结果表明，所提算法开销低、聚类速度快、聚类质量高。

参 考 文 献

[1] WU X X, NI Z W, NI L P. Research on fractal clustering ensemble algorithm based on cloud computing environment[J]. Computer Engineering and Applications, 2015, 51(14): 1-6. (in Chinese)
吴晓璇，倪志伟，倪丽萍. 云计算环境下基于分形的聚类融合算法研究[J]. 计算机工程与应用，2015，51(14)：1-6.

[2] WANG X, ZHOU X M. Research and Simulation on Big Data Reasonable Splitting Technology in Cloud Computing Environment[J]. Computer Simulation, 2016, 33(3): 292-295. (in Chinese)
王欣，周晓梅. 云计算环境下大数据合理分流技术研究与仿真[J]. 计算机仿真，2016，33(3)：292-295.

[3] SI F M. Research on incremental K-means clustering algorithm based on the density[J]. Journal of Changchun Institute of Technology (Natural Science Edition), 2016, 21(2): 114-117. (in Chinese)
司福明. 一种基于密度的增量 k-means 聚类算法研究[J]. 长春工程学院学报(自然科学版)，2016，21(2)：114-117.

[4] XING C Z, YU B S. An existence-level uncertain data stream clustering algorithm[J]. Computer Applications and Software, 2015(4): 252-255. (in Chinese)
邢长征，余彬生. 一种存在级不确定数据流聚类算法[J]. 计算机应用与软件，2015(4)：252-255.

[5] ZHAO L, CHEN Z K, ZHANG Q C, et al. Incomplete Data Imputation Algorithm Based on Distributed Subtractive Clustering [J]. Journal of Chinese Computer Systems, 2015, 36(7): 1409-1414. (in Chinese)
赵亮，陈志奎，张清辰，等. 基于分布式减法聚类的不完整数据填充算法[J]. 小型微型计算机系统，2015，36(7)：1409-1414.

[6] LIAN W W, FU L L, HUANG C. Simulation on Weak Association Mining Model of Data in Cloud Computing Environment [J]. Computer Simulation, 2015, 32(4): 359-362. (in Chinese)
廉文武，傅凌玲，黄潮. 云计算环境下数据弱关联挖掘模型的仿真[J]. 计算机仿真，2015，32(4)：359-362.

[7] XING Y F. MPI Parallel Algorithm for Heterogeneous Resource Scheduling Based on SOM and Particle Swarm Optimization[J]. Computer Measurement & Control, 2014, 22(8): 2523-2525. (in Chinese)
邢永峰. 基于 SOM 和 PSO 的云计算异构资源聚类 MPI 并行算法[J]. 计算机测量与控制，2014，22(8)：2523-2525.

[8] LIU J, GUO H S. K-means Cluster Center Optimization in Cloud Calculation[J]. Bulletin of Science and Technology, 2015, 31(10): 100-102. (in Chinese)
柳静，郭红山. 云计算中 K-means 聚类中心优化求解方法[J]. 科技通报，2015，31(10)：100-102.

[9] YU H Y, FAN J L. Robust Digital Watermarking Based on Ridgelet Transform and s-FCM[J]. Science Technology and Engineering, 2015, 15(11): 80-88. (in Chinese)
于海燕，范九伦. 一种基于 Ridgelet 变换和抑制式 FCM 聚类的数字水印算法[J]. 科学技术与工程，2015，15(11)：80-88.

[10] FAN T K. Research and implementation of user clustering based on MapReduce in cloud environment[J]. Electronic Design Engineering, 2016, 24(10): 35-37. (in Chinese)
樊同科. 云环境下基于 MapReduce 的用户聚类研究与实现[J]. 电子设计工程，2016，24(10)：35-37.

[11] XING C Z, WEN P. Uncertain data streams clustering algorithm based on grid density and force[J]. Application Research of Computers, 2015, 32(1): 98-101. (in Chinese)
邢长征，温培. 基于网格密度和引力的不确定数据流聚类算法[J]. 计算机应用研究，2015，32(1)：98-101.

[12] WU T, TAN G W. Real-time Data Loading of Dynamic Data Warehouse Using Index View Set[J]. Computer Science, 2016, 43(6A): 493-496. (in Chinese)
武彤，谭光炜. 基于索引视图实现动态数据仓库的实时数据加载[J]. 计算机科学，2016，43(6A)：493-496.

[13] XU F S, YAN L M, SHI K Q. Dynamic Data Intelligent Mining with Attributes Disjunctive Reduction and Expansion Characteristics[J]. Computer Science, 2015, 42(5): 215-220. (in Chinese)
徐凤生，闫立梅，史开泉. 具有属性析取萎缩-扩张特征的动态数据智能挖掘[J]. 计算机科学，2015，42(5)：215-220.

[14] LIU J L, CHENG C Y, CHEN Z, et al. Research on Cloud Data Management Model Based k-Means and Gridding Clustering[J]. Journal of Chongqing University of Technology (Natural Science), 2017, 31(9): 119-124. (in Chinese)
刘加伶，程春游，陈庄，等. 基于 k-Means 和网格化聚类的云数据管理模型研究[J]. 重庆理工大学学报(自然科学)，2017，31(9)：119-124.