第 35 卷 第 2 期 计 算 机 工 程 2009 年 1 月
Vol.35 No.2 Computer Engineering January 2009
·软件技术与数据库· 文章编号：1000—3428(2009)02—0063—02 文献标识码：A 中图分类号：TP301.6

基于 DBSCAN 的批量更新聚类算法

易宝林，伍仪强，丰大洋，张小莉
(华中师范大学计算机科学系，武汉 430079)

摘 要：为更新批量数据，提出一种基于 DBSCAN 的新聚类方法。该算法通过扫描原对象确定它们同增量对象间的关系，得到一个相关对象集，同时根据该相关对象和增量对象之间的关系获得新的聚类结果。实验结果表明，该算法与 DBSCAN 是等价的，能更有效地解决批量数据更新时的增量聚类问题。

关键词：空间数据挖掘；增量聚类；空间数据库；批量更新聚类算法

Batch Update Clustering Algorithm Based on DBSCAN

YI Bao-lin, WU Yi-qiang, FENG Da-yang, ZHANG Xiao-li
(Dept. of Computer Science, Central China Normal University, Wuhan 430079)

【Abstract】 In order to update the batch data, a novel clustering algorithm based on DBSCAN is proposed, which determines the relation between the original object and increment object by scanning the original one. Thus, a relevant object set is got, according to which the new clustering result is obtained combined with increment object. Experimental results show this algorithm is equal to DBSCAN, and can solve the increment clustering problem when the batch data is updated effectively.

【Key words】 spatial data mining; increment clustering; spatial database; Batch Update Clustering Algorithm(BUCA)

在过去的十几年中，人们已经提出了许多有效且通用的空间数据聚类算法，这些算法为高维空间数据聚类提供了完整的理论依据和高效的实现方法。然而，用于聚类的数据源可能不断地变化，这意味着聚类结果可能与新的数据集并不匹配。对于这种情况，可以对新的数据集进行重新聚类，但这种做法的代价太大，且未利用已有的聚类信息。如果在已有的聚类信息上仅执行这些增量对象的聚类更新，而非重新聚类整个更新后的数据库，应当是种可行的做法。因此，设计有效的增量聚类算法便成为该方法所面临的任务。

1 相关工作

人们已提出多种聚类思想，如基于划分的算法、基于层次聚类的算法、基于密度聚类的算法等。其中，DBSCAN[1]是基于密度的聚类算法中一个典型算法。

文献[1]基于密度聚类的观点提出DBSCAN，其主要思想是：对于一个簇中的每个核心对象来说，在给定半径ε的邻域内必须至少包含Minpts个对象，即该邻域的对象密度必须超过某个阈值(Minpts)。DBSCAN能够有效地解决异常点问题，对高维空间聚类有着特别的优势。用于聚类分析的数据量基本上都很大，且可能频繁地被更新。从聚类性能方面考虑，每次扫描数据库中所有数据终究不是个好的方法。因此，人们提出增量聚类算法[2-3]来解决这个问题，在充分利用已有聚类信息的前提下，只去修改聚类结果已发生改变的对象，从而提高再聚类的性能。

文献[2]为单个对象的插入和删除提出了增量聚类算法，一般情况下，对象的插入对聚类结果的影响有 3 种：新对象并入已有簇，新对象为噪音或新建一个簇，合并已有簇；对象的删除对聚类结果的影响也有 3 种：直接删除对象，注销已有簇，分裂已有簇。这为整个基于 DBSCAN 的增量聚类算法提供了理论依据，但其将单个对象的修改看成单个对象的先删除再增加，且对批量操作也未做进一步论证。

文献[3]把单对象的插入和删除推广到批量操作，批量地插入和删除对聚类结果的影响和上面所述是一致的，只是通过对象间的联系综合考虑了整个影响，从而提高算法效率。然而分别考虑插入和删除，更新就被看作插入和删除的结合，这虽然能够处理更新问题，但当一个大的数据集进行了批量更新后，这些算法将会做大量无用的工作，从而导致算法效率低下。因此，对将要更新的对象和已经更新的对象之间的关联予以考虑，而不是把这些对象分割为 2 个毫不相关的集合，必然可以进一步提高批量更新时算法的效率，本文基于以上的理论依据，对将要更新的对象和已经更新的对象之间的关联进行研究，同时讨论了批量更新对聚类结果的影响。

2 批量更新聚类算法

2.1 BUCA 的算法基础

衡量增量聚类算法的一个重要指标是算法的等价性，即通过增量聚类算法得到的结果应该和重新运行一次 DBSCAN 的结果相一致。那么在批量更新后，重新运行一次 DBSCAN 与第 1 次运行时有哪些变化，批量更新操作对最终聚类结果所造成的影响有哪些是设计增量聚类算法的基础。

批量更新对聚类结果的影响包括由插入和删除所带来的 6 种影响。为更好地说明这 6 种影响在 BUCA 中的具体表现，本文引入 2 个相关定义。

定义 1(密度可达传递) 设 D 为一组对象的集合，对象

作者简介：易宝林(1969－)，男，副教授、博士，主研方向：空间数据库；伍仪强、丰大洋、张小莉，硕士

收稿日期：2008-05-10 **E-mail：**epower@mail.ccnu.edu.cn

$P \in D$，那么从 P 出发，聚类所有密度可达的对象的过程称为密度可达传递。

假设重新运行 DBSCAN，当扫描一个对象时，在该对象的 ε 邻域内，对象因为更新操作而变多，且超过了 Minpts，根据 DBSCAN 聚类过程，把所有密度可达的对象归入该簇，在 BUCA 中用密度可达传递来实现，实现过程可能遇到以下 2 种情形：(1)q 是个噪音，这时需要生成一个新的簇。(2)q 的 ε 邻域的一个对象已经属于一个簇，这时需要判断这个对象。如果这个对象是一个核心对象，那么在这个对象的簇中的所有对象都会密度可达，因此，需要合并这 2 个簇，这是因为一个非核心对象属于多个簇是允许的。

定义 2(密度可达反向传递) 设 D 为一组对象集合，对象 $P \in D$，那么从 P 出发，重新聚类所有因 P 修改而不再密度可达的对象的过程称为密度可达反向传递。

假设重新运行 DBSCAN，当扫描一个对象时，在该对象的 ε 邻域内，对象因为更新操作而变少，且小于 Minpts，根据 DBSCAN 聚类过程，该邻域内的对象将不能被归入该簇，但这些对象可能在更新前已被归入该簇，BUCA 用密度可达反向传递来处理这种问题，实现过程同样可能遇到以下 2 种情形：(1)q 的簇里面已没有任何核心对象，那么 q 的簇里面的对象就不再能形成一个簇，所以，需要注销这个簇，而该簇的原来对象 r 需要逐一扫描以重新确定它们属于哪个簇。如果 r 的 ε 邻域内还存在其他核心对象，那么 r 属于这个核心对象的簇，否则就将 r 标记为噪音。(2)q 的簇里面存在一个核心对象，但是这个对象与 q 的簇里面的任何其他核心对象都不是密度可达的，那么就要分解这个簇。

2.2 BUCA 的算法思想

当扫描 D 中对象时，需要确定哪些对象是和更新操作相关的。这些对象可以分为 2 类：(1)对象在其邻域中包含了更新对象，这样更新后它的邻域对象的数量发生了变化，所以，可能导致新的密度可达传递或反向传递；(2)对象是从更新对象出发密度可达的对象，这可能导致从已更新对象的密度可达传递或反向传递。

为便于讨论，假设 $D\{p1,p2,\cdots,pi,pj,\cdots,pn\}$ 被更新为 $D'\{p1,p2,\cdots,pi,qj,\cdots,qn\}$，定义 $D1$ 为未被更新对象的集合，这里指 $\{p1,p2,\cdots,pi\}$；$D2$ 为所要更新的原始对象集合，这里为 $\{pj,pj+1,\cdots,pn\}$；$D3$ 为更新后的新对象集合，这里为 $\{qj,qj+1,\cdots,qn\}$。BUCA 主要考虑对象集 $D2$ 和 $D3$ 中的对象。

Step1 扫描 $D2$ 和 $D3$ 中的每个对象 p，并得到 p 的 ε 邻域对象。

Step2 假设 q 为 p 的 ε 邻域对象，先得到 q 的 ε 邻域对象，将其中属于 $D1$ 的对象数量标记为 $m1$，属于 $D2$ 的对象数量标记为 $m2$，属于 $D3$ 的对象数量标记为 $m3$。

Step3 根据以上 3 个值之间的关系，分为以下 4 种情况：

(1)$m1+m2 \geq$ Minpts 且 $m1+m3 \geq$ Minpts

在这种情况下，原始核心对象 q 在更新之后仍然是核心对象，对于 q 的 ε 邻域的每个对象，如果它属于 $D1$，那么它的簇就不会改变；如果它属于 $D2$，需对它执行密度可达反向传递；如果它属于 $D3$，需对它执行密度可达传递。

在执行密度可达反向传递和密度可达传递的过程中，可以用批量的方式来处理以避免重复操作。显然当一个对象既属于 $D2$ 又属于 $D3$(例如，一个对象 $p' \in D2$，如果它的位置被另一个对象 $q' \in D3$ 所代替，那么就认为 $q'=p'$)时，那么对这个对象执行的密度可达反向传递和密度可达传递就会相互抵消。事实上，当一个对象既属于 $D2$ 的 ε 邻域又属于 $D3$ 的 ε 邻域时，从这个对象执行的密度可达反向传递和从这个对象执行的密度可达传递就会相互抵消，所以，不必处理这个对象。

(2)$m1+m2 \geq$ Minpts 且 $m1+m3 <$ Minpts

在这种情况下，原始核心对象 q 在更新之后成了一个边界对象，需要对 q 的 ε 邻域内的这些对象执行密度可达反向传递。

(3)$m1+m2 <$ Minpts 且 $m1+m3 \geq$ Minpts

在这种情况下，一个原始的非核心对象 q 在更新之后变成核心对象，q 的 ε 邻域的所有对象密度可达并形成 q 的簇，所以，需要对这些对象进行密度可达传递。

(4)$m1+m2 <$ Minpts 且 $m1+m3 <$ Minpts

在这种情况下，一个原始边界对象 p 在更新之后仍然是一个边界对象，那么就不必再调整了。

3 分析和实验

3.1 BUCA 的分析

性质 BUCA 与 DBSCAN 的结果等价

证明 在 DBSCAN 中，当它扫描数据库中的每个对象时，所有的密度可达对象同属于一个簇，但由于对象的更新，BUCA 必须确定更新的对象属于哪个簇。

在扫描数据库中的每个对象时，DBSCAN 必须处理 2 种情况：(1)如果已经确定对象属于某个簇，那么就不再处理这个对象，这点和 BUCA 是相同的；(2)对未被聚类的对象执行密度可达传递，BUCA 将这种情况分为 4 类。

如果对象 P 在更新之后成为核心对象，在 DBSCAN 中需要对该对象的 ε 邻域内的每个对象执行密度可达传递。BUCA 将这种情况分为两个方面，一个方面是在更新之前，对象 P 是个核心对象，在这种情况下，P 的 ε 邻域内的所有未改变的对象在生成旧的聚类时已经得到检查，所以，这些对象不必再检查；另一个方面是在更新之前不是核心对象的那些对象，这些对象仍需重新检查。

如果对象 P 在更新之后不是核心对象，在 DBSCAN 中，什么都不必做，但是 DBSCAN 仍然将这种情况分为 2 类：(1)P 在更新之间是个核心对象，所以，在更新之前从 P 密度可达的所有对象需要重新检查；(2)P 在更新之前不是核心对象，BUCA 不必调整聚类。所以，BUCA 与 DBSCAN 的聚类结果是等价的。证毕。

3.2 实验结果

这里给出部分实验结果，并将其与 DBSCAN 比较。测试数据是随机产生的，算法用 C++编辑，实验在配置为 512 MB RAM, AMD XP2800++的电脑上运行。

先以批量的方式在 150 个对象中更新 50 个对象。其中，$\varepsilon=40$, Minpts$=3$，图 1 是 DBSCAN 和 BUCA 的聚类结果，该结果表明 BUCA 和 DBSCAN 是等价的。

(a)更新前 DBSCAN 聚类　(b)更新后 DBSCAN 聚类　(c)更新后 BUCA 聚类

图 1　DBSCAN 和 BUCA 的聚类结果

(下转第 67 页)