基于网格的数据流聚类算法*)

刘青宝　戴超凡　邓　苏　张维明
(国防科学技术大学信息系统与管理学院　长沙 410073)

摘　要　本文提出的基于网格的数据流聚类算法，克服了算法 CluStream 对非球形的聚类效果不好等缺陷，不仅能在噪声干扰下发现任意形状的类，而且有效地解决了聚类算法参数敏感和聚类结果无法区分密度差异等问题。
关键词　聚类，数据流，聚类参数，相对密度

Grid based Data Stream Clustering Algorithm

LIU Qing Bao　DAI Chao Fan　DENG Su　ZHANG Wei Ming
(College of Information System and Management, National University of Defense Technology, Changsha 410073)

Abstract　With strong ability for discovering arbitrary shape clusters and handling noise, grid based data stream clustering algorithm efficiently resolves these problem of being very sensitive to the user defined parameters and difficult to distinguish the density distinction of clusters.
Keywords　Clustering, Data stream, Clustering parameter, Relative density

随着计算机和传感器技术的发展和应用，数据流挖掘技术在国内外得到广泛研究。它在网络监控、证券交易分析、电信记录分析等方面有着巨大的应用前景。特别在军事应用中，为了获得及时的战场态势信息，大量使用了各种传感器，对这些传感器数据流的分析处理已显得极为重要。针对数据流数据持续到达，且速度快、规模大等特点，数据流挖掘技术的研究重点是设计高效的单遍数据集扫描算法[1,2]。数据流聚类问题一直是吸引许多研究者关注的热点问题，已提出多种一次性扫描的方法和算法，如文[1~4]等等，但它们的聚类结果通常是球形的，不能支持对任意形状类的聚类[5]。

本文提出的基于网格的数据流聚类算法，在有限内存条件下，以单遍扫描方式，不仅能在噪声干扰下发现任意形状的类，而且有效地解决了基于绝对密度聚类算法所存在的高密度聚类结果被包含在相连的低密度聚类结果中的问题。

本文第 1 节简要介绍数据流聚类相关研究，并引出基于网格的数据流聚类算法的思路及其与相关研究的异同；第 2 节给出基于网格的数据流聚类算法所使用到的基本概念；第 3 节给出一个完整的基于网格的数据流聚类算法，详细解析算法的执行过程；第 4 节进行算法性能分析对比；最后总结本文的主要工作和贡献，并指出需要进一步研究和改进的工作。

1　相关研究

在有限内存约束下，一般方法很难对数据流进行任意形状的聚类。第一个增量式聚类挖掘方法是文[6]提出的 IncrementalDBSCAN 算法，它是一个用于数据仓库环境(相对稳定的数据流)的有效聚类算法，可以在有噪声的数据集中发现任意形状的类。但是，它为了形成任意形状的类，必须用类中的所有点来表示，要求获得整个数据流的全局信息，这在内存有限情况下是难以做到的。而且，它采用全局一致的绝对密度作参数，使得聚类结果对参数值非常敏感，设置的细微不同即可能导致差别很大的聚类结果。

Aggarwal 在 2003 年提出的一个解决数据流聚类问题的框架 CluStream[1]。它使用了两个过程来处理数据流聚类问题：首先，使用一个在线的 micro cluster 过程对数据流进行初级聚类，并按一定的时间跨度将 micro cluster 的结果按一种称为 pyramid time frame 的结构储存下来。同时，使用另一个离线的 macro cluster 过程，根据用户的具体要求对 micro cluster 聚类的结果进行再分析。但它采用距离作为度量参数，聚类结果通常是球形的，不能支持对任意形状类的聚类。而且，它维护的是 micro cluster 的聚类特征向量(CF^2x; CF^1x; CF^2t; CF^1t; n)，这在噪声情况下，会产生干扰误差。

2006 年，Feng Cao 等人在文[5]中提出了针对动态进化数据流的 DenStream 算法。它相对 CluStream 有很大的改进，继承了 IncrementalDBSCAN 基于密度的优点，能够支持对有噪声的动态进化(非稳定)的数据流进行任意形状的聚类。但由于采用全局一致的绝对密度作参数，使得聚类结果对参数值非常敏感。同时，与 CluStream 算法相比，它只能提供对当前数据流的一种描述，不能反映用户指定时间窗内的流数据的变化情况。

朱蔚恒等在文[13]中提出的基于密度与空间的 ACluStream 聚类算法，通过引入有严格空间的意义聚类块，在对数据流进行初步聚类的同时，尽量保留数据的空间特性，有效克服了 CluStream 算法不能支持对任意形状聚类的缺陷。但它在处理不属于已有聚类块的新数据点时，使用一种类似“抛硬币”的方法来猜测是否为该点创建一个新的聚类块，误差较大。而且它以绝对密度做参考，所以在聚类结果中无法区分密度等级不同的簇[7]。

本文提出的基于网格的数据流聚类算法 GClustream

*)国家自然科学基金(60172012)。刘青宝　博士生，副教授，主要研究方向为数据仓库技术和数据挖掘；戴超凡　博士，副教授，主要研究方向为数据仓库技术和数据挖掘；邓　苏　博士，教授，主要研究方向指挥自动化、信息综合处理与辅助决策；张维明　博士生导师，教授，主要研究方向为军事信息系统、信息综合处理与辅助决策。

?1994-2018 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

(Grid Based Data Stream Clustering Algorithm), 借鉴算法 CluStream 的两阶段聚类思想和 pyramid time frame 的快照储存结构, 采用相对密度作为聚类参数, 通过对数据空间进行网格化处理, 提高了算法处理速度, 并能在噪声干扰条件下发现任意形状的类, 同时解决了基于绝对密度聚类算法所存在的高密度聚类结果被包含在相连的低密度聚类结果中的问题[7]。

2 基本概念

定义 1 网格单元

在各维上定义一个单位格长, 采用网格方式将 n 维空间划分为若干个网格单元。一个网格单元是 n 维空间中各个维上具有单位格长的 n 维超立方体, 即以 n 维向量 o 为起点, 向各维的正方向延伸单位格长所形成的一个区间, 记为 Grid(o)。

定义 2 聚合块

由若干个网格单元组成的超立方体, 称为聚合块, 记为 Cub(o, $\vec{r}$), 其中 $\vec{r}$ 为聚合块的各维边长组成的向量。

采用衰变窗口模型[5], 数据流上的数据对象, 其权重随时间衰减, 即 $w_p(t_c)=2_{cp}^{-\lambda(t-t)}$, 其中 λ 表示衰减速度, t_c 表示当前时间, t_P 表示数据对象 p 到达时间。设数据流在时刻 t_0, t_1, t_2, …, t_c 到达的数据对象数 n_0, n_1, n_2, …, n_c, 则数据流的当前时刻 t_c 权重 $W(t_c)$ 为

$$W(t_c)=\sum_{j=0}^{c} n_j 2^{-\lambda(t_c-t_j)}$$

定义 3 网格单元特征向量

设起点为 o_i 的网格单元包含 n 个分别在时刻 t_{i1}, t_{i2}, …, t_{in} 到达的数据对象 p_{i1}, p_{i2}, …, p_{in}, 在 t_c 时刻网格单元的特征向量记为(o_i, $\overrightarrow{F^1}$, $\overrightarrow{F^2}$, w, t_c), 其中

$$\overrightarrow{F^1}=\sum_{j=1}^{n} p_{ij} 2^{-\lambda(t_c-t_{ij})}$$

$$\overrightarrow{F^2}=\sum_{j=1}^{n} p_{ij}^2 2^{-\lambda(t_c-t_{ij})}$$

$$w=\sum_{j=1}^{n} 2^{-\lambda(t_c-t_{ij})}$$

定义 4 密集网格单元和候选密集网格单元

对于给定的密度阈值 ξ($0<\xi<1$), 设网格单元的特征向量为(o_i, $\overrightarrow{F^1}$, $\overrightarrow{F^2}$, w, t_c)。若 $w>\xi W(t_c)$, 则称该网格单元在 t_c 时刻为密集网格单元, 记为 D-Grid(o_i); 若 $0<w<=\xi W(t_c)$, 则称该网格单元在 t_c 时刻为候选密集网格单元, 记为 C-Grid(o_i)。

定义 5 密集聚合块

对于给定的密度阈值 ξ($0<\xi<1$), 设聚合块的特征向量为(o_i, $\overrightarrow{F^1}$, $\overrightarrow{F^2}$, w, t_c, $\vec{r}$), 若有 $w>\xi W(t_c)\prod r_i$, 则称该聚合块在 t_c 时刻为密集的。

3 基于网格的数据流聚类算法

借鉴算法 CluStream 的思路[1], 基于相对网格的数据流聚类算法 GCluStream 分为两个阶段: 在线的进程和离线的进程。记录当前数据流聚类特征的在线进程称为 GMic Cluster, 而离线的响应查询的进程称为 GMac Cluste。

3.1 GMic Cluster 过程描述

在线进程 GMic Cluster 的具体步骤如下:

(1) 初始化

初始时, 对每个新到来的数据对象, 计算出其所在网格单元的特征向量。积累一定数量的数据对象后, 区分密集网格单元集合和候选密集网格单元集合。

(2) 加入数据对象

对新到的数据对象 p, 若它属于一个已存在的密集网格单元或候选密集网格单元, 则修改该网格单元的特征向量为(o_i, $\overrightarrow{F^1}+p$, $\overrightarrow{F^2}+p^2$, $w+1$, t_c)。否则, 直接定位其所在的网格单元, 计算该网格单元的特征向量, 并把它加入到候选密集网格单元集合。

(3) 生成密集聚合块

在连续数据流条件下, 非密集网格单元通过新数据对象的不断聚集, 可以转换为密集网格单元。在内存空间有限的条件下, 随着密集网格单元的数目增大, 须把相邻且密度相近的密集网格单元进行聚合, 以节省空间消耗。同样, 可把相邻且密度相近的聚合块聚合为更大的块。这一步的聚合条件是要求相邻、密度相近、同体积大小的两个密集网格单元或两个初级聚类块才能聚合。

(4) 密集聚合块的切分

聚合块在新数据对象的不断加入下, 可能导致内部密度失衡。在每次加入数据对象时, 更新聚合块特征向量, 并计算方差, 判断失衡程度。当超过一定阈值 δ 则从失衡程度最大、边长超过 1 的那一维进行居中切分, 对切分形成的两个新聚合块或两个新网格单元进行特征向量分割计算。

(5) 密集聚合块、密集网格单元的退化

由于引进了衰减因子 $2^{-\lambda}$, 若没有新数据对象的加入, 初级聚类块特征向量修改为(o^i, $\overrightarrow{F^1}*2^{-\lambda\Delta}$, $\overrightarrow{F^2}*2^{-\lambda\Delta}$, $w*2^{-\lambda\Delta}$, t_c, $\vec{r}$), 密集网格单元特征向量修改为(o_i, $\overrightarrow{F^1}*2^{-\lambda\Delta}$, $\overrightarrow{F^2}*2^{-\lambda\Delta}$, $w*2^{-\lambda\Delta}$, t_c), 其中 Δt 为特征向量上次修改到当前修改的时间间隔。一旦密集聚合块特征向量(o_i, $\overrightarrow{F^1}$, $\overrightarrow{F^2}$, w, t_c, $\vec{r}$) 的 $w<=\xi W(t_c)\prod_{r_i}$, 则让该密集聚合块“土崩瓦解”成若干个候选密集网格单元。而若密集网格单元特征向量(o_i, $\overrightarrow{F^1}$, $\overrightarrow{F^2}$, w, t_c) 的 $w<=\xi W(t_c)$, 则直接退化候选密集网格单元。

虽然特征向量中的 w 在不断衰减, 但为了节省时间开销, 只需每隔一定时间对无新添数据对象的密集聚合块和密集网格单元进行一次特征向量计算。该时间间隔 Δt 通过方程 $\xi W(t_c)2^{-\lambda\Delta t}+1=\xi W(t_c)$ 计算可得:

$$\Delta t=\lceil 1/\lambda\log(\xi w(t_c)/\xi w(t_c)-1)\rceil$$

(6) 候选网格单元的筛选

不断到来的数据对象, 要是不能被现有的初级聚类块、密集网格单元和候选网格单元所吸收, 则会产生大量的候选密集网格单元, 因此必须把无潜力升级为密集网格单元的候选网格单元进行清除, 以节省内存空间。筛选的策略为: 对没有新数据对象加入的候选网格单元更新其特征向量; 对所有候选网格单元按其特征向量中 w 值进行从大到小排序; 对特征向量中 w 值较小的进行清除, 直到满足内存要求。

(7) 把密集聚合块和密集网格单元的特征向量写入磁盘

按一定的时间跨度将 GMic Cluster 过程的内存结果: 密集聚合块和密集网格单元的特征向量, 写入磁盘, 并按 CluStream 算法提出的 pyramid time frame 的策略组织这些结果。Pyramid time frame 是一种按时间来组织数据流的初步描述的策略, 它保证了对一个用户定义的时间窗 h, 在当前时刻的 $2h$ 窗口内至少存在一个 snapshot。关于 pyramid frame 的思想和具体做法可见文[1]。GCluStream 同样采取 pyramid frame 结构储存数据。

3.2 GMic Cluster 过程描述

GMic Cluster 得到的结果以 pyramid time frame 结构储

?1994-2018 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

存在磁盘上，然后根据用户查询参数：当前查询时刻 t_c 和查询的时间窗宽度 h，得到这两个不同时刻的快照结果：snapshot(t_c) 和 snapshot(t_c-h)，其中 snapshot(t_c-h) 是当前时刻的 $2h$ 窗口中最接近 t_c-h 时刻的快照。比较两个快照，可以得出 3 个集合：{新增的密集网格单元}、{消失的密集网格单元}和{保持的密集网格单元}，据此可以分析数据流的变化情况。

由于网格单元具有严格的空间意义，在密集网格单元集合上进行聚类的过程类似 GMic cluster 过程的步骤(3)，只是这时的聚合条件放宽为相邻的、密度相近的两个密集网格单元或两个密集聚类块进行聚合，不要求它们体积大小相同。

3.3 算法描述

先积累一段时间的数据流，然后形成密集网格单元集合 Dset 和候选密集网格单元集合 Cset，并设置密集聚合块集合 DCubset 为空集。

```
GMic Cluster(λ, ξ, δ DS)
BEGIN
    REPEAT
        p= GetPoint( DS);
        object= FindGridorCub( piont);
{根据点的空间位置，搜索是否存在一个包含该点的密集网格单元或候选密集网格单元或密集聚合块}
IF object<>NULL THEN
 UpdateFV( object);
 {修改此网格单元或所在聚合块的特征向量}
IF object∈DCubset THEN
  CheckDensity( obj);
  {检查密集聚合块的密度情况，步骤(4)}
    END IF;
  ELSE
    C_grid= NewCGrid( p);
   {创建新的候选网格单元}
   GetFV( C_grid);
   {计算其特征向量}
   InsertSet( Cset, C_grid);
   {插入到候选密集网格单元集合 Cset}
   END IF;
Δt=⌈1/λ log(ξw(t_c)/(ξw(t_c)-1))⌉;
   IF (t_c mod Δt)==0 THEN
   FOR each obj∈Dset OR object∈DCubset DO
   CheckDensity( obj);
   {检查密集聚合块和密集网格单元的退化情况，步骤(5)}
    END FOR;
  END IF;
  IF 内存不够 THEN
    FOR each obj∈Dset OR object∈DCubset DO
      Aggr( obj, δ);
      {进行聚合，步骤(3)，参数 δ(0<δ<1)，是判断两对象密度是否相近的阈值}
    END FOR;
    SortCompress( Cset);
    {步骤(6)}
    END IF;
  UNTIL 数据流结束;
END GMic Cluster。
```

4 算法分析

这里只对算法的在线处理部分进行分析。我们从处理时间和参数选择等两个主要指标分析本文提出的基于网格的数据流聚类算法 GClustream。

4.1 处理时间对比

由于对数据空间采用了网格化处理，每当新数据对象到达，则可直接定位其所属网格单元，时间复杂度为常数 1。而算法 CluStream 则要通过计算新数据对象到各子聚类中心的距离，并比较之后才能决定其所属的子聚类，其时间复杂度为 O(N)，其中 N 为子聚类的数量。

在聚合密集网格单元步骤中，判断两个密集网格单元是否可以合并，主要看它们是否相邻，即两者之间是否存在一个公共的超平面，具体判断如下：

设两个密集网格单元为 D_Grid(o_1) 和 D_Grid(o_2)，若向量$\overrightarrow{O_1O_2}$=(0, 0, …, 1, …, 0)或$\overrightarrow{O_1O_2}$=(0, 0, …, -1, …, 0)，则 D_Grid(o_1) 与 D_Grid(o_2) 是相邻的。所以，判断两个密集网格单元之间是否相邻，只需进行一次减法运算，时间复杂度为常数 1。而算法 CluStream 判断两子聚类是否可以合并，则要计算各子聚类中心之间的距离，并进行比较之后才能决定，时间复杂度为 O(N log N)。

4.2 参数选择分析

基于网格的数据流聚类算法 GClustream 采用了具有实际意义的参数 λ、ξ、δ 用户能很好地理解。衰减因子 λ 反映了历史数据对象对当前聚类结果的影响程度，即 λ 取值越大，衰减越快，历史数据对象对当前聚类结果的影响越小；密度阈值 ξ 决定了对噪声的过滤程度，即 ξ 取值越小，则被过滤的背景噪声数据越少；相对密度阈值 δ 反映了用户对密度分辨率的要求程度，δ 取值越接近 1，聚类结果中密度分辨率越高，结果中类的数目越多。

而 CluStream 算法的 micro cluster 过程使用的是类似 BRICH 算法所使用的聚类特征值来记录它所产生的子聚类，BRICH 算法的缺点必然也带入 micro cluster 过程中。BRICH 算法记录聚类特征值的方法对球型的聚类效果好，但对其他形状的聚类，BRICH 不能很好地工作[9]。而且，micro cluster 对新数据的处理，利用数据与最近子聚类中心的距离以及一个预定的距离阈值来判断数据是否属于该子聚类。这种处理方法与算法 GClustream 相比有 3 个缺陷：一是计算量大，这点前面已经分析；二是聚类结果不好，因为距离某一子聚类中心近的点不一定就属于该子聚类；三是预定的距离阈值参数难以确定。为了改进 CluStream 算法的缺陷，ACluStream 算法采用基于密度的聚类方法进行在线记录当前数据流的特征，但由于是以绝对密度作参数，在聚类结果中无法区分密度等级不同的簇。

小结　基于网格的数据流聚类算法 GClustream 通过对数据空间采用网格化处理，大大提高了算法处理速度，并能在噪声干扰条件下发现任意形状的类，同时解决了基于绝对密度聚类算法所存在的高密度聚类结果被包含在相连的低密度聚类结果中的问题。不过，针对高维数据空间数据流聚类问题，无论是基于绝对密度还是基于相对密度的方法，皆因高维空间中对象分布稀疏、噪声难以区分等特性而失效，必须采用适当的降维处理才能适用，这将是我们下一步研究的方向。

(下转第 180 页)

?1994-2018 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

4 McDermott D. PDDL: the planning domain definition language: [Technical Report]. CVC TR-98-003/DCS TR-1165. Yale Center for Computational Vision and Control, 1998
5 Fox M, Long D. PDDL2.1: An extension to PDDL for expressing temporal planning domains. J Artificial Intelligence Res, 2003, 20: 61~124
6 Edelkamp S, Hoffmann J. PDDL2.2: The language for the Classic Part of the 4th International Planning Competition: [T.R. no. 195]. Institut für Informatik, Freiburg, Germany, 2004
7 Hoffmann J, Edelkamp S, Englert R, et al. Towards Realistic Benchmarks for Planning: the Domains Used in the Classical Part of IPC-4- Extended Abstract. In: Proceedings of the 4th International Planning Competition (IPC-4), June, Whistler, Canada, 2004, 7~14
8 Thielscher M. Ramification and causality. Artificial Intelligence, 1997, 89: 317~364
9 McCain N, Turner H. A causal theory of ramifications and qualifications. In: Proceedings IJCAI-95, Montreal, Que, 1995. 1978~1984
10 Davidson M, Garagnani M. Pre-processing planning domains containing Language Axioms. In: Grant T, Witteveen, C, eds. Proc of the 21st Workshop of the UK Planning and Scheduling SIG (PlanSIG-02), 2002. 23~34
11 Garagnani M. A correct algorithm for efficient planning with pre-processed domain axioms. In: Bramer M, Preece A, Coenen F, eds. Research and Development in Intelligent Systems XVII (Proc of ES 2000). 2000. 363~374
12 Vidal V. A lookahead strategy for solving large planning problems. IJCAI 2003. 1524~1525
13 Hoffmann J, Nebel B. The FF Planning System: Fast Plan Generation Through Heuristic Search. Journal of Artificial Intelligence Research, 2001, 14: 253~302

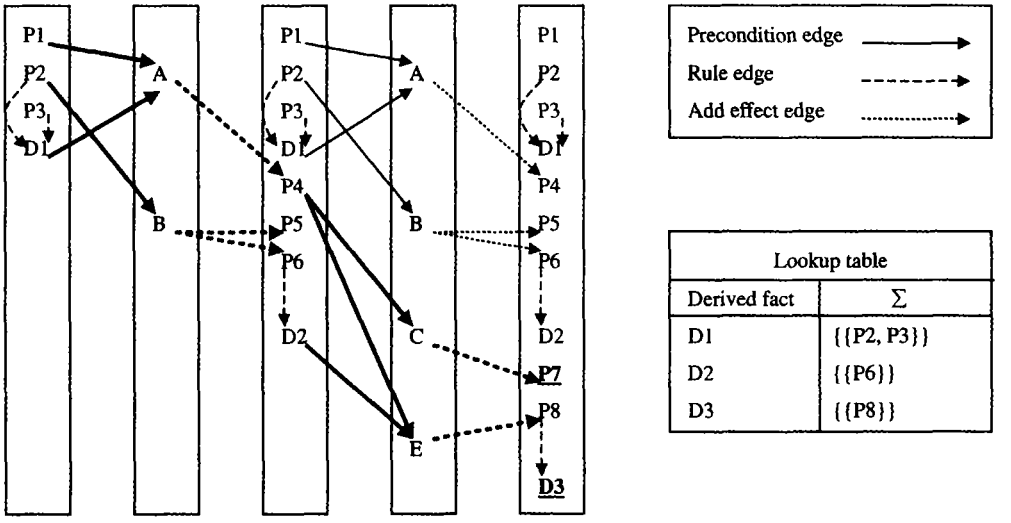

图 7 放宽式规划的一个示例

带下划线的粗体部分表示目标状态的命题。蓝色粗线表示解路径。

(上接第 161 页)

参 考 文 献

1 Aggarwal CC, Han J, Wang J, et al. A framework for clustering evolving data streams. In: Proc. of VLDB, 2003
2 Aggarwal C C, Han J, Wang J, et al. A framework for projected clustering of high dimensional data streams. In: Proc. of VLDB, 2004
3 Beringer J, Hullermeier E. Online Clustering of Parallel Data Streams. Data & Knowledge Engineering, 2005
4 Guha S, Meyerson A, Mishra N, et al. Clustering Data Streams: Theory and Practice. In: TKDE special issue on clustering, Vol. 15, 2003
5 Cao F, Estery M, Qian W, et al. Density-based Clustering over an Evolving Data Stream with Noise. In: Proceedings of the 2006 SIAM Conference on Data Mining (SDM'2006)
6 Ester M, Kriegel H-P, Sander J, et al. Incremental clustering for mining in a data warehousing environment. In: Gupta A, Shmueli O, Widom J, eds. Proceedings of the 24thInternational Conference on Very Large Data Bases. New York: Morgan Kaufmann Publishers Inc, 1998. 323~333
7 Liu Qing Bao, Deng Su, Lu Changhui, et al. Relative Density Based K nearest Neighbors Clustering Algorithm. In: the Second International Conference on Machine Learning and Cybernetics, China, 2003
8 Ester M, Kriegel H-P, Sander J, et al. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In: Proc. 2nd Int Conf on Knowledge Discovery and Data Mining, Portland, OR, 1996. 226~231
9 Han Jiawei, Kamber M, Fan Ming, et al. Data Mining: Conceots and Techniques. Beijing: China Mechine Press, 2001(in Chinese)
10 Ankerst M, Breunig M, Kriegel H-P, et al. OPTICS: Ordering Points To Identify the Clustering Structure. In: Proc. ACM SIGMOD'99, Int Conf. on Management of Data, Philadelphia, PA, 1999
11 Zhou Yong Feng, Liu Qing Bao, Deng Su, et al. An Incremental Outlier Factor Based Clustering Algorithm. In: the First International Conference on Machine Learning and Cybernetics Nov China, 2002
12 金澈, 清卫宁, 周傲英. 流数据分析与管理综述. 软件学报, 2004, 15(8)
13 朱蔚恒, 印鉴, 谢益煌. 基于数据流的任意形状聚类算法. 软件学报, 2006, 17(3)

?1994-2018 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net