时间滑动窗口上数据流极值聚集的空间优化

丁维龙[1,2,3]，韩燕波[1,2]，王菁[1]，赵卓峰[1]
（1. 北方工业大学云计算研究中心，100041，北京；2. 中国科学院计算技术研究所，100190，北京；
3. 中国科学院研究生院，100049，北京）

摘要：传统的数据流极值聚集方法在极端情形下为获得连续的精确解，会因维护大量候选项而导致巨大的内存开销，为此文中提出了一种时间滑动窗口上内存有界的极值聚集方法. 在候选项数量达到指定阈值时，该方法随机抽样新到达窗口的数据，使得内存维护有限数量的候选项，连续返回极值近似解. 设计了一种空间有界的摘要数据结构 REx-link，可以在有界的内存中基于随机抽样进行维护，实现时间滑动窗口上的数据流极值聚集. 从理论上证明了随机算法的出错概率存在上界，并通过仿真实验分析了算法的返回结果与精确解的近似程度. 分析表明，计算精度和空间开销的折中是实际应用可接受的.

关键词：数据流；极值聚集；抽样
中图分类号：TP309 **文献标志码**：A **文章编号**：0253-987X(2012)11-0106-06

A Space Optimization Method to Aggregate Extremes of Data Stream over Time-Based Sliding Windows

DING Weilong[1,2,3], HAN Yanbo[1,2], WANG Jing[1], ZHAO Zhuofeng[1]
(1. Research Center for Cloud Computing, North China University of Technology, Beijing 100041, China;
2. Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China;
3. Graduate University of Chinese Academy of Sciences, Beijing 100049, China)

Abstract: A space-bounded extreme aggregation of data stream is proposed over time-based sliding window to reduce the vast space of traditional methods under the worst cases. When the number of candidates reaches a given threshold, the method would randomly sample the new arrival tuples in the window to maintain limited candidates and continuously return approximate extremes. A space-bounded synopsis structure REx-link is designed to maintain extreme candidates by random sampling in memory, and to implement extreme aggregation over time-based sliding window. It is proved that an upper bound for the error probability of the random algorithm exists. Comprehensive experiments are performed to analyze the accuracy between returned results and exact solutions, and the results show that the tradeoff between the accuracy and the space overhead is acceptable in practice.

Keywords: data stream; extreme aggregation; sampling

近年来，实时数据处理的需求日益增加，数据流的相关研究和应用迅速发展[1-2]. 大数据环境下的传感网络、金融、医疗、交通和军事等领域的许多关键应用，其处理过程表现为连续实时的数据流查询，其中聚集查询是数据流处理中开销最大、实现最复杂的操作[3]. 数据流在时间滑动窗口上的极值聚集

收稿日期：2012-06-30. 作者简介：丁维龙(1983—)，男，博士生；韩燕波(通信作者)，男，教授，博士生导师. 基金项目：国家自然科学基金资助项目(60970132，61003294).
网络出版时间：2012-08-23 网络出版地址：http://www.cnki.net/kcms/detail/61.1069.T.20120823.1805.001.html

(MAX/MIN)方法可广泛应用于业务活动的例行监控分析和异常监测预警，如工厂车间中统计每天最高温度的时段、基于位置的服务(LBS)定位过去 1 个月内降水量最低的地域[4]等.

时间滑动窗口是数据流聚集最常用的一类技术，在固定的窗口长度(窗口的时间跨度)下，窗口规模(窗口的数据数量)的变化与输入的速率和数值分布相关. 但是，在许多极端情况下，如数据高速高并发到达时，窗口可迅速积累大量数据，导致极值聚集计算的困难. 一方面，传统方法的目标是获得查询的精确解[3-6]，为保证结果连续，需要维护大量的极值候选值，空间开销将会骤增；另一方面，在云计算等大数据环境下的分布式处理技术中，单机(如工作节点)的空间开销直接影响处理任务的响应时间和系统整体的吞吐量. 总之，如何权衡空间开销和计算精度，以满足数据流处理实时性的需求，成为当前众多大数据环境下应用的挑战.

本文设计并实现了内存有界的摘要数据结构及极值聚集算法，可维护时间窗口内有限数量的极值候选，连续返回查询的近似解，以满足实时数据处理过程低延迟、高吞吐的需求. 若无直接说明，文中的窗口均指时间滑动窗口，每个数据项的时间戳以递增的长整型描述. 同时，由于 MAX 聚集与 MIN 聚集查询结果的对称性和实现方式的相似性[3-6]，本文的极值查询都以 MAX 聚集为例阐释，MIN 聚集可以类比实现.

1　内存有界的极值聚集方法

1.1　研究背景

堆(heap)是极值聚集最常用的一类传统的数据结构. 以二叉大根堆(Binary Max-heap)为例，该结构是一种递归定义的特殊二叉树，存在如下的性质[7]：堆中最大的元素一定位于树结构的根；每个节点一定是以它为根的子树结构的最大值节点. 堆的维护操作有良好的时间复杂度，在最复杂的排序操作下为 $O(N\ln N)$，适合离线数据处理(如数据库、数据仓库). 但是，堆需要维护所有输入数据，即空间复杂度为 $O(N)$，而这样的空间开销对实时连续的数据输入，特别是在高速高并发环境下，显然是不能接受的.

针对极值聚集空间开销过大的问题，Peng 等人[6]提出了空间剪枝技术，只存储那些可能或将要成为极值的数据项，称之为关键点集. 对于关键点集中的任何 2 个数据项，值大于的偏序关系与时间先于的偏序关系这 2 个条件是不能同时满足的. 该文使用链表实现了相应的摘要数据结构，为方便本文后面的对比分析，称其为极值链表，记作 Ex-link. 该文已证明，其数据结构是精确计算极值聚集所需的最小空间，在数据均匀离散分布的情况下，这个最小空间开销的数学期望为 $O(\ln N)$. 同理，也可证明维护该结构的时间复杂度为 $O(\ln N)$. 但是，这种精确求解的方法在最坏的情况下，如输入值递减分布时，空间开销仍然是 $O(N)$，这对许多数据高速积累的实际情况并不合适.

正是由于上述原因，极值聚集的非精确求解方法成为研究热点.

1.2　摘要数据结构 REx-link 与极值聚集

为降低空间开销，实现数据流上连续的极值聚集，本文设计了一种在内存中维护的摘要数据结构，称为随机版的极值链表 REx-link. 与文献[6]中的 Ex-link 不同的是，REx-link 的链表最多由 $(k+1)$ 个候选数据项组成，k 是根据可用内存空间在构造时设置的常数. 空间开销 S(REx-link)由下式表示

$$S(\text{REx-link})=\begin{cases}N(t), & N(t)<k+1\\ k+1, & \text{otherwise}\end{cases}$$

式中：$N(t)$ 表示 t 时刻的窗口规模. 如图 1 所示，REx-link 是空间有界的摘要数据结构. 窗口中的数据项(tuple)在链表中封装为节点，称为候选项(Item)，其中表头候选项称为极值项(exItem). 极值聚集返回的就是 exItem 的数据值；当前 exItem 过期时，其连接的下一候选项成为新的极值项，并丢弃原有的极值项，即 exItem＝exItem. next. 数据项的到达和过期，使得 REx-link 维护的候选项是动态变化的，而正是由于存在极值项的候选，才使查询可以连续返回极值聚集的结果，这与确定性算法的极值链表相同.

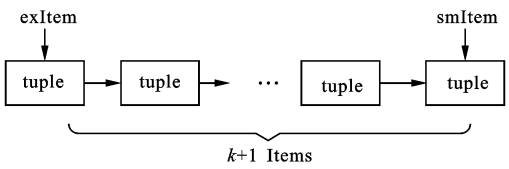


图 1　REx-link 数据结构

当窗口规模大于 $k+1$ 时，REx-link 通过随机抽样，将新入样的数据项替换链表的最后一个候选项，该项被称作链表的随机项(smItem). 这个过程是在新数据项 p 到达时，由图 2 描述的算法通过维护 REx-link 结构实现的.

算法需要维护全局变量窗口规模 N 和链表长

度 lsize. 算法运行前，$N=0$，lsize=0，并设置 exItem =new Item((−1) * Math. exp(32))，以保证 REx-link 中初始极值项的值足够小. 在算法的第 11～14 行，当 p 是第一个到达或者是比 exItem 大的数据时，将作为新的极值项. 当链表长度不足 $k+1$ 且 REx-link 中存在比 p 小的候选项时，根据算法的第 22 行，将新数据项 p 作为候选项，替换链表中所有比 p 小的候选项；否则，按照算法第 15～22 行，当 REx-link 中不存在比 p 小的候选项、且链表长度到达 $k+1$ 时，到达的数据项 p 被随机抽样，入样后将在链表中维护. 其中，random()函数返回[0,1]上概率均匀的随机数，故第 16～22 行实现了以 $(1/N(t))$ 的概率将新数据项入样，同时链表第 $k+1$ 项以同样的概率被 p 替换(见算法第 22 行).

```
arrivalTuple(Tuple p)
1  Item * tmpItem=exItem, * preItem= * kItem=null;
2  N++; i=0;
3  while(p. value<tmpItem. value)
4     if (i==k)
5        kItem=tmpItem;
6     end if
7     i++;
8     preItem=tmpItem;
9     tmpItem=tmpItem. next;
10 end while
11 if (preItem==null)
12    exItem=new Item(p); lsize=1;
13    return;
14 end if
15 if (lsize==k+1 && tmpItem==null)
16    index=(int)Math. floor((N) * random(N));
17    smItem=kItem. next;
18    if (index==k+1)
19       preItem=smItem; lsize=k+1;
20    end if
21 end if
22 preItem. next=new Item(p); lsize++;
```

图 2 维护 REx-link 的随机算法

可以看出，图 2 中的算法使得链表按数据项的时间戳递增、且值递减的顺序维护候选项，与确定性算法版本的极值链表相同. 但是，REx-link 链表的最后一个候选项可被新入样的数据项随机替换，而前 k 个候选项除非因为过期或值更大的数据项到达，否则不会在 REx-link 中丢弃.

1.3 理论分析与证明

时间滑动窗口的特点使得基于 REx-link 极值聚集的效果与输入数据的分布相关. 下面以输入数据值均匀分布为例，给出 2 个定理，从理论上分析该方法的有效性.

定理 1 基于 REx-link 的极值聚集是空间开销有界的，在输入数值均匀分布时，能以 $\frac{1}{N}(k+\ln N)$ 的概率精确地返回极值.

证明 由 1.2 节中分析的数据结构可知，本文方法利用随机算法，通过 REx-link 结构维护最多 $k+1$ 个极值候选项，故是空间开销有界的. 以下证明方法返回极值精确解的概率.

假设窗口规模为 N，含有数据项 $p_1, p_2, \cdots, p_i, \cdots p_N$；随机变量 $Z_i=1$ 表示数据项 p_i 是最大值且在链表的 $k+1$ 个候选项中(即入样). $Z_i=X_i \cap Y_i$，即 Z_i 对应的事件由 2 个子事件组成：X_i 表示 p_i 是窗口中最大值的子事件；Y_i 表示 p_i 入样的子事件. 令

$$X_i=\begin{cases}1, & p_i \text{ is max in } N\\ 0, & \text{otherwise}\end{cases};$$

$$Y_i=\begin{cases}1, & p_i \text{ is in samples}\\ 0, & \text{otherwise}\end{cases}$$

各子事件发生的概率为

$$P(X_i=1)=\frac{1}{N};$$

$$P(Y_i=1)=\begin{cases}1, & i\leqslant k\\ 1/(i-k), & k<i\leqslant N\end{cases}$$

于是得到伯努利随机变量 $Z_i=1$ 时相应事件发生的概率

$$P(Z_i=1)=P((X_i=1)\cap(Y_i=1))=\begin{cases}1/N, & i\leqslant k\\ 1/(N(i-k)), & k<i\leqslant N\end{cases}$$

同时，事件 Z_i 之间相互独立，所以返回精确解的概率为

$$\bigcup_{i=1}^{N} P(Z_i=1)=\sum_{i=1}^{N}P((X_i=1)\cap(Y_i=1))=\sum_{i=1}^{k}\frac{1}{N}+\sum_{i=k+1}^{N}\frac{1}{N(i-k)}=$$

$$\frac{k}{N}+\frac{1}{N}\sum_{i=k+1}^{N}\frac{1}{(i-k)}=\frac{k}{N}+H_N\geqslant\frac{1}{N}(k+\ln N)$$

式中：$H_N=\sum_{i=1}^{N}(1/i)$ 为调和级数. 上式最后一步取不等号的原因是 $H_N=O(\ln N)+\Theta(1)$.

证毕.

当输入数据项的值均匀分布时，定理 1 证明了基于 REx-link 的随机化算法可以返回极值精确解的概率的下界，且这个概率随着样本数量 k(准确说

是 $k+1$)的增加而增大.

定理 2　采用基于 REx-link 的极值聚集方法，在输入数值均匀分布时，抽样始终不能获得精确极值的概率小于 $\exp\left[-\frac{1}{2}(k+\ln N)\delta^2\right]$，其中 $0<\delta<1$ 是任意小的常数.

证明　定义随机变量 $Z=\sum_{i=1}^{N}Z_i$ 是一系列伯努利随机变量 Z_i(在定理 1 的证明中已定义)的和. Z 的期望为 μ，其直观含义是该方法返回精确解的次数在概率上的平均值

$$\mu=E(Z)=\sum_{i=1}^{N}P(Z_i)=\frac{1}{N}(k+\ln N)$$

相互独立的事件 Z_i 可以看作一系列泊松试验，则对于任意小的常数 $0<\delta<1$，可以由 Chernoff 边界得到

$$P(Z\leqslant(1-\delta)\mu)\leqslant\exp(-\mu\delta^2/2)$$

代入 μ，同时由于 δ 可任意小，抽样始终不能获得精确极值的事件 E_r 出现的概率

$$P(E_r=1)\leqslant P(Z\leqslant(1-\delta)(k+\ln N))\leqslant\exp[-(k+\ln N)\delta^2/2]$$

证毕.

由定理 2 同样可知，随着样本数 $k+1$ 的增大，方法失效的概率降低. 可以证明，在此条件下该方法的时间复杂度的期望为 $O(k)$. 根据以上的讨论，基于 REx-link 的极值聚集方法实际上是在空间开销和计算精度之间的折中，其与求解极值聚集的二叉大根堆(Binary Max-heap，简称为 BM-heap)和 Ex-link 算法的性能对比如表 1 所示. 前面的理论分析已表明，本文提出的基于 REx-link 的极值聚集方法的计算精度损失是有下界的，并且失效的概率始终较小，因而适用于许多实际应用场景.

表 1　极值聚集相关的数据结构及其操作时间复杂度和存储空间复杂度

数据结构	时间复杂度	空间复杂度	精度/%
BM-heap	$O(N\ln N)$	$O(N)$	100
Ex-link	$O(\ln N)$	$O(\ln N)$	100
REx-link	$O(k)$	$O(k)$	$\frac{k+\ln N}{N}$

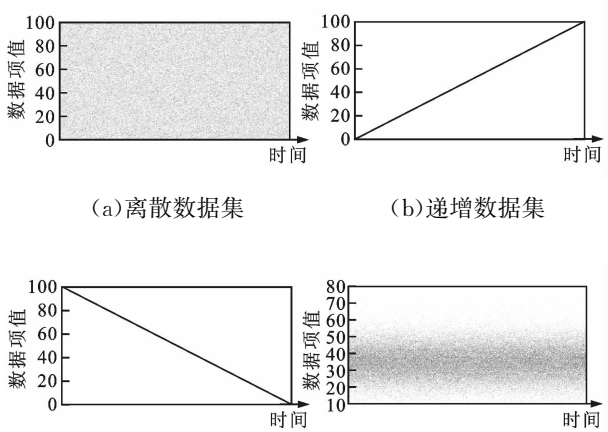
注：REx-link 给出的是在输入数值均匀分布且匀速到达条件下的精度；k 是与样本数量有关的常量；N 为任意时刻数据匀速到达时的窗口规模.

2　实验与分析

我们用 C/C++实现了 REx-link 摘要数据结构，以及基于该结构的随机化最大值聚集算法；为了对比效果，基于同样的开发环境实现了确定算法的 Ex-link. 本文将设计实验来定性和定量地分析本文算法的性能. 实验均在 DELL PowerEdge 机架服务器(AMD Opteron 16 核 CPU，32 GB DDR2 RAM，500 GB SATA 硬盘，CentOS 5.5 x86 64 bit 操作系统)上进行. 实现 Ex-link 与 REx-link 的模块各自部署于独立同配置的服务器，分别完成数据流极值聚集.

2.1　实验设置

前文已经分析过，基于时间滑动窗口的聚集的效果与输入数值的分布有关，所以本文设计了 4 种数据集用于实验，其中 3 种数据集是人工数据，按照数值随时间变化的关系分为均匀离散、递增和递减型，另一种数据集是真实数据，来自某市网格交通的车速监控现场. 4 种数据集均含有 10^5 个数据项，这个规模对应现场环境下 50 s 左右累积的数据. 4 种数据集随到达时间的数值分布如图 3 所示：图 3a～3c 为人工数据集，值域为[0，100]；图 3d 为真实数据集，值域为[10，75]，且大多分布于 20～50 之间.



(a)离散数据集　(b)递增数据集

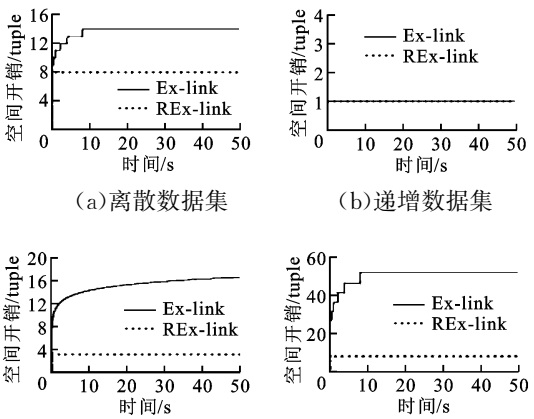
(c)递减数据集　(d)真实数据集

图 3　4 种数据集的数值分布

2.2　空间开销与精度的折中

实验 1　配置我们开发的前端数据发生器，以确定的速率将 4 种数据集发送至部署 Ex-link 和 REx-link 的服务器. 实验中结构参数 k 设置为 8 tuple，时间滑动窗口长度设置为 10 s，数据发送速率为 2 000 tuple/s，故窗口规模 N 为常数 2×10^4. 2 种方法的空间开销对比如图 4 所示.

基于 REx-link 的极值聚集算法在 4 种数据集下的空间开销都随着时间和数据累积趋向于常数. 确定性的 Ex-link 算法对均匀离散输入(见图 3a)的空间开销基本符合 $O(\ln N)$,这也验证了前文的分析. 在 2 种极端的情况下,Ex-link 算法对递增输入(见图 3b)呈现最优的空间开销,与 REx-link 算法的一样同为常数;对递减输入(见图 3c)的空间开销最差,与数据规模呈现线性关系,与 REx-link 算法的差异很大. 为了能够清晰呈现实验效果,图 4c 采用了对数坐标作为纵坐标. Ex-link 算法对实际输入(见图 3d)的空间开销介于最优与最差之间,与离散输入的情况类似.



(a)离散数据集 (b)递增数据集

(c)递减数据集 (d)真实数据集

图 4 实验的空间开销

多次重复上述实验,统计 REx-link 算法返回精确极值的概率,结果见表 2. 本实验设置 N 和 k 为常数,且 k 设置得不大,这使得实验精确返回极值的概率也不大. 但是,在许多实际情况、甚至是极端情形下,以有限的空间开销换取低延迟的连续非精确解,这种空间和精度的折中是值得和可以接受的.

表 2 基于 REx-link 极值聚集的计算精度

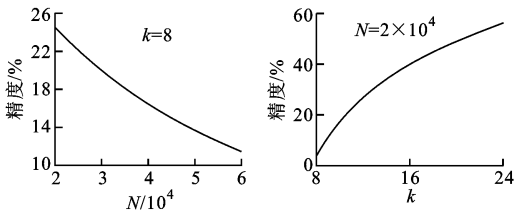
数据类型	离散	递增	递减	真实
计算精度/%	22.51	100	7.83	13.92

2.3 窗口规模与结构参数对计算精度的影响

定理 1 说明,REx-link 算法返回精确解的概率与窗口规模 N 和结构参数 k 有关. 下面在实验 1 的基础上,通过改变参数 N 和 k 来分析这 2 个因素的影响.

实验 2 其他配置不变,改变数据发送速率,从而改变 10 s 窗口的窗口规模,将人工均匀离散的数据集发送至部署 REx-link 的服务器,观察返回结果的精度随窗口规模的变化. 如图 5a 所示,设置 $k=8$,并将数据发送速率逐渐变大,使得窗口规模由 2×10^4 扩大至 6×10^4 时,REx-link 算法返回精确解的概率快速下降,特别是在窗口规模超过 6×10^4 之后. 这说明,$k=8$ 的抽样(9 个样本)难以大概率地获得精确解,需要增大结构参数来提高精度.

实验 3 其他配置不变,将真实数据集发送至部署 REx-link 的服务器,并不断变化结构参数 k,观察返回结果的精度随 k 的变化. 如图 5b 所示,当 $N=2\times10^4$ 并逐渐增大 k 时,REx-link 算法返回精确解的概率也随之增大并保持稳定,说明 k 直接影响计算精度. 这是因为在 N 足够大时,窗口规模仍然是影响精度的主要因素.



(a)精度随窗口规模的变化 (b)精度随结构参数的变化

图 5 REx-link 算法的计算精度随窗口规模和结构参数的变化

3 总结与展望

本文针对时间滑动窗口设计实现了一种随机化的极值聚集方法,以有限的空间开销保证实时返回连续结果. 本文的贡献在于:

(1)设计了一种空间有界的摘要数据结构 REx-link,可以在有限的内存中,基于随机抽样进行构建和维护,实现时间滑动窗口上的数据流极值聚集;

(2)从理论上证明了这种方法的相关概率边界,并通过实验进行了定性和定量分析,验证了方法的有效性.

事实上,作为一种专用的聚集技术,本文方法可以在分布式数据流处理系统中获取处理节点的聚集特征,实现极值聚集节点低延迟和可信的故障恢复,这也是我们进一步的工作方向.

参考文献:

[1] CUGOLA G, MARGARA A. Processing flows of information: from data stream to complex event processing [R]. Politecnico di Milano, Italy: Dip. di Electronica e Informazione, 2010.

[2] RAJARAMAN A, ULLMAN J. Mining of massive datasets [M]. Cambridge, UK: Cambridge University

Press, 2011.

[3] LIU Zhenyu, SIA K C, CHO J. Cost-efficient processing of MIN/MAX queries over distributed sensors with uncertainty [C]// Proceedings of the 2005 ACM Symposium on Applied Computing. New York, USA: ACM, 2005: 634-641.

[4] SILBERSTEIN A, MUNAGALA K, YANG Jun. Energy-efficient monitoring of extreme values in sensor networks [C]// Proceedings of the 2006 ACM SIGMOD International Conference on Management of Data. New York, USA: ACM, 2006: 169-180.

[5] DENG Hanlin, ZHANG Baoxian, LI Cheng, et al. MAX-MIN aggregation in wireless sensor networks: mechanism and modeling [J]. Wireless Communications and Mobile Computing, 2010, 12: 615-630.

[6] PENG Xiaoming, HE Yanxiang, TIAN Li. Communication reduction for continuous extreme values monitoring over distributed data streams [C]// Proceedings of the Workshop on Power Electronics and Intelligent Transportation System. Washington, DC, USA: IEEE Computer Society, 2008: 181-187.

[7] CORMEN T, LEISERSON C, RIVEST R, et al. Introduction to algorithms [M]. Cambridge, MA, USA: The MIT Press, 2001.

[本刊相关文献链接]

杨清宇，孙凤伟，张墨，等. 利用测地线距离的改进谱聚类算法. 2012, 46(8): 1-7.

司刚全，娄勇，张寅松. 鲁棒最小二乘支持向量机及其在软测量中的应用. 2012, 46(8): 15-21.

梁炎明，刘丁. 规则递归 T-S 模糊模型及其辨识方法. 2012, 46(8): 54-58.

黄伯虎，张海宾，王小兵，等. 移动环境中的位置依赖连续轮廓查询. 2012, 46(6): 79-86.

陈衡，钱德沛，伍卫国，等. 传感器网络基于邻居信息量化的能量平衡路由. 2012, 46(4): 1-6.

张慧，韩崇昭，闫小喜. 概率假设密度滤波的谱聚类目标状态提取方法. 2012, 46(2): 1-5.

丁要军，蔡皖东. 采用两阶段策略模型(KTSVM)的 P2P 流量识别方法. 2012, 46(2): 45-50.

聂艳明，李战怀，陈群. 针对不确定射频识别数据流的改进概率推导方法. 2011, 45(12): 45-52.

薛峰，周亚东，高峰，等. 一种突发性热点话题在线发现与跟踪方法. 2011, 45(12): 64-69.

刘弹，杨景明，罗爱玲. 具有全局聚类的多属性离散化算法. 2011, 45(9): 1-5.

王羡慧，覃征，张选平，等. 采用仿射传播的聚类集成算法. 2011, 45(8): 1-6.

彭柳青，张军英. 一种鲁棒的子空间聚类算法. 2011, 45(6): 13-19.

张熠卓，徐光华，梁霖，等. 利用增量式非线性流形学习的状态监测方法. 2011, 45(1): 64-68.

史宝全，梁晋，张晓强，等. 特征保持的点云精简技术研究. 2010, 44(11): 37-40.

赵海祥，伍卫国，赵增，等. 一种应用于远程并行程序调试系统的新型消息聚集机制. 2009, 43(10): 27-31.

(编辑　葛赵青　赵大良)