流数据管理的降载技术:研究进展

李卫民　于守健　骆轶姝　乐嘉锦
(东华大学计算机科学与技术学院　上海 200051)

摘　要　在当今的网络监控、通信服务、传感网络和金融服务等应用领域中产生了大量的流数据。当流数据的输入速率超过系统的容量,系统性能就会下降。在这种情形下,降载虽然降低了结果的准确性,但可以改善系统的性能。本文讨论了流数据降载的研究现状,对相关技术进行了总结和对比。降载的关键是确定降载的方式和实施的降载计划,本文把降载方式概括为语义降载、随机降载和自适应降载,降载计划包含降载的时间、降载的位置和降载的量等。讨论了面临的挑战和未来研究的发展方向。

关键词　流数据,QoS,随机降载,语义降载,自适应降载

Load Shedding in Data Stream Manager System

LI Wei-Min　YU Shou-Jian　LUO Yi-Shu　LE Jia-Jin
(Information Science and Technology Institute, Donghua University, Shanghai 200051)

Abstract　Data streams arise in a number of monitoring applications in domain such as networking, telephone, sensor networks and financial services. Various Quality-of-Service (QoS) requirements should be meet in the process of querying in Data Stream Management Systems(DSMS). When input rates exceed system capacity, the system will become overloaded and cannot meet quality requirement. Under these conditions, the system will shed load to improve the observed latency of the results. This paper discusses the load shedding technology, and concludes that the load shedding should contain the mode and plan. This paper proposes that the mode of load shedding includes random shedding, semantic shedding and adaptive shedding. The plan of load shedding contains when, where to shed load and how much load to shed. Finally, some of the important open issues are addressed.

Keywords　Data stream, QoS, Random load shedding, Semantic load shedding, Adaptive load shedding

1　引言

1.1　流数据

传统的数据库存储的是静态的关系型数据记录的集合,用于需要持久的数据存储和复杂查询。查询比插入、更新、删除等操作频繁,查询的结果反映的是当前数据库的状态。目前,在许多应用领域(如网络监控、通信服务、传感网络和金融服务等)产生一种新型的数据——流数据,这些数据以网络性能参数、电话记录、传感器的读入值和金融数据等体现,它们是实时的、连续的、无限的,是按照数据项到达的顺序控制的、一组有序的点 $x_1, x_2, \cdots, x_n$ 的序列。与传统的数据相比较,流数据的特点表现在:量大、连续性、实时性,随机存取采用的是单一线性数据扫描,完整地将数据流存储到本地是不可行的。对这些流数据进行存储、查询、处理是研究的关键所在,而流数据管理中出现的过载现象将对存储、查询、处理产生很大的影响。

针对流数据的特点,设计开发一个用于管理流数据的系统成为必然。流数据管理系统(DSMS)允许所有数据以连续的流数据的形式存在,可以处理连续的、无限的、时变的流数据。典型的流数据管理系统原型有斯坦福大学的 STREAM 系统[1],麻省理工大学、布朗大学和布兰代斯大学联合开发的 Aurora[2],以及加州大学伯克利分校的 TelegraphCQ[3]等。

1.2　流数据的降载

以上所有应用产生的流数据都有如下特征:数据量大,其到达的速率高且不可预测。当数据的速率及其量超过系统的接受能力,系统不能做出正确的行为。而在处理多个流中,资源分配是个问题,考虑到流数据的量大且高的速率,算法必须处理系统超负载带来的影响。在这种情形下,唯一的方案是卸掉部分负载。降载就是处理掉系统容纳不了的负载。要使系统的性能适度下降,识别并处理掉不重要的数据,对于降载很是关键。也就是在资源有限的情形下如何获得更好的结果,这也是处理流数据降载面临的挑战。

2　流数据降载的关键技术

DSMS 的降载与网络的拥塞控制问题是相关的。网络中的降载是根据时间戳或优先权随机地丢弃个别的包。网络降载和 DSMS 降载尽管在概念上相同,但仍有许多不同的地方。DSMS 不像网络降载有固有的分布性,考虑到整个系统状态,可以做出更多的智能降载。

文[4]根据目前的负载要求,提出一种在查询计划中插入或移除降载操作的技术。这种技术采用两种基于 QoS 的降载算法:随机选择删除比例的随机降载和根据元组内容的重要性进行删除的语义降载。并提出了降载需要解决的 3 个问题:降载的时间、降载的位置以及降载的量。文[5]重点放在通过降载而达到适度降低性能的适应性,对流数据上的聚集查询操作遇到的过载进行了研究,提出了决定在查询计划中某点进行降载和在某点降多少的算法。该算法使由降载带来的查询结果的不准确程度降到最低。文[6,15]设计了一个保证查询质量的质量适应框架,这个框架建立在控制理论的基础之上,由于处理的开销和流数据到达的速率不可预测,质量适应框架会在目前系统状态下调整应用行为,从而保证查询结果的质量。文[7]提出根据降载计划如何对删除的元组进

李卫民　博士生,主要研究方向为流数据查询处理技术、数据挖掘。

行分类，并决定在什么时候在一个数据流上进行删除。当数据的特征值在降载时不能得到，文[7]引入了决策服务(QoD)来度量分类中的不确定水平，并建立 Markov 模型来预测特征值的分布。这种降载计划不仅具有学习功能，而且能自适应地改变流数据的数据特征。而文[8]主要是通过捕捉流数据的峰值行为来决定降载，然后通过在系统中加入数据触发器，获得峰值下较准确的、较满意的结果。

通过上述文献提出各种降载策略，我们可以把流数据管理中的降载方式分为 3 种：随机降载、语义降载和自适应降载。对于每种降载方式产生的降载计划包含 3 个内容：降载的时间、降载的位置和降载的量。

2.1 流数据降载方式

2.1.1 随机降载

文[2]使用贪婪算法来执行降载。主要使用基于丢弃的 QoS 信息来描述静态的降载算法。首先是利用相应 QoS 图的最小负斜率来判断输出，沿着曲线水平移动，直到 QoS 曲线的另外输出在那点有更小的负斜率。这个水平的不同，给我们暗示了要丢弃的输出元组。如插入丢弃操作的选择率，最终使得整个 QoS 减小最小化。然后移动这个丢弃操作，使它尽可能离输入较远，而且会影响到其他的输出，并把它放在这个点上。此时，对恢复的资源量进行计算估计。如果系统的资源仍然不够，那么将重复这个过程。

在实时情况下，算法是一样的，只是我们要使用基于延迟的 QoS 图来判断存在问题的输出。如那些超出延迟极限的输出，我们将重复降载过程，直到等待时间的设定目标达到为止。

2.1.2 语义降载

上面是通过在网络的某点随机地选择丢弃元组进行丢弃。当用这个方法来使整个系统的效用损失达到最小时，却不能控制由丢弃元组而产生的对应用语义的影响。语义降载通过使用基于值的 QoS 信息来对上述的影响进行界定。应该说，语义降载是一种运用可控的方法来丢弃元组，它是使用过滤技术丢弃相对不重要的元组，而不是随机地丢弃元组。

如果可以得到基于值的 QoS 的信息，就可以监视每一个输出，建立一个在值区间内包含值的频率的直方图。为了进行降载，通过最低效用区间来判断输出，把这个区间转换成一个过滤预测，然后把这个相应的过滤操作扩展到一个尽可能远离输入的分支点。

2.1.3 自适应降载

和传统 DBMS 处理快照查询不一样，流数据管理系统的连续查询处理需要满足质量要求。由于处理的开销和数据到达的速率不可预测，保证查询的质量是个难题。文[4]提出了一个质量自适应框架，通过这个框架来调整目前系统状态的应用行为。而在这质量自适应框架中起杠杆作用的是控制理论。

在 DSMS，由于物理资源的有限性和应用行为的不稳定性，对多流、多查询的质量保证是比较困难的。一个流数据系统包含大量的流和连续查询。当系统处于过载状态，质量就会下降。即使利用传统控制方法，存在资源(连接网络的带宽)和使用资源(峰值)会产生暂时性的过载。为了处理这些变量，DSMS 需要自动调整单个实时查询的质量。

流数据本身是动态的，看不出明显的周期性，而且数据元组的处理开销很难预测，所以很难做出目前和将来的资源消耗的估计很难做出。文[6,15]提出了用经典控制理论来解决上述问题，对 DSMS 质量自适应性问题形式化，成为一个反馈控制问题。对 DSMS 行为进行建模是较困难的，文[6,15]使用控制理论来估计模型，而且形式化了自适应决策。目前 DSMS 中的研究主要是对一些质量参数(如查询的精确、损失容差)进行最大化，很少研究质量自适应性。

文[7]提出的降载策略是具有学习功能的自适应降载，引入了决策服务(QoD)来度量分类中的不确定水平，并建立 Markov 模型来预测特征值的分布，能自适应地改变流数据的数据特征。

2.2 流数据的降载计划

2.2.1 降载的时间

在决定已采用什么样的降载方式后，就需要确定降载的时间、地点和量，并获得一个降载计划列表。

不同的 DSMS 系统所用的检测负载方法是不同的，在 Aurora[4] 中，当前查询网的负载情况可以用很简单的公式来计算估计。估计负载涉及到的参数有：当前的流数据的输入速率、操作的执行代价及选择率。网络中每一个输入与可以静态计算的负载系数关联起来。负载系数代表的是处理的周期数量，周期是指单一输入元组通过网络并到达输出。这些负载系数都可以通过静态的计算获得。如果计算出的负载值超出了系统的处理能力，那么就需要降载。

在文[5]中提到了处理代价模型。在若干个数据流上的查询集合与操作的集合构成了一个有向非循环数据流图。在 STREAM 中，模型用到了一些参数值：操作的选择率、执行代价、流的速率等，这些参数由一个统计管理器进行综合评估。在 STREAM 中，降载的目的是提高系统的吞吐能力。被处理的元组的速度至少和新到达的元组的速率一样。如果这种关系不能保持，系统就不能接受新到达的流数据。输入缓冲和响应的等待时间将无极限地上升。STREAM 将这个要求表示成一个方程，就是所谓的负载方程。如果降载的开销可以忽略不计，那么方程表示的是单位时间内到达的元组被系统处理所花费的全部时间。

在 STREAM 中，负载方程所表示的单位时间内到达的元组与 Aurora 中表示的当前的流数据的速度在内涵上是一致的，都是首先对目前的负载进行评估。STREAM 是用统计管理器进行评估计算，Aurora 是对处理元组的周期数进行计算，最后都是与系统的处理能力进行比较。如果超出系统的处理能力，就说明过载，需要进行降载。

2.2.2 降载的位置

如果及时判断出系统处于过载状态，那么就应该插入降载操作来对负载进行降载。但降载操作应该插入什么位置，对系统的性能也有直接的影响。如果插入到过早的位置，会影响到多个输出(单一查询除外)。如果插入到过晚的位置，降载的效果肯定会达不到，因为中间会出现许多不必要的工作。

在 Aurora 系统[4] 中，降载是基于 QoS 信息的。对于一个无共享查询，一个网络中包括这样一个计划：降载操作可以插入到查询网的任何一个位置，但不影响其它查询。降载操作越靠近输入端，就越减少“下游”操作的数据量，当然就减少了这些操作的工作量，处理周期的数量也就少了。因此，如果我们想要在一个没有共享查询的查询计划中插入降载操作，降载操作一般放在查询计划开始的位置。对于一个有共享查询的查询网络，当一个操作输出多个“下游”操作，共享就出现了。插入到靠近输入端的操作将会影响到多个输出。在 Au-

rora 系统中，对候选的降载位置将计算损失/收益比例，然后对计算的损失/收益比例进行排序，最小的比值就是降载操作插入的最佳位置。

在文[5]中，系统可计算出准确的、有效的抽样比例 P_i。如果在查询中没有共享操作，优先的方案是在每个查询的查询路径中第一个操作前面插入降载操作，且降载操作的抽样比与这查询的抽样比相同。如果查询中有共享操作，这时要插入降载操作就较为复杂。此时对两查询都有影响应当是在查询路径的共享部分插入降载操作。通过预先设置的三条规则确定降载的位置和数量。

通过两个系统的降载位置的定位比较，可以得到，在非共享查询计划中，都是在输入处插入降载操作，因为这样一开始就避免了不必要的工作。对于共享查询，Aurora 系统是通过计算候选降载位置的损失收益比，选择其中值最小的位置。STREAM 系统首先在共享的起始处降载，其抽样比等于所有查询中最大的有效抽样比，然后再根据每个查询的有效抽样比来决定插入降载操作的位置和数量。

2.2.3 降载的量

降载的时间和降载的位置都已确定后，下一步就需要确定降载的数量，即确定有多少元组丢弃。降载应保证输出速度最大。根据每个流数据的输入速度、操作的选择率和降载操作的抽样比等参数可计算出降载后输出的速度。决定降载的量，实际是在确保系统不超载情况下，选择合适的降载抽样比，使输出速度最大。由于降载是删除未处理的元组，故降载会对查询结果的准确性产生影响，也就是所谓的近似查询。

文[5]是将查询结果的最大误差最小化，文中也证明了在最好的解决方案中，所有查询的相对误差是相等的。通过自顶向下的算法与负载方程，计算出相对误差值，然后得出每个查询的有效抽样比 P_i，就可确定降载的数量。

文[4]是通过保证系统在丢弃部分元组后，其收益大于损失(也就是获得的周期数大于降载操作本身开销)，确定降载的位置与降载的数量。

2.3 分布式流的降载

文[9]研究了 Borealis 分布式流处理系统。对于大规模的动态系统，当负载以不期望的速度增加，过载的管理变得非常重要。许多 DSMS 系统开发了不同的降载技术，如自适应负载分布、传统控制和降载技术。而技术选择将依赖负载的特征、定义资源分配的协议和应用的需求。考虑到分布式流处理系统存在过载管理的问题，在这种环境下，大量的连续查询以操作链集合的形式分布到多个服务器。这些查询本质上能接受和处理来自外部数据源的连续的流数据。实时监听系统就适合这种系统。在这个领域，提供低等待时间、高吞吐量的查询结果非常重要。

突然到来的流数据会在服务链的某点产生瓶颈，当对服务器不足的处理能力和带宽提出额外的需求时，瓶颈就会出现。文[4,5,8]对流数据处理系统提出了不同的降载技术，这些技术都是针对单一服务器的方案。然而在分布式流处理系统，每一个服务器节点相对于它的“下游”节点是一个负载生成器，因此，在某点的资源管理决策将影响到它的后继节点负载的特征。降载技术的目的是在服务链的某点删除元组来减少负载。和 TCP 拥塞控制不同的是，这里不存在重发，删除的元组也不能恢复。由于负载是在两节点之间，一个给定的节点的降载技术必须考虑到后继节点受到的影响。这也是分布式流降载面临的问题。

3 流数据降载技术的相关研究进展

3.1 AROURA

布朗大学、布兰代斯大学和麻省理工大学联合开发的 Aurora 系统是构建一个新型的流数据处理系统，它的目标是专门处理流式监控。Aurora 系统的核心是一个巨大的触发器网络。每个触发器是一个数据流向图，图中每一个节点则是 7 种 Built-in 操作中的一个。Aurora 流数据管理系统[4,10]使用应用水平语义来做出关于资源定位的智能决策(如图 1)。Aurora 包含：网络拓扑、输入、输出、QoS、相关统计(选择率、一个操作平均处理的开销)。每个输出都和 QoS(服务质量)说明联系起来。在 Aurora 中的 QoS 通过 3 个函数和等待时间图表明删除元组作为一个结果，系统延迟的效用如何？基于值的图表明输出空间的值的重要性；删除容差图：是一个描述应用到近似结果的不适合程度。例如 100%的效用意味着 0 丢失。

Aurora 中的降载组件在数据率上接受信息，从目录中读取网络的描述。它检测系统的负荷，通过在运行的查询网络中插入负荷减少的降载操作来降低负载。查询计划的改变存入目录中。降载器也能决定降载操作什么时间执行是必要的。

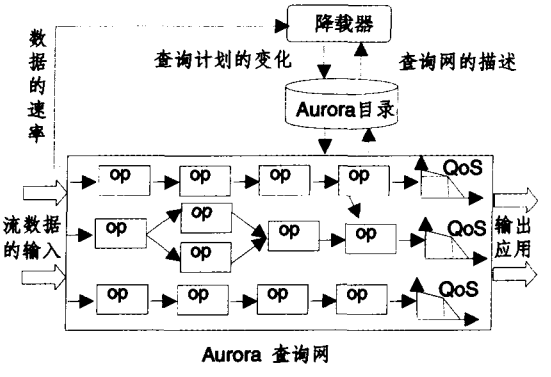

图 1 Aurora 降载框架

3.2 STREAM

斯坦福大学已经开始了一种全面 DSMS 的设计和原型实现，该系统为 STREAM(Stanford Stream Data Manager)[5]。由于流数据经常突发且数据的特性随时间而变化，故处理流数据连续查询的系统必须是自适应的[5]。STREAM 通过降载来达到适当的性能降低。把降载作为一个优化问题来处理，目标函数是查询结果不准性达到最小。其降载集中在聚集查询上，并提出了相应的降载算法，内容包括降载应放在查询计划的什么地方、在某点降载的量应是多少及查询结果的不准确性程度应最小化。另外，降载技术是在查询计划中引入随机抽样操作，每个降载器通过抽样比 p 进行参数化。概率 p 是通过本操作并到下个操作的元组比。为了补偿由于引入降载器而产生的丢弃元组，由系统计算的聚集值的适当比例产生无偏近似答案。

如图 2 所示，操作排列成一具数据图。降载问题的输入是由 $S_1,\cdots,S_m$、流数据上的查询集合 $q_1,\cdots,q_n$、查询操作集合 $O_1,\cdots,O_k$ 以及相关的统计管理器组成。降载还涉及到参数元组平均到达速率 r_j 和流数据 S_j。在 STREAM 系统中，统计管理模块是对这些参数值进行估值，查询操作是对处理元组的个数、操作的输出和操作的总的处理时间进行统计报告的工具。在这些统计的基础上，系统对操作的选择率、操作的处理开销和流数据的速率进行估算。当流的到达速率和数

据特征发生变化时，相适应的负载要脱落，这时降载的位置也发生变化。因此，STREAM 系统是通过统计管理器周期性更新降载输入参数的估计值，降载计划也是周期性地改变。

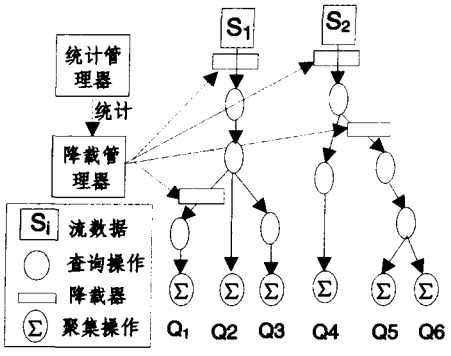

图 2　STREAM 降载数据图

表 1 对 STREAM 系统与 Aurora 系统的降载策略的相同和相异之处进行了比较。

表 1　Aurora 与 STREAM 的降载特点比较

	Aurora	STREAM
支持的操作	过滤，并，连接(long window-size only)	任何除了连接的操作
支持的查询	任何由支持的操作组成的查询（不支持聚集查询）	聚集查询（SUM，COUNT）
删除的类型	随机，语义	随机
准确性度量	每个查询的 QoS	$\epsilon_i=\lvert A_i-\hat{A}_i\rvert/\lvert A_i\rvert$
限制	Load(N′(I)<H＊C	负载方程
查询的操作共享	是	是
不同权重的查询	不	是
主要技术	降载路线图	近似查询结果的概率范围
目标	load ＜ H＊C（结果准确的损失最小）	所有查询误差最大值的最小化

3.3 TelegraphCQ

美国加州大学伯克利分校构建了一个 TelegraphCQ[8] 系统，该系统用于连续的流数据处理。TelegraphCQ 系统中使用的是一个叫数据筛余的降载框架(如图 3)，主要是在每个数据源和查询处理器的中间放置筛余队列。这种设计主要是用来处理突发式负载带来的诸多问题，因为突发数据不仅带来更多的数据，而且带来与众不同的数据。降载必须是用低的等待时间来产生准确的查询结果。由于突发期间包含较多的有趣信息，因此降载不应当简单地丢弃过载的数据，必须捕捉丢失数据的特性。图 5 在网关模块中嵌入筛余队列，把流数据转换成系统的内在格式。如果查询引擎不能处理以一定速率进入筛余队列的元组，系统就会建立过量元组的摘要。

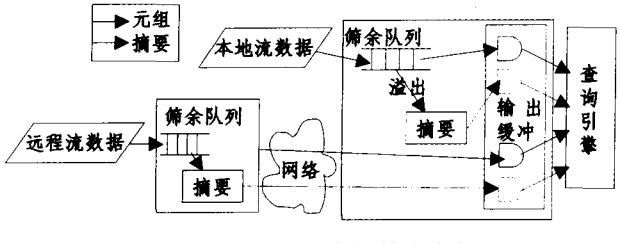

图 3　TelegraphCQ 的数据筛余降载框架

在正常操作期间，数据源向各自的队列输入元组，查询处理器按需要从队列的后面获取元组。如果流数据的输入速率超出了查询处理器处理数据的能力，筛余队列填满。当筛余队列超出空间，系统使用一个丢弃策略从队列中去掉不是很重要的数据，而且使用摘要来捕捉已删除元组集的相似特性。在连续查询的每个时间窗口，筛余子系统把这些摘要传到查询引擎，并计算查询结果的一个近似答案。最后，准确和近似的结果合起来组成一个窗口的查询结果。

3.4 Borealis

布朗大学、布兰代斯大学和麻省理工大学还在设计一种可升级的分布式 Aurora，为 Borealis。Borealis 是一个新的分布式流处理引擎[11,12]，它试图为分布式处理提供单一结构，这个结构可以跨越不同的处理元素：大至服务器，小到传感器。Borealis 的主要目标在于使系统对分布式数据流处理应用达到一种比较高的可量测性和实用性。

负载分布问题包括负载均衡与负载共享，在传统并行与分布式系统中有广泛的研究[13,14]，但在流数据处理环境中没有进行全面的研究。传统的负载分布策略是使用总的负载信息来进行决策，因为它们是为负载波动发生的基于推进系统而设计的。在流数据以一定速率到达而产生负载波动的情形下，即使某一节点的平均负载不是很高，但在峰值期间，一个节点将会遇到暂时的负载峰值，其数据处理的等待时间将受到影响。所以，为了使数据处理时间最小化，就需要使用尽可能避免暂时过载的方法。

负载的平均水平并不是负载分布中的唯一重要因素，负载的变化也是决定系统性能的关键因素。Borealis 系统把操作负载描述为固定长度的时间序列。两个时间序列的关联是通过关联系数来度量的。关联系数是[－1,1]之间的实数，它的真实涵义是当两个时间序列有正的相关系数，那么如果某个下标的一个时间序列的值相对它的平均值是相对大的，则另一个有同样下标的时间序列的值也往往是相对大的。相反，如果相关系数是负数，当一个时间序列的值是相对大的，那么另一个的值往往是相对小的。算法通过观察两个操作的负载时间序列的相关系数是否为小来激励。当对操作的分配进行决策时，尽量对不同节点间的静态负载相关系数最大化。只要两个节点的负载时间序列有大的相关系数，即使负载发生变化，其负载水平自然达到平衡。平均负载变化的最小化对平均负载的相关的最大化也有帮助。

4　流数据降载技术的挑战与研究展望

对于 DSMS 来说，适应环境的变化是非常重要的。由于流数据的到达速率及峰值是不可预测的，其数据的特征不断地发生变化。当流数据的到达速率超出了系统的处理能力，必须对流数据的负载进行控制，以保证系统正常运行。流数据的降载研究是解决过载环境下流数据近似查询，其目标应是使系统的输出速度最大化，且使对查询结果的准确性影响达到最小化。虽然在降载的时间、位置和量已有相关的研究，但在一些关键技术方面仍需深入研究。

传统并行与分布式系统的负载分布问题有广泛的研究，但流数据处理环境中没有进行全面的研究。Aurora、STREAM、TelegraphCQ 系统中降载策略均是单服务器上应用的方法，并没有考虑节点间负载的动态分布与均衡。Borealis 系统虽然对分布式流的降载做了一些初步的研究，但仍有许多问题期待解决，所以分布式流的降载问题有待研究。

(下转第 165 页)

ABB是特征选择算法中较为典型的完全搜索算法，耗时大；本文提出的MRMI-UC算法是启发式的特征选择算法，耗时小；从分类学习的Error rate看出，由MRMI-UC得到的结果并不比由ABB的差，结果相当。

从表中Befor和After项结果看出，经由本算法MRMI-UC特征选择后Decision Table和Naïve Bayes学习的错误率不比用初始特征集学习大，反而有些还低得多，特别是在Naive Bayes十折交叉验证中，对Parity5＋5分类的错误率由61.14%降到41.00%，降低了20%。这表明，用MRMI-UC进行特征选择数据处理，提高了Naive Bayes的学习正确率。

从两表After项中MRMI-UC和ABB算法特征选择后分类器学习错误率来看，由MRMI-UC得到的结果并不比由ABB得到的结果差，如Parity5＋5中由MRMI-UC得到的错误率41.00%比ABB得到的59.57%还低18个百分点。

实验结果表明，RMI来衡量特征间的相关程度是可行的，本文提出的基于相对互信息比和不确定性系数的特征选择算法是有关效的。

总结 本文基于粗糙集中关于非精确集和精确集理论思想，从信息论的角度来研究特征选择问题。通过研究非核属性与核，以及非核属性与类别属性间相关性关系的一些变化规律，提出了度量特征的一个新指标，即相对互信息比RMI，并由此，设计了一种基于粗糙集启发式特征选择算法。实验结果分析表明，本文提出的算法在多数情况下能够得到较优的特征子集，算法是有效的，切实可行的。但是，关于该算法对最小属性约简的完备性问题还需从理论上作进一步的探讨。

参考文献

1 Liu Huan, Motoda H. Feature Selection for Knowledge Discovery and Data Mining [M]. Kluwer Academic Publishers, 1998

2 Pawlak Z. Rough Sets-Theoretical Aspects of Reasoning about Data. Kluwer Academic Pub, 1991

3 张腾飞，肖健梅，王锡淮. 粗糙集理论中属性相对约简算法. 电子学报，2005，33(11)：2080～2083

4 Perkins C E. Ad Hoc Networking [M]. Chapter 3, ADDISON-WESLEY Press, 2000

5 王国胤. Rough集理论与知识获取[M]. 西安交通大学出版社，2001. 51

6 常犁云，王国胤，吴渝. 一种基于Rough Set理论的属性约简及规则提取方法. 软件学报，1999，10(11)：1206～1211

7 杨胜，胡福乔，施鹏飞. 一个新的特征选择判据——不确定性系数. 计算机工程，2004，30(8)：46～47

8 梁霖，徐光华. 基于克隆选择的粗糙集属性约简方法. 西安交通大学学报，2005，39(11)：1231～1235

9 Liu H, Motoda H, Dash M. A Monotonic Measure for Optimal Feature Selection. In: Proc. of ECML-98, 1998

10 于冰，阎保平. 关于粗糙集属性约简的进化算法研究和应用. 微电子学与计算机，2005，22(3)：189～194

11 Yang Sheng, GU Jun. Feature selection based on mutual information and redundancy-synergy coefficient. Journal of Zhejiang University SCIENCE, 2004, 5(11): 1382～1391

(上接第115页)

对流数据降载需要深入研究的工作有：

• 动态负载分布中负载均衡。现有的动态负载分布中负载均衡研究明显存在条件的假设，所以条件的放宽和放在更异构的环境是下一步的研究内容。因为在这种环境下，就会带来诸如带宽、能源消耗等资源的新问题，而已有的算法是否在新的环境下获得最优的结果值得深入研究。

• 自适应策略在DSMS中的应用。对于在DSMS中引入控制理论，在DSMS系统中加入质量控制器，只是涉及一些较简单的问题。对于更复杂流数据系统与环境进行建模和控制仍需进行研究，特别是分布式流数据管理的动态资源配置，是目前研究的热点。

• 全局控制。要保持低的等待时间，算法也必须确保查询结果的质量不能任意地降级。一个降载计划能在某个节点发送高质量的结果，但在全局水平上并不一定有相同的质量，所以，对单点的周期性控制要与全局的控制统一起来。

• 拓扑结构的一般化。Borealis系统中元数据的合并与传播工作是在基于树结构的拓扑服务器环境下进行的。在这种情形下，来自子节点的元数据在其父节点以增量模式合并、修改和向前传播。因此，对每个节点来说足可以和它直接邻近的上下节点通讯。对于更一般的拓扑结构环境下，一个节点可能有多个父节点，要完成全局的协作，不直接相邻的节点间也需通讯。在这种情况下，这些节点中一个节点的降载决策将会影响其它节点的决策。

参考文献

1 Motwani R, Widom J, Arasu A, et al. Query Processing, Resource Management, and Approximation in a Data Stream Management System. In: Proc. of CIDR 2003, Jan. 2003

2 Carney D, Cetintemel U, Cherniack M, et al. Monitoring streams: A new class of data management applications. In: VLDB, 2002

3 Krishnamurthy S, Chandrasekaran S, et al. TelegraphCQ: An Architectural Status Report. IEEE Data Engineering Bulletin, 2003, 26(1): 11～18

4 Tatbul N, Cetintemel U, Zdonik S, et al. Load shedding in a data stream manager. In: Proc. of the 29th Intl Conf. on Very Large Databases (VLDB'03), 2003

5 Babcock B, Datar M, Motwani R. Load shedding for aggregation queries over data streams. In: 20th International Conference on Data Engineering, 2004

6 Tu Yi-Cheng, Hefeeda M, Xia Yuni, et al. Control-based Quality Adaptation in Data Stream Management Systems. In: the Proc. of the International Conference and Workshop on Database and Expert Systems Applications (DEXA), August 2005

7 Chi Yun, Yu P S, Wang Haixun, et al. Loadstar: A load shedding scheme for classifying data streams. In: SIAM International Conference on Data Mining (SDM), 2005

8 Reiss F, Hellerstein J. Data Triage: An Adaptive Architecture for Load Shedding in TelegraphCQ. In: IEEE ICDE Conference, Tokyo, Japan, April 2005

9 Tatbul N, Zdonik S. Dealing with Overload in Distributed Stream Processing Systems. In: IEEE International Workshop on Networking Meets Databases (NetDB'06), Atlanta, GA, April 2006

10 Abadi D J, Carney D, Cetintemel U, et al. Aurora: a new model and architecture for data stream management. The VLDB Journal, 2003, 2(2): 120～139

11 Abadi D, Ahmad Y, Balazinska M, et al. The Design of the Borealis Stream Processing Engine. In: CIDR Conference, Asilomar, CA, January 2005

12 Xing Ying, Zdonik S, Hwang Jeong-Hyon. Dynamic Load Distribution in the Borealis Stream Processor. In: 21st International Conference on Data Engineering (ICDE'05), Tokyo, Japan, April 2005

13 Gupta D, Bepari P. Load sharing in distributed systems. In: Proc. of the National Workshop on Distributed Computing, January 1999

14 Shirazi B A, Hurson A R, Kavi K M. Scheduling and load balancing in parallel and distributed systems. IEEE Computer Science Press, 1995

15 Tu Yi-Cheng, Liu Song, Prabhakar S, et al. Load Shedding in Stream Databases: A Control-Based Approach. In: Proc. of the 32th Intl Conf on Very Large Databases, 2006