第 34 卷 第 2 期 计 算 机 工 程 2008 年 1 月
Vol.34 No.2 Computer Engineering January 2008
·软件技术与数据库· 文章编号：1000—3428(2008)02—0054—03 文献标识码：A 中图分类号：TP311

分布式数据流查询处理的 P2P 中间件研究

杨 颖[1,2]，陈秋莲[1]，杨 磊[3]
(1. 广西大学计算机与信息工程学院，南宁 530004；2. 东华大学计算机科学与技术学院，上海 200051；3. 广西计算中心，南宁 530022)

摘 要：随着 Web 的大规模应用，分布式数据流的数量迅速增长，其查询处理面临极大的挑战。该文开发了分布式数据流响应查询的 P2P 中间件原型，利用基于内容路由所提供的可扩展性、通信负载平衡及动态适应性等特性，能有效地处理相似查询，支持内积查询。模拟实验表明该索引机制能减少网络连接的计算资源，提高数据流查询处理效率。

关键词：数据流；概要结构；中间件

P2P-based Middleware of Distributed Data Stream for Query Processing

YANG Ying[1,2], CHEN Qiu-lian[1], YANG Lei[3]
(1. College of Computer & Information Engineering, Guangxi University, Nanning 530004; 2. College of Computer Science and Technology, Donghua University, Shanghai 200051; 3. Guangxi Computing Center, Nanning 530022)

【Abstract】 With the large-scale application of Web, the number of distributed data stream is increasing rapidly and the query processing is facing the great challenge. This paper proposes the P2P-based middleware prototype of distributed data stream for real-time answer. By utilizing the content-based routing performance such as scalability, load balance of communication and dynamic adaptability, the solution can handle the query of product and similarity efficiently. Simulation experiment verifies that the indexing mechanism can reduce the computing resources of network link and greatly improve the efficiency of data stream processing.

【Key words】 data stream; synopsis structure; middleware

数据流是数据库中迅速增长起来的一个重要分类，主要应用于传感器网络、证券报价机、新闻机构、长途通信及数据网络等。它们大量地从路由器涌入数据处理中心，操作中心计算这些数据的统计特性，分析检测未来的动向，发现数据潜在模式，并为大多数商业、企业投资和交易提供有价值的信息。原有解决方案利用一个数据中心来收集所有数据流的信息，应答所有的数据查询，该方法在当前及未来的数据流系统中都是不可行的，因为服务器及其附近的网络不得不每秒钟处理成万上亿的信息，该数据中心易于变成故障点或瓶颈。当前基于内容路由技术，如Naspter[1], Chord[2], Pastry[3]等作为基于网络点对点应用的解决方案，在分布式数据流系统中，具有为通信底层铺设的潜能，提供均一分配流数据处理负载的方法，也能够平衡系统中数据沿各节点和链接传输时的通信负担，促进系统中新数据流及数据中心的无缝添加，以及处理各种可能的故障，如数据中心毁坏等。本文为分布式环境的数据流处理提供一种可适应的、可扩展的中间件。

1 数据流计算模型

一个数据流由一序列的数据点$\cdots x_i \cdots$组成，每个数据点的数据范围是$[R_{\min},R_{\max}]$。本文讨论的是有M个数据流的系统，并且只对每个数据流中最近的前N个数据项感兴趣，使用K个大小为N的滑动窗口来存储数据流，最近到来的N个流数据存放在最新基本窗口，依次类推，每个新到来的数据流都不断更新滑动窗口，并且处理数据流上的两大类主要查询，即内积查询和相似查询[4]。本文采用离散傅里叶变换DFT来计算流的特性。一个信号$\vec{x}=[x_t]$(t=0,1,$\cdots$,N-1)的N点DFT[21]定义为N个复杂成员x_f(f=0,1,$\cdots$,N-1)的一个$\vec{x}$序列。

$$X_f=(\sum_{t=0}^{N-1}x_t e^{-j2\pi ft/N})/\sqrt{N},\ f=0,1,\cdots,N\text{-}1,\ j=\sqrt{-1} \quad (1)$$

而$\vec{x}$的反傅里叶变换为

$$x_t=(\sum_{t=0}^{N-1}X_f e^{j2\pi ft/N})/\sqrt{N},\ f=0,1,\cdots,N\text{-}1 \quad (2)$$

DFT是一个正交变换，因此，它保持了信号的能量，即$\sum_{t=0}^{N-1}x_t^2=\sum_{t=0}^{N-1}X_t^2$。式(1)证实，正向DFT变换需要$O(N^2)$个操作。当每一新数据到达时，从草图中计算系数，每项处理时间可以控制为最大值N。但由于DFT的可升级能力，信号$\vec{x}=[x_t]$，(t=0,1,$\cdots$,N-1)的每一个系数x_f'(f=0,1,$\cdots$,N-1)都可以由前面已计算得到的x_f和流值x_0和x_N在常量时间内计算：

$$X_f'=e^{j2\pi f/N}(x_f+(x_N-x_0)/\sqrt{N}) \qquad f=0,1,\cdots,N\text{-}1 \quad (3)$$

在大多数的实时级数中，前面 k(k<<N)个 DFT 系数保留这个信息的大部分能量，因此可以很安全地忽略除了前面几个 DFT 系数之外的所有系数，有效地减少工作空间的维度。这一方法的时间复杂度是 $O(N)$，空间复杂度是 $O(k)$，以保存原始时间级数的显著特征。

2 系统结构和解决方案

基于上述流模型，本文提出了分布式的数据流查询处理的P2P中间件，它利用了P2P基于内容路由机制所提供的可扩

基金项目：国家“863”计划基金资助项目(2002AA4Z3430)；广西大学科研基金资助项目(X061001, X061002)
作者简介：杨 颖(1969－)，女，博士研究生，主研方向：数据库，数据仓库，数据流技术；陈秋莲，讲师；杨 磊，研究员
收稿日期：2007-01-30 **E-mail：**yingy2004@126.com

展性、通信负载平衡及动态变化适应性等特性[4]，可以在减小网络计算资源情况下提供各种类型查询的实时响应。

2.1 系统结构

本系统结构的主要设计目标是：在网络数量最小及数据中心和网络链接所消耗的计算资源最小的前提下，提供快速有效的数据流查询和挖掘的实时响应，该中间件能均衡流数据查询处理的负载分配，以及系统中各节点和链接的的通信负担，并能够适应动态变化，如数据中心和链接可能出现的故障，新的数据中心和数据流加入而引起系统操作的临时堵塞等。

图 1 描述了所提出的分布式数据流查询处理的 P2P 中间件结构，它包括应用层、中间层和系统层 3 个组件部分，其中：(1)应用层包括查询订阅和数据源以及它们的接口，查询订阅接口的用户交互模式使客户为信息处理指定订阅规则，指定具体应用的配置信息；数据源接口可实现流数据在 P2P 系统中的某一结点的注册和在线更新。(2)中间层包括核心组件和元数据管理组件，其中核心组件由优化器、数据流处理器和查询处理器组成。优化器负责接收应用层的用户请求，对以往的查询进行聚类，根据查询类选择最有可能包含查询结果的结点发送查询，并利用精确梯度的设置与优化方法来获得根结点的负载最小化。数据流处理器运用流概要结构的算法和优化器的配置信息，来获取数据流的概要结构，并不断更新中心节点的索引结构。查询处理器计算查询模式，并将其路由到相关节点，以获得最优的相似和内积查询结果。此外还周期地更新本地订阅表，并针对各结点间网络带宽异构的特点，采用关系缩减算法和行分块传输技术来减少查询响应的时间。元数据管理组件利用 Web 服务技术，完成 P2P 环境下的信息资源发现、注册、调度等工作，并对各种临时性主体或永久性主体的各类信息进行语义描述和存储，以实现异构数据的相互转换。(3)系统层运用 Chord 协议和改进算法来构建一个 P2P 环境平台，针对大多数 P2P 系统只支持文件标识搜索，采用广播式搜索盲目低效，浪费带宽的问题，提出基于内容路由的分布式索引结构，充分利用系统负载平衡和动态适应性的特点来减少维护的代价，提高搜索效率。

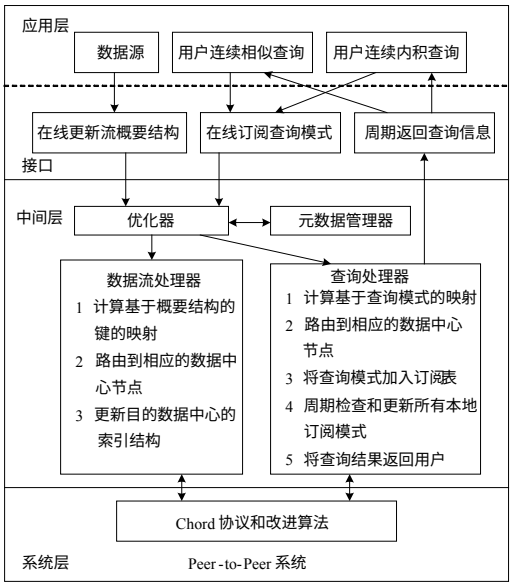

图 1 分布式数据流查询处理的中间件

2.2 流概要到弦环节点的映射

本节的核心在于如何将包含有数据流概要结构的消息映射到弦环节点上。对一个规范化的数据流，概要是一个k维单元特征空间矢量$\boldsymbol{X}\in R^K$，每一个新计算得到的特征矢量$\boldsymbol{X}$都被路由到一个数据中心节点，作为 2^m位标识符的Chord环中由哈希决定的有效标识符i的后继节点。利用一个映射函数h：$R^K\rightarrow\{0,1,\cdots,2^m\text{-}1\}$，它以特征矢量$\boldsymbol{X}\in R^K$作为参数，返回一个有效的Chord环标识符 $i\in\{0,1,\cdots,2^m\text{-}1\}$。若事先知道$X_0$的概率分布函数$f(X_0)$，那么可以计算出间隔$[a_i,\ b_i]$，从而以$\int_{a_i}^{b_i} f(X_0)\mathrm{d}X_0$ 的方式为系统中总共N个节点每一个赋值，这样就可以达到一致的负载平衡。由于所有的流都可以投影到一个单元特征空间中，因此有： $\sum_{i=0}^{K-1} X_i^{\,2}\le\sum_{i=0}^{N-1} X_i^{\,2}=1$，这意味着，对$i$=0,1,⋯,$K$-1，有-1≤$X_i$≤1。如果每一个流$x$都是$Z$轴规范化的，均值$\mu_x$=0 且标准方差$\sigma_x$=1，为了计算标识符，将间隔[-1, 1] 的值按[0, 2^{m-1}] 的比例进行转换，如下：i=$\lfloor(X_0+1)\times 2^{m-1}\rfloor \bmod 2^m$。由$X_0$=-1, X_0'=0, X_0"=1 分别映射为i=0, i'=2^{m-1} 及i"=0。例如，R^2的特征矢量$\boldsymbol{X}$=[0.40, 0.09] 映射到Chord环上，容易得到$\lfloor(0.4+1)\times 2^5\rfloor \bmod 2^6$=45，即Chord环上的键$K$45。

图 2 描述了系统操作的简单过程：由传感器节点 N8 产生的数据流所计算得到的特征矢量 $\boldsymbol{X}$=[0.40,0.09] 映射到键K45，节点 N8 检查其路由表，找到节点 N28 是离 K45 最近的前驱节点，它将 $\boldsymbol{X}$ 路由到节点 N28。节点 N28 检查其路由表，又找到了它的最直接的后继者节点 N48，则节点 N48 就是 K45 的后继节点。这一操作最终将特征矢量存储在节点 N48 中。

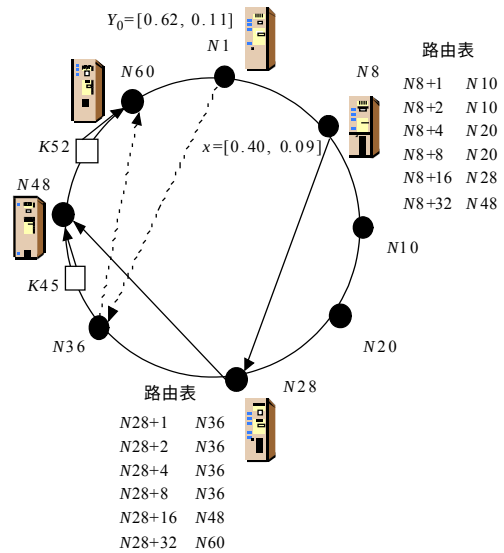

图 2 流概要结构的映

同理，由节点 1 计算得到的特征矢量$\boldsymbol{Y}$的第 0 个系数Y_0=[0.62, 0.11]映射到Chord环上，$\lfloor(0.62+1)*2^5\rfloor \bmod 2^6$=52，由路由表节点$N$1 先将$Y_0$路由到节点$N$36，继而到$N$60，如图 2 虚线部分，因此键$K$52 最终存储于节点$N$60 上。该方法不仅可以发送信息到单个键，还可以发送到一定范围内的键。即给定键所覆盖的所有节点都可以收到这一消息。该方法在有大量节点的系统中起着重要的作用，有效地支持多点传送，提高了系统的健壮性和适应性。

2.3 查询处理

对于内积查询，利用底层的基于内容的路由机制来实现定位服务，利用一个映射函数h_2 将流标识符空间映射到键空

间。当含有标识符S_{id}流登录到系统时，它的流源n将<S_{id}, n>对放入$h_2(S_{id})$所确定的节点中。当节点N_2产生一个内积查询(S_{id}, I, W, T)时，节点N_2首先通过传送一个消息到$h_2(S_{id})$获得流源n的键，然后将查询传递给n。节点n收到查询后，它执行一个X^{Sid}的反变换，重构一个近似信号$\vec{X}$，如下所示：$x_t \approx (\sum_{f=0}^{k-1} X_f^{S_{id}} e^{j2\pi ft/N}) / \sqrt{N}$，t=0,…,$N$-1。由近似流值$x_t$，可计算内容的权值：$\sum_{i=0}^{l} x \cdot I_i \cdot W_i$，$\ell$表示矢量$\boldsymbol{I}$的长度。

给定节点N的相似查询(Q, r, T)，首先特征矢量$\boldsymbol{X}^Q$被抽取，节点的任意特征矢量$\boldsymbol{X}$在不等式$X_0^Q-r \le X_0 \le X_0^Q+r$成立时为$Q$的近似查询的备选答案。因此，查询被传送给[$h$ (X_0^Q-r), h (X_0^Q+r)]范围内的键。图 3 显示了节点$N1$上产生的查询Q，它以特征矢量$\boldsymbol{X}^Q$=[-0.08, 0.12]，以及阈值r=0.29 (查询半径)路由到相关节点。在这种情况下，上界X_0^Q+r的数值为-0.08+0.29=0.21，得到键$K39$，而下界X_0^Q-r的数值为-0.08-0.29=-0.37，指向键$K20$，因此该查询被复制到节点$N20$, $N28$和$N36$。由于利用单一的特征值X_0来复制概要和查询，因此在真实节点集的超集上，运用维度减少的下界属性而得到：$\sqrt{(X_0-Y_0)} \le L_2(\boldsymbol{X},\boldsymbol{Y})$。为响应内积查询，收到查询的节点要传送应答给请求节点，相似查询引起处于范围内的节点定期地传送检测相似性的消息到中间节点，而中间节点则有规律地传送响应信息给客户端。如图 3 中的虚线部分，节点$N20$，$N28$和$N36$定期地将查找到的本地备选路由到节点 48，从而在查询生命周期内，聚集了响应结果，并把它们传送到最初的查询节点$N1$。

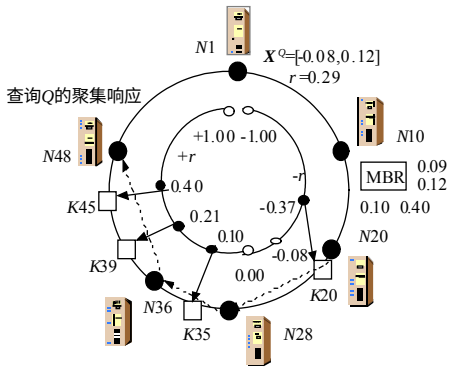

图 3　相似查询的处理

由于所采用的特征提取方法，在同一数据流上计算所得的连续特征矢量呈现很强的方位性。例如，如果在时间t上由流片段x_t, x_{t+1},…, x_{t+N-1}计算得到一个特征矢量，那么下一个特征矢量就由流片段x_{t+1}, x_{t+1},…,x_{t+N}计算得到，与前面流片段有N-1 个口重叠。因此，特征矢量在连续时间单元内的计算具有很强的时间相关性。利用这一相关性来减少通信开销，通过传送批更新到远程数据中心节点，即将每C个特征矢量分成一组，称为最小边界矩形块（MBR B），然后将这一MBR路由出去而不是复制单个特征矢量。图 3 的节点$N10$给出了坐标为X_0^{lo}=[0.10, 0.09]，X_0^{hi}=[0.40, 0.12]的MBR B实例，下界X_0^{lo}指向$K35$，而上界X_0^{hi}指向$K45$。因此，B被复制到节点$N36$和$N48$，它们是键在[35, 45]范围的唯一后继节点。

3　实验

利用 Java 编程来开发分布式数据流查询处理的系统原型。为了获得实验平台，运用了公开的Chord模拟器[2]，通过模拟实验来测试系统中节点事件的执行时间，并利用网　站[5]所提供的历史交易股票数据S&P500(Standard and Poor 500)，该数据包括 500 种不同的股票数据，单个股票文件的每行文字对应于当天交易数据中的一个记录，这一记录存放着表示日期、股票行情自动收录器的开盘、高值、低值、收盘及当日交易量的字段数值。

实验主要测量系统效率，即对系统内每一个输入事件的响应要传递的信息数量，以及系统的响应度，即每个请求经过的跳变次数。系统效率表示了系统为了处理某个类型的输入事件（如一个新 MBR，新的查询或响应）而发送的消息数。图 4 显示了消息的开销对比，用 a 表示在跨越多个节点的MBR 消息数；b 表示传给直接节点的 MBR 数；c 表示跨越多个节点的查询半径中的查询消息数；d 表示传输中的查询消息数；e 表示检测相似性的消息数；f 表示传输中的响应数。由图可看到，响应查询的检测相似性的消息数和传输中的响应消息数占网络总消息的比例最大，而且随节点的增加，该比例加大，说明系统有效地处理了各种类型地消息，并且，随节点数增大，节点更密集地分布在键范围内，一个查询包含更多节点来执行该查询。图 4 的结果说明这种依赖性和节点数是线性关系。

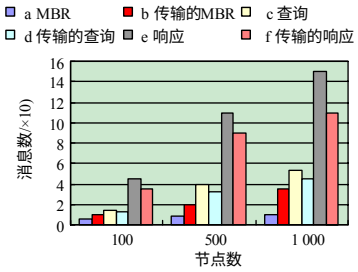

图 4　消息开销

4　结束语

本文设计并开发了分布式数据流响应查询的 P2P 中间件原型。它利用了基于内容路由机制所提供的可扩展性、通信负载平衡及动态适应性等特性，支持大量动态信息流的处理，并适应于不同的精度需求，能减少数据中心节点间网络连接的计算资源，可以为分布式数据流应用提供运行与开发的环境。本文的研究假设分布是一致性的，而通过调整映射函数以适应于不同的分布性是未来将要研究的工作。

参考文献

[1] Napster[Z]. (2006-08-27). http://www.napster.com.

[2] Stoica I, Morris R, Karger D, et al. Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications[C]//Proc. of SIGCOMM. USA: [s. n.], 2001-08.

[3] Rowstron A, Druschel P, Scalable P. Distributed Object Location and Routing for Large-scale Peer-to-peer Systems[C]//Proc. of IFIP/ACM Int'l Conference on Distributed Systems Platforms. Germany: [s. n.], 2001-11.

[4] Ratnasamys, Francis P, Handley M, et al. A Scalable Content-addressable Network[C]//Proc. of ACM SIGCOMM. San Diego, CA: [s. n.], 2001-08.

[5] S&P500 Historical Stock Exchange Data[EB/OL]. (2006-08-27). http://kumo.swcp.com/stocks/.