第 5 章

物联网中流式数据处理

Carolina Fortuna and Timotej Gale

Jožef Stefan Institute, Ljubljana, Slovenia

5.1 引言

我们生活在这样一个时代：数据的生成速度超过了人类的数据消耗速度，而信息的传播速度却比以往任何时候都要快，并且每个能够访问联网设备的用户都可以创建和消费信息内容：新闻、图像、Tweet、视频等。此外，正如第 1 章所讨论的那样，用于生成数据的联网传感器数量将持续快速增加。在过去的三十年中，开发了各种旨在帮助人类总结、组织和检索不断增长的数据的系统。也许使用最广泛的工具是可以在互联网这样庞大的网络中检索内容的网络搜索引擎。大多数此类大型数据组织系统可以分批提取原始数据，并从批处理中提取数据模型。批次的大小和涵盖的时间段会有所不同。例如，一个新批次可以按照小时、天、甚至月来执行。

在某些应用领域，快速传输信息非常重要，并且集中地批量处理大数据无法满足相关要求。其中，实时或近实时地传输数据，并在数据到达时进行处理的系统称为流式数据处理系统。以金融交易系统为例，对新闻、股票价格和其他数据信息的访问必须非常迅速。类似地，物联网（IoT）的一个代表性示例来自交通领域，人们需要立即知道当前哪些道路发生了交通拥堵，而一个小时的信息延迟是没有实际意义的。同时，现代导航系统使用实时传感器数据来告知驾驶员，并给出拥堵较少的替代路线建议。这样的流式数据处理平台大约在二十年前开始出现，并且在现代知识驱动型经济中变得越来越重要，目前数据驱动型决策的执行速度比以前快得多。在物联网环境中，我们通常处理时间序列数据，即按时间顺序生成和交付的一系列数据点，通常是在连续等间隔时间点上采样的序列。

能够处理数据流的系统已经存在了数十年。实际上，任何电信系统中的信号

处理和传输都需要处理数据流。但是，传统流式数据处理系统和现代流式数据处理系统之间的两个主要区别是：①现代方法的实现方式是软件而不是硬件；②处理的数据量要大几个数量级。到2020年为止，业界最大规模部署的流式数据处理系统是Heron，该系统为Twitter提供强大数据处理能力（每分钟可以处理大约35万条推文）。然而，流式数据处理系统正越来越多地为大量运营物理基础设施的企业提供支撑，例如，汽车、工业和能源行业的云服务提供商、电信运营商和企业（另请参见第13章）。

5.2 基础知识

时间序列数据流的处理可以采用多种形式，具体取决于所服务的应用程序。早期的物联网流式数据处理平台能够实时收集传感器测量值，并将其显示在表格或图表中。但是，实践经验表明，只显示原始数值的意义不大。正如本节所讨论的，数据通常必须经过几个处理阶段才能产生可操作的自动或人工决策。对于流式数据，这些处理阶段也称为流式数据分析。流式数据主要涉及5个操作：压缩、降维、摘要（映射）、学习与挖掘、可视化。尽管可以对静态的非流式时间序列数据执行相同的操作，但在流式数据传输中，这些操作的实现方式有很大的不同，因为每个数据点到达时都必须以很高的速度进行相关处理。

5.2.1 压缩

时间序列的压缩是指为减少存储所占用空间和/或传输所需要带宽而进行的一组数据操作。对于资源受限的设备（例如，某些类型的电池、带宽和存储容量有限的传感器）而言，压缩操作尤为重要。尽管存储设备价格一直在下降，但压缩仍然很有用，尤其是对于大型数据的存档需求。压缩可以通过多种方式实现，从非常简单的方法开始，例如，在欠采样中丢弃数据点，在聚合时计算并保留时间窗摘要（例如，平均值、标准偏差、最小值和最大值）。这种方法的主要缺点是容易丢失信息，特别是在发生罕见事件的情况下。此外，业界还提出了更复杂的压缩方法，这些方法可将信息的损失降到最低，或利用分布式传感器测量值来重建数据的分布。在文献[10]中也提出了针对流式数据而设计的压缩算法，该压缩算法已在流式数据库中采用。

5.2.2 降维

通过减少时间序列的一组随机变量（或维数）可以实现降维。在将时间序列数据传递给执行摘要（映射）、学习与挖掘和可视化的算法之前，通常会进行降维处理，以确保仅向用户或算法提供信息最多的随机变量。统计和信息论、图论、线性代数等理论都涉及降维技术。例如，主成分分析（Principal Component Analysis，

PCA)、因子分析和线性回归等方法可以在低维空间中保留高维数据的线性结构，具有直观、易于解释的计算能力优势。此外，还存在能够保留数据非线性结构的复杂方法，它与 PCA 等方法不同，例如，用于处理流式数据的时间序列数据库通常采用改进的流式版本降维算法。

5.2.3　摘要(映射)

时间序列数据摘要(映射)是向人类和机器数据使用者描述时间序列数据转换过程的一种表示形式。摘要(映射)技术分为两类：挖掘和查询时间序列的数据摘要，以及时间序列数据中观察模式的自然语言摘要。可以使用多种技术来实现数据挖掘和查询的摘要，同时，摘要本身可以表示为时间序列、图形、符号等。例如，文献[15]提出了一种与时间相关的数据摘要方法，其中，较新数据更精确，而较旧数据则不精确。时间序列的自然语言摘要通常会将时间序列中观察到的趋势或模式映射到自然语言中的单词，例如，“略有增加”“增加”“重复”等。

5.2.4　学习与挖掘

基于时间序列数据的学习是指从记录的数据中自动构建机器学习模型的方法。然后可以将这些模型应用于新数据，从而进行推断挖掘。挖掘时间序列数据通常涉及使用机器学习模型来提取潜在见解，并了解大型数据集或者得出无法直接分析的结论。通常可以使用监督学习技术(例如分类)或无监督技术(例如聚类)来构建机器学习模型。对于监督学习技术，模型的构建需要带标签的数据，这些技术对于估计将来状态、测量值或识别某些行为最有用。当标签不可用时，通常使用无监督学习技术实现此类数据学习模式。时间序列数据挖掘也可以看作是机器学习模型在实际问题中的应用。机器学习和数据挖掘算法可用于内容检索、聚类、分类、分割、预测、异常检测和主题发现等多种任务。

5.2.5　可视化

时间序列数据的可视化是指将数据转换为针对人类消费者的视觉表示方法。时间序列数据的可视化必须以直观且易于解释的方式表现有关时间序列以及数据本身的信息。数据可以用多种方式表示，例如，符号、颜色、自然语言、几何形状等。在某些情况下，可视化工具应帮助用户能以更详细的方式探索或放大部分数据，通常特定于具体任务/应用程序，其目的是支持人类的理解并呈现相关内容。例如，基因学研究人员必须了解特定基因的时间行为以规划其研究进度，而核发电厂管理者必须了解核冷却系统的状况以安排维护工作等。

流式数据处理的发展进步可以体现在算法、平台以及应用领域。正如本节所述，业界正在开发用于压缩、降维、摘要(映射)、学习与挖掘和可视化流式数据的新算法。

关于支撑技术和基础架构平台，典型案例包括 S4、Storm、Millwheel、Samza、Spark、Flink、Heron、Summingbird 等。文献[5]对上述平台进行了详尽的分析。各种商业云平台还提供了一些流式数据分析服务，包括针对物联网的流式数据分析服务、Microsoft Azure 流式数据分析、Amazon AWS Kinesis 等。

关于具体应用，流式数据分析可用于欺诈检测、网络、流量分析、灾难管理、运行状况检测、智能电网等领域。

5.3 构架与语言

流式数据处理系统主要包括两类。第一种是基于现有易于理解的关系数据库原理，即第一代流式系统或数据流管理系统(Data Stream Management System，DSMS)，例如，STREAM 和 Aurora。新一代流式数据系统引入了对无限连续数据流的连续查询，这与传统数据库管理系统(Database Management System，DBMS)对有界存储数据集的查询无关。虽然适应关系查询语言(例如 SQL)可能足以满足简单的连续查询，但查询的复杂性不断增加，例如，添加聚集、子查询和窗口结构使所应用的语义较为晦涩。因此，创建了各种连续查询语言，如 CQL、ESL、Hancock 等。同时也存在用于传感器网络应用的专用流式系统和查询语言。

第二种类型的系统更适合于流式数据处理，因为它们不执行关系视图，并且可以创建、使用并转换数据流，进而生成新数据流的自定义运算符。这类系统通常称为第二代流式数据处理系统。Flink 和 Samza 是第二代流式数据处理系统的代表。大多数流式处理系统都是分布式的，并且支持自动缩放，例如，文献[4]中介绍的相关系统。流式数据处理系统倾向在数据流上使用各种运算符，并且可以按照不同的方式实现和优化每个运算符。在文献[34]中详细讨论了相关操作符目录及其优化方法，而文献[35]则广泛涵盖了流式数据管理的各个方面。

如上所述，以时间特性和联合处理数据量为特征的数据处理系统主要涉及两类数据处理范例——批处理和流式处理。批处理将新数据累积到分离的组别(即批)中，并在随后的时间对其进行处理，这由某些标准(例如，批大小或新旧)定义。当批量很小或需要经常处理时，这种范式也称为微批量处理。流式处理可以立即单独或在滚动窗口内处理每个新数据。因此，流式处理的延迟要比批处理低得多，并且流式数据处理由每个新到达的数据点触发，而不是在固定的时间间隔或满足某些条件时触发。由于更宽松的时间限制，批处理使计算更加复杂，而流式处理通常与实时操作相关。本章引言已经介绍了一些用例，例如，基于 Hadoop 的批处理、基于 Spark Streaming 的微批处理和基于 Flink 的流式处理。

在应用数据处理范式时，必须考虑如何设计数据处理系统和相关应用程序的体系架构。根据文献[5]，许多应用程序同时需要批处理和流式数据处理功能，这推动了同时支持这两种功能的 Lambda 架构出现。如图 5-1 所示，在 Lambda 架构

中，输入数据被复用到批处理层和流传输层进行处理。批处理层计算批处理视图。然后，服务层为批处理视图进行索引，以便有效地查询。与批处理层并行的是仅用于处理最新数据的流处理层，用于补偿批处理层的高延迟更新，但它以牺牲准确性和完整性为代价。最后，通过合并批处理和流处理层结果来答复查询。例如，Summingbird、Lambdoop 和 TellApart 均属于 Lambda 架构。

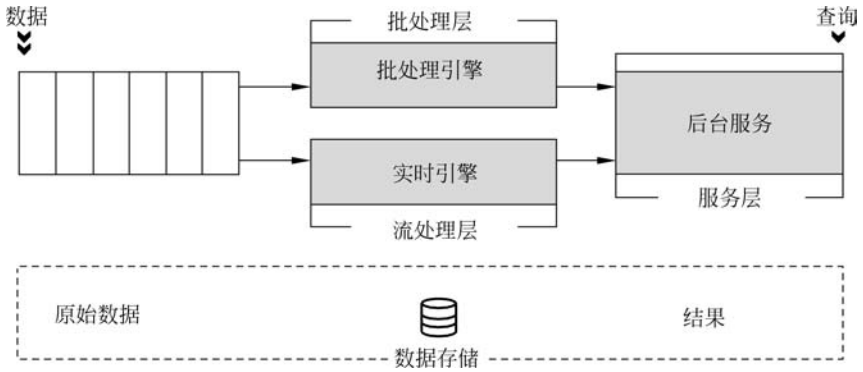

图 5-1　Lambda 体系架构

Lambda 体系架构（如文献[43]所述）可以提供准确和最新的近实时结果。然而，这种架构也引入了新的复杂性。批处理和流处理层通常需要单独的实现，而这些实现需要大量维护工作，并且合并这两层的结果也增加了复杂性。这些缺点促进了简化的替代架构，即所谓的 Kappa 架构。如图 5-2 所示，在 Kappa 体系架构（如文献[43]所述）中，不存在批处理层，并且所有数据仅作流式处理。

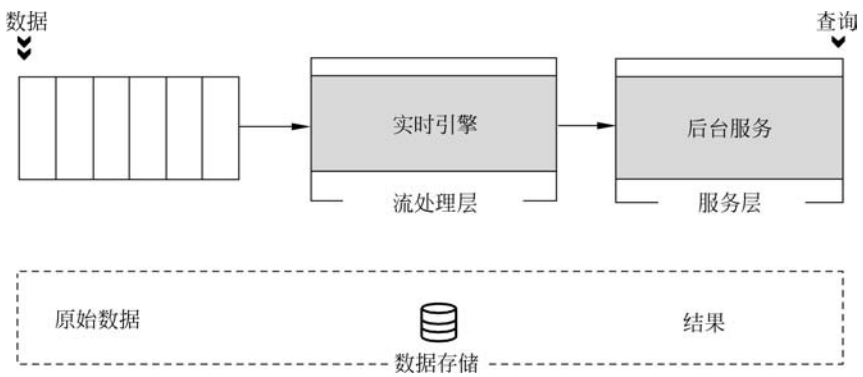

图 5-2　Kappa 体系架构

这种简化的前提是观测到的数据批（batch）为有界的数据流，然后才可以这样处理。因此，可以通过流处理层快速传送数据来模拟批处理。在处理系统发生变化的情况下，可以通过产生一个新的并发流式处理作业来重新计算结果，该作业运行于保留的数据之上，直到赶上旧的处理作业为止。然后，用新作业替换旧作业。

Kappa 体系架构可以由 Kafka(数据存储组件)和 Samza(流式处理组件)实现。此外,文献[37]进一步讨论了上述架构及其折中方案。

5.4 流式数据分析和频谱感知

如本节所述,流式数据分析在一些应用领域越来越重要。在过去的十年中,使用成本较低的频谱传感器来感知射频电磁频谱一直是无线通信的重要研究课题。然而,传感器的数量和生成的数据是有限的,适当的流式数据管理工具和算法正在扩展上述工作的基础设施。

一旦有了数据和基础设施,就有可能以前所未有的时空规模了解射频频谱中正在发生的情况,例如,"何时让设备发送信号?""在哪里放置基站?"或"哪些频段未充分利用?"等。找到上述问题的答案对于以下方面尤为重要。

- 全面、低成本的频谱活动情况报告有助于监管和政策制定,并可以更好地简化相关决策过程。
- 采用被动管理策略的网络管理系统可以更好地配置网络并确定其规模。同时,频谱代理、频谱数据库或需要更快响应时间的机器可以更好地管理无线网络和网络服务配置。
- 无线通信技术开发人员可以更好地了解当前无线网络的运行情况,并有助于后续技术的改进。
- 开发可行的 RF 3D 扫描技术。
- 可以更好地理解 RF 传播技术。

下面讲解一个基本的流式频谱数据分析系统 Spectrum Streamer,该系统能够自动生成 RF 频谱事件的实时通知、摘要和可视化呈现。Spectrum Streamer 基于流式传输架构和开源代码库①,可以实时提取时间、频率和能量数据,并按时间和频率检测传输的开始和停止事件。

5.4.1 实时通知

实时通知对于敏捷或动态频谱和网络管理场景特别有用,机器和软件实体可以交换有关频谱的当前状态信息。检测到的传输可用于更新频谱占用数据库,或直接通知其他设备某个信道中正在发生传输。因此,实时通知会生成相对大量的实时数据。

在参考实现系统中,通过 WebSocket 接口可以向基于 Web 的用户界面发送消息来实现实时通知。通知也可以通过其他可用的消息传输协议(如 MQTT 和 XMPP)来实现。为了进行概念验证,创建了一个用于事件可视化和系统演示的用

① https://github.com/qminer/qminer。

户界面。如图 5-3 所示，在可视化事件的演示系统中，用于显示事件的用户界面可以为用户提供可视化帮助，并了解后台发生的事件。用户界面是基于 Web 标准和 HTML5、Bootstrap 和 Express 等框架实现的。

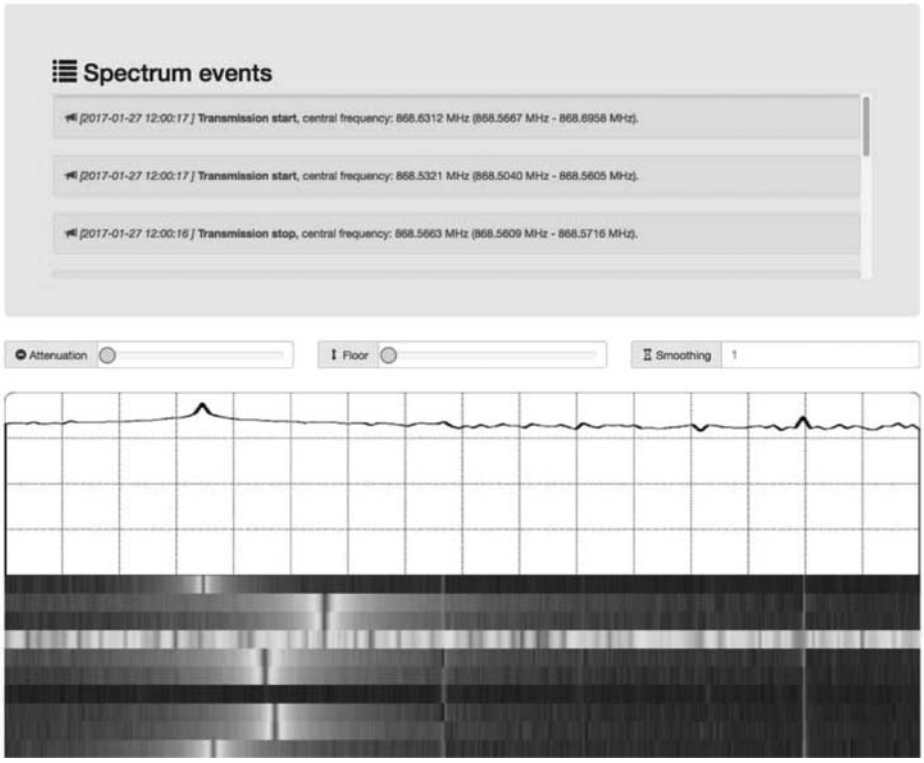

图 5-3　实时监测的可视化界面

其中，用户界面的上半部分列出了已检测到的事件，而下半部分用于描述当前样本和大量最近样本的频谱图。可以看出，近期样本涉及两种不同的传输类型。

5.4.2　统计报告

统计报告可以通过计算已检测到事件的统计信息来生成。历史报告可以被机器或网络管理系统使用，但主要针对人类用户，例如，可以作为监管机构或其他相关机构的报告生成器。

如图 5-4 所示，统计报告的参考实现使用一个额外的自定义聚合器来对各种度量指标进行计数和求平均值，然后显示在报告中。该报告的实现类似于实时通知报告。

报告的顶部显示了整个监测频谱的全局统计信息，例如，传输次数、平均传输持续时间、平均功率和总频谱占用率。然后，该报告分别继续提供每个频率单位和传输统计信息的检测频谱事件。

图 5-4 历史报告的可视化

5.5 定制化应用

除了实时通知和统计报告外，还可以利用自动生成的数据开发自定义应用，并将其订阅到频谱流数据的输出中。保留的数据存储区使用键值索引，该键值索引使自定义应用程序可以请求与特定时间和频率相关的信息，例如，特定时间段（某个小时）发生了多少次传输。当来自不同位置的频谱传感器连接到系统时，可以将位置字段添加到数据存储中，并附加地理位置键，以便进行空间查询。通过这种方式，可以构建多种类型的自定义应用程序，从简单的统计到更复杂的干扰图和隐藏

节点检测。

范围运算符[①](例如<、>和≠)对于时间和频率查询是最有用的。此类查询可以基于类似JSON的查询语言实现,也可以使用Javascript编写过滤器。虽然可以在位置字段上使用范围运算符,但位置信息查询也会受到区域半径(以米为单位)或许多事件记录(即频谱事件)的限制。

5.6　总结

随着数据生产/消费的不断增长以及信息传输的严格时间限制,流式数据处理平台变得越来越重要。早期的物联网流式数据处理平台主要用于收集和显示实时原始传感器测量值。但是,为了做出可行的决策,数据通常必须经过压缩、降维、摘要(映射)、学习与挖掘和可视化等多个处理阶段。

本章讲解了两种类型的数据流处理系统。第一种是基于关系数据库原理的数据流管理系统(DSMS),并引入了连续查询概念。第二种类型不强制执行关系视图,允许创建自定义运算符。同时,基于批处理和流式处理范式出现了两种数据处理架构——支持批处理和流式处理的Lambda架构,以及更简单的流式处理架构Kappa。

5.4节展示了用于频谱分析的基本流式数据处理系统,该系统可以自动生成RF频谱中事件的实时通知,并自动生成统计报告和结果可视化。该系统还支持定制化应用程序的开发。

参考文献

① https://github.com/qminer/qminer/wiki/Query-Language。