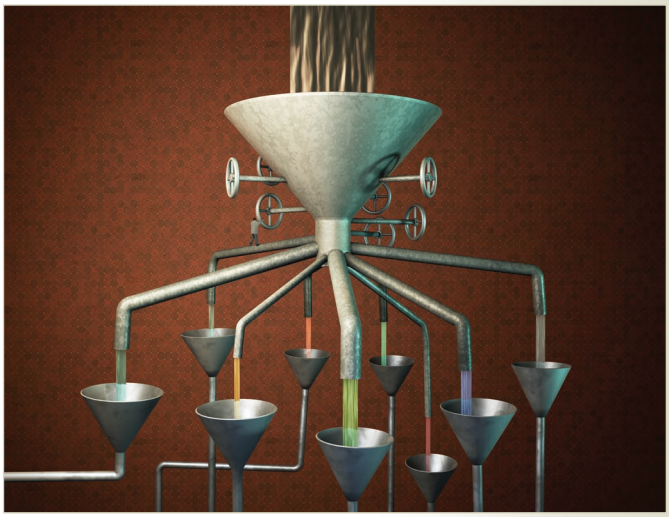

# Data Stream Management

## *Aggregation, Classification, Modeling, and Operator Placement*

**Frank Olken**
*Lawrence Berkeley National Laboratory and US National Science Foundation*

**Le Gruenwald**
*The University of Oklahoma*

The past 40 years have seen the proliferation of microprocessors in everything from watches, PDAs, cell phones, and appliances to automobiles and copy machines. In the coming decade, most (if not all) of these items will be incorporated into computer networks onto which they'll stream sensor data, such as temperature, blood pressure, and room occupancy. RFID chips and readers will provide increasingly fine-grained data on the movement of people and goods, which will be encoded as data streams. Such data will be huge and largely ephemeral, so much of it will exist only as streams rather than be permanently recorded in raw form.

Data stream management is concerned with managing these voluminous data streams arriving from data communications or sensor networks. Interest in this area is increasing, and we anticipate that it will grow rapidly throughout the coming decade, driven by rapid growth in the pervasiveness and bandwidth of digital communications networks and an impending explosion in sensor networks.

Sensor networks involve large distributed networks of intelligent sensors that typically generate streams of time-stamped measurements. Sensor network research overlaps with data stream management research, but also encompasses topics in networking protocols, geospatial data processing, and micro-operating systems. Click streams (the record of user activities on the Web) and other network logs contribute to the increasing demand for data stream management. This special issue provides a snapshot of ongoing work in this area.

### History of Data Stream Management

Typically, data streams exhibit the following characteristics:

- infinite length,
- continuous data arrival,
- high data rates,
- requirements for low-latency, real-time query processing, and
- data that are usually time-stamped and generally arrive in either temporal order or close to it.


1089-7801/08/$25.00 © 2008 IEEE

An early streaming data example was the stock ticker tape, developed in the mid-19th century and driven from telegraphy systems. Modern examples include computer network monitoring streams, event streams for distributed event-notification systems, financial and security trading transaction streams (the modern ticker tape), sensor network data streams, and Web click streams. Current research in this area is only about a decade old, with an explosion of such research occurring in recent years.

For the first several decades of database research and development, the dominant paradigm was that database management systems (DBMSs) stored databases primarily on magnetic disks. The DBMS's job was to process a stream of updates and queries against such disk-resident databases.

In the 1980s and early '90s, several database research and development efforts focused on main memory- (RAM-) based DBMSs built primarily to address very high transaction processing loads or other real-time applications. As high-speed networks (such as the Internet) began to emerge, researchers (and then developers) started to deal with distributed data management issues, addressing topics such as distributed transaction management and query optimization in distributed settings (minimizing query processing's communication costs, for example).

As networks have become faster and more widespread, the use of data stream management has risen. The growing use of sensor networks has also stimulated research in this area. In both cases, the goal is to process data as it streams from the communications network in real (or near-real) time. Such processing occurs largely in main memory, using minimal (or no) magnetic disks, which are usually seen as either too slow or too energy-intensive to be usable with high-bandwidth communication networks or bandwidth- and power-constrained sensor networks.

Other major influences on data stream management have been event-notification systems and publish–subscribe systems. Event-notification systems disseminate event notices across computer networks, combine simple event notices into complex event notices, and selectively disseminate such notices to interested parties. These systems are commonly used in process control and network monitoring systems.

Publish–subscribe systems also deal with disseminating messages across computer networks and the selective "subscription" to such messages via individual processes. Content-based routing or subscriptions are in effect continuous queries on publication streams. These systems are closely related to data stream management, but lack notions of aggregate queries, for example. Common applications include financial news dissemination.

Data stream management thus represents the confluence of ideas from many areas of database management research.

## Key Issues in Data Stream Management

Data stream management presents several key challenges.

First, it must represent infinite streams of data in finite memory, either via moving averages, exponential damped estimators, sliding windows, random sampling, or hidden Markov models (HMMs). This is essential to reconcile the infinite-length input data streams and the finite memory resources data stream management systems must cope with.

Additionally, data stream management systems are frequently subjected to severe resource constraints — in part, because they operate primarily in main memory. In other settings, as with sensor networks, such systems might have energy availability or communications bandwidth constraints. Thus, data stream management must address resource-constrained (RAM, energy, or communications bandwidth) query processing (including aggregate queries), and optimization

Because many data stream management applications are concerned with state estimation of distributed real-time systems (such as electric power grids), we must pay careful attention to temporal issues — especially time stamps, time synchronization, and minimization of temporal skewness — to assure that state estimates are consistent.

Data stream management systems usually can't afford to either store or reprocess an entire input data stream, so they must make decisions and computations as the data arrive. Novel online query processing algorithms help address this problem.

Another issue is specifying query languages and continuous queries over data streams. Traditional DBMSs were concerned primarily with querying the database state at a point in time. Data stream management systems commonly

address continuous queries that run indefinitely as the data arrive.

We must extend data stream management techniques to encompass uncertain and probabilistic data stream management, approximate aggregate query processing, and data mining problems. Traditional DBMSs support accounting applications with exact data values. Data stream management systems are more commonly concerned with uncertain or probabilistic measurement data. Approximate query processing (especially of aggregate queries) is often sufficient and expedient. Data mining from data streams is quite common, but the traditional assumption that the complete set of data is available for mining isn't realistic in data stream management systems.

Developing random sampling-based query answering approaches is important, as is developing statistical estimators of various aggregates (`SUM`, `VARIANCE`, and so on). Random sampling of data streams is one way to describe an infinite data stream in finite memory. Statistical estimators based on random sampling afford an inexpensive approach to providing approximate answers to aggregate queries over data streams.

The ability to integrate data streams from multiple sources is a critical problem and more challenging than traditional data integration because data streams usually come with high speed and changing distribution within each individual source. How to synchronize them across multiple sources to integrate data within a model such as a sliding window is a difficult issue.

In wireless sensor networks, data transmitted from sensors to servers are susceptible to losses, delays, or corruption for many reasons, such as power outages at the sensor's node or a higher bit-error rate with wireless radio transmissions compared to the wired communication alternative. Simply ignoring the missing or corrupted data when processing streams isn't an acceptable solution. Similarly, requiring sensors to resend the missing or corrupted data isn't feasible because sensors would need to remain on a continuous listening mode, which would consume more energy than necessary. A viable approach would find a way to estimate the values of the missing or corrupted data streams that guarantees good quality of service in terms of both errors and time in data estimation.

Recently, we've seen strong research interests in privacy-preserving data management — how to manage data without violating privacy policies that deal with data disclosure. For data streams, however, privacy guarantees based on data that use data processing models such as a sliding window might not hold for the overall data, so new solutions are necessary.

Finally, in mobile environments, clients, servers, or both might move over time. We must consider problems that arise due to mobility, frequent disconnections, and nodes' energy limitations when managing streams in a mobile setting.

## In this Issue

The five articles in this issue address some of the challenges we've just discussed.

"Time-Stamp Management and Query Execution in Data Stream Management Systems," by Yijian Bai, Hetal Thakkar, Haixun Wang, and Carlo Zaniolo, discusses query processing for data stream management. In particular, the authors look at issues of time-stamp management, response-time optimization, and the computation of aggregates over sliding windows on a data stream.

The article by Jin Li, Kristin Tufte, David Maier, and Vassilis Papadimos, entitled "AdaptWID: An Adaptive, Memory-Efficient Window Aggregation Implementation," deals with the efficient computation of aggregate queries over groups in data streams, while minimizing memory usage, execution costs, and latency.

Julie Letchner, Christopher Ré, Magdalena Balazinska, and Matthai Philipose wrote "Challenges for Event Queries over Markovian Streams," which presents query processing over data streams that can be modeled via HMMs. Such HMMs provide a flexible compact representation for discrete-valued time series, permitting the DBMS to represent infinite data streams in finite memory.

"Classifying Data Streams with Skewed Class Distributions and Concept Drifts," by Jing Gao, Bolin Ding, Wei Fan, Jiawei Han, and Philip S. Yu, describes methods for performing data classification over data streams with highly skewed data distributions and changing concept characterizations. The authors address the problem of classifying streams of credit-card transactions into normal versus fraudulent ones.

Finally, "Placement Strategies for Internet-Scale Data Stream Systems," by Geetika T.

Lakshmanan, Ying Li, and Rob Strom, deals with task placement for data stream processing in large distributed networks. This is the data stream management version of classic query optimization problems in distributed data management.

The five articles presented in this issue address only a handful of the topics in data stream management. We anticipate that the growth in the use of smart sensors, microprocessors, networks, and the World Wide Web will fuel an explosion of demand for data stream management systems in the coming decade. More research is needed to support the design of such systems.

### Acknowledgments

We thank Fred Douglis, Doug Lea, Oliver Spatscheck, and the staff of *IEEE Internet Computing* for their great patience, support, advice, and editorial assistance, as well as for supporting the creation of this special issue. We also thank the authors of the submitted articles for their efforts, and the reviewers who carefully read, evaluated, and commented on the various papers and revised versions. We are grateful to the US National Science Foundation (NSF) for its support via the IPA grants to our home institutions (Lawrence Berkeley National Laboratory and the University of Oklahoma), which supported this work via the IPA program's independent R&D mechanism. Le Gruenwald thanks the University of Oklahoma for supporting this work since her return from the NSF.

**Frank Olken** is a database researcher at Lawrence Berkeley National Laboratory, presently detailed to the US National Science Foundation as a program director in the Computer and Information Science Directorate, Intelligent Information Systems Division. His research interests include query optimization, random sampling from databases, scientific databases, bioinformatics, metadata registries, ontology repositories, data semantics, social science data management, and workflow data management. Olken has a PhD in computer science from the University of California, Berkeley. Contact him at folken@nsf.gov.

**Le Gruenwald** is the Presidential and David W. Franke Professor and Director of the School of Computer Science at the University of Oklahoma. Her research interests include data stream management, sensor data management, mobile data management, information privacy and security, and bioinformatics. Gruenwald has a PhD in computer science from Southern Methodist University. Contact her at ggruenwald@ou.edu.