Abstract

Data Stream Management systems (DSMSs) have become an active research area in the database community. Inspiration comes from potentially large application areas, e.g., network monitoring, sensor networks, telecommunication, financial and web applications. Until now, the performance requirements of stream applications (mainly financial) have driven most DSMS architects to venture for a dedicated (bounded) main-memory solution. To illustrate, in StreamBase, the application semantics are defined in a simplified SQL language framework from which a dedicated (Java) program is derived. A small runtime library complements the setup and most attention can be given to application specific input/output adapters. The drawback became clear as the applications demanded more functionality.

In this work, we describe a data stream management system, called MonetDB/DataCell, starting at the other end of the spectrum. We claim that an efficient data stream management system can be built on top of an extensible database engine. We present our vision in the context of the currently emerging column-stores. DataCell is implemented on top of the MonetDB system, an open source column-oriented DBMS. The column orientation together with its clearly defined software stack provides an excellent basis for exploration of novel database techniques.

The main contributions and topics addressed in this work are the following. The DataCell generalizes the sliding window approach predominant in DSMSs to allow for arbitrary table expressions over multiple streams and persistent tables interchangeably through predicate windows. It enables applications to selectively process the stream and prioritize event processing based on application semantics. In addition, the DataCell processing engine is geared at bulk processing of events to amortize overhead incurred by process management and function calls. This favors a skewed arrival distribution, where a peak load can be handled easily, and possibly within the same time frame as an individual event. It capitalizes the performance offered by column-store database systems. Stream applications require the expressiveness of SQL’03. We do not resort a redefinition of the WINDOW concept. Instead, we propose an orthogonal extension to SQL’03. Moreover, the complete state of the system can at any time be inspected using SQL queries.