An Extensibility Approach for Spatio-temporal Stream Processing using Microsoft StreamInsight

Jeremiah Miller[1], Miles Raymond[1], Josh Archer[1], Seid Adem[1], Leo Hansel[1], Sushma Konda[1], Malik Luti[1], Yao Zhao[1], Ankur Teredesai[1], Mohamed Ali[2]

[1] Institute Of Technology, University of Washington, Tacoma, WA, USA
{ jeremmi, reukiodo, archerjb, seidom, lhansel, skonda, mluti, yaozerus, ankurt} @u.washington.edu

[2] Microsoft StreamInsight, Microsoft Corporation, Redmond, WA, USA
{mali}@microsoft.com

Abstract. Integrating spatial operators in commercial data streaming engines has gained tremendous interest in recent years. Whether to support such operators natively or to enable the operator through an extensibility framework is a challenging and interesting debate. In this paper we leverage the Microsoft StreamInsight^{TM} extensibility framework to support spatial operators enabling developers to integrate their domain expertise within the query execution pipeline.

We first justify our choice of adopting an extensibility approach over a native support approach. Then, we present an example set of spatiotemporal operations, e.g., KNN search, and range search; implemented as user defined operators using the extensibility framework within Microsoft StreamInsight. More interestingly, the demo showcases the how embedded devices and smartphones are shaping the future of streaming spatiotemporal applications. The demo scenario specifically features a smartphone based input adapter that provides a continuous stream of moving object locations as well a continuous stream of moving queries. To demonstrate the scalability of the implemented extensibility framework, the demo includes a simulator that generates a larger set of stationary/moving queries and streams of stationary/moving objects.

Keywords: Microsoft StreamInsight, extensibility, spatiotemporal data streaming, geostreaming.

1 Introduction

It has been a debate in the database community whether to support spatial operators natively or to provide an extensibility framework capable of integrating user defined operators across multiple domains within the query execution pipeline. This same debate has naturally evolved to include the data streams domain as more and more business scenarios demand spatiotemporal stream processing. In this demo, we explore the utility of the second approach (extensibility) to design a set of spatiotemporal streaming operators.

1.1 Motivation

Spatial queries are becoming increasingly popular and are a necessary building block for a number of location-enabled applications (e.g. find coffee shops near me as I travel down the freeway, or alert the police cars nearest to a fleeing suspect.) Our approach utilizes a commercial data stream management system (DSMS), Microsoft StreamInsight, to handle streams of spatiotemporal location data. We leverage the extensibility features to update and query our spatial indexes. StreamInsight natively handles the details of the streaming data, so that we can focus our development efforts on the business logic implementation of real-time spatial queries.

1.2 The Case for Extensibility in a DSMS

Microsoft StreamInsight (StreamInsight, for brevity) is a commercial DSMS that adopts the semantics of the *Complex Event Detection and Response* (CEDR) project [1-2]. The underlying basis for processing long-running continuous queries in StreamInsight is a well-defined temporal stream model and operator algebra. It correlates stream data from multiple sources and executes standing queries on a low-latency query processor to extract meaningful patterns and trends [3-4].

Several business domains have explored the value that could be gained by using DSMSs to process real-time workloads. In previous work, StreamInsight has been used as a platform for web click analysis and online behavioral targeting [5], computational finance [6], and spatiotemporal query processing [7-9]. Because of the wide applicability of data streaming in many domains, and because of the domain expertise involved in each domain, the extensibility model provides a way to make DSMS capabilities more readily accessible to developers within their domain of expertise. For this demo, we leverage StreamInsight's extensibility model in the spatiotemporal domain. We discuss StreamInsight's extensibility model [10] in Section 2.3.

1.3 Applying StreamInsight to the Spatial domain

Ali et al present two approaches (the extensibility approach and the native support approach) to enable spatiotemporal query processing in DSMSs and, more specifically, in StreamInsight [7]. Some have also investigated the extensibility approach [8-9] to extend StreamInsight with the capabilities of the SQL Server Spatial Library [11]. However, the SQL Server Spatial Library is tuned for non-streaming data and, hence, performance at real time remains an issue. Using the extensibility framework proposed by Ali et al [10], incremental streaming-oriented versions of spatial operators can be developed and integrated with the query execution pipeline. In particular, we utilize user defined operators (UDOs) in our demo.

This demo implements a set of K- nearest neighbor (KNN) search and range search operators. These operators receive a stream of location updates. Note that there are various flavors of the KNN search and range search problems. Typically, a fixed KNN search or range search can be posed against a stream of continuously moving objects (i.e. querying device is not moving, the query answers are derived from a stream of moving entities). Alternatively, the KNN search or the range search center can be moving while the objects are stationary – e.g. show me locations of coffee shops as I drive along on a freeway. More interestingly, both the objects and the search center can be moving; e.g. a user and the friends nearest to them at any given time in a public location such as a shopping mall. We support all these flavors in the demo.

2 StreamInsight Overview

This section summarizes the major features of StreamInsight and gives an overview of its developer's interface with an emphasis on the extensibility framework.

2.1 Capabilities of StreamInsight

Speculation and consistency levels: StreamInsight handles imperfections in data delivery and provides consistency guarantees on the resultant output. Such consistency guarantees place correctness measures on the output that has been generated so far, given that late and out-of-order stream events are still in transit.

Windowing Semantics: Windowing is achieved by dividing the time-axis into a set of possibly overlapping intervals, called windows. An event belongs to a window if and only if the event's lifetime overlaps with the window's interval (time span). The desired operation (e.g., sum or count) is applied over every window as time moves forward. The output of a window is the computation result of the operator over all events in that window, and has a lifetime that is equal to the window duration.

Scalability: Scalability is achieved by both stream partitioning and query partitioning. Stream partitioning clones a query into multiple queries (of the same operator tree) such that each query operates on a portion of the stream. Query partitioning divides a query into many sub queries, each deployed on an instance of StreamInsight.

Debugging: StreamInsight provides a graphical tool (the Event Flow Debugger) for the inspection of event flow in a query as a means of debugging and performing root cause analysis of problems. The Event Flow Debugger reports the per-query memory and CPU usage, latency, throughput, and other runtime statistics as well.

2.2 Developing a Streaming Solution with StreamInsight

To develop a streaming application using StreamInsight, a set of modules have to be written to interact with the system. These modules are classified as:

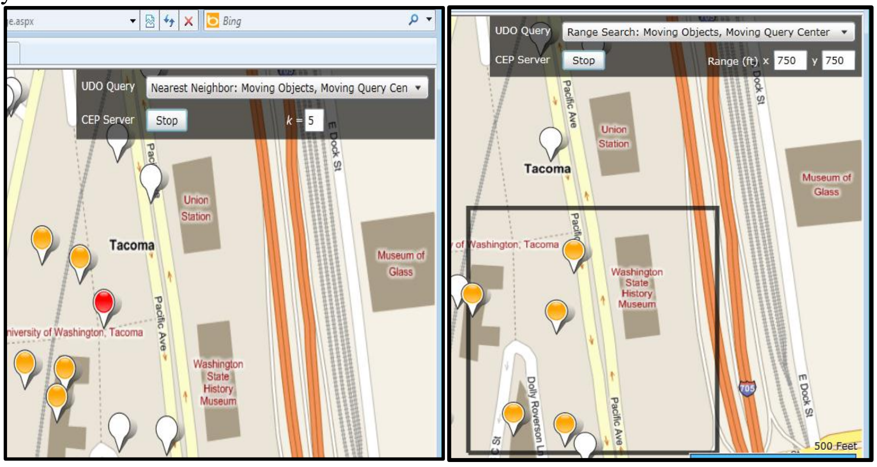
Input/output Adapters: Input data streams are fed to the streaming engine through the appropriate input adapters. Input adapters have the ability to interact with the stream source and to push the stream events to the streaming engine. The engine processes the queries issued by the user and streams the resultant output to the consumer through output adapters. For the demo, we wrote two input adapters – one that generates test data and one for real data streamed from smartphone apps. We also created an output adapter that provides the output data as a service consumed by our display applications on a PC or smartphone.

Declarative Queries in LINQ: Language Integrated Query (LINQ) [12] is the approach taken by StreamInsight to express continuous queries. The LINQ that invokes our spatial query, for example, might look like this:

```
var outputStream = inputStream.Scan(
  new RTreeUDO( queryType, xRange, yRange));
```

This scans the input stream with our R-tree UDO, and provides us with the output stream. The code inside our UDO will accept a set of input event to produce a set of output events. We define the payload of our input and output events to include geographic location, phone ID, and IP address.

User Defined Operators (UDOs): The extensibility framework enables domain experts to extend the system's functionality beyond relational algebra. Domain experts package their logic as libraries of UDOs that are invoked by the continuous query.



UDO Query
Nearest Neighbor: Moving Objects, Moving Query Cen
CEP Server
Stop
k = 5
Tacoma
Union Station
Museum of Glass
Washington State History Museum
Pacific Ave
E Dock St
University of Washington, Tacoma
UDO Query
Range Search: Moving Objects, Moving Query Center
CEP Server
Stop
Range (ft) x 750 y 750
Dolly Roverson Ln
500 Feet

Fig. 1. Query output visualization of KNN search and range search query using the dashboard.

StreamInsight's extensibility framework addresses two types of UDO developers. The first type is software developers who are not trained to think under the data streaming paradigm with its temporal attributes: for them there is the non-incremental model which provides a relational view of the world. The second type is developers of streaming applications where temporal attributes are first class citizens in their business logic. These developers seek maximum achievable performance through incremental query processing and may require full control over the temporal attributes of events as well. For them there is the incremental model, which provides the deltas or changes in the input to the UDO since the UDO's last invocation.

3 Demo Scenario

The demo scenario features a dashboard (Figure 1) that simultaneously visualizes a set of stationary objects (e.g., landmarks, coffee shops, shopping malls) and a set of moving objects (e.g., streamed from handheld devices and smartphones) using Bing Maps [13]. The dashboard is also used to compose and issue continuous queries against these objects. There are two versions of the dashboard: a PC-based dashboard and a smartphone-based dashboard. Note that a query that is issued using the smartphone based dashboard is assumed to have a continuously moving query center (as described in Section 1). The queries are processed using a data streaming engine and the query results are visualized using either version of the dashboard.

The demo utilizes Microsoft StreamInsight as the underlying data streaming engine. To interface with StreamInsight, a set of input/output adapters, a set of LINQ queries and a set of user defined operators (UDOs) are developed. We provide two types of input adapters. The first type of input adapters streams, at real time, GPS locations from handheld devices (e.g. smartphones) to the data streaming engine. The second type of input adapters simulates a larger set of moving objects/queries for the sake of demonstrating scalability. The simulator moves the device locations by random displacement vectors to simulate the variability we expect in real data. Moreover, it can replay historical logs of moving objects and feed them to the streaming engine to process queries over historical data. On the output side, the output adapter visualizes the query results using Bing Maps over the PC-based dashboard or sends the result back to be visualized at the dashboard of the moving object (say, a smartphone) that issued the query.

We implement R-tree and M-tree spatial index structures to track the objects as they roam the space. Our UDOs incrementally update and query these indexes at real time. In this demo, we present UDOs for KNN search and range search operations such that the various flavors of stationary/moving queries issued against stationary/moving objects are supported. Finally, LINQ queries are automatically composed and instantiated through the dashboard. These queries invoke the UDOs along with a set of relational operators (filters and projections) to declaratively define the requested output.

4 References

1. Roger S. Barga, Jonathan Goldstein, Mohamed Ali, and Mingsheng Hong. Consistent Streaming Through Time: A Vision for Event Stream Processing. In Proceedings of CIDR, 412-422, 2007.
2. Jonathan Goldstein, Mingsheng Hong, Mohamed Ali, and Roger Barga. Consistency Sensitive Streaming Operators in CEDR. Tech. Report, MSR-TR-2007-158, Microsoft Research, 2007.
3. B. Chandramouli, J. Goldstein, and D. Maier. On-the-fly Progress Detection in Iterative Stream Queries. In VLDB, 2009.
4. B. Chandramouli, J. Goldstein, and D. Maier. High-Performance Dynamic Pattern Matching over Disordered Streams. In VLDB, 2010.
5. Mohamed Ali et al. Microsoft CEP Server and Online Behavioral Targeting. VLDB 2009.
6. Badrish Chandramouli, Mohamed Ali, Jonathan Goldstein, B. Sezgin, B. Sethu. Data Stream Management Systems for Computational Finance. IEEE Computer 43(12): 45-52 (2010).
7. Mohamed Ali, Badrish Chandramouli, Balan Sethu Raman, Ed Katibah. Spatio-Temporal Stream Processing in Microsoft StreamInsight. IEEE Data Eng. Bull. 33(2): 69-74 (2010).
8. Jalal Kazemitabar, Ugur Demiryurek, Mohamed Ali, Afsin Akdogan, Cyrus Shahabi, Geospatial Stream Query Processing using Microsoft SQL Server StreamInsight, In VLDB, 2010.
9. Mohamed H. Ali, Badrish Chandramouli, Balan Sethu Raman, Ed Katibah: Real-time spatio-temporal analytics using Microsoft StreamInsight. In ACM GIS 2010.
10. Mohamed Ali, Badrish Chandramouli, Jonathan Goldstein, Roman Schindlauer. The Extensibility Framework in Microsoft StreamInsight, In ICDE, 2011.
11. SQL Server Spatial Library, http://www.microsoft.com/sqlserver/2008/en/us/spatial-data.aspx, last accessed in March 2011.
12. Paolo Pialorsi, Marco Russo. Programming Microsoft LINQ, Microsoft Press, May 2008.
13. Bing Maps, http://www.bing.com/maps