Live Analytics on High Velocity Sensor Data Streams using Event-based Systems

Naveen Nandan
SAP Research, Singapore
Email: naveen.nandan@sap.com

Abstract—With the advancement in sensor technology and the penetration of mobile devices, event-based systems are rapidly gaining importance in many big data application domains such as complex event processing and real-time analytics. A considerable amount of effort is being put in by the research communities to focus on developing systems that can analyze, forecast and enable decision making for pervasive sensing applications. This paper describes methods for running real-time complex analytics on high velocity sensor data with the focus on continuous computation of statistics and visualization. The system developed demonstrates real-time analytics for a game of football.

Index Terms—continuous computation, real-time analytics, real-time visualization, sensor data mining.

I. INTRODUCTION

All the information that drives the web today is increasingly taking the form of data streams. By definition, “A data stream is a real-time, continuous ordered (implicitly by arrival time or explicitly by timestamp) sequence of items. It is not possible to control the order in which items arrive, nor is it feasible to locally store a stream in its entirety” [1]. Mostly this form of data is generated by information systems such as telecommunication call records, network management systems, industrial monitoring systems, internet applications such as recommendation systems, click streams, social networks, etc. Generic domains such as financial or stock-market applications, sensor networks, etc. generate data, which are by nature, in streaming fashion.

An evolution towards structured data can be foreseen with the advent of Web 3.0 (Semantic Web) around the corner. Putting these two together, the spotlight tilts towards being able to process and analyze all of this data on the fly. This paradigm of extracting knowledge from continuous, rapid, high-volume, highly-variable data streams is referred to as “Real-time Data Stream Mining”. Through this paper, we discuss high-velocity data stream mining that is applied to sensors used in sports.

The system presented in this paper performs real-time analytics on sensor data for a game of football, which is processed by an event stream processing engine. The raw sensor data is collected using the Real-Time Locating System (RedFIR©), developed by the Fraunhofer Institute for Integrated Circuits IIS, and deployed at the Nuremberg Stadium in Germany.

The sensor data used to develop/test the system was recorded during a test game played in the Nuremberg stadium. The game was played on one half of the field by two teams of eight players each. Players were equipped with a number of location sensors embedded into their kits. Every ball used in the game was also equipped with a sensor. The position of players and the ball was reported up to two thousand times per second with the precision of a few centimeters. Such high frequency updates, in a way, aids the system developed to provide precise, real-time analytics.

Real-time analytics provides unprecedented insight for both managers of football teams as well as spectators enjoying the game. It is of invaluable help when it comes to the quick in-game decisions. Using real-time analytics team managers/coaching staff would have a competitive edge, allowing them to make better decisions on replacements and strategy changes based on live facts and statistics [2].

II. EXPERIMENTAL SETUP

The data is measured using the RedFIR© system developed by the Fraunhofer Institute of Integrated Circuits IIS in Erlangen and deployed at the Easy Credit Stadium in Nuremberg. RedFIR© is based on radio technology that provides real-time location of objects. The miniature transmitters were attached to players, one per foot, and one to each ball. These signals were picked up by the receiver antennas surrounding the area.

Figure 1. Representation of the football field on which the test game was played. The game was played on one half of the field.
The game was played on one half of the football field with 8 players in each team. The goalkeepers have additional sensors attached to each arm. In summary, the system could track 16 players, 1 referee and 4 balls.

The transmitters attached to the players are sampled at 200Hz (approximately 200 events per second) and the ball at 2000Hz (approximately 2000 events per second). On an average, a system that would analyze this stream would receive about 15000 events per second.

III. DATA STREAM MINING

A lot of the research focus, effort and money are being invested in the area of data stream mining by both industry and academia to address the pressing needs of architecting real-time or close-to-real-time systems.

STREAM: The Stanford Data Stream Management System [3] was built on top of contextual query language (CQL) focused on query optimization for memory management. The system made use of “synopses” to approximate results based on summarized information. TelegraphCQ developed at Berkeley [4] was an extension of PostgreSQL using a continuous querying mechanism of CQL type. This system addressed a key requirement of being able to add new queries dynamically. Aurora (superseded by Medusa, then Borealis) [5] brought in the new dimension of scalability through “distributed stream processing”. Other systems that were similar in nature include NiagaraCQ [6] which is a scalable continuous query system for web-based (XML) databases, StatStream [7] for statistical monitoring of multiple data streams, StreamMiner [8] which is a classifier ensemble-based data mining engine, Gigascope—a Stream database [9] and Hancock (a C programming language variant) from AT&T was developed for mass surveillance of their massive networks.

The interesting thing about most of the above systems is that they gained attention from the big names in the market and hence, further development was shifted to the industrial research labs. Streambase is one such system that spun-off from Aurora and focuses on high-performance Complex Event Processing (CEP). Coral8 another famous CEP tool, designed on top of a publish-subscribe architecture, has been used widely by the likes of Microsoft and IBM. The engine that evolved from STREAM is maintained by SAP as part of the SAP Sybase Event Stream Processor. TelegraphCQ became widely popular with the telecommunication network sector, and was integrated by Cisco into their network management platform—Cisco Prime.

By convention, real-time data stream mining systems are built either as batch processing systems or, the more recent, continuous (real-time) processing frameworks.

Batch processing systems, such as “Hadoop” (from Apache Software Foundation and Cloudera), store/buffer raw data streams (files, web, images, etc.) over a period of time and process them at frequent intervals. This in turn is done in two stages where the input (files) is distributed over multiple machines and then a Map-Reduce operation is performed. The results are then pushed to a data-store or a subscribed client. To achieve close to real-time processing, the regular interval in which the input is distributed is narrowed down.

A couple of years ago, BackType (now acquired by Twitter) came up with a promising continuous distributed real-time processing framework called “Storm” which addresses most of the requirements for a system of its genre, such as reliability, robustness, fault-tolerance, scalability, etc. Pachube (now Cosm) is another system that was developed as a platform to address the idea of “The Internet of Things”. This platform acquires data from various external devices and handles them in real-time. DataSift provides a social media data platform which brings in multiple data streams together and aids in performing analysis over them through the steps of extraction and reduction of data. Most of these systems use ZeroMQ and Apache Zookeeper for message queuing and coordination respectively. Other comparable systems are Esper, Streambase, Hstreaming and Yahoo S4. A system such as that of DataSift’s has an estimated data volume that adds up to 1 TB each day.

For this system, we make use of the SAP Sybase Event Stream Processor for running the queries on the soccer sensor data stream.

The first of the queries formulated was to calculate the analysis of the running performance of each of the players currently participating in the game. The intensities are defined as: standing (0-1 km/h), trot (till 11 km/h), low speed run (till 14 km/h), medium speed run (till 17 km/h), high speed run (till 24 km/h), and sprint (faster than 24 km/h). Fig. 2. shows the possible transitions between different states which need to be observed for the running analysis.

The second query formulated was to calculate the ball possession for each of the players as well as for the teams as a whole. A player is considered to be in possession of the ball whenever it is in his proximity (less than one meter away). The ball possession is calculated as time between the first ball contact and the last ball contact.
Another complex query formulated was to dynamically derive the heatmap for each player i.e. considering the field as a grid split into a number of rows and columns, the query must predict the duration each player spends in all of the grid points.

With prior knowledge of dimensions, such as the length ‘x’ and width ‘y’ of the field and that the field is to be divided into ‘r’ rows and ‘c’ columns. Also, the coordinates of the origin is known and denoted as ‘originX’ and ‘originY’. Using this, an algorithm was devised to compute the current grid point of a player based on his position coordinates.

```java
divX ← x / c;
divY ← y / c;
while(i ≤ r) {
    j ← 1;
    while(j ≤ c) {
        x1 ← originX + (divX * (j - 1));
        x2 ← x1 + divX;
        y1 ← originY + (divY * (i - 1));
        y2 ← y1 + divY;
        if(player_within_polygon(x1,y1,x2,y2){
            return current_cell(i, j);
        }
        j++;
    }
    i++;
}
```

Figure 5. Algorithm to compute the current grid position of a player.

IV. SYSTEM ARCHITECTURE

The system developed consists of the following components:-

1) Event generator
2) Event stream processing engine
3) Query result subscriber
4) Websocket bridge
5) Visualization

A. Event Generator

One of the important requirements to develop/validate this system was to be able to generate the events at the exact same rate as it occurred and maintain the inter-arrival time between events. In order to achieve this, a high-frequency multi-threaded scheduler (as shown in Fig. 6.) was implemented that could read off the file containing the raw sensor data and asynchronously publish it to the event processing engine.

B. Event stream Processing Engine

For deploying the stream mining queries, we use the SAP Sybase Event Stream Processor. In order to achieve high performance, the queries are broken down into multiple small operators, with each flow of operators cascaded to produce a desired result. Each operator is custom built using a built-in scripting language called stream processing language shell (SPLASH) that helps to extend the engine’s query language called continuous computation language (CCL).

C. Query Result Subscriber

The query result subscriber is an interface between the event processing engine and the external visualization stack. It is a simple listener API that gets notified every time there is an update in the result stream.

D. Websocket Bridge

After evaluating multiple visualization frameworks, the challenge was to ensure high performance, both in terms of rendering the visualization and making the data points available at the visualization layer. Since the data rate was very fast, the best method was to be able to push the events into the visualization layer, instead of the visualization stack having to poll the events. For this, we make use of the Websocket implementation of the HTML5 stack, which is a high speed transfer protocol for the web.

E. Visualization

Two categories of visualizations are generated: 1) a real-time analytical dashboard visualizing the statistical results from the queries in the form of charts, treemap, etc. 2) a browser based real-time view of the game and a
V. Real-Time Visualization

Rohrdantz et al. [10] summarize a few research questions that are still open and need to be addressed in the field of real-time visualization of streaming text data, which interestingly also applies to visualization of all types of streaming data in general. One of the major concerns with high dimensional data is how to present multiple focus points in a single visualization for a real-time stream. There could be cases when certain data streams need a shift of focus to historic data to arrive at a real-time consensus.

In general, systems are built to address continuous data streams. But usually, this is not the case. The data streams could emit data intermittently causing a fluctuation in the data available to the visualization. On the other hand, the visualization itself could be changing permanently for real-time streams. Interaction within an analytic and reasoning context has to be supported without interrupting the live stream. Another concern for many data stream mining experts is how to perform analysis on multiple streams of data, as often, questions to which an analysis is directed towards is context specific. As most of the analysis would involve temporal alignment (clock synchronization) of multiple data streams, domain-dependent parameterization (data preparation), etc. it is to be investigated if standard methods can be applied to every stream. Also, when there are multiple sources of data, there is a high chance of an overlap of data points. Hence, a mechanism to differentiate such overlap is needed for both storage and processing.

In this system the results of the ESP queries are consumed by a number of visualizations. Fig. 8. shows a Panopticon® dashboard which provides a running analysis for each of the players on the field. Using this dashboard a team manager can easily compare the performance of each of the players and take decisions on replacements in case of underperformance. Using Panopticon®, the team manager can also get a real-time view of the ball possession for each of his players and monitor how it changes over time. This easily allows spotting the playmakers on the field. For example, from Fig. 9, one can clearly see that a player can have a higher ball possession even though he has run a shorter distance in comparison to other team members.

Figure 8. Real-time analytics dashboard that can display the player statistics such as total distance traversed, average speed/distance over a window of time and the overall game.

One of the useful methods to visualize spatio-temporal data would be a heatmap that has varied intensity based on the amount of time each player spends over the field. But, the challenge here was to identify a method to account for the constant change in the intensity simply due to the high frequency nature of the data. For this, we make use of a high-performance WebGL based javascript framework that helps generate a dynamic heatmap of the game.

In Fig. 10, we can see the heatmap that illustrates how and where the players spent most of their time over the duration of the game. It allows both spectators and team managers to watch live the changing strategy of the game.

Figure 9. Real-time treemap view indicating the ball possession of each player.

Figure 10. Heatmap visualization of players of both the teams over the whole game. (a) First half, Team A, (b) First half, Team B, (c) Second half, Team B, (d) Second half, Team A.

The generated real-time analytics can be consumed by a variety of devices and is not only of benefit for the team manager, but it can also provide valuable services to spectators or broadcasters. Fig. 11. shows a live game visualization which can be embedded on any website and consumed by any device.

Figure 11. Real-time HTML5 view of the game on a tablet.
VI. CONCLUSION

In this paper we have presented a data stream mining system that can derive live analytics over high-speed data streams and also visualize the results in real-time. The football sensor stream was used in order to highlight the high-frequency of events that can be generated from sensor networks. The sports analytics use case can be extended to other domains that contain similar spatial-temporal data such as vehicle tracking, smart city management, etc. The idea would be to extend such a system to address 1) multiple high-velocity data streams 2) distribution of the operators on multiple machines and, 3) develop more intuitive methods of data visualization.

ACKNOWLEDGMENT

We would like to thank Fraunhofer IIS, Erlangen for developing the RedFIR© radio system, deployment of the sensors and data collection; the Economic Development Board and National Research Foundation of Singapore for supporting this research which can be extended and applied to any generic geo-spatial real-time application domains; to our research colleagues from the Real-time Intelligence Program, SAP AG, Dresden.

REFERENCES


Naveen Nandan is from India and was born on March 06, 1988. He received a Bachelor of Technology in Electronics and Communication Engineering from Amrita Vishwa Vidyapeetham, Coimbatore in 2009. He continued to pursue Master of Science at the School of Computer Engineering, Nanyang Technological University, Singapore where his thesis work was on “Multi-class Emotion Detection from Suicide Notes” applying techniques from natural language processing for feature extraction, association rule mining for feature selection and machine learning for classification.

He started off his career as SYSTEMS ENGINEER with the Education & Research Department at Infosys, Mysore. He spent a brief stint as SYSTEMS ENGINEER with JamiQ.com, a social media intelligence startup in Singapore, after which, he worked as SOFTWARE ENGINEER (PLATFORM) on the LIVE Singapore! project with the MIT SENSEable City Lab at the Singapore-MIT Alliance for Research and Technology under the Future Urban Mobility program. He currently works for SAP Research, Singapore with the Real-time Intelligence Program as RESEARCH & DEVELOPMENT ENGINEER. His research interests include distributed & real-time systems, software engineering, complex event processing, data mining, machine learning, natural language processing and data visualization.