BCE-Processor: Boost Up Event Processing for Large-scale Sensor-rich Ubiquitous Environment

Jinwon Lee[1], Youngki Lee[1], Seungwoo Kang[1], SangJeong Lee[1], Yunseok Rhee[2], Junehwa Song[1]
[1]Computer Science, KAIST
Daejeon, 305-701, Republic of Korea
[2]Hankuk University of Foreign Studies
Yongin, 449-791, Republic of Korea
{jcircle, youngki, swkang, peterlee, junesong}@nclab.kaist.ac.kr, rheeys@hufs.ac.kr

1. INTRODUCTION

In this work, we are designing and developing a *novel high-performance event detection framework* to address the unique challenges of advanced large-scale monitoring applications, proliferating in the near future (e.g., large-scale urban sensor networks, surveillance, logistics, battlefield watcher). We envision that a fully-functioning event processing framework will be compelling in upcoming sensor-rich ubiquitous environments to facilitate the monitoring applications. On top of it, a new generation of applications will thrive to enrich personal as well as public services.

The emerging large-scale monitoring applications will be situation-aware; the applications proactively detect the occurrences of interested events and trigger proper actions promptly. They commonly rely on massive, real-time event processing over voluminous data streams continuously generated from numerous high-rate sensors, mobile devices, or agents that are increasingly deployed in surrounding spaces or even on the Internet. Moreover, individual users issue numerous monitoring requests, personalized to their own needs; they also expect real-time responses and are not tolerant of stale events and delayed responses. These remarkable scale and complexities of the applications will definitely make existing event processing systems almost impossible to support them effectively.

As the first attempt to enable such large-scale monitoring applications, we propose *BCE-Processor*, a high-performance Border Crossing Event (BCE) detection framework. It effectively achieves semantic and processing requirements newly arising in the large-scale, senor-rich ubiquitous environments. We take note of the practical importance of BCEs. A BCE is intuitively represented as a data stream crossing the borders of a user-specified interest range. Many scenarios [2][5] show that border crossing is an important and frequently-used primitive event. To facilitate the massive processing of BCEs, we specify a set of BCEs in relation to an interest range in a data-centric manner. More importantly, we develop a high-performance processing mechanism, which evaluates a large number of queries over voluminous data streams in a shared and incremental manner. Thereby, BCE-Processor achieves excellent processing performance and low storage cost. In particular, we design BCE-Processor that supports a one-dimensional as well as a multi-dimensional border. We are currently working on supporting diverse shapes of interested borders such as convexes, general polygons and circles beyond rectangles to fully reflect practical requirements of various real-world applications.

We believe that, as a primitive event, BCE will serve various useful complex events, e.g., RTE (Region Transition Event). Note that BCE-Processor directly operates on the high-rate input data streams and generates much lower-rate BCEs by efficiently filtering out unnecessary data (see Figure 1). Thus, the performance of subsequent event processing can be significantly accelerated through BCE-Processor.

Our research can be considered as a step to bridge two independently evolved research efforts, i.e., data stream processing and event processing. More importantly, it overcomes their limitations for upcoming sensor-rich ubiquitous environments. In particular, we envision the necessity of high-performance event detection, i.e., extracts meaningful patterns of data (e.g., border crossing or point of inflection on sensor readings) as primitive events, and further elaborates on their performance issues. Using our research as a basis, we believe research on data stream processing will be extended to defining and detecting various primitive events. On the other hand, research on event processing can be enriched by composing the events derived from data streams as well as achieving high performance. An interesting research can be found on this line in EStream [3], which envisioned the necessity of combining the two domains ahead.

A prototype implemented on an off-the-shelf desktop PC with a CPU of 3 GHz effectively processes events generated by over 600,000 moving people, covering a city. Furthermore, we have built a couple of interesting monitoring applications such as a ubiquitous taxi dispatcher and a battlefield watcher on top of BCE-Processor, and demonstrated them to the public.

2. RELATED WORK

Our study is the first attempt to detect a large number of primitive events in large-scale sensor-rich ubiquitous environments. DSMSs (Data Stream Management Systems) [4] have recently been receiving an attention, focusing on generic system-level abstractions and performance optimizations for stream-based monitoring applications. However, event processing was not the main concern in this context and has not been studied extensively. The focus of DSMSs was on supporting continuous queries, i.e., the extension of relational query language for continuous execution, where a basic processing unit is a data tuple.

Event processing [1] has been an important issue for a long time in many research areas, e.g. active databases, event processing systems, sensor networks, spanning diverse application domains, e.g., logistics, surveillance and facility management, B2B integration, healthcare. They concentrate on supporting various logical and temporal compositions (e.g., SEQ, AND, OR, NOT)

of inputs, where a unit of composition is a primitive event. Event processing systems are expressive enough to specify diverse events on data streams. However, they are still premature to support upcoming large-scale system environments. As mentioned, such systems should be able to effectively handle a massive amount of inputs, requiring highly scalable processing. The trigger mechanisms in DBMSs are limited in scale; only a few triggers per table are allowed. Moreover, in existing systems, the number of registered events as well as the input rates has not been assumed to be very high, e.g., compared to those for data stream processing [3]. They would increase significantly in the upcoming sensor-rich ubiquitous environments.

3. EVENT DETCTION FRMAWORK

3.1 Event Detection Query

We expect that future event detection queries will handle a number of data sources at the same time in a uniform way. Such a query specification will proliferate as large-scale applications tend to be interested in identifying events collectively regarding all data stream sources rather than the events for a specific source. In many sensor network and location-based systems, such an approach is importantly recognized as a data-centric paradigm.

Currently, we propose a Border Monitoring Query (BMQ) to collectively detect BCEs over all input data streams from a large number of sources. Given an interest range parameter, BMQ detects I and O events on data streams, where I and O events representing a data stream coming into or going out from an interest range, respectively.

3.2 Framework Overview

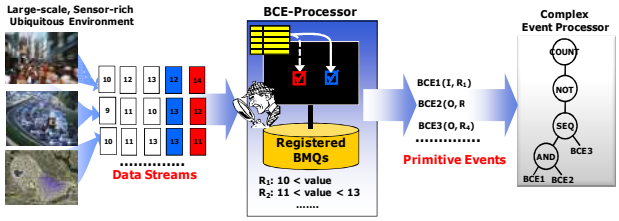

Fig.1. Framework overview and processing flow

Our work targets middle-tier framework between large-scale advanced monitoring applications, and mobile and sensor networks. Atop our framework, various applications can be easily developed by registering their own monitoring queries and run concurrently. As shown in Figure 1, the framework continuously receives a huge amount of data streams from various large-scale mobile and sensor networks (e.g., 3G phone network, WiMAX), e.g., the locations of people, vehicles and environmental information. BCE-Processor efficiently detects BCEs as primitive events from massive data streams. We are designing and implementing a complex event processor which incorporates efficient BCE composition and aggregation techniques to support advanced complex events such as RTE.

3.3 High-Performance Event Detection

As an initial approach to achieve high performance, BCE-Processor develops a shared and incremental processing mechanism. For shared processing, BCE-Processor adopts a query indexing approach, thereby achieving a high level of scalability. Once BCE-Processor is built on registered queries, only relevant queries are quickly searched for upon an incoming data. The main innovation of BCE-Processor compared to previous approaches is that BCE-Processor develops and operates over a *stateful index*. Existing query indices are stateless and optimized only for one-time searching. However, it is extremely important to optimize the index for consecutive searching since the query index is repeatedly searched as data continuously arrives. The proposed BCE-Processor holds the state of the last evaluation. It is structured so that, upon a new data input, the evaluation is efficiently done by starting the operation from the last state.

For incremental processing, BCE-Processor utilizes the locality of data streams. Data updates usually exhibit gradual changes more often than abrupt ones. Thus, in many cases, the matching query set for a data update will be equal to or overlap much with that for the previous update. To fully utilize this fact, BCE-Processor calculates the difference of matching queries in advance and accordingly partitions a domain space. Upon data arrival, evaluation can be quickly done by simply traversing a small number of the partitioned segments without any complicated computation.

To further improve performance, we are considering applying various techniques such as event query planning, lazy evaluation, event graph sharing, and intermediate results reduction.

4. REFERENCES

[1] S. Chakravarthy and R. Adaikkalavan, “Ubiquitous Nature of Event-Driven Approaches: A Retrospective View (Position Paper), Proc. Dagstuhl Seminar 07191, 2007.

[2] K. Cho, I. Hwang, S. Kang, B. Kim, J. Lee, S. Lee, S. Park, Y. Rhee, and J. Song, “HiCon: A Hierarchical Context Monitoring and Composition Framework for Next Generation Context-aware Services”, IEEE Network, July 2008.

[3] V. Garg, R. Adaikkalavan, and S. Chakravarthy, “Extensions to Stream Processing Architecture for Supporting Event Processing”, Proc. DEXA, 2006.

[4] L. Golab and M. Tamer Ozsu, “Data Stream Management Issues – A Survey”, SIGMOD Record, 2003.

[5] J. Lee, Y. Lee, S. Kang, S. Lee, H. Jin, B. Kim, and J. Song, "BMQ-Index: Shared and Incremental Processing of Border Monitoring Queries over Data Streams”, Proc. MDM, 2006.