Complex Event Processing over Uncertain Data Streams

Xin Li[†] Hideyuki Kawashima[†,‡] Hiroyuki Kitagawa[†,‡]

[†] Graduate School of Systems and Information Engineering, University of Tsukuba

[‡] Center for Computational Sciences, University of Tsukuba

Tennodai,Tsukuba, 305-8577 Ibaraki, Japan

1. Introduction

In the past decade, the stream research has received considerable attention since the network and sensor device technologies have been developing rapidly. These sorts of data stream networks such as monitoring sensor networks, Radio Frequency Identification (RFID) networks, GPS systems, camera sensor networks, and radar sensor networks continue to expand into diverse aspects of daily life.

Sensor-kind network devices can generate raw data streams continually. Following the process and analysis the raw data streams, continuously arriving events could be transformed into complex event streams which are rigidly matched against complex event pattern and outputted. Several famous event processing systems for complex event matching have been proposed.

Since the data emanated from a variety of environment are incomplete, imprecise and even misleading, how to generate the complex event steams which can be trusted in is becoming an urgent problem. Thus, the purpose of this research is to calculate the confidence of complex event pattern outputs which are emanated from primitive physical events stream with noise information.

The target of this work is to provide a fundamental evaluation and optimization framework for complex event pattern over uncertain event stream. In this paper, we extend the semantics of SASE+ to support probabilistic data stream. Our main contributions consist of: first, extension of semantic of SASE+ to support complex event processing over uncertain data stream.

2. Probabilistic Event Matching Language

In this section, we present the model of our proposal which is used to match events according to pattern over probabilistic stream.

Probabilistic Event Stream Model

Inputs: The input to event processing system is a primitive uncertain event stream consisted of an infinite sequence of probabilistic events. Each probabilistic event from input has a concrete probability value used to present occurrence probability of this event. Here, we make assumption that each event is independent. Naturally, occurrence probability of each event must be less or equal to 1. On the other hand, the probability that the primitive event does not occur is (1-Pr(x)), where Pr(x) is the occurrence possibility of event x.

Outputs: the output of the event processing system is a composite probabilistic event stream, containing a sequence of output probabilistic events. Each output stream also contains probability, time and useful value to users. The type in output stream will be fixed by certain query.

Overview of the language

We extended and developed subset of SASE+ query language with probability. Our research focused on one of the four selection strategies: strict contiguity. Strict contiguity requires two selected events must be contiguous in the input stream. Let's introduce how to make a query to uncertain data stream. The query language structure of our model is following:

```
[FROM <input stream>]
PATTERN <pattern structure>
[WHERE <pattern matching condition>]
[WITHIN <sliding window>]
[HAVING <pattern filtering condition and
confidence condition>]
RETURN <output specification>
```

In order to understand the query language especially in confidence condition, we illustrate one simple example of RFID-use smart hospital.

Example query detects movement of one person in RFID-based monitoring library. It retrieves the item that one person take one book from the shelf and exit the library without registering.

```
[FROM    LIBRARY]
PATTERN SEQ(Shelf a, ~(Register b), Exit c)
WHERE
{    a.tag_id = b.tag_id
and a.tag_id =c.tag_id           }
WITHIN 12 hour
HAVING CONF(*)>0.50
RETURN S.time T.time Confidence
```

"~" is a negation make which denotes the non-occurrence of an event.

3. Query Processing

The framework of our system decoupled into two parts. The first one is a general deterministic state automaton for matching the pattern. The second one is, for probabilistic input stream, calculate the confidence degree.

Pattern can be easily found by automaton during certain stream processing. However, users require the confidence of output pattern over uncertain stream

Since uncertainty, for example, caused by noise, generates too many possible worlds, it becomes impossible to calculate probability of the whole possible worlds, especially when systems process huge volume uncertain data stream.

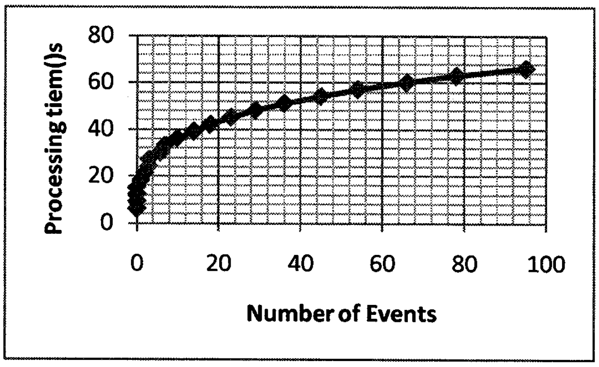
In the basic plan of, we combine binary tree to match the pattern. If the input data stream is "abcabc" we can obtain the responding sequences by the matching tree. The color of each node depicts the state in automaton. For example, green circle depicts the second state in automaton.

Root node indicates that the first input symbol a_1 triggered S_1, and create a new binary tree. And two children nodes of root will be generate when consider to consume b_1. The left child node indicates the b_1 occurs and the state is changed to the next one; on the other hand, the right child node $\overline{b_1}$ doesn't occur with a probability 1-Pr(b_1). Matching tree is developing until it overpasses the time sliding window. During this time, once one input takes one branch to the leaf node (final state), pattern sequences would be generated and outputted to users with confidence value. For example, after system processed c1, the sequence $a_1(P)b_1(P)c_1(P)$ will be outputted with probability of the pattern. And after system processed c_2, the three sequences $a_1(N)b_1(N)c_1(N)a_2(P)b_2(P)c_2(P)$; $a_1(P)b_1(N)c_1(N)a_2(N)b_2(P)c_2(P)$; $a_1(P)b_1(P)c_1(N)a_2(N)b_2(N)c_2(P)$; Each responding answer is corresponding one probability value which is used to output to user.

4. Experiment

To examine the performance of probabilistic event stream processing, we implemented a simulator, which includes a random event stream generator and an event query processor. In this experiment, we investigated the relationship between number of events and processing time. Here we process the query pattern (a b c) with the random input data which include a, b and c three instances. The results are all possible responsible answers to query which corresponding one probability value is.

We vary the number of events from 6 to 96, and examined performance of the system. From the figure we can observe the processing time is increasing when more events are processed.



5. Conclusions

The target of this paper was to realize a query language for compound event over uncertain steams. To output more confidential results we proposed computation framework.

Acknowledgement This research has been supported in part by the Grant-Aid for Scientific Research from JSPS(#21240005).