Fusion of Event Stream and Background Knowledge for Semantic-Enabled Complex Event Processing

Challenge Paper

Kia Teymourian, Malte Rohde, Ahmad Hasan, and Adrian Paschke

Freie Universität Berlin
Institute for Computer Science
Corporate Semantic Web Research Group
http://www.inf.fu-berlin.de/groups/ag-csw/
{kia, malte.rohde, ahmadhaidar, paschke}@inf.fu-berlin.de

Abstract. Usage of ontological knowledge about events and their relations to other concepts in the application domain can improve the quality of complex event processing (CEP). In this paper, we present a solution for knowledge-based event entity extraction using background knowledge bases. For our experiments we use the DeRiVE-2011 workshop dataset. We enrich the incoming event data stream with background knowledge from an external knowledge base, e.g., DB-Pedia so that our event processing engine has more knowledge about events and their relations to other concepts in application domain. [1]

Keywords: Complex Event Processing, Semantic CEP, Event Query Pre-Processing

1 Motivation

Semantic models of events can improve the quality of event processing by using event metadata in combination with ontologies and rules. The success of the Semantic Web research community in building standards and tools for semantic technologies such as formalized vocabularies/ontologies and declarative rules is opening novel research and application areas. One of these promising application areas is Semantic Complex Event Processing (SCEP), for which we previously proposed a new approach in [5, 4]. We claim that semantic models of events can improve the quality of event processing by using event data in combination with knowledge bases.

Existing methods for event processing can be categorized into two main categories, logic-based approaches and non-logic-based approaches [2]. One of the logic-based approaches is introduced in [1] which proposes a homogeneous reaction rule language for complex event processing.

In this paper, we describe a method for knowledge-based complex event processing to extract complex event entities from an event stream. Fusion of event data stream and background knowledge about events and other non-event objects in the application

[1] This work has been partially supported by the “InnoProfile-Corporate Semantic Web" project funded by the German Federal Ministry of Education and Research (BMBF) and the BMBF Innovation Initiative for the New German Länder - Entrepreneurial Regions.

domain can build up a complete knowledge about events and their relationships to other concepts. We use the workshop dataset to demonstrate how our method can be used for SCEP. In Section 2, we focus on use case scenarios and show which kind of complex events can be detected from the workshop data set using DBpedia[2] as a knowledge base. Section 3 describes our method for event processing which includes data fusion with the background knowledge base. Finally in Section 4, we describe our demonstration using SCEP.

2 Use Case Scenarios

The DeRiVE 2011 event dataset consists of over 100,000 events from music and entertainment websites. This dataset includes data about most popular concerts, festivals, kids events, sports events, and other social events. In the following, we describe three concrete complex situations where a person or an organizations defines a complex event query and is interested in detecting specific events from the upcoming event stream:
Scenario 1 - Specific Music Concerts Interests: Consider that Mr. Smith is interested in a special music type. For example he is especially interested in *"Canadian alternative rock music from the city Toronto"*. Mr. Smith lives in Europe and might be able to travel to different European cities when there are interesting concerts. Mr. Smith is married and has two children. For his travel plans, he has also to consider his family situation. He would like to travel only if there are some interesting kids or family events in the same city at the same time, so that they can travel together. His children like *"kids theater"*. If there are such upcoming events in combination, Mr. Smith would like to be informed in advance. This kind of application scenario can be seen as a real-time recommendation system, which may also trigger some automated reactions after the detection of a complex event.
Scenario 2 - Surveillance, Riot Prevention and Control: A security organization might be interested to know which concert types or which events are happening at the same time in the same cities, which might cause some potential conflicts between different fan parties. For example, if there are *"rock music concerts"* together with *"hip-hop music concerts"* in a small city at the same time, conflicts might arise if fans meet each other on the streets or in bars. In order to detect such potentially dangerous situations in advance, the security organization needs to be able to define its own high-level complex event queries, not only depending on the event data itself but also on background information, e.g. known hostility between fans of soccer clubs, etc.
Scenario 3 - Music Market Monitoring: A concert promoter company is eager to find out about gaps on the music concert market. It is interested to know in which European cities which types of concerts are organized and happening now. For example, when some outstanding *"60s hard-rock band"* goes onto its reunion tour, it might be the time to host a concert of a *"cheap cover band"*. Also, whenever a city is overwhelmed with concerts of a specific type of music, there may emerge a demand for other types of concerts. Thus, the concert promoter company needs to detect these event patterns early, so that they can organized concerts which are well attended by its fans.

[2] http://www.dbpedia.org, July 2011

3 Semantic Enabled Event Processing

The fusion of background knowledge with the data from an event stream can help the event processing engine to know more about incoming events and their relationships to other related concepts. We propose to use an external knowledge base which can provide background conceptual and assertional information about the events as it is shown in Figure 1. This means that events can be detected based on reasoning on their type hierarchy relationships, or temporal/spatial relationships. It can also be based on their connections to other relevant concepts from the domain, e.g., relationship of a concert event to the health situation of a band member. Some of the existing CEP systems [3] can integrate and access external static or reference data sources. But these systems do not provide any inferencing on external knowledge bases and do not consider reasoning on relationships of events to other non-event concepts.

The realization if SCEP is a challenging task, because it should provide real-time processing and high scalability. The naïve approach for SCEP might be a storage-based approach. This means to store all of the background knowledge in knowledge bases and start pulling the knowledge base, every time when a new event comes into the system, and then process the result from the external knowledge base with event data. This approach may have several problems when the throughput of the event stream is high, the size of background knowledge is high, or even when expressive reasoning should be done on the knowledge base.

Event Query Pre-Processing:

We propose to do an Event Query Pre-Processing (EQPP) before the event processing is down on the event stream. In this approach, the original complex event query can be pre-processed by use of a knowledge base and rewritten into a single *new query*. This *new query* is a query which can be syntactically processed only with the knowledge from the event stream and without an external knowledge base.

In this paper, we are addressing a simple pre-processing of event queries and illustrate the potential of such a pre-processing approach for SCEP. In our method the user query is pre-processed and rewritten into a single new query which has the same semantic meaning as the original one. The advantage of this method is that the user can define event queries in a high level abstraction view and does not need to care about some details, e.g., the user only defines *"alternative rock music band from the city Toronto"* and does not need to know all of the names of such music bands which might be a long list and might not be simple for humans to remember. One other advantage is that the SCEP system is able to provide real-time event processing as events arrive into the system because the external reasoning on knowledge base is done in advance. On the other side, one dist-advantage of this approach is that the query needs to be updated each time when the knowledge base is changed (or when a part of the KB is changed). We assume that in some of the use cases (like music concert events) the rate of background

[3] Several rule-based and storage-based event processing system are already proposed and developed, some of the commercial products are:
TIBCO BusinessEvents, `http://www.tibco.com/`, July 2011
Oracle CEP `http://www.oracle.com`, July 2011
Sybase CEP `http://www.sybase.de`, July 2011

knowledge updates is not very high as the rate of the main event stream, e.g., frequency of happening of music concerts compare to changes in a music band.

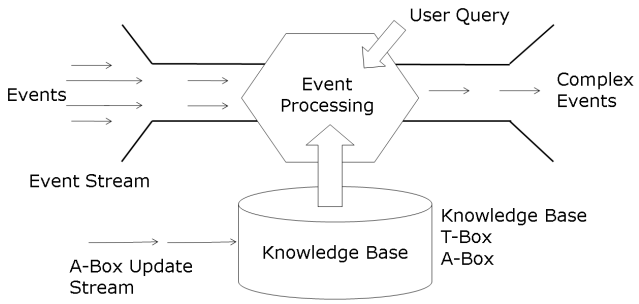

Fig. 1. High Level Architecture of Semantic-Enabled Complex Event Processing

4 Experiments and Demonstration

For our experiments, we use Prova[4] as reaction rule language formalization and as a rule-based execution which can be used as event processing engine. Prova uses reactive messaging[5], reaction groups and guards[6] for complex event processing. Multiple messages can be revived using revMult(XID, Protocol, Destination, Performative, Payload) ; XID, a conversation id of the message; Protocol, message passing protocol; Destination, an endpoint; Performative, message type; Payload, the content of message. Prova implements a new inference extension called literal guards. During the unification only if a guard condition evaluates to true, the target rule will proceed with further evaluation. We implemented the *sparql_select* built-in[7] to run SPARQL queries from Prova which can start a SPARQL query from inside Prova on an RDF file or a SPARQL endpoint. This buit-in can use results which come from the SPARQL query and use them inside Prova. It also provides the possibility to replace variables in SPARQL string which are starting with $ with variables.

In our experiments we use the Prova rule engine. We use the *sparql_select* built-in: the rule engine first sends the embedded SPARQL query to triple store, gets the results back and then waits for incoming event stream to process. It processes the sequence of

[4] Prova, ISO Prolog syntax with extensions `http://prova.ws` , July 2011

[5] Prova Reactive Messaging `http://www.prova.ws/confluence/display/RM/Reactive+messaging` , July 2011

[6] Event Processing Using Reaction Groups `http://www.prova.ws/confluence/display/EP/Event+processing+using+reaction+groups`, July 2011

[7] Source codes for Semantic Web extensions in Prova 3 can be found in `https://mandarax.svn.sourceforge.net/svnroot/mandarax/prova3/prova-compact/branches/prova3-sw/` , July 2011

events using the provided results from the knowledge base. In our experiments, we use a replicated version of part of DBpedia as an external knowledge base and the YAGO classification [3] for the classification of different music types and bands, e.g., a user can use the YAGO classification to express his music interest.

The complete pre-processing step should be updated on the knowledge base, whenever there is a change in the knowledge base, e.g., if new music bands are added to DBpedia our event query has no knowledge about them. In many use case like ours, the frequency of such updates can be considered to not be very high. Here, one useful approach is to implement the updates also in an event-based manner, if any relevant changes are done on the knowledge base a notification informs the event processing engine to update the event query.

Our experiments show clearly that the EQPP can achieve a better perofmance than the naïve storage-based approach (or pulling approach). They also show that the EQPP approach is an applicable approach for the above described use case. It shows also that the scalability of SCEP systems has five different dimensions; 1. Discharge rate of events 2. Number of rules in main memory 3. Number of triples in KB (amount of knowledge) 4. Rate of knowledge updates 5. Expressive level of reasoning on KB.

5 Conclusion and Outlook

We described our initial work on semantic event processing and semantic pre-processing of event queries, and illustrated the potential of this approach by use of a demonstration.

Our future steps are to work on semantics of event processing languages, and define which semantics can be adequate semantic for Complex Event Processing. Furthermore, we are working on an algorithm for rewriting of complex event queries to several simple queries which can be distributed on an event processing network to achieve high performance and scalability.

References

1. Adrian Paschke, Alexander Kozlenkov, and Harold Boley. A homogeneous reaction rule language for complex event processing. *CoRR*, abs/1008.0823, 2010.
2. Kay-Uwe Schmidt, Darko Anicic, and Roland Stühmer. Event-driven reactivity: A survey and requirements analysis. In *SBPM2008: 3rd international Workshop on Semantic Business Process Management in conjunction with the 5th European Semantic Web Conference (ESWC'08)*. CEUR Workshop Proceedings (CEUR-WS.org, ISSN 1613-0073), June 2008.
3. Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*, WWW '07, pages 697–706, New York, NY, USA, 2007. ACM.
4. Kia Teymourian and Adrian Paschke. Semantic rule-based complex event processing. In *RuleML 2009: Proceedings of the International RuleML Symposium on Rule Interchange and Applications*, 2009.
5. Kia Teymourian and Adrian Paschke. Towards semantic event processing. In *DEBS '09: Proceedings of the Third ACM International Conference on Distributed Event-Based Systems*, pages 1–2, New York, NY, USA, 2009. ACM.