Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI 2015)

Expressive Rule-Based Stream Reasoning*

(Extended Thesis Abstract / Proposal)

Harald Beck
Institute of Information Systems, Vienna University of Technology
Favoritenstraße 9-11, A-1040 Vienna, Austria
beck@kr.tuwien.ac.at

Abstract

Stream reasoning is the task of continuously deriving conclusions on streaming data. As a research theme, it is targeted by different communities which emphasize different aspects, e.g., throughput vs. expressiveness. This thesis aims to advance the theoretical foundations underlying diverse stream reasoning approaches and to convert obtained insights into a prototypical expressive rule-based reasoning system that is lacking to date.

1 Introduction

Stream reasoning [7] emerged from stream processing [2] to reason about information from data streams in real time and to provide users with live results for instant decision making.

As research theme, stream reasoning is emerging in different communities along different perspectives, and different aspects are emphasized with regard to data models, computational processing, data frequency, and crucially, the semantics of possible queries. Due to a lack of a common theoretical underpinning for various stream processing and reasoning approaches, the exact semantics of respective engines is often hard to predict and compare.

While stream/event *processing* and reactive programming are rapidly evolving, advanced *reasoning* over data streams is at a comparatively early stage. Rule-based systems (such as Answer Set Programming; ASP; see overview [6]) have proven to be fruitful for advanced reasoning on static data; however, only preliminary (or less expressive [1]) attempts exist to extend them to the streaming setting.

The aim of this thesis is twofold. First, the foundation of stream reasoning shall be advanced to obtain a rigorous means for its mathematical analysis. Second, based on theoretical insight and inspiration from existing stream processing engines, a prototypical rule-based reasoning engine shall be implemented that is expressive enough for problem solving in AI. To this end, the challenging trade-off between scalability and expressiveness must be tackled.

*This research has been supported by the Austrian Science Fund (FWF) project P26471.

2 Goals

This motivates further development of expressive rule-based stream reasoning in both theory and practice.

Theoretical Foundation for Stream Reasoning. The thesis shall provide a *theoretical foundation* for advanced stream reasoning, i.e., a *logical* formalism that provides a rigorous means for *comparison* of various existing semantics. A central idea to tackle the expressiveness/throughput trade-off is to make us of data *abstractions*: Changes of relevant high-level information (that is implicit but not appearing as such in the stream) might occur less frequently than the low-level input of the stream. For instance, a predicate 'cold' could hold throughout a July's afternoon in Buenos Aires, regardless of the exact temperature that is being updated continuously by real-valued sensor data. Rules are a natural choice for specifying queries or models that include such abstractions.

Thus, the language should be purely *declarative*, suitable for mathematical analysis, and provide *rule-based* reasoning for high expressiveness. The result should give an *analytic toolbox*, i.e., allow for (i) formal definitions of the semantics of a stream reasoning engine and (ii) studies of its properties, e.g., semantic behaviour, complexity, etc.

The envisaged theoretical underpinning for various existing approaches and formal results on top should aid developers in choosing suitable approaches for real-life applications.

Semantic Benchmarking. The obtained logic-based framework shall then be used to

- *formalize* the semantics of existing engines/languages that are only informally or operationally defined; and to
- *capture* the formally specified semantics of diverse approaches.
- This will allow to relate the actual output of engines with their *intended* output according to their *ideal semantics* (which can then be made precise), and to
- *compare* the differences in language semantics in a systematic, uniform way.

Expressiveness and Scalability. Simultaneous expressivity and performance for stream reasoning is highly challenging. A research goal is *to understand the trade-off for this combination and the limit for pushing both targets*, as regards theory (complexity analysis) and practice (experimental evaluation). Thus, the aim is to *formally represent various aspects and "ingredients"* of stream reasoning, such as

- different *window* mechanisms to discard outdated information,
- different modes of temporal reference (e.g. now, some time, always),
- relations of time points vs. time intervals,
- language features such as negation and recursion, or
- push- vs. pull-based querying;

and relate them to their *semantic implications* (e.g. model uniqueness) and their *computational cost*.

Algorithms and Implementation. Given the early stage of developments in advanced reasoning on streams in KR&R, a fully developed reasoning engine for high expressiveness is beyond the scope of this thesis. However, we aim to provide

- a *prototypical implementation* for rule-based stream reasoning that focuses on the correctness of the computed semantics (but presumes throughput limitations); and allows for a parametrization on
- *different levels of expressiveness*. The theoretical understanding of different fragments of a general, high-level language should be reflected in algorithms of varying efficiency.
- *Efficient update of previous results* is of particular interest in this regard.

In principle, one may use an existing reasoning engine for static data (such as Clingo [10]), restrict to queries that can be encoded in the respective language (such as ASP) and run the engine whenever data is coming in. However, computing derived conclusions repeatedly from scratch (instead of maintaining them) is very inefficient and will work in practice only for very low frequency/throughput. Accordingly, in the context of ASP, initial ideas for reactive [8] and incremental [9] computation have been proposed. However, no algorithms exist to efficiently update models for languages with flexible operators tailored for stream reasoning.

3 Research Issues

Formal Semantics for Stream Reasoning. Advanced reasoning systems use formal semantics in terms of well-defined mathematical structures such as models, as a basis to provide declarative means for reasoning. The model generation of technologies like SAT solving or ASP allows for computing alternative answers, which is important in scenario generation where branching into different possible futures is needed. We also need formal means to *precisely* describe properties that semantics have or miss in comparison. This will aid designing suitable pragmatic semantics.

Algorithms and Scalability. To push for scalability, also high-level reasoning on abstracted (i.e. less frequent) data needs efficient algorithms that cope with challenges such as overlapping input, service outages or high input rate. For this, advanced techniques from related fields (dynamic query plans, incremental evaluation, incremental reasoning [11], load shedding) will be considered. Furthermore, a *complexity analysis* of various reasoning tasks and fragments of the query language will give hints on the scalability border. Finally, an experimental evaluation will assess the effectiveness of the obtained methods and techniques.

Generic Measurements for Comparison. In order to compare stream-oriented semantics, one needs a suite of sufficiently *generic measurements and metrics* based on a theoretical framework (that did not exist prior to this thesis). Following theoretical results, stream processing/reasoning engines will be benchmarked and compared qualitatively.

4 Expected Outcomes

This leads to the following expected outcomes.

1. **Formal semantics for expressive rule-based stream reasoning**, as reference language for complex AI problems; now established by the LARS framework [5].
2. **Study and comparison of existing semantics** and their considered aspects. In particular, the limits of the prominent snapshot semantics (e.g. in [3]) should be clarified and contrasted with alternatives.
3. **Algorithms** for practical fragments of LARS and techniques for incremental model update; see a first result in [4].
4. **Experimental prototype** that allows for empirical evaluation of the proposed language and its considered fragments.

In summary, the thesis will yield *thorough theoretical results* on expressive stream reasoning for AI, with a focus on rule-based semantics, a *benchmarking framework* with analytical and practical tools to compare stream processing/reasoning approaches, and a *prototype implementation*.

References

[1] D. Anicic, P. Fodor, S. Rudolph, R. Stühmer, N. Stojanovic, and R. Studer. A rule-based language for complex event processing and reasoning. In *RR 2010*, pages 42–57, 2010.

[2] S. Babu and J. Widom. Continuous queries over data streams. *SIGMOD Record*, 3(30):109–120, 2001.

[3] D. F. Barbieri, D. Braga, S. Ceri, E. Della Valle, and M. Grossniklaus. Incremental reasoning on streams and rich background knowledge. In *ESWC (1)*, pages 1–15, 2010.

[4] H. Beck, M. Dao-Tran, and T. Eiter. Answer update for rule-based stream reasoning. In *IJCAI*, 2015.

[5] H. Beck, M. Dao-Tran, T. Eiter, and M. Fink. LARS: A Logic-based Framework for Analyzing Reasoning over Streams. In *AAAI*, 2015.

[6] G. Brewka, T. Eiter, and M. Truszczyński. Answer set programming at a glance. *Communications of the ACM*, 54(12):92–103, 2011.

[7] E. Della Valle, S. Ceri, F. van Harmelen, and D. Fensel. It's a streaming world! reasoning upon rapidly changing information. *IEEE Intelligent Systems*, 24:83–89, 2009.

[8] M. Gebser, T. Grote, R. Kaminski, and T. Schaub. Reactive answer set programming. In *LPNMR*, pages 54–66, 2011.

[9] M. Gebser, R. Kaminski, B. Kaufmann, M. Ostrowski, T. Schaub, and S. Thiele. Engineering an incremental ASP solver. In *ICLP*, pages 190–205, 2008.

[10] M. Gebser, R. Kaminski, B. Kaufmann, and T. Schaub. Clingo = asp + control: Extended report. Technical report, University of Potsdam, 2014.

[11] M. Gebser, O. Sabuncu, and T. Schaub. An incremental answer set programming based system for finite model computation. *AI Commun.*, 24(2):195–212, 2011.