On Representing Activity Context via Semantic Rule Methods (Summary of Invited Talk)

Benjamin N. Grosof
Benjamin Grosof & Associates, LLC
Mercer Island, WA 98040, USA
http://www.mit.edu/<the tilde character>bgrosof/

Abstract

We analyze several of the key technical and practical challenges involved in representing activity context across a large variety of knowledge, components, and applications. We present two novel broad methods that enable semantic knowledge capture and interchange, and suggest how they can be used for activity context-awareness. The first is knowledge representation and reasoning (KRR) in Rulelog, an expressively extended form of declarative logic programs that features defeasible higher-order logic formulas yet is computationally tractable, and is a draft dialect of W3C RIF. Rulelog's expressiveness enables representation of exceptions and change, and thus processes, agreements, and policies, e.g., for confidentiality. The second broad method is Textual Logic, an approach to mapping between natural language (text) and logic, where the mapping itself is logic-based. Textual Logic leverages Rulelog's expressiveness to enable relatively rapid text-based authoring of rich knowledge, reducing the knowledge acquisition bottleneck. Together, Rulelog and Textual Logic help address the potential for ontological and KRR Babel that lurks when representing activity context using previous semantic technologies.

Introduction and Requirements

Architecting activity context-aware systems requires representing context knowledge across large and heterogeneous variety of information, components, and applications. The Workshop's Call For Participation ably gives a list, but it can seem nigh overwhelming.

An immediately apparent key challenge is:

- Problem 1: *How is one to avoid Babel?*

I.e., how can one to achieve enough reusability of the knowledge and reasoning involved in order to make useful and sophisticated activity context-awareness become economically feasible?

Logical knowledge representation and reasoning (KRR) has advantages in regard to accuracy and explicability, as compared to techniques based on inductive (i.e., statistical) machine learning / data mining. The high-level vision of semantic technologies and semantic web, founded upon KRR, is attractive as a means of capturing and interchanging the context, insofar as it is largely focused on reusability via knowledge interchange. However, there is as yet a long way still to go in order to achieve that vision.

Some of the key underlying technical challenges include:

- How to acquire needed/useful logical knowledge (K) that is expressively rich? When specified manually, in past this has been quite costly. A key goal in the field of expressive knowledge representation and reasoning (KRR) is to reduce the cost of authoring rich logical knowledge.
- How to represent processes?
- How to represent policies and agreements? E.g., about confidentiality.
- What logical ontologies? E.g., that can be agreed upon that are widely useful yet sufficiently comprehensive?
- How to map between ontologies? E.g., that are developed by different organizations or communities?
- What KRR: what forms of K and what kinds of reasoning?
- How to interchange between multiple KRR's? E.g., between different semantic industry standards that are based on different fundamental logics? Those logics include classical logic (notably, first order logic), declarative logic programs (used in SQL, SPARQL, and many rule-based systems), and others.
- How to integrate deductive reasoning and inductive reasoning?

Particularly relevant for addressing those challenges are two novel broad methods we have recently developed for semantic rules, that enable semantic knowledge capture and interchange. The research on them was done largely in the Halo Advanced Research (HalAR) program (SILK 2013) (part of overall Project Halo) funded by Vulcan, Inc. These methods build on a longer line of R&D work that goes back more than 20 years: in declarative logic programs, web rules, semantic web services, and commercial rule systems.

In this presentation we give a sense of where the frontier is in regard to these semantic methods, and make some fairly high-level suggestions on several aspect of their potential applicability to activity context-awareness.

Rulelog

Rulelog is an expressive knowledge representation logic that is an extended form of declarative logic programs (LP). Rulelog transforms into normal (unextended) LP. Previous work on Rulelog, implemented in XSB (XSB 2013; Swift and Warren 2012), Flora-2 (Flora-2 2013), SILK (SILK 2013), and Cyc (Cyc 2013), has developed novel methods that help improve scale-able evolution and combination of such KB's, and thus the cost of overall knowledge acquisition (KA). These methods enable: defeasibility, based on argumentation theories (AT's) (Wan et al. 2009), i.e., *AT-defeasibility*; higher-order syntax, based on hilog (Chen, Kifer, and Warren 1993) and other meta-knowledge enabled by rule id's, i.e., *hidlog*; bounded rationality *restraint*; (Grosof and Swift 2013); interactive authoring, based on a rapid edit-test-inspect loop and incremental truth maintenance; knowledge debugging, based on a graphical integrated development environment with justification graphs and reasoning trace analysis; and knowledge interchange, based on strong semantics and knowledge translations. Rulelog's full set of major features was first implemented in SILK (SILK 2013). A W3C RIF dialect based on Rulelog is in draft (SILK 2013), in cooperation also with RuleML (RuleML 2013).

Defeasibility is needed to gracefully handle exceptions and change:

- to represent the empirical character of knowledge;
- to aid the evolution and combination of KB's, i.e., to be *socially scalable*; and
- to represent causal processes and "what-if's" (hypotheticals, e.g., counterfactual).

In other words, defeasibility is needed to represent *change in knowledge and change in the world*. Yet, despite this expressive richness, inferencing in the logic must be computationally scalable, and thus at least *tractable* (worst-case polynomial-time). SPARQL and SQL databases are tractable, for example.[1]

Rulelog is the first KRR to achieve the combination of defeasibility and tractability — along with higher-order formulas.

Rulelog Advantages for Policies, Processes, Agreements, and Ontology Mappings

Work from HalAR and previously, e.g., in the Semantic Web Services Framework (SWSF 2005), has established that representing policies (including about confidentiality), processes, and agreements/contracts requires gracefully handling exceptions and change, and therefore, the expressive feature of defeasibility.

That work in HalAR and previously, and other work too (e.g., in the RuleML Symposia) has also established that defeasible semantic rules KRR — to wit, Rulelog — is sufficient to represent well rich knowledge and reasoning about policies (e.g., in confidentiality, e-commerce, regulations, and law), processes (e.g., in college-level biology and e-commerce), and agreements/contracts (e.g., in e-commerce, systems management, and finance).

Mappings between ontologies require rules (OWL lacks enough expressiveness). Just as a dictionary's info is mostly about mappings between words, practical engineering that works with ontologies is continually needing to map ontologies in flexible ways. This requires the hilog feature. Rulelog combines hilog with its other features, and thus can richly represent both ontologies and mappings between ontologies.

Textual Logic

Next we present a more radical step that we have developed in order to further reduce the KA cost of rich logical K: a method that enables text-based authoring, based on a novel approach called *textual logic (TL)*.

Textual Logic (TL) overall is a logic-based approach to both text interpretation and text generation, for both KA and question answering (QA). In TL: text is mapped to logic; logic is mapped to text; and these mappings themselves are based on logic.

A novel aspect of TL is *textual terminology* — a phrasal style of knowledge. Words, and more generally word senses, are employed directly as logical constants. Each constant is a hilog functor (i.e., a function or predicate). A textual phrase corresponds (one-to-one) to a logical term; there is a natural style of composition.

Another novel aspect of TL is that it leverages defeasibility. "The thing about NL is that there's a gazillion special cases."[2].

During TL text interpretation, authors (1.) *articulate* sentences in text, then (2.) logically *disambiguate* those sentences, using a novel *rapid interactive disambiguation* technique, and (3.) *generate* logical axioms in Rulelog as output. These three steps are, in general, *interactive*, i.e., semi-automatic. Multiple authors may collaborate in these steps, for each sentence, including to divide the labor, edit, and review/comment/rate. Interactive disambiguation treats the parse, logical quantifiers and their scopes, co-reference, and word sense.

Rulelog's full expressiveness is needed in order to represent naturally arising text in college-level biology — the focal domain for HalAR — and many other domains.

We conducted a TL knowledge acquisition (KA) experiment during January-March 2013, that resulted in a case study in the rapid acquisition of rich logical knowledge from one chapter (on cell membranes) of a popular college-level biology textbook, with implications for biomedical education and research. A distributed team of knowledge engineers (KE's) started from effectively unconstrained natural language text and disambiguated various aspects of English sentences, semi-automatically translating text into defeasible higher-order logic formulas expressed in Rulelog, implemented in SILK. The interactive disambiguation was performed using the Linguist^{TM} tool from Automata, Inc. The

[1] i.e., for querying, when the number of distinct logical variables per query is bounded; this is often described in terms of data complexity being tractable

[2] Peter E. Clark, private communication

distributed team's workflow authored and curated the knowledge base from the text into several thousand Rulelog axioms targeting question answering by a Digital Aristotle as part of Vulcan Inc.'s Project Halo.

In this TL KA experiment, about 2,500 English encoding sentences were axiomatized. These included hundreds of questions.

A number of questions, some of them sophisticated, answered successfully using Rulelog inferencing (in SILK) on the axioms. However, due to resource limitations of the study, only relatively limited tests of question-answering (QA) were conducted. The focus of the experiment was on KA productivity, primarily, and KA coverage, secondarily.

Encoding sentence length averaged 10 words and ranged up to 25 words. One main defeasible axiom in Rulelog (SILK syntax) resulted from each sentence. On average, each such main axiom transformed into over 5 rules in normal (unextended) LP.

It took less than 10 minutes (of KE labor) on average per sentence to: author, disambiguate, formalize, review, and revise a sentence.

The resulting axioms were typically more sophisticated than what skilled KE's typically produce when directly authoring into logical syntax.

Textual Logic Advantages for Ontological Pivoting and User Interaction

TL enables user interaction to be in terms of NL: for capturing and presenting (e.g., explaining) knowledge that is relevant to activity context.

Text-based authoring and interaction is desirable for several reasons. Natural language (NL) — not logic — is the language of "business users", for KA and also for QA. NL is required for broad accessibility by the knowledgeable community of (potential) contributors, in science and many similar areas. In particular, NL is much more broadly accessible and familiar to subject matter experts (SMEs), as opposed to knowledge engineers (KEs) trained in logic. Examples of SME's include scientists, business process owners, executives, lawyers, doctors, educators, engineers, analysts, civil servants, merchants, soldiers, chefs, and members of many other occupations. NL is required also for ordinary end users, e.g., students, citizens, shoppers, salespersons, clerks, and patients — i.e., for the community of (potential) "consumers" of the knowledge in science and many similar areas. Even KE's usually find much easier to articulate and understand text than logic. Most of the world's knowledge is currently described in text, so working from text sources is crucial. Economic scalability of KA thus requires authoring to be text-based, rather than directly in strict logical syntax which requires KE skill.

TL also enables lightly restricted NL text, e.g., English, to be used as a common ontology and representation to pivot through when representing and interchanging activity context between multiple components, applications, etc.

Future Directions

It's early days still in developing and pursuing the approach of Textual Logic plus Rulelog, so lots of future work remains to do. One direction is tooling, e.g., to leverage inductive learning to aid disambiguation. Another direction is more KA experiments, e.g.: to push on QA; and to scale up.

A third direction is to try out the approach in various applications. In terms of system architecture, this usage context will often call for specialized UI and service interfaces from apps to TL. Rulelog KRR can make use of databases and other service resources in the apps-relevant environment.

A fourth direction is to develop fundamental methods for more powerful, but still scalable, integration of deductive and inductive (statistical) reasoning and knowledge.

Acknowledgments

The work presented here on Textual Logic and Rulelog was partly supported by Vulcan, Inc., as part of Halo Advanced Research (HalAR), a large project from fall 2007 to spring 2013, which the author led and which centered around the development of Rulelog and SILK. The work on Textual Logic was also supported in part by Automata, Inc. Much of this presentation describes work done by the author jointly with members of the HalAR team during and after HalAR. Thanks to all of them. Thanks also to the overall Project Halo team at Vulcan, Inc.

References

Chen, W.; Kifer, M.; and Warren, D. 1993. HiLog: A foundation for higher-order logic programming. *Journal of Logic Programming* 15(3):187–230.

Cyc. 2013. Cyc. http://www.cyc.com (project begun in approx. 1984).

Flora-2. 2013. Flora-2. http://flora.sourceforge.net (project begun in approx. 2000).

Grosof, B., and Swift, T. 2013. Radial Restraint: A Semantically Clean Approach to Bounded Rationality for Logic Programs. In *Proc. AAAI-13, the 27th AAAI Conf. on Artificial Intelligence*.

RuleML. 2013. Rule Markup and Modeling Initiative. http://www.ruleml.org (project begun in approx. 2000).

SILK. 2013. SILK: Semantic Inferencing on Large Knowledge. http://silk.semwebcentral.org (project begun in 2008).

Swift, T., and Warren, D. S. 2012. XSB: Extending Prolog with Tabled Logic Programming. *TPLP* 12:157–187.

SWSF. 2005. Semantic Web Services Framework. http://www.w3.org/Submission/SWSF/.

Wan, H.; Grosof, B.; Kifer, M.; Fodor, P.; and Liang, S. 2009. Logic Programming with Defaults and Argumentation Theories. In *Int'l Conference on Logic Programming*.

XSB. 2013. XSB. http://xsb.sourceforge.net (project begun in approx. 1993).