Adimen-SUMO: Reengineering an Ontology for First-Order Reasoning

Javier Álvez, Department of Languages and Information Systems, University of the Basque Country, Donostia, Gipuzkoa, Spain
Paqui Lucio, Department of Languages and Information Systems, University of the Basque Country, Donostia, Gipuzkoa, Spain
German Rigau, Department of Languages and Information Systems, University of the Basque Country, Donostia, Gipuzkoa, Spain

ABSTRACT

In this paper, the authors present Adimen-SUMO, an operational ontology to be used by first-order theorem provers in intelligent systems that require sophisticated reasoning capabilities (e.g., Natural Language Processing, Knowledge Engineering, Semantic Web infrastructure, etc.). Adimen-SUMO has been obtained by automatically translating around 88% of the original axioms of SUMO (Suggested Upper Merged Ontology). Their main interest is to present in a practical way the advantages of using first-order theorem provers during the design and development of first-order ontologies. First-order theorem provers are applied as inference engines for reengineering a large and complex ontology in order to allow for formal reasoning. In particular, the authors’ study focuses on providing first-order reasoning support to SUMO. During the process, they detect, explain and repair several important design flaws and problems of the SUMO axiomatization. As a by-product, they also provide general design decisions and good practices for creating operational first-order ontologies of any kind.

Keywords: Automated Reasoning, First-order Logic, Knowledge Engineering, Linked Open Data, Ontologies

INTRODUCTION

Recently, the Semantic Web community has become very interested in having practical inference engines for ontological reasoning (Chandrasekaran, Josephson, & Benjamins, 1999; Noy & McGuinness, 2001; Staab & Studer 2009; d’Aquín & Noy, 2012). A well-known necessary condition for enabling the automated treatment of knowledge – in particular, automated reasoning with ontologies – is that ontologies must be written in a formal language whose syntax and semantics are both formally defined. Automated reasoning uses mechanical procedures to obtain a large body of deducible knowledge that can be inferred from a compact modelling.
A significant feature of interest for formal ontologies is expressiveness. Since an ontology is a conceptualization of a certain domain of interest, the language should allow to express the properties that characterize that domain (Gruber, 2009). It is also well-known that the expressive power of the underlying logic jeopardizes the computational tractability and the inherent complexity of logical reasoning. Indeed, very expressive logics are undecidable or semi-decidable with high worst-case complexity. Beyond undecidability, a highly expressive logic language may be incomplete, i.e. there is no finite proof system that will prove all entailed sentences in the logic. As a consequence, the trade-off between expressiveness and reasoning efficiency (even decidability) is a key point for the design of formal ontologies.

Today, the family of Web Ontology Languages, including OWL-DL (Horrocks & Patel-Schneider, 2004), is the most common formal knowledge representation formalism, being accepted and standardized by the W3C (World Wide Web Consortium). OWL-DL is a powerful knowledge representation formalism with high computational efficiency and theoretically founded in description logics (DL), which can be embedded into fragments of first-order logic. Additionally, pure DLs are subsets of the guarded fragment (GF) of first-order logic, defined through the relativisation of quantifiers by atomic formulas. The guarded fragment was introduced in Andráska, Németi, and van Benthem (1998), where the authors prove that the satisfiability problem for GF is decidable. Indeed, OWL-DL reasoners, such as Pellet (Sirin, Parsia, Grau, Kalyanpur, & Katz, 2007) or Fact++ (Tsarkov & Horrocks, 2006), implement very efficient decision algorithms for OWL-DL theories (Motik, Shearer, & Horrocks, 2009). However, OWL-DL decidability is achieved at the price of losing expressiveness. Thus, state-of-the-art OWL-DL reasoners are unable to cope with more expressive ontologies such as Cyc (Matuszek, Cabral, Witbrock, & DeOliveira, 2006), DOLCE (Gangemi, Guarino, Masolo, Oltramari, & Schneider, 2002) or SUMO (Niles & Pease, 2001b).

Likewise, first-order logic (FOL) is a very well-known and expressive formalism, although reasoning in FOL is undecidable. Lately, impressive progress has been made in first-order (FO) automated reasoning. Every year, the CASC competition2 (Pelletier, Sutcliffe, & Suttner, 2002, Sutcliffe & Suttner, 2006) evaluates the performance of sound, fully automatic, classical FO automated theorem provers (ATP) on a bounded number of eligible problems, chosen from the TPTP Problem Library3 (Sutcliffe, 2009). As a result, there exists an important collection of FO theorem provers, such as Vampire (Riazanov & Voronkov, 2002) and E-Prover (Schulz, 2002).

Some of these FO theorem provers have been used for reasoning with expressive ontologies such as SUMO (Suggested Upper Merged Ontology).4 In particular, the authors of Horrocks and Voronkov (2006) give many interesting hints for adapting the existing general purpose FO theorem provers for checking its consistency. The authors also provide two examples of inconsistencies that were detected in SUMO. Additionally, in Pease and Sutcliffe (2007), the authors report some preliminary experimental results evaluating the query timeout for different options when translating SUMO into FOL.5 Evolved versions of the translation described in Pease and Sutcliffe (2007) can be found in the TPTP Library. In the sequel, we refer to this translation as TPTP-SUMO.

In this paper, we present Adimen-SUMO,6 an operational off-the-shelf FO version of SUMO. Our main interest is to present in a practical way the advantages of using FO theorem provers as inference engines for working with large and complex ontologies. In particular, we have concentrated our efforts on studying, revising and improving SUMO, although a similar approach could be applied to Cyc, DOLCE or any other expressive ontology. We decided to focus on SUMO since we are aware of its intrinsic problems, and also of the fact that its design and development have been harshly criticized in the community of ontology developers and users from the very beginning. For example, in Oberle et al. (2007) the authors describe some
Semantically Enhanced Business Process Modeling Notation
www.igi-global.com/chapter/semantically-enhanced-business-process-modeling/60065?camid=4v1a

OOPS! (OntOlogy Pitfall Scanner!): An On-line Tool for Ontology Evaluation
www.igi-global.com/article/oops-ontology-pitfall-scanner/116450?camid=4v1a