Web Semantics: Science, Services and Agents on the World Wide Web 12–13 (2012) 22–40

Contents lists available at SciVerse ScienceDirect

Web Semantics: Science, Services and Agents on the World Wide Web

journal homepage: http://www.elsevier.com/locate/websem

Context-dependent views to axioms and consequences of Semantic Web ontologies

Franz Baader [a], Martin Knechtel [b], Rafael Peñaloza [a,*]

[a] Theoretical Computer Science, TU Dresden, Germany
[b] SAP AG, SAP Research, Center Dresden, Germany

ARTICLE INFO

Article history:
Available online 2 December 2011

Keywords:
Access restrictions
Views
Contexts
Ontologies

ABSTRACT

The framework developed in this paper can deal with scenarios where selected sub-ontologies of a large ontology are offered as views to users, based on contexts like the access rights of a user, the trust level required by the application, or the level of detail requested by the user. Instead of materializing a large number of different sub-ontologies, we propose to keep just one ontology, but equip each axiom with a label from an appropriate context lattice. The different contexts of this ontology are then also expressed by elements of this lattice. For large-scale ontologies, certain consequences (like the subsumption hierarchy) are often pre-computed. Instead of pre-computing these consequences for every context, our approach computes just one label (called a boundary) for each consequence such that a comparison of the user label with the consequence label determines whether the consequence follows from the sub-ontology determined by the context. We describe different black-box approaches for computing boundaries, and present first experimental results that compare the efficiency of these approaches on large real-world ontologies. Black-box means that, rather than requiring modifications of existing reasoning procedures, these approaches can use such procedures directly as sub-procedures, which allows us to employ existing highly-optimized reasoners. Similar to designing ontologies, the process of assigning axiom labels is error-prone. For this reason, we also address the problem of how to repair the labelling of an ontology in case the knowledge engineer notices that the computed boundary of a consequence does not coincide with her intuition regarding in which context the consequence should or should not be visible.

1. Introduction

Description Logics (DL) [1] are a successful family of knowledge representation formalisms, which can be used to represent the conceptual knowledge of an application domain in a structured and formally well-understood way. They are employed in various application domains, such as natural language processing, conceptual modelling in databases, and configuration of technical systems, but their most notable success so far is the adoption of the DL-based language OWL as standard ontology language for the Semantic Web. From the DL point of view, an ontology is a finite set of axioms, which formalize our knowledge about the relevant concepts of the application domain. From this explicitly described knowledge, the reasoners implemented in DL systems can then derive implicit consequence. Application programs or human users interacting with the DL system thus have access not only to the explicitly represented knowledge, but also to its logical consequences. In order to provide fast access to the implicit knowledge, certain consequences (such as the subsumption hierarchy between named concepts) are often pre-computed by DLs systems.

In this paper, we investigate how this sort of pre-computation can be done in an efficient way in a setting where users can access only parts of an ontology, and should see only what follows from these parts. To be more precise, assume that you have a large ontology $\mathcal{O}$, but you want to offer different users different views on this ontology with respect to their context. In other words, each user can see only a subset of the large ontology, which is defined by the context she operates in. The context may be the level of expertise of the user, the access rights that she has been granted, or the level of detail that is deemed to be appropriate for the current setting, etc. More concretely, one could use context-dependent views for reducing information overload by providing only the information appropriate to the experience level of a user. For example, in a medical ontology we might want to offer one view for a patient that has only lay knowledge, one for a general practitioner, one for a cardiologist, one for a pulmonologist, etc. Another example is provided by proprietary commercial ontologies, where access is restricted according to a certain policy. The policy evaluates the context of each user by considering the assigned user roles, and then decides whether some axioms and the implicit

* Corresponding author.
E-mail addresses: baader@tcs.inf.tu-dresden.de (F. Baader), martin.knechtel@sap.com (M. Knechtel), penaloza@tcs.inf.tu-dresden.de (R. Peñaloza).

1570-8268/$ - see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.websem.2011.11.006

consequences that can be derived from them are available to this user or not.

One naïve approach towards dealing with such context-dependent views of ontologies would be to materialize a separate sub-ontology of the overall large ontology for each possible user context. However, this could potentially lead to an exponential number of ontologies having to be maintained, if we define one user context for each subset of the original ontology. This would imply that any update in the overall ontology needs to be propagated to each of the sub-ontologies, and any change in the context model, such as a new user role hierarchy or a new permission for a user role, may require removing or adding such subsets. Even worse, for each of these sub-ontologies, the relevant implicit consequences would need to be pre-computed and stored separately. To avoid these problems, we propose a different solution in this paper. The idea is to keep just the large ontology O, but assign "labels" to all axioms in the ontology and to all users in such a way that an appropriate comparison of the axiom label with the user label determines whether the axiom belongs to the sub-ontology for this user or not. This comparison will be computationally cheap and can be efficiently implemented with an index structure to look up all axioms with a given label. To be more precise, we use a set of labels L together with a partial order $\leqslant$ on L and assume that every axiom $\alpha \in O$ has an assigned label $\mathrm{lab}(\alpha) \in L$.[1] The labels $\ell \in L$ are also used to define user contexts (which can be interpreted as access rights, required level of granularity, etc.). The sub-ontology accessible for the context with label $\ell \in L$ is defined to be

$$O_{\geqslant \ell} := \{\alpha \in O | \mathrm{lab}(\alpha) \ \geqslant \ \ell\}.$$

Clearly, the user of a DL-based ontology is not only able to access its axioms, but also the consequences of these axioms. That is, a user whose context has label ℓ should also be allowed to see all the consequences of $O_{\geqslant \ell}$.

As mentioned already, certain consequences are usually pre-computed by DL systems in order to avoid expensive reasoning during the deployment phase of the ontology. For example, in the version of the large medical ontology SNOMED[2] that is distributed to hospitals and doctors, all the subsumption relationships between the concept names occurring in the ontology are pre-computed. For a labelled ontology as introduced above, pre-computing that a certain consequence c follows from the whole ontology O is not sufficient. In fact, a user whose context has label ℓ should only be able to see the consequences of $O_{\geqslant \ell}$, and since $O_{\geqslant \ell}$ may be smaller than O, the consequence c of O may not be a consequence of $O_{\geqslant \ell}$. As said above, pre-computing consequences for all possible user labels is not a good idea since then one might have to compute and store consequences for exponentially many different subsets of O. Our solution to this problem is to compute a so-called *boundary* for the consequence c, i.e., an element v of L such that c follows from $O_{\geqslant \ell}$ iff $\ell \leqslant v$. Thus, instead of pre-computing whether this consequence is valid for every possible sub-ontology, our approach computes just one label for each consequence such that a simple comparison of the context label with the consequence label determines whether the consequence follows from the corresponding sub-ontology or not.

There are two main approaches for computing a boundary. The *glass-box approach* takes a specific reasoner (or reasoning technique) for an ontology language and modifies it such that it can compute a boundary. Examples for the application of the glass-box approach to specific instances of the problem of computing a boundary are tableau-based approaches for reasoning in possibilistic Description Logics [2,3] (where the lattice is the interval $[0,1]$ with the usual order), glass-box approaches to axiom pinpointing in Description Logics [4–8] (where the lattice consists of (equivalence classes of) monotone Boolean formulae with implication as order [8]), and RDFS reasoning over labelled triples with modified inference rules for access control and provenance tracking [9,10]. The problem with glass-box approaches is that they have to be developed and implemented for every ontology language and reasoning approach anew and optimizations of the original reasoning approach do not always apply to the modified reasoners.

In contrast, the *black-box approach* can re-use existing optimized reasoners without modifications, and it can be applied to arbitrary ontology languages: one just needs to plug in a reasoner for this language. In this paper, we introduce three different black-box approaches for computing a boundary. The first approach uses an axiom pinpointing algorithm as black-box reasoner, whereas the second one modifies the Hitting-Set-Tree-based black-box approach to axiom pinpointing [11,12]. The third uses binary search and can only be applied if the context lattice is a linear order. It can be seen as a generalization of the black-box approach to reasoning in possibilistic Description Logics described in [13].

Of course, the boundary computation only yields the correct results if the axiom labels have been assigned in a correct way. Unfortunately, just like creating ontology axioms, appropriately equipping these axioms with context labels is an error-prone task. For instance, in an access control application, several axioms that in isolation may seem innocuous could, together, be used to derive a consequence that a certain user is not supposed to see. If the knowledge engineer detects that a consequence c has an inappropriate boundary, and thus allows access to the consequence by users that should not see it, then she may want to modify the axiom labelling in such a way that the boundary of c is updated to the desired label. This problem is very closely related to the problem of repairing an ontology. Indeed, to correct the boundary of a consequence, one needs to be able to detect the axioms that are responsible for it, since only their labels have an influence on this boundary. In a large-scale ontology, this task needs to be automated, as analysing hundreds of thousands of axioms by hand is not feasible.

To provide for such an automated label repair mechanism, we develop a black-box method for computing minimal sets of axioms that, when relabelled, yield the desired boundary for c; we call these *minimal change sets*. The main idea of this method is again based on the HST algorithms that have been developed for axiom-pinpointing. However, we show that the original labelling function can be exploited to decrease the search space. This algorithm can be used to output all minimal change sets. The knowledge engineer can then choose which of them to use for the relabelling, depending on different criteria. Unfortunately, just as in axiom-pinpointing, there may be exponentially many such minimal change sets, and thus analysing them all by hand may not be possible. We thus also develop an algorithm that computes only one change set having the smallest cardinality. This choice is motivated by a desire to make as few changes in the original labelled ontology as possible during the repair. We show that, in this case, a cardinality limit can be used to further optimize the algorithm.

All the algorithms described in this paper have been implemented and tested over large-scale ontologies from real-life applications, and using a context lattice motivated by an access control application scenario. Our experimental results show that our methods perform well in practice.

This paper extends and improves the results previously published in [14,15]. More precisely, the algorithms for computing the boundaries of consequences were presented in [14], while the problem of repairing the boundaries was addressed in [15]. Here, we (i) provide full proofs for all the theoretical results presented, (ii) present better optimizations to our algorithms, and (iii) provide a thorough comparison of the different algorithmic approaches through our experimental results. In order to make

[1] We will in fact impose the stronger restriction that $(L, \leqslant)$ defines a lattice (see Section 2).

[2] http://www.ihtsdo.org/snomed-ct/.