Towards Using OWL Integrity Constraints in Ontology Engineering*

Trung-Kien Tran and Christophe Debruyne

STARLab, Department of Computer Science, Vrije Universiteit Brussel
{truntran,chrdebru}@vub.ac.be

Abstract. In the GOPSL ontology engineering methodology, integrity constraints are used to guide communities in constraining their domain knowledge. This paper presents our investigation on OWL integrity constraints and its usage in ontology engineering.

Keywords: Ontology, Integrity Constraint, Ontology Engineering.

1 Introduction

The information contained in legacy systems can be exposed as RDF by annotating the system's symbols with concepts and relations in ontologies. Those mappings, however, sometimes disregard relational constraints [4]. In addition, annotated data also should meet some intended hypothesis in the ontologies. Therefore checking constraints against the annotated data-sources is important. This problem can be addressed by using OWL axioms as *integrity constraints* [5,9]. For example, the following constraint is common in cultural application domain: *every Event must has a Location where it takes place*. It can be expressed in OWL axiom (1).

$$Event \sqsubseteq \exists hasLocation.Location \tag{1}$$

Suppose that in our dataset $beerFestival$ is annotated as an instance of $Event$. In database setting, constraint (1) is violated. However, in OWL semantics, constraint (1) is not violated as it can be satisfied by unknown instances. The mismatch between Open World Assumption (OWA) in OWL and Closed World Assumption (CWA) in relational database needs to be addressed in order to use OWL axioms as integrity constraints. In this paper, we present a survey of prominent approaches, analyze the advantages and disadvantages, and an ongoing prototype validator.

2 OWL Integrity Constraints

In relational databases, *Integrity constraints* are used to validate the integrity of data. This idea is brought to knowledge representation and reasoning to enforce the legal state of a knowledge base [7,8]. We discuss some proposals here.

* This work was partially funded by the Brussels Institute for Research and Innovation through the Open Semantic Cloud for Brussels Project.

P. Herrero et al. (Eds.): OTM 2012 Workshops, LNCS 7567, pp. 282–285, 2012.

Definition 1 (Integrity constraints by consistency [3]). *A knowledge base $\mathcal{K}$ satisfies an integrity constraint IC iff $\mathcal{K} \cup IC$ is satisfiable.*

Example 1. Let IC_1 contains only axiom (1), $\mathcal{K}_1$ consists of the following axioms.

$$MusicEvent(jazzNight), MusicEvent \sqsubseteq Event \tag{2}$$

Obviously $\mathcal{K}_1 \cup IC_1$ is satisfiable, then $\mathcal{K}_1$ satisfies IC_1 by Definition 1. However, it does not fit our purpose: we want to have explicit location for $jazzNight$.

Definition 2 (Integrity constraints by entailment [7]). *A knowledge base $\mathcal{K}$ satisfies an integrity constraint IC iff $\mathcal{K} \models IC$.*

Example 2. Let IC_2 contains only axiom (1), $\mathcal{K}_2$ consists of (2) and (3).

$$hasLocation(jazzNight, grandPalace), Location(grandPalace) \tag{3}$$

Follow our intuition, $\mathcal{K}_2$ should satisfies IC_2. However, there is a model model $\mathcal{I}_1$ of $\mathcal{K}_2$ but $\mathcal{I}_1 \not\models IC_2$. By Definition 2, $\mathcal{K}_2$ does not satisfies IC_2.

$$\begin{aligned}\mathcal{I}_1 = \{&MusicEvent(jazzNight), hasLocation(jazzNight, grandPalace)\\ &Location(grandPalace), Event(jazzNight), Event(beerFestival)\}\end{aligned}$$

The above example suggests that entailment in Definition 2 should be restricted to *minimal models* of $\mathcal{K}$. This idea has been nicely captured in [5], where *outer skolemization* [6] is applied to deal with existential quantifiers.

Definition 3 (Integrity constraints by minimal models and skolemization [5]). *Let $\pi(\mathcal{K})$ and $\pi(IC)$ are first-order formulae that express $\mathcal{K}$ and IC respectively, and $\mathsf{sk}(\mathcal{K})$ is obtained from $\pi(\mathcal{K})$ by skolemization. $\mathcal{K}$ satisfies integrity constraint IC iff $\mathcal{I} \models \pi(IC)$ (simply written as $\mathcal{I} \models IC$) for every minimal (Herbrand) model $\mathcal{I}$ of $\mathsf{sk}(\mathcal{K})$.*

Reconsidering Example 1, the only minimal model of $\mathcal{K}_1$ is $\mathcal{I}_0 = \{Event(jazzNight), MusicEvent(jazzNight)\}$. By Definition 3, $\mathcal{K}_1$ does not satisfy IC_1 because $\mathcal{I}_0 \not\models IC_1$. In Example 2, the only minimal model of $\mathcal{K}_2$ is $\mathcal{I}_2 = \mathcal{I}_1 \setminus \{Event(beerFestival)\}$. $\mathcal{I}_2 \models IC_2$, then $\mathcal{K}_2$ satisfies IC_2. This matches our intuition, however *skolemization* could lead to unexpected consequences.

Example 3. Let $\mathcal{K}_3$ consists of axiom (2) and (4), IC_3 contains only the axiom (5).

$$Event \sqsubseteq \exists hasLocation.Location \tag{4}$$

$$MusicEvent \sqsubseteq \exists hasLocation.Location \tag{5}$$

We have $\mathcal{I}_3$ is the minimal Herbrand model of $\mathcal{K}_3$, where $unknownLocation$ is generated by skolemization of axiom (4).

$$\begin{aligned}\mathcal{I}_3 = &MusicEvent(jazzNight), hasLocation(jazzNight, unkownLocation)\\ &Event(jazzNight), Location(unknownLocation)\end{aligned}$$

Since $\mathcal{I}_3 \models IC_3$, $\mathcal{K}_3$ satisfies the IC_3 although $jazzNight$ has an unknown location.

To overcome this unexpected consequence, one can use a special concept O to bind the existential variables to known individuals. In addition, every known individual should be the instance of O. An alternative for the semantics of OWL integrity constraints has been proposed in [9]. Given a knowledge base $\mathcal{K}$ and a set of its models $\mathcal{U} = \{\mathcal{I}_1, \ldots, \mathcal{I}_n\}$, a concept name A, role name R and individual a are interpreted under *IC-interpretation* $\mathcal{I}_\mathcal{U}$ as follows:

$$\begin{aligned} A^{\mathcal{I}_\mathcal{U}} &= \{d^\mathcal{I} \mid d \in N_I \text{ such that } d^\mathcal{J} \in A^\mathcal{J}, \text{for all } \mathcal{J} \in \mathcal{U}\} \\ R^{\mathcal{I}_\mathcal{U}} &= \{(c^\mathcal{I}, d^\mathcal{I}) \mid c, d \in N_I \text{ such that } (c^\mathcal{J}, d^\mathcal{J}) \in R^\mathcal{J}, \text{for all } \mathcal{J} \in \mathcal{U}\} \\ a^{\mathcal{I}_\mathcal{U}} &= a^\mathcal{I} \end{aligned}$$

In addition to the new semantics, this approach utilizes the notion of *Weak Unique Name Assumption* (WUNA), that is: in the model of $\mathcal{K}$, *equalities* are derived as less as possible. The resulting model is called the *minimal equality model*. The integrity constraint satisfaction is then defined in Definition 4.

Definition 4 (Integrity constraints by WUNA and *IC-interpretation* [9]). *A knowledge base $\mathcal{K}$ satisfies integrity constraint IC iff for every axiom $\alpha \in IC$ and for all model $\mathcal{I} \in \mathcal{U}$, $\mathcal{I}_\mathcal{U} \models \alpha$; where $\mathcal{U}$ is a set of minimal equality models of $\mathcal{K}$ and $\mathcal{I}_\mathcal{U}$ is an IC-interpretation.*

Reconsidering previous examples, it is easy to check that Definition 4 matches our intuition. We agree with the proposed semantics of integrity constraints. However, we claim that UNA is more suitable. The reason is that *equality* can be seen as a special role and should be interpreted likes other roles in the integrity constraints. UNA, therefore, correctly captures semantics of *equality* under IC-interpretations.

3 Using OWL Integrity Constraints in Ontology Engineering

We will briefly describe the ontology-engineering method adopted within this project; GOSPL [1]. It is a hybrid approach as a special linguistic artifact is used to describe terms by means of natural language definitions for human reasoning on one side, and formal descriptions based on the other. For the formal part, the world is described by means of binary fact-types, which are generalizations of facts observed in the world. For instance, the observed fact "Christophe attends U2 360 tour" is refined into "the person with name "Christophe" attends the concert with concert title "U2 360" and then generalized into the following binary fact-types: (i) Person with / of Name, (ii) Concert with / of Concert Title, and (iii) Person attends / attended by Concert. Fact-types hold within a particular community, and the community is used as a means to disambiguate the labels inside those facts.

Commitments in GOSPL are a selection of lexon that are constrained to describe the intended usage of these facts. Commitments also contain information on how one individual application commits to the ontology[1]. The constraints are based on ORM [2].

The adoption of GOSPL within OSCB is motivated by the fact that the fact-types are grounded in natural language and that everything is described in terms of fact-types (no separate notions of concepts and properties), leveraging the modeling activities for domain experts.

[1] For more information on this part, we refer the reader to [1].

4 Implementation and Conclusion

We have implemented a prototype validator that can check three types of constraint: *mandatory constraint*, *uniqueness constraint*, and *key constraint* [2]. Table 1 shows the SPARQL queries (without prefixes) to check the validity of those integrity constraints and get the counter examples. To get complete answers, we use HermiT[2] reasoner to perform *forward reasoning* before running those SPARQL queries.

Table 1. SPARQL queries for integrity constraints

Constraint type / OWL integrity constraint	SPARQL query
Mandatory constraint / $C \sqsubseteq \exists R.D$	`SELECT ?x WHERE {?x rdf:type C. OPTIONAL {?x R ?y. ?y rdf:type D. } FILTER (!BOUND(?y))}`
Uniqueness constraint / $C \sqsubseteq \leq 1R.D$	`SELECT ?x WHERE {?x rdf:type C; R ?y1; R ?y2. ?y1 rdf:type D. ?y2 rdf:type D. FILTER (?y1 !=?y2)}`
Key constraint / $haskey(C, R)$	`SELECT ?x1 ?x2 WHERE {?x1 rdf:type C; R ?y. ?x2 rdf:type C; R ?y. FILTER (?x1!=?x2)}`

Our prototype validator is under development. Although we have not performed experiments on well-known benchmarks, our validator is tested with some toy ontologies. We plan to support more constraint types (cfr. ORM [2]), do further experiments, and fully integrate it to our ontology engineering platform.

References

1. Debruyne, C., Meersman, R.: Semantic Interoperation of Information Systems by Evolving Ontologies through Formalized Social Processes. In: Eder, J., Bielikova, M., Tjoa, A.M. (eds.) ADBIS 2011. LNCS, vol. 6909, pp. 444–459. Springer, Heidelberg (2011)
2. Halpin, T.: Information Modeling and Relational Databases. Morgan Kaufmann, San Francisco (2008)
3. Kowalski, R.A.: Logic for data description. In: Logic and Data Bases, pp. 77–103 (1977)
4. Lausen, G., Meier, M., Schmidt, M.: Sparqling constraints for rdf. In: EDBT, pp. 499–509 (2008)
5. Motik, B., Horrocks, I., Sattler, U.: Bridging the gap between owl and relational databases. J. Web Sem. 7(2), 74–89 (2009)
6. Nonnengart, A., Weidenbach, C.: Computing small clause normal forms. In: Handbook of Automated Reasoning, pp. 335–367 (2001)
7. Reiter, R.: Towards a logical reconstruction of relational database theory. In: On Conceptual Modelling (Intervale), pp. 191–233 (1982)
8. Reiter, R.: On integrity constraints. In: TARK, pp. 97–111. Morgan Kaufmann (1988)
9. Tao, J., Sirin, E., Bao, J., McGuinness, D.L.: Integrity constraints in owl. In: AAAI. AAAI Press (2010)

[2] `http://www.hermit-reasoner.com`