Translating FOL-theories into SROIQ-Tboxes

Fatima Danash, Danielle Ziébelin

Translating FOL-theories into SROIQ-Tboxes

*

Fatima Danash
Université Grenoble Alpes
Laboratoire d'Informatique de Grenoble
Grenoble, France
fatme.danash@univ-grenoble-alpes.fr

Danielle Ziebelin
Université Grenoble Alpes
Laboratoire d'Informatique de Grenoble
Grenoble, France
danielle.ziebelin@univ-grenoble-alpes.fr

ABSTRACT

Logical languages provide rigid formalisms for theories with varying expressive and scalable powers. In ontology engineering, it is popular to to provide a two-folded formalization of a theory; an expressive FOL formalization, and a decidable $\mathcal{SROIQ}$ fragment. Such a task requires a systematic and principled translation of the set of FOL formulas to achieve a maximally expressive decidable fragment. While no principled work exists for providing guidelines for the translation of FOL theories into $\mathcal{SROIQ}$ knowledge bases, this paper contributes with such a translation procedure.

CCS CONCEPTS

• **Computing methodologies** → **Knowledge representation and reasoning**; *Description logics*; Ontology engineering;

KEYWORDS

First-order Logic, Description Logics, Translations

ACM Reference Format:
Fatima Danash and Danielle Ziebelin. 2023. Translating FOL-theories into SROIQ-Tboxes: . In *Proceedings of ACM SAC Conference (SAC'23).* ACM, New York, NY, USA, Article 4, 4 pages. https://doi.org/10.1145/3555776.3577870

1 INTRODUCTION

Logical languages provide rigid formalisms for theories, standards, and knowledge domains. Several languages have been designed with varying expressive powers and scalable complexities. For the Semantic Web [12], which is the next step in the evolution of the World Wide Web, the goal is to have a standard formal representation that is expressive enough to model any knowledge domain, yet decidable enough to be read, understood, and compiled by machines. Thus, the trade-off arises between the expressivity and the decidability properties of logical languages. The Web Ontology Language [11] (OWL), a World Wide Web Consortium (W3C) recommendation language for the SW, achieves a balance between these requirements as based on the $\mathcal{SHOIN}$ [8] logical language.

$\mathcal{SHOIN}$ is a logic from the family called Description Logics [2] (DLs). DLs form a decidable fragment of the expressive First-Order Logic [13] (FOL), thus, a compromise between expressivity and scalability. As a semantic web standardization, the latest version of OWL; OWL2DL [1] is based on the $\mathcal{SROIQ}$ [6] logic which is the most expressive (yet decidable) commonly used DL.

In ontology engineering, it is becoming more popular to offer a two-folded formalization of a knowledge domain; a FOL-formalization as an initial *reference ontology* capturing the domain of interest with a wide expressive power, and a DL-formalization as a secondary *lightweight ontology* [5] supporting its implementation and application in the semantic web. The passage from the former (FOL-formalization) to the latter (DL-formalization) is a highly critical task that allows for the transition from expressive theories to building semantic web applications for the world wide web. To fulfill this task, a translation is required to guide a systematic and principled rewriting of the set of formulas present in the theory. The investigation on the extant literature concerning the transition from FOL to $\mathcal{SROIQ}$ (e.g. [10] and [7]) reveals a noticeable effort made on adding rule fragments to DLs within different integration manners, but no principled work or even tools providing guidelines for the translation of FOL theories into $\mathcal{SROIQ}$ knowledge bases.

In this paper, we build a translation procedure that takes as input an FOL-theory and gives as an output a $\mathcal{SROIQ}$ knowledge base. The procedure consists of six consecutive steps, in which a particular logical formalism/serialization of the initial theory is computed in each. Some operations call for syntactic/semantic checks to qualify a subset of the inputted set. Other steps allow for choice making when options arise.

2 TRANSLATION PROCEDURE

In this section, we illustrate the translation procedure's steps as shown in figure 1. The procedure takes as input a set of axioms S corresponding to a theory T formalized in FOL. Each step of the 5 steps comprises an input set S_i, where $0 \leqslant i \leqslant 4$, resembling the theory's serialization in a specific logical formalism e.g. S_1 is the set of axioms in clausal form CF, the operation(s) to be performed on each axiom in S_i, and an output set S_{i+1} which forms the input of the next step. The final output of the translation corresponds to a $\mathcal{SROIQ}$ serialization as a liter version (decidable fragment) of the initial inputted FOL serialization.

2.1 Transforming to Clausal Form (CF)

▹ Input: S_0 set of n formulas in first-order logic. For each axiom, we apply the following four operations.

*This is a short version of the article, please contact the authors to access the full extended version document.

SAC'23, March 27 –31, 2023, Tallinn, Estonia

ACM ISBN 978-1-4503-9517-5/23/03…$15.00
https://doi.org/10.1145/3555776.3577870

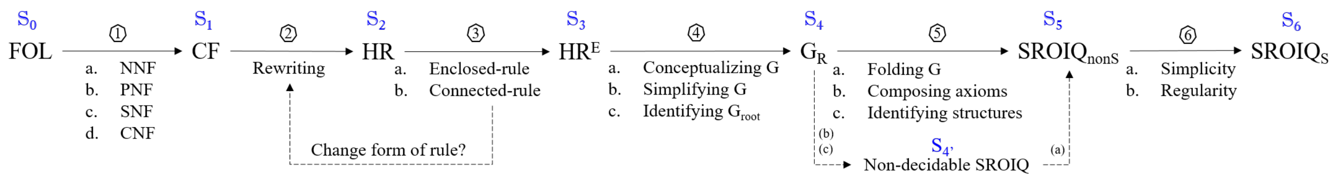

Figure 1: The different steps of the translation procedure and their corresponding operations, starting from a set S_0 in first-order logic, and resulting in structured set S_6 of $\mathcal{SROIQ}$ axioms.

2.1.1 Negation Normal Form (NNF). Simplify the formula, using logical equivalences, and move negations inwards so that it contains connectives of the form $\forall$, $\exists$, $\wedge$, $\vee$, and $\neg$ only.

2.1.2 Prenex Normal Form (PNF). Demarcate the variables of the formula, by renaming different variables that have same notation, and move all the quantifiers to the left.

2.1.3 Skolem Normal Form (SNF). Remove all existential quantifiers by replacing an existentially quantified variable e.g. x by a *Skolem constant* if x is not preceded by universally quantified variables, and by a *Skolem function* in terms of the universally quantified variables that precede x otherwise.

2.1.4 Clausal Normal Form (CNF). Distribute disjunctions over conjunctions so that the formula becomes a set of conjunctions of disjunctions.

All four operations formulate a sequence of transformations which guarantees that the final form of a, as a set of clauses, is the simplest and best form for the rewriting of a as (a group of) rule(s) as required by step 2.2. The resulting set of axioms S_1 is not equivalent to S_0, but a sub-set of S_0 ($S_1 \leq S_0$) where $S_1 \models S_0$, but the converse is not true.

2.2 Rewriting as Horn Rules (HR)

▹ Input: S_1 set of m clauses in clausal form. For each clause C_i in S_1, let n and m be the numbers of non-negated and negated atoms respectively. Let A, B, and C be atoms of C_i resembling unary/binary predicates.

- if $n \leqslant 1$, then C_i is a horn clause.
  - if $n = 1$, then C_i which is a positive horn clause of the form $\neg A \vee \neg B \vee C$ is rewritten into $[\neg(A \wedge B) \vee C] \equiv [A \wedge B \longrightarrow C]$.
  - if $n = 0$, then C_i which is a negative horn clause of the form $\neg A \vee \neg B \vee \neg C$ can be rewritten into one of the following m favorable options e.g. $[\neg(A \wedge B) \vee \neg C] \equiv [A \wedge B \longrightarrow \neg C]$, in comparison to any of the n options e.g. $[\neg(A \wedge B \wedge C)] \equiv [A \wedge B \wedge C \longrightarrow \emptyset]$.
- if $n > 1$, then C_i is not a horn clause, and has the form $\neg A \vee B \vee C$. However, it is still possible to force the clause to be rewritten in the horn implication form by qualifying any of the n atoms for the rule's head e.g. $[\neg(A \wedge \neg B) \vee C] \equiv [A \wedge \neg B \longrightarrow C]$ as possible favorable options.

This step is critical in establishing an implication form of C_i preparing the formula to have the form of *inclusion axioms*. The resulting set of horn rules S_{HR} of all axioms is equivalent to the inputted set S_1 ($S_H \equiv S_1$).

2.3 Qualifying Expressible rules (HR^E)

▹ Input: S_2 (= $S_H R$) set of m horn rules in their implication form. Each rule R_i ($a_1 \wedge a_2 \wedge .. \wedge a_n \longrightarrow h$ in S_2) shall satisfy the two syntactic restrictions below to be qualified for the upcoming steps of the translation.

2.3.1 Enclosed-rule constraint. Restricts the variables in the head of the rule h to be present in at least one of the body's atoms a_i for $1 \leqslant i \leqslant n$ i.e. enclosed-variables.

2.3.2 Connected-rule constraint. Assures that for each pair of variables x and y in r_i, there exists a sequence $z_1, z_2, ..., z_n$ such that $z_1 = x$, $z_n = y$, and for $1 \leqslant j \leqslant n$ there is a binary predicate R in r_i such that $R(z_i, z_{i+1})$ or $R^-(z_{i+1}, z_i)$, i.e. connected-variables.

These constraints ensure the passage from the initial open-world assumption of FOL, to the intended closed-world assumption of DL. Since a variable that is not enclosed or not connected results in an open-world view i.e. could be bound to any predicate, which yields to an inexpressible rule. The resulting set of expressible horn rules S_{HR^E} is a sub-set of the inputted set S_2 ($S_H \leq S_2$).

2.4 Constructing the rule graph (G_R)

▹ Input: S_3 (= S_{HR^E}) set of *expressible* horn rules in their implication form. For each horn rule R_i apply the following steps to construct the graph G_{R_i}. Note that for the remaining part of the paper, we refer to the unary and binary predicates of R_i as concepts and roles respectively.

2.4.1 Conceptualizing G. Conceptualize the rule R_i as a directed labeled graph G_{R_i}, following [4], defined as $G = \langle V, E, L, H \rangle$ where;

- V is a finite set of the variables of R_i resembling the vertices of the graph;
- E is a finite set of the roles in the body of R_i resembling the edges of the graph, such that a role $S(x, y)$ is added to E in the form of an edge S_{xy};
- L is a finite set of label sets corresponding to each variable in R_i of the form $L = \{L_{x1} = \{C_1, C_2\}, L_{x2} = \{C_2, C_3\}, ..., L_{xn} = \{C_1, C_3\}\}$ where $L_{x1}, L_{x2}, ..., andL_{xn}$ are the label sets of the variables $x1, x2, ..., andxn$ respectively, C_1, C_2, C_3 are concepts in R_i which the variables satisfy, and L resembles the labels of all the vertices of the graph;
- H is the head of R_i written in the form of an assertion; either a concept assertion of the form $x : C$ if the rule is concept-headed i.e. $C(x)$, or a role assertion of the form $x, y : R$ if the rule is role-headed i.e. $R(x, y)$.

2.4.2 Simplifying G. Simplify the roles and concepts in E and L by removing edges and labels that do not alter the satisfiability of the rule. Such edges/labels correspond to roles/concepts that subsume other roles/concepts in E/L i.e. are entailed (implied) by other edges/labels within E/L, thus they do not further constraint G. We simplify $G = \langle V, E, L, H \rangle$ where $V = \{x, y, z\}$ according to the rules:

Simplifying roles: For each role in E, apply the following rules in order until no rule is applicable anymore. Let $\mathcal{R}$ be the role hierarchy of the roles in E.

(1) *if* (a) $\mathcal{R}$ contains a *RIA* of the form $R^1 \circ R^2 \circ ... \circ R^n \sqsubseteq S$, and (b) E contains the roles R^1_{x1x2}, R^2_{x2x3}, ..., R^n_{xnxn+1}, and S_{x1xn+1}, or their inverses, *then* remove S_{x1xn+1} from E.
(2) *if* (a) $\mathcal{R}$ contains a *RIA* of the form $R \circ R \sqsubseteq R$ i.e. $Tra(R)$, and (b) E contains the roles R_{xy}, R_{yz}, and R_{xz}, or their inverses, *then* remove R_{xz} from E.
(3) *if* (a) $\mathcal{R}$ contains a *RIA* of the form $S \sqsubseteq R^-$, and (b) E contains the roles R_{xy} and S_{yx}, *then* remove S_{yx} or R_{xy} from E.
(4) *if* (a) $\mathcal{R}$ contains a *RIA* of the form $S \sqsubseteq R$, and (b) E contains the roles R_{xy}, and S_{xy}, and (c) $R \neq S$, *then* remove R_{xy}.

Simplifying concepts: For each variable labels' set $L_i = C_j for 1 \leqslant j \leqslant n$ in L,apply the following rules until no rule is applicable anymore. Let $\mathcal{T}$ be the concepts hierarchy consisting of *general concept inclusion axioms* (GCIs) of the concepts in L.

(1) *if* $\bigcap_{C_j \in L_i} C_j \equiv \top$ in $\mathcal{T}$, *then* empty L_i i.e. $L_i = \emptyset$.
(2) *if* $C_j \sqsubseteq C_{j'}$ in $\mathcal{T}$, *then* remove $C_{j'}$ i.e. $L_i = L_i - \{C_{j'}\}$.

2.4.3 Identifying the root G_{root}. Identify the root of G which is a (set of) variable(s) depending on the form of the rule's head H. If the rule is concept-headed i.e. $H = x : C$, then the root is a single variable expressed $G_{root} = \{x\}$. If the rule is role-headed i.e. $H = x, y : R$, then the root is a path of variables starting by x in G to y in G and encompassing all the vertices in between x and y, expressed $G_{root} = \{x, .., y\}$.

Constructing a conceptualization of a rule as a graph helps to visualize the links between the variables of the rule and the overall shape of the rule (e.g. a tree or a cycle), and make the necessary operations (simplification and identification of root). Roles' simplification maximizes the possibility that G satisfies the semantic restrictions required later. Concepts' simplification minimizes the label's set of each vertex by removing subsumers and maintaining (a list) of the *most specific concepts.* And last, identifying the root of G is the key to the next step's conversion. The resulting set of rule graphs S_G is equivalent to the inputted set S_3 ($S_G \equiv S_3$).

2.5 Converting into *SROIQ* axioms

▹ Input: S_4 (= S_G) set of rules graphs in their simplified form. For each rule graph G perform the subsequent operations to convert it to a $\mathcal{SROIQ}$ inclusion axiom A, and prepare next step's inputs I and $\mathcal{R}_{NS}$ resembling the proposition builders and the set of non-simple roles in S_4, respectively.

2.5.1 Folding G. Qualify tree rule graphs only, i.e. G must not contain cycles when considered as an undirected graph, and shrink G to have V equivalent to G_{root}. This operation is referred to as folding since the vertices of G that are not in the G_{root} set, are folded back into a neighbor vertex [4]. The folding uses the *rolling-up technique* [14] which allows tree-like structures to be expressed as concept expressions. Thus, for each vertex z of G that is a leaf node and does not appear in G_{root}, we fold z into a neighbor vertex y by using the edge between z and y i.e. R_{zy}. This is done by rolling-up z into y as follows;

(1) eliminating R_{zy} from E; by rolling the edge into a concept expression and adding it to y's set of labels L_y:
  - if $L_z \neq \emptyset$, then $L_y = L_y \cup \exists R. \bigcap_{C_i \in L_z} C_i$
  - if $L_z = \emptyset$, then $L_y = L_y \cup \exists R.\top$
(2) eliminating z from V; $V = V - \{z\}$

2.5.2 Composing axioms. Compose the axiom(s) A from G as *general concept inclusion axioms* or *role inclusion axioms* based on G.

- if G is concept-headed i.e. it has the form $G = \langle \{x\}, \emptyset, L_x, x : C_{root} \rangle$, then G is converted into a single axiom A of the form $\bigcap_{C_i \in L_x} C_i \sqsubseteq C_{root}$ i.e. a *general concept inclusion axioms.*
- if G is role-headed i.e. it has the form $G = \langle \{x_1, ..., x_n\}, \{R_{x1,x2}, R_{x2,x3}, ..., R_{xn-1,xn}\}, \{L_x 1, L_x 2, ..., L_x n\}, x_1, x_n : R_{root} \rangle$, then G must not contain concepts but only roles to be converted to role insertion axiom(s). Thus we apply the following;
  (1) for every vertex x_i in V, rewrite its label $L_x i$ as a role expression using a fresh role, and fresh concept. Let C' be a fresh concept that is the intersection of all x's labels as $C' = \bigcap_{C_i \in L_x} C_i$, and RC' be an auxiliary property associated to C' as $C' \equiv \exists RC'.SELF$. Thus, each instance of C' will have the role RC' with itself, and the existence of such a loop implies that the individual upon which RC' loops over is an instance of C'. This results in emptying V, and G becoming a set of edges only in which there is a single path between x_1 and x_n.
  (2) convert the list of roles in E into a *role inclusion axiom* $\Omega \sqsubseteq R_{root}$, where Ω is the concatenation of the roles in E in the form of $\Omega = S_{x1,x2} \circ ... \circ S_{xn-1,xn}$.

2.5.3 Identifying structures. Verify the syntax of *role inclusion axioms* to be compliant with one of the forms restricted by the simplicity and regularity constraint of $\mathcal{SROIQ}$, without achieving decidability (yet), and to form the inputs of the next final step. Considering *RIAs* having the form $(a_j) : \Omega \sqsubseteq R_{root}$, we apply the following;

- restrict Ω to satisfy one of the following forms; (i) $R \circ R$; or (ii) R^-; or (iii) $S_1 \circ S_2 \circ .. \circ S_n$; or (iv) $R \circ S_1 \circ .. \circ S_n$; or (v) $S_1 \circ .. \circ S_n \circ R$; or (vi) S and S is simple.
- if Ω is a role composition, then;
  (1) for every S_i in Ω, that is not an inverse role and different from R_{root}; we define a "proposition builder" $\mathbb{I}_j$ to represent that the axiom a_j having index j is included in the TBox, and a_j holds some ordering relations between S_i and R_{root} as follows; $\mathbb{I}_j \rightarrow (S_i \prec R_{root}) \wedge (S_i^- \prec R_{root}) \wedge ..$ for all $1 \leqslant i \leqslant n$, following [3]. Each proposition builder $\mathbb{I}_j$ is added to the set of proposition builders I.
  (2) add R_{root} to the list of non-simple roles $\mathcal{R}_{NS}$ as follows; $\mathcal{R}_{NS} = \mathcal{R}_{NS} \cup \{R_{root}\}$.

The resulting set of axioms $S_{SROIQ(nonS)}$ is a sub-set of the inputted set S_4 ($S_{SROIQ(nonS)} \leq S_4$).

2.6 Establishing decidability - Generalization

▹ Inputs: S_5 ($= S_{SROIQ(nonS)}$) set of $\mathcal{SROIQ}$ axioms in their non-structured form, $\mathcal{I}$ set of proposition builders, and $\mathcal{R}_{NS}$ set of non-simple roles. In this final step, the goal is to structure the TBox by extracting a decidable fragment (structured) of the inputted set. This is done by applying two rules imposed by the two syntactic constraints of $\mathcal{SROIQ}$; simplicity and regularity [9]. These constraints target the theory as a whole rather than each single axiom per separately, to guarantee that the reasoning algorithms are correct and do terminate [9], and that the satisfiability problem is decidable [6]. Thus, in contrast to the preceding steps in which we have treated the inputted sets S_i as per element, at this step we deal with all the axioms of S_5 at once.

2.6.1 Simplicity rule. Track all the occurrences of non-simple roles in S_5 and drop axioms that violate decidability. For each role R_i in the set of non-simple role $\mathcal{R}_{NS}$, for each A_i in S_5, if A_i is of the form; (i) $\exists R_i.SELF$; or (ii) $< | = | > nR_i.C$; or (iii) $Irr(R)$; or (vi) $Asy(R)$; or (v) $Dis(R, S)$, then drop A_i, and $S_5' = S_5 - \{A_i\}$.

2.6.2 Regularity rule. Compute all the incompatibilities of different regular orders in $\mathcal{I}$ and track their corresponding axioms to be dropped. We follow the approach proposed in [3] for tracking incompatibilities caused by irregularities of contradicting partial orders over a role hierarchy. The approach builds a meta-theory $\mathcal{I}$ in propositional logic and defines the problem of finding subsets of S_5 (i.e. the TBox) that satisfy regularity as a decidable SAT problem. $\mathcal{I}$ is the set of proposition builders of the form $\mathbb{I}_j \rightarrow (S_i \prec R) \wedge (S_i^- \prec R)$ capturing the fact that the inclusion of axiom a_j in the TBox requires both orders $S_i \prec R$ and $S_i^- \prec R$ to hold in the role hierarchy. Using $\mathcal{I}$, we are able to track incompatibilities whenever there exists two different regular orders of roles, each indicating the inclusion in one another in a different direction. For example, the two proposition builders $\mathbb{I}_5 \rightarrow (S_1 \prec S_2) \wedge (S_1^- \prec S_2)$ and $\mathbb{I}_7 \rightarrow (S_2 \prec S_1) \wedge (S_2^- \prec S_1)$ signify that axioms a_5 and a_7 are incompatible with respect to each other. Such incompatibility is deduced by means of $\mathbb{I}_5$ and $\mathbb{I}_7$, and is represented in the meta-theory as a meta axiom **m1**:$\mathbb{I}_5 \rightarrow \neg\mathbb{I}_7$ indicating that for TBox to be decidable, one of the two axioms cancels the other i.e. TBox $- \{a_5\}$ or TBox $- \{a_7\}$.

After computing all the incompatibilities, we have the following:

- $S_5' = \{a_i\}$; the initial unstructured set of axioms inputted
- $\mathcal{M} = \{m_k\}$; the set of meta axioms from the meta level propositional theory
- $\mathcal{U} = \{a_i\}$; the set of unstructured axioms in the meta axioms of the set $\mathcal{M}$ i.e. those that cause incompatibilities e.g. a_5 and a_7.

To interpret the above, $S_5' - \mathcal{U}$ contains a set of structured (safe) axioms that comply with the syntactic restrictions. The goal is however to find the maximal structured set of axioms by finding the structured subsets of $\mathcal{U}$. The meta axioms in $\mathcal{U}$ can be linked depending on the axioms they tackle. For instance, if **m1** which captures $\mathbb{I}_5$ and $\mathbb{I}_7$, is the only meta axiom that captures these two proposition builders while no other meta axiom does, then **m1** is to be extracted from $\mathcal{M}$ into a sub-theory $\mathcal{M}_1$, and axioms a_5 and a_7 are to be removed from $\mathcal{U}$ into a tuple $\mathbb{U}_1 = \langle\{a_5\}, \{a_7\}\rangle$ indicating an ordered subset from $\mathcal{U}$ where one set of axioms is to be chosen to be included in the structured TBox (e.g. in our case either $\{a_5\}$ or $\{a_7\}$.

To generalize what preceded, we apply the following:

(1) divide the meta axioms in $\mathcal{M}$ into subsets of dependent meta axioms $\mathbb{M}_n$, where each set $\mathbb{M}_n$ contains m_k's that capture overlapping axioms a_i i.e. an $\mathbb{M}_n$ consists of m_k's overlapping over the roles that are tackled in their axioms.
(2) for each set $\mathbb{M}_n$, specify subsets of axioms within a tuple $\mathbb{U}_n$ upon which one should be excluded.

The result is a number of tuples $\mathbb{U}_n$ equivalent to that of the meta theory subsets $\mathbb{M}_n$, where each tuple presents choices of sets of axioms that, together, violate decidability, and one set exactly shall be chosen to be excluded from the TBox. Thus, for $S_6 = S_5' - \bigcup_n Choice[\mathbb{U}_n]$ i.e. a subset of S_5.

3 CONCLUSION

In this paper, we proposed a procedure for translating FOL theories into decidable $\mathcal{SROIQ}$ fragments. The procedure computes different equivalent/subset logical formalisms at each step by performing operations such as rewriting formulas, syntactic/semantic checks, graph transformations, and rule-rolling techniques. For the semantic web, developing ontological models from logical formalisms is widely growing nowadays. Our procedure facilitates the task of providing the DL-logical formalism, upon which an ontology implementation is based on, from the original FOL formalism, by following a systematic and principled translation.

REFERENCES

[1] 2012. *OWL 2 Web Ontology Language Document Overview (Second Edition).* W3C Recommendation. W3C. https://www.w3.org/TR/2012/REC-owl2-overview-20121211/.
[2] Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider. 2003. The Description Logic Handbook: Theory, Implementation, and Applications. *Kybernetes* 32 (2003).
[3] Alessander Benevides, Jean-Rémi Bourguet, Giancarlo Guizzardi, Rafael Peñaloza, and João Almeida. 2019. Representing a reference foundational ontology of events in SROIQ. *Applied Ontology* 14 (07 2019), 1–42. https://doi.org/10.3233/AO-190214
[4] Francis Gasse, Uli Sattler, and Volker Haarslev. 2008. Rewriting Rules into SROIQ Axioms.
[5] Giancarlo Guizzardi. 2006. On Ontology, ontologies, Conceptualizations, Modeling Languages, and (Meta)Models. 18–39.
[6] Ian Horrocks, Oliver Kutz, and Uli Sattler. 2006. The even more irresistible SROIQ. *The even More Irresistible SROIQ*, 57–67.
[7] Ian Horrocks and Peter F Patel-Schneider. 2004. A proposal for an OWL rules language. In *Proceedings of the 13th international conference on World Wide Web.* 723–731.
[8] Ian Horrocks, Ulrike Sattler, and Stephan Tobies. 1999. Practical reasoning for expressive description logics. In *International Conference on Logic for Programming Artificial Intelligence and Reasoning.* Springer, 161–180.
[9] Markus Krötzsch, Frantisek Simancik, and Ian Horrocks. 2012. A Description Logic Primer. *CoRR* abs/1201.4089 (2012). arXiv:1201.4089 http://arxiv.org/abs/1201.4089
[10] Riccardo Rosati. 2005. On the decidability and complexity of integrating ontologies and rules. *Journal of Web Semantics* 3, 1 (2005), 61–73.
[11] Guus Schreiber and Mike Dean. 2004. *OWL Web Ontology Language Reference.* W3C Recommendation. W3C. https://www.w3.org/TR/2004/REC-owl-ref-20040210/.
[12] N. Shadbolt, T. Berners-Lee, and W. Hall. 2006. The Semantic Web Revisited. *IEEE Intelligent Systems* 21, 3 (2006), 96–101. https://doi.org/10.1109/MIS.2006.62
[13] Raymond M Smullyan. 1995. *First-order logic.* Courier Corporation.
[14] Sergio Tessaris. 2001. *Questions and answers: reasoning and querying in Description Logic.* The University of Manchester (United Kingdom).