Model-Finding for Externally Verifying FOL Ontologies: A Study of Spatial Ontologies

Shirly STEPHEN and Torsten HAHMANN [1]
[a] *School of Computing and Information Sciences, University of Maine, USA*

Abstract. Use and reuse of an ontology requires prior ontology verification which encompasses, at least, proving that the ontology is internally consistent and consistent with representative datasets. First-order logic (FOL) model finders are among the only available tools to aid us in this undertaking, but proving consistency of FOL ontologies is theoretically intractable while also rarely succeeding in practice, with FOL model finders scaling even worse than FOL theorem provers. This issue is further exacerbated when verifying FOL ontologies against datasets, which requires constructing models with larger domain sizes.

This paper presents a first systematic study of the general feasibility of SAT-based model finding with FOL ontologies. We use select spatial ontologies and carefully controlled synthetic datasets to identify key measures that determine the size and difficulty of the resulting SAT problems. We experimentally show that these measures are closely correlated with the runtimes of Vampire and Paradox, two state-of-the-art model finders. We propose a definition elimination technique and demonstrate that it can be a highly effective measure for reducing the problem size and improving the runtime and scalability of model finding.

Keywords. ontology verification, first-order logic, satisfiability, model finding, definitions

1. Introduction

Recently, more and more first-order logic (FOL) ontologies have become available, ranging from upper ontologies such as DOLCE or GFO to domain ontologies for space, processes, or the geosciences. But using and reusing them requires extensive prior *evaluation* [1, 2], including, at the very least, verification of their logical consistency [3]. This includes (1) verifying an ontology's *internal consistency* that rules out contradictions between axioms by constructing *any* model, and (2) checking the ontology's *external consistency* with datasets that are representative of the ontology's intended domain or application by constructing a model from these datasets. While ontologies specified in OWL and other Description Logics (DL) are routinely verified both internally and externally even against large datasets (i.e., ABoxes) [4–6], FOL ontologies are currently verified only internally

[1]This material is based in part upon work supported by the National Science Foundation under Grant Numbers III-1565811 and OIA-1937099.

if at all. The reasons are manifold. Most importantly, the very few model-finding experiments with FOL ontologies that have been published in the literature are rather discouraging; model finders routinely time out. They have only been successful where very small models with fewer than 20 individuals exist. For example, Bos states that model finding "doesn't seem to scale up very well" [7] after using Paradox and Mace2 to construct models with up to 17 individuals for rather simple first-order logic theories, while Baumgartner et al. conclude that model finding "is a struggle with more than 20 individuals" [8]. Others [9,10] have also reported little success with FOL model finding though without elaborating further. Despite these experiences, we revisit the feasibility of model finding with FOL ontologies here for several reasons:

- Logical verification of FOL ontologies with and without data is key to the larger endeavor of developing and reliably (re)using FOL ontologies;
- Model-finding with ontologies has much broader utility for other reasoning tasks such as query answering, data cleaning, or identifying which ontologies or ontological assumptions are consistent with a dataset; and
- Prior results are 10 years or older with improvements in model finders and computing resources, especially working memory, since.

We specifically want to identify the key factors that limit model finding with FOL ontologies and potential approaches for improving its scalability.

To construct a model for a FOL ontology, traditional FOL model finders such as Paradox [11] or Vampire [12,13] translate the ontology into an equi-satisfiable Clausal Normal Form (CNF) and then instantiate those clauses for increasing domain sizes. This constructs a series of propositional satisfiability (SAT) problems on which standard SAT solving techniques are used to determine satisfiability (e.g., by constructing a model) or unsatisfiability, in which case the next domain size is tried. While SAT solvers capably handle large SAT problems, SAT-based model finders rarely succeed in constructing models for FOL ontologies. For this reason, we want to better understand what makes model finding with FOL ontologies – with or without data – so difficult in practice. We specifically want to quantify the size of the resulting SAT problems. To do that, we formalize the concept of a FOL ontology with data and define various size measures on its FOL-CNF and SAT conversions relative to the ontology's axioms and dataset. We show that not the number and length of sentences in the ontology's axiomatization, but the size of its signature and, especially, the number of binary predicates and predicates of higher arities (including functions, though we concentrate on predicates) are a critical source of the explosion in the SAT problem size. To address this issue, we introduce a technique – optional definition elimination (ODE) – for reducing the size of the signature. We theoretically quantify and empirically test the effectiveness of this technique on different sets of eliminated definitions using three ontologies. Because model finding results are highly susceptible to seemingly minuscule differences, we carefully control both the numbers of distinct objects (the domain size d) and the number of relational assertions for each predicate via constructing synthetic sample datasets for our experiments. Our results show that ODE can dramatically reduce the size of the resulting SAT problems and significantly speed up and scale up model finding in practice.

While we systematically study different sets of definitions for elimination and different sizes of ABoxes with a total of over 3,000 samples, we were quite restricted in the ontologies we could use for a number of reasons: (1) FOL ontologies typically only contain structural knowledge without datasets of their own (i.e. their ABox is empty); (2) because many FOL ontologies are used as reference ontologies, suitable data sources are not readily available; (3) many FOL ontologies are not even shared in computer-interpretable formats; and (4) FOL ontologies with defined predicates of arity greater 2 and for which any data is available are hard to come by. The chosen spatial ontologies CODI [14], RCC [15] and INCH [16][2] are ideally suited because all of them are available from COLORE at colore.oor.net, they all contain many defined binary predicates (though no predicates of higher arity), and we can easily extract suitable datasets from standard GIS databases using already implemented qualitative spatial operations.

2. Preliminaries

First-Order Logic Syntax and Semantics: A FOL ontology $\mathcal{O}$ is a set of sentences in FOL with equality[3]. Its nonlogical symbols, i.e., all constants, function symbols, and predicates, form its *signature* $\lambda(\mathcal{O})$. For simplicity, we consider here only ontologies with predicates and constants in their signature, because each n-ary function symbol can be encoded as a $n+1$-ary predicate symbol by adding axioms that capture its functional nature[4]. Each predicate symbol $\Omega \in \lambda(\mathcal{O})$ has an arity $a(\Omega) \geq 1$ and constants have arity 0. An *atom* is an expression of the form $\Omega(t_1, ..., t_n)$ where every t_i is a term – typically either a constant or a variable. A *literal* is an atom or its negation $\neg\Omega(t_1, ..., t_n)$. A *FOL formula* is constructed from atoms using the logical connectives $\wedge, \vee, \rightarrow, \leftrightarrow$ and $\neg$ and/or the quantifiers $\forall$ and $\exists$. A *FOL sentence* is a closed formula, that is, all variables are bound. A formula is *ground* if it does not contain any variables, that is, constants are the only terms therein. A FOL *clause* is a special kind of FOL sentence, namely a disjunction of a set of n literals, i.e. $L_1 \vee ... \vee L_n$.

An *interpretation* of an ontology $\mathcal{O}$ is a tuple $\mathcal{I} = \langle D, \Phi, \Psi \rangle$ over a non-empty domain D where Φ maps variables to individuals in D and Ψ maps nonlogical symbols in $\lambda(\mathcal{O})$ to individuals (for constants), sets (for unary predicates) and n-ary relations (for predicates of arity ≥ 2). An interpretation $\mathcal{I}$ under which all sentences in $\mathcal{O}$ are true (i.e., are *satisfied*) is called a *model*. An ontology is *consistent* (or *satisfiable*) if it has some model. Two ontologies are equi-satisfiable if they are both either satisfiable or unsatisfiable.

2.1. SAT-Based Model Finding with FOL Ontologies

The Mace-style finite-model building approach [11, 18, 19] used in popular ATPs (Paradox, Vampire, iProver) uses a two-stage process of, first, clausification and, second, propositional instantiation to convert a FOL ontology into a set of propositional clauses before handing them off to a SAT solver.

[2]The results for INCH are not further discussed here for space reasons, see [17] for details.

[3]We treat equality as a primitive logical predicate.

[4]Because constants typically represent objects from the domain of interest, we include them to allow specifying factual knowledge, i.e., data points.

Clausification: Through applying Skolem's algorithm (see, e.g. [20]) a FOL ontology is converted into an existential quantifier-free *clausal normal form* (CNF) – a set of FOL clauses. This process may introduce additional Skolem constants and functions. Thus, the resulting FOL-CNF ontology, which we refer to as $\mathcal{O}_{\text{FOL-CNF}}$, is not necessarily logically equivalent but still equi-satisfiable to the original ontology $\mathcal{O}$. The size of the *signature* of $\mathcal{O}_{\text{FOL-CNF}}$ is determined as follows:

Definition 1. *Let $\mathcal{O}_{FOL\text{-}CNF}$ be an ontology's FOL-CNF representation. Then*

1. *$sf_{a=n}(\mathcal{O}_{FOL\text{-}CNF})$ denotes the set of n-ary Skolem functions introduced by skolemization. If treated as predicates[5], the set $sf_{a=n}(\mathcal{O}_{FOL\text{-}CNF})$ adds that many $(n+1)$-ary predicates to $\mathcal{O}_{FOL\text{-}CNF}$.*
2. *$\Omega_{a=n}(\mathcal{O}_{FOL\text{-}CNF}) = \{\Omega \in \lambda(\mathcal{O}) \mid a(\Omega) = n)\} \cup sf_{a=n-1}(\mathcal{O}_{FOL\text{-}CNF})$ defines the set of predicates of arity n, which includes the n-ary predicates from $\mathcal{O}$ as well as any newly introduced $(n-1)$-ary Skolem functions.*

The size of $\mathcal{O}_{\text{FOL-CNF}}$ itself is defined in terms of its number of clauses:

Definition 2. *Let $\mathcal{O}_{FOL\text{-}CNF}$ be an ontology's FOL-CNF representation treated as set of clauses. Then for any single clause $C \in \mathcal{O}_{FOL\text{-}CNF}$, the clause-width $w(C)$ is the number of FOL literals therein.*
The formula-width of $\mathcal{O}_{FOL\text{-}CNF}$ is the maximal clause-width of all clauses in $\mathcal{O}_{FOL\text{-}CNF}$, defined as $W(\mathcal{O}) = \max\{w(C) | C \in \mathcal{O}_{FOL\text{-}CNF}\}$.

Propositionalization of the FOL-CNF ontology: This second step involves instantiating all variables within the FOL-CNF clauses over all combinations of individuals from a fixed domain. This requires first fixing the domain size (i.e. the number of distinct individuals) via a set of inequalities [21]. If the domain size is not known in advance, the model finder starts with domain size 1 and incrementally increases it each time the search space is exhausted. If, for example, the smallest model has 8 individuals, then the model finder will prove 7 SAT instances to be unsatisfiable before finding an 8th one that is satisfiable.

Note that propositionalization instantiates every predicate of arity n with d^n propositional variables in $\mathcal{O}_{\text{CNF-d}}$. For example, each binary predicate leads to d^2 and each ternary predicate to d^3 propositional variables.

Lemma 1. *Let $\mathcal{O}_{CNF\text{-}d}$ be the propositional instantiation of $\mathcal{O}_{FOL\text{-}CNF}$, a FOL ontology in CNF form with maximal arity a^*, over a domain with d individuals. Then $\mathcal{O}_{CNF\text{-}d}$ has $P_v = \sum_{i=1}^{a^*} \left(d^i \cdot |\Omega_{a=i}|\right)$ propositional variables.*

Likewise, each clause in an FOL-CNF ontology is instantiated for every combination of its (implicitly universally quantified) variables, leading to the following number of propositional clauses P_c.

Lemma 2. *Let $\mathcal{O}_{FOL\text{-}CNF}$ be a FOL-CNF ontology where C_v denotes the subset of clauses with v distinct FOL variables per clause (we refer to v as the*

[5] We only treat n-ary functions here as (n+1)-ary predicate symbols in order to approximate their influence on the SAT problem size. Model finders typically treat them differently. Because of that, we do not take into account the additional axioms one would need to capture their functional nature when treated as predicates.

variable density), and v^* is the maximal number of variables in any clause in $\mathcal{O}_{FOL\text{-}CNF}$ (the maximal variable density). Then for a domain size d, $\mathcal{O}_{CNF\text{-}d}$ has $P_c = \sum_{i=0}^{v^*} \left(d^i \cdot |C_{v=i}|\right)$ *propositional clauses.*

Thus, the 'size' of the propositional instantiation $\mathcal{O}_{\text{CNF-d}}$ can be jointly described using P_c and P_v; their ratio $r = \frac{P_c}{P_v}$ describes its clause density. While our approach for calculating P_v and P_c is rather naive and only a worst-case measure, preprocessing techniques built into modern model finders are meant to reduce these numbers. Nevertheless, we will show in Section 5 that these calculated measures are closely correlated to the experimental runtimes of model finders.

3. SAT-based Model Finding for FOL Ontologies with Data

For simply proving the internal consistency of a FOL ontology, no data (i.e. *ground facts*) are needed. However, to prove that an ontology is consistent with a given dataset, we need to take the dataset's size into account when estimating the size of the resulting SAT problem. To investigate how the size of $\mathcal{O}_{\text{CNF-d}}$ changes with different amounts of data, we adapt the notions of Terminological Box (TBox), Relations Box (RBox), and Assertion Box (ABox) from Description Logic (DL) ontologies [22, 23]. The TBox and RBox capture axioms that constrain the interpretations of concepts (i.e., unary predicates) and roles (i.e., binary predicates), respectively. We will not distinguish between them, but draw the distinction between the TBox (for all terminological axioms) and the ABox, the latter of which captures assertions about individuals, i.e., ground statements about an individual being an instance of a particular concept or being related to another individual via a particular relation.

3.1. Assertion Box (ABox) and Terminology Box (TBox)

A FOL ontology can mix structural knowledge and assertions about individuals, even in a single sentence. Because clausification tends to separate those to some degree, we define an ontology's ABox in terms of its FOL-CNF version.

Definition 3. *Let $\mathcal{O}$ be an ontology with signature $\lambda(\mathcal{O})$ and let $\mathcal{O}_{FOL\text{-}CNF}$ be its corresponding set of FOL-CNF clauses. Then the FOL assertion box* $\text{ABox}(\mathcal{O})$ *is the subset of $\mathcal{O}$'s sentences that only yield ground clauses in $\mathcal{O}_{FOL\text{-}CNF}$ that do not use symbols outside $\lambda(\mathcal{O})$*[6]*.*

While an ABox may contain disjunctive knowledge – reflected in ground clauses with multiple literals – many clauses are so-called *unit clauses* consisting of only a single literal, which are *facts*. In our experiments, we limit the ABox to such unit clauses. For simplicity, we further require that the ABox itself, and not just its clausal conversion, is represented as a set of ground clauses. In other words, *the ABox is the dataset* we want to verify an ontology against.

Definition 4. *An* $\text{ABox}(\mathcal{O})$ *is called factual iff it contains only unit clauses.*

[6] Clauses that are ground but use newly introduced Skolem constants or functions are not considered part of the ABox as the Skolem symbols arise from existential quantifiers.

The spatial ontologies (CODI, RCC, and INCH) we use in our experiments rely – like many other ontologies – only on unary and binary predicates. If the ABox for such an ontology is factual, it consists of two kinds of assertions: *class assertions* (e.g., *ArealRegion*('*penobscotCounty*')) and *relational assertions* (e.g., *Inc*('*i95*','*penobscotCounty*')). A special kind of assertions we (and many model finders) add, are so-called *distinctness assertions* that ensure that distinct constants denote distinct individuals (e.g., '*i95*'≠'*penobscotCounty*').

A FOL ontology's TBox captures its structural, i.e., non-factual knowledge. It consists of all sentences that either yield non-ground clauses or that contain Skolem symbols after conversion to CNF-FOL:

Definition 5. *Let $\mathcal{O}$ be a FOL ontology and* ABox($\mathcal{O}$) *its ABox. Then its FOL terminology box is defined as* TBox($\mathcal{O}$) $= \mathcal{O} \setminus$ ABox($\mathcal{O}$).

For an ontology with a factual ABox, the TBox will not contain any ground clauses except possibly ones involving Skolem symbols.

3.2. The Size of the Resulting SAT Problem

The following example demonstrates the clausification and propositionalization of an FOL ontology and the number of propositional variables and propositional clauses that are created in the process.

Example 1. *Consider $\mathcal{O}_{RCC\text{-}s}$ as a small subset of RCC's FOL axiomatization that consists of one axiom and two definitions with signature $\lambda(\mathcal{O}) = \{C, P, PP\}$ denoting contact $C(x, y)$, parthood $P(x, y)$, and proper parthood $PP(x, y)$.*

$$(\sigma_C)\ C(x,y) \rightarrow C(y,x)$$
$$(\sigma_P)\ P(x,y) \leftrightarrow \forall z[C(z,x) \rightarrow C(z,y)]$$
$$(\sigma_{PP})\ PP(x,y) \leftrightarrow P(x,y) \wedge \neg P(y,x)$$

Clause 1	$\neg C(x,y) \vee C(y,x)$.	**Clause 2**	$PP(x,y) \vee \neg P(x,y) \vee P(y,x)$.
Clause 3	$P(x,y) \vee C(f(x,y),x)$.	**Clause 4**	$P(x,y) \vee \neg C(f(x,y),y)$.
Clause 5	$\neg PP(x,y) \vee P(x,y)$.	**Clause 6**	$\neg PP(x,y) \vee \neg P(y,x)$.
Clause 7	$\neg P(x,y) \vee \neg C(z,x) \vee C(z,y)$.		

Table 1. FOL-CNF clauses for TBox($\mathcal{O}_{RCC\text{-}s}$). Clauses are joined by conjunctions.

The FOL-CNF version of $\mathcal{O}_{RCC\text{-}s}$ (Table 1) contains 7 clauses with 4 nonlogical symbols, which in addition to the 3 predicates from $\mathcal{O}_{RCC\text{-}s}$ includes one binary Skolem function f which can be encoded as a ternary predicate. Propositionalization for domain size $d = 20$ yields the following number of propositional variables:

$$P_v = |\Omega_{a=2}| \cdot d^2 + |\Omega_{a=3}| \cdot d^3 = 3 \cdot 20^2 + 1 \cdot 20^3 = 9,200$$

Out of the 7 clauses, one clause has 3 FOL variables (clause 7) while the other six all have 2 FOL variables (FOL variables in different clauses are different for the purpose of propositionalization). For an ABox($\mathcal{O}$) *with exactly one relational assertion, namely PP('m','n'), exactly one ground clause is added. Then for domain size 20 the following number of propositional clauses are created:*

$$\begin{aligned} P_c &= |C_{(v=3)}| * d^3 + |C_{(v=2)}| * d^2 + |C_{(v=1)}| * d^1 + |C_{(v=0)}| * d^0 \\ &= 1 \cdot 20^3 + 6 \cdot 20^2 + 0 \cdot 20^1 + 1 \cdot 20^0 = 10,401 \end{aligned}$$

Thus the number of propositional variables in the SAT representation is largely dependent upon the number and arity of predicates – each predicate of arity a results in d^a propositional variables for domain size d. This number determines the search space for the propositional SAT problem, which consists (without using any heuristics) of 2^{P_v} possible interpretations. A simple ontology with b binary and u unary predicates (and no other non-logical symbols) then yields $(2^b)^{d^2} \cdot (2^u)^d$ interpretations, which is exponential in both the number of binary predicates – and more generally the number of predicates of highest arity – and the domain size d. While modern SAT solvers are able to deal with thousands of variables and tens of thousands of clauses [24] via highly effective strategies for pruning the search space, P_v and P_c quickly grow into the millions even for ontologies with modestly-sized signatures with only a handful of binary predicates. But this also suggests that improvements can be realized by reducing the total number of predicates, especially those of highest arity to construct larger and more realistic models. Definition elimination, as formalized in Section 4, can accomplish this when many predicates are defined. But we first look more closely at how the ABox impacts the size of the resulting SAT problem.

3.3. The Impact of the Ontology's ABox on the Size of the SAT Problem

The composition of ABoxes can vary widely: it may contain a handful or thousands of facts, and some predicates may be used much more than others. In the extreme case, many predicates may only rarely or not at all be used in an ABox. To study the impact of the ABox in a more systematic way, we need to carefully control its size and makeup. To do that, we introduce the idea of an *(r-d)ABox* where d is the domain size (i.e., the number of distinct individuals in the domain) and r is a factor that controls how many relational assertions the ABox contains for each of the ontology's predicates of arity greater than 1.

Definition 6. *Let $\mathcal{O}$ be an ontology and D a domain of individuals. ABox($\mathcal{O}$) is called an (r-d)ABox iff it only contains the following assertions:*

1. *For each $\Omega \in \lambda(\mathcal{O})$ with arity $a(\Omega) \geq 2$, ABox($\mathcal{O}$) contains exactly r ground positive assertions (i.e. of the form $\Omega(d_1, d_2, \dots)$) and exactly r ground negated assertions (i.e. of the form $\neg\Omega(d'_1, d'_2, \dots)$ where $d_i, d'_i \in D$;*
2. *ABox($\mathcal{O}$) contains at most one sentence of the format $\Omega(d)$ for each $d \in D$ where Ω is a unary predicate (i.e. $\Omega \in \lambda(\mathcal{O})$ with $a(\Omega) = 1$)[7];*
3. *Distinctness assertions of the form $d_i \neq d_j \in$ ABox($\mathcal{O}$) for each pair $(d_i, d_j) \in D \times D$ with $d_i \neq d_j$.*

Note that these ABoxes are stratified in the sense that all non-unary predicates are used equally many times. While this may rarely happen in practical datasets[8], it allows us to avoid introducing noise caused by unrelated differences between ABoxes. Building on Lemmas 1 and 2, the size of the SAT problem resulting from an ontology with an (r-d)ABox can be calculated as:

[7] This criteria captures the idea that each individual in the domain can be asserted to be a member of some class; but this restriction has a limited impact on the overall size of the resulting SAT problem, which will be dominated by the number of relational assertions.

[8] To estimate the size of SAT problems resulting from practical datasets, we could treat r as an maximal number of assertions that use one predicate. But as it turns out, r mostly influences the number of propositional clauses but rarely the number of propositional variables.

Lemma 3. *Let $\mathcal{O}$ be a FOL ontology with ABox($\mathcal{O}$) being an (r-d)ABox thereof. Then the resulting propositional SAT problem contains*

- $P_v = \sum_{i=1}^{a^*} d^i \cdot |\Omega_{a=i}| + r \cdot \sum_{i=1}^{a^*} d^i \cdot |sf_{A,a=i}|$ *propositional variables; and*
- $P_c = \sum_{i=0}^{v^*} d^i \cdot |C_{T,v=i}| + r \cdot \sum_{i=0}^{v^*} d^i \cdot |C_{A,v=i}|)$ *propositional clauses.*

where a^* denotes the maximum arity of predicates (including those resulting from Skolem functions) and v^* the maximal variable density in $\mathcal{O}_{FOL\text{-}CNF}$.

P_v and P_c are driven by the maximal arity of predicates (a^*) and the maximal variable density (v^*), respectively, while the ABox's contribution (the second term in each equation) is rather small for factual ABoxes without definition elimination: P_v will not change at all because ground unit clauses yield no Skolem functions, while P_c increases by the number of facts contained in the ABox. Even for an ABox with thousands of facts, this is negligible compared to the number of clauses generated from the TBox for growing domain sizes.

4. Definition Elimination

We will now formalize optional definition elimination as a technique that reduces an FOL ontology's signature size. The following example shows how it can potentially reduce the size of the resulting SAT problems.

Example 2. *We reuse the* TBox *from Example 1 with $\lambda = \{C, P, PP\}$. One binary Skolem function (analogous to a ternary predicate) is introduced by clausification. Propositionalization yields* $68,800$ *and* $531,200$ *propositional variables for domain sizes* 40 *and* 80*, respectively. Adding one binary predicate O (overlap), which is explicitly defined by* (σ_O) $O(x, y) \leftrightarrow \exists z[P(z, x) \wedge P(z, y)]$*, almost doubles the number of propositional variables to* $134,400$ *and* $1,049,600$ *for* $d = 40$ *and* 80*, respectively. This increase is largely attributable to an additional ternary predicate that captures the introduction of a binary Skolem function as the result of eliminating the existential quantifier in σ_O. While the number of FOL-CNF clauses increases from 7 to 10, one of the new clauses has a variable density of 3. This leads to an eight-fold increase in the number of propositional clauses from* $73,600$ *to* $640,000$ *(for* $d = 40$*) and from* $550,400$ *to* $5,120,000$ *(for* $d = 80$*).*

In this example the addition of just one binary predicate causes large increases in P_v and P_c. But because the added predicate O is explicitly defined, its removal does not change the ontology's satisfiability and O's interpretation can be easily constructed after finding a model. To formalize this approach, we first define optional definitions and the DBox as a maximal set of optional definitions that can be easily removed from an ontology. They are based on the notion of explicit definitions [25] as special types of TBox sentences:

Definition 7. *An explicit definition of an n-ary predicate $\Omega \in \lambda(\mathcal{O})$ in an ontology $\mathcal{O}$ is a sentence $\sigma \in \mathrm{TBox}(O)$ of the form*

$$\forall x_1, \ldots, x_n[\Omega(x_1, \ldots, x_n) \leftrightarrow \alpha(x_1, \ldots, x_n)]$$

wherein α is a formula with x_1 to x_n as only free variables and with $\lambda(\mathcal{O}) \setminus \Omega$ as the only nonlogical symbols. Then Ω is said to be explicitly defined in T.

Optional definitions are explicit definitions of predicates that are not used in other sentences of the ontology's TBox:

Definition 8. *An explicit definition* $\sigma \in \mathrm{TBox}(\mathcal{O})$ *of a symbol* $\Omega \in \lambda(\mathcal{O})$ *is an optional definition in* $\mathcal{O}$ *iff* Ω *does not appear in any sentence in* $\mathrm{TBox}(\mathcal{O}) \setminus \sigma$.

Note that after removing an optional definition from $\mathrm{TBox}(\mathcal{O})$, other previously non-optional explicit definitions can also become optional in $\mathrm{TBox}(\mathcal{O}) \setminus \sigma$.

Example 3. *In* $\mathcal{O}_{RCC\text{-}s}$*, initially* σ_{PP} *is an optional definition but* σ_P *is not because it is used in the definiens of* σ_{PP}*. After removing* σ_{PP}*,* σ_P *becomes optional and can also be removed.*

Because of this effect we recursively define larger *definition sets*, with the maximal one being referred to as the ontology's DBox:

Definition 9. *A definition set of an ontology* $\mathcal{O}$ *is defined recursively as:*

B. *The set of all optional definitions in* $\mathrm{TBox}(\mathcal{O})$ *forms a definition set;*
R. *For any definition set* D *of* $\mathcal{O}$ *and for any optional definition* σ *of* Ω *in* $\mathrm{TBox}(\mathcal{O}) \setminus D$*, the set* D' *defined as follows is a definition set:*
$D' = \{\varsigma' | \varsigma \in D \text{ and } \varsigma' = \varsigma[\Omega(x_1, \ldots, x_n)/\alpha(x_1, \ldots, x_n)]\} \cup \{\sigma\}$,
that is, D' *substitutes all occurrences of* Ω *in other definitions in* D *by* Ω*'s definiens from* σ *and adds* σ *as a new definition to the set.*

Definition 10. *For an ontology* $\mathcal{O}$, $\mathrm{DBox}(\mathcal{O})$ *is a definition set such that no optional definition exists in* $\mathrm{TBox}(\mathcal{O}) \setminus \mathrm{DBox}(\mathcal{O})$.
$\Omega \in \lambda(T)$ *is optionally defined in* $\mathcal{O}$ *iff* Ω *does not appear in* $\mathrm{TBox}(\mathcal{O}) \setminus \mathrm{DBox}(\mathcal{O})$.

To study how removing optionally defined predicates impacts the SAT representation, we also need to substitute the eliminated predicates in the ABox without changing the ontology's semantics. This is achieved by replacing assertions that use optionally defined predicates by *defined assertions*.

Definition 11. *Let* $\mathcal{O}$ *be an ontology and* D *some definition set of* $\mathcal{O}$.
Then $\mathbf{ABox_D}(\mathcal{O}) = \mathrm{ABox}(\mathcal{O})\big[\bigcup_{\sigma_i \in D}[\Omega_i(x_1, \ldots, x_n)/\alpha_i(x_1, \ldots, x_n)]\big]$.
Any sentence $\sigma \in \mathrm{ABox}_D(\mathcal{O})$ *with* $\sigma \notin \mathrm{ABox}(\mathcal{O})$ *is called a defined assertion.*

In other words, $\mathbf{ABox_D}(\mathcal{O})$ is $\mathcal{O}$'s ABox with all optionally defined predicates Ω_i from D (which typically would be the entire DBox of $\mathcal{O}$) substituted by their definiens α_i. Such an ABox with defined assertions may no longer only contain only ground unit clauses. Defined assertions may contain variables or Skolem functions introduced during the substitution. For example, a fact $O(\text{'i95'}, \text{'295w'})$ results in the defined assertion $\exists z[P(z, \text{'i95'}) \wedge P(z, \text{'295w'})]$ if O is substituted by its definiens from Example 2. But the substitution of facts from the ABox by defined assertions ensures maintaining (non-)satisfiability. This follows directly from the well-known relationship between explicit and implicit definability formalized by Beth's definability theorem [25]. The following theorem captures this; the straightforward proof is provided in [17, Sec. 5.3]:

Theorem 1. *Let* $\mathcal{O}$ *be a FOL ontology and* D *be a definition set of* $\mathcal{O}$*. Then there is a bijection between the models of* $\big(\mathrm{TBox}(\mathcal{O}) \setminus D\big) \cup \mathrm{ABox}_D(\mathcal{O})$ *and the models of* $\mathrm{TBox}(\mathcal{O}) \cup \mathrm{ABox}(\mathcal{O})$*, that is, every model of* $\big(\mathrm{TBox}(\mathcal{O}) \setminus D\big) \cup \mathrm{ABox}_D(\mathcal{O})$ *can be uniquely expanded into a model of* $\mathrm{TBox}(\mathcal{O}) \cup \mathrm{ABox}(\mathcal{O})$.

Again, such a bijection exists specifically where $D = \mathrm{DBox}(\mathcal{O})$, the maximal set of optional definitions that can be easily removed without altering the ontology's semantics. This idea forms the basis of our strategy for improving model finding because $\big(\mathrm{TBox}(\mathcal{O}) \setminus \mathrm{DBox}(\mathcal{O})\big) \cup \mathrm{ABox}_D(\mathcal{O})$ has a smaller signature than $\mathcal{O} \equiv \mathrm{TBox}(\mathcal{O}) \cup \mathrm{ABox}(\mathcal{O})$ but is equi-satisfiable.

The effect of ODE on the resulting SAT problem's size is illustrated next.

Example 4. *Consider the ontology $\mathcal{O}_{RCC\text{-}s}$ again with $\beta \equiv PP(\text{'m'},\text{'n'})$ as the only assertion in the ABox. Now applying ODE only on PP removes σ_{PP} from the TBox and substitutes all occurrences of PP in the ABox with its definiens $P(x,y) \wedge \neg P(y,x)$. β will become $\beta' \equiv P(\text{'m'}, \text{'n'}) \wedge \neg P(\text{'n'},\text{'m'})$. ODE reduces the ontology's signature from three to two binary predicates. For a domain size 20, this reduces the number of propositional variables from $3 * 20^2 + 1 * 20^3 = 9,200$ to $2 * 20^2 + 1 * 20^3 = 8,800$. Likewise, the number of propositional clauses is slightly reduced from $10,400$ to $10,000$. A much larger decrease can be realized by eliminating syntactically more complex definitions, such as the definition σ_P of P, which contains an existential quantifier. σ_P is optionally defined after σ_{PP}'s removal. Its removal eliminates the ternary predicate, leading to a SAT problem with only $2 * 20^2 = 800$ propositional variables that arise from its TBox*[9].

It becomes clear that optional definition elimination is especially useful for ontologies that have many optionally defined predicates. This is the case for many spatial ontologies, such as the RCC or CODI, which we use next to study the benefit of ODE in more detail with respect to the estimated number of propositional variables and clauses and the actual change in runtimes of model finders.

5. Calculated and Experimental Results

Materials and Experimental Setup: We now present our quantitative and experimental results for two ontologies of qualitative space: CODI [14] and RCC [15]. For space reasons we do not discuss results from a third ontology, the INCH Calculus[10], which are included in [17]. Complete FOL axiomatization of CODI[11] and RCC[12] are available in the COLORE github repository. CODI and RCC each have a terminology of a total of 21/6 mostly binary predicates, with 8/5 of those being explicitly defined. The main reason for working with these ontologies is that their existing semantic alignment with the Simple Features Access (SFA) Model [26] – an FOL ontology that bridges qualitative spatial relations with standard geometric representations as used in GIS databases – allows easy extraction of datasets from standard GIS datasets.

For constructing (*r-d*)ABoxes (cf. Def. 6) we first extracted a *Master ABox* consisting of 425 spatial objects (i.e. individuals) related via 130,256 ground relational assertions (4,937 positive ones and 125,319 negated ones) that use the binary predicates (within, overlaps, intersects, crosses and touches) from the SFA standard. These map to the binary predicates used by CODI and RCC. To con-

[9] The overall number of propositional variables would be larger if the ABox heavily uses the eliminated predicate, as that would reintroduce some variables via Skolemization.

[10] http://colore.oor.net/inch/inch_calculus.clif

[11] http://colore.oor.net/multidim_space_codi/codi_basic.clif

[12] http://colore.oor.net/mereotopology/rcc_basic.clif

struct sample datasets from this Master ABox, we used a stratified sampling process to semi-randomly pick 5, 10, 15, or 20 relational assertions for each binary relation in the ontology's signature that relate between 20 to 50 individuals (to fix the domain size). We further added distinctness assertions (inequalities) for these individuals and, for CODI, class assertions for each unary predicate. This systematic construction of ABoxes helps attribute performance differences to the removal of specific predicates as all predicates are used with equal frequency. It avoids biasing or even hiding the effect of definition elimination that could result from real datasets, of which one expects a large variations in predicate use: some may be used much more than others while others potentially not at all.

We conducted our experiments with the state-of-the-art model finders Paradox [11] and Vampire [27][13] using 10 dataset samples for each ontology, signature, and combination of parameters, resulting in a total of 2,080 problems for CODI and 840 problems for RCC. To discount external effects and extreme outliers (since problems are created semi-randomly, some are extremely difficult), we only plot the *low mean* – the mean of all 10 samples that terminated before the mean plus one standard deviation ($\mu + \sigma$). Cases where the majority of samples did not terminate are indicated as such in our graphs.

5.1. Growth of the Number of Propositional Variables and Clauses

The dashed lines in Fig. 1 show how P_v differs across different sets of eliminated predicates. P_v increases polynomially with increasing d (cf. Lemma 3) while r has no impact for CODI and little impact on RCC except for case 1 there. RCC case 1 is special as it is the only case that introduces clauses with a variable density of 3 as shown in the RCC's ABox measures in Table 2. P_v decreases significantly when more and more defined predicates are eliminated. For example, the elimination of the 5 binary predicates SC, Inc, PO, PP and C from CODI (from case 13 to case 1) reduces P_v by roughly two-thirds even though 9 out of 14 binary predicates are preserved.

The number of propositional clauses P_c is exponential in the maximal variable density over all FOL-CNF clauses, polynomial in d, and increases linearly (though minimally) with increasing r. Overall, P_c grows in step with P_v. However, over-eager ODE can lead to (1) more FOL-CNF clauses, (2) much longer clauses (i.e. more clauses with a width ≥ 3), and (3) clauses with higher variable density. The latter two things happen when eliminating all defined predicates, including P, from RCC as shown by the ABox calculations for case 1 in Table 2, which results in the large increases in P_c and P_v shown in Fig. 1. This is caused by the nesting of optional definitions in RCC: EC is defined in terms of O and both PP and O are defined in terms of P. The construction of defined assertion then leads to multiple nested quantifiers, resulting in a high variable density after clausification.

5.2. Experimental Results

While Vampire is consistently faster than Paradox, the model finding times of both exhibit a similar pattern (especially for CODI) that is also closely correlated to the number of propositional variables. As more defined predicates are

[13] We also experimented with IProver but the runtimes are very inconsistent but for larger domain sizes always significantly greater than those for Vampire.

CODI		TBox					Basic ABox ($r = 1$)				
CODI Case	defined predicates included	$\Omega_{a=1}$, $\Omega_{a=2}$	$C_{T,v}$ with v FOL variables			C_T with $w \geq 3$	$C_{A,v}$ with v FOL variables			C_A with $w \geq 3$	$\Omega_{a=1}$
			v=3	v=2	v=1		v=2	v=1	v=0		
1	- (all cases include 22 other predicates)	13, 9	3	32	30	31	12	12	3	7	6
2	PP	13, 10	3	35	30	33	12	12	2	7	6
3	C	13, 11	4	34	30	33	12	9	4	7	5
4	C, PP	13, 12	4	37	30	35	12	9	4	7	5
5	PO	13, 11	4	34	30	33	12	9	5	7	5
6	PO, PP	13, 12	4	37	30	33	12	9	4	7	5
7	PO, PP, C	13, 14	5	39	30	35	12	6	6	7	4
8	Inc	13, 14	5	41	30	43	3	10	5	5	4
9	Inc, PP	13, 13	5	44	30	43	3	10	4	5	4
10	Inc, PP, C, PO	14, 16	7	48	30	45	3	4	8	5	2
11	SC	13, 12	8	34	30	36	9	8	5	2	4
12	SC, PP	13, 13	8	37	30	38	9	8	4	2	4
13	SC, PP, C, PO, Inc	13, 20	12	50	30	51	0	0	10	0	0

RCC		TBox					Basic ABox ($r = 1$)					
RCC Case	defined predicates included	$\Omega_{a=2}$	$C_{T,v}$ with v FOL variables			C_T with $w \geq 3$	$C_{A,v}$ with v FOL variables				C_A with $w \geq 3$	$\Omega_{a=1}, \Omega_{a=2}$
			v=3	v=2	v=1		v=3	v=2	v=1	v=0		
1	- (all cases include C)	1	0	1	1	0	4	16	20	5	32	7, 10
2	P	3	1	4	1	2	0	0	8	12	8	3, 2
3	P, PP	4	1	7	1	3	0	0	8	10	6	2, 0
4	P, O	5	2	6	1	3	0	0	1	14	5	0, 0
5	P, PP, O	6	2	9	1	4	0	0	1	12	1	0, 0
6	P, PP,O, EC	7	2	12	1	5	0	0	1	9	0	0, 0
7	P, PP, O, EC, NTTP	8	3	11	1	8	0	0	0	8	0	0, 0

Table 2. Each case represents version of CODI/RCC with a different set of its explicitly defined binary predicates included. Statistics for the resulting size of the FOL-CNF ontology for the TBox only (left) and an ABox with $r = 1$ (i.e. that includes exactly one positive and one negated assertion for each binary predicate, including the non-optional predicates).

eliminated and P_v decreases, the runtimes decrease as well[14]. The speed-up can be significant. For example for $d = 20$, case 1 contains less than half the number of propositional variables (3,860) than case 13 (8,260), which contains five additional binary predicates. This leads to a runtime deduction from 345s to 7s and from 713s to 134s for $r = 5$ and $r = 20$, respectively. This reduction is even more dramatic for $d = 30$, where the runtimes decrease from 30,000s (over 8h) to 16s and 164s for $r = 5$ and $r = 20$, respectively.

While there are slight differences about how well certain cases perform (e.g., cases 11 and 12 are more difficult for Paradox, whereas cases 8 to 10 are more difficult for Vampire), invariably CODI case 13 (without ODE) takes the longest for both solvers, with timeouts encountered consistently for $d = 30$ (Paradox) or $d = 50$ (Vampire). Case 1, which performs the most aggressive ODE, terminates fastest for Paradox throughout. However, the Vampire results show that removing as many definitions as possible does not always yield the best performance, in fact, case 2 which retains the definition of PP performs better. Overall, CODI demonstrates that ODE can improve model-finding scalability noticeably: In the best cases the model finders can construct models with up to 50 individuals in the same times previously needed for half that domain size.

By looking at the RCC results, an even more nuanced story emerges. While the runtimes mostly follow the trend of P_v, the steep increase in P_c and P_v in

[14]Beware that the cases are not ordered by P_v, and for CODI not even by the number of binary predicates preserved in the FOL-CNF ontology as shown column 3 of Table 2.

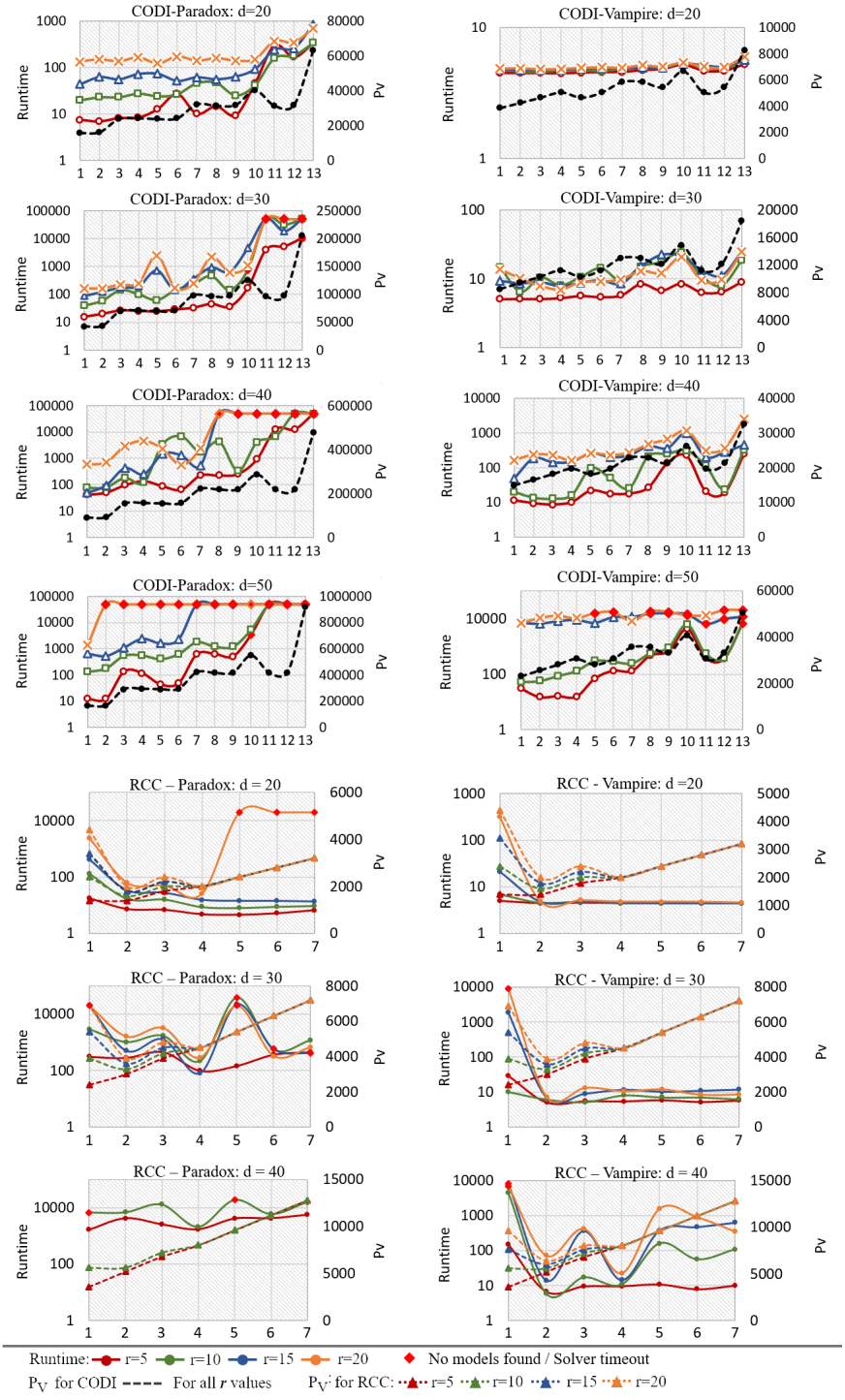

Figure 1. Low mean model finding times for CODI and RCC for different r-dABoxes using Paradox and Vampire. The numbered cases along the x-axis correspond to the case numbers in Table 2. The calculated number P_v (dashed) is plotted against the secondary axis.

case 1 yields a performance worse than without any ODE at all. P_v and P_c are the lowest in case 2, which removes all optionally defined predicates except for P. This case keeps both the number of newly introduced Skolem functions and the number of clauses with higher variable density relatively low. This is the best case for Vampire for both domain sizes 20 and 30. Paradox performs slightly better on case 4, which additionally retains O and results in even fewer clauses of highest variable density ($C_{A,1}$ is 1 compared to 8 as in cases 2 and 3).

In summary, we find that the quantitative measures of the FOL-CNF ontologies are good indicators for the expected runtimes of the employed SAT-based model finders. The results suggest that ODE is beneficial as long as the number of clauses with high variable density (i.e. with more variables) increases only marginally. Because such increases arise primarily from the introduction of quantifiers and Skolem functions in defined assertions, the optimal extent of ODE could be easily automatically determined during the clausification process.

6. Related Work

The elimination of either implicitly [28] or explicitly defined concepts [29] has been studied for expressive DL ontologies, but not for FOL ontologies. Verification of FOL ontologies has mostly focused on theorem proving [3, 30] as a means to find inconsistencies or to prove competency questions, while model finding has been dismissed as infeasible. Prior studies of automated reasoning over FOL ontologies, such as [9, 31, 32] have focused on the feasibility of theorem proving (as a means for query answering) rather than model finding. Those studies have shown that theorem proving scales poorly for large ontologies, but they also noted that the challenges for model finding are even greater. For example, in the study in [10] Paradox generated only models up to size 5 and Darwin timed out for most cases, while [9] also reports to have constructed only very small models. Likewise, no models for domain sizes larger than 17 were found by [7] using Paradox and Mace2 even for simple FOL theories. Their limited success let the authors in [8] conclude that model finding is "a struggle with more than 20 individuals". Our study pushes this limit to construct models with up to 120 individuals (for CODI) through the simple techniques of eliminating optional definitions.

While heuristics for improving SAT solving via various preprocessing techniques abound, most are only employed after clausification and propositionalization of an FOL axiomatization. Only few techniques (see [19] for a recent overview) have been successfully lifted to FOL model finders. For example, syntactic predicate elimination is performed on the CNF version of an ontology by Mace4 [18], but in our experiments, Mace4 did not even get close to the performance of Paradox or Vampire. Likewise, iProver eliminates some non-self-referential predicates [33] but behaved much more unpredictably than Paradox or Vampire and did not scale as well as Vampire.

7. Conclusion and Future Work

This work is only a first step in systematically studying the feasibility of model finding for FOL ontologies and in identifying avenues to improve FOL model finding in practice. We have shown that the size of an ontology's signature, es-

pecially the number of predicates of arity two or higher, has an outsized impact on the size of models that can be created, while the size of the dataset is much less critical. We have also shown that optional definition elimination (ODE) can effectively push the size of models that can be constructed. For example, ODE reduces the model finding time by over an order of magnitude for CODI, which in turn allowed us to construct models containing up to 120 individuals in the same amount of time previously needed to construct models with about 40 individual. This is a small but noticeable improvement over previous reports that indicated difficulties constructing any models with 20 or more individuals [7, 8]. Thus, our results are evidence that external verification of mid-sized FOL ontologies can be feasible in practice, at least for ontologies with only unary and binary predicates and wherein many terms are defined. However, as perhaps expected, it is also clear that model finding for FOL ontologies will likely never compete with the size of datasets that DL ontologies can be externally verified against.

Future Work It has become clear that ODE cannot be applied blindly, especially when dealing with ground facts that use the predicates slated for elimination. The example of RCC showed that eliminating the wrong predicate can significantly decrease model finding performance. However, simple measures, such as the number of quantified variables, the number and arity of introduced Skolem functions, and the number of ABox facts that use a specific predicate can help decide which predicates to eliminate. Future work needs to develop and study specific heuristics that employ such measures for automatically deciding and applying ODE as a preprocessing step for FOL model finding. More work is also necessary to study the effectiveness of ODE for predicates of higher arities and to test ODE on a more diverse set of FOL ontologies, including much larger ontologies.

Acknowledgments We thank the four anonymous reviewers for their sage feedback which greatly contributed to an improved final version of this paper.

References

[1] Vrandečić D. Ontology Evaluation. In: Handbook on Ontologies. 2nd ed. Springer; 2009. p. 293–313.

[2] Gómez-Pérez A. Ontology Evaluation. In: Handbook on Ontologies. 1st ed. Springer; 2004. p. 251–274.

[3] Grüninger M, Hahmann T, Hashemi A, Ong D. Ontology verification with repositories. In: Conf. on Formal Ontology in Inf. Systems (FOIS-10). IOS Press; 2010. p. 317–330.

[4] Xu J. Improving ABox reasoning for efficient and scalable object queries over large ontologies [PhD Thesis]. University of Miami; 2015.

[5] Dentler K, Cornet R, Ten Teije A, De Keizer N. Comparison of reasoners for large ontologies in the OWL2 EL profile. Semantic Web. 2011;2(2):71–87.

[6] Srinivas K. OWL Reasoning in the Real World: Searching for Godot. In: Description Logic Workshop (DL 2009), Jul 27-30, Oxford, UK. CEUR Vol. 477; 2009. p. 2:1–10.

[7] Bos J. Exploring model building for natural language understanding. In: Workshop on Inference in Computational Semantics, Sep 25-26, Nancy, France; 2003. p. 41–55.

[8] Baumgartner P, Fuchs A, De Nivelle H, Tinelli C. Computing finite models by reduction to function-free clause logic. Journal of Applied Logic. 2009;7(1):58–74.

[9] Pease A, Sutcliffe G. First Order Reasoning on a Large Ontology. In: CADE-21 ESARLT Workshop, Jul 17, Bremen, Germany. CEUR Vol. 257; 2007. p. 2:1–24.

[10] Schneider M, Sutcliffe G. Reasoning in the OWL2 Full ontology language using first-order automated theorem proving. In: Conf. on Automated Deduction (CADE-23), Jul 31-Aug 5, Wroclaw, Poland. Springer; 2011. p. 461–475.

[11] Claessen K, Sörensson N. New techniques that improve MACE-style finite model building. In: CADE-19 Workshop on Model Computation, Miami Beach, FL; 2003. p. 11–27.
[12] Kovács L, Voronkov A. First-order theorem proving and Vampire. In: Conf. on Computer Aided Verification (CAV 2013), Jul 13-19, St. Petersburg, Russia. Springer; 2013. p. 1–35.
[13] Biere A, Dragan I, Kovács L, Voronkov A. Experimenting with SAT solvers in Vampire. In: Mexican Intern. Conf. on Artif. Intell. (MICAI'14), Nov 16-22, Tuxtla Gutierrez, Mexico. Springer; 2014. p. 431–442.
[14] Hahmann T. CODI: A Multidimensional Theory of Mereotopology with Closure Operations. Applied Ontology. 2020;15(3):251–311.
[15] Randell DA, Cui Z, Cohn AG. A spatial logic based on regions and connection. In: Conf. on Principles of Knowledge Representation and Reasoning (KR'92), Oct 26-29, Cambridge, MA. Morgan Kaufmann; 1992. p. 165–176.
[16] Gotts NM. Formalizing commonsense topology: the INCH calculus. In: Symp. on Artif. Intell. and Math. (ISAIM 1996), Fort Lauderdale, FL; 1996. p. 72–75.
[17] Stephen S. Improving Model-Finding for Integrated Quantitative-Qualitative Spatial Reasoning with First-Order Logic Ontologies [PhD Thesis]. University of Maine; 2020.
[18] McCune W. Mace4 reference manual and guide. arXiv preprint cs/0310055. 2003.
[19] Janota M, Suda M. Towards Smarter MACE-style Model Finders. In: Conf. on Logic for Programming, Artificial Intelligence and Reasoning (LPAR-22), Nov 17-21, Awassa, Ethopia. EPiC Series in Computing 57; 2018. p. 454–470.
[20] Ben-Ari M. Mathematical logic for computer science. Springer; 2012.
[21] Zhang H, Zhang J. MACE4 and SEM: A Comparison of Finite Model Generators. In: Bonacina MP, Stickel ME, editors. Automated Reasoning and Mathematics: Essays in Memory of William W. McCune. Springer; 2013. p. 101–130.
[22] De Giacomo G, Lenzerini M. TBox and ABox reasoning in expressive description logics. In: Conf. on Principles of Knowledge Representation and Reasoning (KR'96), Nov 5-8, Cambridge, MA. Morgan Kaufmann; 1996. p. 316–327.
[23] Horrocks I, Sattler U, Tobies S. Reasoning with individuals for the description logic SHIQ. In: Conf. on Automated Deduction (CADE-17), Jun 17-20, Pittsburgh, PA,. Springer; 2000. p. 482–496.
[24] Gomes CP, Kautz H, Sabharwal A, Selman B. Satisfiability solvers. In: Handbook of Knowledge Representation. Elsevier; 2008. p. 89–134.
[25] Beth EW. On Padoa's method in the theory of definition. Indagationes Math. 1953;15:330–339.
[26] Stephen S, Hahmann T. Formal Qualitative Spatial Augmentation of the Simple Feature Access Model. In: Conf. on Spatial Information Theory (COSIT-19), Sep 9-13, Regensburg, Germany. LIPIcs Vol. 142; 2019. p. 15:1–14.
[27] Riazanov A, Voronkov A. The design and implementation of Vampire. AI Commun. 2002;15(2–3):91–110.
[28] Ten Cate B, Franconi E, Seylan I. Beth definability in expressive description logics. Journal of Artificial Intelligence Research. 2013;48:347–414.
[29] Wang K, Wang Z, Topor R, Pan JZ, Antoniou G. Eliminating Concepts and Roles from Ontologies in Expressive Description Logics. Comput Intell. 2014;30(2):205–232.
[30] Katsumi M, Grüninger M. Theorem proving in the ontology lifecycle. In: Conf. on Knowledge Engineering and Ontology Development (KEOD 2010), Oct 25-28, Valencia, Spain; 2010. p. 37–49.
[31] Hinrichs TL, Genesereth MR. Extensional reasoning. In: CADE-21 ESARLT Workshop, Jul 17, Bremen, Germany. CEUR Vol. 257; 2007. p. 6:1–10.
[32] Ramachandran D, Reagan P, Goolsbey K. First-orderized research Cyc: Expressivity and efficiency in a common-sense ontology. In: AAAI'05 Workshop on Context and Ontologies, Technical Report WS-05-01. AAAI Press; 2005. p. 33–40.
[33] Khasidashvili Z, Korovin K. Predicate elimination for preprocessing in first-order theorem proving. In: Conf. on Theory and Application of Satisfiability Testing (SAT 2016), Jul 5-8, Bordeaux, France. Springer; 2016. p. 361–372.