An Ordered Theory Resolution Calculus for Hybrid Reasoning in First-order Extensions of Description Logic

Scott Sanner and **Sheila McIlraith**
University of Toronto
Department of Computer Science
Toronto, ON, M5S 3H5, CANADA
{ssanner, sheila} @cs.toronto.edu

Abstract

Systems for hybrid reasoning with first-order logic (FOL) extensions of description logic (DL) date back at least 20 years and are enjoying a renewed interest in the context of recent FOL extensions of OWL DL for the Semantic Web. However, current systems for reasoning with such languages can only handle subsets of FOL or they do not fully exploit recent advances in *both* FOL theorem proving and DL inference. In response, we present an ordered theory resolution calculus for hybrid reasoning in *unrestricted* FOL extensions of the DL $\mathcal{SHI}$. This calculus permits near-seamless integration of highly optimized FOL theorem provers and DL reasoners while minimizing redundant reasoning and maintaining soundness and refutational completeness. Empirical results demonstrate the potential of this approach in comparison to the state-of-the-art FOL theorem provers Vampire, Otter, and SPASS.

1 Introduction

It is widely acknowledged that knowledge representation languages based on (decidable) description logics (DLs), such as OWL DL (Patel-Schneider, Hayes, & Horrocks 2003), are not sufficiently expressive to encode many real-world application domains (e.g., (Borgida 1996; Dean 2004)). The acknowledged merits of DL languages and the desire for expressive languages that are upwardly compatible with emerging DL-based Semantic Web languages have been the driving force behind the development of a number of rule-based extensions to OWL DL and the like, most notably RuleML (Boley *et al.* 2002) and SWRL (Horrocks *et al.* 2004). Unfortunately, even these languages are not suitable for certain tasks, such as the description of Web service process models and their associated automated reasoning tasks (Berardi *et al.* 2004). This has contributed to the development of first-order logic (FOL) extensions to Semantic Web DL-based languages e.g., SWRL-FOL (Patel-Schneider 2004), FOL RuleML (Boley *et al.* 2004), and SWSL (Battle *et al.* 2005).

In this paper, we address the issue of how to reason efficiently with first-order extensions of DLs. Most DLs are fragments of FOL; In particular OWL DL, which corresponds to the expressive $\mathcal{SHOIND}_n^-$ DL is a fragment of FOL (Horrocks & Patel-Schneider 2003; Tsarkov *et al.* 2004). As such, an obvious way to reason with an FOL extension of OWL DL is to translate it into FOL and use a state-of-the-art theorem prover. Unfortunately, initial experimentation by Tsarkov et al (Tsarkov *et al.* 2004) indicates that an efficient DL reasoner can outperform a highly optimized FOL theorem prover. Tsarkov and colleagues performed a number of experiments translating OWL DL to FOL and performing inference using the highly optimized Vampire (Riazanov & Voronkov 2002) theorem prover. Without optimization of the DL-to-FOL translation, Vampire performance was markedly worse than the FaCT++ (Tsarkov & Horrocks 2004) DL reasoner. Even when the translation was optimized, performance improved but was still two orders of magnitude worse on problems that could be solved. While Vampire was not optimized for deciding DLs (c.f. Hustadt *et al* (2005)), it does indicate that efficient DL reasoners are a good alternative in the absence of DL-optimized theorem proving implementations.

Motivated by these observations, we propose a hybrid approach to reasoning with FOL extensions of expressive DLs, henceforth referred to as DL-FOL. In DL-FOL, the DL handles the $\mathcal{SHI}$-expressible axioms of the theory and the FOL handles the remaining axioms possibly relating to, and *refining*, the DL axioms. Rather than translating the entire DL-FOL KB to FOL, we propose an instantiation of the ordered theory resolution calculus (Baumgartner 1992) that permits the use of highly optimized DL reasoners with optimized strategies for resolution-based theorem proving. Our calculus exploits the inherent structure of DL taxonomic hierarchies together with ordered resolution techniques that substantially reduce the resolution search space. Preliminary empirical results comparing our proof-of-concept DL-FOL reasoner implementation with the state-of-the-art theorem provers Vampire (Riazanov & Voronkov 2002), Otter (McCune 2003), and SPASS (Weidenbach 2001) demonstrate the potential of this approach.

The paper proceeds as follows: In Section 2 we provide background on relevant logical languages and automated reasoning systems. In Section 3 we define our DL-FOL language and in Section 4, we propose a general ordered theory resolution calculus, proving its soundness and completeness in Section 4.3. In Section 5, we present initial empirical results and we conclude in Section 6 with a discussion of our contributions and areas of future work.

2 Background

In this section we briefly review state-of-the-art automated reasoning techniques that are relevant to DL-FOL reasoning.

2.1 Description Logic

Recent advances in tableaux-based DL inference algorithms for (un)satisfiability checking have resulted in sound and complete inference procedures for expressive DL languages such as $\mathcal{SHIQ}$ (Horrocks, Sattler, & Tobies 1999; 2000). Implementations of these inference procedures such as Racer (Haarslev & Moeller 2001) and FaCT++ (Tsarkov & Horrocks 2004) are known to be very efficient in practice. In this paper, we treat description logic reasoning as a black box for unsatisfiability queries; we only require that the DL be decidable and the reasoning system be sound and complete.

2.2 First-order Logic

Automated theorem proving in first-order logic is a well-explored field and we refer the reader to a standard reference (Chang & Lee 1973) for background on clausal normal form, unification, and resolution theorem proving. Refinements of resolution have yielded techniques such as ordered resolution (Bachmair & Ganzinger 2001) for restricting the search space, and theory resolution for incorporating special-purpose theory reasoning (Stickel 1985). Most importantly for our work is the combination of both ideas in a sound and complete ordered theory resolution framework (Baumgartner 1992). On the implementation side, there are a number of highly-optimized reasoners for full first-order theorem proving such as Vampire (Riazanov & Voronkov 2002), Otter (McCune 2003), and SPASS (Weidenbach 2001). SPASS is of particular interest because it has special-purpose unification algorithms for handling monadic predicates that frequently occur in sorted logics and FOL translations of DLs.

2.3 Horn Extensions of Description Logic

Previous work on the CARIN language combines DLs and Horn rules (Levy & Rousset 1996) but is extremely restricted such that *all* inference is decidable. AL-Log (Donini *et al.* 1998) combines DLs with Datalog rules, but requires the concept and role symbols of the structural (DL) component to be disjoint from the predicate symbols in the relational (Datalog) component.

A number of more expressive rule *language extensions* of OWL DL have been defined for the Horn clause subset of FOL with OWL DL types (Horrocks *et al.* 2004; Boley *et al.* 2002). There have also been a number of proposals for *reasoning systems* using default or closed-world assumptions (Grosof *et al.* 2003; Golbreich 2004) in these languages. We note that given the extremely expansive nature of the Semantic Web, we often want to avoid a closed-world assumption (Reiter 1978) so that inferences are monotonic and hold even in the presence of additional KB content that could be encountered in the future. In this vein, a full theorem prover has been used for a rule language extension of OWL DL by simply translating the DL portion to FOL (Tsarkov *et al.* 2004). However, as previously noted, such reasoning can be inefficient in certain inference tasks for which DL reasoners are directly optimized.

2.4 FOL Extensions of Description Logic

Various extensions of FOL have used simple DL sort theories to restrict quantified variables (Frisch 1985; Cohn 1989). However, these extensions place significant limitations on the use of DLs in FOL statements. Constrained resolution (Buerckert 1994) allows a constraint theory over quantified variables that permits the use of DLs; however, with a few extensions, it can be viewed as a variant of theory resolution (Stickel 1985) that we discuss shortly. Recent work on resolution calculi for *deciding* expressive DLs (Hustadt *et al.* 2004; Hustadt, Motik, & Sattler 2005) holds the interesting possibility of providing efficient reasoning for DLs *fully within the resolution framework*. However, current implementations are limited to function-free disjunctive Datalog (Motik 2006).

Recent language extensions of OWL DL extend it with FOL constructs, e.g., SWRL-FOL (Patel-Schneider 2004), FOL RuleML (Boley *et al.* 2004), and SWSL (Battle *et al.* 2005). For reasoning in such languages, systems dating back to Krypton (Brachman, Fikes, & Levesque 1983) provided techniques for enhancing resolution using a DL system. (Ordered) theory resolution (Stickel 1985; Baumgartner 1992) somewhat enhanced and generalized this result by providing a refutation-complete (ordered) resolution procedure for incorporating decidable first-order theories into reasoning *without* duplicating the theory axioms in the KB. However this work did not address theory-specific computational issues and no follow-on work appears to provide an *efficient* theory resolution procedure for an expressive DL theory such as $\mathcal{SHI}$. As we will see, theories such as $\mathcal{SHI}$ make it particularly difficult to identify the theory-refuting substitutions required by ordered theory resolution and carefully working around these difficulties comprises one of the major contributions of this paper.

3 DL-FOL: an FOL Extension of DL

In this section we introduce DL-FOL, an FOL extension of the DL $\mathcal{SHI}$[1]. A DL-FOL KB comprises the following components:

1. **DL Component:** expressed in standard $\mathcal{SHI}$ DL syntax. We allow *general concept inclusion* (GCI) axioms since decision procedures exist for this case (Horrocks, Sattler, & Tobies 1999). We also allow *cyclic terminologies*, but assume they have been internalized according to Horrocks *et al* (1999).
2. **FOL Component**: expressed in a minor modification to a standard equality-free FOL syntax (Chang & Lee 1973) that includes the distinguished symbols $\top$ and $\bot$. Internally the FOL component is stored in clausal form.

[1] We note that $\mathcal{SHI}$ is a subset of OWL DL which corresponds to $\mathcal{SHOIND}_n^-$. Some of the additional expressivity provided by OWL DL over $\mathcal{SHI}$ are individuals $\mathcal{O}$, number restrictions $\mathcal{N}$, and concrete data types $\mathcal{D}$ (such as strings and integers).

DL-FOL $\mathcal{SHI}$ constructors with DL, DL' and FOL components

Constructor	*DL*	*DL'*	*FOL*
Atomic concept	A	A	
Top (Thing)	$\top$	$\top$	
Bottom (Nothing)	$\bot$	$\bot$	
Atomic role	R	R	
Inverse role	$R^*(R^* \equiv R^-)$	R^*	$\forall x, y.\ R(x,y) \equiv R^*(y,x)$
Transitive role	$R^*(R^* \equiv R, R \in \mathrm{R}_+)$	R^*	$\forall x, y, z.\ R^*(x,y) \wedge R^*(y,z) \rightarrow R^*(x,z)$
Negation	$\neg C$	$\neg C$	
Conjunction	$C \sqcap D$	$C \sqcap D$	
Disjunction	$C \sqcup D$	$C \sqcup D$	
Value restriction	$A^*(A^* \equiv \forall R.C)$	A^*	$\forall x.\ A^*(x) \equiv \forall y.\ R(x,y) \rightarrow C(y)$
Exists restriction	$A^*(A^* \equiv \exists R.C)$	A^*	$\forall x.\ A^*(x) \equiv \exists y.\ R(x,y) \wedge C(x)$
Role filler restr.	A^*	A^*	$\forall x.\ A^*(x) \equiv R(x,c)$

DL-FOL axioms with DL, DL' and FOL components

Axiom	*DL*	*DL'*	*FOL*
Concept inclusion	$C \sqsubseteq D$	$C \sqsubseteq D$	
Concept equivalence	$C \equiv D$	$C \equiv D$	
Role hierarchy	$R \sqsubseteq S$	$R \sqsubseteq S$	
Role equivalence	$R \equiv S$	$R \equiv S$	
Concept assertion			$C(a)$
Role assertion			$R(a,b)$
FOL axiom ψ			ψ
FOL query ψ			$\neg\psi$

Table 1: Decomposition of DL-FOL into its $\mathcal{SHI}$ DL, DL' and FOL components. The DL' component is a restriction of the DL component that we will utilize in our completeness proofs. For the DL component, the additional statements in parenthesis denote assertions that should be made when the respective constructors are used. A^* and R^* refer to *newly* generated concept and role names that should be used when referring to the respective restrictions and roles they define.

Given a DL-FOL KB consisting of $\mathcal{SHI}$ DL and FOL assertions that can freely reference the same concept and role symbols, it is decomposed into DL and FOL components according to Table 1.[2] In Table 1 we also introduce an alternate DL' component, which we will utilize in our completeness proofs. It is important to note that *the DL-FOL decomposition introduces redundancy between the DL and FOL components in the cases of inverse and transitive roles and role restrictions*. This redundancy does not occur between the DL' and FOL components.

To make this decomposition more concrete, we introduce a simple DL-FOL KB and provide its DL, DL' and FOL components. Given a DL-FOL KB consisting of one assertion $\{C \equiv A \sqcap \exists R^-.B\}$, we would assert the following DL, DL' and FOL components:

$$\begin{aligned} \text{DL} &: \{C \equiv A \sqcap A^*,\ A^* \equiv \exists R^*.B,\ R^* \equiv R^-\} \\ \text{DL'} &: \{C \equiv A \sqcap A^*\} \\ \text{FOL} &: \{\forall x.\ A^*(x) \equiv \exists y.\ R^*(x,y) \wedge B(y), \\ &\quad \forall x, y.\ R(x,y) \equiv R^*(y,x)\} \end{aligned}$$

Note that A^* and R^* are newly generated symbols that should be used to refer to their respective restriction and role definitions. Also note that in the DL' component $A^* \equiv \exists R.B$ and $R^* \equiv R^-$ are redundant with the FOL component and are omitted in accordance with Table 1.

When the DL-FOL KB is queried, the query should be negated and asserted in the FOL component regardless of whether it is DL expressible (DL is a known subset of FOL). In the next section, we will provide an ordered theory resolution procedure for refuting the negated query.

It is important to mention that while our DL-FOL language may evoke comparisons to sorted or constrained logics with taxonomic sort theories (Frisch 1985; Cohn 1989; Buerckert 1994), the DL component of DL-FOL is not restricted to variable typing only. For example, we can define a DL-FOL KB with the following components:

$$\begin{aligned} \text{DL} &: \{CompetentCEO \equiv CEO \sqcap CompetentWorker\} \\ \text{FOL} &: \{\forall x.\ CompetentWorker(x) \equiv \\ &\quad Person(x) \wedge \exists y.\ hasJob(x,y) \\ &\quad \wedge \exists z.\ requiresSkill(y,z) \wedge hasSkill(x,z)\} \end{aligned}$$

In this case, the FOL component asserts a non-DL expressible concept, and the DL component builds upon this assertion with a term definition making use of the FOL concept. Consequently, in DL-FOL, the FOL component actually *extends* the DL component as opposed to using it as a simple taxonomic sort theory. Thus, any non-fully redundant calculi for reasoning in DL-FOL must address the complex interactions between the FOL and DL components while maintaining completeness.

[2] While we do not provide a procedure here, it is possible to recognize many FOL axioms that can be represented within the $\mathcal{SHI}$ DL syntax, thus allowing such FOL axioms to be asserted and reasoned within the DL component.

Ordered Factoring $\frac{C}{C\sigma}$	if (1) σ is the most general syntactic unifier for some $\{L_1,\ldots,L_n\} \subseteq C$, and (2) $L_1\sigma$ is maximal in $C\sigma$
Ordered Narrow Theory Resolution $\frac{C_1 \ldots C_n}{(C_1\sigma - \{L_1\sigma\}) \cup \ldots \cup (C_n\sigma - \{L_n\sigma\})}$	if (1) $\sigma \in CSR_{\mathcal{T}}(\{L_1,\ldots,L_n\})$ for some $L_1 \in C_1,\ldots,L_n \in C_n$, and (2) $L_i\sigma$ is maximal in $C_i\sigma$ (for $i = 1\ldots n$)

Table 2: The inference rules of the ordered *narrow* theory resolution calculus.

4 Ordered Theory Resolution for DL-FOL

We begin in Section 4.1 by summarizing Baumgartner's ordered theory resolution (OTR) (Baumgartner 1992) for a generic theory $\mathcal{T}$ and then instantiate it with a specific $\mathcal{SHI}$ DL theory for DL-FOL in Section 4.2. We prove the soundness and completeness of OTR for DL-FOL in Section 4.3 and then discuss practical resolution refinements and search strategies in Section 4.4.

4.1 Ordered Theory Resolution

For completeness, we summarize Baumgartner's definition of the narrow[3] OTR calculus (Baumgartner 1992):

Definition 1. LITERAL ORDERING *(Baumgartner 1992) Let $\succeq$ be a partial ordering on terms and let $\succ$ denote the strict subset of $\succeq$. Let $\succ$ satisfy the following conditions, where $(X,Y) \in Term \times Term$ or $(X,Y) \in Literal \times Literal$:*

1. *$\succ$ is stable, i.e. for all substitutions σ: if $X \succ Y$ then $X\sigma \succ Y\sigma$.*
2. *$\succ$ is total on ground terms and $\succ$ is total on ground literals.*

We define $X \preceq Y$ iff $Y \succeq X$ and $X \prec Y$ iff $Y \succ X$. Let M be a literal set. $L \in M$ is $maximal$ in M iff for all $L' \in M$ it holds that $L \not\prec L'$ (or, equivalently, iff there does not exist a $L' \in M$ s.t. $L \prec L'$). $max(M)$ denotes the set of all maximal literals of M.

Examples of orderings meeting these criteria are the well-known lexicographic path orderings and recursive path orderings (Dershowitz & Plaisted 2001). Orderings are extremely useful since they substantially restrict the resolution search space.

A clause is a set of literals $\{L_1,\ldots,L_n\}$, often written as $L_1 \vee \ldots \vee L_n$. The non-theory portion of the KB axioms are converted to a set of clauses. We require that a theory $\mathcal{T}$ be representable by a set of satisfiable clauses and that it provide a decision procedure for determining the unsatisfiability of a set of literals $\{L_1,\ldots,L_n\}$. In determining (un)satisfiability, it is sufficient to limit the model theory to consider Herbrand intepretations only, so we define a Herbrand $\mathcal{T}$-$interpretation$ to be any total function from the set of ground atoms to $\{true, false\}$. A $\mathcal{T}$-$interpretation$ is an interpretation satisfying the theory $\mathcal{T}$. A clause set Φ is $satisfiable$ iff there exists an interpretation that simultaneously assigns $true$ to all ground instances of its members, or else it is $unsatisfiable$. A *literal set* $\mathcal{L}$ is $\mathcal{T}$-$satisfiable$ iff there exists an interpretation that satisfies theory $\mathcal{T}$ and simultaneously assigns $true$ to all ground instances of its members, or else it is $\mathcal{T}$-$unsatisfiable$.

Unlike ordinary resolution, the uniqueness of a most general unifier (MGU) is not guaranteed in theory resolution. Thus, we must generalize the concept of most general unifiers (MGUs) to that of a set of most general theory refuting substitions.

Definition 2. THEORY REFUTING SUBSTITUTION *(Baumgartner 1992) Let $\mathcal{L}$ be a literal set. $\mathcal{L}$ is $\mathcal{T}$-$complementary$[4] iff for all ground substitutions γ the set $\mathcal{L}\gamma$[5] is $\mathcal{T}$-$unsatisfiable$. $\mathcal{L}$ is minimal $\mathcal{T}$-$complementary$ iff $\mathcal{L}$ is $\mathcal{T}$-$complementary$ and all subsets $\mathcal{L}' \subset \mathcal{L}$ are not $\mathcal{T}$-$complementary$.*

We say that $\mathcal{L}$ is (minimal) $\mathcal{T}$-$refutable$ by σ iff $\mathcal{L}\sigma$ is (minimal) $\mathcal{T}$-$complementary$.

A set of substitutions is a complete and most general set of $\mathcal{T}$-$refuting$ substitutions for $\mathcal{L}$ (or short $CSR_{\mathcal{T}}(\mathcal{L})$) iff

1. *(Correctness) for all $\sigma \in CSR_{\mathcal{T}}$, $\mathcal{L}$ is $\mathcal{T}$-$refutable$ by σ*
2. *and (Completeness) for all substitutions θ such that $\mathcal{L}$ is $\mathcal{T}$-$refutable$ by σ, there exists a $\sigma \in CSR_{\mathcal{T}}$ and a substitution σ' such that $\theta = \sigma\sigma'|var(\theta)$.*

We are now ready to provide the rules of inference for the *narrow* ordered theory resolution calculus. These are given in Table 2. We note that Baumgartner proves soundness and completeness of this calculus when a procedure can be provided for theory $\mathcal{T}$ that determines the complete and most general set of $\mathcal{T}$-$refuting$ substitutions for a set of literals $\mathcal{L}$ (i.e., $CSR_{\mathcal{T}}(\mathcal{L})$).

4.2 Ordered Theory Resolution for DL-FOL

Having defined the general ordered theory resolution calculus, we now explain how we apply it to reasoning in DL-FOL. In our case, our theory $\mathcal{T}$ will consist of an $\mathcal{SHI}$ DL theory that we assume is satisfiable[6] and which meets the previously outlined conditions of being expressible as a

[3] We say *narrow* because the theory $\mathcal{T}$ must decide the unsatisfiability of two *or more* literals (Stickel 1985).

[4] This subsumes the notion of "syntactically complementary" and thus standard resolution where two literals are complementary if they are identical but of opposite polarity.

[5] The substitution γ is applied to each element of $\mathcal{L}$.

[6] If satisfiability of the DL theory is in question, we can easily do a consistency check to verify this.

set of clauses and having a decision procedure for unsatisfiability. Whenever an assertion (or query) is added to the DL-FOL system, it is asserted directly in the DL and FOL components according to Table 1.

At each step of ordered narrow theory resolution, one of the inference rules from Table 2 is applied to the FOL component.[7] Search terminates with a refutation if the empty clause is derived in the FOL component at any inference step.[8] The *only* part of the narrow OTR calculus that is specific to DL-FOL is the task of finding a correct and complete set of theory refuting substitutions $CSR_{\mathcal{T}}$. In Algorithm 1, we provide the procedure $FIND\text{-}CSR_{DL}(\mathcal{L})$ which uses an $\mathcal{SHI}$ DL theory as the theory $\mathcal{T}$ to determine a set of $\mathcal{T}\text{-}refuting$ substitutions for a set of literals $\mathcal{L}$.

The $FIND\text{-}CSR_{DL}(\mathcal{L})$ proceeds in a straightforward manner. If the set of literals $\mathcal{L}$ contains mixed monadic, dyadic, and n-arity ($n > 2$) literals, the procedure calls itself recursively for the monadic and dyadic subsets. Since n-arity ($n > 2$) literals cannot occur in the DL theory, any MGUs for pairs of n-arity literals are returned along with the substitutions returned by the recursive $FIND\text{-}CSR_{DL}(\mathcal{L})$ calls for the monadic and dyadic subsets of $\mathcal{L}$. When $\mathcal{L}$ contains only dyadic (monadic) literals, the CSR set is initialized to the pairs of syntactically complementary literals in $\mathcal{L}$ and CSR is augmented with any unifying substitutions for role (concept) literals whose (conjoined) predicate names extracted by $Pred(\cdot)$ are disjoint w.r.t. the DL role (concept) taxonomy. In this way $FIND\text{-}CSR_{DL}(\mathcal{L})$ covers both theory and standard resolution in accordance with Def. 2.

We note that the expressiveness of $\mathcal{SHI}$ poses some difficult problems that have been carefully worked around in the definition of the DL and FOL components and the design of $FIND\text{-}CSR_{DL}(\mathcal{L})$. Specifically, we note that theory refuting substitutions for a full $\mathcal{SHI}$ DL component *without* FOL redundancies for role restrictions can introduce arbitrarily large function symbols, even when the literals being refuted contain only variables and constants!

Following, we provide a few examples to demonstrate these issues. However, we begin our examples by making an important observation: the complete set of theory refuting substitutions for our theory $\mathcal{T}$ (i.e., $CSR_{\mathcal{T}}(\mathcal{L})$) is necessarily independent of the procedure we use for deciding $\mathcal{T}\text{-}unsatisfiability$. This is a consequence of the fact that Def. 2 for $CSR_{\mathcal{T}}(\mathcal{L})$ is based on $\mathcal{T}\text{-}complementarity$, which is a model-theoretic notion independent of any decision procedure for $\mathcal{T}$. Thus, if we use a decidable resolution procedure for the *clausal* representation of a theory $\mathcal{T}$, then the $CSR_{\mathcal{T}}(\mathcal{L})$ given by this decision procedure *must* match the $CSR_{\mathcal{T}}(\mathcal{L})$ given by any other decision procedure for $\mathcal{T}$.

Consequently, in the following examples, we use resolution on the clausal representation of an $\mathcal{SHI}$ DL theory as a decision procedure for determining $CSR_{\mathcal{DL}}(\mathcal{L})$.[9] Any alternative decision procedure for $CSR_{\mathcal{DL}}(\mathcal{L})$ *must* necessarily derive the same substitutions (modulo renaming of functions occurring only in the clausal representation of $\mathcal{DL}$).

Algorithm 1: $FIND\text{-}CSR_{DL}(\mathcal{L}) \longrightarrow CSR_{DL}(\mathcal{L})$

```
input   : DL, L :  an SHI DL theory and a set of literals
          {L_1, ..., L_n} to refute using theory or standard resolution
output  : CSR_DL(L) : complete & most general T-refuting subst.
          of L w.r.t. SHI DL plus MGUs for standard resolution
begin
    // Find refuting subst. of monadic, dyadic and compl. literals separately
    if   (L consists of mixed monadic, dyadic, and n-arity (n > 2) literals)
    then
        L_M := monadic literal subset of L from DL component;
        L_D := dyadic literal subset of L from DL component;
        σ_C := set of MGUs for pairs of n-arity syntactically
               complementary literals of L;
        return FIND-CSR_DL(L_M) ∪ FIND-CSR_DL(L_D) ∪ σ_C;
    CSR := set of MGUs for any pair of syntactically complementary
           literals of L;
    if (L consists of dyadic literals) then
        // Find refuting substitution of dyadic literals in DL component
        // (i.e., DL roles)
        foreach (pair of literals ⟨L_1, L_2⟩ from L where L_1 has positive
                 polarity and L_2 has negative polarity) do
            // Predicate sumbols ignored, MGU of term lists only
            σ := MGU(L_1, L_2);
            if (σ ≠ null ∧ Pred(L_1) ⊑_DL Pred(L_2)) then
                CSR := CSR ∪ σ;
    else
        // Find refuting substitution of monadic literals in DL component
        // (i.e., DL concepts)
        for (s := 2..|L|) do
            foreach (set of literals ⟨L_1, ..., L_s⟩ from L of size s
                     where a subset has not already been refuted) do
                σ := ∅;
                L_C := ⊤;
                for (k := 2..s) do
                    // Pred. symbols ignored, MGU of term lists only
                    σ := compose(σ, MGU(L_{k-1}, L_k));
                    L_C := L_C ⊓ Pred(L_{k-1});
                if (σ ≠ null ∧ L_C ⊑_DL ¬Pred(L_s)) then
                    CSR := CSR ∪ σ;
    return CSR;
end
```

Example 1. *Suppose that we are given the DL-FOL KB assertions* $\{\exists R.A \sqsubseteq B,\ \neg B(c),\ R(c,d),\ A(d)\}$.[10] *Then we obtain the* clausal *representation of the DL and FOL components for this KB:*

$$\begin{aligned} DL &: \{\neg R(x,y) \vee \neg A(y) \vee B(x)\} \\ FOL &: \{\neg B(c),\ R(c,d),\ A(d)\} \end{aligned}$$

We can refute this DL-FOL KB by resolving all three singleton FOL clauses. In this case, it is clear that $CSR_{\mathcal{DL}}(\mathcal{L}) = \{\{x/c, y/d\}\}$ *(this is the only possible theory resolution).*

[7] We defer discussion of specific clause selection strategies to Section 4.4.

[8] We assume that $\perp$ literals are automatically removed from clauses in a preprocessing step.

[9] Standard resolution terminates on all of our DL examples. Consult Grosof *et al* (2003) for an FOL (and by CNF transformation, clausal) representation of the DL theory.

[10] In our examples, we use w, x, y, z to denote variables, all remaining 0-arity terms should be interpreted as constants.

The last example was simple and no unexpected surprises occurred. However, in the next example, we will see that if we change the DL-FOL KB slightly then the result is not so straightforward.

Example 2. *Suppose that we are given the DL-FOL KB assertions* $\{\forall R.A \sqsubseteq B,\ \neg B(c),\ \forall z.\ \neg R(c,z) \vee A(z)\}$. *Then we obtain the* clausal *representation of the DL and FOL components for this KB:*

$$\begin{aligned} DL :& \{R(x, f(x)) \vee B(x),\ \neg A(f(y)) \vee B(y)\} \\ FOL :& \{\neg B(c),\ \neg R(c,z) \vee A(z)\} \end{aligned}$$

While this DL-FOL KB is refutable, we must do it in two narrow theory resolution steps. In the first step, we choose $\mathcal{L} = \{\neg R(c,z), \neg B(c)\}$ *to refute. In this case* $CSR_{\mathcal{DL}}(\mathcal{L}) = \{\{z/f(c)\}\}$ *(this is the only possible resolution). Because the literal* $\neg R(c,z)$ *came from the clause* $\neg R(c,z) \vee A(z)$*, this yields the resolvent* $A(f(c))$*. In the next step, we choose to refute* $\mathcal{L} = \{A(f(c)), \neg B(c)\}$*. In this case* $CSR_{\mathcal{DL}}(\mathcal{L}) = \{\emptyset\}$ *(the empty substitution) and the resolvent is* $\perp$*. There is only one other sequence of possible resolutions and it yields the same final result.*

While the previous example demonstrated a straightforward application of resolution, we note an interesting phenomenon: although the FOL theory initially contained no function symbols, the first theory resolution step introduced a function symbol into the FOL theory. While this is obvious from the clausal representation of the DL theory, we note that any non-resolution decision procedure for determining $CSR_{\mathcal{DL}}(\mathcal{L})$ would *also* need to introduce function symbols in this example case. But this is only the beginning of the problem, as it turns out in the next example, narrow theory resolution with an $\mathcal{SHI}$ DL theory can introduce even larger arity functions symbols into a function-free FOL theory.

Example 3. *Suppose that we are given the following DL-FOL KB assertions* $\{\exists S.\forall R.A \sqsubseteq B,\ \neg B(c),$ $\exists w \forall z.\ S(c,w) \wedge (\neg R(w,z) \vee A(z))\}$*. Then we obtain the* clausal *representation of the DL and FOL components for this KB:*

$$\begin{aligned} DL :& \{\neg S(x,y) \vee R(y, f(x,y)) \vee B(x), \\ & \neg S(x,y) \vee \neg A(f(x,y)) \vee B(x)\} \\ FOL :& \{\neg B(c),\ S(c,d),\ \neg R(d,z) \vee A(z)\} \end{aligned}$$

This DL-FOL KB is refutable in two narrow theory resolution steps. In the first steps we refute $\mathcal{L} = \{\neg B(c), S(c,d), \neg R(d,z)\}$ *with* $CSR_{\mathcal{DL}}(\mathcal{L}) = \{\{z/f(c,d)\}\}$ *to obtain the resolvent* $A(f(c,d))$*. In the next step, we choose to refute* $\mathcal{L} = \{A(f(c,d)), \neg B(c), S(c,d)\}$*. In this case* $CSR_{\mathcal{DL}}(\mathcal{L}) = \{\emptyset\}$ *(the empty substitution) and the resolvent is* $\perp$*. There is only one other sequence of possible resolutions and it yields the same final result.*

This is all to say that narrow theory resolution for the $\mathcal{SHI}$ DL theory has introduced a function symbol of arity 2 into the originally function-free FOL theory. The astute observer will notice that we can generalize the structure of this example to obtain arbitrary size function symbols.

Theorem 1. *A complete set of theory refuting substitutions for an* $\mathcal{SHI}$ *DL theory can introduce arbitrary sized function symbols into a function-free FOL theory.*

Proof Sketch. Following the above examples, we can construct DL-FOL KB of the following form for for arbitrarily large n:

$$\begin{aligned} &\{\exists S_1.\exists S_2.\ldots.\exists S_n.\forall R.A \sqsubseteq B,\ B(c), \\ &\quad \exists w_1, w_2, \ldots, w_n \forall z.\ S_1(c, w_1) \wedge S_2(w_1, w_2) \wedge \\ &\quad \ldots \wedge S_n(w_{n-1}, w_n) \wedge (\neg R(w_n, z) \vee A(z))\} \end{aligned}$$

While the derivation is tedious, it is a straightforward procedure to verify that the DL theory introduces a function symbol of size n into a function-free FOL theory. This follows from the fact that to convert the DL GCI ($\sqsubseteq$) axiom to clausal form, we will need to *negate* the LHS of the GCI resulting in a chain of quantifiers $\forall w_1\ \ldots\ \forall w_n\ \exists z$ where the innermost variable z will be Skolemized to $f(w_1, \ldots, w_n)$. On the other hand, the FOL theory will have *no function symbols* because it has the quantifier chain $\exists w_1\ \ldots\ \exists w_n\ \forall z$. Now, given the complementary structure of this DL-FOL KB it is trivial to show that it must be refutable. Following a resolution derivation similar to Ex. 3 that requires exactly two steps, we can show a refutation of this DL-FOL KB that introduces the n-arity function symbol $f(\cdots)$ into the FOL theory after the first step. Again, this holds for any resolution refutation of this KB, no matter what order the resolutions are applied. Thus, we know that any theory refutation of this KB must introduce an n-arity function symbol.

Now, Theorem 1 introduces a particular difficulty for us if we want to use a blackbox DL reasoner to determine $CSR_{\mathcal{DL}}(\mathcal{L})$ for our $\mathcal{SHI}$ DL theory. The blackbox DL reasoner must be able to properly interpret the function symbols that the DL theory can introduce into the FOL theory on account of its clausal representation. Yet it is not clear how this could be done with today's state-of-the-art tableau reasoners that neither perform resolution nor handle function symbols. We work around this in our calculus by making DL role restrictions (and thus the potential source of function symbols in the DL theory) redundant with the FOL component of the DL-FOL KB. In addition, to avoid the need for the DL theory to reason with function symbols from the FOL theory, all terms including constants (i.e., nominals) are also offloaded to the FOL component. It is not immediately clear whether there is a generalization of Algorithm 1 that will yield $CSR_{\mathcal{DL}}(\mathcal{L})$ for non-redundant inverse and transitive DL role definitions so we have also made these redundant w.r.t. the FOL theory.

We briefly mention a few reasons which help mitigate the fact that we have have introduced FOL redundancy w.r.t. the DL theory: 1) Most of the redundant FOL axioms can be represented as Horn clauses for which many efficient ordered resolution strategies exist. 2) Tabling and term indexing optimizations permit efficient FOL resolution with KBs consisting of large amounts of terms, so it is not necessarily a bad idea to handle nominals and complex terms in the FOL component. 3) If we extend the DL theory to $\mathcal{SHIN}$, including nominals in the theory would require that Algorithm 1 attempt to refute *arbitrary non-unifiable* sets

Given DL-FOL Axioms: (PT is for partially-tangible, BW for between, SL for spatial location, and OFIL for object found in location)

$$PT \sqsubseteq \top,\ Location \sqsubseteq \top,\ SL \sqsubseteq Location,\ City \sqsubseteq PT,\ River \sqsubseteq PT,\ ColdLocation \sqsubseteq SL$$
$$ColdLocation \equiv (Cold \sqcap Location) \sqcup \exists hasRegion(\exists hasClimate.arctic)$$
$$Location(canada),\ hasRegion(canada, nunavut),\ hasClimate(nunavut, arctic)$$
$$\forall w, x, y, z\ PT(w) \wedge PT(x) \wedge PT(y) \wedge SL(z) \rightarrow OFIL(w, z) \wedge OFIL(x, z) \wedge BW(w, x, y) \rightarrow OFIL(y, z)$$

Query:

$$\forall w, x, y.\ City(w) \wedge City(x) \wedge River(y) \rightarrow \exists z.\ SL(z) \wedge (OFIL(w, z) \wedge OFIL(x, z) \wedge BW(w, x, y) \rightarrow OFIL(y, z))?$$

Convert the given axioms and *negated* query to the DL-FOL DL and FOL components and apply the ordered theory resolution inference rules for DL-FOL from Table 2 as follows. We omit literal ordering in this proof, its specification would simply restrict the order in which the following inferences are made.

1a,b,c.	Given FOL Component	$Location(canada)$, $hasRegion(canada, nunavut)$, $hasClimate(nunavut, arctic)$
1d.	Given FOL Component	$\neg SL(z) \vee \neg PT(w) \vee \neg PT(x) \vee \neg PT(y) \vee \neg OFIL(w, z) \vee \neg OFIL(x, z) \vee \neg BW(w, x, y) \vee OFIL(y, z)$
2a.	Given FOL Component	$\neg hasClimate(x, arctic) \vee A^*(x)$ $\longleftarrow$ **Def. for** $\exists hasClimate.arctic \equiv A^*$
2b.	Given FOL Component	$\neg hasRegion(x, y) \vee \neg A^*(y) \vee A^{**}(x)$ $\longleftarrow$ **Def. for** $\exists hasRegion(A^*) \equiv A^{**}$
3.	Negated Query	$City(c_1)$
4.	Negated Query	$City(c_2)$
5.	Negated Query	$River(r)$
6.	Negated Query	$\neg SL(z) \vee OFIL(c_1, z)$
7.	Negated Query	$\neg SL(z) \vee OFIL(c_2, z)$
8.	Negated Query	$\neg SL(z) \vee BW(c_1, c_2, r)$
9.	Negated Query	$\neg SL(z) \vee \neg OFIL(r, z)$
10.	Narrow OTR, 1c with 2a, $\theta = \{x/nunavut\}$	$A^*(nunavut)$
11.	Narrow OTR, 2b with 1b,10, $\theta = \{x/canada, y/nunavut\}$	$A^{**}(canada)$
12.	Narrow OTR, 11 with 6, $\theta = \{z/canada\}$	$OFIL(c_1, canada)$
13.	Narrow OTR, 11 with 7, $\theta = \{z/canada\}$	$OFIL(c_2, canada)$
14.	Narrow OTR, 11 with 8, $\theta = \{z/canada\}$	$BW(c_1, c_2, r)$ $\longleftarrow$ **Complex refutation:** $\neg SL, A^{**} \sqsubseteq ColdLocation \sqsubseteq SL$
15.	Narrow OTR, 11 with 9, $\theta = \{z/canada\}$	$\neg OFIL(r, canada)$
16.	Narrow OTR, 1d with 3-5, $\theta = \{w/c_1, x/c_2\}$	$\neg SL(z) \vee \neg OFIL(c_1, z) \vee \neg OFIL(c_2, z) \vee \neg BW(c_1, c_2, r) \vee OFIL(r, z)$
17.	Narrow OTR, 16 with 11-15, $\theta = \{z/canada\}$	$\bot$ $\longleftarrow$ **Refutation, query proved!** $\square$

Table 3: Sample query-answering with the DL-FOL reasoning procedure.

of literals. As an example, consider the refutable DL-FOL KB $\{\leq 2R \sqsubseteq B, \neg B(a), R(a,1), R(a,2), R(a,3)\}$. In this case, the four literals $\{B(a), R(a,1), R(a,2), R(a,3)\}$ will need to be simultaneously refuted at some point even though they are not unifiable. Having to check all non-unifable literals could lead to a huge explosion in the complexity of DL theory reasoning. In contrast Algorithm 1 currently only tests satisfiability of *unifiable* sets of literals.

Having now explained the reasoning for our DL-FOL decomposition given in Table 1, we demonstrate a full application of our algorithm to an application of query-answering in Table 3 for the OpenCyc KB (CycCorp, Inc. 2005) augmented with a few additional assertions.

4.3 Soundness and Completeness

We now show that this procedure satisfies the two conditions of correctness and completeness from Definition 2 that are required to show soundness and completeness of the DL-FOL ordered theory resolution calculus. As a preliminary step, we note that the DL' component given in Table 1 is just a weakening of the DL component where the redundant DL axioms for the exists and value restrictions, and inverse and transitive roles are removed. We will first prove properties for a DL-FOL theory using a DL' component and then show that these results trivially generalize to a DL-FOL theory using the DL component.

Theorem 2. *Procedure $FIND\text{-}CSR_{DL'}(\mathcal{L})$ is correct. I.e., for all $\sigma \in CSR_{\mathcal{T}}$, $\mathcal{L}$ is $\mathcal{T}\text{-}refutable$ by σ.*

Proof Sketch. It is straightforward to verify that in each case where $FIND\text{-}CSR_{DL'}(\mathcal{L})$ adds a substitution σ to $CSR_{DL}(\mathcal{L})$, then $\mathcal{L}\sigma$ is $\mathcal{T}\text{-}complementary$ under theory DL'. Thus, no $\mathcal{T}\text{-}interpretation$ could simultaneously satisfy all literals in $\mathcal{L}\sigma$. and by definition, for all $\sigma \in CSR_{\mathcal{T}}$, $\mathcal{L}$ is $\mathcal{T}\text{-}refutable$ by σ.

Theorem 3. *Procedure $FIND\text{-}CSR_{\mathcal{DL}'}(\mathcal{L})$ is complete. I.e., for all substitutions θ such that $\mathcal{L}$ is $\mathcal{T}\text{-}refutable$ by σ, there exists a $\sigma \in CSR_{\mathcal{T}}$ and a substitution σ' such that $\theta = \sigma\sigma'|var(\theta)$.*

Ordered Partial Narrow Theory Resolution	
$\dfrac{C_1, C_2}{(C_1\sigma - \{L_1\sigma\}) \cup (C_2\sigma - \{L_2\sigma\}) \cup (L_1 \sqcap L_2)\sigma}$	if (1) $\sigma \in MGU(\{L_1, L_2\})$ (term-only MGU) for some $L_1 \in C_1, L_2 \in C_2$, and (2) $L_i\sigma$ is maximal in $C_i\sigma$ (for $i = 1 \ldots 2$)

Table 4: The ordered *partial narrow* theory resolution inference rule that replaces the *narrow* version.

Proof Sketch. $\mathcal{T}$-$refuting$ substitutions θ for $\mathcal{L}$ are only possible in three cases: (1) θ unifies two syntactically complementary literals in $\mathcal{L}$ where their MGU *must* subsume θ and be in $CSR_{\mathcal{DL}'}(\mathcal{L})$; (2) θ unifies two dyadic literals in $\mathcal{L}$ that are complementary via a role subsumption chain where their MGU *must* subsume θ and be in $CSR_{\mathcal{DL}'}(\mathcal{L})$. This follows from the observation that the only DL' role axioms are simple role inclusions and these are all representable as binary Horn clauses. In this case, the role taxonomy provides the closure of all inferences w.r.t. these axioms; Or (3) θ unifies a set (or subset) of monadic literals in $\mathcal{L}$ that renders the conjunction of the literal concept-names unsatisfiable in the DL' theory. This last statement follows from the fact that all DL' axioms for monadic literals consist solely of monadic literals sharing the same variable (the DL' axioms were designed to enforce a separation of the concept and role theories). This implies that no dyadic literals could influence unsatisfiability of $\mathcal{L}\theta$ since there would be no way to derive a refutation from the clausal representation of DL'. So we know that if $\mathcal{L}\theta$ is purely monadic, it is unsatisfiable under the DL' theory iff $L_1 \sqcap \ldots \sqcap L_n$ is unsatisfiable under the DL' theory. Now, assume there *does not* exist a $\sigma \in CSR_{\mathcal{DL}'}(\mathcal{L})$ and a substitution σ' such that $\theta = \sigma\sigma'|var(\theta)$. But we know that θ must unify some unsatisfiable set of literals, and that the MGUs for all possible sets and subsets of unsatisfiable literals $\mathcal{L}$ are in $CSR_{\mathcal{DL}'}(\mathcal{L})$. Then $\theta = \sigma\sigma'|var(\theta)$ for some $\sigma \in CSR_{\mathcal{DL}'}(\mathcal{L})$ and substitution σ'. Thus, by contradiction, we satisfy completeness for case (3). By construction of the DL and FOL theories, no other cases could exist. Thus, from the fact that cases (1), (2), and (3) are exhaustive and individually complete, we can infer the completeness of $FIND\text{-}CSR_{\mathcal{DL}'}(\mathcal{L})$.

Having shown that $FIND\text{-}CSR_{\mathcal{DL}'}(\mathcal{L})$ satisfies the correctness and completeness conditions, the soundness and refutation completeness of ordered theory resolution for DL-FOL for a weakened DL theory DL' is a direct consequence of Baumgartner's proof of the soundness and completeness of the ordered narrow theory resolution calculus (Baumgartner 1992). By strengthening the DL theory to the full DL of Table 2 and using $FIND\text{-}CSR_{\mathcal{DL}}(\mathcal{L})$, we only introduce redundancy (with the advantage of shorter refutation derivations), so soundness and refutation completeness are clearly preserved. This gives us our final result:

Theorem 4. *The ordered narrow theory resolution calculus for DL-FOL is sound and refutation complete.*

We note that in standard resolution calculi, tautological clauses are redundant and can be safely deleted without affecting completeness. Unfortunately, this result does not extend to (ordered) theory resolution (Stickel 1985; Baumgartner 1992). For example, consider a DL-FOL KB with DL component $\{A \equiv B, B \equiv C\}$ and FOL component Φ containing the clauses $\{A(x) \vee B(x) \vee C(x), \neg A(x) \vee \neg B(x) \vee \neg C(x)\}$. While this set of clauses is refutable via *narrow* OTR, any derivation of the empty clause necessarily requires the intermediate derivation of a tautology. We conjecture that extending ordered factoring to consider theory implication as opposed to syntactic equivalence may resolve this problem.

4.4 Ordered Theory Resolution Strategies

Ordered theory resolution leaves open the possibility of resolution strategy, yet this is perhaps the most critical aspect of the system w.r.t. efficient and effective reasoning. In this section, we adapt concepts used in modern resolution theorem proving strategies to exploit structure in DL-FOL.

A refutation-complete resolution strategy would assign each clause an index $1 \ldots n$ (newly generated clauses receive the next free index) and apply all inference rules for clause index k that involve clauses $1 \ldots k$.[11] We'll call this the *age* selection strategy, i.e., older clauses are selected before younger, more recently inferred clauses. A major refinement of this idea used in many modern theorem provers is the *age-weight* ratio $a : w$ selection strategy (McCune 2003) where for every $a + w$ clauses chosen, a are chosen according to the age selection strategy and w are chosen from a priority queue where each clause is assigned a weight. Clearly, so long as a is non-zero, the strategy remains refutation-complete, while allowing the incorporation of heuristic knowledge to select clauses that are likely to contribute to a refutation (e.g., clauses with fewer literals).

In a moment, we show how we can additionally exploit DL-FOL structure for determining the heuristic weight, but first we digress with a discussion of how to reduce *narrow* OTR to *partial narrow* OTR so that we need only resolve a maximum of two clauses at a time. To do this, we note that $FIND\text{-}CSR_{DL}(\mathcal{L})$ is extremely efficient for theory resolution cases where $|\mathcal{L}| = 2$. That is, to determine whether the conjunction of two role or concept literals L_1 and L_2 are unsatisfiable, it suffices to check whether $Pred(L_1) \sqsubseteq \neg Pred(L_2)$ in a DL taxonomy (where the $Pred(\cdot)$ function returns the predicate name for the literal).

Unfortunately, binary resolution is not complete for OTR. For example, take the following DL KB $\{D_1 = A \sqcap \neg B, D_2 = B \sqcap \neg C, D_3 = C \sqcap \neg A\}$ and FOL KB with three clauses $\{D_1(x)\ ;\ D_2(y)\ ;\ D_3(z)\}$. While

[11] Of course, the literal ordering is also important and we note that using a lexicographic path ordering where non-DL literals and DL role literals are given precedence over DL concept literals is a good choice since it postpones the most difficult literal resolutions until they are needed to obtain a refutation.

the FOL KB can be refuted by one *narrow* OTR step, no binary OTR steps with $|\mathcal{L}| = 2$ can be applied.

Fortunately, there is a refinement of theory resolution that restricts $\mathcal{T}\text{-}unsatisfiability$ checking to cases where $|\mathcal{L}| = 2$ *and* retains completeness. This refinement is known as *partial narrow* theory resolution (Stickel 1985) and it is given in Table 4. Of course, such simplicity must come with a catch, and this catch is that a partial narrow OTR must be applied regardless of whether $(L_1 \sqcap L_2)\sigma$ can be refuted. If $(L_1 \sqcap L_2)\sigma$ can be refuted by the DL theory then this literal can be removed from the consequence (this case is just standard OTR), otherwise the compound literal $(L_1 \sqcap L_2)\sigma$ is a *residue* that must be resolved away by additional partial narrow OTR steps. With proper precedence assigned to the residue literals, we can prove the completeness of partial narrow OTR.

Theorem 5. *If all compound residue literals are assigned an ordering precedence that is equal to the maximal precedence among their primitive constituents, then partial narrow OTR is complete.*

Proof Sketch. The proof of completeness follows from the fact that partial narrow OTR with the specified precedence for compound literals essentially simulates narrow OTR. By the age-weight ratio selection strategy, we know that we will eventually derive all possible combinations of axioms using partial narrow OTR (if the resolved literals do not refute, they are retained as compound residue literals to be resolved later). It is only sufficient to show that when literals would be resolved in the narrow OTR case, they would also be resolved in the case of partial narrow OTR. In short, if a non-binary resolution was performed on the clause set $\{C_1, \ldots, C_n\}$ with maximal refutable literal set $\mathcal{L}$, then two binary clauses together representing partial narrow theory resolvents for $\{C_1, \ldots, C_n\}$ would have to have a set of (potentially compound) maximal literals equivalent to $\mathcal{L}$. This follows from the fact that literal maximality is maintained under substitutions due to the stability property of orderings, and the precedence conditions of compound literals ensure their maximality in derived clauses. (We know these literals were maximal in their original clauses because of the narrow OTR resolution.) Thus, partial narrow OTR essentially simulates narrow OTR which was already proved complete.

To deal with the potential inefficiencies of partial narrow OTR, we must introduce weighting heuristics to be used with the age-weight ratio selection strategy. Heuristically, the larger a residue literal $(L_1 \sqcap \ldots \sqcap L_n)\sigma$ grows, the less likely it is to be refuted given that no proper subset could be refuted. We can build this heuristic into our strategy by assigning the priority weight w of a clause to *scale with the size of a residue literal in the clause*. When used with the *age-weight* ratio selection strategy, this postpones resolution of clauses with large residues that are unlikely to be resolved away, and thus yield even larger residues.

Another useful weighting heuristic is that of giving higher priority to literals associated with concepts and roles that are deeper in the taxonomy. We call this the *Prefer-Deep* strategy and its inverse the *Prefer-Shallow* strategy. The reason that we expect the *Prefer-Deep* strategy to be more efficient is that it prefers inferences relevant to specialized concepts and roles that are often much less prolific than inferences for concepts and roles near the top of their respective hierarchies. Since we need to refute all literals in a clause, it only makes sense to try refuting the more difficult ones first, i.e. the ones that deal with more specific requirements and for which fewer inference opportunities exist. In doing this, the size of the KB is also minimized which is important for efficient inference.

Altogether, partial narrow OTR with age-weight selection and the above weighting strategies give us a *refutation-complete* strategy that need only refute *binary* sets of literals and that exploits *taxonomic* information for selecting clauses to resolve.

5 Experiments with a Proof-of-Concept System

We investigated the application of DL-FOL reasoning to the spatial reasoning subset of the OpenCyc KB (CycCorp, Inc. 2005). For this KB, we extracted a small subset of the CycL language that could be represented as a subset of the $\mathcal{SHI}$ DL and translated the rest to FOL. We applied the partial narrow theory resolution procedure for DL-FOL using FaCT++ (Tsarkov & Horrocks 2004) as our black-box DL reasoner and compared it to the highly-optimized Vampire (Riazanov & Voronkov 2002), Otter (McCune 2003), and SPASS (Weidenbach 2001) theorem provers. We used three versions of our DL-FOL reasoner: the DL-FOL ordered resolution prover using only the full FOL translation of the DL-FOL KB and two versions of the partial narrow OTR inference system presented in this paper – one for the *Prefer-Deep* weighting heuristic and one for the *Prefer-Shallow* weighting heuristic.

While our experimental results illustrate the potential of our technique in minimizing clause generation and proof length, it should be noted that Vampire typically posted over an order of magnitude faster CPU time than our DL-FOL prover. Clearly an objective comparison of our proposed technique with Vampire, Otter and SPASS is not achievable. In particular, because of a lack of suitable DL plus FOL KBs[12], we constructed a set of benchmark problems that we felt would be difficult for current theorem provers to address (i.e., inferences that involve complex subsumption chains). This skewed the results slightly in our favor although Vampire slightly edges out DL-FOL in the average number of clauses generated. On the other hand, it is unfair to compare our proof-of-concept system with these highly optimized theorem provers. Better CPU performance by these systems is bound to reflect engineering efforts as much as an intrisically better approach. We additionally note that DL ontologies such as Galen (Rector, Nowlan, & Glowinski 1993) and Tambis (Baker *et al.* 1998) that have proved difficult for theorem provers such as Vampire (Tsarkov *et al.* 2004) *may* fare better for DL-FOL OTR reasoners if they were extended with full FOL axioms since the FaCT++ reasoner performed very well on these ontologies.

[12]Probably in part because there are no effective reasoners for DL plus FOL KBs.

Combined Results for the OpenCyc KB

Reasoner	# Successes	Avg. Clauses Gen.	Avg. Resolution Proof Length
Vampire v8	25/25	137	10.5
Otter v3.3	25/25	603	9.6
SPASS v2.1	25/25	4763	9.4
DL-FOL (FOL Translation Only)	5/25	N/A	N/A
DL-FOL (Partial Narrow OTR – *Prefer-Shallow*)	25/25	346	7.3
DL-FOL (Partial Narrow OTR – *Prefer-Deep*)	25/25	147	7.3

Table 5: For our subset of the OpenCyc KB, each reasoner was run on 25 queries. We report the total number of successful queries (all were provable as verified by the results), the average number of clauses generated, and the average number of resolution steps in the resulting proofs. Results are not shown when the theorem prover could not answer all queries within a 5 minute time limit.

Since our DL-FOL prover was not heavily optimized, perhaps the most telling results come from self-comparison. The DL ontology in our OpenCyc KB was fairly small, containing only 132 concepts, but this proved to be a problem when the entire DL-FOL KB was translated to FOL. In this case, ordered resolution spent the majority of its time making pointless inferences with the DL subclass axioms and only managed to prove a very small subset of the queries. On the other hand, this pointless inference did not occur for the partial narrow OTR reasoners which performed much better. While they both found the same proofs, we note that the *Prefer-Deep* strategy generated fewer clauses than the *Prefer-Shallow* strategy since it focused on refuting the "deeper" literals in the taxonomy for which there were fewer inferences to be made. Consequently, we see that the intuitions behind the *Prefer-Deep* heuristic were appropriate for this particular test KB and set of queries.

6 Conclusion

We have presented an instantiation of Baumgartner's ordered theory resolution calculus (Baumgartner 1992) for hybrid inference in FOL extensions of the $\mathcal{SHI}$ DL that minimizes redundancy of DL reasoning relative to the FOL component. This is a notable accomplishment since theory refuting substitutions for the $\mathcal{SHI}$ DL can introduce arbitrarily large function terms and DL reasoning languages (even those handling nominals) were not designed for such reasoning. In addition, we presented refinements of the basic ordered theory resolution strategy that were intended to further restrict and guide search using taxonomic information. As a result of all these considerations, the DL-FOL ordered theory resolution calculus permits the near-seamless integration of highly optimized off-the-shelf DL reasoners with optimized strategies for resolution-based theorem proving. Empirically, a proof-of-concept implementation of the DL-FOL hybrid reasoner demonstrates the potential of this approach in comparison to the alternate strategy of using a theorem prover on an FOL translation of the DL-FOL KB.

This research is just the beginning of an exciting new class of reasoners, and there are many extensions. We would like to explore saturation and redundancy extensions (Bachmair & Ganzinger 2001). In addition, we would like to look at superposition extensions of ordered resolution with equality as it appears in DL theories (c.f. (Hustadt, Motik, & Sattler 2005)), thus ultimately allowing us to define a complete ordered resolution calculus for the $\mathcal{SHIQ}$ DL or beyond. Also, given the difficulty of implementing an optimized theorem prover, but the simple way in which the DL-FOL calculus allows theorem provers and DL reasoners to interact, we would like to consider integrating our ordered theory resolution technique into a highly-optimized theorem prover. Such approaches hold the promise of allowing us to engineer state-of-the-art reasoning systems that can scale to the inference demands of large DL-FOL ontologies, especially those that we expect to encounter as such languages take hold on the Semantic Web.

Acknowledgements

We would like to thank Mark Stickel and the anonymous reviewers for their thorough comments and suggestions regarding this work. Many ideas in this paper stemmed from their input.

References

Bachmair, L., and Ganzinger, H. 2001. Resolution theorem proving. In Robinson, A., and Voronkov, A., eds., *Handbook of Automated Reasoning*, volume I. Elsevier Science. chapter 2, 19–99.

Baker, P.; Brass, A.; Bechhofer, S.; Goble, C.; Paton, N.; and Stevens, R. 1998. Tambis: Transparent access to multiple bioinformatics information sources: an overview. In *Sixth International Conference on Intelligent Systems for Molecular Biology*, 25–34.

Battle, S.; Bernstein, A.; Boley, H.; Grosof, B.; Gruninger, M.; Hull, R.; Kifer, M.; Martin, D.; McIlraith, S.; McGuinness, D.; and Su, J. 2005. Semantic web service language (swsl). Document located on-line at http://www.w3.org/Submission/SWSF-SWSL/.

Baumgartner, P. 1992. An Ordered Theory Resolution Calculus. In Voronkov, A., ed., *Logic Programming and Automated Reasoning (Proceedings)*, volume 624, 119–130. St. Petersburg, Russia: Springer.

Berardi, D.; Gruninger, M.; Hull, R.; and McIlraith, S. 2004. Towards a first-order ontology for semantic web services. In *Working notes of the W3C Workshop on Constraints and Capabilities for Web Services*. Document located on-line at http://www.w3.org/2004/09/ws-cc-program.html.

Boley, H.; Grosof, B.; Sintek, M.; Tabet, S.; and Wagner, G. 2002. RuleML design. Document located on-line at http://www.ruleml.org/indesign.html.

Boley, H.; Dean, M.; Grosof, B.; Sintek, M.; Spencer, B.; Tabet, S.; and Wagner, G. 2004. FOL RuleML: The first-order logic web language. Document located on-line at http://www.ruleml.org/fol/.

Borgida, A. 1996. On the relative expressiveness of description logics and predicate logics. *Artif. Intell.* 82(1-2):353–367.

Brachman, R. J.; Fikes, R. E.; and Levesque, H. J. 1983. KRYPTON: Integrating terminology and assertion. In *AAAI-83*, 31–35.

Buerckert, H.-J. 1994. A resolution principle for constrained logics. *Artif. Intell.* 66(2):235–271.

Chang, C.-L., and Lee, R. 1973. *Symbolic Logic and Mechanical Theorem Proving*. Orlando, FL, USA: Academic Press, Inc.

Cohn, A. G. 1989. Taxonomic reasoning with many-sorted logics. *Artificial Intelligence Review* 3:89–128.

CycCorp, Inc. 2005. OpenCyc. OpenCyc website: http://www.opencyc.org/.

Dean, M. 2004. Semantic Web rules: Covering the use cases. In *RuleML '04*, volume 3323 of *Lecture Notes in Computer Science*, 1–5.

Dershowitz, N., and Plaisted, D. 2001. Rewriting. In Robinson, A., and Voronkov, A., eds., *Handbook of Automated Reasoning*, volume I. Elsevier Science. chapter 9, 535–610.

Donini, F. M.; Lenzerini, M.; Nardi, D.; and Schaerf, A. 1998. Al-log: integrating datalog and description logics. *J. of Intelligent and Cooperative Information Systems* 10:227–252.

Frisch, A. M. 1985. An investigation into inference with restricted quantification and a taxonomic representation. *SIGART Bull.* (91):28–31.

Golbreich, C. 2004. Combining rule and ontology reasoners for the Semantic Web. In *RuleML '04*, volume 3323 of *Lecture Notes in Computer Science*, 6–22.

Grosof, B. N.; Horrocks, I.; Volz, R.; and Decker, S. 2003. Description logic programs: Combining logic programs with description logic. In *WWW 2003*.

Haarslev, V., and Moeller, R. 2001. Racer reasoner implementation. Racer website: http://www.racer-systems.com/.

Horrocks, I., and Patel-Schneider, P. 2003. Reducing owl entailment to description logic satisfiability. In *2nd International Semantic Web Conference (ISWC-03)*.

Horrocks, I.; Patel-Schneider, P. F.; Boley, H.; Tabet, S.; Grosof, B.; and Dean., M. 2004. SWRL: A semantic web rule language combining OWL and RuleML. Document located on-line at http://www.daml.org/2004/04/swrl/.

Horrocks, I.; Sattler, U.; and Tobies, S. 1999. Practical reasoning for expressive description logics. In Ganzinger, H.; McAllester, D.; and Voronkov, A., eds., *Proceedings of the 6th International Conference on Logic for Programming and Automated Reasoning (LPAR'99)*, number 1705 in Lecture Notes in Artificial Intelligence, 161–180. Springer-Verlag.

Horrocks, I.; Sattler, U.; and Tobies, S. 2000. Practical reasoning for very expressive description logics. *Logic Journal of the IGPL* 8(3):239–264.

Hustadt, U.; Motik, B.; ; and Sattler, U. 2004. Reducing SHIQ- description logic to disjunctive datalog programs. In *KR 2004*. Los Altos, CA: Morgan Kaufmann.

Hustadt, U.; Motik, B.; and Sattler, U. 2005. A decomposition rule for decision procedures by resolution-based calculi. In *LPAR 2004*, number 3452 in Lecture Notes in Computer Science. Springer Verlag.

Levy, A. Y., and Rousset, M.-C. 1996. CARIN: A representation language combining horn rules and description logics. In *ECAI-96*, 323–327.

McCune, W. 2003. Otter: An Automated Deduction System. Technical Report ANL/MCS-TM-263, Argonne National Laboratory, Illinois.

Motik, B. 2006. KAON2 reasoner implementation. KAON2 website: http://kaon2.semanticweb.org/.

Patel-Schneider, P. F.; Hayes, P.; and Horrocks, I. 2003. Web ontology language (owl): Abstract syntax and semantics. Document located on-line at http://www.w3.org/TR/owl-semantics/.

Patel-Schneider, P. F. 2004. A proposal for a SWRL extension to first-order logic. Document located on-line at http://www.daml.org/2004/11/fol/proposal.

Rector, A.; Nowlan, W.; and Glowinski, A. 1993. Goals for concept representation in the GALEN project. In *17th Annual Symposium on Computer Applications in Medical Care*, 414–418.

Reiter, R. 1978. On closed world data bases. *Logic and databases*.

Riazanov, A., and Voronkov, A. 2002. The design and implementation of Vampire. *AI Communications* 15(2–3).

Stickel, M. J. 1985. Automated deduction by theory resolution. 4(1):333–355.

Tsarkov, D., and Horrocks, I. 2004. FaCT++ reasoner implementation. FaCT++ website: http://owl.man.ac.uk/factplusplus/.

Tsarkov, D.; Riazanov, A.; Bechhofer, S.; and Horrocks, I. 2004. Using Vampire to reason with OWL. In *Proceedings of the 3rd International Semantic Web Conference*, volume 3298 of *Lecture Notes in Computer Science*, 471–485.

Weidenbach, C. 2001. SPASS: Combining superposition, sorts and splitting. In Robinson, A., and Voronkov, A., eds., *Handbook of Automated Reasoning*, volume II. Elsevier Science. chapter 27, 1965–2013.