Reasoning in Description Logics with a Concrete Domain in the Framework of Resolution

Ullrich Hustadt[1] and **Boris Motik**[2] and **Ulrike Sattler**[3]

Abstract. In description logics, concrete domains are used to model concrete properties such as weight, name, or age, having concrete values such as integers or strings, with *built-in* predicates, such as $\leq$ or $=$. Until now, reasoning with concrete domains has been studied predominantly in the context of tableaux and automata calculi. In this paper, we present a general approach for concrete domain reasoning in the resolution framework. We apply this approach to devise an optimal decision procedure for $\mathcal{SHIQ}(\mathbf{D})$, the extension of $\mathcal{SHIQ}$ with a restricted form of concrete domains, serving as the logical underpinning of the web ontology language OWL-DL.

1 Introduction

Description logics are a family of knowledge representation formalisms closely related to first-order and modal logic. They are useful in various application fields, such as reasoning about database conceptual models, as the logical underpinning of ontology languages, for schema representation in information integration systems or for metadata management [1]. Practical applications of DLs usually require *concrete* properties with values from a fixed domain, such as integers or strings, supporting *built-in* predicates. In [2], DLs were extended with *concrete domains*, where partial functions map objects of the abstract domain to values of the concrete domain, and can be used for building complex concepts. For example, the following axiom defines minors as those humans whose age is at most 17: $Minor \equiv Human \sqcap \exists age.\leq_{17}$. In [2, 7, 15, 16, 10] decidability, computational complexity, and reasoning algorithms for different DLs with concrete domains have been studied. This research influenced the design of the Ontology Web Language (OWL), which supports a basic form of concrete domains, so-called datatypes.

Recently, resolution-based decision procedures for various DLs have been developed [14, 12]. To extend these algorithms with support for concrete domains, we present a general approach for reasoning with a concrete domain in clausal calculi whose completeness proof is based on the model generation method. Hence, our approach can be combined with most state-of-the-art calculi, such as basic superposition [4]. The main difference to (ordered) theory resolution [19, 5] is that our calculus is compatible with redundancy elimination. Moreover, we address the problem of identifying the minimal subset of clauses unsatisfiable in the theory. We next apply our approach to the decision procedure for $\mathcal{SHIQ}$ from [13], and show that adding concrete domains does not increase reasoning complexity.

Due to space limitations, we are not able to include all technical details in this paper, and refer the interested reader to [12].

[1] Department of Computer Science, University of Liverpool, UK
[2] FZI Research Center for Information Technologies at the University of Karlsruhe, Germany
[3] Department of Computer Science, University of Manchester, UK

2 Resolution with Concrete Domains

Definition of a Concrete Domain. With $\mathbf{x}$ we denote a vector of variables $x_1, \ldots, x_n$ and, for a function δ, with $\delta(\mathbf{x})$ we denote a vector of values $\delta(x_1), \ldots, \delta(x_n)$.

Definition 1. *A concrete domain $\mathbf{D}$ is a pair $(\triangle_\mathbf{D}, \Phi_\mathbf{D})$, where $\triangle_\mathbf{D}$ is a set, called the* domain *of $\mathbf{D}$, and $\Phi_\mathbf{D}$ is a finite set of* predicate names. *Each $d \in \Phi_\mathbf{D}$ is associated with an arity n and an* extension *$d^\mathbf{D} \subseteq \triangle^n_\mathbf{D}$. A concrete domain $\mathbf{D}$ is* admissible *if: (i) $\Phi_\mathbf{D}$ is closed under negation, i.e. for each $d \in \Phi_\mathbf{D}$, there exists $\overline{d} \in \Phi_\mathbf{D}$ with $\overline{d}^\mathbf{D} = \triangle_\mathbf{D} \setminus d^\mathbf{D}$, (ii) it contains a unary predicate $\top_\mathbf{D}$ interpreted as $\triangle_\mathbf{D}$, and (iii) $\mathbf{D}$-satisfiability of finite conjunctions of the form $\bigwedge_{i=1}^n d_i(\mathbf{x_i})$ is decidable. The latter is the case if there is a solution δ such that $\delta(\mathbf{x_i}) \in d_i^\mathbf{D}$, for each $1 \leq i \leq n$.*

Sometimes, we consider conjunctions over literals containing terms, rather than variables. Let $S = \{d_i(\mathbf{t_i})\}$ be a set of literals, where $\mathbf{t_i}$ is a vector of terms $t_{i1}, \ldots, t_{ik}$. With $\widehat{S}$ we denote a conjunction $C = \bigwedge d_i(\mathbf{x_i})$, obtained from S by replacing each occurrence of a term with the same variable, such that different terms are replaced with distinct variables. For two conjunctions C_1 and C_2, we write $C_1 \equiv C_2$ if they are equivalent up to variable renaming.

We use the standard definitions of first-order clausal logics (see, e.g. [6]). To separate the extension of the concrete and abstract domain, we assume a two-sorted signature with sorts a and c for the abstract and the concrete domain. Moreover, we consider only admissible concrete domains $\mathbf{D}$, so we may assume that all predicates from $\Phi_\mathbf{D}$ occur positively in clauses.

Definition 2. *Let $\mathbf{D}$ be an admissible concrete domain, N a set of clauses over a signature that includes all predicates from $\Phi_\mathbf{D}$, and I a "classic" Herbrand model of N, written $I \models N$, where concrete predicates are treated as "normal" predicates. Let $S = \{d_i(\mathbf{t_i})\}$ be the set of all concrete domain literals $d_i(\mathbf{t_i}) \in I$. Then I is a* $\mathbf{D}$-model *of N, written $I \models_\mathbf{D} N$, if, additionally, $\widehat{S}$ is $\mathbf{D}$-satisfiable.*

Concrete Domain Resolution on Ground Clauses. We now present the *ground concrete domain resolution* calculus, $\mathcal{G}^\mathbf{D}$ for short, which extends the ordered resolution calculus from [3] with a *concrete domain resolution* rule. As usual, $\mathcal{G}^\mathbf{D}$ is parameterized with a reduction ordering $\succ$ total on ground literals such that $\neg A \succ A$. A literal L is (strictly) maximal in a clause $C \vee L$ if there is no literal $L' \in C$ such that $L' \succ L$ ($L' \succeq L$). With $\succ$ we also denote the multiset extension of the literal ordering to clauses.

Definition 3. *A set $S = \{d_i(\mathbf{t_i})\}$ of positive concrete literals is a* $\mathbf{D}$-constraint *if $\widehat{S}$ is* not *$\mathbf{D}$-satisfiable. A $\mathbf{D}$-constraint S is* minimal *if $\widehat{S'}$ is $\mathbf{D}$-satisfiable, for each $S' \subsetneq S$; S is* connected *if it cannot be decomposed into two subsets without a common term.*

It is easy to see [12] that each minimal $\mathbf{D}$-constraint S is connected, or conversely, if S is not connected, it is not minimal.

We now present the rules of $\mathcal{G}^{\mathbf{D}}$. The clauses $C \vee A \vee \ldots \vee A$ and $D \vee \neg A$ are called *main premises*, whereas the clauses $C \vee A$ and $C_i \vee d_i(\mathbf{t_i})$ are called *side premises*.

Positive factoring: $\dfrac{C \vee A \vee \ldots \vee A}{C \vee A}$	(*i*) A is strictly maximal in C.
Ordered resolution: $\dfrac{C \vee A \qquad D \vee \neg A}{C \vee D}$	(*i*) A is strictly maximal in C, (*ii*) $\neg A$ is maximal in D.
Concrete domain resolution: $\dfrac{C_i \vee d_i(\mathbf{t_i}),\ 1 \leq i \leq n}{C_1 \vee \ldots \vee C_n}$	(*i*) $d_i(\mathbf{t_i})$ are strictly maximal in C_i, (*ii*) the set $S = \{d_i(\mathbf{t_i})\}$ is a minimal $\mathbf{D}$-constraint.

It is well-known that effective redundancy elimination criteria are necessary for theorem proving to be applicable in practice. A powerful *standard notion of redundancy* was introduced in [3]. We adapt this notion slightly to take into account the fact that the concrete domain resolution rule does not have a main premise. Then we show the soundness and completeness of $\mathcal{G}^{\mathbf{D}}$.

Definition 4. *Let N be a set of ground clauses. A ground clause C is* redundant *in N if there are clauses $D_i \in N$, $C \succ D_i$, such that $D_1, \ldots, D_m \models C$. A ground inference π with side premises C_i and a conclusion D is* redundant *in N if there are clauses $D_i \in N$, such that $C_1, \ldots, C_n, D_1, \ldots, D_m \models D$; if π has a main premise C, then additionally $C \succ D_i$.*

Lemma 1 (Soundness). *Let N be a set of ground clauses, I a $\mathbf{D}$-model of N, and $N' = N \cup \{C\}$, where C is the conclusion of an inference by $\mathcal{G}^{\mathbf{D}}$ with premises from N. Then I is a $\mathbf{D}$-model of N'.*

Proof. For positive factoring or ordered resolution, soundness is trivial [3]. Let C be obtained by the concrete domain resolution rule, with S being as in the rule definition. Since S is a $\mathbf{D}$-constraint, I is a $\mathbf{D}$-model of N only if some $d_i(\mathbf{t_i}) \in S$ exists such that $d_i(\mathbf{t_i}) \notin I$. Since $C_i \vee d_i(\mathbf{t_i})$ is by assumption true in I, some literal from C_i must be true in I. Since $C_i \subseteq C$, C is true in I as well. $\square$

Lemma 2 (Completeness). *Let N be a set of clauses such that each inference by $\mathcal{G}^{\mathbf{D}}$ from premises in N is redundant in N. If N does not contain the empty clause, then N is $\mathbf{D}$-satisfiable.*

Proof. (Sketch) We extend the model building method from [3] to handle the concrete domain resolution rule. For a set of ground clauses N, we define an interpretation I by induction on the clause ordering $\succ$ as follows: for some clause C, we set $I_C = \bigcup_{C \succ D} \varepsilon_D$, where $\varepsilon_D = \{A\}$ if (*i*) $D \in N$, (*ii*) D is of the form $D' \vee A$, such that A is strictly maximal in D', and (*iii*) D is false in I_D; otherwise, $\varepsilon_D = \emptyset$. Let $I = \bigcup \varepsilon_D$. A clause D such that $\varepsilon_D = \{A\}$ is called *productive*, and it said to *produce* the atom A in I.

Invariant (*): if C is false in $I_C \cup \varepsilon_C$, then C is false in I. Since C is false in I_C, all negative literals from C are false in I_C, and since $I_C \subseteq I$, all negative literals from C are false in I as well. Furthermore, since $\neg A \succ A$, any atom produced by a clause $D \succ C$ must be larger than any literal occurring in C, so no clause greater than C can produce an atom that will make C true.

We now show that, if all inferences from premises in N are redundant, I is a $\mathbf{D}$-model of N. Assume that this were not the case. The first possibility is that there is a counter-example clause from N which is false in I. This case is identical to the one from [3], so we omit the details. Roughly speaking, since $\succ$ is well-founded and total, there is a smallest counterexample C. Furthermore, there is a redundant inference by $\mathcal{G}^{\mathbf{D}}$ with the main premise C and side premises C_i which are true in I, and a conclusion D which is false in I. Since the inference is redundant, there are clauses $D_i \in N$, $C \succ D_i$, such that $D_1, \ldots, D_m, C_1, \ldots C_n \models D$. Since C is the smallest clause false in I and all D_i are smaller than C, all D_i are true in I, so D would have to be true in I, which is a contradiction.

The second possibility is that all clauses from N are true in I, but the set of concrete domain literals in I is not $\mathbf{D}$-satisfiable. Then, I contains a minimal connected $\mathbf{D}$-constraint $S = \{d_i(\mathbf{t_i})\}$. By definition of I, literals in S were produced by some $E_i = C_i \vee d_i(\mathbf{t_i}) \in N$, where E_i is false in I_{E_i}. For any i, since $d_i(\mathbf{t_i})$ is strictly maximal in C_i, C_i is false in $I_{E_i} \cup \varepsilon_{E_i}$, and by (*) C_i is false in I. Since N is saturated, the inference by concrete domain resolution resulting in $D = C_1 \vee \ldots \vee C_n$ is redundant in N. Obviously, D is false in I. Since the inference is redundant, non-redundant clauses $D_i \in N$ exist, such that $D_1, \ldots, D_m, C_1 \vee d_1(\mathbf{t_1}), \ldots, C_n \vee d_n(\mathbf{t_n}) \models D$. All D_i and $C_i \vee d_i(\mathbf{t_i})$ are by assumption true in I, implying that D is true in I, which is a contradiction. $\square$

A *fair derivation* by $\mathcal{G}^{\mathbf{D}}$ is defined as usual, as a sequence of clause sets $N_0, N_1, \ldots$ where $N_i = N_{i-1} \cup \{C\}$ and C is a consequence of an inference rule of $\mathcal{G}^{\mathbf{D}}$ from premises in N_{i-1} or $N_i = N_{i-1} \setminus \{C\}$ where C is redundant in N_{i-1}, and no inference is delayed indefinitely. With N_∞ we denote the set of clauses which never become redundant in a derivation. By Lemmata 1 and 2, and the fact that N_∞ is saturated, we get the following result:

Theorem 1. *The set of ground clauses N is $\mathbf{D}$-unsatisfiable iff the limit N_∞ of a fair derivation by $\mathcal{G}^{\mathbf{D}}$ contains the empty clause.*

Concrete Domain Resolution on General Clauses. Lifting $\mathcal{G}^{\mathbf{D}}$ to general clauses is not trivial. Consider e.g. the clause set $N = \{d_1(x_1, y_1), d_2(x_2, y_2)\}$. A connected $\mathbf{D}$-constraint can be obtained by $\{x_1 \mapsto x_2, y_1 \mapsto y_2\}$, $\{x_1 \mapsto y_2, x_2 \mapsto y_1\}$, or similar substitutions. To check all possible $\mathbf{D}$-constraints at the ground level, one has to consider all possible substitutions producing a connected $\mathbf{D}$-constraint at the non-ground level. We formalize this idea as follows.

Definition 5. *Let $S = \{d_i(\mathbf{t_i})\}$ be a multiset of positive concrete domain literals. A substitution σ is a* partitioning unifier *of S if the set $S\sigma$ is connected. Furthermore, σ is a* most general partitioning unifier *if, for any partitioning unifier θ such that $\widehat{S\sigma} \equiv \widehat{S\theta}$, a substitution η exists such that $\theta = \sigma\eta$. With $\mathsf{MGPU}(S)$ we denote the set of all most general partitioning unifiers of S.*

In [12], we show the property ($\diamondsuit$): if θ is a partitioning unifier of S, then there exists a most general partitioning unifier σ unique up to variable renaming, such that $\widehat{S\sigma} \equiv \widehat{S\theta}$. Also, in [12] we give a complete, but inefficient algorithm for computing $\mathsf{MGPU}(S)$. In Section 4 we present a specialized algorithm suitable for $\mathcal{SHIQ}(\mathbf{D})$.

We assume, as usual, that all clauses involved in an inference rule do not have variables in common. The rules of the *concrete domain resolution* calculus, $\mathcal{R}^{\mathbf{D}}$ for short, are given next, along with the adaptation of the lifting lemma.

Lemma 3 (Lifting). *Let N be a set of clauses with the set of ground instances N^G. For each ground inference π^G by $\mathcal{G}^{\mathbf{D}}$ applicable to premises $C_i^G \in N^G$, there is an inference π by $\mathcal{R}^{\mathbf{D}}$ applicable to premises $C_i \in N$, where π^G is an instance of π.*

Positive factoring: $\dfrac{C \vee A \vee B}{C\sigma \vee A\sigma}$	(*i*) $\sigma = \mathsf{MGU}(A, B)$, (*ii*) $A\sigma$ is strictly maximal in $C\sigma$.
Ordered resolution: $\dfrac{C \vee A \quad D \vee \neg B}{C\sigma \vee D\sigma}$	(*i*) $\sigma = \mathsf{MGU}(A, B)$, (*ii*) $A\sigma$ is strictly maximal in $C\sigma$, (*iii*) $\neg B\sigma$ is maximal in $D\sigma$.
Concrete domain resolution: $\dfrac{C_i \vee d_i(\mathbf{t_i}),\ 1 \leq i \leq n}{C_1\sigma \vee \ldots \vee C_n\sigma}$	(*i*) $C_i \vee d_i(\mathbf{t_i})$ are not necessarily unique, (*ii*) for $S = \{d_i(\mathbf{t_i})\}$, $\sigma \in \mathsf{MGPU}(S)$, (*iii*) $d_i(\mathbf{t_i})\sigma$ are strictly maximal in $C_i\sigma$, (*iv*) $S\sigma$ is a minimal $\mathbf{D}$-constraint.

Proof. If π^G is an inference by positive factoring or ordered resolution, the argumentation is the same as in the case of ordered resolution [3]. Let π^G be an inference by concrete domain resolution, and let $S = \{d_i(\mathbf{t_i})\}$ be the set of corresponding non-ground literals. Since $S\tau$ is a minimal $\mathbf{D}$-constraint, it is connected, so τ is obviously a partitioning unifier of S. By $(\Diamond)$, there is a most general partitioning unifier σ such that $\tau = \sigma\eta$ for some η, and $\widehat{S\sigma} \equiv \widehat{S\tau}$. Obviously, if $S\tau$ is a minimal $\mathbf{D}$-constraint, so is $S\sigma$. Furthermore, if literals from $S\tau$ are strictly maximal in C_i^G, corresponding literals from $S\sigma$ are strictly maximal in $C_i\sigma$, so $D^G = D\eta$. □

We briefly comment on the constraint (*i*) of the concrete domain resolution rule. Consider a clause $d(x, y)$. It is possible that, for the set $S = \{d(a, b), d(b, c)\}$ of ground instances of $d(x, y)$, the conjunction $\widehat{S}$ is $\mathbf{D}$-unsatisfiable. This is detected by the concrete domain resolution rule only if several "copies" of the clause $d(x, y)$ are considered simultaneously. Without any assumptions on the nature of the predicate d, there is no upper bound on the number of "copies" that should be considered simultaneously. This property of the calculus obviously leads to undecidability in the general case. To obtain a decision procedure for $\mathcal{SHIQ}(\mathbf{D})$, in Section 4 we show that the number of such "copies" to be considered is bounded.

The notion of redundancy is lifted as usual [3]: a clause C (an inference π) is redundant in N if all ground instances of C (π) are redundant in N^G. This is enough for soundness and completeness of $\mathcal{R}^{\mathbf{D}}$, shown in [12] in the usual way, as in [3].

Theorem 2. *The set of clauses N is $\mathbf{D}$-unsatisfiable if and only if the limit N_∞ of a fair derivation by $\mathcal{R}^{\mathbf{D}}$ contains the empty clause.*

Combining Concrete Domains with other Calculi. The concrete domain resolution rule may be added to any calculus whose completeness proof is based on the model generation method [3]: the usual arguments show that, if N_∞ does not contain the empty clause, then it has a model. Furthermore, the argument that I is a $\mathbf{D}$-model from Lemma 2 applies to each such calculus without modification.

3 Deciding $\mathcal{SHIQ}$ by Basic Superposition

We briefly overview our previous work on deciding $\mathcal{SHIQ}$ by basic superposition from [13], which provides the foundation for the decision procedure we present in Section 4.

$\mathcal{SHIQ}$ Description Logic. Let N_C and N_{R_a} be sets of concept and abstract role names, respectively. An abstract *role* is an abstract role name or the inverse S^- of an abstract role name S. A $\mathcal{SHIQ}$ RBox $\mathcal{R}$ is a finite set of abstract role inclusion axioms $R \sqsubseteq S$ and transitivity axioms $\mathsf{Trans}(R)$. The set of $\mathcal{SHIQ}$ *concepts* is defined as the smallest set containing N_C and, for C and D $\mathcal{SHIQ}$ concepts, R an abstract role, S an abstract *simple* role (i.e. a role without transitive subroles) and n an integer, $\neg C$, $C \sqcap D$, $C \sqcup D$, $\exists R.C$, $\forall R.C$, $\geq n\, S.C$, and $\geq n\, S.C$ are also $\mathcal{SHIQ}$ concepts. A $\mathcal{SHIQ}$ *knowledge base* KB consists of an RBox $KB_{\mathcal{R}}$, a TBox $KB_{\mathcal{T}}$ (i.e. a finite set of concept inclusion axioms $C \sqsubseteq D$ for C, D possibly complex concepts), and an ABox $KB_{\mathcal{A}}$ (i.e. a finite set of axioms $C(a)$, $R(a, b)$, and (in)equalities $a \approx b$ and $a \not\approx b$). In [13], the semantics of KB is given by translating KB into first-order logic, by means of the operator π, in the standard way.[4] The basic inference problem for $\mathcal{SHIQ}$ is checking KB satisfiability, i.e. determining whether a first-order model of $\pi(KB)$ exists. $\mathcal{ALCHIQ}$ is the logic obtained from $\mathcal{SHIQ}$ by disallowing transitivity axioms.

Basic Superposition Calculus. The basic superposition calculus, $\mathcal{BS}$ for short, has been developed to optimize theorem proving with equality [4]. A similar calculus was independently developed in [17]. The main feature of $\mathcal{BS}$ is that it prohibits superposition into terms introduced by previous unification steps. Consider the following example, where the initial clause set contains clauses (1) – (5). Clauses (1), (2) and (3) can be resolved to produce (6). In $[a] \approx [b]$, a and b have been produced through unification, which is denoted by *marking* them with $[\,]$. Clause (6) is superposed into (1) to obtain (7). Since b in (7) is marked, superposition into it is prohibited. Hence, (7) is resolved with (4) to produce (8) and then with (5) to produce the empty clause.

(1)	$A(a)$	(6)	$[a] \approx [b]$
(2)	$B(b)$	(7)	$A([b])$
(3)	$\neg A(x) \vee \neg B(y) \vee x \approx y$	(8)	$C([b])$
(4)	$\neg A(x) \vee C(x)$	(9)	□
(5)	$\neg C(b)$		

In $\mathcal{BS}$, clauses with marked terms are called *closures*. Furthermore, $\mathcal{BS}$ is parameterized with an *admissible* ordering on terms $\succ$, and a *selection function*, selecting negative literals in the closure. For the inference rules of $\mathcal{BS}$, we direct the reader to [12]. Under the notion of redundancy from [4], $\mathcal{BS}$ is sound and complete.

Deciding $\mathcal{SHIQ}$ by $\mathcal{BS}$. To decide satisfiability of a $\mathcal{SHIQ}$ knowledge base KB, we first eliminate transitivity axioms from KB by translating it into an equisatisfiable $\mathcal{ALCHIQ}$ knowledge base $\Omega(KB)$. A slight variant of this relatively standard transformation has been presented in [20].

Next, we translate $\Omega(KB)$ into an equisatisfiable set of first-order formulae $\pi(KB)$, which we then transform into an equisatisfiable set of closures $\Xi(KB)$ using structural transformation [18]. For unary coding of numbers, these steps are polynomial. By definition of π and Ξ, we can show that $\Xi(KB)$ contains only eight different so-called $\mathcal{ALCHIQ}$-closure types, presented below, where $\langle t \rangle$ means that t may or may not be marked. $\mathcal{ALCHIQ}$-closures share the following property (♠): the depth of a functional term in a closure is limited to two, and the number of variables in a closure is limited by the maximal number n occurring in number restrictions $\geq n\, R.C$ and $\leq n\, R.C$. The maximal number of different $\mathcal{ALCHIQ}$-closures is at most exponential in $|KB|$, for unary coding of numbers.

With $\mathcal{BS}_{DL}$ we denote $\mathcal{BS}$ parameterized with a specific *lexicographic path ordering* [3] and a selection function which selects all negative binary literals. We saturate $\Xi(KB)$ using $\mathcal{BS}_{DL}$ with eager elimination of redundancy. The chosen parameters make $\mathcal{BS}_{DL}$ perform inferences only on literals containing a term of maximal

[4] The set-theoretic semantics of $\mathcal{SHIQ}$ from [11] coincides with the one provided by the translation operator π.

1	$\neg R(x,y) \vee \mathsf{Inv}(R)(y,x)$
2	$\neg R(x,y) \vee S(x,y)$
3	$\mathbf{P^f}(x) \vee R(x, \langle f(x) \rangle)$
4	$\mathbf{P^f}(x) \vee R([f(x)], x)$
5	$\mathbf{P_1}(x) \vee \mathbf{P_2}(\langle \mathbf{f}(x) \rangle) \vee \bigvee \langle f_i(x) \rangle \approx / \not\approx \langle f_j(x) \rangle$
6	$\mathbf{P_1}(x) \vee \mathbf{P_2}([g(x)]) \vee \mathbf{P_3}(\langle \mathbf{f}([g(x)]) \rangle) \vee \bigvee \langle t_i \rangle \approx / \not\approx \langle t_j \rangle$ where t_i and t_j are either $f([g(x)])$ or x
7	$\mathbf{P_1}(x) \vee \bigvee \neg R(x, y_i) \vee \mathbf{P_2}(\mathbf{y}) \vee \bigvee y_i \approx y_j$
8	$\mathbf{R}(\langle \mathbf{a} \rangle, \langle \mathbf{b} \rangle) \vee \mathbf{P}(\langle \mathbf{t} \rangle) \vee \bigvee \langle t_i \rangle \approx / \not\approx \langle t_j \rangle$ where t, t_i and t_j are either some constant b or $f_i([a])$

depth. This, combined with the fact that basic superposition prohibits inferences into terms introduced by previous unification steps, implies another key property (♣): applying a $\mathcal{BS}_{DL}$ inference rule to $\mathcal{ALCHIQ}$-closures results in an $\mathcal{ALCHIQ}$-closure or is redundant.

Properties (♠) and (♣) imply that any $\mathcal{BS}_{DL}$ derivation from $\Xi(KB)$ terminates: in the worst case, all possible $\mathcal{ALCHIQ}$ closures are computed. The number of such closures is exponential in $|KB|$, and this set can be computed in time exponential in $|KB|$. Finally, since $\mathcal{BS}_{DL}$ is sound and complete, saturation of $\Xi(KB)$ by $\mathcal{BS}_{DL}$ decides satisfiability of KB.

Unfortunately, $\mathcal{BS}_{DL}$ decides only the $\mathcal{ALCHIQ}^-$ fragment of $\mathcal{ALCHIQ}$, where number restrictions are allowed only on roles without subroles. E.g., a superposition from $[f(g(x))] \approx [h(g(x))]$ into $R(x, f(x))$ results in $C = R([g(x)], [h(g(x))])$. Such a closure is not redundant in the general case and is not an $\mathcal{ALCHIQ}$-closure. In further inferences, C can produce closures with functional terms of depth more than two, so termination of saturation is not guaranteed. We solve this problem by a technique called *decomposition*, which does not affect completeness of $\mathcal{BS}_{DL}$. It replaces certain closures with simpler ones, which do not cause termination problems. E.g., for C, decomposition introduces a new predicate $Q_{R,h}$, and replaces C with $Q_{R,h}([g(x)])$ and $\neg Q_{R,h}(x) \vee R(x, [h(x)])$. These two closures are $\mathcal{ALCHIQ}$-closures, so the properties (♠) and (♣) hold. The predicate $Q_{R,h}$ is unique for a pair of role and function symbols R and h. Since there are only finitely many such pairs, only a finite number of predicates is introduced, so saturation by $\mathcal{BS}_{DL}$ and eager decomposition terminates.

4 Deciding $\mathcal{SHIQ}(\mathbf{D})$ Description Logic

We now apply the concrete domain resolution to obtain a decision procedure for $\mathcal{SHIQ}(\mathbf{D})$.

$\mathcal{SHIQ}(\mathbf{D})$ Description Logic. Apart from abstract roles, we assume a disjoint set of *concrete* roles N_{R_c}. For $\mathbf{D}$ an admissible concrete domain, $\mathcal{SHIQ}(\mathbf{D})$ is the extension of $\mathcal{SHIQ}$ with concepts of the form $\exists T_1, \ldots, T_m.d$, $\forall T_1, \ldots, T_m.d$, $\leq n\, T$, and $\geq n\, T$, for $T_{(i)} \in N_{R_c}$ and $d \in \Phi_{\mathbf{D}}$. Role inclusion axioms are not allowed between abstract and concrete roles, and inverse concrete roles are not allowed. The new concepts are translated into FOL as follows:

$$\pi_y(\bowtie n\, T, x) = \exists^{\bowtie n} y : T(x,y), \text{ for } \bowtie \in \{\leq, \geq\}$$
$$\pi_y(\forall T_1, \ldots, T_m.d, x) = \forall y_i : \bigwedge T_i(x, y_i) \rightarrow d(y_1, \ldots, y_m)$$
$$\pi_y(\exists T_1, \ldots, T_m.d, x) = \exists y_i : \bigwedge T_i(x, y_i) \wedge d(y_1, \ldots, y_m)$$

Closures with Concrete Predicates. With $\mathcal{BS}^{\mathbf{D}}_{DL}$ we denote the $\mathcal{BS}_{DL}$ calculus extended with the concrete domain resolution rule and eager decomposition. Transitivity elimination applies to $\mathcal{SHIQ}(\mathbf{D})$ as well [13], so in the rest we focus on deciding satisfiability of an $\mathcal{ALCHIQ}$ knowledge base KB. Besides $\mathcal{ALCHIQ}$-closures, $\Xi(KB)$ can additionally contain closures of the form listed below. We call such closures $\mathcal{ALCHIQ}(\mathbf{D})$-closures, and show that the analogue of (♣) holds for them as well.

8	closure can additionally contain $\ldots \vee \mathbf{T}(\langle \mathbf{a} \rangle, \langle \mathbf{b} \rangle) \vee \mathbf{d}(\langle \mathbf{t_1} \rangle, \ldots, \langle \mathbf{t_n} \rangle) \vee \bigvee \langle t_i \rangle \approx / \not\approx \langle t_j \rangle$ where t_i and t_j are either a constant b or a term $f_i([a])$
9	$\neg T(x,y) \vee U(x,y)$
10	$\mathbf{P^f}(x) \vee T(x, \langle f(x) \rangle)$
11	$\mathbf{P}(x) \vee \mathbf{d}(\langle \mathbf{f_1}(x) \rangle, \ldots, \langle \mathbf{f_n}(x) \rangle) \vee \bigvee \langle f_i(x) \rangle \approx / \not\approx \langle f_j(x) \rangle$
12	$\mathbf{P}(x) \vee \bigvee \neg T(x, y_i) \vee \bigvee y_i \approx y_j$
13	$\mathbf{P}(x) \vee \bigvee \neg T_i(x, y_i) \vee d(y_1, \ldots, y_n)$

Lemma 4. *Let $\Xi(KB) = N_0, \ldots, N_i \cup \{C\}$ be a $\mathcal{BS}^{\mathbf{D}}_{DL}$-derivation, where C is the conclusion derived from premises in N_i. Then C is either an $\mathcal{ALCHIQ}(\mathbf{D})$-closure or it is redundant in N_i.*

Proof. (Sketch) $\mathcal{ALCHIQ}(\mathbf{D})$-closures are almost identical in structure to $\mathcal{ALCHIQ}$-closures. Hence, it is easy to see that this property holds for all inferences by $\mathcal{BS}_{DL}$ as before. The new aspect is the application of the concrete domain resolution rule. Consider n side premises $C_i \vee d_i(\langle \mathbf{t_i} \rangle)$, each possibly having a free variable x_i, and let $S = \{d_i(\mathbf{t_i})\}$. For any most general partitioning unifier σ of S, $S\sigma$ is connected, so there are two possibilities. If all side premises are of type 11, σ is of the form $\{x_2 \mapsto x_1, \ldots, x_n \mapsto x_1\}$, so the result is of type 11. If there is a side premise of type 8, since $S\sigma$ is connected, it may not contain a variable, so σ is of the form $\{x_1 \mapsto c_1, \ldots, x_n \mapsto c_n\}$, and the result is of type 8. □

Termination and Complexity Analysis. We define the size of the new constructs as follows: $|\exists T_1, \ldots, T_m.d| = |\forall T_1, \ldots, T_m.d| = 2 + m$. Let m denote the maximal arity of a concrete domain predicate, d the number of concrete domain predicates, and f the number of function symbols in the signature of $\Xi(KB)$. As in [12], f is linear in $|KB|$ for unary coding of numbers, since in skolemizing each $\exists T_1, \ldots, T_m.d$, we introduce m new function symbols. $\mathcal{ALCHIQ}(\mathbf{D})$-closures additionally contain literals of the form $d(\langle f_1(x) \rangle, \ldots, \langle f_m(x) \rangle)$. The maximal non-ground closure will contain all such literals, whose number is bounded by $d(2f)^m$ (the factor 2 allows each functional term to be marked or not). Hence, there are at most $2^{d(2f)^m}$ closures containing different combinations of literals. Thus we have to decide whether m is a parameter of the input, or generally bounded. In the former case, the number of closures is doubly-exponential in the size of the input, whereas it is "only" exponential in the latter. Note that this latter choice is justifiable from a practical view point: it is hard to imagine a concrete domain with predicates of unbounded arity as well as a sequence of concrete domains with increasing arities, but where the arity of each domain is bounded. Under this assumption, m is a constant, the maximal length of a closure is polynomial in $|KB|$, and the number of closures is exponential in $|KB|$. The following lemma is the last step in determining the complexity of the decision algorithm.

Lemma 5. *The maximal number of side premises participating in a concrete domain resolution rule in a $\mathcal{BS}^{\mathbf{D}}_{DL}$-derivation from Lemma 4 is at most polynomial in $|KB|$, assuming a limit on the arity of concrete predicates.*

Proof. In an application of the concrete domain resolution rule, $S\sigma$ does not contain repeated literals; otherwise $\widehat{S\sigma}$ contains repeated conjuncts and is not minimal. Hence, in the worst case, each $d_i(\mathbf{t_i})\sigma$ should be unique in $S\sigma$. Let i be the number of individual names in $\Xi(KB)$. There are at most $\ell_{ng} = df^m$ distinct non-ground concrete domain literals, and at most $\ell_g = d(i + if)^m$ distinct ground concrete domain literals. Assuming a bound on m and unary coding of numbers, both ℓ_{ng} and ℓ_g are polynomial in $|KB|$.

If all side premises are non-ground, since $S\sigma$ should be connected, σ must be $\{x_2 \mapsto x_1, \ldots, x_n \mapsto x_1\}$. Hence, $S\sigma$ contains only one variable x_1, so the number of distinct literals in $S\sigma$ is bounded by ℓ_{ng}. The number of non-ground side premises is also bounded by ℓ_{ng}, and all side premises are unique up to variable renaming.

If there is a ground side premise, since $S\sigma$ should be connected, σ is $\{x_1 \mapsto c_1, \ldots, x_n \mapsto c_n\}$. All literals in $S\sigma$ are ground, and the number of distinct such literals is bounded by ℓ_g. The number of side premises to be considered in this case is bounded by ℓ_g. $\square$

Theorem 3. *Let KB be an $\mathcal{ALCHIQ}(\mathbf{D})$ knowledge base, defined over an admissible concrete domain $\mathbf{D}$, for which $\mathbf{D}$-satisfiability of finite conjunctions over $\Phi_\mathbf{D}$ can be decided in deterministic exponential time. Then saturation of $\Xi(KB)$ by $\mathcal{BS}_{DL}^{\mathbf{D}}$ with eager elimination of redundancy decides $\mathbf{D}$-satisfiability of KB and runs in time exponential in $|KB|$, for unary coding of numbers and assuming a bound on the arity of concrete predicates.*

Proof. Since property (♣) holds for $\mathcal{ALCHIQ}(\mathbf{D})$-closures, the application of $\mathcal{BS}_{DL}^{\mathbf{D}}$ inferences to $\mathcal{ALCHIQ}(\mathbf{D})$-closures can be performed in exponential time. In addition, we show that applying the concrete domain resolution rule does not take "too long".

To apply the concrete domain resolution rule to a set of closures N, one selects a subset $N' \subseteq N$. By Lemma 5, $\ell = |N'|$ is polynomial in $|KB|$, and N' contains at most ℓ variables. If all closures from N' are non-ground, there is exactly one σ which unifies all of these variables. If there is at least one ground closure, then each one of ℓ variables can be assigned to one of i individuals, thus giving i^ℓ combinations, which is exponential in $|KB|$. Closures in N' are chosen from the maximal set of all closures which is exponential in $|KB|$, and since $|N'|$ is polynomial in $|KB|$, the number of different sets N' is exponential in $|KB|$. Hence, the maximal number of $\mathbf{D}$-constraints that needs to be examined by the concrete domain resolution rule is exponential in $|KB|$, where the length of each $\mathbf{D}$-constraint is polynomial in $|KB|$. Since satisfiability of each $\mathbf{D}$-constraint can be checked in deterministic exponential time, all necessary inferences by $\mathcal{BS}_{DL}^{\mathbf{D}}$ can be performed in exponential time. $\square$

Lemma 5 demonstrates how to implement the concrete domain resolution rule. For non-ground closures, the most general partitioning unifier is $\sigma = \{x_2 \mapsto x_1, \ldots, x_n \mapsto x_1\}$. Hence, one should consider only closures with maximal literals unique up to variable renaming, so several "copies" of a closure need not be considered.

For a concrete domain resolution inference where at least one closure is ground, one first chooses a connected set of ground closures Δ. Let Δ_c be the set of constants, and Δ_f the set of function symbols occurring in a ground functional term $f(c)$ in a closure from Δ. The most general partitioning unifier σ may contain only mappings of the form $x_i \mapsto c$, where $c \in \Delta_c$. Hence, one selects the set Δ_{ng} of non-ground closures containing a functional term in Δ_f. To apply the concrete domain resolution rule, one considers at most $|\Delta_c|$ "copies" of a closure from Δ_{ng}, since σ might assign a distinct value from Δ_c to each such closure.

5 Conclusion

In this paper we have shown how a wide range of resolution-based decision procedures for first order logic can be extended with admissible concrete domains. Our approach is applicable to any clausal calculus whose completeness proof is based on the model generation method. We have exemplified the usefulness of our approach by extending the basic superposition decision procedure for $\mathcal{SHIQ}$ with concrete domains, thus providing an optimal decision procedure for $\mathcal{SHIQ}(\mathbf{D})$, a logic lying at the heart of the Semantic Web ontology language OWL-DL. These algorithms can be used to extend the reduction of $\mathcal{SHIQ}$ knowledge bases to disjunctive datalog presented in [13] to handle $\mathcal{SHIQ}(\mathbf{D})$ knowledge bases.

Part of our future theoretical work will be the extension of our decision procedure to handle nominals. This is a non-trivial task since, in the presence of nominals, we will have to handle novel forms of closures, and thus have to find new termination mechanisms. On the practical side, we are currently implementing a new description logic reasoner using the calculus presented here and the above mentioned reduction to disjunctive datalog. Once the reasoner is complete, we shall compare its performance with state-of-the-art systems such as FaCT and Racer [9, 8] to validate our approach in practice.

REFERENCES

[1] *The Description Logic Handbook*, eds., F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider, Cambridge University Press, January 2003.

[2] F. Baader and P. Hanschke, 'A Scheme for Integrating Concrete Domains into Concept Languages', in *Proc. IJCAI-91*, pp. 452–457, Sydney, Australia, (1991).

[3] L. Bachmair and H. Ganzinger, 'Resolution Theorem Proving', in *Handbook of Automated Reasoning*, eds., A. Robinson and A. Voronkov, volume I, chapter 2, 19–99, Elsevier Science, (2001).

[4] L. Bachmair, H. Ganzinger, C. Lynch, and W. Snyder, 'Basic Paramodulation', *Information and Computation*, **121**(2), 172–192, (1995).

[5] P. Baumgartner, 'An Ordered Theory Resolution Calculus', in *Proc. LPAR-92*, pp. 119–130, St. Petersburg, Russia, (1992). Springer.

[6] M. Fitting, *First-Order Logic and Automated Theorem Proving, 2nd Edition*, Springer, 1996.

[7] V. Haarslev and R. Möller, 'Expressive ABox Reasoning with Number Restrictions, Role Hierarchies, and Transitively Closed Roles', in *Proc. KR-2000*, pp. 273–284, San Francisco, (2000). Morgan Kaufmann.

[8] V. Haarslev and R. Möller, 'High Performance Reasoning with Very Large Knowledge Bases: A Practical Case Study', in *Proc. IJCAI-01*, pp. 161–168, Seattle, USA, (August 2001). Morgan Kaufmann.

[9] I. Horrocks, 'Using an Expressive Description Logic: FaCT or Fiction?', in *Proc. KR-98*, pp. 636–647, Italy, (1998). Morgan Kaufmann.

[10] I. Horrocks and U. Sattler, 'Ontology Reasoning in the $\mathcal{SHOQ}(\mathbf{D})$ Description Logic', in *Proc. IJCAI-01*, pp. 199–204. Morgan Kaufmann, (2001).

[11] I. Horrocks, U. Sattler, and S. Tobies, 'Practical Reasoning for Very Expressive Description Logics', *Logic Journal of the IGPL*, **8**(3), 239–263, (2000).

[12] U. Hustadt, B. Motik, and U. Sattler, 'Reasoning for Description Logics around $\mathcal{SHIQ}$ in a Resolution Framework', Technical Report 3-8-04/04, FZI, Germany, (2004). http://www.fzi.de/wim/publikationen.php?id=1172.

[13] U. Hustadt, B. Motik, and U. Sattler, 'Reducing $\mathcal{SHIQ}^-$ Description Logic to Disjunctive Datalog Programs', in *Proc. KR-2004*. Morgan Kaufmann, (June 2004).

[14] U. Hustadt and R. A. Schmidt, 'Issues of Decidability for Description Logics in the Framework of Resolution', in *Automated Deduction in Classical and Non-Classical Logics*, volume 1761 of *LNAI*, pp. 192–206. Springer, (1999).

[15] C. Lutz, *The Complexity of Reasoning with Concrete Domains*, Ph.D. dissertation, Teaching and Research Area for Theoretical Computer Science, RWTH Aachen, 2002.

[16] C. Lutz, 'Description Logics with Concrete Domains—A Survey', in *AiML*, volume 4. King's College Publications, (2003).

[17] R. Nieuwenhuis and A. Rubio, 'Theorem Proving with Ordering and Equality Constrained Clauses', *Journal of Logic and Computation*, **19**(4), 312–351, (April 1995).

[18] A. Nonnengart and C. Weidenbach, 'Computing Small Clause Normal Forms', in *Handbook of Automated Reasoning*, eds., A. Robinson and A. Voronkov, volume I, chapter 6, 335–367, Elsevier Science, (2001).

[19] M. E. Stickel, 'Automated Deduction by Theory Resolution', *Journal of Automated Reasoning*, **1**(4), 333–355, (1985).

[20] S. Tobies, *Complexity Results and Practical Algorithms for Logics in Knowledge Representation*, Ph.D. dissertation, RWTH Aachen, 2001.