Deciding Expressive Description Logics in the Framework of Resolution

Ullrich Hustadt [a] Boris Motik [b] Ulrike Sattler [c]

[a] *Department of Computer Science, University of Liverpool, Liverpool, UK*

[b] *Computing Laboratory, University of Oxford, Oxford, UK*

[c] *School of Computer Science, University of Manchester, Manchester, UK*

Abstract

We present a decision procedure for the description logic $\mathcal{SHIQ}$ based on the basic superposition calculus, and show that it runs in exponential time for unary coding of numbers. To derive our algorithm, we extend basic superposition with a *decomposition* inference rule, which transforms conclusions of certain inferences into equivalent, but simpler clauses. This rule can be used for general first-order theorem proving with any resolution-based calculus compatible with the standard notion of redundancy.

Key words: Description Logics, Resolution Decision Procedure, Basic Superposition

1 Introduction

Description logics (DLs) are a family of knowledge representation formalisms with a well-defined model-theoretic semantics and well-understood reasoning problems [1]. Most DLs can be viewed as decidable fragments of first-order logic. The basic building blocks of DL ontologies are *concepts*, which can be understood as unary predicates, and *roles*, which can be understood as binary predicates. Using concept constructors, atomic concepts and roles can be combined into *complex concepts*. The *terminological* part of a DL knowledge base KB, called TBox, consists of *concept inclusion* axioms of the form $C \sqsubseteq D$. For example, the TBox axiom $Person \sqsubseteq \exists hasFather.(Person \sqcap Male)$ specifies that each person must have a father who is a male person. Additionally, a TBox may contain *role inclusions* such as $hasFather \sqsubseteq hasParent$. The *assertional* part of a knowledge base, called ABox, contains axioms that (partially) specify the actual state of the world. For example, the axiom $Person(Peter)$

states that *Peter* is a *Person*, and the axiom $hasFather(Peter, Bob)$ states that *Peter* has a father *Bob*. A standard reasoning service for DLs is deciding *subsumption* between two concepts with respect to a TBox—that is, checking whether, in all models of the TBox, an instance of the first concept is also an instance of the second one. Other standard reasoning services are deciding whether a concept is satisfiable w.r.t. a knowledge base, or deciding whether an individual from an ABox is an instance of a concept in all models of the knowledge base. Description logics have found their application in various fields of computer science. For example, they are used in information integration [1, Chapter 16] for schema modeling, and the DLs $\mathcal{SHIQ}$ and $\mathcal{SHOIQ}$ form the logical underpinning of the Web Ontology Language (OWL) [2]. These DLs provide a rich set of concept constructors, such as Boolean operators on concepts, existential and universal value restrictions, inverse and transitive roles, and qualified number restrictions—concepts of the form $\geq n\, S.C$ and $\leq n\, S.C$ that restrict the number of S-successors in C that an object can have to at least (at most) n. For example, $Person \sqcap \leq 2\, hasChild.Man$ describes persons having at most two sons.

Various DL reasoning systems have been implemented and applied to practical problems.[1] These systems are usually based on tableau algorithms [3], which perform quite well mainly due to sophisticated heuristics [4,5]. $\mathcal{SHIQ}$ and $\mathcal{SHOIQ}$ are ExpTime- and NExpTime-complete, respectively [6], whereas most tableau algorithms run in 2NExpTime. Hence, tableau algorithms for these logics are not worst-case optimal. An exception is the worst-case optimal algorithm for $\mathcal{ALC}$ by Donini and Massacci [7]; however, this algorithm is very complex, it supports only a very simple logic, and, to the best of our knowledge, it has not been implemented. Hence, finding a practical, worst-case optimal algorithm for expressive DLs remains an open problem.

Resolution-based calculi are nowadays among the most widely used calculi for first-order theorem proving. On the theoretical side, the original resolution calculus has been significantly refined. On the practical side, implementation techniques for efficient theorem provers have been devised and applied in practice; an overview of such techniques is given in [8]. Resolution-based calculi can be used to reason about DL knowledge bases, as most DLs can be translated into a first-order theory. This approach has been implemented and tested in MSPASS—an implementation of decision procedures for a variety of DLs based on the resolution theorem prover SPASS [9].

Following this translation-based approach, we present in this paper a practical, worst-case optimal decision procedure for the DL $\mathcal{SHIQ}$ based on *basic superposition*—a refutational theorem proving method for first-order logic with equality. Equality is used in the translation of DL number restrictions

[1] See `http://www.cs.man.ac.uk/~sattler/reasoners.html`.

into first-order logic. For example, the concept $Person \sqcap \leq 2\, hasChild.Man$ is translated into first-order logic as

$$Person(x) \wedge \forall y_1, y_2, y_3 : \bigwedge_{1 \leq i \leq 3} [hasChild(x, y_i) \wedge Man(y_i)] \rightarrow \bigvee_{1 \leq i < j \leq 3} y_i \approx y_j.$$

To the best of our knowledge, this is the first decision procedure based on basic superposition. While similar procedures employ subsumption to restrict the number of variables in a clause and thus limit the clause *length*, our algorithm employs subsumption to restrict the clause *depth*. Interestingly, basic superposition alone is not restrictive enough to yield a decision procedure for $\mathcal{SHIQ}$. Therefore, we extend basic superposition with *decomposition*—a new inference rule that can be used to simplify the syntactic structure of certain conclusions. Decomposition is a very general rule with applications not limited to DL reasoning. We show that it is sound and complete when combined with basic superposition; however, we argue that decomposition can be combined with any calculus compatible with the standard notion of redundancy [10]. Our combination with basic superposition is interesting because of a nonstandard approach to lifting.

This procedure is also important because it is used as the first step in an algorithm for reducing a $\mathcal{SHIQ}$ knowledge base into a disjunctive datalog program [11]. The latter algorithm is particularly suitable for reasoning with knowledge bases containing large ABoxes as it allows one to reuse optimization techniques known from the field of deductive databases.

To verify the practicability of our algorithm, we have implemented it in a new DL reasoner KAON2. [2] Our initial experiments with practical problems have revealed significant potential for optimization. By reducing the number of clauses produced after translation into first-order logic, one can significantly reduce the search space of the theorem prover. We summarize all relevant optimizations in this paper. After incorporating the optimizations into our system, we conducted experiments whose results are summarized in [12]. The results show that, for ontologies with large ABoxes, the reduction algorithm exhibits performance improvements for query answering of one or more orders of magnitude. Furthermore, for TBox reasoning, the resolution-based algorithm performs similarly to existing reasoners. Hence, the algorithm from this paper can be seen as another stepping stone towards DL reasoners that can be used in realistic applications.

This paper describes an extended version of the results presented in [11,13].

[2] `http://kaon2.semanticweb.org/`

2 Preliminaries

2.1 Description Logic $\mathcal{SHIQ}$

The syntax of $\mathcal{SHIQ}$ is given by the following definition [3].

Definition 1 *For N_R a set of* atomic roles*, the set of* roles *is defined as $N_R \cup \{R^- | R \in N_R\}$. For $R \in N_R$, let $\mathsf{Inv}(R) = R^-$ and $\mathsf{Inv}(R^-) = R$. For R and S roles, a* role inclusion axiom *is a statement of the form $R \sqsubseteq S$, and a* transitivity axiom *is a statement of the form $\mathsf{Trans}(R)$. An* RBox *$KB_\mathcal{R}$ over N_R is a finite set of role inclusion and transitivity axioms. Let $\sqsubseteq^*$ be the reflexive-transitive closure of $\{R \sqsubseteq S, \mathsf{Inv}(R) \sqsubseteq \mathsf{Inv}(S) \mid R \sqsubseteq S \in KB_\mathcal{R}\}$. A role R is* transitive *if $\mathsf{Trans}(S) \in KB_\mathcal{R}$ or $\mathsf{Trans}(\mathsf{Inv}(S)) \in KB_\mathcal{R}$ for some S with $S \sqsubseteq^* R$ and $R \sqsubseteq^* S$. A role R is* simple *if there is no role S such that $S \sqsubseteq^* R$ and S is transitive; R is* complex *if it is not simple.*

Let N_C be a set of atomic *concepts. The set of* concepts *over N_C and $KB_\mathcal{R}$ is the smallest set such that $\top$, $\bot$, A, $\neg C$, $C \sqcap D$, $C \sqcup D$, $\exists R.C$, $\forall R.C$, $\leq n\, S.C$, and $\geq n\, S.C$ are concepts, for an atomic concept A, concepts C and D, a role R, a simple role S, and a nonnegative integer n. Possibly negated atomic concepts are called* literal concepts*. A concept C is a* subconcept *of a concept D if C syntactically occurs in D.*

A concept inclusion axiom *is a statement of the form $C \sqsubseteq D$, for C and D concepts. A* TBox *$KB_\mathcal{T}$ over N_C and $KB_\mathcal{R}$ is a finite set of concept inclusion axioms.*

Let N_I be a set of individuals*. An* assertion *is a statement of the form $C(a)$, $R(a, b)$, $\neg S(a, b)$, and an* (in)equality axiom *is a statement of the form $a \approx b$, for C a concept, R a role, S a simple role, and a and b individuals. An* ABox *$KB_\mathcal{A}$ is a finite set of assertions and (in)equality axioms.*

A $\mathcal{SHIQ}$ knowledge base *KB is a triple $(KB_\mathcal{R}, KB_\mathcal{T}, KB_\mathcal{A})$. Furthermore, $|KB|$ is the* size *of KB with numbers in number restrictions coded in unary.*

The semantics of $\mathcal{SHIQ}$ is defined by translating KB into a first-order formula $\pi(KB)$, as shown in Table 1. This translation maps each atomic concept into a unary predicate, each atomic role into a binary predicate, and each individual into a constant. Each inverse role R^- is mapped into a binary predicate with the name R^-, and $\pi(R)$ axiomatizes the required relationship between R and R^-. We typically do not distinguish an atomic concept, role, or an individual from the first-order (predicate or constant) symbol used to represent it. The translation of number restrictions uses counting quantifiers $\exists^{\geq n}$ and $\exists^{\leq n}$. It is well known that these can be represented in first-order logic using standard

Table 1
Semantics of $\mathcal{SHIQ}$ by Mapping to FOL

Translating Concepts to FOL	
$\pi_x(\top) = \top$	$\pi_y(\top) = \top$
$\pi_x(\bot) = \bot$	$\pi_y(\bot) = \bot$
$\pi_x(A) = A(x)$	$\pi_y(A) = A(y)$
$\pi_x(\neg C) = \neg\pi_x(C)$	$\pi_y(\neg C) = \neg\pi_y(C)$
$\pi_x(C \sqcap D) = \pi_x(C) \wedge \pi_x(D)$	$\pi_y(C \sqcap D) = \pi_y(C) \wedge \pi_y(D)$
$\pi_x(C \sqcup D) = \pi_x(C) \vee \pi_x(D)$	$\pi_y(C \sqcup D) = \pi_y(C) \vee \pi_y(D)$
$\pi_x(\exists R.C) = \exists y : [R(x,y) \wedge \pi_y(C)]$	$\pi_y(\exists R.C) = \exists x : [R(y,x) \wedge \pi_x(C)]$
$\pi_x(\forall R.C) = \forall y : [R(x,y) \rightarrow \pi_y(C)]$	$\pi_y(\forall R.C) = \forall x : [R(y,x) \rightarrow \pi_x(C)]$
$\pi_x(\geq n\, S.C) = \exists^{\geq n} y : [S(x,y) \wedge \pi_y(C)]$	$\pi_y(\geq n\, S.C) = \exists^{\geq n} x : [S(y,x) \wedge \pi_x(C)]$
$\pi_x(\leq n\, S.C) = \exists^{\leq n} y : [S(x,y) \wedge \pi_y(C)]$	$\pi_y(\leq n\, S.C) = \exists^{\leq n} x : [S(y,x) \wedge \pi_x(C)]$
Translating Axioms to FOL	
$\pi(C \sqsubseteq D) = \forall x : [\pi_x(C) \rightarrow \pi_x(D)]$	
$\pi(R \sqsubseteq S) = \forall x, y : [R(x,y) \rightarrow S(x,y)]$	
$\pi(\mathsf{Trans}(R)) = \forall x, y, z : [R(x,y) \wedge R(y,z) \rightarrow R(x,z)]$	
$\pi(C(a)) = \pi_x(C)\{x \mapsto a\}$	
$\pi(R(a,b)) = R(a,b)$	
$\pi(a \approx b) = a \approx b$	
$\pi(a \not\approx b) = a \not\approx b$	
Translating KB to FOL	
$\pi(R) = \forall x, y : [R(x,y) \leftrightarrow R^-(y,x)]$ for each atomic role R	
$\pi(KB) = \bigwedge_{R \in N_R} \pi(R) \wedge \bigwedge_{\alpha \in KB_\mathcal{T} \cup KB_\mathcal{R} \cup KB_\mathcal{A}} \pi(\alpha)$	

quantifiers and equality as follows, for $\mathbf{y}$ a vector of variables:

$$(1) \qquad \exists^{\geq n} x : \varphi(x, \mathbf{y}) = \exists x_1, \ldots, x_n : \left[\bigwedge_{i=1}^{n} \varphi(x_i, \mathbf{y}) \wedge \bigwedge_{1 \leq i < j \leq n} x_i \not\approx x_j \right]$$

$$(2) \qquad \exists^{\leq n} x : \varphi(x, \mathbf{y}) = \forall x_1, \ldots, x_{n+1} : \left[\bigwedge_{i=1}^{n+1} \varphi(x_i, \mathbf{y}) \rightarrow \bigvee_{1 \leq i < j \leq n+1} x_i \approx x_j \right]$$

Definition 2 *The semantics of a $\mathcal{SHIQ}$ knowledge base KB is defined as the formula $\pi(KB)$ of first-order logic with equality and counting quantifiers, where π is defined in Table 1. KB is* satisfiable *if and only if $\pi(KB)$ is satisfiable.*

All other standard inference problems can be reduced to satisfiability [1, Chapter 2]. We now define several restrictions of $\mathcal{SHIQ}$.

Definition 3 *For a knowledge base KB, a role R is called a* leaf role *if no role S different from R exists such that $S \sqsubseteq^* R$. The description logic $\mathcal{SHIQ}^-$ is the fragment of $\mathcal{SHIQ}$ obtained by requiring the role R in number restrictions $\leq n\, R.C$ and $\geq n\, R.C$ to be a leaf role. The description logic $\mathcal{ALCHIQ}$ ($\mathcal{ALCHIQ}^-$) is the fragment of $\mathcal{SHIQ}$ ($\mathcal{SHIQ}^-$) that does not allow for transitivity axioms.*

For a concept C, with $\mathsf{nnf}(C)$ we denote the *negation-normal form* of C—the concept equivalent to C in which negation occurs only in front of atomic concepts. It is well-known that $\mathsf{nnf}(C)$ can be computed from C in polynomial time by repeated application of de Morgan laws [1].

$\mathcal{SHIQ}$ is a very expressive DL. In particular, it can express most features of conceptual data models [14], such as the entity-relationship model [15] or UML.[3] The following is a simple $\mathcal{SHIQ}$ knowledge base that demonstrates the expressivity of $\mathcal{SHIQ}$. Axiom (3) states that every person must have exactly one social security number. Axioms (4)–(5) define *Person* as a partition of *Man* and *Woman*. Finally, (6) defines the range of *hasSSN*—it states that *hasSSN* can point only to social security numbers.

$$\text{(3)} \qquad \mathit{Person} \sqsubseteq \exists \mathit{hasSSN}.\mathit{SSN} \sqcap \leq 1\, \mathit{hasSSN}.\top$$
$$\text{(4)} \qquad \mathit{Person} \sqsubseteq \mathit{Man} \sqcup \mathit{Woman}$$
$$\text{(5)} \qquad \mathit{Man} \sqcap \mathit{Woman} \sqsubseteq \bot$$
$$\text{(6)} \qquad \exists \mathit{hasSSN}^-.\top \sqsubseteq \mathit{SSN}$$

2.2 Basic Superposition Calculus

The basic superposition calculus [16,17] was developed to optimize theorem proving with equality. We use the standard definitions of terms, atoms, and literals. For convenience, we assume function symbols to have a nonzero arity, and constants to be of zero arity. We write positive equality literals as $s \approx t$, and negative equality literals as $s \not\approx t$. A *clause* is a finite multiset of literals. A *position* p is a finite sequence of integers that describes the "address" of a subterm in a term. A subterm of t at position p is denoted by $t|_p$, and a replacement of a subterm of t at position p with the term s is denoted by $t[s]_p$.

It is common practice in equational theorem proving to consider logical theories containing only the equality predicate, as this simplifies the theoretical treatment without loss of generality. A literal $P(t_1, \ldots, t_n)$, where P is not

[3] `http://www.omg.org/uml/`

the equality predicate, is encoded as $P(t_1, \ldots, t_n) \approx \mathsf{tt}$, where tt is a new constant. Assuming that P and tt are of a sort different from the sort of terms t_i, this encoding preserves satisfiability. Technically speaking, P thus becomes a function symbol; however, when ambiguity does not arise, we call it a "predicate symbol" and we take $P(t_1, \ldots, t_n)$ to be a syntactic shortcut for $P(t_1, \ldots, t_n) \approx \mathsf{tt}$. Furthermore, we assume the predicate $\approx$ to have built-in symmetry: a literal $s \approx t$ should also be read as $t \approx s$, and a literal $s \not\approx t$ should also be read as $t \not\approx s$.

The inference rules of basic superposition work with *closures*. A closure $C \cdot \sigma$ consists of a *skeleton* clause C and a *substitution* σ. A closure $C \cdot \sigma$ can conveniently be represented by *marking* the terms in $C\sigma$ occurring at variable positions of C with $[\cdot]$. A position at or below a marked position is called a *substitution position*. For example, the clause $P(f(y)) \vee g(b) \approx b$ is logically equivalent to the closure $(P(x) \vee z \approx b) \cdot \{x \mapsto f(y), z \mapsto g(b)\}$, which can conveniently be represented as $P([f(y)]) \vee [g(b)] \approx b$. Semantically, $C \cdot \sigma$ is equivalent to $C\sigma$. A closure $C \cdot \sigma$ is *ground* if $C\sigma$ does not contain a variable.

The basic superposition calculus requires two parameters. The first parameter is an *admissible* ordering $\succ$ on terms—that is, a *reduction ordering* total on ground terms. The second parameter is a *selection function*, which selects an arbitrary (possibly empty) set of negative literals in each closure.

A standard admissible term ordering is the *lexicographic path ordering* (LPO) [18,19]. Given a well-founded strict ordering $>$ over function, predicate, and constant symbols (also called a *precedence*), it is defined as follows: $s \succ_{lpo} t$ if

(1) t is a variable occurring as a proper subterm of s, or
(2) $s = f(s_1, \ldots, s_m)$, $t = g(t_1, \ldots, t_n)$, and at least one among the following conditions holds:
   (a) $f > g$ and, for all i with $1 \leq i \leq n$, $s \succ_{lpo} t_i$, or
   (b) $f = g$ and, for some j, $(s_1, \ldots, s_{j-1}) = (t_1, \ldots t_{j-1})$, $s_j \succ_{lpo} t_j$, and $s \succ_{lpo} t_k$ for all k with $j < k \leq n$, or
   (c) $s_j \succeq_{lpo} t$ for some j with $1 \leq j \leq m$.

A term ordering $\succ$ is extended to literals by identifying a positive literal $s \approx t$ with a multiset $\{\{s\}, \{t\}\}$, a negative literal $s \not\approx t$ with a multiset $\{\{s, t\}\}$, and by comparing multisets using a two-fold multiset extension of $\succ$;[4] we denote the literal ordering also with $\succ$. A literal $L \cdot \theta$ is *(strictly) maximal* w.r.t. a closure $C \cdot \theta$ if there is no literal $L' \in C$ such that $L'\theta \succ L\theta$ ($L'\theta \succeq L\theta$).

[4] For $\succ$ a strict ordering on some set D, the *multiset extension* of $\succ$, written $\succ_{mul}$, is the strict ordering on finite multisets on D and is defined as follows: $M \succ_{mul} N$ if (i) $M \neq N$, and (ii) if $N(s) > M(s)$ for some s, then there is some $t \succ s$ such that $M(t) > N(t)$, where $M(s)$ is the number of occurrences of s in M.

Table 2
Inference Rules of Basic Superposition

Positive superposition:
$$\frac{(C \vee s \approx t) \cdot \rho \quad (D \vee w \approx v) \cdot \rho}{(C \vee D \vee w[t]_p \approx v) \cdot \theta}$$

where (i) $\sigma = \mathsf{MGU}(s\rho, w\rho|_p)$ and $\theta = \rho\sigma$, (ii) $t\theta \not\succeq s\theta$ and $v\theta \not\succeq w\theta$, (iii) $(s \approx t) \cdot \theta$ is strictly eligible for superposition in $(C \vee s \approx t) \cdot \theta$, (iv) $(w \approx v) \cdot \theta$ is strictly eligible for superposition in $(D \vee w \approx v) \cdot \theta$, (v) $s\theta \approx t\theta \not\succeq w\theta \approx v\theta$, (vi) $w|_p$ is not a variable.

Negative superposition:
$$\frac{(C \vee s \approx t) \cdot \rho \quad (D \vee w \not\approx v) \cdot \rho}{(C \vee D \vee w[t]_p \not\approx v) \cdot \theta}$$

where (i) $\sigma = \mathsf{MGU}(s\rho, w\rho|_p)$ and $\theta = \rho\sigma$, (ii) $t\theta \not\succeq s\theta$ and $v\theta \not\succeq w\theta$, (iii) $(s \approx t) \cdot \theta$ is strictly eligible for superposition in $(C \vee s \approx t) \cdot \theta$, (iv) $(w \not\approx v) \cdot \theta$ is eligible for resolution in $(D \vee w \not\approx v) \cdot \theta$, (v) $w|_p$ is not a variable.

Reflexivity resolution:
$$\frac{(C \vee s \not\approx t) \cdot \rho}{C \cdot \theta}$$

where (i) $\sigma = \mathsf{MGU}(s\rho, t\rho)$ and $\theta = \rho\sigma$, (ii) $(s \not\approx t) \cdot \theta$ is eligible for resolution in $(C \vee s \not\approx t) \cdot \theta$.

Equality factoring:
$$\frac{(C \vee s \approx t \vee s' \approx t') \cdot \rho}{(C \vee t \not\approx t' \vee s' \approx t') \cdot \theta}$$

where (i) $\sigma = \mathsf{MGU}(s\rho, s'\rho)$ and $\theta = \rho\sigma$, (ii) $t\theta \not\succeq s\theta$ and $t'\theta \not\succeq s'\theta$, (iii) $(s \approx t) \cdot \theta$ is eligible for superposition in $(C \vee s \approx t \vee s' \approx t') \cdot \theta$.

Ordered Hyperresolution:
$$\frac{(C_1 \vee A_1) \cdot \rho \ldots (C_n \vee A_n) \cdot \rho \quad (D \vee \neg B_1 \vee \ldots \vee \neg B_n) \cdot \rho}{(C_1 \vee \ldots \vee C_n \vee D) \cdot \theta}$$

where (i) σ is the most general substitution such that $A_i\theta = B_i\theta$ for $1 \leq i \leq n$ and $\theta = \rho\sigma$, (ii) each $A_i \cdot \theta$ is strictly eligible for superposition in E_i, (iii) either $\neg B_i \cdot \theta$ are selected, or nothing is selected, $n = 1$, and $\neg B_1 \cdot \theta$ is maximal w.r.t. $D \cdot \theta$.

A literal $L \cdot \theta$ is *(strictly) eligible for superposition* in a closure $(C \vee L) \cdot \theta$ if there are no selected literals in $(C \vee L) \cdot \theta$ and $L \cdot \theta$ is (strictly) maximal w.r.t. $C \cdot \theta$. A literal $L \cdot \theta$ is *eligible for resolution* in a closure $(C \vee L) \cdot \theta$ if it is selected in $(C \vee L) \cdot \theta$, or there are no selected literals in $(C \vee L) \cdot \theta$ and $L \cdot \theta$ is maximal w.r.t. $C \cdot \theta$. The basic superposition calculus, $\mathcal{BS}$ for short, consists of the inference rules presented in Table 2. As usual in resolution calculi, we assume that the premises do not share a common variable (neither in the skeleton nor in the substitution part). The last premise of all inference rules from the table is called the *main premise*, and all other premises are called *side premises*. Ordered hyperresolution is a "macro" inference that combines the effects of several negative superposition inferences with eager elimination of the literals $\mathsf{tt} \not\approx \mathsf{tt}$; we use it in our work for convenience.

It is important to distinguish an *inference rule* from an *inference*. An inference rule can be understood as a template that specifies actions to be applied to

any premises. An inference is an application of an inference rule to concrete premises. An inference ξ' is an *instance* of an inference ξ if ξ' is obtained by applying a substitution σ to all premises and the conclusion of ξ; the inference ξ' is also written as $\xi\sigma$. An inference is *ground* if all its closures are ground.

Basic superposition is compatible with powerful *redundancy elimination rules* that allow us to delete certain *redundant* closures [16]. Some of the redundancy elimination rules of $\mathcal{BS}$ rely on the notion of η-*domination*: for a substitution η, a term $s \cdot \sigma$ is η-dominated by a term $t \cdot \theta$, written $s \cdot \sigma \sqsubseteq_\eta t \cdot \theta$, if (i) $s\sigma\eta = t\theta$, and (ii) a substitution ρ exists such that $s\rho = t$. For literals, $(s \approx t) \cdot \sigma \sqsubseteq_\eta (w \approx v) \cdot \theta$ if $s \cdot \sigma \sqsubseteq_\eta w \cdot \theta$ and $t \cdot \sigma \sqsubseteq_\eta v \cdot \theta$, or $s \cdot \sigma \sqsubseteq_\eta v \cdot \theta$ and $t \cdot \sigma \sqsubseteq_\eta w \cdot \theta$. The definition is analogous for negative literals. For closures, $C \cdot \sigma \sqsubseteq_\eta D \cdot \theta$ if a permutation π of literals of $D \cdot \sigma$ exists such that $L_i \cdot \sigma \sqsubseteq_\eta L'_{\pi(i)} \cdot \theta$ for each $L_i \cdot \sigma \in C \cdot \sigma$ and the corresponding $L'_{\pi(i)} \cdot \theta \in D \cdot \theta$.

A closure $C \cdot \sigma$ *subsumes* a closure $D \cdot \theta$ if $|D| < |C|$ and a substitution η exists such that $D \cdot \theta \sqsubseteq_\eta C \cdot \sigma$. Each closure C that is subsumed by some closure from a set of closures N is *redundant in* N. Furthermore, tautologies are redundant in any closure set.

For N_0 a set of closures of form $C \cdot \{\}$, a *derivation* is a sequence of closure sets $N_0, N_1, \ldots, N_i$, where $N_i = N_{i-1} \cup \{D \cdot \rho\}$ and $D \cdot \rho$ is derived by applying a $\mathcal{BS}$ inference rule to premises from N_{i-1}, or $N_i = N_{i-1} \setminus \{D \cdot \rho\}$ and $D \cdot \rho$ is redundant in N_{i-1}. Each derivation is required to be *fair*; intuitively, no inference should be postponed infinitely often. For a precise definition of fairness and for ways of achieving it, please refer to [10,8]. $\mathcal{BS}$ [16,17] is refutationally complete: N_0 is satisfiable if and only if each derivation from N_0 contains a closure set N_i that is saturated up to redundancy and does not contain the empty closure.

Our proofs are based on the completeness proof for $\mathcal{BS}$ [17,16], which we summarize next. We assume familiarity with the basic definitions of term rewriting [19]. For equational theories, models are represented using a ground and convergent rewrite system. The main difference between basic and ordinary superposition lies in their approach to lifting: in basic superposition, a nonground closure $C \cdot \sigma$ represents only those ground instances of $C\sigma$ whose terms at substitution positions are in normal form w.r.t. the rewrite system defining the candidate model. This intuition is captured by the following definitions.

Let R be a ground and convergent rewrite system and $C \cdot \sigma$ a ground closure. A variable x in the skeleton C is *variable irreducible w.r.t.* R if (i) $x\sigma$ is irreducible by R, or (ii) x occurs in C only in literals of the form $x \approx s$ such that $x\sigma \succ s\sigma$, and $x\sigma$ is irreducible by those rules $l \Rightarrow r \in R$ for which $x\sigma \approx s\sigma \succ l \approx r$. Furthermore, $C \cdot \sigma$ is *variable irreducible w.r.t.* R if all variables from C are variable irreducible w.r.t. R. For $C \cdot \sigma$ a possibly nonground

closure, $\mathsf{irred}_R(C \cdot \sigma)$ is the set of all ground closures $C \cdot \sigma\tau$ that are variable irreducible w.r.t. R. For N a set of closures, $\mathsf{irred}_R(N) = \bigcup_{C \cdot \sigma \in N} \mathsf{irred}_R(C \cdot \sigma)$. For a $\mathcal{BS}$ inference ξ, the ground inference $\xi\tau$ is *variable irreducible w.r.t.* R if all closures in $\xi\tau$ are variable irreducible w.r.t. R.

We extend the literal ordering $\succ$ to closures by multiset extension, and we denote it also by $\succ$. With $\mathsf{irred}_R(N)^{\prec D}$ we denote the subset of $\mathsf{irred}_R(N)$ of closures that are smaller than some ground closure D. For N a set of closures, the notion of redundancy for $\mathcal{BS}$ is defined as follows. A closure $C \cdot \sigma$ is *redundant in* N if, for all rewrite systems R and all ground substitutions τ such that $C \cdot \sigma\tau$ is variable irreducible w.r.t. R, we have $R \cup \mathsf{irred}_R(N)^{\prec C \cdot \sigma\tau} \models C \cdot \sigma\tau$. An inference ξ with premises $D_1 \cdot \sigma$ and $D_2 \cdot \sigma$ and a conclusion $C \cdot \rho$ is redundant in N if, for all rewrite systems R and all ground substitutions τ such that $\xi\tau$ is variable irreducible w.r.t. R, we have $R \cup \mathsf{irred}_R(N)^{\prec D} \models C \cdot \rho\tau$, for $D = \max(D_1 \cdot \sigma\tau, D_2 \cdot \sigma\tau)$. A set of closures N is *saturated up to redundancy by* $\mathcal{BS}$ if all inferences from the premises in N are redundant in N.

A set of closures N is *well-constrained* if $\mathsf{irred}_R(N) \cup R \models N$ for any rewrite system R. If ρ is empty for all $C \cdot \rho \in N$, then N is well-constrained, since each variable reducible position of a ground instance of $C \cdot \rho$ can be reduced by R to a closure in $\mathsf{irred}_R(N)$. Furthermore, N' is well-constrained if it is obtained from a well-constrained set N by a sound inference rule.

For R a ground and convergent rewrite system, R^* is the smallest Herbrand interpretation such that $a \approx b \in R^*$ if and only if a and b have the same normal form in R. Let N be a closure set not containing the empty closure, obtained by saturating a well-constrained set N_0 up to redundancy by $\mathcal{BS}$. Using the model building technique [16,17], one can generate a ground convergent rewrite system R_N such that ${R_N}^* \models \mathsf{irred}_{R_N}(N)$. Finally, $R_N \subseteq {R_N}^*$ and N is well-constrained, so ${R_N}^* \models N$. Hence, N is satisfiable, and so is N_0.

3 Decision Procedure Overview

The fundamental principles for deciding a first-order fragment $\mathcal{L}$ by resolution have been formulated by Joyner [20]. First, one selects a sound and complete resolution calculus $\mathcal{C}$. Then, one identifies a set of clauses $\mathcal{N}_\mathcal{L}$ such that

(1) each formula $\varphi \in \mathcal{L}$, when translated into clauses, yields clauses from $\mathcal{N}_\mathcal{L}$;
(2) $\mathcal{N}_\mathcal{L}$ is finite for a finite signature; and
(3) $\mathcal{N}_\mathcal{L}$ is closed under $\mathcal{C}$—that is, applying an inference of $\mathcal{C}$ to clauses from $\mathcal{N}_\mathcal{L}$ produces a clause in $\mathcal{N}_\mathcal{L}$.

This is sufficient to obtain a refutation decision procedure for $\mathcal{L}$: in the worst case, $\mathcal{C}$ will derive all clauses of $\mathcal{N}_{\mathcal{L}}$.

A minor problem in applying these principles to deciding satisfiability of a $\mathcal{SHIQ}$ knowledge base KB is caused by transitivity axioms, which produce clauses without *covering literals*—literals containing all variables of a clause [21]. As shown in [22], termination of resolution on such clauses is very difficult to achieve. To address this, we show in Section 4 how to eliminate transitivity axioms by polynomially encoding a $\mathcal{SHIQ}$ knowledge base KB into an equisatisfiable $\mathcal{ALCHIQ}$ knowledge base $\Omega(KB)$. After this initial transformation step, we consider only $\mathcal{ALCHIQ}$ knowledge bases.

A more complex problem is that, due to an interaction between role hierarchies, inverse roles, and number restrictions, basic superposition may produce closures with terms of ever increasing depth in a derivation, thus preventing the calculus from terminating. We deal with this problem in two stages: we first derive a decision procedure for a slightly weaker logic $\mathcal{ALCHIQ}^{-}$, and then extend it to the full $\mathcal{ALCHIQ}$.

In Section 5, we present a procedure for deciding satisfiability of an $\mathcal{ALCHIQ}^{-}$ knowledge base KB. We start by preprocessing KB into a set of closures $\Xi(KB)$ as explained in Section 5.1. We then saturate $\Xi(KB)$ under $\mathcal{BS}_{DL}$ with eager application of redundancy elimination rules, where $\mathcal{BS}_{DL}$ is the $\mathcal{BS}$ calculus with its two parameters, ordering and selection function, instantiated according to Definition 9 in Section 5.2. Since $\mathcal{BS}_{DL}$ is sound and complete [16], the saturated set of closures contains the empty closure if and only if $\Xi(KB)$ is unsatisfiable. To show in Sections 5.3 and 5.4 that saturation always terminates, we use the following proof-theoretic argument:

- We give a syntactic characterisation of *sets of $\mathcal{ALCHIQ}^{-}$-closures* and show in Lemma 8 that $\Xi(KB)$ is such a set.
- In Lemma 11, we show that, in any $\mathcal{BS}_{DL}$-derivation starting from $\Xi(KB)$, each inference produces either a set of $\mathcal{ALCHIQ}^{-}$-closures, or a closure that is redundant and can therefore be deleted.
- In Lemma 13, we show that, assuming a finite knowledge base, each set of $\mathcal{ALCHIQ}^{-}$-closures occurring in each $\mathcal{BS}_{DL}$-derivation is finite.
- Termination is a consequence of Lemmas 11 and 13 and the fact that we do not produce too many redundant closures. The bound on the size of the maximal set of $\mathcal{ALCHIQ}^{-}$-closures yields the algorithm's complexity, as shown in Theorem 14.

To handle $\mathcal{ALCHIQ}$, in Section 6.1 we extend the $\mathcal{BS}$ calculus with a *decomposition* inference rule that transforms certain closures into simpler ones. We show that decomposition is sound and complete, and, in Section 6.2, we show that it guarantees the termination of basic superposition for $\mathcal{ALCHIQ}$.

4 Eliminating Transitivity Axioms

In this section, we show how to eliminate transitivity axioms from a $\mathcal{SHIQ}$ knowledge base KB by transforming it into an equisatisfiable $\mathcal{ALCHIQ}$ knowledge base $\Omega(KB)$. Since $\Omega(KB)$ is satisfiable if and only if KB is, in the remaining sections we can indeed restrict our attention to $\mathcal{ALCHIQ}$ knowledge bases without loss of generality.

Definition 4 *For KB a $\mathcal{SHIQ}$ knowledge base, $\mathsf{clos}(KB)$ is the smallest set satisfying the following conditions:*

- $\mathsf{nnf}(\neg C \sqcup D) \in \mathsf{clos}(KB)$ *if* $C \sqsubseteq D \in KB_{\mathcal{T}}$;
- $\mathsf{nnf}(C) \in \mathsf{clos}(KB)$ *if* $C(a) \in KB_{\mathcal{A}}$;
- $D \in \mathsf{clos}(KB)$ *if* $C \in \mathsf{clos}(KB)$ *and* D *is a subconcept of* C;
- $\mathsf{nnf}(\neg C) \in \mathsf{clos}(KB)$ *if* $\leq n\, R.C \in \mathsf{clos}(KB)$;
- $\forall S.C \in \mathsf{clos}(KB)$ *if* $\forall R.C \in \mathsf{clos}(KB)$, $S \sqsubseteq^* R$, *and* S *is transitive.*

$\Omega(KB)$ is the $\mathcal{ALCHIQ}$ knowledge base obtained from KB by removing all transitivity axioms, and by adding an axiom $\forall R.C \sqsubseteq \forall S.(\forall S.C)$ for each concept $\forall R.C \in \mathsf{clos}(KB)$ and each transitive role S with $S \sqsubseteq^* R$.

This encoding is similar to the transformation of formulae of modal logic K4 into formulae of modal logic K by Schmidt and Hustadt [23]. Another related algorithm for transforming $\mathcal{SHIQ}$ concepts to $\mathcal{ALCIQ}b$ concepts was presented by Tobies [6]. Intuitively, $\mathsf{clos}(KB)$ is the set of the "relevant" concepts occurring in KB; it is analogous to the Fisher-Ladner closure of propositional dynamic logic [24] or the set of subformulae of a modal formula. The transformation $\Omega(KB)$ is based on the following observation: if some individual a is an instance of some concept $\forall R.C$ and R is transitive, then each individual reachable from a through an R-path must be an instance of C; this can be ensured by propagating the concept $\forall R.C$ along a single R step in the path, which is achieved by an axiom of the form $\forall R.C \sqsubseteq \forall R.\forall R.C$. Definition 4 differs from the well-known transformations mainly by taking into account role hierarchies: given $\forall R.C$, the concept C can be propagated to each individual connected by a path consisting of the transitive subroles of R. Furthermore, in number restrictions $\leq n\, R.C$, the concept C occurs effectively negatively, so $\mathsf{clos}(KB)$ must contain the negation-normal form of $\neg C$, and not just C.

We remind the reader that the definition of $\mathcal{SHIQ}$ allows only simple roles—that is, roles without transitive subroles—to occur in number restrictions. This restriction is necessary because allowing transitive roles in number restrictions makes the logic undecidable [3]. Thus, Definition 4 does not need to propagate number restriction concepts along transitive roles.

The presented encoding is polynomial in $|KB|$: the number of concepts in $\mathsf{clos}(KB)$ stemming from a concept C is bounded by $2 \cdot |C| \cdot |N_R|$ and, for each concept from $\mathsf{clos}(KB)$, we generate at most $|N_R|$ axioms in $\Omega(KB)_{\mathcal{T}}$. Our transformation differs only slightly from existing ones, so we leave the proof of the following theorem to [25].

Theorem 5 ([25]) *KB is satisfiable if and only if $\Omega(KB)$ is satisfiable.*

The models of KB and $\Omega(KB)$ may differ on complex roles. Therefore, in Definition 1, we require the role S to be simple in assertions $\neg S(a, b)$.

5 Deciding $\mathcal{ALCHIQ}^-$ by Basic Superposition

We now present an algorithm for deciding satisfiability of an $\mathcal{ALCHIQ}^-$ knowledge base KB by basic superposition.

5.1 Preprocessing

To decide satisfiability of KB, we transform it into clausal form. A straightforward transformation of $\pi(KB)$ into conjunctive normal form might exponentially increase the formula size and could destroy the structure of the formula. Therefore, before clausification, we apply to KB the *structural transformation* [26–28], also known as *renaming*.

Definition 6 *For KB an $\mathcal{ALCHIQ}$ knowledge base, $\Theta(KB)$ is the knowledge base obtained by applying the* structural transformation *to KB, where the operator Θ is defined in Table 3. Furthermore, $\Xi(KB)$ is the set of closures obtained by skolemizing $\pi(\Theta(KB))$ and rewriting it into conjunctive normal form (see, e.g., [27]). The* designated role *for a function symbol f, written $\mathsf{role}(f)$, is the role occurring in the concept whose skolemization introduced f.*

Definition 7 generalizes the closures of $\Xi(KB)$ to closures that can be obtained from $\Xi(KB)$ by basic superposition.

Definition 7 *A set of closures N is a set of* $\mathcal{ALCHIQ}^-$-closures *if each closure in N is of the form from Table 4 and it satisfies Conditions (*i*)–(*vii*).*

Note that one cannot determine whether a closure alone is an $\mathcal{ALCHIQ}^-$-closure, since properties (*iii*)–(*vii*) from Table 4 require existence of other closures. Therefore, we only speak of sets of $\mathcal{ALCHIQ}^-$-closures, and not of individual $\mathcal{ALCHIQ}^-$-closures. This is in contrast with the usual definitions (e.g., [29]), which typically can consider each clause in isolation.

Table 3
The Structural Transformation of KB

$$\Theta(KB) = \bigcup_{\alpha \in KB_{\mathcal{R}} \cup KB_{\mathcal{A}}} \Theta(\alpha) \cup \bigcup_{C_1 \sqsubseteq C_2 \in KB_{\mathcal{T}}} \Theta(\top \sqsubseteq \mathsf{nnf}(\neg C_1 \sqcup C_2))$$

$$\Theta(C(a)) = \{Q_C(a)\} \cup \Theta(Q_C \sqsubseteq C)$$

$$\Theta(A \sqsubseteq C_1 \sqcap C_2) = \Theta(A \sqsubseteq C_1) \cup \Theta(A \sqsubseteq C_2)$$

$$\Theta(A \sqsubseteq C_1 \sqcup C_2) = \{A \sqsubseteq Q_{C_1} \sqcup Q_{C_2}\} \cup \Theta(Q_{C_1} \sqsubseteq C_1) \cup \Theta(Q_{C_2} \sqsubseteq C_2)$$

$$\Theta(A \sqsubseteq \exists R.C) = \{A \sqsubseteq \exists R.Q_C\} \cup \Theta(Q_C \sqsubseteq C)$$

$$\Theta(A \sqsubseteq \forall R.C) = \{A \sqsubseteq \forall R.Q_C\} \cup \Theta(Q_C \sqsubseteq C)$$

$$\Theta(A \sqsubseteq \geq n\, R.C) = \{A \sqsubseteq \geq n\, R.Q_C\} \cup \Theta(Q_C \sqsubseteq C)$$

$$\Theta(A \sqsubseteq \leq n\, R.C) = \{A \sqsubseteq \leq n\, R.\neg Q_D\} \cup \Theta(Q_D \sqsubseteq D) \text{ for } D = \mathsf{nnf}(\neg C)$$

$$\Theta(\beta) = \{\beta\} \text{ for any other type of axiom } \beta$$

Note: A and B are atomic concepts or $\top$; C, C_1, and C_2 are arbitrary concepts; R and S are roles; and Q_X is a new atomic concept not occurring in KB that is unique for the concept X.

We now summarize the properties of preprocessing:

Lemma 8 *For KB an $\mathcal{ALCHIQ}^-$ knowledge base, the following claims hold:*

(1) KB is satisfiable if and only if $\Xi(KB)$ is satisfiable.
(2) $\Xi(KB)$ can be computed in time polynomial in $|KB|$ for unary coding of numbers in input.
(3) $\Xi(KB)$ is a set of $\mathcal{ALCHIQ}^-$-closures.

PROOF. (Claim 1) Since the operator Θ can be seen as a syntactic variant of the structural transformation [26–28], $\Theta(KB)$ is satisfiable if and only if KB is satisfiable. This is because each model I of $\Theta(KB)$ is clearly a model of I; furthermore, each model I of KB can be extended to a model of $\Theta(KB)$ by interpreting each new atomic concept Q_X as the concept X. Finally, $\Xi(KB)$ is computed from $\Theta(KB)$ using satisfiability-preserving transformations.

(Claim 2) The operator Θ is applied at most once for each subconcept occurring in KB, so the number of new concepts Q_X is linear in $|KB|$, and $\Theta(KB)$ can be computed in polynomial time. The number of function symbols introduced by skolemizing an existential quantifier is bounded by the maximal number occurring in a number restriction. For unary coding of numbers, this number is linear in $|KB|$, so $\Xi(KB)$ can be computed in polynomial time.

(Claim 3) By the definition of π from Table 1, inverse properties yield closures of type 1; role inclusions yield closures of type 2; ABox axioms yield closures of types 8. For TBox axioms, observe that $\Theta(KB)$ contains only axioms of the

Table 4
$\mathcal{ALCHIQ}^-$-Closures

1	$\neg R(x,y) \vee \mathsf{Inv}(R)(y,x)$
2	$\neg R(x,y) \vee S(x,y)$
3	$\mathbf{P}(x) \vee R(x, \langle f(x) \rangle)$
4	$\mathbf{P}(x) \vee R([f(x)], x)$
5	$\mathbf{P_1}(x) \vee \mathbf{P_2}(\langle \mathbf{f}(x) \rangle) \vee \bigvee \langle f_i(x) \rangle \bowtie \langle f_j(x) \rangle$
6	$\mathbf{P_1}(x) \vee \mathbf{P_2}([g(x)]) \vee \mathbf{P_3}(\langle \mathbf{f}([g(x)]) \rangle) \vee \bigvee \langle t_i \rangle \bowtie \langle t_j \rangle$ where t_i and t_j are either of the form $f([g(x)])$ or of the form x
7	$\mathbf{P_1}(x) \vee \bigvee_{i=1}^{n} \neg R(x, y_i) \vee \bigvee_{i=1}^{n} \mathbf{P_2}(y_i) \vee \bigvee_{i=1}^{n} \overset{n}{\underset{j=i+1}{}} y_i \approx y_j$
8	$\mathbf{R}(\langle \mathbf{a} \rangle, \langle \mathbf{b} \rangle) \vee \mathbf{P}(\langle \mathbf{t} \rangle) \vee \bigvee \langle t_i \rangle \bowtie \langle t_j \rangle$ where t, t_i, and t_j are either a constant b or a term $f_i([a])$

Conditions:

(i) In each term $f(t)$, the inner term t occurs marked.

(ii) In all positive equality literals with at least one function symbol, both sides are marked.

(iii) For each function symbol f occurring in N, the set N contains exactly one *generator* closure $\mathbf{P^f}(x) \vee R(x, f(x))$ with $f(x)$ unmarked; $\mathsf{role}(f) = R$; and $\mathbf{P^f}(x)$ is the unique disjunction of unary literals from the generator for f.

(iv) Each closure that contains a term $f(t)$, contains $\mathbf{P^f}(t)$ as well.

(v) In each literal of the form $[f_i(t)] \approx [f_j(t)]$, we have $\mathsf{role}(f_i) = \mathsf{role}(f_j)$.

(vi) In each literal of the form $[f(g(x))] \approx x$, we have $\mathsf{role}(f) = \mathsf{Inv}(\mathsf{role}(g))$.

(vii) For each $[f_i(a)] \approx [b]$ in a closure C, a *witness* closure of C exists in N that is of form $R(\langle a \rangle, \langle b \rangle) \vee D$, $\mathsf{role}(f_i) = R$, D does not contain function symbols or negative binary literals, and D is contained in C.

Notation: $\mathbf{P}(t)$ is a possibly empty disjunction of the form $(\neg)P_1(t) \vee \ldots \vee (\neg)P_n(t)$; $\mathbf{P}(\mathbf{f}(x))$ is a possibly empty disjunction of the form $\mathbf{P_1}(f_1(x)) \vee \ldots \vee \mathbf{P_m}(f_m(x))$; $\langle t \rangle$ means that the term t may, but need not be marked; each closure of type 6 contains at least one term $f(g(x))$; and the symbol $\bowtie$ denotes either $\approx$ or $\not\approx$.

form $A \sqcup (\neg)D$, where A is a an atomic concept and D is a $\mathcal{SHIQ}$ concept containing only literal subconcepts. An existential restriction D yields closures of type 3 and 4; an at-least number restriction D yields closures of types 3, 4, and 5; a conjunction or disjunction of literals D yields a closure of type 5; a universal restriction or at-most number restriction D yields a closures of type 7. Conditions (*i*), (*ii*), and (*iv*)–(*vii*) are trivially true for $\Xi(KB)$. For Condition (*iii*), notice that each function symbol f is obtained by skolemizing $\exists R.C$ or $\geq n\, R.C$, producing a closure of type 3 for each f. $\square$

Using binary coding, a number n can be represented using $\lceil \log_2 n \rceil$ bits, so in the presence of number restrictions in a knowledge base KB, the number of function symbols introduced by skolemization is exponential in $|KB|$. Hence, for binary coding of numbers, our translation incurs an exponential blowup.

To understand the following definition, remember that literals $P(t_1, \ldots, t_n)$ are encoded as $P(t_1, \ldots, t_n) \approx \mathsf{tt}$, as discussed in Section 2.2.

Definition 9 *Let $\mathcal{BS}_{DL}$ denote the $\mathcal{BS}$ calculus parameterized as follows. The term ordering $\succ$ is any admissible ordering that satisfies the following conditions, for all unary predicates P, P', and P'', binary predicates R, function symbols f and g, and constants a and b:*

- $P''(f(g(x))) \succ f(g(x)) \succ P'(g(x)) \succ g(x) \succ P(x) \succ \mathsf{tt}$,
- $R(x, f(x)) \succ f(x)$,
- $R(f(x), x) \succ f(x)$,
- $P(f(a)) \succ f(a) \succ P'(b) \succ c$, *and*
- $f(a) \succ R(b, c)$.

Furthermore, the selection function selects every negative binary literal.

Definition 9 may seem redundant, since most term orderings used in practice have the *subterm property*: $t \succ t|_p$ for each term t and a nonempty position p. Note, however, that an admissible ordering need not have the subterm property in general.

A term ordering compatible with Definition 9 can be obtained by instantiating a lexicographic path ordering (LPO) with any precedence over function, constant, and predicate symbols such that $f > c > P > \mathsf{tt}$ for each function symbol f, constant symbol c, and a predicate symbol P. It is easy to see that such an LPO is compatible with Definition 9. Any LPO has the subterm property [19], so $f(t) \succ t$. Furthermore, because $f > P$ for each f and P, we have $f(t) \succ P(t)$. Now this immediately gives us the first three properties of Definition 9. The other properties follow similarly.

$\mathcal{ALCHIQ}^-$-closures of types 1, 2, and 7 contain selected literals so, to apply $\mathcal{BS}_{DL}$ to $\Xi(KB)$, we need to compare literals only in closures of types 3–6 and 8 from Table 4. Definition 9 requires the literals from such closures to be comparable, which allows us to use a simplified definition of the literal ordering. For each literal $L = s \bowtie t$ with $\bowtie \in \{\approx, \not\approx\}$, we consider the triple $c_L = (\max(s,t), p_L, \min(s,t))$, where (i) $\max(s,t)$ is the larger of the two terms; (ii) $p_L = 1$ if $\bowtie$ is $\not\approx$; (iii) $p_L = 0$ if $\bowtie$ is $\approx$; and (iv) $\min(s,t)$ is the smaller of the two terms. Then, $L_1 \succ L_2$ if and only if $c_{L_1} \succ c_{L_2}$, where c_{L_i} are compared lexicographically, using the term ordering $\succ$ for the first and the third position, and using $1 \succ 0$ for the second position. On $\mathcal{ALCHIQ}^-$-closures of types 3–6 and 8, this definition is equivalent to the one based on the two-fold multiset extension from Section 2.2.

We now state a lemma that shows which literals can be maximal under the ordering and the selection function of $\mathcal{BS}_{DL}$. The proof follows immediately from the properties of the term ordering of $\mathcal{BS}_{DL}$ from Definition 9.

Lemma 10 *The maximal literal of an $\mathcal{ALCHIQ}^-$-closure that participates in an inference by $\mathcal{BS}_{DL}$ satisfies the following conditions:*

- *In a closure of type 3, the literal $R(x, \langle f(x)\rangle)$ is always maximal.*
- *In a closure of type 4, the literal $R([f(x)], x)$ is always maximal.*
- *In a closure of type 5, a literal $(\neg)P(x)$ can be maximal only if the closure does not contain a term $f(x)$.*
- *In a closure of type 6, the maximal literal contains a term $f([g(x)])$.*
- *In a closure of type 8, a literal $(\neg)R(a, b)$, $(\neg)P(a)$, $a \approx b$, or $a \not\approx b$ can be maximal only if the closure does not contain a function symbol.*

To simplify the presentation, we make a technical assumption that, whenever $\mathcal{BS}_{DL}$ derives a conclusion of the form $C \cdot \rho \vee t \not\approx t$, this conclusion is eagerly simplified to $C \cdot \rho$. Since an application of reflexivity resolution to $C \cdot \rho \vee t \not\approx t$ produces $C \cdot \rho$, and $C \cdot \rho$ makes $C \cdot \rho \vee t \not\approx t$ redundant, this simplification does not affect completeness of $\mathcal{BS}_{DL}$.

We now prove that $\mathcal{ALCHIQ}^-$-closures are closed under $\mathcal{BS}_{DL}$.

Lemma 11 *Let N be a set of $\mathcal{ALCHIQ}^-$-closures and $C \cdot \rho$ a closure obtained by applying a $\mathcal{BS}_{DL}$ inference to premises from N. Then, either $C \cdot \rho$ is redundant in N or $N \cup \{C \cdot \rho\}$ is a set of $\mathcal{ALCHIQ}^-$-closures.*

PROOF. We show that all possible inferences by $\mathcal{BS}_{DL}$ on all types of $\mathcal{ALCHIQ}^-$-closures produce a closure with syntactic structure from Table 4, for which we then verify Conditions (i)–(vii).

Consider any inference between closures of types 5, 6, or 8. Because the term $g(x)$ in some $f([g(x)])$ is marked, such an inference unifies unary terms only at their root positions. Hence, each unifier used in such an inference is either empty, or it has the form $\sigma = \{x \mapsto t\}$, where the depth of t is limited by the difference of the depths of the terms from the premises. Thus, the depth of terms in the conclusion is also bounded by the depth of the terms in the premises. Notice that superposition from $f(g(x)) \approx x$ into $f(g(x')) \not\approx x'$ results in $x' \not\approx x'$, and superposition from $P(t) \approx \mathsf{tt}$ into $P(s) \not\approx \mathsf{tt}$ results in $\mathsf{tt} \not\approx \mathsf{tt}$; these literals are eagerly eliminated by reflexivity resolution. Hence, the conclusion is of type 5, 6, or 8.

Because negative binary literals are always selected, a closure C of type 7 can participate only in a hyperresolution inference as the main premise. The side premises can have the maximal literals of the form $R(x_i, \langle f_i(x_i)\rangle)$, $R([g_i(x_i)], x_i)$, or $R(\langle a\rangle, \langle b_i\rangle)$. The following combinations are possible:

- Assume that the first side premise has the maximal literal $R([g(x')], x')$. Because $g(x')$ is unified with x, each other premise of type 4 must have a maximal literal of the form $R([g(x'')], x'')$; however, because the closure C contains a literal $y_i \approx y_j$ for each i and j, the conclusion contains $x \approx x$, so it is a tautology. For a nonredundant conclusion, since $g(x')$ does not unify with a constant, side premises for $i > 1$ must be of type 3, so the unifier has the form $\sigma = \{x \mapsto g(x'), x_i \mapsto g(x'), y_1 \mapsto x', y_i \mapsto f_i(g(x')) \text{ for } 2 \leq i \leq n\}$. If $n = 1$, the conclusion is of type 5; otherwise, it is of type 6.
- Assume that $R(x_i, \langle f_i(x_i)\rangle)$ is the maximal literal in all side premises. The unifier σ is of the form $\{x_i \mapsto x, y_i \mapsto f_i(x)\}$, so the conclusion is of type 5.
- Assume that some side premises have the maximal literal $R(x_i, \langle f_i(x_i)\rangle)$, for $1 \leq i \leq k$, and $R(\langle a\rangle, \langle b_i\rangle)$ for $k < i \leq n$ (the latter literals must have the same first argument since all these arguments must unify with x). The unifier σ contains mappings of the form $x \mapsto a, x_i \mapsto a, y_i \mapsto f_i(a)$ for $1 \leq i \leq k$, and $y_i \mapsto b_i$ for $k + 1 \leq i \leq n$, so the conclusion is of type 8.

Consider a superposition into a closure of type 3 with a free variable x'. By Lemma 10, $(w \approx v) \cdot \rho$ can only be the literal $R(x', f(x'))$, with R being the designated role for f. There are three possibilities:

- $(C \vee s \approx t) \cdot \rho$ is of type 5, 6, or 8 with $(s \approx t) \cdot \rho$ being $[f(u)] \approx [g(u)]$. Then, $\sigma = \{x' \mapsto u\}$, so the conclusion is $S = \mathbf{P^f}([u]) \vee R([u], [g(u)]) \vee C \cdot \rho$, where $\mathbf{P^g}([u]) \subseteq C \cdot \rho$. By Condition (v), $\mathsf{role}(f) = \mathsf{role}(g)$, so by Condition (iii), $\mathbf{P^g}(y) \vee R(y, g(y))$ exists; it subsumes S via the substitution $\{y \mapsto u\}$.
- $(C \vee s \approx t) \cdot \rho$ is of type 6 with $(s \approx t) \cdot \rho$ being $[f(g(x))] \approx x$. Then, $\sigma = \{x' \mapsto g(x)\}$, so the conclusion is $S = \mathbf{P^f}([g(x)]) \vee R([g(x)], x) \vee C \cdot \rho$, where $\mathbf{P^g}(x) \subseteq C \cdot \rho$. By Condition (vi), $\mathsf{role}(f) = \mathsf{Inv}(\mathsf{role}(g))$, so by Condition (iii), the closure $\mathbf{P^g}(y) \vee \mathsf{Inv}(R)(y, g(y))$ exists. Because $\mathsf{Inv}(R)(y, g(y))$ and $R(g(y), y)$ are semantically equivalent, the latter closure subsumes S via the substitution $\{y \mapsto x\}$.
- $(C \vee s \approx t) \cdot \rho$ is of type 8 with $(s \approx t) \cdot \rho$ being $[f(a)] \approx [b]$. The unifier is $\sigma = \{x' \mapsto a\}$, so the conclusion is $S = \mathbf{P^f}([a]) \vee R([a], [b]) \vee C \cdot \rho$. By Condition (vii), a witness of the form $R(\langle a\rangle, \langle b\rangle) \vee D$ with $D \subseteq C \cdot \rho$ exists, and it subsumes S via the empty substitution.

Equality factoring is possible only for a closure of type 5, 6, or 8. Such closures contain only one variable, so, due to occurs-check in unification, the unifier between literals within the closure is always empty. Similarly, reflexivity resolution can only be applied to a closure of type 5, 6, or 8 with the empty unifier. Hence, the conclusion of these two inferences is always of type 5, 6,

or 8. The conclusion of reflexivity resolution always subsumes the premise, so this inference rule should always be applied eagerly. The only remaining inference is hyperresolution of a closure of type 1 or 2 with a closure of type 3, 4, or 8, which obviously yields a closure of type 3, 4, or 8.

We now show that all the Conditions (i)–(vii) from Table 4 hold for each nonredundant inference conclusion. No inference removes markers from a closure. Condition (i) holds for a conclusion of each inference because each term $f(t)$ in the conclusion either stems from some premise, or it is obtained by instantiating some $f(x)$ to $f([t])$. Condition (ii) holds because all equality literals containing functional terms are generated by instantiating a literal $y_i \approx y_j$ in a hyperresolution inference with a closure of type 7. Condition (iii) holds because all conclusions of type 3 are produced by resolving a closure of type 3 or 4 with a closure of type 1 or 2.

If a closure contains $f([t])$ and satisfies Condition (iv), by Lemma 10, literals from $\mathbf{P^f}([t])$ cannot participate in an inference. Furthermore, a variable x is instantiated simultaneously in $f([t])$ and $\mathbf{P^f}([t])$. The terms containing function symbols occurring in an inference conclusion always stem from one of the inference premises, so Condition (iv) holds for each conclusion.

All equality literals containing function symbols are generated by hyperresolution with a main premise of type 7. Since the role R occurring in the premise is a leaf role, a closure of type 3 or 4 containing R cannot be resolved with a closure of type 2. Hence, for all side premises $\mathbf{P^f}(x) \vee R(x, \langle f(x)\rangle)$, we have $\mathsf{role}(f) = R$, and, for a side premise $\mathbf{P^g}(x) \vee R([g(x)], x)$, we have $\mathsf{role}(g) = \mathsf{Inv}(R)$. Hence, Conditions (v) and (vi) are satisfied for each conclusion of such a hyperresolution inference. Furthermore, by Condition (ii), superposition into positive equality literals containing function symbols is not possible, and, in each $[f_i(x)] \approx [f_j(x)]$, the variable x is instantiated simultaneously. Hence, Conditions (v) and (vi) hold for each inference conclusion.

Finally, all literals of the form $[f(a)] \approx [b]$ are generated by hyperresolution involving a side premise E_1 of type 8 with the maximal literal $R(\langle a\rangle, \langle b\rangle)$, and a side premise E_2 of type $\mathbf{P^f}(x) \vee R(x, \langle f(x)\rangle)$. Since the role associated with R occurring in such C is a leaf role, a closure of type 8 cannot be resolved on R with a closure of type 2, so $\mathsf{role}(f) = R$. Since the literal $R(\langle a\rangle, \langle b\rangle)$ is maximal in E_1, by Lemma 10, no literal from E_1 contains a function symbol, and E_2 does not contain a negative binary literal. Hence, the conclusion of such an inference satisfies Condition (vii). Assume now that Condition (vii) holds for some closure C, which contains a literal $[f(a)] \approx [b]$ whose witness is some closure D. Since no literal from D contains a function symbol and no such D is a negative binary literal, by Lemma 10, no literal from C occurring in D can participate in an inference, so all literals from D are present in each conclusion; hence, Condition (vii) holds for a conclusion of each inference. □

The following corollary follows from the case analysis of Lemma 11:

Corollary 12 *Superposition into an $\mathcal{ALCHIQ}^-$-closure of type 3 always results in a redundant conclusion.*

5.4 Termination and Complexity Analysis

We now show that the number of $\mathcal{ALCHIQ}^-$-closures is finite for a finite signature. This, in combination with Lemma 11 and the soundness and completeness of $\mathcal{BS}_{DL}$, implies that $\mathcal{BS}_{DL}$ with eager application of redundancy elimination rules decides satisfiability of $\mathcal{ALCHIQ}^-$ knowledge bases.

To simplify the following presentation, we make several technical assumptions about the implementation of the saturation process. First, instead of physically deleting redundant closures, we just mark them as deleted, thus preventing them from participating in future inferences. In this way we ensure that subsumption does not delete generator and witness closures from the closure so Conditions (*iii*) and (*vii*) always hold, and we ensure that no inference between the same set of premises is performed twice. Second, due to Corollary 12, we assume that $\mathcal{BS}_{DL}$ does not perform any superposition inferences into $\mathcal{ALCHIQ}^-$-closures of type 3.

Lemma 13 *Let $N_0, N_1, \ldots, N_i$ be a $\mathcal{BS}_{DL}$ derivation such that $N_0 = \Xi(KB)$ and C is a closure from some N_i. Then, $|C|$ is at most polynomial in $|KB|$, and $|N_i|$ is at most exponential in $|KB|$, for unary coding of numbers in the input.*

PROOF. By a straightforward inductive application of Lemma 11, each N_i contains only $\mathcal{ALCHIQ}^-$-closures. Let r be the number of role predicates, a the number of atomic concept predicates, c the number of constants, f the number of function symbols, and v the number of variables in a closure from N_i. Obviously, r, c, and a are linear in $|KB|$. The number f is bounded by the sum of all numbers n in $\geq n\, R.C$ and $\leq n\, R.C$, plus one for each $\exists R.C$ and $\forall R.C$ occurring in KB. Since numbers are coded in unary, f is linear in $|KB|$. Similarly, no $\mathcal{BS}_{DL}$ inference on $\mathcal{ALCHIQ}^-$-closures derives a closure of type 7, so v is bounded by the maximal number in a number restriction, and it is linear in $|KB|$ for unary coding of numbers.

The number of unary terms in a closure is bounded by $t = (f^2 + f) \cdot (v + c)$. Hence, the number of unary literals is bounded by $\ell_1 = 2a \cdot 2t$, the number of equality literals by $\ell_2 = 2(2t)^2$, and the number of binary literals by $\ell_3 = 2r \cdot (2t)^2$ (the leading factor 2 takes into account that each atom can be positive or negative, and $2t$ takes into account that each term can be marked

or not). Obviously, the number of different literals $\ell = \ell_1 + \ell_2 + \ell_3$ is polynomial in $|KB|$. Each closure contains a subset of these literal, and there are 2^ℓ such subsets, so the number of closures in N_i is exponential in $|KB|$. □

Theorem 14 *For an $\mathcal{ALCHIQ}^-$ knowledge base KB, saturation of $\Xi(KB)$ by $\mathcal{BS}_{DL}$ with eager elimination of redundancy decides satisfiability of KB, and it runs in time exponential in $|KB|$ for unary coding of numbers.*

PROOF. The set $\Xi(KB)$ can be computed in polynomial time by Lemma 8. Furthermore, by Lemmas 8 and 11, starting from $\Xi(KB)$, all nonredundant closures derived by $\mathcal{BS}_{DL}$ are $\mathcal{ALCHIQ}^-$-closures. By Lemma 13, the number of literals ℓ in an $\mathcal{ALCHIQ}^-$-closure is polynomial in $|KB|$, and the number c of different $\mathcal{ALCHIQ}^-$-closures is exponential in $|KB|$. Then, each inference involves at most c^ℓ premises, so the number of different inferences on $\mathcal{ALCHIQ}^-$-closures is exponential in $|KB|$. Subsumption can be checked in time exponential in the closure size [30]. Hence, $\Xi(KB)$ can be saturated by $\mathcal{BS}_{DL}$ in exponential time. Finally, $\mathcal{BS}_{DL}$ is sound and complete, so $\Xi(KB)$ is satisfiable if and only if the empty closure is derived in the saturation. □

6 Removing the Restriction to Leaf Roles

Without the restriction to leaf roles, saturation of $\Xi(KB)$ by $\mathcal{BS}_{DL}$ can produce a closure set whose closures correspond in structure to Table 4, but for which Conditions (*iii*)–(*vii*) do not hold. For example, consider a KB containing axioms (7)–(15) and the corresponding set $\Xi(KB)$:

(7)	$R \sqsubseteq T$	$\neg R(x,y) \vee T(x,y)$
(8)	$S \sqsubseteq T$	$\neg S(x,y) \vee T(x,y)$
(9)	$C \sqsubseteq \exists R.\top$	$\neg C(x) \vee R(x, f(x))$
(10)	$\top \sqsubseteq \exists S^-.\top$	$S^-(x, g(x))$
(11)	$\top \sqsubseteq \leq 1\, T$	$\neg T(x,y_1) \vee \neg T(x,y_2) \vee y_1 \approx y_2$
(12)	$\exists S.\top \sqsubseteq D$	$\neg S(x,y) \vee D(x)$
(13)	$\exists R.\top \sqsubseteq \neg D$	$\neg R(x,y) \vee \neg D(x)$
(14)	$\top \sqsubseteq C$	$C(x)$
(15)		$\neg S^-(x,y) \vee S(y,x)$

Consider a saturation of $\Xi(KB)$ by $\mathcal{BS}_{DL}$; the notation R(xx;yy) means that a closure is derived by (hyper)resolving closures xx and yy:

(16)	$S([g(x)], x)$	R(10;15)

(17)	$\neg C(x) \vee T(x, [f(x)])$	R(7;9)
(18)	$T([g(x)], x)$	R(8;16)
(19)	$\neg C([g(x)]) \vee [f(g(x))] \approx x$	R(11;17;18)

Condition (*vi*) from Table 4 is not satisfied for the closure (19): we have $\mathsf{role}(f) = R \neq \mathsf{Inv}(\mathsf{role}(g)) = \mathsf{Inv}(S^-) = S$. This is because a number restriction occurs in (11) on a role that is not a leaf role. Now (19) can be superposed into (9), resulting in (20):

$$\neg C([g(x)]) \vee R([g(x)], x) \tag{20}$$

No closure exists that subsumes (20); notice that this does not contradict Corollary 12 since (19) does not satisfy Condition (*vi*).

Hence, we must keep (20), which is obviously not of the form from Table 4. This might cause termination problems since, in general, (20) might be resolved with some closure of type 6 of the form $C([g(h(x))])$, producing a closure of the form $R([g(h(x))], [h(x)])$. The term depth in a binary literal is now two, and resolving such a closure with a closure of type 7 can yield closures with even deeper terms. Hence, the number of closures that can be derived is no longer finite and, indeed, saturation does not terminate.

We did not find a way to refine the ordering or the selection function that would solve this problem. Furthermore, (20) is necessary for completeness. The knowledge base KB is unsatisfiable, and the empty closure is derived by the following derivation, which involves (20):

(21)	$D([g(x)])$	R(12;16)
(22)	$\neg D([g(x)]) \vee \neg C([g(x)])$	R(13;20)
(23)	$\neg C([g(x)])$	R(21;22)
(24)	$\square$	R(14;23)

6.1 Transformation by Decomposition

To solve the outlined termination problem, we introduce *decomposition*—a transformation that can be applied to the results of certain $\mathcal{BS}$ inferences. It is generally applicable and is not limited to DL reasoning. Here, we show that it can be combined with basic superposition but, in a similar vein, it can be combined with any clausal calculus compatible with the standard redundancy notion [10]. Unlike the inference rules of $\mathcal{BS}$, decomposition can extend the signature of a closure set with new predicate symbols. In order to reuse these symbols whenever possible, an application of decomposition depends on the

previous applications of decomposition in a derivation, and not only on the current closure set.

In the following, for $\mathbf{x}$ a vector of distinct variables $x_1, \ldots, x_n$ and $\mathbf{t}$ a vector of (not necessarily distinct) terms $t_1, \ldots, t_n$, let $\{\mathbf{x} \mapsto \mathbf{t}\}$ denote the substitution $\{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\}$, and let $Q([\mathbf{t}])$ denote $Q([t_1], \ldots, [t_n])$.

Definition 15 *The closures $C_1 \cdot \rho$ and $C_2 \cdot \theta$ and a vector of m terms $\mathbf{t}$ constitute a* decomposition *of a closure $C \cdot \rho$ if $C \cdot \rho = C_1 \cdot \rho \vee C_2 \cdot \theta\{\mathbf{x} \mapsto \mathbf{t}\}$, where $\mathbf{x}$ is a vector of m free variables of $C_2\theta$. The closure $C_2 \cdot \theta$ is called the* fixed *part and the closure $C_1 \cdot \rho$ is called the* variable *part.*

Let $N_0, N_1, \ldots, N_n \cup \{C \cdot \rho\}$ be a $\mathcal{BS}$ derivation and $C_1 \cdot \rho$, $C_2 \cdot \theta$, and $\mathbf{t}$ a decomposition of $C \cdot \rho$. Then, an application of decomposition *to $C \cdot \rho$ produces the closure set $N_n \cup \{C_1 \cdot \rho \vee Q([\mathbf{t}]),\ \neg Q(\mathbf{x}) \vee C_2 \cdot \theta\}$. If decomposition has previously been applied in the derivation with the same variable part $C_2 \cdot \theta$, then Q is reused; otherwise, it is a fresh predicate. The predicate Q is called the* definition predicate. *The ordering $\prec$ used to parameterize the inferences of $\mathcal{BS}$ must be admissible over the signature that includes all definition predicates. An application of decomposition is written as follows:*

$$C \cdot \rho \rightsquigarrow \begin{array}{l} C_1 \cdot \rho \ \vee\ Q([\mathbf{t}]) \\ C_2 \cdot \theta \ \vee\ \neg Q(\mathbf{x}) \end{array}$$

Let ξ be a $\mathcal{BS}$ inference with a side premise $D_s \cdot \eta$, a main premise $D_m \cdot \eta$, a most general unifier σ, and a conclusion $C \cdot \rho$; furthermore, let $L_m \cdot \eta$ be the literal from $D_m \cdot \eta$ on which the inference takes place. Then, $C \cdot \rho$ is eligible for decomposition *with a definition predicate Q if, for each ground substitution τ such that $\xi\tau$ satisfies the ordering constraints of $\mathcal{BS}$, we have $\neg Q(\mathbf{t})\tau \prec L_m\eta\sigma\tau$. If the conclusion of a $\mathcal{BS}$ inference ξ is eligible for decomposition, we say that ξ itself is eligible for decomposition. Finally, $\mathcal{BS}^+$ is the $\mathcal{BS}$ calculus in which conclusions of eligible inferences are possibly decomposed.*

For example, $[f(g(x))] \approx [h(g(x))]$ can be superposed into $C(x) \vee R(x, f(x))$, resulting in $C([g(x)]) \vee R([g(x)], [h(g(x))])$. This closure is not of type from Table 4; however, it can be decomposed into $C([g(x)]) \vee Q_{R,f}([g(x)])$ and $\neg Q_{R,f}(x) \vee R(x, [h(x)])$, which are both of types from Table 4. The inference is eligible for decomposition if we parameterize basic superposition with a term ordering such that $\neg Q_{R,f}(g(x)) \prec R(g(x), f(g(x)))$.

We now show that decomposition is sound. Since decomposition can reuse the predicate Q in a derivation, soundness must be considered w.r.t. a derivation, and not only w.r.t. a single inference.

Lemma 16 *If a set of closures N_0 is satisfiable, then each set N_i in a derivation $N_0, N_1, \ldots, N_i$ by $\mathcal{BS}^+$ is satisfiable as well.*

PROOF. We inductively construct a model I_i of N_i satisfying the following property (*): each definition predicate Q is interpreted exactly as the corresponding variable part $C_2\theta$. For the base case, any model I_0 of N_0 satisfies (*) since N_0 does not contain definition predicates. For the induction step, we consider the inference steps deriving a closure $C \cdot \rho$ from premises in N_{i-1}, and possibly decomposing $C \cdot \rho$ into $C_1 \cdot \rho \vee Q([\mathbf{t}])$ and $\neg Q(\mathbf{x}) \vee C_2 \cdot \theta$. The inference rules of $\mathcal{BS}$ ensure that $C \cdot \rho$ is true in I_{i-1}.

- If $C \cdot \rho$ is not decomposed, then $I_i := I_{i-1}$ is a model of N_i satisfying (*).
- If $C \cdot \rho$ is decomposed and the predicate Q is not new, we set $I_i := I_{i-1}$. Now $C \cdot \rho$ is true in I_{i-1}, and, because I_{i-1} satisfies (*) by induction assumption, $C_1 \cdot \rho \vee Q([\mathbf{t}])$ is true in I_i. Furthermore, I_i obviously satisfies (*).
- If $C \cdot \rho$ is decomposed and the predicate Q is new, then we extend I_{i-1} to I_i by making $Q(\mathbf{x})$ true exactly for those $\mathbf{x}$ for which $C_2\theta$ is true in I_{i-1}. Clearly, I_i is a model of N_i and it satisfies (*).

□

We next show that decomposition is compatible with the standard notion of redundancy. This is the key step in showing completeness of $\mathcal{BS}^+$.

Lemma 17 *Let ξ be a $\mathcal{BS}$ inference applied to premises from a closure set N, resulting in a closure $C \cdot \rho$. If ξ is eligible for decomposition of $C \cdot \rho$ into $C_1 \cdot \rho \vee Q([\mathbf{t}])$ and $C_2 \cdot \theta \vee \neg Q(\mathbf{x})$, and the two latter closures are both redundant in N, then ξ is redundant in N as well.*

PROOF. Let ξ be an inference on a literal $L_m \cdot \eta$ from a main premise $D_m \cdot \eta$ and a side premise $D_s \cdot \eta$, with a most general unifier σ, resulting in $C \cdot \rho$. Furthermore, let R be a rewrite system and τ a ground substitution such that $\xi\tau$ satisfies the ordering constraints of $\mathcal{BS}$ and $\xi\tau$ is a variable irreducible ground instance of ξ w.r.t. R. Finally, let $E_1 = (C_1 \cdot \rho \vee Q([\mathbf{t}]))\tau$ and $E_2 = (\neg Q(\mathbf{x}) \vee C_2 \cdot \theta)\{\mathbf{x} \mapsto \mathbf{t}\}\tau$. Note that $\max(D_m \cdot \eta\sigma\tau, D_s \cdot \eta\sigma\tau) = D_m \cdot \eta\sigma\tau$, so let $D = D_m \cdot \eta\sigma\tau$.

By the ordering constraints of $\mathcal{BS}$ inference rules, $D_s\eta\sigma\tau \prec L_m\eta\sigma\tau$. Furthermore, superposition inferences are allowed only from the maximal side of an equality, so the inference always produces a literal $L'\eta\sigma\tau \prec L_m\eta\sigma\tau$. Finally, because ξ is eligible for decomposition, $\neg Q(\mathbf{t})\tau \prec L_m\eta\sigma\tau$. Thus, if a literal $L\eta\sigma\tau \succ L_m\eta\sigma\tau$ occurs n times in $E_1 \cup E_2$, then it also occurs n times in D. In other words, both E_1 and E_2 contain at most those literals larger than $L_m\eta\sigma\tau$

that also occur in D. All other literals in E_1 or E_2 are smaller than $L_m\eta\sigma\tau$. Since $L_m\eta\sigma\tau \in D$, we conclude that $E_1 \prec D$ and $E_2 \prec D$.

The vector of terms $\mathbf{t}$ is "extracted" from the substitution part of $C \cdot \rho$. Hence, if a term t occurs in E_1 and E_2 at a substitution position, then t occurs in $C \cdot \rho\tau$ also at a substitution position. Therefore, if $\xi\tau$ is variable irreducible w.r.t. R, so is $C \cdot \rho\tau$, and so are E_1 and E_2.

By assumption, E_1 is redundant in N, so $R \cup \mathsf{irred}_R(N)^{\prec E_1} \models E_1$; since $E_1 \prec D$, we have $R \cup \mathsf{irred}_R(N)^{\prec D} \models E_1$. Similarly, $R \cup \mathsf{irred}_R(N)^{\prec D} \models E_2$. Since $\{E_1, E_2\} \models C \cdot \rho\tau$, we have $R \cup \mathsf{irred}_R(N)^{\prec D} \models C \cdot \rho\tau$. $\square$

Soundness and compatibility with the standard notion of redundancy imply that $\mathcal{BS}^+$ is a sound and complete calculus, as shown by Theorem 18. To obtain the saturated set N, we can use any fair saturation strategy [10].

Theorem 18 *For N_0 a set of closures of the form $C \cdot \{\}$, let N be a set of closures obtained by saturating N_0 under $\mathcal{BS}^+$. Then, N_0 is satisfiable if and only if N does not contain the empty closure.*

PROOF. The ($\Rightarrow$) direction follows immediately from Lemma 16. For the ($\Leftarrow$) direction, assume that N is saturated under $\mathcal{BS}^+$. Then, by Lemma 17, N is saturated under $\mathcal{BS}$ as well. Using the model generation method [16,17], we can build a rewrite system R such that $R^* \models \mathsf{irred}_R(N)$. Unlike for basic superposition without decomposition, the set of closures N does not need to be well-constrained, so we cannot immediately conclude that R^* is a model of N. We can, however, conclude that $R^* \models \mathsf{irred}_R(N_0)$: consider a closure $C \in N_0$ and its variable irreducible ground instance $C\tau$. If $C \in N$, then R^* is obviously a model of $C\tau$. Furthermore, $C \notin N$ only if it is redundant in N; then, for any τ, there are ground closures $D_i\tau \in \mathsf{irred}_R(N)$ such that $D_1\tau, \ldots, D_n\tau \models C\tau$. Since $R^* \models D_i\tau$ by assumption, we have $R^* \models C\tau$.

Now consider a closure $C \in N_0$ and its (not necessarily variable irreducible) ground instance $C\eta$. Let η' be a substitution obtained by replacing each mapping $x \mapsto t$ in η with $x \mapsto t'$, where t' is the normal form of t w.r.t. R. Since the substitution part of C is empty, $C\eta' \in \mathsf{irred}_R(N_0)$, so $R^* \models C\eta'$ implies $R^* \models C\eta$. Hence, $R^* \models N_0$, and by Lemma 16, there is a model of N. $\square$

Discussion. Decomposition is essentially structural transformation applied in the course of a theorem proving process. Since the formulae obtained by structural transformation are equisatisfiable with the original formula, decomposition is sound.

A potential problem might be that decomposition somehow interferes with the markers of basic superposition. This does not occur because, for any rewrite system R, decomposing $C \cdot \rho$ into $C_1 \cdot \rho \vee Q([\mathbf{t}])$ and $\neg Q(\mathbf{x}) \vee C_2 \cdot \theta$ actually decomposes any variable irreducible ground instance of the premise into corresponding variable irreducible ground instances of the conclusions. Hence, for each variable irreducible ground instance of the premise, there are variable irreducible ground instances of the conclusions that imply the ground instance of the premise in question. Therefore, we do not lose any variable irreducible ground instance used in detecting a potential inconsistency of the closure set. Note that, for an arbitrary rewrite system R, the closures $C_1 \cdot \rho \vee Q([\mathbf{t}])$ and $\neg Q(\mathbf{x}) \vee C_2 \cdot \theta$ can have variable irreducible ground instances that do not correspond to a variable irreducible ground instance of $C \cdot \rho$. These variable irreducible ground instances, however, do not cause problems since decomposition is sound.

Another potential problem might arise if a closure $C \cdot \rho$ is derived and decomposed into $C_1 \cdot \rho \vee Q([\mathbf{t}])$ and $\neg Q(\mathbf{x}) \vee C_2 \cdot \theta$ an infinite number of times. This might happen if the ordering constraints on predicates require the fixed and the variable parts to be resolved on $Q([\mathbf{t}])$ and $\neg Q(\mathbf{x})$: obviously, the theorem proving process would be stuck in an infinite loop. This is avoided by requiring an inference to be eligible for decomposition, which makes decomposition compatible with the standard notion of redundancy. Hence, the fixed and the variable parts together make the original inference redundant, so the inference need not be repeated in a derivation.

The completeness argument for decomposition relies on the standard completeness argument of basic superposition, which in turn relies on the fact that the ordering $\prec$ used in $\mathcal{BS}^+$ is admissible. This must hold for the entire signature, including not only the predicates and function symbols used in the original closure set, but also all the definition predicates introduced in the saturation. It this condition is satisfied, decomposition can even be applied an infinite number of times in a saturation. As long as each application of decomposition satisfies the eligibility condition, the final set will be saturated, and we can generate a model using the standard model construction method of basic superposition.

If decomposition is used to obtain a decision procedure, as it is the case in Section 6.2, then the number of the introduced definition predicates must be bounded. Thus, the entire signature is bounded, so an ordering $\prec$ admissible on that signature can be readily constructed.

Notion of Eligibility. Eligibility, as defined in Definition 15, ensures that the closures obtained by decomposing the conclusion of an inference ξ are smaller than the main premise of ξ. Consider the following $\mathcal{BS}$ inference ξ,

followed by decomposition (the unifier is $\sigma = \{x' \mapsto x\}$):

$$\frac{\neg A(x) \vee B(x) \vee C(y) \vee D(y) \quad A(x') \vee E(x')}{B(x) \vee E(x) \vee C(y) \vee D(y) \qquad \rightsquigarrow \begin{array}{l} B(x) \vee C(y) \vee Q(x,y) \\ \neg Q(x,y) \vee E(x) \vee D(y) \end{array}}$$

To check if ξ is eligible for decomposition, we have a problem: $\neg A(x)$ is the main literal on which the inference takes place, and it is not comparable with $\neg Q(x,y)$ since $Q(x,y)$ contains an additional variable y. The remedy is to consider each ground substitution τ such that $\xi\tau$ satisfies the ordering constraints of $\mathcal{BS}$. If $\neg A(x)$ is not selected, $\neg A(x)\sigma\tau \succ (B(x) \vee C(y))\sigma\tau$ must hold, implying that $x\sigma\tau \succ y\sigma\tau$. If we compare literals, say, using an LPO in which $A > Q$, then $\neg A(x)\sigma\tau \succ \neg Q(x,y)\sigma\tau$, so the eligibility condition is satisfied. On the other hand, assume that $\neg A(x)$ is selected. Then, $\neg A(x)\sigma\tau$ is not necessarily larger than $(B(x) \vee C(y))\sigma\tau$. Therefore, we cannot conclude that $x\sigma\tau \succ y\sigma\tau$, and that $\neg A(x)\sigma\tau \succ \neg Q(x,y)\sigma\tau$. Indeed, for $\tau = \{x \mapsto a, y \mapsto f(a)\}$, we have $\neg A(x)\sigma\tau \prec Q(x,y)\sigma\tau$ even if the term ordering is based on an LPO in which $A > Q$; hence, the eligibility condition is not satisfied. Next, we present two simpler eligibility tests:

Proposition 19 *Let ξ be an inference by $\mathcal{BS}$ as in Definition 15. Then, ξ is eligible for decomposition if*

(1) $\neg Q(\mathbf{t}) \prec L_m\eta\sigma$, or
(2) the side premise $D_s \cdot \eta$ contains a literal $L \cdot \eta$ such that $\neg Q(\mathbf{t}) \prec L\eta\sigma$.

PROOF. (1) Since $\prec$ is stable under substitutions, we have $\neg Q(\mathbf{t})\tau \prec L_m\eta\sigma\tau$ for each τ.

(2) For ξ a $\mathcal{BS}$ inference and τ a ground substitution as in Definition 15, the ordering conditions of $\mathcal{BS}$ ensure that $L_s\eta\sigma\tau \prec L_m\eta\sigma\tau$, where $L_s \cdot \eta$ is the maximal literal of the side premise. Because $\prec$ is stable under substitutions, the assumption implies $\neg Q(\mathbf{t})\tau \prec L\eta\sigma\tau \preceq L_s\eta\sigma\tau \prec L_m\eta\sigma\tau$. □

Combining Decomposition with Other Calculi. Lemma 16 applies to any sound clausal calculus. Furthermore, for any calculus compatible with the standard notion of redundancy [10], Lemma 17 can be proved in a similar manner with minor modifications.

We now extend the decision procedure from Section 5 with the decomposition rule from Section 6.1 to obtain an algorithm for $\mathcal{ALCHIQ}$, and, by the results from Section 4, for $\mathcal{SHIQ}$ as well.

Definition 20 *Let $\mathcal{BS}^{+}_{DL}$ be the $\mathcal{BS}_{DL}$ calculus where conclusions are, whenever possible, decomposed according to the following pattern, for any term t:*

$$D \cdot \rho \vee R([t], [f(t)]) \rightsquigarrow \begin{array}{c} D \cdot \rho \vee Q_{R,f}([t]) \\ \neg Q_{R,f}(x) \vee R(x, [f(x)]) \end{array}$$

$$D \cdot \rho \vee R([f(x)], x) \rightsquigarrow \begin{array}{c} D \cdot \rho \vee Q_{\mathsf{Inv}(R),f}(x) \\ \neg Q_{\mathsf{Inv}(R),f}(x) \vee R([f(x)], x) \end{array}$$

In addition to the requirements of Definition 9, the term ordering $\succ$ must be such that $R(x, f(x)) \succ \neg Q_{S,g}(x)$ and $f(x) \succ x$ for each binary predicate R, function symbol f, and definition predicate $Q_{S,g}$.

A term ordering satisfying Definition 20 can be obtained by instantiating an LPO with a precedence such that $f > c > P > \mathsf{tt}$, for each function symbol f, constant c, and predicate P (including definition predicates). Note that the number of definition predicates $Q_{R,f}$ is quadratic in the size of the initial closure set $\Xi(KB)$, so such an LPO indeed exists.

By Definition 15, for a (possibly inverse) role S and a function symbol f, the definition predicate $Q_{S,f}$ is unique. Furthermore, a strict application of Definition 15 would require introducing a distinct definition predicate $Q'_{R,f}$ for $R([f(x)], x)$. By the definition of the operator π for translating KB into first-order logic, however, $R([f(x)], x)$ and $\mathsf{Inv}(R)(x, [f(x)])$ are logically equivalent. Therefore, $Q_{\mathsf{Inv}(R),f}$ can be used as the definition predicate for $R([f(x)], x)$ instead of $Q'_{R,f}$, thus avoiding the need to introduce an additional predicate in the second form of decomposition in Definition 20. This optimization is not essential for the correctness of our results; however, it is good practice to keep the number of predicate symbols minimal. We next relax the conditions of $\mathcal{ALCHIQ}^{-}$-closures to $\mathcal{ALCHIQ}$-closures:

Definition 21 *A set of closures N is a set of* $\mathcal{ALCHIQ}$-closures *if each closure in N is of the form from Table 4 and it satisfies Conditions (*i*)–(*ii*).*

As we shall see next, decomposition ensures that all non-$\mathcal{ALCHIQ}$-closures derived in a saturation are decomposed into $\mathcal{ALCHIQ}$-closures. Furthermore, since the definition predicate $Q_{R,f}$ is unique for a pair of role and function symbols R and f, at most a polynomial number of definition predicates is introduced during saturation. Since the number of $\mathcal{ALCHIQ}$-closures is finite

according to Lemma 13, saturation by $\mathcal{BS}^+_{DL}$ terminates.

Theorem 22 *For an $\mathcal{ALCHIQ}$ knowledge base KB, saturation of $\Xi(KB)$ by $\mathcal{BS}^+_{DL}$ decides satisfiability of KB, and runs in time exponential in $|KB|$ for unary coding of numbers.*

PROOF. We first show that inferences of $\mathcal{BS}^+_{DL}$, when applied to $\mathcal{ALCHIQ}$-closures, always produce $\mathcal{ALCHIQ}$-closures. The proof of Lemma 11 applies even if Conditions (*iii*)–(*vii*) from Table 4 do not hold; the only exception is a superposition into a generator closure $\mathbf{P^f}(x) \vee R(x, f(x))$ with the maximal literal $L_m\eta = R(x, f(x))$ and a most general unifier σ. There are three different types of such inferences, depending on the structure of the premise that superposition is performed from:

- For $[f(t)] \approx [g(t)] \vee D \cdot \rho$ of type 5, 6, or 8, where t is either a variable x', a term $h(x')$, or a constant a, we get $D \cdot \rho \vee \mathbf{P^f}([t]) \vee R([t], [g(t)])$. This conclusion is decomposed into a closure $\neg Q_{R,g}(x) \vee R(x, [g(x)])$ of type 3 and a closure $D \cdot \rho \vee \mathbf{P^f}([t]) \vee Q_{R,g}([t])$ of type 5, 6, or 8. Clearly, $L_m\eta\sigma = R(t, f(t))$. By Definition 20, $R(x, f(x)) \succ \neg Q_{R,g}(x)$. Since $\succ$ is stable under substitutions, $R(t, f(t)) \succ \neg Q_{R,g}(t)$ as well. Hence, the inference is eligible for decomposition by Proposition 19.
- For $[f(g(x'))] \approx x' \vee D \cdot \rho$ of type 6, we get $D \cdot \rho \vee \mathbf{P^f}([g(x')]) \vee R([g(x')], x')$. This conclusion is decomposed into a closure $\neg Q_{\mathsf{Inv}(R),g}(x) \vee R(x, [g(x)])$ of type 4 and a closure $D \cdot \rho \vee \mathbf{P^f}([g(x')]) \vee Q_{\mathsf{Inv}(R),g}(x')$ of type 5 or 6. Since $R([g(x')], x')$ and $\mathsf{Inv}(R)(x', [g(x')])$ are logically equivalent due to the translation operator π, the predicate $Q_{\mathsf{Inv}(R),g}$ can be used as the definition predicate for $R([g(x')], x')$. Clearly, $L_m\eta\sigma = R(g(x'), f(g(x')))$. By Definition 20, $R(x', f(x')) \succ \neg Q_{\mathsf{Inv}(R),g}(x')$. Each reduction ordering is *stable under contexts* [19]: $s \succ t$ implies $u[s]_p \succ u[t]_p$ for all terms s, t, u and positions p. Thus, $g(x') \succ x'$ implies $R(g(x'), f(g(x'))) \succ R(g(x'), f(x')) \succ R(x', f(x'))$, so $R(g(x'), f(g(x'))) \succ \neg Q_{\mathsf{Inv}(R),g}(x')$, and the inference is eligible for decomposition by Proposition 19.
- Superposition from a closure of type 8 of the form $[f(a)] \approx [b] \vee D \cdot \rho$ results in a closure of the form $\mathbf{P^f}(a) \vee R([a], [b]) \vee D \cdot \rho$, which is of type 8.

Hence, the conclusion of each inference of $\mathcal{BS}^+_{DL}$ is an $\mathcal{ALCHIQ}$-closure.

Let r be the number of roles and f the number of function symbols occurring in $\Xi(KB)$; as in Lemma 13, both r and f are linear in $|KB|$ for unary coding of numbers. The number of definition predicates $Q_{R,f}$ introduced by decomposition is bounded by $r \cdot f$, which is quadratic in $|KB|$, so the number of different predicates is polynomial in $|KB|$. Hence, Lemma 13 applies in this case as well, which, together with Theorem 18, implies the claim of this theorem. □

Note that Definition 20 applies decomposition eagerly; however, in some cases, decomposition can be made optional. For example, a resolution of a closure of type 3 or 4 with a closure of type 1 or 2 produces a closure of type 3; similarly, a superposition from $[f(x)] \approx [g(x)] \vee C(x)$ into $D(x) \vee R(x, f(x))$ produces $D(x) \vee C(x) \vee R(x, [g(x)])$, which is also of type 3. Decomposing such a closure is not strictly necessary to obtain termination.

7 Optimizations of Clausification

To obtain a practical procedure, it is important keep the input closure set as small as possible. In this section, we present several optimizations that were crucial to obtain an algorithm capable of solving practical problems [25].

Optional Positions. Certain complex concepts need not be replaced with an atomic concept in the clausification process. For example, by Definition 6, $\Theta(A \sqsubseteq \exists R.B) = \{A \sqsubseteq \exists R.Q_1,\ Q_1 \sqsubseteq B\}$; that is, the concept B is replaced with Q_1. Clausifying $A \sqsubseteq \exists R.B$, however, produces closures of the form from Table 4 even if B is not replaced with Q_1.

Definition 23 Optimized structural transformation *is obtained by modifying the definition of the operator* Θ *from Table 3 such that*

$$\Theta(L \sqsubseteq C \sqcup \bigsqcup L_i) = \{L \sqsubseteq C \sqcup \bigsqcup L_i\}$$

if C *contains only literal subconcepts and* $L_{(i)}$ *are literal concepts.*

Proposition 24 *Each closure in* $\Xi(KB)$ *is of type from Table 4, even if* $\Theta(KB)$ *is computed by using the optimized structural transformation.*

Renaming concepts can be beneficial. Consider the knowledge base KB, consisting of axioms shown in (25)–(27) on the left-hand side:

$$A \sqsubseteq \exists R.C \quad \rightsquigarrow \quad \begin{array}{l} \neg A(x) \vee R(x, f(x)) \\ \neg A(x) \vee C(f(x)) \end{array} \tag{25}$$

$$B \sqsubseteq \exists R.C \quad \rightsquigarrow \quad \begin{array}{l} \neg B(x) \vee R(x, g(x)) \\ \neg B(x) \vee C(g(x)) \end{array} \tag{26}$$

$$D \sqsubseteq \forall R.C \quad \rightsquigarrow \quad \neg D(x) \vee \neg R(x, y) \vee \neg C(y) \tag{27}$$

In (25) and (26), the concepts $\exists R.C$ need not be renamed, so KB can be clausified without introducing new predicates. This yields closures shown in (25)–(27) on the right-hand side.

The concept $\exists R.C$ is now skolemized twice, yielding $\neg A(x) \vee R(x, f(x))$ and $\neg B(x) \vee R(x, g(x))$ as candidates for an inference with (27). Additional axioms $A_i \sqsubseteq \exists R.C$ would produce additional closures $\neg A_i(x) \vee R(x, f(x))$, which could participate in an inference with (27), thus increasing the search space.

The search space can be reduced by renaming $\exists R.C$, even though this is not strictly necessary. The axioms and the closures obtained by applying the structural transformation to KB are shown in (28)–(31):

$$A \sqsubseteq Q \quad \rightsquigarrow \quad \neg A(x) \vee Q(x) \tag{28}$$

$$B \sqsubseteq Q \quad \rightsquigarrow \quad \neg B(x) \vee Q(x) \tag{29}$$

$$Q \sqsubseteq \exists R.C \quad \rightsquigarrow \quad \begin{array}{l} \neg Q(x) \vee R(x, f(x)) \\ \neg Q(x) \vee C(g(x)) \end{array} \tag{30}$$

$$D \sqsubseteq \forall R.C \quad \rightsquigarrow \quad \neg D(x) \vee \neg R(x, y) \vee \neg C(y) \tag{31}$$

This set contains only one closure $\neg Q(x) \vee R(x, f(x))$, which can participate in an inference with (31); also, any additional axiom $A_i \sqsubseteq \exists R.C$ produces only a closure $\neg A_i(x) \vee Q(x)$. Note that renaming $\exists R.C$ pays off because the concept occurs in KB twice.

Hence, Definition 23 can be modified to rename the concept C if it occurs in KB more than once. Tableau algorithms use similar techniques, such as lazy unfolding and introducing new names for early clash detection [3,31].

Functional Roles. Let KB be the knowledge base consisting of the axioms shown on the left-hand side of (32)–(34):

$$\top \sqsubseteq \leq 1\, R \quad \rightsquigarrow \quad \neg R(x, y_1) \vee \neg R(x, y_2) \vee y_1 \approx y_2 \tag{32}$$

$$A \sqsubseteq \exists R.C \quad \rightsquigarrow \quad \begin{array}{l} \neg A(x) \vee R(x, f(x)) \\ \neg A(x) \vee C(f(x)) \end{array} \tag{33}$$

$$B \sqsubseteq \exists R.D \quad \rightsquigarrow \quad \begin{array}{l} \neg B(x) \vee R(x, f(x)) \\ \neg B(x) \vee D(f(x)) \end{array} \tag{34}$$

The role R is functional by (32), which means that an object in a model can have at most one R-successor. This allows us to use the same function symbol f in skolemizing $\exists R.C$ and $\exists R.D$, yielding closures (32)–(34) shown on the right-hand side.

The main benefit of such clausification is that resolving $\neg A(x) \vee R(x, f(x))$ and $\neg B(x) \vee R(x, f(x))$ with (32) produces a closure with a literal $f(x) \approx f(x)$,

which is a tautology. Without reusing function symbols, clausifying (33) produces $\neg B(x) \vee R(x, g(x))$, which, resolved with $\neg A(x) \vee R(x, f(x))$ and (32), produces a closure containing a literal $f(x) \approx g(x)$. Such a closure is not a tautology and it participates in further inferences.

Definition 25 Clausification with optimized skolemization *is obtained from Definition 6 such that, if* $\top \sqsubseteq \leq 1\, R \in KB$*, then the existential quantifiers in* $\exists R.C$ *and* $\geq n\, R.C$ *are skolemized using a function symbol* f_R *unique for* R*.*

Proposition 26 *KB and* $\Xi(KB)$ *are equisatisfiable, even if clausification with optimized skolemization is used.*

8 Related Work

Joyner has established the basic principles for deriving resolution-based decision procedures [20]. He observed that, if clauses derivable in a saturation by a resolution refinement have a bounded term depth and clause length, then saturation necessarily terminates. By choosing appropriate refinements, he presented decision procedures for the Ackermann class, the Monadic class, and the Maslov class. Joyner's approach was applied to numerous other decidable classes such as $\mathbf{E}^+$ [32], $\mathcal{PVD}$ [33], and $\mathcal{PVD}^g_=$ [34], to name just a few. An overview of these results is given in [35].

De Nivelle, Hustadt, and Schmidt studied extensively the decidability of description logics in the resolution framework [29,36,21]. They embed the description logic $\mathcal{ALB}$ into the DL* clausal class, which they decide using the resolution framework by Bachmair and Ganzinger [10]. The main advantage of using this framework lies in its effective redundancy elimination methods, which were shown to be essential for the practical applicability of resolution calculi. $\mathcal{ALB}$ is a very expressive logic: it allows for Boolean role expressions and inverse roles; however, it does not provide for counting quantifiers.

Ganzinger, Hustadt, Meyer, and Schmidt developed a decision procedure for the modal logic with a single transitive modality K4 [37]. To deal with transitivity, the algorithm is based on the ordered chaining calculus [38]. This calculus consists of inference rules aimed at optimizing theorem proving with chains of binary roles. Unfortunately, our attempts to decide $\mathcal{SHIQ}$ using ordered chaining proved unsuccessful due to certain negative chaining inferences that produced undesirable equality literals. Therefore, we adopted the approach for eliminating transitivity presented in Section 4.

Andréka, van Benthem, and Németi introduced the guarded fragment to explain and generalize the good properties of modal and description logic, such

as decidability [39]. De Nivelle presented a decision procedure by resolution using a nonliftable ordering [40]; this approach was later modified to handle the (loosely) guarded fragment with equality [41] by basing the algorithm on superposition [42]. Since the basic description logic $\mathcal{ALC}$ is actually a syntactic variant of the multi-modal logic K_m [43], it can be embedded into the guarded fragment and decided using an algorithm by Ganzinger and de Nivelle [41]. Using the approach by Hustadt and Schmidt, certain extensions of $\mathcal{ALC}$—such as role transitivity—can be encoded into $\mathcal{ALC}$ knowledge bases [23], so the algorithm by Ganzinger and de Nivelle can decide these extensions as well. $\mathcal{SHIQ}$ is, however, not a fragment of the (loosely) guarded fragment because of the counting quantifiers: equality is available in the guarded fragment, but each two pairs of free variables of a guarded formula must occur in a guard atom. In fact, Hodkinson has shown that the guarded fragment has the finite-model property [44], which does not hold for $\mathcal{SHIQ}$ [3], and Grädel et. al have shown that combining the guarded fragment with counting quantifiers or transitive roles leads to undecidability [45]—thus suggesting that other mechanisms are necessary for handling the latter logic.

$\mathcal{SHIQ}$ can easily be embedded into the two-variable fragment of first-order logic with counting quantifiers $\mathcal{C}^2$. This fragment was shown to be decidable [45,46], and a decision procedure based on a combination of resolution and integer programming was presented by Pratt-Hartmann [47]. Deciding satisfiability of $\mathcal{C}^2$, however, is NExpTime-complete [46], and $\mathcal{SHIQ}$ is an ExpTime-complete logic [6]. Thus, the decision procedure by Pratt-Hartmann is not worst-case optimal for $\mathcal{SHIQ}$.

The decomposition rule is closely related to structural transformation [26–28]. Structural transformation is, however, usually applied as a preprocessing step and not in the theorem proving process. De Nivelle [48] and Riazanov and Voronkov [49] proposed *splitting by propositional symbols*, which can split variable-disjoint subsets of a clause and connect them by a propositional symbol. Hustadt and Schmidt introduced the *separation* rule to decide fluted logic [9], which is a generalisation of splitting by propositional symbols. They show that resolution remains complete if the separation rule is applied a finite number of times during saturation. In contrast to these related approaches, our rule decomposes complex terms. Moreover, we demonstrate compatibility of the decomposition rule with the standard redundancy notion. Finally, extending basic superposition with decomposition is not trivial, due to the nonstandard approach to lifting employed by basic superposition.

9 Conclusion

In this paper, we presented a worst-case optimal decision procedure for reasoning in the description logic $\mathcal{SHIQ}$. The algorithm is based on basic superposition and is, to the best of our knowledge, the first decision procedure based on that calculus. Basic superposition alone decides only the slightly weaker logic $\mathcal{SHIQ}^-$, in which number restrictions are allowed only on roles that do not have subroles. To obtain a decision procedure for full $\mathcal{SHIQ}$, we introduced *decomposition*—a new inference rule that can be used to simplify the conclusions of some inferences. This is a general inference rule, and it is not restricted to DL reasoning. Furthermore, it can be combined with any calculus compatible with the standard notion of redundancy.

The practicability of our algorithms has been confirmed by our implementation in a DL reasoner KAON2 and experiments, as discussed in [25]. Hence, we believe this algorithm to be an important step towards obtaining a practical alternative to tableau calculi, capable of reasoning over DL knowledge bases with large ABoxes.

Acknowledgment

We thank the anonymous reviewer for many useful suggestions for improvement of this paper.

References

[1] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, P. F. Patel-Schneider (Eds.), The Description Logic Handbook: Theory, Implementation and Applications, Cambridge University Press, 2003.

[2] P. F. Patel-Schneider, P. Hayes, I. Horrocks, OWL Web Ontology Language: Semantics and Abstract Syntax, W3C Recommendation, `http://www.w3.org/TR/owl-semantics/` (February 10 2004).

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

[4] I. Horrocks, Optimising Tableaux Decision Procedures for Description Logics, Ph.D. thesis, University of Manchester, UK (1997).

[5] V. Haarslev, R. Möller, RACER System Description, in: R. Goré, A. Leitsch, T. Nipkow (Eds.), Proc. of the 1st Int. Joint Conf. on Automated Reasoning (IJCAR 2001), Vol. 2083 of LNAI, Springer, Siena, Italy, 2001, pp. 701–706.

[6] S. Tobies, Complexity Results and Practical Algorithms for Logics in Knowledge Representation, Ph.D. thesis, RWTH Aachen, Germany (2001).

[7] F. M. Donini, F. Massacci, ExpTime Tableaux for $\mathcal{ALC}$, Artificial Intelligence 124 (1) (2000) 87–138.

[8] C. Weidenbach, Combining Superposition, Sorts and Splitting, in: A. Robinson, A. Voronkov (Eds.), Handbook of Automated Reasoning, Vol. II, Elsevier Science, 2001, Ch. 27, pp. 1965–2013.

[9] R. A. Schmidt, U. Hustadt, A Resolution Decision Procedure for Fluted Logic, in: D. McAllester (Ed.), Proc. of the 17th Int. Conf. on Automated Deduction (CADE-17), Vol. 1831 of LNAI, Springer, Pittsburgh, PA, USA, 2000, pp. 433–448.

[10] L. Bachmair, H. Ganzinger, Resolution Theorem Proving, in: A. Robinson, A. Voronkov (Eds.), Handbook of Automated Reasoning, Vol. I, Elsevier Science, 2001, Ch. 2, pp. 19–99.

[11] U. Hustadt, B. Motik, U. Sattler, Reducing $\mathcal{SHIQ}^-$ Description Logic to Disjunctive Datalog Programs, in: D. Dubois, C. A. Welty, M.-A. Williams (Eds.), Proc. of the 9th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2004), AAAI Press, Whistler, Canada, 2004, pp. 152–162.

[12] B. Motik, U. Sattler, A Comparison of Reasoning Techniques for Querying Large Description Logic ABoxes, in: M. Hermann, A. Voronkov (Eds.), Proc. of the 13th Int. Conf. on Logic for Programming Artificial Intelligence and Reasoning (LPAR 2006), Vol. 4246 of LNCS, Springer, Phnom Penh, Cambodia, 2006, pp. 227–241.

[13] U. Hustadt, B. Motik, U. Sattler, A Decomposition Rule for Decision Procedures by Resolution-based Calculi, in: F. Baader, A. Voronkov (Eds.), Proc. of the 11th Int. Conf. on Logic for Programming Artificial Intelligence and Reasoning (LPAR 2004), Vol. 3452 of LNAI, Springer, Montevideo, Uruguay, 2005, pp. 21–35.

[14] D. Calvanese, M. Lenzerini, D. Nardi, Unifying Class-Based Representation Formalisms, Journal of Artificial Intelligence Research (JAIR) 11 (1999) 199–240.

[15] P. P. Chen, The Entity-Relationship Model—Toward a Unified View of Data, ACM Transaction on Database Systems 1 (1) (1976) 9–36.

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

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

[18] N. Dershowitz, D. A. Plaisted, Rewriting, in: A. Robinson, A. Voronkov (Eds.), Handbook of Automated Reasoning, Vol. I, Elsevier Science, 2001, Ch. 9, pp. 535–610.

[19] F. Baader, T. Nipkow, Term Rewriting and All That, Cambridge University Press, 1998.

[20] W. H. Joyner Jr., Resolution Strategies as Decision Procedures, Journal of the ACM 23 (3) (1976) 398–417.

[21] U. Hustadt, Resolution-Based Decision Procedures for Subclasses of First-Order Logic, Ph.D. thesis, Universität des Saarlandes, Germany (1999).

[22] Y. Kazakov, H. de Nivelle, A Resolution Decision Procedure for the Guarded Fragment with Transitive Guards, in: D. Basin, M. Rusinowitch (Eds.), Proc. of 2nd Int. Joint Conf. on Automated Reasoning (IJCAR 2004), Vol. 3097 of LNAI, Springer, Cork, Ireland, 2004, pp. 122–136.

[23] R. A. Schmidt, U. Hustadt, A Principle for Incorporating Axioms into the First-Order Translation of Modal Formulae, in: F. Baader (Ed.), Proc. of the 19th Int. Conf. on Automated Deduction (CADE-19), Vol. 2741 of LNAI, Springer, Miami Beach, FL, USA, 2003, pp. 412–426.

[24] N. J. Fisher, R. E. Ladner, Propositional Dynamic Logic of Regular Programs, Journal of Computer and System Sciences 18 (1979) 194–211.

[25] B. Motik, Reasoning in Description Logics using Resolution and Deductive Databases, Ph.D. thesis, Univesität Karlsruhe, Germany (2006).

[26] D. A. Plaisted, S. Greenbaum, A Structure-Preserving Clause Form Translation, Journal of Symbolic Logic and Computation 2 (3) (1986) 293–304.

[27] A. Nonnengart, C. Weidenbach, Computing Small Clause Normal Forms, in: A. Robinson, A. Voronkov (Eds.), Handbook of Automated Reasoning, Vol. I, Elsevier Science, 2001, Ch. 6, pp. 335–367.

[28] M. Baaz, U. Egly, A. Leitsch, Normal Form Transformations, in: A. Robinson, A. Voronkov (Eds.), Handbook of Automated Reasoning, Vol. I, Elsevier Science, 2001, Ch. 5, pp. 273–333.

[29] H. D. Nivelle, R. A. Schmidt, U. Hustadt, Resolution-Based Methods for Modal Logics, Logic Journal of the IGPL 8 (3) (2000) 265–292.

[30] G. Gottlob, A. Leitsch, On the Efficiency of Subsumption Algorithms, Journal of the ACM 32 (2) (1985) 280–295.

[31] V. Haarslev, R. Möller, Optimization Strategies for Instance Retrieval, in: I. Horrocks, S. Tessaris, J. Z. Pan (Eds.), Proc. of the 2002 Int. Workshop on Description Logics (DL 2002), Vol. 53 of CEUR Workshop Proceedings, Toulouse, France, 2002.

[32] T. Tammet, Resolution Methods for Decision Problems and Finite-Model Building, Ph.D. thesis, Göteborg University, Sweden (1992).

[33] A. Leitsch, Deciding clause classes by semantic clash resolution, Fundamenta Informaticae 18 (1993) 163–182.

[34] N. Peltier, On the decidability of the PVD class with equality, Logic Journal of the IGPL 9 (4) (2001) 601–624.

[35] C. Fermüller, T. Tammet, N. Zamov, A. Leitsch, Resolution Methods for the Decision Problem, Vol. 679 of LNAI, Springer, 1993.

[36] U. Hustadt, R. A. Schmidt, Issues of Decidability for Description Logics in the Framework of Resolution, in: R. Caferra, G. Salzer (Eds.), Selected Papers from Automated Deduction in Classical and Non-Classical Logics, Vol. 1761 of LNAI, Springer, 1999, pp. 191–205.

[37] H. Ganzinger, U. Hustadt, C. Meyer, R. A. Schmidt, A Resolution-Based Decision Procedure for Extensions of K4, in: M. Zakharyaschev, K. Segerberg, M. de Rijke, H. Wansing (Eds.), Proc. of the 2nd Int. Workshop on Advances in Modal Logic (AiML '98), CSLI Publications, Uppsala, Sweden, 2000, pp. 243–263.

[38] L. Bachmair, H. Ganzinger, Ordered Chaining Calculi for First-Order Theories of Transitive Relations, Journal of the ACM 45 (6) (1998) 1007–1049.

[39] H. Andréka, J. van Benthem, I. Németi, Modal Languages and Bounded Fragments of Predicate Logic, Journal of Philosophical Logic 27 (3) (1998) 217–274.

[40] H. de Nivelle, A Resolution Decision Procedure for the Guarded Fragment, in: C. Kirchner, H. Kirchner (Eds.), Proc. of the 15th Int. Conf. on Automated Deduction (CADE-15), Vol. 1421 of LNAI, Springer, Lindau, Germany, 1998, pp. 191–204.

[41] H. Ganzinger, H. de Nivelle, A Superposition Decision Procedure for the Guarded Fragment with Equality, in: Proc. of the 14th IEEE Symposium on Logic in Computer Science (LICS '99), IEEE Computer Society, Trento, Italy, 1999, pp. 295–305.

[42] L. Bachmair, H. Ganzinger, Rewrite-based Equational Theorem Proving with Selection and Simplification, Journal of Logic and Computation 4 (3) (1994) 217–247.

[43] K. Schild, A Correspondence Theory for Terminological Logics: Preliminary Report, in: J. Mylopoulos, R. Reiter (Eds.), Proc. of 12th Int. Joint Conf. on Artificial Intelligence (IJCAI '91), Morgan Kaufmann Publishers, Sydney, Australia, 1991, pp. 466–471.

[44] I. Hodkinson, Loosely Guarded Fragment of First-Order Logic has the Finite Model Property, Studia Logica 70 (2) (2002) 205–240.

[45] E. Grädel, M. Otto, E. Rosen, Two-Variable Logic with Counting is Decidable, in: Proc. of the 12th IEEE Symposium on Logic in Computer Science (LICS '97), IEEE Computer Society, Warsaw, Poland, 1997, pp. 306–317.

[46] L. Pacholski, W. Szwast, L. Tendera, Complexity Results for First-Order Two-Variable Logic with Counting, SIAM Journal on Computing 29 (4) (2000) 1083–1117.

[47] I. Pratt-Hartmann, Complexity of the Two-Variable Fragment with Counting Quantifiers, Journal of Logic, Language and Information 14 (3) (2005) 369–395.

[48] H. de Nivelle, Splitting Through New Proposition Symbols, in: R. Nieuwenhuis, A. Voronkov (Eds.), Proc. of the 8th Int. Conf. on Logic for Programming, Artificial Intelligence, and Reasoning (LPAR 2001), Vol. 2250 of LNAI, Springer, Havana, Cuba, 2001, pp. 172–185.

[49] A. Riazanov, A. Voronkov, Splitting Without Backtracking, in: B. Nebel (Ed.), Proc. of the 7th Int. Joint Conf. on Artificial Intelligence (IJCAI 2001), Morgan Kaufmann Publishers, Seattle, WA, USA, 2001, pp. 611–617.