Issues of Decidability for Description Logics in the Framework of Resolution

Ullrich Hustadt and Renate A. Schmidt
Department of Computing and Mathematics,
Manchester Metropolitan University*

Since the work of Kallick [10] resolution-based decision procedures for subclasses of first-order logic have drawn continuous attention [3, 5, 9]. There are two research areas where decidability issues likewise played a prominent role: extended modal logics and description logics [4, 7, 11]. Although is is not difficult to see that most of the logics under consideration can be translated to first-order logic, the exact relation to decidable subclasses of first-order logic and in particular to subclasses decidable by resolution is still under investigation. A recent important result describes a resolution decision procedure for the guarded fragment using a non-liftable ordering refinement [3]. The restrictions on the polarity of guards in guarded formulae are too strong to capture description logics with role negation (correspondingly, extended modal logics with relational negation).

Description logics with role negation can be embedded into the class One-Free, for which a resolution decision procedure using a non-liftable ordering refinement exists [5, 15]. However, this method cannot be extended easily to description logics with transitive roles. The method of this paper is based on the resolution framework of Bachmair and Ganzinger [2] which is also suitable for overcoming the problems associated with transitivity axioms, in particular non-termination of resolution on the relational translation of certain transitive modal logics [1, 6].

The most prominent description logic is $\mathcal{ALC}$ [14]. It can be embedded in a subclass of the Bernays-Schönfinkel class. The subclass, called basic path logic, can be decided by resolution and condensing using any compatible ordering or selection strategy [13]. Recent experimental work provides evidence that resolution theorem provers can serve as reasonable and efficient inference tools for description logics [8, 12].

In this paper we consider an expressive description logic, which we believe has not been considered in the literature on description logics or modal logics. We call the logic $\mathcal{ALB}$ which is short for 'attribute language with boolean algebras on concepts and roles'. $\mathcal{ALB}$ extends $\mathcal{ALC}$ with the top role, full role negation, role intersection, role disjunction, role converse, domain restriction, and range restriction.

We describe two methods on the basis of which efficient resolution decision procedures can be developed for a range of description logics. The first method uses an ordering restriction and applies to $\mathcal{ALB}$ and its reducts. The second method is based solely on a selection restriction and applies to reducts of $\mathcal{ALB}$ without the top role and role negation. On $\mathcal{ALC}$, the latter method can be viewed as a polynomial simulation of tableaux-based theorem proving. This result is a first contribution towards a better understanding of the relationship of tableaux-based and resolution-based reasoning for description logics, similar to our understanding of the relationship of various calculi for propositional logic [16].

*Chester Street, Manchester M1 5GD, UK, `U.Hustadt@doc.mmu.ac.uk`, `R.A.Schmidt@doc.mmu.ac.uk`

1 Syntax and Semantics of $\mathcal{ALB}$

The *signature* is given by a tuple $\Sigma = (\mathsf{O}, \mathsf{C}, \mathsf{R})$ of three disjoint alphabets, the set C of *concept symbols*, the set R of *role symbols*, and the set O of *object symbols*. *Concept terms* (or just *concepts*) and *role terms* (or just *roles*) are defined as follows. Every concept symbol is a concept and every role symbol is a role. If C and D are concepts, and R and S are roles, then $\top$, $\bot$, $C \sqcap D$, $C \sqcup D$, $\neg C$, $\forall R.C$, and $\exists R.C$ are *concept terms*, and $\bigtriangledown$, $\bigtriangleup$, $R \sqcap S$, $R \sqcup S$, $\neg R$, R^{-1} (*converse*), $R\restriction C$ (*domain restriction*), and $R\downharpoonright C$ (*range restriction*) are roles.

A *knowledge base* has two parts: A *TBox* comprising of terminological sentences and an *ABox* comprising of assertional sentences. *Terminological sentences* are of the form $C \dot{\sqsubseteq} D$, $C \doteq D$, $R \dot{\sqsubseteq} S$, and $R \doteq S$, and *assertional sentences* are of the form $a \in C$ and $(a, b) \in R$, where C and D are concepts, R and S are roles, and a and b are object symbols.

A symbol S_0 *directly uses* a symbol S_1 in a TBox T if and only if T contains a sentence of the form $S_0 \doteq E$ or $S_0 \dot{\sqsubseteq} E$ such that S_1 occurs in E. A symbol S_0 *uses* S_n if and only if there is a chain of symbols $S_0, \ldots, S_n$ such that S_i directly uses S_{i+1}, for every i, $1 \leq i \leq n-1$. A knowledge base Γ is said to contain a *terminological cycle* if and only if some symbol uses itself in the TBox of Γ. The standard definition of knowledge bases imposes the following restriction on the set of admissible terminological sentences: (i) The concepts on the left-hand sides of terminological sentences have to be concept symbols or role symbols, (ii) any concept symbol occurs at most once on the left-hand side of any terminological sentence, and (iii) there are no terminological cycles. Knowledge bases obeying these restrictions are known as *descriptive knowledge bases*. In this context terminological sentences are called *definitions*.

The semantics is specified by the following embedding into first-order logic. For sentences:

$$\Pi(C \dot{\sqsubseteq} D) = \forall x{:}\, \pi(C, x) \rightarrow \pi(D, x) \qquad \Pi(R \dot{\sqsubseteq} S) = \forall x, y{:}\, \pi(R, x, y) \rightarrow \pi(S, x, y)$$
$$\Pi(C \doteq D) = \forall x{:}\, \pi(C, x) \leftrightarrow \pi(D, x) \qquad \Pi(R \doteq S) = \forall x, y{:}\, \pi(R, x, y) \leftrightarrow \pi(S, x, y)$$
$$\Pi(a \in C) = \pi(C, \underline{a}) \qquad \Pi((a, b) \in R) = \pi(R, \underline{a}, \underline{b})$$

where $\underline{a}$ and $\underline{b}$ are constants uniquely associated with a and b. For terms:

$$\begin{aligned}
\pi(A, X) &= p_A(X) & \pi(P, X, Y) &= p_P(X, Y)\\
\pi(\neg C, X) &= \neg\pi(C, X) & \pi(\neg R, X, Y) &= \neg\pi(R, X, Y)\\
\pi(\top, X) &= \top & \pi(\bigtriangledown, X, Y) &= \top\\
\pi(\bot, X) &= \bot & \pi(\bigtriangleup, X, Y) &= \bot\\
\pi(C \sqcap D, X) &= \pi(C, X) \wedge \pi(D, X) & \pi(R \sqcap S, X, Y) &= \pi(R, X, Y) \wedge \pi(S, X, Y)\\
\pi(C \sqcup D, X) &= \pi(C, X) \vee \pi(D, X) & \pi(R \sqcup S, X, Y) &= \pi(R, X, Y) \vee \pi(S, X, Y)\\
\pi(\forall R.C, X) &= \forall y{:}\, \pi(R, X, y) \rightarrow \pi(C, y) & \pi(R\downharpoonright C, X, Y) &= \pi(R, X, Y) \wedge \pi(C, Y)\\
\pi(\exists R.C, X) &= \exists y{:}\, \pi(R, X, y) \wedge \pi(C, y) & \pi(R\restriction C, X, Y) &= \pi(R, X, Y) \wedge \pi(C, X)\\
& & \pi(R^{-1}, X, Y) &= \pi(R, Y, X)
\end{aligned}$$

X and Y are meta-variables for variables and constants. p_A (respectively p_P) denotes a unary (binary) predicate symbol uniquely associated with the concept symbol A (role symbol P).

All common inferential services for knowledge bases, like subsumption tests for concepts, TBox classification, realization, retrieval, can be reduced to tests of the satisfiability of a knowledge base.

2 The Resolution Framework

We adopt the resolution framework of [2].

As usual clauses are assumed to be multisets of literals. The components in the variable partition of a clause are called split components, that is, split components do not share variables. A clause which is identical to its split component is indecomposable. The condensation $\mathrm{Cond}(C)$ of a clause C is a minimal subclause of C which is a factor of C.

The calculus is parameterised by an ordering $\succ$ and a selection function S. A well-founded and total ordering on ground literals is called *admissible* if it is compatible with a certain *complexity measure* c_L on ground literals L. If $c_L \succ c_{L'}$ implies $L \succ L'$ for any two ground literals L and L', then the ordering is said to be *compatible.* A *selection* function assigns to each clause a possibly empty set of occurrences of negative literals. If C is a clause, then the literal occurrences in $S(C)$ are *selected.* No restrictions are imposed on the selection function.

The calculus consists of general *expansion rules* of the form

$$\frac{N}{N_1 \mid \cdots \mid N_n},$$

each representing a finite derivation of the leaves $N_1, \ldots, N_k$ from the root N. The following rules describe how derivations can be expanded at leaves.

Deduce: $$\frac{N}{N \cup \{\mathrm{Cond}(C)\}}$$

if C is either a resolvent or a factor of clauses in N.

Delete: $$\frac{N \cup \{C\}}{N}$$

if C is a tautology or N contains a clause which is a variant of C.

Split: $$\frac{N \cup \{C \cup D\}}{N \cup \{C\} \mid N \cup \{D\}}$$

if C and D are variable-disjoint.

Resolvents and factors are derived by the following rules.

Ordered Resolution: $$\frac{C \vee A \quad D \vee \neg B}{C\sigma \vee D\sigma}$$

where (i) σ is the most general unifier of A and B, (ii) no literal is selected in C and $A\sigma$ is strictly $\succ$-maximal with respect to $C\sigma$, and (iii) $\neg B$ is either selected, or $\neg B\sigma$ is maximal in $D\sigma$ and no literal is selected in D. $C \vee A$ is called the *positive premise* and $D \vee \neg B$ the *negative premise.*[1]

Ordered Factoring: $$\frac{C \vee A \vee B}{C\sigma \vee A\sigma}$$

where (i) σ is the most general unifier of A and B; and (ii) no literal is selected in C and $A\sigma$ is $\succ$-maximal with respect to $C\sigma$.

We will restrict our attention to derivations which are generated by strategies in which "Delete", "Split", and "Deduce" are applied in this order. In addition, no application of the "Deduce" expansion rule with identical premises and identical consequence may occur twice on the same path in the derivation.

3 Decidability by ordered resolution

The conversion to clausal form of first-order formulae resulting from the translation of $\mathcal{ALB}$ knowledge bases, makes use of a particular form of structural transformation. For ease of presentation we assume any first-order formula ϕ is in negation normal form.

Let $\mathrm{Pos}(\phi)$ be the set of positions of a formula ϕ. If λ is a position in ϕ, then $\phi|_\lambda$ denotes the subformula of ϕ at position λ and $\phi[\lambda \leftarrow \psi]$ is the result of replacing ϕ at position λ by ψ. We associate with each element λ of $\Lambda \subseteq \mathrm{Pos}(\phi)$ a new predicate symbol Q_λ and a new literal $Q_\lambda(x_1, \dots, x_n)$, where $x_1, \dots, x_n$ are the free variables of $\phi|_\lambda$. The *definition* of Q_λ is the formula

$$\mathrm{Def}_\lambda(\phi) = \forall x_1, \dots, x_n : (Q_\lambda(x_1, \dots, x_n) \leftrightarrow \phi|_\lambda).^2$$

Define $\mathrm{Def}_\Lambda(\phi)$ inductively by: $\mathrm{Def}_\emptyset(\phi) = \phi$ and

$$\mathrm{Def}_{\Lambda \cup \{\lambda\}}(\phi) = \mathrm{Def}_\Lambda(\mathrm{Def}_\lambda(\phi) \wedge \phi[\lambda \leftarrow Q_\lambda(x_1, \dots, x_n)]),$$

where λ is maximal in $\Lambda \cup \{\lambda\}$ with respect to the prefix ordering on positions. Let $\mathrm{Pos}_r(\phi)$ be the set of positions of non-atomic subformulae of ϕ with at least one free variable. By Ξ we denote the transformation taking $\Pi(\Gamma)$ to the *definitional form* $\mathrm{Def}_{\mathrm{Pos}_r(\Pi(\Gamma))}(\Pi(\Gamma))$ *of* $\Pi(\Gamma)$.

Theorem 1. *Let Γ be any knowledge base. Γ is satisfiable iff $\Xi\Pi(\Gamma)$ is satisfiable.*

Next we characterise a class of clauses which we call *DL-clauses.* Let C be a clause and L a literal in C. Extending the usual notion of covering, we refer to a literal L as *covering* in C if for every L' in C, $\mathcal{V}(L') \cap \mathcal{V}(L) \neq \emptyset^3$ implies $\mathcal{V}(L') \subseteq \mathcal{V}(L)$ (that is, it contains all variables occurring in the split component in which it occurs). A term t in C is called *covering* if for every L' in C, $\mathcal{V}(L') \cap \mathcal{V}(t) \neq \emptyset$ implies $\mathcal{V}(L) \subseteq \mathcal{V}(t)$. A term is called *compound* if it is neither a variable nor a constant. A literal L is *singular* if it contains no compound term and $\mathcal{V}(L)$ is a singleton. A literal is *flat* if it is non-ground and contains no compound term.

In the context of this paper a *regular* literal has either no compound term arguments or if it does then there is a compound term which contains all the variables of the literal. By definition, L is a *DL-literal* if the following is true.

1. L is regular,
2. L is either monadic or dyadic, and contains at most 2 variables,
3. L is ground whenever it contains a constant symbol, and
4. the maximal arity of any function symbol in L is 1.

[1] As usual we implicitly assume that the premises have no common variables.

[2] As any formula ϕ is assumed to be in negation normal form, we could of course also use $\mathrm{Def}^+_\lambda(\phi) = \forall x_1, \dots, x_n : (Q_\lambda(x_1, \dots, x_n) \rightarrow \phi|_\lambda)$ without altering the validity of our results.

[3] $\mathcal{V}(L)$ denotes the set of variables of L.

A clause C is a *DL-clause*, if

1. when C contains a compound term t, then t is covering,
2. C is ground whenever C contains a constant symbol,
3. all literals in C are DL-literals, and
4. the argument multisets of all flat, dyadic literals coincide.

Property (4) excludes clauses like $\{p(x,x), q(x,y)\}$. The problem is that both literals are maximal with respect to any ordering which is stable under substitutions. Nevertheless, in order to avoid possibly unbounded chains of variables across literals we need to restrict resolution inferences to the literal $q(x,y)$. By contrast, clauses like $\{p(x,x), q(x,x)\}$ and $\{p(x,y), q(x,y)\}$ are DL-clauses, and may occur in a derivation from $\Xi\Pi(\Gamma)$.

Lemma 2. *Let Γ be a knowledge base. Every clause in the clausal form of $\Xi\Pi(\Gamma)$ belongs to the class of DL-clauses.*

For every ground literal L, let the complexity measure c_L be the multiset of arguments of L. We compare complexity measures by the multiset extension of the strict subterm ordering $\succ^s_{mul}$. The ordering is lifted from ground to non-ground expressions as follows: $E \succ E'$ if and only if $E\sigma \succ E'\sigma$, for all ground instances $E\sigma$ and $E'\sigma$. We show that ordered resolution and ordered factoring on DL-clauses with respect to any ordering $\succ_{cov}$ which is compatible with this complexity measure will result in DL-clauses.

Lemma 3. *Let $C = \{L_1, L_2\} \cup D$ be an indecomposable, DL-clause with σ a most general unifier of L_1 and L_2 such that $L_1\sigma$ is $\succ_{cov}$-maximal in $C\sigma$. The split components of $(\{L_1\} \cup D)\sigma$ are DL-clauses.*

Lemma 4. *Let $C_1 = \{A\} \cup D_1$ and $C_2 = \{\neg B\} \cup D_2$ be two variable-disjoint, indecomposable, DL-clauses such that A and B are unifiable with most general unifier σ, and $A\sigma$ and $B\sigma$ are $\succ_{cov}$-maximal in $C_1\sigma$ and $C_2\sigma$, respectively. The split components of $(D_1 \cup D_2)\sigma$ are DL-clauses.*

Theorem 5. *Let Γ be a knowledge base of $\mathcal{ALB}$ and let N be the clausal form of $\Xi\Pi(\Gamma)$. Then any derivation from N by ordered resolution and ordered factoring based on $\succ_{cov}$ terminates.*

Proof. By Theorem 1 and Lemmas 2, 3 and 4; because any class of non-variant indecomposable DL-clauses built from finitely many predicate and function symbols is bound; and the fact that any application of "Deduce" will be followed immediately by applications of the "Split" rule, as well as the fact that "Delete" is applied eagerly. $\square$

The techniques used by Tammet and Zamov to decide the classes One-Free [15] and KS [5] can also be utilised to provide a decision procedure for the class of DL-clauses. However, these techniques are based on non-liftable orderings which have limitations regarding the application of some standard simplification rules.

4 Decidability by selection

In this section we focus on descriptive knowledge bases Γ over reducts of $\mathcal{ALB}$ without role negation and the top role. We define a decision procedure based solely on the use of a particular selection function. Our intention is to restrict resolution inferences with clauses stemming from the translation of terminological sentences of the form $S \doteq E$ and $S \mathrel{\dot{\sqsubseteq}} E$ to the literals associated with S. As only negative literals can be selected, it is necessary to transform the given knowledge base. Essentially, for concepts, occurrences of $\neg A$ will be replaced by a new symbol $\overline{A}$, and for roles, positive occurrences of P will be replaced by a new symbol P^d while negative occurrences will be replaced by P^u.

Without loss of generality, all expressions occurring in Γ are assumed to be in negation normal form. Formally, let $\mathrm{D}_{\doteq}(\Gamma)$ denote the set of symbols $S_0 \in \mathsf{C} \cup \mathsf{R}$ such that Γ contains a terminological sentence $S_0 \doteq E$. We obtain the transformed knowledge base $\overline{\Gamma}$, defined over $(\mathsf{O}, \overline{\mathsf{C}}, \overline{\mathsf{R}})$, in the following way. Extend C to $\overline{\mathsf{C}}$, by adding a concept symbol $\overline{A}$ for every concept symbol A in $\mathrm{D}_{\doteq}(\Gamma)$. Replace R by $\overline{\mathsf{R}}$, which is obtained by replacing every role symbol $P \in \mathrm{D}_{\doteq}(\Gamma)$ by new symbols P^u and P^d. The following steps transform Γ to $\overline{\Gamma}$.

1. Replace concept definitions $A \doteq C$, by $A \mathrel{\dot{\sqsubseteq}} C$ and $\neg A \mathrel{\dot{\sqsubseteq}} \mathrm{nnf}(\neg C)$, and replace role definitions $P \doteq R$, by $P^d \mathrel{\dot{\sqsubseteq}} R$ and $R \mathrel{\dot{\sqsubseteq}} P^u$, where $\mathrm{nnf}(\neg C)$ is the negation normal form of $\neg C$.
2. Replace every occurrence of a concept $\neg A$, for A in $\mathrm{D}_{\doteq}(\Gamma)$, by $\overline{A}$.
3. Replace every positive occurrence[4] of a role symbol $P \in \mathrm{D}_{\doteq}(\Gamma)$ by P^d, and every negative occurrence of P by P^u.
4. For every concept symbol A in $\mathrm{D}_{\doteq}(\Gamma)$, add the terminological sentence $A \mathrel{\dot{\sqsubseteq}} \neg\overline{A}$ and add for every role symbol P in $\mathrm{D}_{\doteq}(\Gamma)$, the terminological sentence $P^d \mathrel{\dot{\sqsubseteq}} P^u$.

For example, if Γ contains the terminological axioms $A \doteq B \sqcap C$ and $P \doteq R \sqcap S$ where A, B, and C are concept symbols and P, R, and S are role symbols, then the transformed knowledge base contains

$$\begin{array}{lr} A \mathrel{\dot{\sqsubseteq}} B \sqcap C & P^d \mathrel{\dot{\sqsubseteq}} R \sqcap S \\ \overline{A} \mathrel{\dot{\sqsubseteq}} \neg B \sqcup \neg C & R \sqcap S \mathrel{\dot{\sqsubseteq}} P^u \\ A \mathrel{\dot{\sqsubseteq}} \neg\overline{A} & P^d \mathrel{\dot{\sqsubseteq}} P^u \end{array}$$

In this section, the definitional form is produced by a variant $\overline{\Xi}$ of the transformation Ξ described in the previous section. First, $\overline{\Xi}$ uses *definitions*, $\mathrm{Def}_\lambda(\phi)$, of the form

$$\forall x_1, \ldots, x_n \colon (Q_\lambda(x_1, \ldots, x_n) \rightarrow \phi|_\lambda) \quad \text{or} \quad \forall x_1, \ldots, x_n \colon (\phi|_\lambda \rightarrow Q_\lambda(x_1, \ldots, x_n)),$$

depending on whether $\phi|_\lambda$ occurs positively or negatively in ϕ. Second, for subformulae χ of the form $\forall y \colon (\neg\phi(x,y) \vee \psi(y))$ or $\exists y \colon (\phi(x,y) \wedge \psi(y))$, $\overline{\Xi}$ will only introduce definitions for χ itself and $\phi(x,y)$ and $\psi(y)$ if necessary, but not for $\neg\phi(x,y) \vee \psi(y)$ and $\phi(x,y) \wedge \psi(y)$. For the sample knowledge base above we obtain the following set of clauses.

$$\begin{array}{ll} \{\neg p_A(x), p_B(x)\} & \{\neg p^d_P(x,y), p_R(x,y)\} \\ \{\neg p_A(x), p_C(x)\} & \{\neg p^d_P(x,y), p_S(x,y)\} \\ \{\neg p_{\overline{A}}(x), \neg p_B(x), \neg p_C(x)\} & \{\neg p_R(x,y), \neg p_S(x,y), p^u_P(x,y)\} \\ \{\neg p_A(x), \neg p_{\overline{A}}(x)\} & \{\neg p^d_P(x,y), p^u_P(x,y)\} \end{array}$$

[4]An occurrence of a subformula (subexpression) is a *positive occurrence* if it is one inside the scope of an even number of (explicit or implicit) negations, and an occurrence is a *negative occurrence* if it is one inside the scope of an odd number of negations.

Define a dependency relation $\succ_c^1$ on the predicate symbols by: $p_A \succ_c^1 p_B$, if there is a definition $\phi \rightarrow \psi$ in $\overline{\Xi}\Pi(\overline{\Gamma})$ such that p_A occurs in ϕ and p_B occurs in ψ. Let $\succ_S$ be an ordering on the predicate symbols in $\overline{\Xi}\Pi(\overline{\Gamma})$ which is compatible with the transitive closure $\succ_c^*$ of $\succ_c^1$. Due to the acyclicity of the terminology and due to fact that we split role definitions, it is possible to find such an ordering.

While an ordering is optional, our selection function $S_{\mathcal{TAB}}$ selects the literal $\neg p_{\overline{A}}(x)$ in C if C is the clause $\{\neg p_{\overline{A}}(x), \neg p_A(x)\}$ originating from $A \dot{\sqsubseteq} \neg\overline{A}$. For all other clauses, let $\neg L$ be an occurrence of a negative literal in C with predicate symbol p_A. Then $\neg L$ is selected in C if and only if either p_A is the $\succ_S$-maximal predicate symbol in C, or $\neg L$ is a literal of the form $\neg p_A(s, y)$, where s is a ground term and y is a variable. In our sample clause set above all negative literal occurrences are selected except for the last two clauses in the left column, where in each case only $\neg p_{\overline{A}}(x)$ is selected. All clauses originating from the translation of assertional sentences are ground unit clauses. All clauses stemming from a terminological sentence or from a definition introduced by $\overline{\Xi}$ contain negative literals, one of which is selected. Consequently, no factoring steps are possible and the clauses may only be used as negative premises of resolution steps.

In all clauses except those of the form

$$\{\neg p_0(x)^+, \neg p_1(x,y), p_2(y)\} \tag{1}$$

the selected literal (marked by $^+$) contains all variables of the clause, and with the exception of

$$\{\neg p_0(x)^+, p_1(x, f(x))\} \quad \text{and} \quad \{\neg p_0(x)^+, p_2(f(x))\}. \tag{2}$$

no variables occur as arguments of compound terms.

Inferences with premises like (1) are problematic, since the resolvent may contain more free variables than the positive premise of the inference step. Suppose we have derived a clause of the form $\{p_1(\underline{a}), p_2(\underline{a}), p_3(\underline{a})\}$ (momentarily ignoring the "Split" rule) and $\overline{\Xi}\Pi(\overline{\Gamma})$ contains the clauses $\{\neg p_i(x)^+, \neg r_i(x,y), q_i(y)\}$, for $1 \leq i \leq 3$. Without taking further restrictions into account, we can derive the clause

$$\{\neg r_1(\underline{a}, x), q_1(x), \neg r_2(\underline{a}, y), q_2(y), \neg r_3(\underline{a}, z), q_3(z)\}.$$

It contains more variables than any clause in $\overline{\Xi}\Pi(\overline{\Gamma})$.

In general, the positive premise of a resolution inference step with a clause like (1) is a ground clause $\{p_0(s)\} \cup D_1$ such that no literals in D_1 are selected. The conclusion of the inference step is a clause $C_1 = \{\neg p_1(s,y)^+, p_2(y)\} \cup D_1$, with one free variable. However, the literal $\neg p_1(s,y)$ is selected by $S_{\mathcal{TAB}}$ and no inference steps are possible on D_1 (which contains no selected literals). The only clauses we can derive containing a positive literal with predicate symbol p_1 will be ground clauses, that is, clauses of the form $C_2 = \{p_1(s,t)\} \cup D_2$. The conclusion of an inference step between C_1 and C_2 is the ground clause $\{p_2(t)\} \cup D_1 \cup D_2$. Consequently, all clauses occurring in a derivation from the clausal form of $\overline{\Xi}\Pi(\overline{\Gamma})$ contain at most two variables. Note that C_1 is not a DL-clause.

Also note for the argument it is important that a negative binary literal occurs in (1). This is the reason for excluding role negation as well as the top role from the language.

The problem with inferences involving negative premises of the forms (2) is that resolvents may contain terms of greater depth than the positive premise of the inference. Nevertheless, we can still show that there is an upper bound on the height of terms. With every clause C we associate a complexity measure $c_C = \{c_L \mid L \in C\}$. Complexity measures on ground literals are compared by the ordering $\succ_c^{\mathrm{lit}}$ given by the lexicographic combination of the ordering $\succ_{\mathrm{S}}$ and the multiset extension of the strict subterm ordering $\succ_{mul}^s$. The ordering is lifted from ground to non-ground expressions in the usual way. The ordering on clauses is the multiset extension of $\succ_c^{\mathrm{lit}}$. It is straightforward to check that any inference step from a positive premise C by (ordered) resolution or (ordered) factoring will result in a clause D such that c_C is greater than c_D with respect to $\succ_c^{\mathrm{mul}}$.

Theorem 6. *Let Γ be a descriptive knowledge base and let N be the clausal form of $\Xi\Pi(\overline{\Gamma})$. Then any derivation from N by (ordered) resolution with selection as determined by $S_{\mathcal{TAB}}$ and (ordered) factoring terminates.*

The correspondence between the tableaux-based decision procedure and the selection-based decision procedure is not difficult to see.

Recall, for example, from [14], the satisfiability test of an ABox in a tableaux-based system for $\mathcal{ALC}$ is done by applying the following completion rules:

1. $\Delta \Rightarrow_{\sqcap} \Delta \cup \{a \in C, a \in D\}$, if $a \in (C \sqcap D)$ is in Δ, $a \in C$ and $a \in D$ are not both in Δ.
2. $\Delta \Rightarrow_{\sqcup} \Delta \cup \{a \in E\}$, if $a \in (C \sqcup D)$ is in Δ, neither $a \in C$ nor $a \in D$ is in Δ, and $E = C$ or $E = D$.
3. $\Delta \Rightarrow_{\exists} \Delta \cup \{(a,b) \in R, b \in C\}$, if $a \in \exists\, R.C$ is in Δ, there is no b such that both $(a,b) \in R$ and $b \in C$ are in Δ, and b is a new object symbol with respect to Δ.
4. $\Delta \Rightarrow_{\forall} \Delta \cup \{b \in C\}$, if $a \in \forall\, R.C$ and $(a,b) \in R$ are in Δ, and $b \in C$ is not in Δ.

Let $\Rightarrow_{\mathcal{TAB}}$ be the transitive closure of the union of the transformation rules given above. An ABox Δ contains a *clash* if both $a \in C$ and $a \in \neg C$ are in Δ, or $a \in \bot$ is in Δ. An ABox Δ is satisfiable if there exists an ABox Δ' such that (i) $\Delta \Rightarrow_{\mathcal{TAB}} \Delta'$, (ii) no further applications of $\Rightarrow_{\mathcal{TAB}}$ to Δ' are possible, and (iii) Δ' is clash-free.

Note that for every concept C and every role R, which may possibly occur in an ABox during a satisfiability test, there exist corresponding predicate symbols p_C and p_R in the clausal form of $\Xi\Pi(\overline{\Gamma})$. Likewise for every object symbol a we will have a corresponding term t_a.

1. An application of the $\Rightarrow_{\sqcap}$ rule corresponds to a resolution inference step between a ground clause $\{p_{C\sqcap D}(t_a)\}$ and clauses $\{\neg p_{C\sqcap D}(x), p_C(x)\}$ and $\{\neg p_{C\sqcap D}(x), p_D(x)\}$, generating the resolvents $\{p_C(t_a)\}$ and $\{p_D(t_a)\}$.

2. An application of the $\Rightarrow_{\sqcup}$ rule corresponds to a resolution inference step between a ground unit clause $\{p_{C\sqcup D}(t_a)\}$ and the clause $\{\neg p_{C\sqcup D}(x), p_C(x), p_D(x)\}$. We then apply the splitting rule to the conclusion $\{p_C(t_a), p_D(t_a)\}$ which will generate two branches, one on which our set of clauses contains $\{p_C(t_a)\}$ and one on which it contains $\{p_D(t_a)\}$.

3. An application of the $\Rightarrow_{\exists}$ rule corresponds to two resolution inference steps between $\{p_{\exists R.C}(t_a)\}$ and clauses $\{\neg p_{\exists R.C}(x), p_R(x, f(x))\}$ and $\{\neg p_{\exists R.C}(x), p_C(f(x))\}$. This will add $\{p_R(t_a, f(t_a))\}$ and $\{p_C(f(t_a))\}$ to the clause set. The term $f(t_a)$ corresponds to the new object symbol b introduced by the $\Rightarrow_{\exists}$ rule, that is, $t_b = f(t_a)$.

4. An application of the $\Rightarrow_{\forall}$ rule corresponds to two consecutive inference steps. Here, the set of clauses contains $\{p_{\forall R.C}(t_a)\}$ and $\{p_R^u(t_a, t_b)\}$ (to obtain $\{p_R^u(t_a, t_b)\}$ an

inference step with a clause $\{\neg p^d_R(x,y), p^u_R(x,y)\}$ may be necessary). First, the clause $\{p_{\forall R.C}(t_a)\}$ is resolved with $\{\neg p_{\forall R.C}(x), \neg p^u_R(x,y), p_C(y)\}$ to obtain the clause $\{\neg p^u_R(t_a,y), p_C(y)\}$. Then the conclusion is resolved with $\{p^u_R(t_a,t_b)\}$ to obtain $\{p_C(t_b)\}$.

Note that all these resolution inference steps strictly obey the restrictions enforced by the selection function $S_{\mathcal{TAB}}$.

When the TBox of a decriptive knowledge base is non-empty, then concept and role symbols in the ABox are *unfolded*, that is, replaced by the right-hand side of their definitions, before the tableaux-based transformation system is applied to the ABox. The unfolding steps can be simulated by resolution inference steps with the clauses we obtain from the translation of the TBox. Again, these inference steps obey the restrictions enforced by the selection function $S_{\mathcal{TAB}}$.

Theorem 7. *The selection-based resolution decision procedure with selection function $S_{\mathcal{TAB}}$ p-simulates tableaux-based decision procedures (for $\mathcal{ALC}$).*

Moreover, the described procedure provides a basis for defining tableaux-based decision procedures for extensions of $\mathcal{ALC}$ with role conjunction and/or role disjunction.

5 Conclusion

The class of DL-clauses is not comparable with the guarded fragment or the loosely guarded fragment. In the guarded fragments the conditional quantifiers may not include negations or disjunctions. On the other hand, the guarded fragments allow predicates of arbitrary arity. Recently it has been shown that the extension of the guarded fragment with two interacting transitive relations and equality is undecidable. However, basic modal logic plus transitivity is known to be decidable. Therefore, looking at more restricted classes than the guarded fragment may lead to better characterisations of the connection between modal logics and decidable subclasses of first-order logic [6].

The class of DL-clauses is more restrictive than the class One-Free, which stipulates that quantified subformulae have at most one free variable. But it is possible to extend $\mathcal{ALB}$ by certain restricted forms of role composition (e.g., positive occurrences), for which the procedure described in Section 3 remains a decision procedure. The corresponding clausal class is distinct from the One-Free class. It is known from the literature on algebraic logic that arbitrary occurrences of composition in the presence of role negation leads to undecidability.

The resolution decision procedures of [5, 15] have the disadvantage that they are based on a non-liftable ordering refinement. As a consequence certain standard simplification rules, e.g. tautology deletion, have to be restricted for completeness. Real world knowledge bases typically contain hundreds of concept definitions. The corresponding clauses can be used to derive an extensive number of tautologies. Our approach does not have this drawback. In addition to using liftable orderings, the resolution framework here is equipped with a general notion of redundancy which accomodates most standard simplification rules including tautology deletion, condensing, subsumption deletion, as well as non-standard theory specific simplification rules. For a discussion of redundancy and fairness see [2].

References

[1] L. Bachmair and H. Ganzinger. Ordered chaining calculi for first-order theories of binary relations. Research report MPI-I-95-2-009, Max-Planck-Institut für Informatik, Saarbrücken, Germany, 1995. To appear in the *Journal of the ACM*.

[2] L. Bachmair and H. Ganzinger. A theory of resolution. Research report MPI-I-97-2-005, Max-Planck-Institut für Informatik, Saarbrücken, Germany, 1997. Preliminary version of a chapter in the *Handbook of Automated Reasoning*, edited by J. A. Robinson and A. Voronkov.

[3] H. de Nivelle. A resolution decision procedure for the guarded fragment. In *Proceedings of the 15th International Conference on Automated Deduction (CADE-15)*, volume 1421 of *LNAI*, pages 191–204. Springer, 1998.

[4] F. M. Donini, M. Lenzerini, D. Nardi, and A. Schaerf. Deduction in concept languages: From subsumption to instance checking. *Journal of Logic and Computation*, 4(4):423–452, 1994.

[5] C. Fermüller, A. Leitsch, T. Tammet, and N. Zamov. *Resolution Method for the Decicion Problem*, volume 679 of *LNCS*. Springer, 1993.

[6] H. Ganzinger, U. Hustadt, C. Meyer, and R. A. Schmidt. A resolution-based decision procedure for extensions of K4. Presented at AiML'98, Uppsala, Sweden, October 1998.

[7] E. Hemaspaandra. The price of universality. *Notre Dame Journal of Formal Logic*, 37(2):174–203, 1996.

[8] U. Hustadt and R. A. Schmidt. On evaluating decision procedures for modal logic. In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence (IJCAI'97)*, pages 202–207. Morgan Kaufmann, 1997.

[9] W. H. Joyner Jr. Resolution strategies as decision procedures. *Journal of the Association for Computing Machinery*, 23(3):398–417, 1976.

[10] B. Kallick. A decision procedure based on the resolution method. In *Information Processing 68: Proceedings of IFIP Congress 1968. Volume 1*, pages 269–275. North-Holland, 1968.

[11] M. Marx. Mosaics and cylindric modal logic of dimension 2. In *Advances in Modal Logic, Volume 1*, volume 87 of *Lecture Notes*, pages 141–156. CSLI Publications, 1996.

[12] M. Paramasivam and D. A. Plaisted. Automated deduction techniques for classification in description logic systems. *Journal of Automated Reasoning*, 20(3):337–364, 1998.

[13] R. A. Schmidt. Decidability by resolution for propositional modal logics. To appear in *Journal of Automated Reasoning*, 1998.

[14] M. Schmidt-Schauß and G. Smolka. Attributive concept description with complements. *Artifical Intelligence*, 48:1–26, 1991.

[15] T. Tammet. Using resolution for extending KL-ONE-type languages. In *Proceedings of the Fourth International Conference on Information and Knowledge Management (CIKM'95)*, 1995.

[16] A. Urquhart. The complexity of propositional proofs. *The Bulletin of Symbolic Logic*, 1(4):425–467, 1995.