On the Relation of Resolution and Tableaux Proof Systems for Description Logics

Ullrich Hustadt and **Renate A. Schmidt**
Department of Computing and Mathematics, Manchester Metropolitan University,
Chester Street, Manchester M1 5GD, UK
U.Hustadt@doc.mmu.ac.uk, R.A.Schmidt@doc.mmu.ac.uk

Abstract

This paper investigates the relationship between resolution and tableaux proof system for the satisfiability of general knowledge bases in the description logic $\mathcal{ALC}$. We show that resolution proof systems can polynomially simulate their tableaux counterpart. Our resolution proof system is based on a selection refinement and utilises standard redundancy elimination criteria to ensure termination.

1 Introduction

Recently a number of results concerning resolution decision procedures for subclasses of first-order logic have been obtained. The considered subclasses are expressive enough to encompass a variety of non-classical logics, in particular, description logics and extended modal logics. De Nivelle [1998] describes a resolution decision procedure for the guarded fragment using a non-liftable ordering refinement. The guarded fragment is a generalisation of the restricted quantifier fragment corresponding to basic modal logic and allows for the embedding of a variety of extended modal logics and description logics [Grädel, 1998]. Expressions and knowledge bases of the description logic $\mathcal{ALC}$ can also be embedded into Maslov's class K and its subclasses One-Free [Fermüller *et al.*, 1993] and the class of DL-clauses [Hustadt and Schmidt, 1999]. Again, ordering refinements of resolution provide decision procedures for these classes. A non-standard translation into the Bernays-Schönfinkel class combined with resolution and arbitrary refinements provide decision procedures for the satisfiability of $\mathcal{ALC}$ expressions [Schmidt, 1999]. This approach was adopted in the experiments of Hustadt, Schmidt, and Weidenbach [1997; 1998]. Experiments using the standard translation and a combination of a first-order theorem prover augmented with a finite-model finder are described in [Paramasivam and Plaisted, 1998].

The problem of empirical investigations based on competitive testing is the difficulty in identifying the major factors having a positive or negative influence on the performance of a theorem prover. As long as the theorem provers which are being compared follow different proof strategies this difference is likely to have a dominating effect on the overall performance. This has two consequences. One, we can say little about the other factors influencing the performance, for example, fundamental differences between the underlying proof systems or sophisticated redundancy elimination techniques used by the theorem prover. Two, while it is easy to find benchmark problems illustrating the superiority of one theorem prover it is just as easy to find benchmark problems showing the opposite. Therefore, it is always advisable to complement empirical investigations with a theoretical analysis of the relative proof and search complexity of the underlying proof systems. In the first case, the task is to determine whether a proof system $\mathcal{A}$ is able to polynomially simulate a proof system $\mathcal{B}$. This is to say, for any given theorem ϕ there is a function g, computable in polynomial time, mapping proofs of ϕ in $\mathcal{B}$ to proofs of ϕ in $\mathcal{A}$. In the second case, the task is to determine the relative size of the search space, that is the potential number of inference steps performed until a proof is found [Plaisted and Zhu, 1997].

In this paper we focus on the aspect of relative proof complexity of tableaux proof systems and resolution proof systems for the description logic $\mathcal{ALC}$ with general terminological sentences and ABox elements. This logic is of particular interest, since all tableaux proof systems presented in the literature require some form of blocking or loop-checking to force termination [Buchheit *et al.*, 1993; Donini *et al.*, 1996; Horrocks, 1997]. We describe a resolution proof system based on a selection refinement of resolution, instead of an ordering refinement, which provides a new resolution decision procedure for this logic. We show that this proof system is able to polynomially simulate tableaux proof systems for this logic. The technique for simulating blocking described in this paper can also be applied for obtaining other simulation results, for example, analytic modal KE tableaux proof systems or sequent proof systems for modal logics.

The structure of the paper is as follows. Section 2 defines the syntax and semantics of $\mathcal{ALC}$ and describes a standard tableaux proof system. We adopt the resolution framework of Bachmair and Ganzinger [1998] which is described briefly in Section 3. Section 4 presents the

simulation result for the tableaux proof system and Section 5 shows how termination of the resolution proof system can be enforced in analogy to blocking in tableaux systems. In Section 6 we discuss some optimisations which are naturally available in the resolution framework and can be transferred to the corresponding tableaux proof systems.

2 Inference for $\mathcal{ALC}$

We work with a *signature* 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 *objects*. *Concept terms* (or just *concepts*) are defined as follows. Every concept symbol is a concept. If C and D are concepts, and R is a role symbol, then $\top$, $\bot$, $C \sqcap D$, $C \sqcup D$, $\neg C$, $\forall R.C$, and $\exists R.C$ are concept terms. A concept symbol is also called a *primitive concept*.

A *knowledge base* has two parts: A *TBox* comprising of terminological sentences of the form $C \doteq D$ and an *ABox* comprising of assertional sentences of the form $a \in C$ and $(a, b) \in R$, where C and D are concepts, R is a role, and a and b are objects.

Although the language does contain any role forming operators, it is still possible to express properties of the domain and range of a role R [Buchheit *et al.*, 1993].

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

$$\begin{aligned}\Pi(C \doteq D) &= \forall x\colon \pi(C, x) \rightarrow \pi(D, x)\\ \Pi(a \in C) &= \pi(C, \underline{a})\\ \Pi((a, b) \in R) &= \pi(R, \underline{a}, \underline{b})\end{aligned}$$

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

$$\begin{aligned}\pi(A, X) &= p_A(X)\\ \pi(R, X, Y) &= p_R(X, Y)\\ \pi(\neg C, X) &= \neg\pi(C, X)\\ \pi(\top, X) &= \top\\ \pi(\bot, X) &= \bot\\ \pi(C \sqcap D, X) &= \pi(C, X) \wedge \pi(D, X)\\ \pi(C \sqcup D, X) &= \pi(C, X) \vee \pi(D, X)\\ \pi(\forall R.C, X) &= \forall y\colon \pi(R, X, y) \rightarrow \pi(C, y)\\ \pi(\exists R.C, X) &= \exists y\colon \pi(R, X, y) \wedge \pi(C, y)\end{aligned}$$

where X and Y are meta-variables for variables and constants, and p_A (respectively p_R) denotes a unary (binary) predicate symbol uniquely associated with the concept symbol A (role symbol R). The variable y is distinct from X.

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. Our definition of a tableaux proof system, also called a constraint system, largely follows Buchheit *et al.* [1993]. All terminological sentences $C \sqsubseteq D$ are assumed to have been replaced by $\top \sqsubseteq \neg C \sqcup D$ and all concepts in the resulting knowledge based are assumed to have been transformed into negation normal form.

Let I be a subset of O such that no element of I occurs in Γ. Let $\prec_{\mathsf{I}}$ be a well-founded total ordering on I. The elements of I are called *introduced objects*. We assume that the elements of I are introduced during inference according to $\prec_{\mathsf{I}}$, that is, if a is introduced into Γ, then for all $b \in \mathsf{I}$ with $b \prec_{\mathsf{I}} a$, b already occurs in Γ.

Following Buchheit *et al.* [1993] we define the following set of transformation rules for the purpose of testing the satisfiability of a knowledge base:

1. $\Gamma \Rightarrow_{\sqcap} \Gamma \cup \{a \in C, a \in D\}$, if $a \in (C \sqcap D)$ is in Γ, and $a \in C$ and $a \in D$ are not both in Γ.
2. $\Gamma \Rightarrow_{\sqcup} \Gamma \cup \{a \in E\}$, if $a \in (C \sqcup D)$ is in Γ, neither $a \in C$ nor $a \in D$ is in Γ, and $E = C$ or $E = D$.
3. $\Gamma \Rightarrow_{\exists} \Gamma \cup \{(a, b) \in R, b \in C\}$, if $a \in \exists R.C$ is in Γ, there is no d such that both $(a, d) \in R$ and $d \in C$ are in Γ, $b \in \mathsf{I}$ is a newly introduced object.
4. $\Gamma \Rightarrow_{\forall} \Gamma \cup \{b \in C\}$, if $a \in \forall R.C$ and $(a, b) \in R$ are in Γ, and $b \in C$ is not in Γ.
5. $\Gamma \Rightarrow_{\bot} \Gamma \cup \{a \in \bot\}$, if $a \in A$ and $a \in \neg A$ are in Γ, where A is a concept symbol.
6. $\Gamma \Rightarrow_{\sqsubseteq} \Gamma \cup \{a \in C\}$, if a occurs in Γ and a sentence $\top \sqsubseteq C$ is in Γ, and $a \in C$ is not in Γ.

Let $\Rightarrow_{\mathcal{TAB}}$ be the transitive closure of the union of the transformation rules given above. A knowledge base Γ contains a *clash* if $a \in \bot$ is in Γ. A knowledge base Γ is satisfiable if there exists a knowledge base Γ' such that (i) $\Gamma \Rightarrow_{\mathcal{TAB}} \Gamma'$, (ii) no further applications of $\Rightarrow_{\mathcal{TAB}}$ to Γ' are possible, and (iii) Γ' is clash-free. Note that the rule $\Rightarrow_{\sqcup}$ is don't know nondeterministic.

3 The Resolution Framework

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. The ordering has to satisfy certain restrictions as detailed in [Bachmair and Ganzinger, 1998], in particular, it is required to be a reduction ordering. 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* (over clause sets)

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

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: $$\frac{C \cup \{A_1\} \quad D \cup \{\neg A_2\}}{(C \cup D)\sigma}$$

where (i) σ is the most general unifier of A_1 and A_2, (ii) no literal is selected in C and $A_1\sigma$ is strictly $\succ$-maximal with respect to $C\sigma$, and (iii) $\neg A_2$ is either selected, or $\neg A_2\sigma$ is maximal in $D\sigma$ and no literal is selected in D. $C \vee A_1$ is called the *positive premise* and $D \vee \neg A_2$ the *negative premise.*[1]

Ordered Factoring: $$\frac{C \cup \{A_1, A_2\}}{(C \cup \{A_1\})\sigma}$$

where (i) σ is the most general unifier of A_1 and A_2; and (ii) no literal is selected in C and $A_1\sigma$ is $\succ$-maximal with respect to $C\sigma$.

Let N be a set of ground clauses. A ground clause C is *redundant* in N if there are clauses $C_1, \ldots, C_n$ in N such that $C_1, \ldots, C_n$ are smaller than C with respect to $\succ$ and logically imply C. The notion of redundancy is lifted to the non-ground case in the expected way. An inference is redundant if one of the parent clauses or its conclusion is redundant.

Theorem 1 (Bachmair and Ganzinger [1998]). *Let N be a set of clauses. Then N is unsatisfiable iff the saturation of N up to redundancy contains the empty clause.*

4 Simulation by Resolution

Our intention is to restrict resolution inferences in such a way that admissible resolution steps correspond to inference steps in tableaux proof systems. Furthermore, the resolution proof system will be a decision procedure whenever the tableaux system terminates without the help of loop-checking or blocking techniques.

It is necessary to modify the translation mapping Π slightly. Without loss of generality, all expressions occurring in Γ are assumed to be in negation normal form. Let $\blacktriangledown$ be a concept symbol not occurring in Γ. Intuitively, $\blacktriangledown$ has the same semantics as the concept symbol $\top$. But while $\top$ is translated to the true formula and will vanish during the conversion to clausal form, the translation treats $\blacktriangledown$ as an ordinary concept symbol. By adding certain formulae to the translation of the knowledge base we provide sufficient information about $\blacktriangledown$ to ensure that the introduction of $\blacktriangledown$ preserves satisfiability equivalence. This allows us to obtain the desired computational behaviour in our resolution proof system.

The modified translation $\overline{\Pi}$ is defined as follows.

$$\overline{\Pi}(C \dot{\sqsubseteq} D) = \forall x\colon \pi(\blacktriangledown, x) \to \pi(\mathrm{nnf}(\neg C) \sqcup D, x)$$
$$\overline{\Pi}(a \in C) = \pi(C, \underline{a}) \wedge \pi(\blacktriangledown, \underline{a})$$
$$\overline{\Pi}((a,b) \in R) = \pi(R, \underline{a}, \underline{b}) \wedge \pi(\blacktriangledown, \underline{a}) \wedge \pi(\blacktriangledown, \underline{b})$$

The occurrence of $\pi(\blacktriangledown, x)$ on the left-hand side of the implication ensures that all clauses in the clausal form contain the negative literal $\neg p_{\blacktriangledown}(x)$. Since

$$\overline{\Pi}(C \sqsubseteq D) = \overline{\Pi}(\top \sqsubseteq \mathrm{nnf}(\neg C) \sqcup D)$$

it is immaterial whether the terminological sentences in Γ take the first or second form.

The conversion to clausal form of first-order formulae resulting from the translation of $\mathcal{ALC}$ knowledge bases, makes use of a particular form of structural transformation Ξ [Baaz *et al.*, 1994], which is based on two mappings Ξ_1 and Ξ_2.

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 predicate symbol Q_λ and a literal $Q_\lambda(x_1, \ldots, x_n)$, where the x_i are the free variables of $\phi|_\lambda$, Q_λ does not occur in ϕ and two symbols Q_λ and $Q_{\lambda'}$ are equal iff ϕ_λ and $\phi_{\lambda'}$ are variant formulae. Ξ_1 uses *definitions* of the form

$$\mathrm{Def}_\lambda(\phi) = \forall\, x_1, \ldots, x_n\colon Q_\lambda(x_1, \ldots, x_n) \to \phi|_\lambda.$$

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

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

where λ is maximal in $\Lambda \cup \{\lambda\}$ with respect to the prefix ordering on positions. Let $\mathrm{Pos}_d(\phi)$ be the set of positions of subformulae of ϕ corresponding to positions of non-primitive concepts in the knowledge base Γ. By Ξ_1 we denote the transformation taking $\overline{\Pi}(\Gamma)$ to its *definitional form*

$$\mathrm{Def}_{\mathrm{Pos}_d(\overline{\Pi}(\Gamma))}(\overline{\Pi}(\Gamma)).$$

Note that in this case, every predicate symbol Q_λ is a unary predicate associated with a concept C (although not necessarily uniquely associated). Thus, we will henceforth denote Q_λ by p_C. By Ξ_2 we denote the function which produces for every unary predicate symbol p occurring in $\Xi_1\overline{\Pi}(\Gamma)$ the conjunction of all formulae

$$\forall x\colon p(x) \to p_{\blacktriangledown}(x).$$

Finally, let $\Xi\overline{\Pi}(\Gamma) = \Xi_1\overline{\Pi}(\Gamma) \wedge \Xi_2\Xi_1\overline{\Pi}(\Gamma)$.

Theorem 2. *Let Γ be any knowledge base. $\Xi\overline{\Pi}(\Gamma)$ can be computed in polynomial time, and Γ is satisfiable iff $\Xi\overline{\Pi}(\Gamma)$ is satisfiable.*

The clausal form of $\Xi\overline{\Pi}(\Gamma)$ consists of three types of clauses: (i) clauses stemming from terminological axioms

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

which all contain an occurrence of the negative literal $\neg p_{\blacktriangledown}(x)$, (ii) clauses stemming from formulae introduced by Ξ_1 and Ξ_2 which all contain an occurrence of some negative literal $\neg p_C(x)$, and (iii) clauses originating from the translation of assertional sentences which are ground unit clauses.

Our selection function $S_{\mathcal{TAB}}$ selects the literal $\neg p_{\blacktriangledown}(x)$ in clauses of type (i) and $\neg p_C(x)$ in clauses of types (ii). In addition, a binary literal of the form $\neg p_R(s,t)$ is selected whenever s is a ground term and t is a variable. All clauses stemming from a terminological sentence or from an additional formula introduced by $\overline{\Xi}$ contain negative literals, one of which is selected. We will mark selected literals by $\cdot_+$. For Theorem 3 an arbitrary reduction ordering $\succ$ may be used.

For every concept C and every role R, which may possibly occur in a knowledge base during a satisfiability test, there exist corresponding predicate symbols p_C and p_R in the clausal form of $\overline{\Xi}\Pi(\overline{\Gamma})$. Likewise every object a is associated with a term t_a.

We show that every application of one of the transformation rules is simulated by at most two resolution inference steps.

1. The $\Rightarrow_{\sqcap}$ rule, by two resolution inference steps between the 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. The $\Rightarrow_{\sqcup}$ rule, by an inference step between the ground unit clause $\{p_{C\sqcup D}(t_a)\}$ and $\{\neg p_{C\sqcup D}(x)_+, p_C(x), p_D(x)\}$, followed by an application of the splitting rule to the conclusion $\{p_C(t_a), p_D(t_a)\}$ which will generate two branches, one on which the set of clauses contains $\{p_C(t_a)\}$ and one on which it contains $\{p_D(t_a)\}$.

3. The $\Rightarrow_{\exists}$ rule, by resolution inference steps between the clauses $\{p_{\exists R.C}(t_a)\}$, $\{\neg p_{\exists R.C}(x)_+, p_R(x, f(x))\}$, 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 object $b \in \mathsf{I}$ introduced by the $\Rightarrow_{\exists}$ rule, that is, $t_b = f(t_a)$.

4. The $\Rightarrow_{\forall}$ rule, by two consecutive resolution inference steps. Here, the set of clauses contains $\{p_{\forall R.C}(t_a)\}$ and $\{p_R(t_a, t_b)\}$. First, the clause $\{p_{\forall R.C}(t_a)\}$ is resolved with $\{\neg p_{\forall R.C}(x)_+, \neg p_R(x,y), p_C(y)\}$ to obtain the clause $\{\neg p_R(t_a, y)_+, p_C(y)\}$. Then the conclusion is resolved with $\{p_R(t_a, t_b)\}$ to obtain $\{p_C(t_b)\}$.

5. The $\Rightarrow_{\bot}$ rule, by two consecutive inference steps using $\{p_A(t_a)\}$, $\{p_{\neg A}(t_a)\}$, and $\{\neg p_{\neg A}(x)_+, \neg p_A(x)\}$ (which is the clausal form of the definition of $\neg A$) to derive the empty clause.

6. The $\Rightarrow_{\sqsubseteq}$ rule is simulated as follows. Since the object a occurs in Γ, the corresponding term t_a occurs in our set of clauses. In particular, there is a ground unit clause $\{p_D(t_a)\}$, a clause $\{\neg p_D(x)_+, p_{\blacktriangledown}(x)\}$ introduced by Ξ_2, and a clause $\{\neg p_{\blacktriangledown}(x)_+, p_C(x)\}$ stemming from the translation of the terminological sentence $\top \sqsubseteq C$. By two resolution inference steps we obtain $\{p_C(t_a)\}$ which corresponds to the sentence $a \in C$ added by $\Rightarrow_{\sqsubseteq}$ to Γ.

Note that all the inference steps strictly obey the restrictions enforced by the selection function $S_{\mathcal{TAB}}$ and are in accordance with the resolution calculus. This proves:

Theorem 3. *The resolution proof system with selection function $S_{\mathcal{TAB}}$ p-simulates the tableaux proof system for $\mathcal{ALC}$.*

Interestingly, factoring plays no role for the clause sets under consideration, that is, the only possible factoring steps are condensations of ground conclusions. Moreover, no resolution inference steps other than those of the simulation are possible. Thus, the following stronger result holds.

Theorem 4. *Let Γ be a knowledge base and N the clausal form of $\overline{\Xi}\Pi(\overline{\Gamma})$. Then the search space of the resolution procedure for N can be polynomially reduced to the search space of the tableaux procedure for Γ.*

5 Termination

By $\mathsf{C}(\Gamma, a)$ we denote the set of all concepts C such that $a \in C$ is an element of the knowledge base Γ. Two objects $a, b \in \mathsf{I}$ are Γ-*equivalent*, denoted by $a \equiv_\Gamma b$, if $\mathsf{C}(\Gamma, a) = \mathsf{C}(\Gamma, b)$. If $b \prec_{\mathsf{I}} a$ and $a \equiv_\Gamma b$, then b is a *witness* for a. Similarly, let $\mathsf{P}(N, t)$ denote the set of predicate symbols $\{p \mid \{p(t)\} \in N\}$ in a clause set N. (Remember whenever $a \in \neg A$ is an element of Γ, then the positive clause $\{p_{\neg A}(t_a)\}$ is an element of N.)

The strategy (S) employed by Buchheit et al. [1993] restricts the application of rules as follows: (i) apply a transformation to an introduced object only if no rule is applicable to an object $a \in \mathsf{O} \setminus \mathsf{I}$, (ii) apply a rule to an introduced object a only if no rule is applicable to an introduced object b such that $b \prec_{\mathsf{I}} a$, (iii) apply $\Rightarrow_{\exists}$ only if no other rule is applicable, and (iv) apply $\Rightarrow_{\exists}$ to an introduced object a in Γ only if there is no witness for a. Restrictions (i)–(iii) ensure that whenever $\Rightarrow_{\exists}$ becomes applicable to an introduced object a in a knowledge base Γ, then for every Γ' with $\Gamma \Rightarrow^*_{\mathcal{TAB}} \Gamma'$ we have $\mathsf{C}(\Gamma, a) = \mathsf{C}(\Gamma', a)$. The strategy guarantees the termination of the tableaux proof system.

Restriction (iv) may be viewed as an instance of the Leibniz principle, identifying two objects which are indistinguishable with respect to their properties. Since we confine ourselves to applications of this principle to introduced objects, it is sufficient to consider properties expressible by concepts. In this case, the principle can be expressed as a set of first-order formulae of the form

$$\forall x, y\colon p_{C_1}(x) \wedge \ldots \wedge p_{C_n}(x) \wedge p_{C_1}(y) \wedge \ldots \wedge p_{C_n}(y) \rightarrow x = y,$$

with the antecedents representing all possible truth assignments to concepts and subconcepts occurring in Γ. (That is, each p_{C_i} corresponds to a concept C_i in Γ and for every subconcept C in Γ, either p_C or $p_{\neg C}$ occurs in each formula.) The notion of $\equiv_\Gamma$-equivalence also has a non-monotonic aspect: During a tableaux derivation it can happen that $a \equiv_\Gamma b$ at one state and $a \not\equiv_\Gamma b$ at a later state. But restrictions (i)–(iii) ensure that eventually either $a \equiv_\Gamma b$ or $a \not\equiv_\Gamma b$ holds for all future states in the derivation. Furthermore, it is assumed that for concepts

C not occurring in $\mathsf{C}(\Gamma, a)$ or $\mathsf{C}(\Gamma, b)$ we can assume that neither a nor b are in the semantical interpretation of C.

To account for these aspects and to reduce the computational overhead introduced by these formulae we choose to add a special expansion rule instead.

Blocking:
$$\frac{N}{N \cup \{t_a \approx t_b\}}$$

where (i) t_a and t_b are distinct ground, functional terms, and (ii) $\mathsf{P}(N, t_a) = \mathsf{P}(N, t_b)$. ('$\approx$' is the equality symbol.)

This rule is sound. Using the correspondence between the application of one of the propagation rules and particular resolution inference steps, we can restrict ourselves to a corresponding strategy in the resolution proof system. It follows that whenever resolution inference steps corresponding to $\Rightarrow_\exists$ become applicable to a term t_a in the clause set N, then for every clause set N' derivable from N we have $\mathsf{P}(N, t_a) = \mathsf{P}(N', t_a)$.

On the basis of the ordering $\prec_{\mathsf{I}}$ on introduced objects we will now define a reduction ordering $\succ_{\mathcal{TAB}}$. Suppose a and b are introduced objects and t_a and t_b are the corresponding terms according to the simulation result above. Let $\succ_{\mathcal{TAB}}$ be a reduction ordering with the following properties: (i) if $b \prec_{\mathsf{I}} a$ then $t_a \succ_{\mathcal{TAB}} t_b$, and (ii) for arbitrary non-equality atoms A, if $t_a \succ_{\mathcal{TAB}} t_b$, then $A[t_a] \succ_{\mathcal{TAB}} A[t_b]$ and $A[t_a] \succ_{\mathcal{TAB}} (t_a \approx t_b)$. It is not difficult to show that such a reduction ordering exists. Note that it is sufficient that only ground expressions are ordered by $\succ_{\mathcal{TAB}}$.

Assume now that by restriction (iv) of strategy (S) the rule $\Rightarrow_\exists$ is not applicable to an introduced object a in Γ, because there is a witness b for a. Then there are terms t_a and t_b such that $\mathsf{P}(N, t_a) = \mathsf{P}(N, t_b)$. In this situation an application of the blocking expansion rule will add an equation $\{t_a \approx t_b\}$ to N. Since $b \prec_{\mathsf{I}} a$, it follows $t_a \succ_{\mathcal{TAB}} t_b$, and the ground clauses $\{p_C(t_b)\}$ and $\{t_b \approx t_a\}$ are smaller than $\{p_C(t_a)\}$ with respect to $\succ_{\mathcal{TAB}}$. Also, $\{p_C(t_b)\}$ and $\{t_b \approx t_a\}$ logically imply $\{p_C(t_a)\}$. Consequently, the clause $\{p_C(t_a)\}$ is redundant and does not participate in any further inferences. This mimics restriction (iv) of strategy (S). To establish logical implications of this form, the redundancy elimination algorithm will require some form of equality reasoning, for example, superposition. In our special case, all that is required are one-step rewrite transformations.

Theorem 5. *The strategy (S) for tableaux proof systems can be polynomially simulated by blocking and redundancy elimination as outlined above.*

It is now straightforward to show that any inference in our resolution proof system terminates.

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

Corollary 7. *The resolution proof system and the tableaux proof system have the same time complexity, namely NEXPTIME [Buchheit* et al.*, 1993].*

6 Optimisations

In practice a principal cause for intractability is the presence of a large number of terminological sentences. Every application of the $\Rightarrow_{\dot\sqsubseteq}$ rule to an object a and terminological sentence $\top \dot\sqsubseteq \neg C \sqcup D$ will be followed by an application of $\Rightarrow_\sqcup$ to $a \in \neg C \sqcup D$. The number of branches in the search space generated in this way is too large to be manageable for implementations relying on chronological backtracking to systematically investigate all the branches.

As indicated by Horrocks and Patel-Schneider [1998] and Hustadt and Schmidt [1998] one possible optimisation is the use of more sophisticated backtracking techniques like backjumping or branch condensing. However, it is even more desirable to avoid unnecessary branching in the first place. A closer look at the intention behind the introduction of $\blacktriangledown$ in the modified translation $\overline{\Pi}$ reveals one possible optimisation in this direction. Suppose the knowledge base contains a terminological sentence of the form $\neg A \dot\sqsubseteq C$. Using the standard embedding Π we obtain a clause $\{p_A(x), p_C(x)\}$ which contains no negative literal we could select. Using $\overline{\Pi}$ we obtain $\{\neg p_{\blacktriangledown}(x), p_A(x), p_C(x)\}$ which contains a selectable literal. However, whenever for a terminological sentence $C \dot\sqsubseteq D$, the concept $\mathrm{nnf}(\neg C) \sqcup D$ contains a negative occurrence of a primitive concept A, the corresponding clauses under the standard embedding Π will contain a selectable negative literal $\neg p_A(x)$. The transformation Ξ_1 is modified such that these occurrences are preserved. Now the selection function can select an arbitrary negative literal. For example, if we have a terminological sentence of the form $A_1 \sqcap \ldots \sqcap A_n \dot\sqsubseteq C$, the selection function can choose an arbitrary $\neg p_{A_i}(x)$ among the negative literals in $C_1 = \{\neg p_{A_1}(x), \ldots, \neg p_{A_n}(x), p_C(x)\}$. This prevents any inference with C_1 until a unit clause $\{A_i(t_a)\}$ has been derived. Independent of these consideration, this optimisation has been incorporated in the FaCT system [Horrocks, 1997].

From correspondences with propositional dynamic logic it is known that the satisfiability problem for general $\mathcal{ALC}$ knowledge bases is in EXPTIME. The algorithms presented in Sections 2 and 4 require double exponential time in the worst case. Buchheit et al. [1993] note that this can be improved by caching contradictory sets $\mathsf{C}(\Gamma, a)$ of previously investigated branches introduced by applications of the $\Rightarrow_\sqcup$ rule. This has been formalised in [Donini *et al.*, 1996]. Evidently, this form of caching will have the same effects for the resolution procedure described in this paper.

7 Conclusion

The prime motivation for this work has been our interest in possible links between different proof systems for description logics and modal logics. This paper focuses on a particular tableaux proof system for description logics with general inclusion sentences and shows how this system and certain optimisations can be simulated with polynomial overhead in the context of reso-

lution. Our results provide new insight into the relative proof complexity of these systems similar to corresponding results for propositional logic. Although we have considered only the logic $\mathcal{ALC}$, our results may be extended to description logics with role conjunction and role hierarchies. We expect similar results can also be obtained for other forms of tableaux proof systems or sequent calculi. Resolution procedures following tableaux proof strategies have the advantage that proofs may be easily translated back into tableaux or sequent-style proofs of the original source logic. Related work on backward translation is by Caferra and Demri [1993; 1995].

The resolution decision procedure described in this paper offers just one of many possible search strategies. Other resolution strategies utilised in the literature, mentioned in the Introduction, are implemented by ordering strategies which do not rely on blocking or loop-checking techniques. Such techniques are also not needed in the ordered chaining calculus for modal logics with transitive modalities, or $\mathcal{ALC}$ with transitive roles [Ganzinger *et al.*, 1999].

Although experimental results with SPASS using ordered resolution are encouraging [Hustadt and Schmidt, 1997; Hustadt *et al.*, 1998], there are classes of problems on which tableaux proof systems have better performance. The results of this paper now provide a basis for the scientific testing of the comparative performance of the two orthogonal strategies for resolution proof systems, and for establishing guidelines indicating which strategy is most appropriate for particular classes of problems.

References

[Baaz *et al.*, 1994] M. Baaz, C. Fermüller, and A. Leitsch. A non-elementary speed-up in proof length by structural clause form transformation. In *Proc. LICS'94*, pages 213–219. IEEE Computer Society Press, 1994.

[Bachmair and Ganzinger, 1998] L. Bachmair and H. Ganzinger. Equational reasoning in saturation-based theorem proving. In W. Bibel and P. Schmitt, eds., *Automated Deduction: A Basis for Applications, Vol. I*, pages 353–397. Kluwer, 1998.

[Buchheit *et al.*, 1993] M. Buchheit, F. M. Donini, and A. Schaerf. Decidable reasoning in terminological knowledge representation systems. *J. Aritificial Intelligence Research*, 1:109–138, 1993.

[Caferra and Demri, 1993] R. Caferra and S. Demri. Cooperation between direct method and translation method in non classical logics: Some results in propositional S5. In R. Bajcsy, ed., *Proc. IJCAI-93*, pages 74–79. Morgan Kaufmann, 1993.

[de Nivelle, 1998] H. de Nivelle. A resolution decision procedure for the guarded fragment. In C. Kirchner and H. Kirchner, eds., *Proc. CADE-15*, volume 1421 of *LNAI*, pages 191–204. Springer, 1998.

[Demri, 1995] S. Demri. A hierarchy of backward translations: Applications to modal logics. In M. de Glas and Z. Pawlak, eds., *Proc. WOCFAI 95*, pages 121–132. Angkor, 1995.

[Donini *et al.*, 1996] F. M. Donini, G. De Giacomo, and F. Massacci. EXPTIME tableaux for $\mathcal{ALC}$. In L. Padgham, *et al.*, ed., *Proc. DL'96,* AAAI Technical Reports Series WS-96-05, pages 107–110, 1996.

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

[Ganzinger *et al.*, 1999] H. Ganzinger, U. Hustadt, C. Meyer, and R. A. Schmidt. A resolution-based decision procedure for extensions of K4. To appear in volume 2 of *Advances in Modal Logic*. CSLI Publications, 1999.

[Grädel, 1998] E. Grädel. Guarded fragments of first-order logic: A perspective for new description logics? In *Proc. DL'98*, volume 11 of *CEUR Electronic Workshop Proceedings*, 1998.

[Horrocks and Patel-Schneider, 1998] I. Horrocks and P. F. Patel-Schneider. Optimising propositional modal satisfiability for description logic subsumption. In *Proc. AISC'98*, 1998.

[Horrocks, 1997] I. Horrocks. *Optimising Tableaux Decision Procedures for Description Logics*. PhD thesis, University of Manchester, Manchester, UK, 1997.

[Hustadt and Schmidt, 1997] U. Hustadt and R. A. Schmidt. On evaluating decision procedures for modal logic. In M. E. Pollack, ed., *Proc. IJCAI-97*, pages 202–207. Morgan Kaufmann, 1997.

[Hustadt and Schmidt, 1998] U. Hustadt and R. A. Schmidt. Simplification and backjumping in modal tableau. In H. de Swart, ed., *Proc. TABLEAUX'98*, volume 1397 of *LNAI*, pages 187–201. Springer, 1998.

[Hustadt and Schmidt, 1999] U. Hustadt and R. A. Schmidt. Issues of decidability for description logics in the framework of resolution. To appear in R. Caferra and G. Salzer, eds., *Proc. FTP'98*, *LNAI*. Springer, 1999.

[Hustadt *et al.*, 1998] U. Hustadt, R. A. Schmidt, and C. Weidenbach. Optimised functional translation and resolution. In H. de Swart, ed., *Proc. TABLEAUX'98*, volume 1397 of *LNAI*, pages 36–37. Springer, 1998.

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

[Plaisted and Zhu, 1997] D. A. Plaisted and Y. Zhu. *The Efficiency of Theorem Proving Strategies*. Vieweg, 1997.

[Schmidt, 1999] R. A. Schmidt. Decidability by resolution for propositional modal logics. To appear in *J. Automated Reasoning*, 1999.