University of Ostrava

Institute for Research and Applications of Fuzzy Modeling

Non-clausal Resolution Theorem Proving for Description Logic

Hashim Habiballa

Research report No. 66

2005

Submitted/to appear:

FTP'2005, International Workshop on First-Order Theorem Proving, Koblenz, Germany

Supported by:

Research grant of Czech Ministry of Education - MSM 6198898701

University of Ostrava
Institute for Research and Applications of Fuzzy Modeling
30. dubna 22, 701 03 Ostrava 1, Czech Republic

tel.: +420-69-6160234 fax: +420-69-6120 478
e-mail: Hashim.Habiballa@osu.cz

Non-clausal Resolution Theorem Proving for Description Logic

Hashim Habiballa *

Institute for Research and Applications of Fuzzy Modeling
and Department of Computer Science
University of Ostrava
30. dubna 22
Ostrava, Czech Republic
Hashim.Habiballa@osu.cz
http://www.volny.cz/habiballa/index.htm

Abstract. The article presents refutational resolution theorem proving system for Description Logic (DL) based on general (non-clausal) resolution rule. There is also presented unification algorithm handling existentiality without the need of skolemization in first-order logic. Its idea follows from general resolution with existentiality for first-order logic. When the prover is constructed it provides the deductive system, where existing resolution strategies and its implementations may be used.

keywords: Automated theorem proving, non-clausal resolution, general resolution, unification, description logic.

1 Introduction

Description logic covers several deductive systems like well studied tableaux algorithm [Lu05] or clausal form resolution [Hu00]. Tableaux algorithm doesn't require any transformations (it may use some rewrite rules), but from the implementation point of view this procedure is slightly "non-deterministic", when one performs usage of γ and δ rules. On the other hand there is more practical deductive system based on resolution principle with unification for first-order logic, which allows to use variety of strategies, techniques and also implementations. Conversion to clausal form destroys the original meaning of a formula and requires removal of existentiality, which is also present in DL. We will present refutational resolution theorem proving system for enriched ALC-like DL ($RRTP_{DL}$) based on general (non-clausal) resolution principle in first-order logic (FOL) [Ba97], [Mu82]. The system may handle formulas without any transformation algorithms. It requires only more complex unification algorithm based on polarity criteria and quantifier mapping. The below presented idea have its origin in implementation of non-clausal resolution theorem prover [Ha05].

2 Background from first-order logic

2.1 General resolution - brief overview

For the purposes of ($RRTP_{DL}$) we will use generalized principle of resolution, which is defined in the Handbook [Ba01] concerning basics of the theory (or [Ba97]).

* This work was supported by research grant of Czech Ministry of Education - MSM 6198898701

General resolution

$$\frac{F[G] \quad F'[G]}{F[G/\bot] \vee F'[G/\top]} \tag{1}$$

where F and F' are formulas - premises of first-order logic and G represents an occurrence of a subformula (we may consider atoms only) of F and F'. The expression below the line means the resolvent of premises on G. Every occurrence of G is replaced by false in the first formula and by true in the second one. It is also called F the positive, F' the negative premise, and G the resolved subformula.

Example 1. **General resolution with equivalence**
1. $a \wedge c \leftrightarrow b \wedge d$ (axiom)
2. $a \wedge c$ (axiom)
3.$\neg[b \wedge d]$ (axiom) - negated goal
4. $[a \wedge \bot] \vee [a \wedge \top]$ (resolvent from (2), (2) on c) $\Rightarrow$ a
5. $[a \wedge \bot] \vee [a \wedge \top \leftrightarrow b \wedge d]$ ((2), (1) on c) $\Rightarrow a \leftrightarrow b \wedge d$
6. $\bot \vee [\top \leftrightarrow b \wedge d]$ ((4), (5) on a) $\Rightarrow b \wedge d$
7. $\bot \wedge d \vee \top \wedge d$ ((6), (6) on b) $\Rightarrow$ d
8. $b \wedge \bot \vee b \wedge \top$ ((6), (6) on d) $\Rightarrow b$
9. $\bot \vee \neg[\top \wedge d]$ ((8), (3) on b) $\Rightarrow \neg d$
10. $\bot \vee \neg\top$ ((7), (9) on d) $\Rightarrow \bot$ (refutation)

When trying to refine the general resolution rule into first-order logic and description logic, it is important to devise sound unification algorithm. Standard unification algorithms require variables to be treated only as universally quantified ones. We will present more general unification algorithm, which simulates skolemization without the need of real performation of it. It should be stated that the following unification process doesn't allow an occurrence of the equivalence connective. It is needed to remove equivalence by the following rewrite rule: $A \leftrightarrow B \Rightarrow [A \rightarrow B] \wedge [B \rightarrow A]$.

We assume that the language and semantics of first-order logic (FOL) is standard. We use terms - individuals $(a, b, c, ...)$, functions (with n arguments) $(f, g, h, ...)$, variables $(X, Y, Z, ...)$, predicates(with n arguments) $(p, q, r, ...)$, logical connectives $(\wedge, \vee, \rightarrow, \neg)$, quantifiers $(\exists, \forall)$ and logical constants $(\bot, \top)$.

Definition 1. ***Structural notions of a FOL formula***
Let F be a formula of FOL then the structural mappings Sub (subformula), Sup (superformula), Pol (polarity) and Lev (level) are defined as follows:

$F = G \wedge H$ *or* $F = G \vee H \Rightarrow$	$Sub(F) = \{G, H\}$, $Sup(G) = F$, $Sup(H) = F$ $Pol(G) = Pol(F)$, $Pol(H) = Pol(F)$
$F = G \rightarrow H \Rightarrow$	$Sub(F) = \{G, H\}$, $Sup(G) = F$, $Sup(H) = F$ $Pol(G) = -Pol(F)$, $Pol(H) = Pol(F)$
$F = \neg G \Rightarrow$	$Sub(F) = \{G\}$, $Sup(G) = F$ $Pol(G) = -Pol(F)$
$F = \exists \alpha G$ *or* $F = \exists \alpha G \Rightarrow$ *(α is a variable)*	$Sub(F) = \{G\}$, $Sup(G) = F$ $Pol(G) = Pol(F)$
F *is a formula*	$Sup(F) = \emptyset \Rightarrow Lev(F) = 0$, $Pol(F) = 1$ $Sup(F) \neq \emptyset \Rightarrow Lev(F) = Lev(Sup(F)) + 1$

For mappings Sub and Sup reflexive and transitive closures Sub^* and Sup^* are defined recursively as follows:
1. $Sub^*(F) \supseteq \{F\}$, $Sup^*(F) \supseteq \{F\}$
2. $Sub^*(F) \supseteq \{H|G \in Sub^*(F) \wedge H \in Sub(G)\}$, $Sup^*(F) \supseteq \{H|G \in Sup^*(F) \wedge H \in Sup(G)\}$

These structural mappings provide framework for assignment of quantifiers to variable occurrences. Subformula and superformula mappings and its closures encapsulate essential hierarchical information of a formula structure. Level gives the ordering with respect to the scope of variables (which is essential for skolemization simulation - unification is restricted for existential varibles). Polarity enables to decide the global meaning of a variable (e.g. globally an existential variable is universal if its quantification subformula has negative polarity). Sound unification requires further definitions on variable quantification.

Definition 2. ***Variable assignment, substitution and significance***
Let F be a formula of FOL, $G = p(t_1, ..., t_n) \in Sub^*(F)$ atom in F and α a variable occurring in t_i. Variable mappings Qnt(quantifier assignment), Sbt (variable substitution) and Sig(significance) are defined as follows:

$Qnt(\alpha) = \{Q\alpha H|Q = \exists \vee Q = \forall, H, I \in Sub^*(F), Q\alpha H \in Sup^*(G),$
$\forall Q\alpha I \in Sup^*(G) \Rightarrow Lev(Q\alpha I) < Lev(Q\alpha H)\}$
For substitution of term t' into α in F ($F[\alpha/t']$) it holds:
$Sbt(Qnt(\alpha)) = t'$
$Sig(\alpha) = 1$ *iff variable is significant,* $Sig(\alpha) = 0$ *iff variable is not significant w.r.t. existential substitution.*

Note that with Qnt mapping (assignment of first name matching quantifier variable in a formula hierarchy from bottom) we are able to distinguish between variables of the same name and there is no need to rename any variable. Sbt mapping holds substituted terms in a quantifier and there is no need to rewrite all occurrences of a variable when working with this mapping within unification. It is also clear that if $Qnt(\alpha) = \emptyset$ then α is a free variable. These variables could be simply avoided by introducing new universal quantifiers to F. Significance mapping is important for differentiating between original formula universal variables and newly introduced ones during proof search (an existential variable can't be bounded with it).

Lemma 1. ***Free variables quantification***
Let F be a formula of FOL and α be a free variable in F ($Qnt(\alpha) = \emptyset$), then the formula $F' = \forall\alpha F$ is equivalent to F.

Before we can introduce standard unification algorithm, we should formulate variable unification restriction for existential and universal variable through the notion of globally universal and globally existential variable (it simulates conversion into prenex normal form).It is clear w.r.t. skolemization technique that an existential variable can be substituted into an universal one only if all globally universal variables over the scope of it have been already substituted by a term. Skolem functors function in the same way.

Definition 3. ***Global quantification and variable unification restriction***
Let F be a formula without free variables and α be a variable occurring in a term of F.

1. *α is globally universal variable iff* $(Qnt(\alpha) = \forall\alpha H \wedge Pol(Qnt(\alpha)) = 1)$ *or* $(Qnt(\alpha) = \exists\alpha H \wedge Pol(Qnt(\alpha)) = -1)$
2. *α is globally existential variable iff* $(Qnt(\alpha) = \exists\alpha H \wedge Pol(Qnt(\alpha)) = 1)$ *or* $(Qnt(\alpha) = \forall\alpha H \wedge Pol(Qnt(\alpha)) = -1)$

Let F_1 be a formula and α be a variable occurring in F_1, F_2 be a formula, t be a term occurring in F_2 and β be a variable occurring in F_2. Variable Unification Restriction (VUR) for (α,t) holds if one of the conditions 1. and 2. holds:

1. *α is a globally universal variable and $t \neq \beta$, where β is a globally existential variable and α not occurring in t (non-existential substitution)*
2. *α is a globally universal variable and $t = \beta$, where β is a globally existential variable and $\forall F \in Sup^*(Qnt(\beta))$, $F = Q\gamma G$, $Q \in \{\forall, \exists\}$, γ is a globally universal variable, $Sig(\gamma) = 1 \Rightarrow Sbt(\gamma) \neq \emptyset$ (existential substitution)*

Now we can define the most general unification algorithm based on recursive conditions (extended unification in contrast to standard MGU [Lt93]).

Definition 4. ***Most general unifier algorithm***
Let F_1, F_2 be formulas of FOL, $G = p(t_1, ..., t_n)$ be an atom occurring in F_1 and $G' = r(u_1, ..., u_n)$ be an atom occurring in F_2. Most general unifier (MGU)(substitution mapping) σ is obtained by following atom and term unification steps or the algorithm returns fail-state for unification.
Atom unification

1. *if $n = 0$ and $p = r$ then $\sigma = \emptyset$ and the unifier exists (success-state).*
2. *if $n > 0$ and $p = r$ then perform term unification for every pair (t_1, u_1); if for every pair unifier exists then $MGU(G, G') = \sigma$ (success state).*
3. *In any other case unifier doesn't exist (fail-state).*

Term unification (t', u')

1. *if $t' = a$, $u' = b$ are individual constants and $a = b$ then for (t',u') unifier exist (success-state).*
2. *if $t' = f(t'_1, ..., t'_m)$, $u' = g(u'_1, ..., u'_n)$ are function symbols with arguments and $f = g$ then unifier for (t', u') exists iff unifier exists for every pair (t'_1, u'_1), ..., (t'_n, u'_n) (success-state).*
3. *if $t' = \alpha$ is a variable and VUR for (t', u') holds then unifier exists for (t', u') holds and $(Sbt(\alpha) = u') \in \sigma$ (success-state).*
4. *if $u' = \alpha$ is a variable and VUR for (u', t') holds then unifier exists for (t', u') holds and $(Sbt(\alpha) = t') \in \sigma$ (success-state).*
5. *In any other case unifier doesn't exist (fail-state).*

With above defined notions it is simple to state the general resolution rule for FOL (without the equivalence connective). It extends the definition from [Ba97].

Definition 5. ***General resolution for first-order logic*** (GR_{FOL})

$$\frac{F[G_1, , ..., G_k] \quad F'[G'_1, ..., G'_n]}{F\sigma[G/\bot] \vee F'\sigma[G/\top]} \tag{2}$$

where σ is the union of the most general unifiers (mgu) of the atom pairs (G_1, G_i) and (G_1, G'_j), $G_1, ..., G_k, G'_1, ..., G'_n, G = G_1\sigma$. For every variable α in F or F', $(Sbt(\gamma) = \alpha) \cap \sigma = \emptyset \Rightarrow Sig(\alpha) = 1$ in F or F' iff $Sig(\alpha) = 1$ in $F\sigma[G/\bot] \vee F'\sigma[G/\top]$. F is called positive and F' is called negative premise, G represents an occurrence of an atom. The expression below the line represents the resolvent of premises on G.

Note that with Qnt mapping we are able to distinguish variables not only by its name (which may not be unique), but also with this mapping (it is unique). Sig property enables to separate variables, which were not originally in the scope of an existential variable. When utilizing the rule it should be set the Sig mapping for every variable in axioms and negated goal to 1. We present a very simple example of existential variable unification before we introduce the refutational theorem prover for FOL.

Example 2. It doesn't hold $\forall X \exists Y p(X, Y) \models \exists Y \forall X p(X, Y)$
and it holds $\exists Y \forall X p(X, Y) \models \forall X \exists Y p(X, Y)$.
(General Y for all X can't be deduced from Y specific for X but contrary it holds)
Source formulas (axioms) :
F0 : $\forall X \exists Y p(X, Y)$. F1 ($\neg$query) : $\forall Y \exists X \neg p(X, Y)$.
R[F1&F1] : $\bot \vee \top$. R[F0&F0] : $\bot \vee \top$.
In this example F0 and F1 can't resolve, since $\forall X \exists Y p(X, Y)$ and $\forall Y \exists X \neg p(X, Y)$ have no unifier. It is impossible to substitute universal X from F0 with X from F1, because X from F1 is existential and its superior variable Y is not assigned with a value. Counter-example with variable Y is the same instance and it is not allowed to substitute anything into an existential variable. However, in the next example a unifier exists.
Source formulas (axioms) :
F0 : $\exists Y \forall X p(X, Y)$. F1 ($\neg$query) : $\exists X \forall Y \neg p(X, Y)$.
R[F1&F0] : YES. R[F0&F1] : YES.
In the second case the universal variable X from F0 could be assigned with existential Y because Y in F1 has no superior variable. Then existential Y from F0 can substitute the universal Y from F1 for the same reason.

Now we entail above presented definitions by introduction of refutational theorem proving with the rule GR_{FOL}. We assume standard notions and theorems stated in [Ba01].

Definition 6. ***Refutational resolution theorem prover for FOL***
Refutational non-clausal resolution theorem prover for FOL ($RRTP_{FOL}$) is the inference system with the inference rule GR_{FOL} and sound auxiliary simplification rules for $\bot$, $\top$ (standard equivalencies for logical constants). A refutational proof of the goal G from the set of axioms $N = \{A_1, ..., A_m\}$ is a sequence of formulas $F_1, F_2, ..., F_n, \bot$, where F_i is an axiom from N, $\neg G$ or a resolvent from premises F_k and F_l $(k, l < i)$, where simplification rules may be applied to the resolvent. It is assumed that $Sig(\alpha) = 1$ for $\forall \alpha$ in $F \in N \cup \neg G$ formula, every formula in a proof has no free variable and has no quantifier for a variable not occurring in the formula.

Theorem 1. *$RRTP_{FOL}$ system is a sound inference system.*

Proof. The proof is based on standard properties of General resolution [Ba97], which is sound and complete rule in propositional logic (ground case in FOL) and could be lifted

also to non-ground cases without existentiality (MGU works with the concept of universal variables and skolems functors). By this means we are able to translate the soundness problem into clausal normal form logic, where soundness of resolution is preserved. Every formula of FOL could be transformed into formula in clausal normal form preserving satisfiability. The notion of globally universal and existential variable simulates the conversion into prenex normal form (i.e. a variable in original formula is universal in prenex normal iff it is globally universal and contrary). This property could be simply proved by standard equivalencies of FOL. Then the skolem functors (and constants as a special case) are introduced. Functors have arguments being universal variables prefixing the quantifier for the original existential variable. Unification in $RRTP_{FOL}$ has the same meaning. At first it is clear that two different globally existential variables can't be unified like two different skolem functors. Further an existential variable α can't be substituted into universal one if it has any universal variable γ in $Sup^*(Qnt(\alpha))$, where $Sbt(\gamma) = \emptyset$.

Theorem 2. *$RRTP_{FOL}$ system is a complete inference system.*

Proof. The proof can be done by the similar way like the previous. We can translate the problem into clausal normal form formulas. For every proof in the $RRTP_{FOL}$ there is equivalent proof in clausal normal form. It could be done by induction on the Sup mapping. It is clear that general resolution encapsulates several steps of clausal form resolution and existentiality could be again simulated skolem functors and vice-versa.

Proofs for both the properties are rather intuitive and should be done exactly. We presented only its essential ideas since the main intent of the article is to present RRTP for Description Logic. The non-clausal theorem prover for FOL is already implemented in the form of an experimental application [Ha00], [Ha05].

3 General resolution for description logic

In this section we will naturally build up the Refutational Resolution Theorem Prover for Description Logic $RRTP_{DL}$ based on previously presented general resolution rule for FOL. Whole the theory is based on mapping DL formulas into FOL formulas and hence the properties of interpretation and resolution are preserved. The paper works with standard interpretation structures and rules of FOL [Lt93] and also for ALC-like DL [Bd03]. We use atomic concepts to be bound with unary predicates $A_1, ..., A_n$ and atomic roles are bound with binary predicates $R_1, ..., R_m$. Terms are constants $a_1, ..., a_l$ or variables $x_1, ..., x_k$. There is also top and bottom concepts $(\top, \bot)$. Concepts may be constructed using negation $\neg$, conjunction $\sqcap$, disjunction $\sqcup$, implication $\rightarrow$ (introduction of implication will not affect the expressive power), restricted quantification $\forall R.C$ bounded with $\forall y(R(x, y) \rightarrow C(y))$, $\exists R.C$ bounded with $\exists y(R(x, y) \wedge C(y)))$. Instances of concepts could be written in the form $C(t)$, where t is a term. When speaking about instances of quantified concepts its meaning in FOL is equivalent to replacing the occurrence of x variable by t. For the purposes of the prover we extend the notion of roles. At first we enable to use the atomic role negation $\neg R$, which is equivalent to its FOL representation and in semantic meaning it relates to complement operation on the extent of R. We also enable to use negation, conjunction and disjunction of roles $(\neg, \sqcap, \sqcup)$ and we introduce notion

of full role ($\top_R$) and empty role ($\bot_R$). Interpretation of newly introduced full and empty roles is $\top_R^I = (\Delta^I \times \Delta^I)$ and $\bot_R^I = \emptyset$ (where $I = (\Delta^I, \cdot^I)$). We call all the formulas of DL - DL formulas.

Definition 7. ***Embedding of structural notions into FOL***
Let D be a DL formula. We call a formula F of FOL equivalent embedding formula by the following mapping $Emb(D) = F$ (embedding). All the above defined notions for FOL holds also for $Emb(D)$. Additionally we define a mapping Ext (extension) for D, which returns an explicit form of DL-formula with assigned terms to formulas and subformulas (new free variables x, y should have unique names). Further let $Ext_{Max} \in Ext$ be a maximal atomic extension, where every atomic concept (or atomic role) has assigned a term by Ext (or a term pair) and there are no other terms on non-atomic subformulas. For every DL formula F it holds $F \in Ext(F)$.

Let C, D be concepts and R, S be roles then: $Ext(A_i) \ni A_i(x)$, $Emb(A_i(t)) = A_i(t)$, $Ext(R_i) \ni R_i(x_1, x_2)$, $Emb(R_i(t_1, t_2) = R_i(t_1, t_2))$ (atomic concept and role)

For every $\bullet \in \{\sqcap, \sqcup, \rightarrow\}$ it holds: $Ext(C \bullet D) \ni (C \bullet D)(x), Ext((C \bullet D)(t)) \ni C(t) \bullet D(t)$ and $Ext(R \bullet S) \ni (R \bullet S)(x, y), Ext((R \bullet S)(t_1, t_2)) \ni R(t_1, t_2) \bullet S(t_1, t_2)$
$Emb((C \sqcap D)(t)) = C \wedge D$, $Emb((C \sqcup D)(t)) = C \vee D$, $Emb((C \rightarrow D)(t)) = C \rightarrow D$,$Emb((\neg C)(t)) = \neg C$, $Ext(\neg C) \ni (\neg C)(x)$, $Ext((\neg C)(t)) \ni \neg(C(t))$, $Emb((R \sqcap S)(t_1, t_2)) = R \wedge S$, $Emb((R \sqcup S)(t_1, t_2)) = R \vee S$, $Emb((R \rightarrow S)(t_1, t_2)) = R \rightarrow S$, $Emb((\neg R)(t_1, t_2)) = \neg R$, $Ext(\neg R) \ni (\neg R)(x_1, x_2)$, $Ext((\neg R)(t_1, t_2)) \ni \neg(R(t_1, t_2))$, (where atomic concepts and roles have free variables replaced by t)

For every $Q \in \{\forall, \exists\}$ it holds: $Ext(QR.C) \ni (QR.C)(x, y)$,
$Ext((QR.C)(t_1, t_2)) \ni QR(t_1, t_2).C(t_2)$, $Ext(QR.C)(t) \ni (QR.C)(t, y)$,
$Emb(\forall R.C)(t, y) = \forall y(X)$,
where $Emb(R.C)(t, y) = X = R(t, y) \rightarrow C(y)$, $Emb((\exists R.C)(t, y)) = \exists y(X)$,
where $Emb((R.C)(t, y)) = X = R(t, y) \wedge C(y)$,
$Emb(\top) = \top$, $Emb(\bot) = \bot$, $Emb(\top_R) = \top$, $Emb(\bot_R) = \bot$

For every DL formula D or DL formula part $R.C = D$, a variable y occurring in D or $R.C$, where $Emb(D) = F$, and formula mappings
$Map \in \{Sub, Sup, Sub^*, Sup^*, Pol, Lev\}$, it holds $Map(D) = Map(F)$. For a variable y in an atomic concept or role (C or R) and F it holds $Qnt(y) = Qnt(\alpha)$, where α is a variable occurring in $Emb(C)$ or $Emb(R)$. Sig and Sbt mappings could be defined subsequently. If not stated otherwise $Sig(\alpha) = 0$. No free variables should occur in $Emb(D)$.

Example 3. **Examples of structural mappings**
Let $C \rightarrow \exists R.C'$ be a DL formula D. Then there are some notions (not all possible) holding for D:
$Ext_{Max}(D) = C(x) \rightarrow \exists R(x, y).C'(y)$ and
$Pol(D) = 0$, $Pol(C) = -1$, $Pol(\exists R.C') = 1$, $Pol(C') = 1$, $Lev(D) = 0$, $Lev(\exists R.C') = 1$, $Lev(R.C) = 2$, $Lev(R) = 3$, $Sub(\exists R.C') = R(x, y) \rightarrow C'(y)$, $Sub^*(\exists R(x, y).C'(y)) = \{R(x, y) \rightarrow C'(y), R(x, y), C'(y)\}$,
$Sub(D) = \{C(x), \exists y(R(x, y) \rightarrow C'(y))\}$, $Sup(C'(y)) = R(x, y) \rightarrow C(y)$
For y in $C'(y)$ $Qnt(y) = \exists y(R(x, y) \rightarrow C(y))$

For the purposes of the prover we will need to use simplification rules for DL-formulas.

Definition 8. ***Simplification rules for*** $RRTP_{DL}$
Let C be a concept, R be a role and D be a DL formula. Auxiliary simplification rules follow (with additional use of the commutative and associative properties of the connectives):
$D \sqcap \bot \Rightarrow \bot$, $D \sqcap \top \Rightarrow D$, $D \sqcup \bot \Rightarrow D$, $D \sqcup \top \Rightarrow \top$, $D \rightarrow \bot \Rightarrow \neg D$, $D \rightarrow \top \Rightarrow \top$, $\bot \rightarrow D \Rightarrow \top$, $\top \rightarrow D \Rightarrow D$, $\neg\neg D \Rightarrow D$

$\forall \top_R.C \Rightarrow C$, $\forall \bot_R.C \Rightarrow \top$, $\forall R.\top \Rightarrow \top$, $\forall \top_R.\bot \Rightarrow \bot$, $\forall \bot_R.\bot \Rightarrow \top$
$\exists \top_R.\bot \Rightarrow \bot$,$\exists \top_R.\top \Rightarrow \top$, $\exists \bot_R.C \Rightarrow \bot$, $\bot_R \Leftrightarrow \bot$, $\neg\bot \Leftrightarrow \top$, $\neg\top \Leftrightarrow \bot$

For introduction of general resolution rule we have to construct unification algorithm for atomic concepts and atomic roles.

Definition 9. ***Unification algorithm for*** $RRTP_{DL}$
Let D_1, D_2 be formulas of DL, $G = C(t_1)$ be an atomic concept or $G = R(t_1, t_2)$ be an atomic role occurring in D_1 and $G' = C'(t'_1)$ be an atomic concept or $G' = R'(t'_1, t'_2)$ be anatomic role occurring in D_2. Most general unifier (MGU)(substitution mapping) σ is obtained by following atom and term unification steps or the algorithm returns fail-state for unification.
Atom unification

1. *if $G = C(t_1)$, $G' = C'(t'_1)$ and $C = C'$ then perform term unification for terms (t_1, t'_1); if for this pair unifier exists then $MGU(G, G') = \sigma$ (success state).*
2. *if $G = R(t_1, t_2)$, $G' = R'(t'_1, t'_2)$ and $R = R'$ then perform term unification for terms (t_1, t'_1), (t_2, t'_2); if for both the pairs unifier exists then $MGU(G, G') = \sigma$ (success state).*
3. *In any other case unifier doesn't exist (fail-state).*

Term unification (t', u')

1. *if $t' = a$, $u' = b$ are individual constants and $a = b$ then for (t',u') unifier exist (success-state).*
2. *if $t' = \alpha$ is a variable and VUR for (t', u') holds then unifier exists for (t', u') and $(Sbt(\alpha) = u') \in \sigma$ (success-state).*
3. *if $u' = \alpha$ is a variable and VUR for (u', t') holds then unifier exists for (t', u') and $(Sbt(\alpha) = t') \in \sigma$ (success-state).*
4. *In any other case unifier doesn't exist (fail-state).*

Definition 10. ***General resolution for Description Logic*** (GR_{DL})
Let D and D' be DL formulas projected by Ext_{Max}, G_i, G'_i be an atomic concept or role (a subformula of D, D').

$$\frac{D[G_1, , ..., G_k]) \quad D'[G'_1, ..., G'_n])}{(D\sigma[G/\circ] \vee D'\sigma[G/\bullet])} \tag{3}$$

where σ is the union of the Most General Unifiers (MGU) of the atom pairs (G_1, G_i) and (G_1, G'_j), $G_1, ..., G_k, G'_1, ..., G'_n, G = G_1\sigma$. For G being an atomic concept $\circ = \bot$, $\bullet = \top$, for G being an atomic role $\circ = \bot_R$, $\bullet = \top_R$. For every variable α in D or D', $(Sbt(\gamma) = \alpha) \cap \sigma = \emptyset \Rightarrow (Sig(\alpha) = 1$ in D or D' iff $Sig(\alpha) = 1$ in $D\sigma[G/\circ] \vee D'\sigma[G/\bullet])$.

Definition 11. ***Refutational resolution theorem prover for DL***
Refutational non-clausal resolution theorem prover for DL ($RRTP_{DL}$) is the inference system with the inference rule GR_{DL} and sound auxiliary simplification rules. A refutational proof of the goal G from the set of axioms $N = \{A_1, ..., A_m\}$ is a sequence of DL formulas with full terms on atoms (maximal atomic extension) $Ext_{Max}(D_1)$, $Ext_{Max}(D_2)$, ... , $Ext_{Max}(D_n)$, $\perp$, where D_i is an axiom from N, $\neg G$ or a resolvent from premises D_k and D_l $(k, l < i)$; simplification rules may be applied to the resolvent. It is assumed that $Sig(\alpha) = 1$ $\forall \alpha$ variable in $N \cup \neg G$.

The proof of soundness and completeness of the $RRTP_{DL}$ is straightforward. Since the FOL embedding is semantically equivalent to the interpretation of DL formulas, all the properties of $RRTP_{FOL}$ holding for FOL should also hold for $RRTP_{DL}$.

Now we present a simple example of a proof in the prover. Since the $RRTP_{FOL}$ is already implemented in the form of an experimental computer application GERDS [Ha05] it was used for proof generation in $RRTP_{DL}$ (with hand rewriting into DL).

Example 4. **Non-clausal resolution in** $RRTP_{DL}$
Suppose we have the following concepts:
person, male, female and *happy_child*
and *role is_child_of* representing the role of being X child of Y.
We would like to express the knowledge that:
If person is child of at least one male and one female, then it is a happy child (since has both mother and father).
It can be expressed in ALC-like DL with implication like:
$(person \sqcap \exists is\_child\_of.female \sqcap \exists is\_child\_of.male) \rightarrow happy\_child$
Then we will suppose the following instances:
is_child_of(johana, hashim). is_child_of(johana, lucie). male(hashim). female(lucie). person(johana).

Source formulas (axioms) :

$F0:$ $(person(X) \sqcap \exists is\_child\_of(X,Y).female(Y)$
$\sqcap \exists is\_child\_of(X,Y).male(Y)) \rightarrow happy\_child(X)$

$F1:$ $is\_child\_of(johana, hashim).$

$F2:$ $is\_child\_of(johana, lucie).$

$F3:$ $male(hashim).$

$F4:$ $female(lucie).$

$F5:$ $person(johana).$

$F6:$ $\neg happy\_child(johana).(\neg$ query$)$

$[F6\&F0]:$ $\bot \sqcup (person(johana) \sqcap \exists is\_child\_of(johana,Y).female(Y)$
$\sqcap \exists is\_child\_of(johana,Y).male(Y) \rightarrow \bot).$

$\Rightarrow R0:$ $\neg(person(johana) \sqcap \exists is\_child\_of(johana,Y).female(Y)$
$\sqcap \exists is\_child\_of(johana,Y).male(Y)).$

$[R0\&F5]:$ $\neg(\top \sqcap \exists is\_child\_of(johana,Y).female(Y)$
$\sqcap \exists is\_child\_of(johana,Y).male(Y)) \sqcup \bot.$

$\Rightarrow R1:$ $\neg(\exists is\_child\_of(johana,Y).female(Y)$
$\sqcap \exists is\_child\_of(johana,Y).male(Y)).$

$[R1\&F4]:$ $\neg(\exists is\_child\_of(johana, lucie).\top \sqcap \exists is\_child\_of(johana,Y).male(Y))$
$\sqcup \bot.$

$\Rightarrow R2:$ $\neg(is\_child\_of(johana, lucie).\top \sqcap \exists is\_child\_of(johana,Y).male(Y)).$

$[R2\&F3]:$ $\neg(is\_child\_of(johana, lucie).\top \sqcap \exists is\_child\_of(johana, hashim).\top)$
$\sqcup \bot.$

$\Rightarrow R3:$ $\neg(is\_child\_of(johana, lucie).\top \sqcap is\_child\_of(johana, hashim).\top).$

$[R3\&F2]:$ $\neg(\top_R.\top \sqcap is\_child\_of(johana, hashim).\top).$

$\Rightarrow R4:$ $\neg is\_child\_of(johana, hashim).\top.$

$[R4\&F1]:$ $\neg(\top_R.\top) \sqcup \bot.$

$\Rightarrow:$ $\bot$ (refutation)

4 Conclusions and further research

The above presented prover represents an alternative deductive system for DL. Its advantage is based on wide usage of resolution principle in FOL, where a lot of implementations and strategies exist. We mentioned an application GERDS, which utilizes General resolution in FOL and it seems to be natural to extend GERDS also for Description Logic. Since the Emb notion allows to use FOL-style representation, it will probably require only rewrite algorithm (w.r.t. Ext mapping). There are also efficient techniques like chain resolution [Ta03], which could be added to the inference engine. Another interesting and natural extension also arise in conjunction with fuzzy logic, where normal form transformations are hard to realise. Fuzzy Description Logic forms useful formalization since concepts are often "fuzzy" in real situations. Presented notions and inference systems also require solid proofs of soundness and complemeteness and other properties.

References

[Ba97] Bachmair L., Ganzinger H. A theory of resolution. Technical report: Max-Planck-Institut für Informatik, 1997.

[Ba01] Bachmair, L., Ganzinger, H. Resolution theorem proving. in Handbook of Automated Reasoning, MIT Press, 2001.

[Bd03] Baader et col. (eds.). The Description Logic Handbook - Theory, Interpretation and Applications. Cambridge Univ. Press, 2003.

[Ha00] Habiballa, H. Non-clausal resolution - theory and practice. Research report: University of Ostrava, 2000, http://www.volny.cz/habiballa/files/gerds.pdf

[Ha05] Habiballa, H. Non-clausal Resolution Theorem Prover. Research report, No.64: University of Ostrava, 2005, http://ac030.osu.cz/irafm/ps/rep64.ps.gz

[Hu00] Hustadt, U., Schmidt, R.A. Issues of Decidability for Description Logics in the Framework of Resolution. In G. Salzer and R. Caferra, editors, Automated Deduction in Classical and Non-Classical Logics, pp. 192-206. LNAI 1761, Springer 2000.

[Lt93] Letz, R. First-order calculi and proof procedures for automated deduction. Ph.D. thesis, TH Darmstadt, 1993

[Lu05] Lukasová, A. Reasoning in Description Logic with semantic tableau binary trees. Research report: Institute for Research and Applications of Fuzzy Modeling, University of Ostrava, 2005.

[Mu82] Murray, N. Completely non-clausal theorem proving. Artificial Intelligence, 18, 1982, 6785.

[Ta03] Tammet, T. Extending Classical Theorem Proving for the Semantic Web. CEUR Workshop Proceedings, Technical University of Aachen (RWTH), 2003.