A Hybrid Reasoning System for Terminologies and Clause Sets

Ken Kaneiwa
National Institute of Informatics
2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo 101-8430, Japan
kaneiwa@nii.ac.jp
http://research.nii.ac.jp/~kaneiwa

ABSTRACT
Description logics (DLs) theoretically explore knowledge representation and its reasoning in concept languages. However, due to the concept-oriented notion, these logics are not equipped with rule-based reasoning mechanisms for assertional knowledge bases; specifically, rules and facts in logic programming, or the interaction of rules and facts with terminology. In order to deal with the enriched reasoning, this paper presents a hybrid reasoning system for combining DL-knowledge bases (TBox and ABox) and first-order clause sets. The main result of this study is that a sound and complete *resolution* method for the composed knowledge bases is designed, and it has the feature of an effective deduction procedure such as Robinson's resolution principle.

KEY WORDS
Logic Programming, Knowledge Representation, Rule-based Reasoning

1 Introduction

Description logics are a theoretical foundation of knowledge representation and its reasoning in concept languages. Standard reasoning tasks in description logics are to decide satisfiability and subsumption of DL-concepts [3]. For many description logics, the tasks can be completely determined by applying tableau-like algorithms [8, 2]. On the other hand, for practical purposes of knowledge bases there is reasoning from rules and facts in logic programming [12] together with terminological knowledge. Thus, it is required that the reasoning algorithm deals with not only DL-knowledge bases but also clause sets in first-order logic (being able to represent rules and facts). To treat the issue, logic programming languages combining Horn clauses and description logics were proposed in [11, 5].

However, the combination with description logics is limited where concept and role names (corresponding to unary and binary predicates) do not appear in the head of each Horn clause. Namely, in Horn clauses with DL-concepts:

$$p_1(\vec{t_1}), \ldots, p_n(\vec{t_n}) \rightarrow q(\vec{t}),$$

the predicates $p_1, \ldots, p_n$ are either concept names, role names or ordinary predicates (n-ary predicates), but the predicate q must be an ordinary predicate. This limitation avoids occurring negative expressions or disjunctive expressions in the head, but we cannot completely obtain the expressivity of combining logic programming and description logics. For example, consider the following Horn clauses (facts and a rule):

$$\rightarrow acted(John, Mary, e_1)$$
$$\rightarrow died(Mary, e_2)$$
$$\rightarrow after(e_2, e_1)$$
$$acted(x, y, z_1), died(y, z_2), after(z_2, z_1), Human(x), Human(y) \rightarrow killed(x, y)$$

where $acted$, $died$ and $after$ are ordinary predicates, $Human$ is a concept name (as a unary predicate) and $killed$ is a role name (as a binary predicate). This rule means "if a human x acted against a human y at z_1 and the human y died at z_2 after z_1, then the fact that x killed y is concluded." In addition to them, consider the following equations of DL-concepts:

$$Murderer \equiv \exists killed.Human \sqcap Human$$
$$Human \equiv Male \sqcup Female$$

where $Murderer$, $Human$, $Male$ and $Female$ are concept names and $killed$ is a role name. These equations indicate "murderers are humans who have killed humans" and "humans are male or female." For the two kinds of logical form, if the head $killed(t_1, t_2)$ as a role assertion is derived from the rule, the concept equations imply $Human(t_2)$ and therefore $Male \sqcup Female(t_2)$ holds. However, the disjunctive assertion $Male \sqcup Female(t_2)$ exceeds the expressivity of Horn clauses[1].

In order to remedy the insufficient combination of logic programming and description logics, we need to embed *general* clauses and DL-concepts into a rule-based reasoning system. Resolution proof systems [13] for clausal forms in first-order logic have been generally used as rule-based reasoning methods for knowledge bases, e.g., logic programming is representative of them. As an unusual approach, a resolution system for description logics was proposed by Areces et.al. [1], based on the modal resolution system [6][2].

[1] In Section 3.3, we will consider such expressions derived from a knowledge base where concept and role names can be used in the head of each Horn clause, which are forbidden in [11, 5].

[2] Moreover, a resolution system for non-classical logics was developed by Gabbay and Reyle [7].

In this paper, we present a hybrid resolution system for combining (i) knowledge bases consisting of the TBox and ABox in the description logic $\mathcal{ALC}$ and (ii) clause sets in first-order logic. To make it easily resolve both ABox-statements and first-order clauses, we introduce a clausal form of DL-concepts (which we will call clausal concepts) and compose this form and first-order clauses. We generalize a resolution method by incorporating the following resolution rules:

1. resolution principle and assertional rules for the composition of first-order clauses and ABox-statements of clausal concepts
2. taxonomic rules for concept definitions of clausal concepts (as TBox-statements)

and unification for first-order terms in assertions of clausal concepts and n-ary predicates. Technically, the important point is that each resolution rule must be designed so as to refute clauses composed from DL-assertions and first-order clauses (called extended clauses); namely, its resolution step deletes an inconsistent pair $(E, \neg E)$ of literals in extended clauses.

2 Combining DL and First-order Clauses

2.1 Description Logic $\mathcal{ALC}$

A concept language in the basic description logic $\mathcal{ALC}$ [14] contains the set $\mathbf{C}$ of concept names A, the set $\mathbf{R}$ of role names R, and the set $\mathbf{I}$ of individual names a, b. The concepts in $\mathcal{ALC}$ (called $\mathcal{ALC}$-concepts) are constructed by concept names A, role names R, the connectives $\neg, \sqcap, \sqcup$, and the universal and existential quantifiers $\forall, \exists$.

Definition 2.1 *The set of $\mathcal{ALC}$-concepts C, D is defined inductively by the following:*

$$C, D \longrightarrow A \mid \neg C \mid C \sqcap D \mid C \sqcup D \mid \forall R.C \mid \exists R.C$$

Let $Male$ be a concept name, and let $has\text{-}child$ be a role name. Then, for instance, the $\mathcal{ALC}$-concept $\exists has\text{-}child.Male$ represents "individuals who have sons."

The meaning of $\mathcal{ALC}$-concepts are formally given by an interpretation in the following definition.

Definition 2.2 *A DL-interpretation is an ordered pair $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ of a non-empty set $\Delta^{\mathcal{I}}$ (called the universe of $\mathcal{I}$) and an interpretation function $\cdot^{\mathcal{I}}$ for $\mathbf{C} \cup \mathbf{R} \cup \mathbf{I}$ where $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ (in particular, $\bot^{\mathcal{I}} = \emptyset$ and $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$), $R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ and $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$. The DL-interpretation $\mathcal{I}$ is expanded to $\mathcal{ALC}$-concepts including connectives and quantifiers as follows:*

$$(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} - C^{\mathcal{I}}$$
$$(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$$
$$(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$$
$$(\forall R.C)^{\mathcal{I}} = \{d_1 \in \Delta^{\mathcal{I}} \mid \forall d_2[(d_1, d_2) \in R^{\mathcal{I}} \rightarrow d_2 \in C^{\mathcal{I}}]\}$$
$$(\exists R.C)^{\mathcal{I}} = \{d_1 \in \Delta^{\mathcal{I}} \mid \exists d_2[(d_1, d_2) \in R^{\mathcal{I}} \wedge d_2 \in C^{\mathcal{I}}]\}$$

This interpretation will be used to define the semantics of a first-order language with concept and role names.

2.2 First-order clauses with Concept and Role Names

Description logics do not deal with reasoning for facts A and rules $B_1, \ldots, B_n \rightarrow B$, where each A, B_i and B are atomic formulas, as in logic programming. We employ general clausal forms (not restricted to Horn clauses) in first-order logic to be embedded in knowledge base reasoning for description logics. For the compatibility with concept languages, concept names and role names in $\mathcal{ALC}$ are respectively used to denote unary predicates and binary predicates in a first-order language $\mathcal{L}$, including ordinary n-ary predicate names. Hence, the language $\mathcal{L}$ includes the set $\mathbf{P}$ of n-ary predicate names p with $\mathbf{C} \cup \mathbf{R} \subseteq \mathbf{P}$, the set $\mathbf{F}$ of n-ary function names f, the set $\mathbf{I}$ of individual names a, b (as constant names) and the set $\mathbf{V}$ of variables x, y. The set of terms t is defined inductively by: (i) individual names and variables are terms, and (ii) if f is an n-ary function name and $t_1, \ldots, t_n$ are terms, then $f(t_1, \ldots, t_n)$ is a term.

Definition 2.3 *The set of formulas (in language $\mathcal{L}$) is defined inductively by the following:*

1. *If $p \in \mathbf{P}$ is an n-ary predicate name and $t_1, \ldots, t_n$ are terms, then $p(t_1, \ldots, t_n)$ is an atomic formula (simply called atom).*
2. *If $l_1, \ldots, l_n, l'_1, \ldots, l'_m$ are atomic formulas, then $l_1, \ldots, l_n \rightarrow l'_1, \ldots, l'_m$ is a clausal form. In particular, it is called the empty clause if $n = m = 0$.*

$\forall F$ is the universal closure of F, i.e., $\forall x_1 \cdots \forall x_n F$ where $x_1, \ldots, x_n$ are all the free variables occurring in F. So $l_1, \ldots, l_n \rightarrow l'_1, \ldots, l'_m$ expresses the universal closure $\forall(\neg l_1 \vee \cdots \vee \neg l_n \vee l'_1 \vee \cdots \vee l'_m)$ in first-order logic. Let E be an expression. $Var(E)$ denotes the set of free variables occurring in E. A substitution is a mapping θ from a finite subset of $\mathbf{V}$ into the set of terms such that $\theta(x) \neq x$. The substitution is expanded to terms and formulas in the usual way of first-order logic. A substitution θ to variables occurring in E is denoted by $E\theta$ (called an instance of E). An expression E is ground if it is without variables. Let E_1, E_2 be expressions. A substitution θ is a unifier for E_1 and E_2 if $E_1\theta = E_2\theta$. A most general unifier for E_1 and E_2 is expressed by $mgu(E_1, E_2)$.

2.3 Extended Knowledge Bases

We define a knowledge base consisting of the TBox and ABox and a clause set. Let A be a concept name and C, D be $\mathcal{ALC}$-concepts. A TBox $\mathcal{T}$ is a set of TBox-statements of the form $C \equiv D$. Note that we use TBoxes that are acyclic and sets of concept definitions of the form $A \equiv C$. Let a, b be individual names and R be a role name. An

ABox $\mathcal{A}$ is a set of ABox-statements of the forms $C(a)$, $R(a,b)$. Normally, a DL-knowledge base is defined as an ordered pair $(\mathcal{T},\mathcal{A})$ of a TBox $\mathcal{T}$ and an ABox $\mathcal{A}$. We extend it to an ordered tuple $KB=(\mathcal{T},\mathcal{A},\mathcal{P})$ by attaching a clause set $\mathcal{P}$. TBox- and ABox-statements and clausal forms are generally called knowledge base statements. A knowledge base KB is said to be ground if all the clausal forms in KB are ground.

We here interpret first-order formulas including concept and role names and variables. A DL-interpretation $\mathcal{I}$ is extended with $p^{\mathcal{I}} \subseteq (\Delta^{\mathcal{I}})^n$ for every n-ary predicate $p \in \mathbf{P}$ and $f^{\mathcal{I}} : (\Delta^{\mathcal{I}})^n \to \Delta^{\mathcal{I}}$ for every n-ary function $f \in \mathbf{F}$. A variable assignment is a mapping α from the set $\mathbf{V}$ of variables into the universe $\Delta^{\mathcal{I}}$. The variable assignment $\alpha[x/d]$ denotes $(\alpha - \{(x,\alpha(x))\}) \cup \{(x,d)\}$. An interpretation of a first-order language $\mathcal{L}$ with concept and role names (called an FL-interpretation) is an ordered pair $\mathcal{I}_\alpha = (\mathcal{I},\alpha)$ where $\mathcal{I}$ is a DL-interpretation extended with $p^{\mathcal{I}} \subseteq (\Delta^{\mathcal{I}})^n$ and $f^{\mathcal{I}} : (\Delta^{\mathcal{I}})^n \to \Delta^{\mathcal{I}}$ and α is a variable assignment. The interpretation $t^{\mathcal{I}_\alpha}$ of terms t is defined by: (i) $x^{\mathcal{I}_\alpha} = \alpha(x)$ and $a^{\mathcal{I}_\alpha} = a^{\mathcal{I}}$, and (ii) $(f(t_1,\ldots,t_n))^{\mathcal{I}_\alpha} = f^{\mathcal{I}}(t_1^{\mathcal{I}_\alpha},\ldots,t_n^{\mathcal{I}_\alpha})$. The satisfiability relation for knowledge base statements is given by the following definition.

Definition 2.4 *Let $\mathcal{I}_\alpha = (\mathcal{I},\alpha)$ be an FL-interpretation and E be a knowledge base statement. The satisfiability relation $\mathcal{I}_\alpha \models E$ is defined as follows:*

1. $\mathcal{I}_\alpha \models A \equiv C$ *iff* $A^{\mathcal{I}} = C^{\mathcal{I}}$
2. $\mathcal{I}_\alpha \models C(a)$ *iff* $a^{\mathcal{I}} \in C^{\mathcal{I}}$
3. $\mathcal{I}_\alpha \models R(a,b)$ *iff* $(a^{\mathcal{I}},b^{\mathcal{I}}) \in R^{\mathcal{I}}$
4. $\mathcal{I}_\alpha \models p(t_1,\ldots,t_n)$ *iff* $(t_1^{\mathcal{I}_\alpha},\ldots,t_n^{\mathcal{I}_\alpha}) \in p^{\mathcal{I}}$
5. $\mathcal{I}_\alpha \models l_1,\ldots,l_n \to l'_1,\ldots,l'_m$ *iff* $\mathcal{I}_\alpha \models \forall(\neg l_1 \vee \cdots \vee \neg l_n \vee l'_1 \vee \cdots \vee l'_m)$

The satisfiability $\mathcal{I}_\alpha \models \forall(\neg l_1 \vee \cdots \vee \neg l_n \vee l'_1 \vee \cdots \vee l'_m)$ is obtained in the usual semantics of first-order logic. An FL-interpretation $\mathcal{I}_\alpha$ satisfies a knowledge base KB (denoted $\mathcal{I}_\alpha \models KB$) if $\mathcal{I}_\alpha$ satisfies all the elements in $KB = (\mathcal{T},\mathcal{A},\mathcal{P})$. A knowledge base statement E (resp. a knowledge base KB) is satisfiable if, for some $\mathcal{I}_\alpha$, $\mathcal{I}_\alpha \models E$ (resp. $\mathcal{I}_\alpha \models KB$). Otherwise, E (resp. KB) is unsatisfiable. A knowledge base statement E is a consequence of KB (denoted $KB \models E$) if every model of KB is a model of E.

3 Hybrid Resolution

In this section, we develop a hybrid resolution system for extended knowledge bases $KB = (\mathcal{T},\mathcal{A},\mathcal{P})$. By transforming $\mathcal{ALC}$-concepts into a kind of clausal form of $\mathcal{ALC}$-concepts, this resolution system can be applied to both $\mathcal{ALC}$-concepts and first-order clauses in KB.

3.1 Clausal Concepts

We simplify the form of $\mathcal{ALC}$-concepts by the following operations. Let $KB = (\mathcal{T},\mathcal{A},\mathcal{P})$ be a knowledge base. First of all, we eliminate the symbols $\neg\neg, \sqcap, \exists$ from $\mathcal{ALC}$-concepts in the TBox $\mathcal{T}$ and ABox $\mathcal{A}$. Any concept is transformed into an equivalent concept by applying the rewrite rules [3]:

$$\neg\neg C \equiv C$$
$$C \sqcap D \equiv \neg(\neg C \sqcup \neg D)$$
$$\exists R.C \equiv \neg\forall R.\neg C$$

If a left-hand side form occurs in concepts of $\mathcal{T}$ or $\mathcal{A}$, then it is transformed to its right-hand side form. This transformation is applied to all the elements in $\mathcal{T}$, $\mathcal{A}$, and therefore $\mathcal{T}'$, $\mathcal{A}'$ without $\neg\neg, \sqcap, \exists$ are derived.

Notations. L denotes a concept name A or its negation $\neg A$, and Q represents $\forall R$ or $\neg\forall R$. $\overline{L}$ is $\neg A$ if $L = A$, or A if $L = \neg A$. $Q.L$ or L is called a *literal concept*, written as $Q^*.L$.

Next, concepts in the TBox $\mathcal{T}'$ and ABox $\mathcal{A}'$ derived above are further transformed by the two operations. First, if a concept E contains a concept of the form $\neg(C \sqcup D)$ or $\forall R.(C \sqcup D)$, then $A \equiv C \sqcup D$ (where A is a new concept name) is added to $\mathcal{T}'$, and $C \sqcup D$ (in E) is replaced with A. Secondly, if a concept E includes an expression of the form $Q_1.Q_2.C$, then $A \equiv Q_2.C$ is added to $\mathcal{T}'$, and $Q_2.C$ (in E) is transformed to A. Repeating these operations results in a concept of the form:

$$Q_1^*.L_1 \sqcup \cdots \sqcup Q_n^*.L_n.$$

We call this simplified concept a *clausal concept*. For example, $\neg\forall has\text{-}child.\neg Female \sqcup \forall has\text{-}child.Female \sqcup Human$ is a clausal concept. Then, we obtain $\mathcal{T}'',\mathcal{A}''$ by transforming all the elements in $\mathcal{T}',\mathcal{A}'$ to equivalent clausal concepts.

Lemma 3.1 *Let $\mathcal{I}$ be a DL-interpretation. Let C' be a clausal concept transformed from a concept C and let $A_1 \equiv C_1,\ldots,A_n \equiv C_n (n \geq 0)$ be the TBox-statements added by the transformation where each A_i is a new concept name. Then, there exists a DL-interpretation $\mathcal{I}'$ that is an extension of $\mathcal{I}$ to interpret $A_1,\ldots,A_n$, and $(C')^{\mathcal{I}'} = C^{\mathcal{I}}$ and $A_1^{\mathcal{I}'} = C_1^{\mathcal{I}'},\ldots,A_n^{\mathcal{I}'} = C_n^{\mathcal{I}'}$.*

By this lemma, if a knowledge base $KB = (\mathcal{T},\mathcal{A},\mathcal{P})$ is satisfiable, then also the equivalent knowledge base $KB' = (\mathcal{T}'',\mathcal{A}'',\mathcal{P})$ obtained by transforming $\mathcal{T},\mathcal{A}$ is satisfiable.

3.2 Hybrid Resolution System

Our hybrid resolution system contains three kinds of inference rules: resolution principle, taxonomic rules and assertional rules. We construct resolution rules that can be applied to TBox- and ABox-statements of clausal concepts, first-order clauses and their composed expressions.

[3] In [1], $\neg\neg, \sqcup, \exists$ are removed from concepts. The transformation in this paper is adjusted to obtain compatible expressions with clausal forms.

In resolution steps, we express clausal forms as sets of literals, and ABox-statements of clausal concepts as sets of assertions of literal concepts. Given a clausal form $l_1, \ldots, l_n \rightarrow l'_1, \ldots, l'_m$, the set of the literals (called a *clause*) is $\{\neg l_1, \ldots, \neg l_n, l'_1, \ldots, l'_m\}$. For an ABox-statement $Q^*_1.L_1 \sqcup \cdots \sqcup Q^*_n.L_n(a)$ of a clausal concept, we have the set $\{Q^*_1.L_1(a), \cdots, Q^*_n.L_n(a)\}$ of assertions of the literal concepts. Note that applying hybrid resolution rules to sets of literals and sets of assertions of literal concepts leads to an expression as composed of these sets. Let P^+_i be a concept of the form $\forall R.A$, $\forall R.\neg A$ or A, a role name R or an n-ary predicate p, and let $\vec{t}$ be a sequence of terms $t_1, \ldots, t_k$. The compositional expression is a set of extended literals, called an *extended clause*:

$$\{P^+_1(\vec{t_1}), \ldots, P^+_n(\vec{t_n}), \neg P^+_{n+1}(\vec{t}_{n+1}), \ldots, \neg P^+_m(\vec{t}_m)\}.$$

This is allowed to include an expressive assertion $Q^*.L(t)$ of a literal concept $Q^*.L$ where t is a first-order term. Since the extended clauses exceed the expressivity of ordinary clausal forms, an additional definition of the satisfiability is needed. The satisfiability of extended clauses $\Phi = \{P^+_1(\vec{t_1}), \ldots, P^+_n(\vec{t_n}), \neg P^+_{n+1}(\vec{t}_{n+1}), \ldots, P^+_m(\vec{t}_m)\}$ is defined as follows:

1. $\mathcal{I}_\alpha \models P^+_i(\vec{t_i})$ *iff* $(\vec{t_i})^{\mathcal{I}_\alpha} \in (P^+_i)^{\mathcal{I}}$.
2. $\mathcal{I}_\alpha \models \neg P^+_i(\vec{t_i})$ *iff* $\mathcal{I}_\alpha \not\models P^+_i(\vec{t_i})$.
3. $\mathcal{I}_\alpha \models \Phi$ with $Var(\Phi) = \{x_1, \ldots, x_k\}$ *iff* for all d_1, $\ldots$, $d_k \in \Delta^{\mathcal{I}}$, $\{E \in \Phi \mid \mathcal{I}_{\alpha'} \models E\} \neq \emptyset$ with $\alpha' = \alpha[x_1/d_1] \cdots [x_k/d_k]$.

We proceed to the definition of hybrid resolution rules. The first rule is an extension of the resolution principle to extended clauses.

Definition 3.1 (Resolution principle) *Let Φ_1, Φ_2 be extended clauses. The resolution principle for knowledge bases is given as follows:*

$$\frac{\Phi_1 \cup \{\neg P^+(\vec{t})\} \quad \{P^+(\vec{t'})\} \cup \Phi_2}{(\Phi_1 \cup \Phi_2)\theta} \ (res)$$

where $\theta = mgu(\vec{t}, \vec{t'})$.

Notations. Let Ψ be a clausal concept $Q^*_1.L_1 \sqcup \cdots \sqcup Q^*_n.L_n$. $\Psi(t)$ expresses the sequence of assertions $Q^*_1.L_1(t), \ldots, Q^*_n.L_n(t)$ of the literal concepts in Ψ. We write L_A for the concept name A or its negation $\neg A$. $\delta(E)$ denotes E_0 if $E = \neg\neg E_0$, or E otherwise. The function $N(E)$ is 0 if the number of negation symbols ($\neg$) in E is even, or 1 if it is odd. Moreover, we have the following convenient notation:

$$\{\neg R_Q(t, t')\} = \begin{cases} \{\neg R(t, t')\} & if\ Q = \forall R \\ \emptyset & otherwise \end{cases}$$

Next, we introduce taxonomic rules with regard to TBox-statements.

Definition 3.2 (Taxonomic rules) *Let Φ be an extended clause and Ψ be a clausal concept. The taxonomic rules for knowledge bases are given as follows:*

$$\frac{\Phi \cup \{\neg A(t)\} \quad A \equiv Q^*.L \sqcup \Psi}{\Phi \cup \{\delta(\neg Q^*.L)(t)\}} \ (T1)$$

$$\frac{\Phi \cup \{A(t)\} \quad A \equiv Q^*.L \sqcup \Psi}{\Phi \cup \{Q^*.L(t), \Psi(t)\}} \ (T2)$$

$$\frac{\Phi \cup \{Q_1.L_A(t)\} \quad A \equiv Q^*_2.L_2 \sqcup \Psi}{\Phi \cup \{\delta(\neg Q^*_2.L_2)(t')\} \cup \{\neg R_{Q_1}(t, t')\}} \ (T3)$$

where $N(Q_1.L_A) = 1$, and $t' = x$ (new variable) if $Q_1 = \forall R$ and $t' = c$ (new constant) otherwise.

$$\frac{\Phi \cup \{Q_1.L_A(t)\} \quad A \equiv Q^*_2.L_2 \sqcup \Psi}{\Phi \cup \{Q^*_2.L_2(t'), \Psi(t')\} \cup \{\neg R_{Q_1}(t, t')\}} \ (T4)$$

where $N(Q_1.L_A) = 0$, and $t' = x$ (new variable) if $Q_1 = \forall R$ and $t' = c$ (new constant) otherwise.

In the hybrid resolution system, assertions of literal concepts in extended clauses are refuted by the following assertional rules .

Definition 3.3 (Assertional rules) *Let Φ_1, Φ_2 be extended clauses. The assertional rules for knowledge bases are given as follows:*

$$\frac{\Phi_1 \cup \{\forall R.L_A(t_1)\} \quad \{\overline{L_A}(t_2)\} \cup \Phi_2}{\Phi_1 \cup \Phi_2 \cup \{\neg R(t_1, t_2)\}} \ (A1)$$

$$\frac{\Phi_1 \cup \{\neg\forall R.L_A(t)\} \quad \{L_A(x)\} \cup \Phi_2}{(\Phi_1 \cup \Phi_2)\theta} \ (A2)$$

where c is a new constant and $\theta = \{x/c\}$.

$$\frac{\Phi_1 \cup \{\forall R_1.L_A(t_1)\} \quad \{Q_2.L'_A(t_2)\} \cup \Phi_2}{\Phi_1 \cup \Phi_2 \cup \{\neg R_1(t_1, t)\} \cup \{\neg R_{Q_2}(t_2, t)\}} \ (A3)$$

where $N(\forall R_1.L_A) \neq N(Q_2.L'_A)$, and $t = x$ (new variable) if $Q_2 = \forall R$ and $t = c$ (new constant) otherwise.

$$\frac{\Phi_1 \cup \{\neg\forall R.L(t_1)\} \quad \{\neg R(t_2, t)\} \cup \Phi_2}{(\Phi_1 \cup \Phi_2)\theta} \ (A4)$$

where c is a new constant or the constant introduced by an application of $(T3)$, $(T4)$, $(A2)$ or $(A3)$ with its premise $\Phi_1 \cup \{\neg\forall R.L(t_1)\}$, and θ is a mgu of (t_1, c) and (t_2, t).

In the form $\dfrac{E_1 \quad E_2}{E}$ of hybrid resolution rules, E_1, E_2 are called the premises, and E the conclusion. We assume that the premises E_1, E_2 do not have the same variables, i.e., $Var(E_1) \cap Var(E_2) = \emptyset$. Compared with the DL-resolution system [1], our hybrid resolution system is extended to enhance resolution for (i) assertions of clausal concepts (in $(A1), \ldots, (A4)$ and (res)), (ii) TBox- and ABox-statements (in $(T1)$, $\ldots$,$(T4)$) and (iii) extended clauses represented by clausal concepts, n-ary predicates and first-order terms (in (res)), and with unification for

first-order terms. The resolution principle (res) contains resolution for positive and negative literals and can be applied to ABox-statements of clausal concepts and role names. In [1], concept definitions in the TBox are unfolded in the ABox beforehand. In our case, the taxonomic rules $(T1)$, ... ,$(T4)$ generate inferences from concept definitions $A \equiv C$ in the TBox and assertions $L_A(t)$ or $Q.L_A(t)$ of the defined concept names A in extended clauses. The assertional rules $(A1), \ldots, (A3)$ eliminate an inconsistent pair $(A, \neg A)$ of a concept name A and its negation $\neg A$ in the premises. The assertional rule $(A4)$ is a resolution rule for a negative assertion $\neg R(t_1, t_2)$ of a role name R and assertions of the form $\neg\forall R.L(t_1)$.

Definition 3.4 (Resolution) *Let KB be a knowledge base and E be a knowledge base statement. The derivability relation $KB \vdash E$ is defined by the following:*

1. *If $E \in KB$, then $KB \vdash E$.*
2. *If $KB \vdash E_1$ and $KB \vdash E_2$ where E_1, E_2 are premises and E is the conclusion in a hybrid resolution rule, then $KB \vdash E$.*

A derivation of E from KB is called a refutation if $E = \emptyset$. A knowledge base KB is refutable if we have a refutation $KB \vdash \emptyset$.

3.3 An Example of Refutation

We show an example of refutation by applying the hybrid resolution system. Consider the first-order language with

$\mathbf{C} = \{\ Murderer, Human, Male, Female\ \}$
$\mathbf{R} = \{\ killed\ \}$
$\mathbf{P} = \mathbf{C} \cup \mathbf{R} \cup \{\ acted, died, after\ \}$
$\mathbf{F} = \emptyset$
$\mathbf{I} = \{\ John, Mary, e_1, e_2\ \}$

and the two knowledge bases $KB_1 = (\mathcal{T}_1, \mathcal{A}_1, \mathcal{P}_1)$ and $KB_2 = (\mathcal{T}_1, \mathcal{A}_2, \mathcal{P}_1)$ with

TBox $\mathcal{T}_1 = \{\ Murderer \equiv \exists killed.Human \sqcap Human,$
$Human \equiv Male \sqcup Female\ \}$
ABox $\mathcal{A}_1 = \{\ Male(John), Female(Mary)\ \}$
ABox $\mathcal{A}_2 = \{\ Human(John), Human(Mary)\ \}$
Clause set $\mathcal{P}_1 = \{ \rightarrow acted(John, Mary, e_1),$
$\rightarrow died(Mary, e_2),$
$\rightarrow after(e_2, e_1),$
$acted(x, y, z_1), died(y, z_2), after(z_2, z_1),$
$Human(x), Human(y) \rightarrow killed(x, y)\ \}$

In the TBox $\mathcal{T}_1$, the concept name $Murderer$ is defined by the concept $\exists killed.\ Human \sqcap Human$, and the concept name $Human$ is defined by the disjunctive concept $Male \sqcup Female$. The ABox $\mathcal{A}_1$ asserts that John is male and Mary is female, and the ABox $\mathcal{A}_2$ expresses the fact that John and Mary are humans. In the clause set $\mathcal{P}_1$, three facts and a rule are described. Notice that this rule is not expressible in the logic programming languages combining with description logics [11, 5], since the role name $killed$ occurs in the head of the rule.

We consider reasoning to answer the following query in the knowledge bases KB_1, KB_2.

$?\text{-}Murderer(x).$

which means "is there a murderer x?" Before applying hybrid resolution rules, we remove the connectives $\sqcap, \exists, \neg\neg$ from the TBox $\mathcal{T}_1$. Consequently, the following TBox is obtained.

TBox $\mathcal{T}_1' = \{\ Human \equiv Male \sqcup Female,$
$Murderer \equiv \neg(\forall killed.\neg Human \sqcup \neg Human)\ \}$

Moreover, the concepts in the TBox $\mathcal{T}_1'$ are transformed to equivalent clausal concepts as follows:

TBox $\mathcal{T}_1'' = \{\ Murderer \equiv \neg A_1,\ A_1 \equiv A_2 \sqcup A_3,$
$A_2 \equiv \forall killed.\neg Human,\ A_3 \equiv \neg Human,$
$Human \equiv Male \sqcup Female\ \}$

The ABox-statements in $\mathcal{A}_1$,$\mathcal{A}_2$ and the clausal forms in $\mathcal{P}_1$ are transformed into sets of assertions of literal concepts and sets of literals as follows:

ABox $\mathcal{A}_1' = \{\ \{Male(John)\}, \{Female(Mary)\}\ \}$
ABox $\mathcal{A}_2' = \{\ \{Human(John)\}, \{Human(Mary)\}\ \}$
Clause set $\mathcal{P}_1' = \{\ \{acted(John, Mary, e_1)\},$
$\{died(Mary, e_2)\}, \{after(e_2, e_1)\},$
$\{\neg acted(x, y, z_1), \neg died(y, z_2), \neg after(z_2, z_1),$
$\neg Human(x), \neg Human(y), killed(x, y)\}\ \}$

The answer to the query $?\text{-}Murderer(x)$ in the clause set $\mathcal{P}_1'$ is decided by refutation for $\mathcal{P}_1' \cup \{\mathcal{G}\}$ with $\mathcal{G} = \{\neg Murderer(x)\}$. In Figure 1, (3) and (4) show derivation processes for $KB_1' \cup \{\mathcal{G}\} = (\mathcal{T}_1'', \mathcal{A}_1', \mathcal{P}_1' \cup \{\mathcal{G}\})$ and $KB_2' \cup \{\mathcal{G}\} = (\mathcal{T}_1'', \mathcal{A}_2', \mathcal{P}_1' \cup \{\mathcal{G}\})$. These determine whether there exists a term t and whether $Murderer(t)$ is valid in the knowledge bases $KB_1' = (\mathcal{T}_1'', \mathcal{A}_1', \mathcal{P}_1')$ and $KB_2' = (\mathcal{T}_1'', \mathcal{A}_2', \mathcal{P}_1')$. In both cases, we can conclude that $Murderer(John)$ is true since the empty clause is derived with the substitution $\theta = \{x/John\}$, i.e., $KB_1' \cup \{\mathcal{G}\}$ and $KB_2' \cup \{\mathcal{G}\}$ are refutable.

Theorem 3.1 (Completeness of resolution) *Let KB be a knowledge base. KB is unsatisfiable if and only if KB is refutable ($KB \vdash \emptyset$).*

Due to the limitations of space, the proof of the theorem has been omitted. The completeness of resolution can be proved by the construction of a tree model for each knowledge base [6], the completeness of unrestricted resolution and ground resolution, and the lifting lemma [4, 9].

(1)

$$\dfrac{\dfrac{\{aft(e_2,e_1)\}\quad \dfrac{\{act(x,M,e_1)\}\quad \dfrac{\{die(M,e_2)\}\quad \{\neg act(x,y,z_1),\neg die(y,z_2),\neg aft(z_2,z_1),\neg Hum(x),\neg Hum(y),kil(x,y)\}}{\{\neg act(x,M,e_1),\neg aft(z_2,e_1),\neg Hum(x),\neg Hum(M),kil(x,M)\}}\,(res)}{\{kil(J,M),\neg aft(e_2,e_1),\neg Hum(J),\neg Hum(M)\}}\,(res)}{\{kil(J,M),\neg Hum(J),\neg Hum(M)\}}\,(res)$$

$$\dfrac{\dfrac{\{Fem(M)\}\quad \dfrac{\{Mal(J)\}\quad \dfrac{Hum\equiv Mal\sqcup Fem\quad \dfrac{Hum\equiv Mal\sqcup Fem\quad \{kil(J,M),\neg Mal(J),\neg Hum(M)\}}{\{kil(J,M),\neg Mal(J),\neg Hum(M)\}}\,(T1)}{\{kil(J,M),\neg Mal(J),\neg Fem(M)\}}\,(T1)}{\{kil(J,M),\neg Fem(M)\}}\,(res)}{\{kil(J,M)\}}\,(res)$$

(2)

$$\dfrac{\dfrac{\dfrac{\dfrac{\{\neg Mur(x)\}\quad Mur\equiv\neg A_1}{\{A_1(x)\}}\,(T1)\quad A_1\equiv A_2\sqcup A_3}{\{A_2(x),A_3(x)\}}\,(T2)\quad A_2\equiv\forall kil.\neg Hum}{\{\forall kil.\neg Hum(x),A_3(x)\}}\,(T2)\quad A_3\equiv\neg Hum}{\{\forall kil.\neg Hum(x),\neg Hum(x)\}}\,(T2)$$

(3)

$$\dfrac{\dfrac{\dfrac{\vdots\,(2)}{\{\forall kil.\neg Hum(x),\neg Hum(x)\}}\quad Hum\equiv Mal\sqcup Fem}{\{\neg Fem(v),\neg kil(x,v),\neg Hum(x)\}}\,(T2)\quad \dfrac{\vdots\,(1)}{\{kil(J,M)\}}}{\{\neg Fem(M),\neg Hum(J)\}}\,(T1)\quad \{Fem(M)\}}{\{\neg Hum(J)\}}\,(res)$$

$$\vdots$$

$$\emptyset$$

(4)

$$\dfrac{\dfrac{\dfrac{\vdots\,(2)}{\{\forall kil.\neg Hum(x),\neg Hum(x)\}}\quad \{Hum(M)\}}{\{\neg kil(x,M),\neg Hum(x)\}}\,(A1)\quad \dfrac{\vdots\,(1)}{\{kil(J,M)\}}}{\{\neg Hum(J)\}}\,(res)\quad \{Hum(J)\}}{\emptyset}\,(res)$$

Figure 1. Refutation from a knowledge base

4 Conclusion

We have presented a rule-based reasoning system as an extension of the DL-resolution system [1] where DL-knowledge bases and first-order clause sets are combined. From the viewpoint of knowledge representation, the extended knowledge bases can treat practical and general data as follows:

- Facts and ABox-statements represent assertional knowledge in a context (or a concrete situation).
- Rules are taken as general knowledge in a particular application domain.
- TBox-statements express terminological knowledge commonly employed in many application domains.

We have designed a *proper* resolution method for capturing the two kinds of logical form in the DL $\mathcal{ALC}$ and logic programming. A resolution rule is called proper if its two premises are composed into one conclusion by deleting an inconsistent pair $(E, \neg E)$ of literals. Hence, we have obtained proper hybrid resolution rules, whereas some of the inference rules in the DL-resolution system [1] were not proper. Although we have not discussed the computation of our reasoning system, the proper resolution method can be expected to provide an effective deduction procedure. For example, consider the inconsistency of the assertions: $\neg\forall R_1.\neg A_1(t_1)$, $\forall R_1.\neg A_1(t_1)$, $\forall R_2.A_4(t_2)$, $\neg\forall R_2.\neg A_2(t_2)\vee\neg\forall R_2.\neg A_3(t_2)$. As shown in Figure 2, the inconsistency can be derived in two resolution steps because proper resolution rules are applied only to clauses including an inconsistent pair of literals. However, tableau-like reasoning (as the standard DL reasoning method) needs nine steps in the worst case since redundant steps are derived. Alternatively, Kaneiwa and Tojo [10] proposed a resolution system with complex sort expressions, but its language was not able to represent and

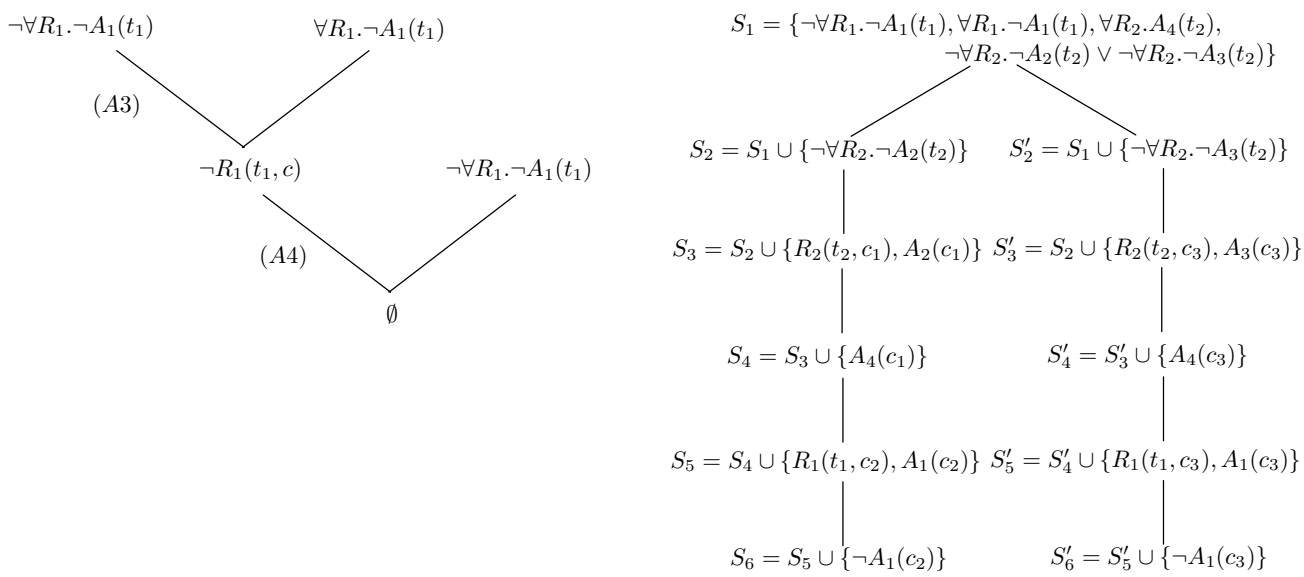

Figure 2. Comparison between resolution and tableau-like reasoning

reason with respect to DL-concepts.

This study leads to a further extension of rule-based reasoning with terminology, by using other approaches in logic programming and automated reasoning (e.g. typed logic programming, nonmonotonic reasoning, etc.). Furthermore, we now plan to formalize the combination of first-order clauses and other description logics, such as sublanguages of $\mathcal{ALC}$ (e.g. $\mathcal{AL}$, $\mathcal{ALU}$) and superlanguages of $\mathcal{ALC}$ (e.g. $\mathcal{ALCN}$ [8], $\mathcal{ALCQI}$).

References

[1] C. Areces, H. de Nivelle, and M. de Rijke. Resolution in modal, description and hybrid logics. *Journal of Logic and Computation*, 11(5):717–736, 2001.

[2] F. Baader and U. Sattler. An overview of tableau algorithms for description logics. *Studia Logica*, 69:5–40, 2001.

[3] R. J. Brachman, R. E. Fikes, and H. J. Levesque. KRYPTON: A functional approach to knowledge representation. *Computer*, 16(10):67–73.

[4] K. Doets. *From Logic to Logic Programming*. The MIT Press, 1994.

[5] F. M. Donini, M. Lenzerini, D. Nardi, and A. Schaerf. AL-LOG: Integrating datalog and description logic. *Journal of Intelligent Information Systems*, 10(3):227–252, 1998.

[6] P. Enjalbert and L. Farinas del Cerro. Modal resolution in clausal form. *Theoretical Computer Science*, 65(1):1–33, 1989.

[7] D. Gabbay and U. Reyle. Labelled resolution for classical and non-classical logics. *Studia Logica*, 59(2):179–216, 1997.

[8] B. Hollunder. Hybrid inferences in KL-ONE-based knowledge representation systems. In *Proceedings of GWAI–90, Fourteenth German Workshop on Artificial Intelligence*, pages 38–47, 1990.

[9] K. Kaneiwa. The completeness of logic programming with sort predicates. *Systems and Computers in Japan*, 35(1), 2004 (to appear).

[10] K. Kaneiwa and S. Tojo. An order-sorted resolution with implicitly negative sorts. In *Proceedings of the 2001 Int. Conf. on Logic Programming*, pages 300–314. Springer-Verlag, 2001. LNCS 2237.

[11] A. A. Levy and M.-C. Rousset. CARIN: A representation language combining Horn rules and description logics. *Artificial Intelligence*, 104(1–2):165–209, 1998.

[12] J. W. Lloyd. *Foundations of Logic Programming*. Springer-Verlag, 1987.

[13] J. A. Robinson. A machine-oriented logic based on the resolution principle. *Journal of the ACM (JACM)*, 12(1):23–41, 1965.

[14] M. Schmidt-Schauss and G. Smolka. Attributive concept descriptions with complements. *Artificial Intelligence*, 48:1–26, 1991.