PUC

ISSN 0103-9741

Monografias em Ciência da Computação
n° 25/07

A Sequent Calculus for *ALC*

Alexandre Rademaker
Fernando Naufel do Amaral
Edward Hermann Haeusler

Departamento de Informática

PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO DE JANEIRO
RUA MARQUÊS DE SÃO VICENTE, 225 - CEP 22453-900
RIO DE JANEIRO - BRASIL

Monografias em Ciência da Computação, No. 25/07 ISSN: 0103-9741
Editor: Prof. Carlos José Pereira de Lucena Setembro, 2007

A Sequent Calculus for $\mathcal{ALC}$

Alexandre Rademaker and Fernando Naufel do Amaral and Edward Hermann Haeusler

{arademaker,hermann}@inf.puc-rio.br, fnaufel@ic.uff.br

Abstract. Description Logics is a family of formalisms used to represent knowledge of a domain. In contrast with others knowledge representation systems, Description Logics are equipped with a formal, logic-based semantics. Knowledge representation systems based on description logics provide various inference capabilities that deduce implicit knowledge from the explicitly represented knowledge.

We present a sequent calculus for $\mathcal{ALC}$, a basic Description Logic. The first motivation for developing such system is the extraction of computational content of $\mathcal{ALC}$ proofs. The present calculus is an intermediate step towards a Natural Deduction System for $\mathcal{ALC}$.

Keywords:
Description logics, Sequent Calculus, Proof Theory.

Resumo.

Lógicas de descrição são uma família de formalismos usados para representação de conhecimento de um domínio. Em constrate com outros sistemas de representação do conhecimento, lógicas de descrição são equipadas com uma semântica formal. Sistemas de representação do conhecimento baseados em lógicas de descrição oferecem várias capacidades de inferência de conhecimentos implícitos a partir de conhecimentos explícitos.

Apresentamos um cálculo de sequents para $\mathcal{ALC}$, uma lógica de descrição básica. A motivação principal para desenvolvimento deste sistema dedutivo é a extração de conteúdo computacional a partir de provas (deduções). O presente cálculo é um passo intermediário no desenvolvimento de um sistema em dedução natural para $\mathcal{ALC}$.

Palavras-chave:
Lógicas de descrição, Cálculo de Sequents, Teoria da Prova.

In charge of publications:
Rosane Teles Lins Castilho
Assessoria de Biblioteca, Documentação e Informação
PUC-Rio Departamento de Informática
Rua Marquês de São Vicente, 225 - Gávea
22451-900 Rio de Janeiro RJ Brasil
Tel. +55 21 3527-1516 Fax: +55 21 3527-1530
E-mail: bib-di@inf.puc-rio.br
Web site: http://bib-di.inf.puc-rio.br/techreports/

1 Introduction

Description Logics is a family of formalisms used to represent knowledge of a domain. In contrast with others knowledge representation systems, Description Logics are equipped with a formal, logic-based semantics. Knowledge representation systems based on description logics provide various inference capabilities that deduce implicit knowledge from the explicitly represented knowledge [1].

The use of Description Logics by regular users, that is, non-technical users, would be wider if the computed inferences could be presented as a natural language text – or any other presentation format at the domain's specification level of abstraction – without requiring any knowledge on logic to be understandable [11].

We present a sequent calculus for $\mathcal{ALC}$ [1], a basic Description Logic. The first motivation for developing such system is the extraction of computational content of $\mathcal{ALC}$ proofs. More precisely, this system was developed to allow the use of natural language to rendering of a Natural Deduction System proof. So that, the present calculus for $\mathcal{ALC}$ is an intermediate step towards a Natural Deduction System for $\mathcal{ALC}$ [3]. A natural language rendering of a Natural Deduction System is worthwhile in a context like proof of conformance in security standards [4].

Our Sequent Calculus, compared with other approaches like Tableaux [16] and the Sequent Calculus for $\mathcal{ALC}$ [5, 11, 2] based on this very Tableaux, does not use individual variables (first-order ones) at all. The main mechanism in our system is based on labeled formulas. The labeling of formulas is among one of the most successful artifacts for keeping control of the context in the many existent quantification in Logical system and modalities. For a detailed reading on this approach, we point out [14, 8, 12, 13, 6].

This paper is structured as follows. Section 2 presents the basic notions of Description Logics. Section 3 presents our Sequent Calculus for $\mathcal{ALC}$. Sections 4 and 5 present, respectively, the proofs of soundness and completeness of the Sequent Calculus presented. In Section 6 we present an example of proof for a security standards conformance using our calculus. Finally, in Section 7 we point out further works and present some conclusions.

2 Description Logics

Description Logics is a family of knowledge representation formalisms used to represent knowledge of a domain, usually called "world". For that, it first defines the relevant concepts of the domain – "terminology" – and then, using these concepts, specify properties of objects and individuals of that domain. Comparing to its predecessors formalisms, Description Logics are equipped with a formal, logic-based semantics. Description Logics differ each other from the constructors they provide. Concept constructors are used to build more complex descriptions of concepts from *atomic concepts* and *atomic roles.*

$\mathcal{ALC}$ is a basic Description Logics [1] and its syntax of concept descriptions is as following:

$$\phi_c ::= \bot \mid A \mid \neg\phi_c \mid \phi_c \sqcap \phi_c \mid \phi_c \sqcup \phi_c \mid \exists R.\phi_c \mid \forall R.\phi_c$$

where A stands for atomic concepts, α for concepts, R for atomic roles.

[1] Attributive Language with Complements

As usually, the semantics of concept descriptions is defined in terms of an *interpretation* $\mathcal{I} = (\Delta^\mathcal{I}, .^\mathcal{I})$. The domain $\Delta^\mathcal{I}$ of $.^\mathcal{I}$ is a non-empty set of individuals and the interpretation function $.^\mathcal{I}$ maps each atomic concept A to a set $A^\mathcal{I} \subseteq \Delta^\mathcal{I}$ and for each atomic role a binary relation $r^\mathcal{I} \subseteq \Delta^\mathcal{I} \times \Delta^\mathcal{I}$. The interpretation function $.^\mathcal{I}$ is extended to concept descriptions inductive as follows:

$$\begin{array}{rcl}
\top^\mathcal{I} & = & \Delta^\mathcal{I} \\
\bot^\mathcal{I} & = & \emptyset \\
(\neg C)^\mathcal{I} & = & \Delta^\mathcal{I} \setminus C^\mathcal{I} \\
(C \sqcap D)^\mathcal{I} & = & C^\mathcal{I} \cap D^\mathcal{I} \\
(C \sqcup D)^\mathcal{I} & = & C^\mathcal{I} \cup D^\mathcal{I} \\
(\exists R.C)^\mathcal{I} & = & \{a \in \Delta^\mathcal{I} \mid \exists b.(a,b) \in R^\mathcal{I} \wedge b \in C^\mathcal{I}\} \\
(\forall R.C)^\mathcal{I} & = & \{a \in \Delta^\mathcal{I} \mid \forall b.(a,b) \in R^\mathcal{I} \rightarrow b \in C^\mathcal{I}\}
\end{array}$$

Knowledge representation systems based on description logics provide various inference capabilities that deduce implicit knowledge from the explicitly represented knowledge. One of the most important inference services of DL systems is computing the subsumption hierarchy of a given finite set of concept descriptions.

Definition 1. *The concept description D subsumes the concept description C, written $C \sqsubseteq D$, if and only if $C^\mathcal{I} \sqsubseteq D^\mathcal{I}$ for all interpretations $\mathcal{I}$.*

Definition 2. *C is satisfiable if and only if there exists an interpretation $\mathcal{I}$ such that $C^\mathcal{I} \neq \emptyset$.*

Definition 3. *C and D are equivalent, written $C \equiv D$, if and only if $C \sqsubseteq D$ and $D \sqsubseteq C$.*

We used to call $C \sqsubseteq D$ and $C \equiv D$ *terminological axioms*. Axioms of the first kind are called *inclusions*, while axioms of the second kind are called *equalities*. If and interpretation safisfies an axiom (or a set of axioms), then we say that is a *model* of this axiom (or set of axioms).

An equality axiom whose left-hand side is an atomic concept is a *definition*. Definitions are used to introduce *names* for complex descriptions. For instance, the axiom

$$Mother \equiv Woman \sqcap \exists hasChild.Person$$

associates to the description on the right-hand side the name $Mother$.

A finite set of definitions $\mathcal{T}$ where no symbolic name is defined more than once is called a *terminology* or *TBox*. In other words, for every atomic concept A there is at most one axiom in $\mathcal{T}$ whose left-hand side is A. Given a $\mathcal{T}$, we divide the atomic concepts occurring in it into two sets, the *name symbols* $\mathcal{N}_\mathcal{T}$ that occur on the left-hand side of some axiom and the *base symbols* $\mathcal{B}_\mathcal{T}$ that occur only on the right-hand side of axioms. Name symbols are called *defined* concepts and base symbols *primitive* concepts. The terminology should *defines* the name symbols in terms of the base symbols.

Of course, with the definitions of the last paragraph, we must also extend the definitions of *interpretations*. A *base interpretation* $.^\mathcal{I}$ for $\mathcal{T}$ is an interpretation just for the base symbols. An interpretation that interprets also the name symbols is called an *extension* of $.^\mathcal{I}$. Since our purpose here is not a complete review of description logics, we will not go into more details. There are a lot of concerns about such extensions, cyclic definitions in a *TBox* and assertions about individuals in a knowledge base – *world description* or *ABox* – we cite [1] for a complete reference.

A knowledge base – TBox and ABox – equipped with its semantics is equivalent to a set of axioms in first-order predicate logic. Thus, as said before, like any other set of axioms, it contains implicit knowledge that *logical inferences* can make explicit.

When we are constructing a TBox $\mathcal{T}$, by defining new concepts, possibly in terms of others that have been defined before, it is important to enforce the consistence of the TBox. That is, it is important that new concepts make sense and do not be contradictory with old ones. Formally, a concept makes sense if there is some interpretation that satisfies the axioms of $\mathcal{T}$ such that the concept denotes a nonempty set in that interpretation.

Definition 4 (Satisfiability)**.** *A concept C is* satisfiable *with respect to $\mathcal{T}$ if there exist a model $\cdot^{\mathcal{I}}$ of $\mathcal{T}$ such that $C^{\mathcal{I}}$ is nonempty. In this case, $\cdot^{\mathcal{I}}$ is a* model *of C.*

While modeling a domain of knowledge into a TBox other important inference service is necessary. For instance, it is usually interesting to organize the concepts of a TBox into a taxonomy. That is, it is important to know whether some concept is more general than another one: the *subsumption problem*. Furthermore, other interesting relationships between concepts is the *equivalence*.

Definition 5 (Subsumption)**.** *A concept C is* subsumed *by a concept D with respect to $\mathcal{T}$ if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ for every model $\cdot^{\mathcal{I}}$ of $\mathcal{T}$. In this case we write $C \sqsubseteq_{\mathcal{T}} D$ or $\mathcal{T} \models C \sqsubseteq D$.*

Definition 6 (Equivalence)**.** *Two concepts C and D are* equivalent *with respect to $\mathcal{T}$ if $C^{\mathcal{I}} = D^{\mathcal{I}}$ for every model $\cdot^{\mathcal{I}}$ of $\mathcal{T}$. In this case we write $C \equiv_{\mathcal{T}} D$ or $\mathcal{T} \models C \equiv D$.*

If the TBox is clear from the context or empty we can drop the qualification and simply write $\models C \sqsubseteq D$ if C is subsumed by D, and $\models C \equiv D$ if they are equivalent.

Since it is not our main concern in this paper, we will not go into more details about the equivalence and reductions between reasoning problems in Description Logics. Basically, the different kinds of reasoning can be reduced to a main inference problem, named the consistency check for ABox [1].

There exist two main algorithms to reasoning in Description Logics: *structural subsumption algorithms* and *tableaux-based algorithms* [1]. One of the differences between them relies on the logical languages that each one can handle.

For the description logic $\mathcal{ALN}$ [2] and its subsets, that is, the Description Logic not allowing full negation ($\neg C$), disjunction ($C \sqcup D$) nor full existential ($\exists R.C$), the subsumption of concepts can be computed by structural subsumption algorithms. The idea of these algorithms is compare the syntactic structure of concept descriptions. These algorithms are usually very efficient, polynomial time complexity [10] indeed.

For $\mathcal{ALC}$ and its extensions, the satisfiability of concepts and the subsumption of concept usually can be computed by *tableau-based algorithms* which are sound and complete for these problems [1]. The first tableau-based algorithm for satisfiability of $\mathcal{ALC}$-concepts was presented by [16]. As we said before, some reasoning problems in Description Logics can be reduced to others, in special, the problem to test the subsumption of concepts is reduced to the problem of test the (un)satisfiability of a concept description. These

[2] The letter $\mathcal{N}$ means *number restrictions* and provides the language with the ability to describe at-least restriction ($\geq nR$) and at-most restriction ($\leq nR$).

algorithms use the fact that $C \sqsubseteq D$ if and only if $C \sqcap \neg D$ is unsatisfiable [1]. Regarding the complexity, the tableau-based satisfiability algorithm for $\mathcal{ALC}$ is a PSPACE-hard problem [16].

As a final remark, it is well-known after [9] and [15] that any Classical Propositional Logic axiomatization containing the axiom $\forall R.(C \sqcap D) \equiv \forall R.C \sqcap \forall R.D$ and the necessitation rule is sound and complete for $\mathcal{ALC}$. As usual, $\exists R.C$ can be taken as a shorthand for $\neg\forall R.\neg C$, as well as $\forall R.C$ as a shorthand for $\neg\exists R.\neg C$. Taking $\exists R.C$ as the defined concept, the axiom changes to $\exists R.(C \sqcup D) \equiv \exists R.C \sqcup \exists R.D$.

3 A Sequent Calculus for $\mathcal{ALC}$

The Sequent Calculus for $\mathcal{ALC}$ that it is shown in Figure 1 considers the extension of the language ϕ_c defined in the previous section for labeled concepts. The labels are two lists of (possibly skolemized) role symbols. Its syntax is as following:

$$L ::= R, L \mid R(L), L \mid \emptyset$$
$$\phi_{lc} ::= {}^{L}\phi_c{}^{L}$$

where R stands for roles and L for list of roles. That is, the fragment $\{\bot, \sqcap, \sqcup, \neg, \exists, \forall\}$ of $\mathcal{ALC}$ for labeled concepts.

Each labeled $\mathcal{ALC}$ concept has an $\mathcal{ALC}$ concept equivalent. For example, the labeled $\mathcal{ALC}$ concept ${}^{Q_2,Q_1}\alpha^{R_1(Q_2),R_2}$ is equivalent to $\exists R_2.\forall Q_2.\exists R_1.\forall Q_1.\alpha$.

Considering A as an atomic concept and α an $\mathcal{ALC}$ formula, the function $\sigma : \phi_{lc} \to \phi_c$ transform a labeled $\mathcal{ALC}$ concept into an $\mathcal{ALC}$ concept. It is defined recursive as follows:

$$\sigma\left({}^{\emptyset}\alpha^{\emptyset}\right) = \alpha$$
$$\sigma\left({}^{L_1,R}\alpha^{\emptyset}\right) = \sigma\left({}^{L_1}\forall R.\alpha^{\emptyset}\right)$$
$$\sigma\left({}^{L_1}\alpha^{R,L_2}\right) = \sigma\left({}^{\emptyset}\exists R.\sigma\left({}^{L_1}\alpha^{\emptyset}\right)^{L_2}\right)$$
$$\sigma\left({}^{L_1}\alpha^{R(L_1),L_2}\right) = \sigma\left({}^{L_1}\exists R.\alpha^{L_2}\right)$$
$$\sigma\left({}^{L,L1}\alpha^{R(L),L_2}\right) = \sigma\left({}^{L}\sigma\left({}^{L_1}\alpha^{\emptyset}\right)^{R(L),L2}\right)$$

We define $\Delta \Rightarrow \Gamma$ as a *sequent* where Δ and Γ are finite sequences of labeled concepts. The natural interpretation of the sequent $\Delta \Rightarrow \Gamma$ is the $\mathcal{ALC}$ formula $\bigsqcap_{\delta\in\Delta} \sigma(\delta) \sqsubseteq \bigsqcup_{\gamma\in\Gamma} \sigma(\gamma)$. Note that σ is defined only over labeled $\mathcal{ALC}$ concepts.

The lists of labels will be omitted whenever it is clear that a rule does not take into account their specific form. This is the case for the structural rules. Capital greek letters stand for lists of labeled formulas. If ${}^{L_1}\gamma^{L_2}$ is a consistently labeled formula then $\mathcal{D}(L_2)$ is the set of role symbols that occur inside the *skolemized role expressions* in L_2. Note that $\mathcal{D}(L_2) \subseteq L_1$ always holds. The notation ${}^{L_1}\Gamma^{L_2}$ has to be taken as a list of labeled formulas of the form ${}^{L_1}\gamma_1{}^{L_2}, \ldots, {}^{L_1}\gamma_k{}^{L_2}$ for all $\gamma \in \Gamma$.

Considering the labeled formula ${}^{L_1}\alpha^{L_2}$, the notation ${}^{\frac{L_2}{L_1}}\beta^{\frac{L_1}{L_2}}$ denotes exchanging the universal roles occurring in L_1 for the existential roles occurring in L_2 in a consistent way such that the skolemization is dualy placed. Thus, if $\beta \equiv \neg\alpha$ the formulas will be a negation each other. For example, ${}^{\frac{R(Q)}{Q}}\alpha^{\frac{Q}{R(Q)}}$ is ${}^{R}\alpha^{Q(R)}$.

The restrictions in the rules ($\forall$-r) and ($\forall$-l) means that the role R can only be removed from the left list of labels if none of the skolemized role expressions in the right list depends on it.

4 Soundness

The soundness of $\mathcal{S}_{\mathcal{ALC}}$ is proved by taking into account the intuitive meaning of each sequent and establishing that the truth preservation holds. As said in the last section, a sequent $\Delta \Rightarrow\ \Gamma$ is equivalent in meaning to the $\mathcal{ALC}$ formula:

$$\bigsqcap_{\delta \in \Delta} \sigma\left(\delta\right) \sqsubseteq \bigsqcup_{\gamma \in \Gamma} \sigma\left(\gamma\right)$$

A sequent is defined to be *valid* or a *tautology* if and only if its corresponding $\mathcal{ALC}$ formula is.

When using the calculus, the usual axioms of a particular DL theory (TBox or an ontology) of the form $C \sqsubseteq D$ should be taken as the sequent $C \Rightarrow D$. Labeled formulas occur only during the proof procedure, since they are in practical terms taken as intermediate data.

Theorem 1 ($\mathcal{S}_{\mathcal{ALC}}$ is sound)**.** *Considering Ω a set of sequents, a theory or a TBox, let an Ω-proof be any $\mathcal{S}_{\mathcal{ALC}}$ proof in which sequents from Ω are permitted as initial sequents (in addition to the logical axioms). The soundness of $\mathcal{S}_{\mathcal{ALC}}$ states that if a sequent $\Delta \Rightarrow \Gamma$ has an Ω-proof, then $\Delta \Rightarrow \Gamma$ is satisfied by every interpretation which satisfies Ω. That is,*

$$\textit{if} \quad \Omega \vdash_{\mathcal{S}_{\mathcal{ALC}}} \Delta \Rightarrow\ \Gamma \quad \textit{then} \quad \Omega \models \bigsqcap_{\delta \in \Delta} \sigma\left(\delta\right) \sqsubseteq \bigsqcup_{\gamma \in \Gamma} \sigma\left(\gamma\right)$$

for all interpretation $.^{\mathcal{I}}$.

Proof. We proof Theorem 1 by induction on the length of the Ω-proofs. The length of a Ω-proof is the number of applications for any derivation rule of the calculus.

base case Proofs with length zero are proofs $\Omega \vdash \Delta \Rightarrow\ \Gamma$ where $\Delta \Rightarrow\ \Gamma$ occurs in Ω. In that case, it is easy to see that the theorem holds.

For the initial sequents, logical axioms like $C \Rightarrow C$, it is easy to see that $\sigma(C)^{\mathcal{I}} \subseteq \sigma(C)^{\mathcal{I}}$ for every interpretation $\mathcal{I}$ since every set is a subset of itself.

Induction hypothesis As inductive hypothesis, we will consider that for proofs of length n the theorem holds. It is now sufficient to show that each of the derivation rules preserves the truth. That is, if the premises holds, the conclusion must also hold.

$$\frac{}{\alpha \Rightarrow \alpha}$$

$$\frac{\Delta \Rightarrow \Gamma}{\Delta, \alpha \Rightarrow \Gamma}\ \text{weak-l} \qquad \frac{\Delta \Rightarrow \Gamma}{\Delta \Rightarrow \Gamma, \alpha}\ \text{weak-r}$$

$$\frac{\Delta_1, \delta_1, \delta_2, \Delta_2 \Rightarrow \Gamma}{\Delta_1, \delta_2, \delta_1, \Delta_2 \Rightarrow \Gamma}\ \text{perm-l} \qquad \frac{\Delta \Rightarrow \Gamma_1, \gamma_1, \gamma_2, \Gamma_2}{\Delta \Rightarrow \Gamma_1, \gamma_2, \gamma_1, \Gamma_2}\ \text{perm-r}$$

$$\frac{\Delta, {}^{L_1,R}\alpha^{L_2} \Rightarrow \Gamma}{\Delta, {}^{L_1}(\forall R.\alpha)^{L_2} \Rightarrow \Gamma}\ \forall\text{-l} \qquad \frac{\Delta \Rightarrow \Gamma, {}^{L_1,R}\alpha^{L_2}}{\Delta \Rightarrow \Gamma, {}^{L_1}(\forall R.\alpha)^{L_2}}\ \forall\text{-r}$$

$$\frac{\Delta, {}^{L_1}\alpha^{R(L_1),L_2} \Rightarrow \Gamma}{\Delta, {}^{L_1}(\exists R.\alpha)^{L_2} \Rightarrow \Gamma}\ \exists\text{-l} \qquad \frac{\Delta \Rightarrow \Gamma, {}^{L_1}\alpha^{R(L_1),L_2}}{\Delta \Rightarrow \Gamma, {}^{L_1}(\exists R.\alpha)^{L_2}}\ \exists\text{-r}$$

$$\frac{\Delta, {}^{L}\alpha^{\emptyset}, {}^{L}\beta^{\emptyset} \Rightarrow \Gamma}{\Delta, {}^{L}(\alpha \sqcap \beta)^{\emptyset} \Rightarrow \Gamma}\ \sqcap\text{-l} \qquad \frac{\Delta \Rightarrow \Gamma, {}^{L}\alpha^{\emptyset} \quad \Delta \Rightarrow \Gamma, {}^{L}\beta^{\emptyset}}{\Delta \Rightarrow \Gamma, {}^{L}(\alpha \sqcap \beta)^{\emptyset}}\ \sqcap\text{-r}$$

$$\frac{\Delta, {}^{\emptyset}\alpha^{L} \Rightarrow \Gamma \quad \Delta, {}^{\emptyset}\beta^{L} \Rightarrow \Gamma}{\Delta, {}^{\emptyset}(\alpha \sqcup \beta)^{L} \Rightarrow \Gamma}\ \sqcup\text{-l} \qquad \frac{\Delta \Rightarrow \Gamma, {}^{\emptyset}\alpha^{L}, {}^{\emptyset}\beta^{L}}{\Delta \Rightarrow \Gamma, {}^{\emptyset}(\alpha \sqcup \beta)^{L}}\ \sqcup\text{-r}$$

$$\frac{\Delta \Rightarrow \Gamma, {}^{L_1}\alpha^{L_2}}{\Delta, {}^{\frac{L_2}{L_1}}\neg\alpha^{\frac{L_1}{L_2}} \Rightarrow \Gamma}\ \neg\text{-l} \qquad \frac{\Delta, {}^{L_1}\alpha^{L_2} \Rightarrow \Gamma}{\Delta \Rightarrow \Gamma, {}^{\frac{L_2}{L_1}}\neg\alpha^{\frac{L_1}{L_2}}}\ \neg\text{-r}$$

$$\frac{{}^{L_1}\alpha^{L_2} \Rightarrow {}^{M_1}\gamma^{N_1}, \ldots, {}^{M_n}\gamma^{N_n}}{{}^{L_1}\alpha^{L_2,R} \Rightarrow {}^{M_1}\gamma^{N_1,R}, \ldots, {}^{M_n}\gamma^{N_n,R}}\ \text{prom-1}$$

$$\frac{{}^{M_1}\gamma^{N_1}, \ldots, {}^{M_n}\gamma^{N_n} \Rightarrow {}^{L_1}\alpha^{L_2}}{{}^{R,M_1}\gamma^{N_1}, \ldots, {}^{R,M_n}\gamma^{N_n} \Rightarrow {}^{R,L_1}\alpha^{L_2}}\ \text{prom-2}$$

$$\frac{\Delta_1 \Rightarrow \Gamma_1, {}^{L_1}\alpha^{L_2} \quad {}^{L_1}\alpha^{L_2}, \Delta_2 \Rightarrow \Gamma_2}{\Delta_1, \Delta_2 \Rightarrow \Gamma_1, \Gamma_2}\ \text{cut}$$

In all rules α, β means $\mathcal{ALC}$ concepts (formulas without labels), γ_i and δ_i stands for labeled concepts, Γ_i and Δ_i for list of labeled concepts (for clean presentation of the rules the list of labels are omitted). In the rules ($\forall$-r) and ($\forall$-l) note that $R \notin \mathcal{D}(L_2)$. In the rule ($\exists$-l), $R \notin (L_1 \cup L_2)$. In the rules (prom-1) and (prom-2), M_i and N_i stands for list of roles.

Figure 1: The System $\mathcal{S}_{\mathcal{ALC}}$

Cut rule Given the sequents $\Delta_1 \Rightarrow \Gamma_1, {}^{L_1}C^{L_2}$ and ${}^{L_1}C^{L_2}, \Delta_2 \Rightarrow \Gamma_2$ then, by hypothesis, we know that they are valid and so

$$\bigcap_{\delta \in \Delta_1} \sigma(\delta)^{\mathcal{I}} \subseteq \bigcap_{\gamma \in \Gamma_1} \sigma(\gamma)^{\mathcal{I}} \cup \sigma({}^{L_1}C^{L_2})^{\mathcal{I}}$$

and

$$\sigma({}^{L_1}C^{L_2})^{\mathcal{I}} \cap \bigcap_{\delta \in \Delta_2} \sigma(\delta)^{\mathcal{I}} \subseteq \bigcap_{\gamma \in \Gamma_2} \sigma(\gamma)^{\mathcal{I}}$$

Let $A = \bigcap_{\delta \in \Delta_1} \sigma(\delta)^{\mathcal{I}}$, $B = \bigcap_{\gamma \in \Gamma_1} \sigma(\gamma)^{\mathcal{I}}$, $C = \bigcap_{\delta \in \Delta_2} \sigma(\delta)^{\mathcal{I}}$, $D = \bigcap_{\gamma \in \Gamma_2} \sigma(\gamma)^{\mathcal{I}}$ and $X = \sigma({}^{L_1}C^{L_2})^{\mathcal{I}}$. Now me must show that the application of the *cut rule* preserves the set inclusion. In other words, given $A \subseteq (B \cup X)$ and $(X \cap C) \subseteq D$, we must have $(A \cap C) \subseteq (B \cup D)$. Which is easy to show using the standard set theory.

Rules *weak-l* and *weak-r* Given the sequent $\Delta \Rightarrow \Gamma$, by the inductive hypothesis we know that

$$\bigcap_{\delta \in \Delta} \sigma(\delta)^{\mathcal{I}} \subseteq \bigcap_{\gamma \in \Gamma} \sigma(\gamma)^{\mathcal{I}}$$

Let $A = \bigcap_{\delta \in \Delta} \sigma(\delta)^{\mathcal{I}}$ and $B = \bigcap_{\gamma \in \Gamma} \sigma(\gamma)^{\mathcal{I}}$. By the hypothesys, $A \subseteq B$ and so, by set theory, $A \cap X \subseteq B$ and $A \subseteq B \cup X$ for any set X interpretation of an arbitrary α formula. In the first case, we archive the interpretation of $\Delta, \alpha \Rightarrow \Gamma$. In the second case, we archive the interpretation of $\Delta \Rightarrow \Gamma, \alpha$. Showing that both rules are sound.

Rules *perm-l* and *perm-r* By the definition of the intuitive meaning of a sequent and its semantics, it is easy to see that both rules are sound. Note that the order of the formulas in both sides of a sequent do not change the intuitive meaning of its respective $\mathcal{ALC}$ formulas.

Rules *prom-1* and *prom-2* First, we note that for any C, D $\mathcal{ALC}$ concepts and R role, by the semantics of $\mathcal{ALC}$ and subsumption definition, we easily show that

$$C \sqsubseteq D \Rightarrow \exists R.C \sqsubseteq \exists R.D \tag{1}$$

$$C \sqsubseteq D \Rightarrow \forall R.C \sqsubseteq \forall R.D \tag{2}$$

The soundness of rule (prom-1) if easily proved using (1) and the axiom $\exists R.C \sqcup \exists R.D \equiv \exists R.(C \sqcup D)$. The soundness of rule (prom-2) is proved using (2) and the axiom $\forall R.C \sqcap \forall R.D \equiv \forall R.(C \sqcap D)$.

Rules $\forall$-r, $\forall$-l, $\exists$-r and $\exists$-l From the definition of σ function, we know that in all those four rules, both the premises and the conclusions have, given a interpretation function, the same semantics.

Rule $\sqcap$-l Taking the sequent $\Delta, {}^{L}\alpha^{\emptyset}, {}^{L}\beta^{\emptyset} \Rightarrow \Gamma$ valid as hypothesis, we know that

$$\bigcap_{\delta \in \Delta} \sigma(\delta)^{\mathcal{I}} \cap \sigma({}^{L}\alpha^{\emptyset})^{\mathcal{I}} \cap \sigma({}^{L}\beta^{\emptyset})^{\mathcal{I}} \subseteq \bigcap_{\gamma \in \Gamma} \sigma(\gamma)^{\mathcal{I}}$$

holds. To show that the rule ($\sqcap$-l) is sound, We must prove that $\Delta, {}^{L}(\alpha \sqcap \beta)^{\emptyset} \Rightarrow \Gamma$ is also valid. In other words,

$$\bigcap_{\delta \in \Delta} \sigma(\delta)^{\mathcal{I}} \cap \sigma({}^{L}(\alpha \sqcap \beta)^{\emptyset})^{\mathcal{I}} \subseteq \bigcap_{\gamma \in \Gamma} \sigma(\gamma)^{\mathcal{I}}$$

Let $A = \bigcap_{\delta \in \Delta} \sigma(\delta)^{\mathcal{I}}$ and $B = \bigcap_{\gamma \in \Gamma} \sigma(\gamma)^{\mathcal{I}}$, we can rewrite the hypothesis as $(A \cap \sigma({}^{L}\alpha^{\emptyset})^{\mathcal{I}} \cap \sigma({}^{L}\beta^{\emptyset})^{\mathcal{I}}) \subseteq B$. Now, by the definition of σ and the axiom (Section 2)

$$\forall R.\alpha \sqcap \forall R.\beta \equiv \forall R.(\alpha \sqcap \beta)$$

we have that $A \cap \sigma({}^{L}\alpha \sqcap \beta^{\emptyset})^{\mathcal{I}} \subseteq B$ by induction over the list of labels L.

Rule $\sqcap$-r By induction hypothesis $\Delta \Rightarrow \Gamma, {}^{L}\alpha^{\emptyset}$ and $\Delta \Rightarrow \Gamma, {}^{L}\beta^{\emptyset}$ are valid sequents, and so,

$$\bigcap_{\delta \in \Delta} \sigma(\delta)^{\mathcal{I}} \subseteq \bigcap_{\gamma \in \Gamma} \sigma(\gamma)^{\mathcal{I}} \cup \sigma({}^{L}\alpha^{\emptyset})^{\mathcal{I}}$$

and

$$\bigcap_{\delta \in \Delta} \sigma(\delta)^{\mathcal{I}} \subseteq \bigcap_{\gamma \in \Gamma} \sigma(\gamma)^{\mathcal{I}} \cup \sigma({}^{L}\beta^{\emptyset})^{\mathcal{I}}$$

holds for all interpretations $.^{\mathcal{I}}$. Now, suppose the application of the rule ($\sqcap$-r) over the two sequent above.

We must show that $\Delta \Rightarrow \Gamma, {}^{L}(\alpha \sqcap \beta)^{\emptyset}$ is also valid, that is,

$$\bigcap_{\delta \in \Delta} \sigma(\delta)^{\mathcal{I}} \subseteq \bigcap_{\gamma \in \Gamma} \sigma(\gamma)^{\mathcal{I}} \cup \sigma({}^{L}(\alpha \sqcap \beta)^{\emptyset})^{\mathcal{I}}$$

holds.

Let $A = \bigcap_{\delta \in \Delta} \sigma(\delta)^{\mathcal{I}}$ and $B = \bigcap_{\gamma \in \Gamma} \sigma(\gamma)^{\mathcal{I}}$ to rewrite the hypothesis to $A \subseteq (B \cup \sigma({}^{L}\alpha^{\emptyset})^{\mathcal{I}})$ and $A \subseteq (B \cup \sigma({}^{L}\beta^{\emptyset})^{\mathcal{I}})$. By basic set theory axioms, given that hypothesis we have

$$A \subseteq ((B \cup \sigma({}^{L}\alpha^{\emptyset})^{\mathcal{I}}) \cap (B \cup \sigma({}^{L}\beta^{\emptyset})^{\mathcal{I}}))$$

that, by distributive law

$$A \subseteq B \cup (\sigma({}^{L}\alpha^{\emptyset})^{\mathcal{I}} \cap \sigma({}^{L}\beta^{\emptyset})^{\mathcal{I}})$$

Finally, by the definition of σ and the axiom $\forall R.C \sqcap \forall R.D \equiv \forall R.(C \sqcap D)$ we conclude that $A \subseteq B \cup \sigma({}^{L}C \sqcap D^{\emptyset})^{\mathcal{I}}$ is valid.

Rule $\sqcup$-l As inductive hypothesis the sequents $\Delta, {}^{\emptyset}\alpha^{L} \Rightarrow \Gamma$ and $\Delta, {}^{\emptyset}\beta^{L} \Rightarrow \Gamma$ are valid. That is, given $A = \bigcap_{\delta\in\Delta} \sigma(\delta)^{\mathcal{I}}$ and $B = \bigcap_{\gamma\in\Gamma} \sigma(\gamma)^{\mathcal{I}}$, we know that

$$A \cap \sigma({}^{\emptyset}\alpha^{L})^{\mathcal{I}} \subseteq B \qquad \text{and} \qquad A \cap \sigma({}^{\emptyset}\beta^{L})^{\mathcal{I}} \subseteq B$$

holds. Now considering the application of the rule ($\sqcup$-l) over the two sequents above we must prove that the resulting sequent $\Delta, {}^{\emptyset}(\alpha \sqcup \beta)^{L} \Rightarrow \Gamma$ is also valid:

$$A \cap \sigma({}^{\emptyset}(\alpha \sqcup \beta)^{L})^{\mathcal{I}} \subseteq B$$

Following from the hypothesis and basic set theory we know that if $A \cap X_1 \subseteq B$ an $A \cap X_2 \subseteq B$ than $(A \cap X_1) \cup (A \cap X_2) \subseteq B$ which gives

$$A \cap (\sigma({}^{\emptyset}\alpha^{L})^{\mathcal{I}} \cup \sigma({}^{\emptyset}\beta^{L})^{\mathcal{I}}) \subseteq B$$

and by the axiom $\exists R.C \sqcup \exists R.D \equiv \exists R.(C \sqcup D)$ and the induction over the list L we have the desired semantics of the resulting sequent:

$$A \cap (\sigma({}^{\emptyset}(\alpha \sqcup \beta)^{L})^{\mathcal{I}}) \subseteq B$$

Rule $\sqcup$-r The inductive hypothesis is that $\Delta \Rightarrow \Gamma, {}^{\emptyset}\alpha^{L}, {}^{\emptyset}\beta^{L}$ is valid. As we did before, taken $A = \bigcap_{\delta\in\Delta} \sigma(\delta)^{\mathcal{I}}$ and $B = \bigcap_{\gamma\in\Gamma} \sigma(\gamma)^{\mathcal{I}}$, it means

$$A \subseteq B \cup \sigma({}^{\emptyset}\alpha^{L})^{\mathcal{I}} \cup \sigma({}^{\emptyset}\beta^{L})^{\mathcal{I}}$$

holds. Now by the axiom $\exists R.(\alpha \sqcup \beta) \equiv \exists R.\alpha \sqcup \exists R.\beta$, we get $A \subseteq B \cup \sigma({}^{\emptyset}(\alpha \sqcup \beta)^{L})^{\mathcal{I}}$. Wish is the semantics of the sequent $\Delta \Rightarrow \Gamma, {}^{\emptyset}(\alpha \sqcup \beta)^{L}$.

Rules $\neg$-l and $\neg$-r Given a concept ${}^{L_1}\alpha^{L_2}$ and a interpretation $.^{\mathcal{I}}$ we define the set $X = \sigma({}^{L_1}\alpha^{L_2})^{\mathcal{I}}$ and the interpretation of its negation, ${}^{\frac{L_2}{L_1}}\neg\alpha^{\frac{L_1}{L_2}}$, will be the set $\overline{X} = \Delta^{\mathcal{I}} \setminus X$. Moreover, lets take $A = \bigcap_{\delta\in\Delta} \sigma(\delta)^{\mathcal{I}}$ and $B = \bigcap_{\gamma\in\Gamma} \sigma(\gamma)^{\mathcal{I}}$ to prove that each of the rules are sound.

For rule ($\neg$-l), the inductive hypothesis is that the premise $\Delta \Rightarrow \Gamma, {}^{L_1}\alpha^{L_2}$ is valid. Which means that $A \subseteq (B \cup X)$. From the basic set theory this implies that $(A \cap \overline{X}) \subseteq B$, witch is the interpretation of the conclusion.

For rule ($\neg$-r), the inductive hypothesis is that the premise $\Delta, {}^{L_1}\alpha^{L_2} \Rightarrow \Gamma$ is valid. Which means that $(A \cap X) \subseteq B$. From the basic set theory this implies that $A \subseteq (B \cup \overline{X})$, the interpretation of the conclusion as desired. □

5 Completeness

We show relative completeness of $\mathcal{S}_{\mathcal{ALC}}$ regarding the axiomatic presentation of $\mathcal{ALC}$. Since $\mathcal{ALC}$ formulas are not labeled, the completeness must take into account only formulas with, both, the universal list of roles and the existential list of roles as empty lists. Proceeding in this way, the $\mathcal{ALC}$ sequent calculus deduction rules without labels behave exactly as sequent calculus rules for classical propositional logic. Thus, in order to prove that $\mathcal{S}_{\mathcal{ALC}}$ is complete, we have only to derive the axiom $\forall R.(\alpha \sqcap \beta) \equiv \forall R.\alpha \sqcap \forall R.\beta$ and the necessitation rule. The reader must note that the derivation of the necessitation rule is accomplished by

$$\dfrac{\dfrac{\Rightarrow \alpha}{\Rightarrow {}^{R}\alpha}\;{\scriptstyle prom-2}}{\Rightarrow \forall R.\alpha}\;{\scriptstyle \forall - r}$$

On the other hand, the derivation of the axiom $\forall R.(\alpha \sqcap \beta) \equiv \forall R.\alpha \sqcap \forall R.\beta$ is obtained from the following derivations.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{{}^{R}\alpha \Rightarrow {}^{R}\alpha}{{}^{R}\alpha, {}^{R}\beta \Rightarrow {}^{R}\alpha}\;{\scriptstyle weak-l}}{{}^{R}\alpha, {}^{R}\beta \Rightarrow \forall R.\alpha}\;{\scriptstyle \forall - r} \qquad \dfrac{\dfrac{{}^{R}\beta \Rightarrow {}^{R}\beta}{{}^{R}\alpha, {}^{R}\beta \Rightarrow {}^{R}\beta}\;{\scriptstyle weak-l}}{{}^{R}\alpha, {}^{R}\beta \Rightarrow \forall R.\beta}\;{\scriptstyle \forall - r}}{{}^{R}\alpha, {}^{R}\beta \Rightarrow \forall R.\alpha \sqcap \forall R.\beta}\;{\scriptstyle \sqcap - r}}{{}^{R}(\alpha \sqcap \beta) \Rightarrow \forall R.\alpha \sqcap \forall R.\beta}\;{\scriptstyle \sqcap - l}}{\forall R.(\alpha \sqcap \beta) \Rightarrow \forall R.\alpha \sqcap \forall R.\beta}\;{\scriptstyle \forall - l}$$

, and

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{{}^{R}\alpha \Rightarrow {}^{R}\alpha}{\forall R.\beta, {}^{R}\alpha \Rightarrow {}^{R}\alpha}\;{\scriptstyle weak-l}}{\forall R.\beta, \forall R.\alpha \Rightarrow {}^{R}\alpha}\;{\scriptstyle \forall - l} \qquad \dfrac{\dfrac{{}^{R}\beta \Rightarrow {}^{R}\beta}{\forall R.\alpha, {}^{R}\beta \Rightarrow {}^{R}\beta}\;{\scriptstyle weak-l}}{\forall R.\alpha, \forall R.\beta \Rightarrow {}^{R}\beta}\;{\scriptstyle \forall - l}}{\forall R.\alpha, \forall R.\beta \Rightarrow {}^{R}(\alpha \sqcap \beta)}\;{\scriptstyle \sqcap - r}}{\forall R.\alpha \sqcap \forall R.\beta \Rightarrow {}^{R}(\alpha \sqcap \beta)}\;{\scriptstyle \sqcap - l}}{\forall R.\alpha \sqcap \forall R.\beta \Rightarrow \forall R.(\alpha \sqcap \beta)}\;{\scriptstyle \forall - r}$$

6 An example of proof using $\mathcal{S}_{\mathcal{ALC}}$

The formalization of text-based information is an important issue for many organizations. It is very common to encounter situations where knowledge stored in natural-language documents must be made available to agents (human or software-based) for processing and decision-making.

In [4] we discuss the principles involved in an ontology-based approach to the formalization of normative texts in the domain of Information Security (IS). In that paper, we discuss the use of tools and techniques from the fields of natural-language understanding, Description Logics and ontologies to formalize (and extract knowledge from) natural-language texts.

We must briefly present some IS-related terminology: security controls (or simply *controls*) are low-level technical measures that can be deployed in order to protect the organization's devices and processes against potential threats; and *security policy*, consisting of set of *actions* to be taken in order to comply with the adopted *security standards*. All of these concepts will be formalize as concepts in an ontology.

An important issue after the formalization of the normative texts is the *validation* of security controls against the policies which the controls are supposed to implement. For the *validation* task we developed the $\mathcal{S}_{\mathcal{ALC}}$. This section presents a simple example of how a subsumption inference could be proved using $\mathcal{S}_{\mathcal{ALC}}$ system and also the outline of the proof constructed. This outline gives the basic ideas of how this proof could be further explained to a non-technical users.

$$\begin{array}{lcl}
\mathit{AdministerRemotely} & \sqsubseteq & \mathit{AccessRemotely} \\
\mathit{NetworkConnect} & \sqsubseteq & \mathit{NetworkTraffic} \\
\mathit{NetwareServer} & \sqsubseteq & \mathit{System} \\
\mathit{Action0002} & \equiv & \exists \mathit{hasVerb}.(\mathit{Configure} \sqcap \\
 & & \quad \exists \mathit{hasTheme}.\mathit{System} \sqcap \\
 & & \quad \exists \mathit{hasPurpose}.(\mathit{Encrypt} \sqcap \\
 & & \qquad \exists \mathit{hasTheme}.(\mathit{NetworkConnect} \sqcap \\
 & & \qquad\quad \exists \mathit{isInstrumentOf}.(\mathit{AccessRemotely} \sqcap \\
 & & \qquad\qquad \exists \mathit{hasTheme}.\mathit{System})))) \\
\mathit{Control0001} & \equiv & \exists \mathit{hasVerb}.(\mathit{Encrypt} \sqcap \\
 & & \quad \exists \mathit{hasTheme}.\mathit{NetworkTraffic} \sqcap \\
 & & \quad \exists \mathit{hasInstrument}.\mathit{SSL} \sqcap \\
 & & \quad \exists \mathit{isInstrumentOf}.(\mathit{AdministerRemotely} \sqcap \\
 & & \qquad \exists \mathit{hasTheme}.\mathit{NetwareServer}))
\end{array}$$

$$\exists \mathit{hasVerb}.Y \equiv \exists \mathit{hasVerb}.(\mathit{Configure} \sqcap \exists \mathit{hasTheme}.X \sqcap \exists \mathit{hasPurpose}.Y)$$

Figure 2: Some axioms of an IS ontology

Figure 2 presents some axioms of an IS ontology. The intuitive meaning of the last axiom is that "Configuring X to achieve Y" is equivalent to "Achieving Y".

We usually would like to verify if a "security control" is the implementation of a specific "action". From a logical point of view, this can be stated as a subsumption problem – given two concepts to represent the control and the action. Let us considering the $\mathit{Control}0001$ and the $\mathit{Action}0002$ from the ontology in Figure 2. For the sake of better comprehension, the meaning of *Action0002* is "Configure every system to encrypt connections used for remote access to the system" and the meaning of *Control0001* is "Network traffic for the remote administration of the Netware server must be encrypted using SSL". [3]

Figure 3 presents the complete proof derivation of *Control0001* $\sqsubseteq$ *Action0002* using $\mathcal{S}_{\mathcal{ALC}}$. The outline of this proof is given bellow:

> Since "*Encrypt* the *NetworkConnection*" is the same as "*Encrypt* the *NetworkTraffic*", *NetwareServer* is a *System*, and *AdministerRemotely* implies *AccessRemotely*, then
>
> - *Control0001*, requiring that one
> - *Encrypt* the *NetworkTraffic* using *SSL* in order to *AdministerRemotely* the *NetwareServer*, implies
> - *Encrypt* the *NetworkTraffic* in order to *AdministerRemotely* the *NetwareServer*, and hence,
> - *Encrypt* the *NetworkTraffic* in order to *AccessRemotely* a *System*, and hence,
> - *Encrypt* the *NetworkConnection* in order to *AccessRemotely* a *System*, which conforms to

[3] That is, the concepts are the *formalizations* of the natural language texts.

- *Action0002, according to this detailed proof on Figure 3*

7 Conclusion and Further Works

As a the natural sequence of this article we are currently working on the proof that our system is still complete and sound without the “cut-rule”. This proof will follow the general idea of Gentzen’s Cut-Elimination Theorem [7].

Future investigation must also include: (1) the extension of this calculus in order to deal with stronger Description Logics, mainly, $\mathcal{SHIQ}$ [1]; (2) the development of a Natural Deduction System based on $\mathcal{S}_{\mathcal{ALC}}$.

Another interesting thing to be investigated is a comparison with others inference algorithms – like the structural subsumption algorithms and Tableaux [1] – regarding complexity. Furthermore, we have also start the development of a prototype theorem prover for $\mathcal{S}_{\mathcal{ALC}}$.

References

[1] F. Baader. *The Description Logic Handbook: theory, implementation, and applications.* Cambridge University Press, 2003.

[2] A. Borgida, E. Franconi, I. Horrocks, D. McGuinness, and P.F. Patel-Schneider. Explaining ALC subsumption. In *Proceddings of the International Workshop on Description Logics*, pages 33–36, 1999.

[3] Denise Aboim Sande de Oliveira, Clarisse Sieckenius de Souza, and Edward Hermann Haeusler. Structured argument generation in a logic based kb-system. In Lawrence S. Moss, Jonathan Ginzburg, and Maarten de Rijke, editors, *Logic Language and Computation*, volume 2 of *CSLI Lecture Notes*, pages 237–265. Stanford, California, 1 edition, 1999.

[4] Fernando Náufel do Amaral, Carlos Bazílio, Geiza Maria Hamazaki da Silva, Alexandre Rademaker, and Edward Hermann Haeusler. An Ontology-based Approach to the Formalization of Information Security Policies. *EDOCW*, 0:1, 2006.

[5] M. Fitting. *Proof methods for modal and intuitionistic logics.* Reidel, 1983.

[6] D. M. Gabbay. *Labelled deductive systems*, volume 1. Oxford University Press, 1996.

[7] J.Y. Girard et al. *Proofs and types.* Cambridge University Press New York, 1989.

[8] Edward Hermann Haeusler and Christian Jacques Renteria. A natural deduction system for CTL. *Bulletin of The Section of Logic*, 31(4):231, 2002.

[9] E. J. Lemmon. Algebraic semantics for modal logics i. *Journal of Symbolic Logic*, 31(1):46–65, 1966.

[10] HJ Levesque and RJ Brachman. Expressiveness and tractability in knowledge representation and reasoning. *Computational intelligence*, 3(2):78–93, 1987.

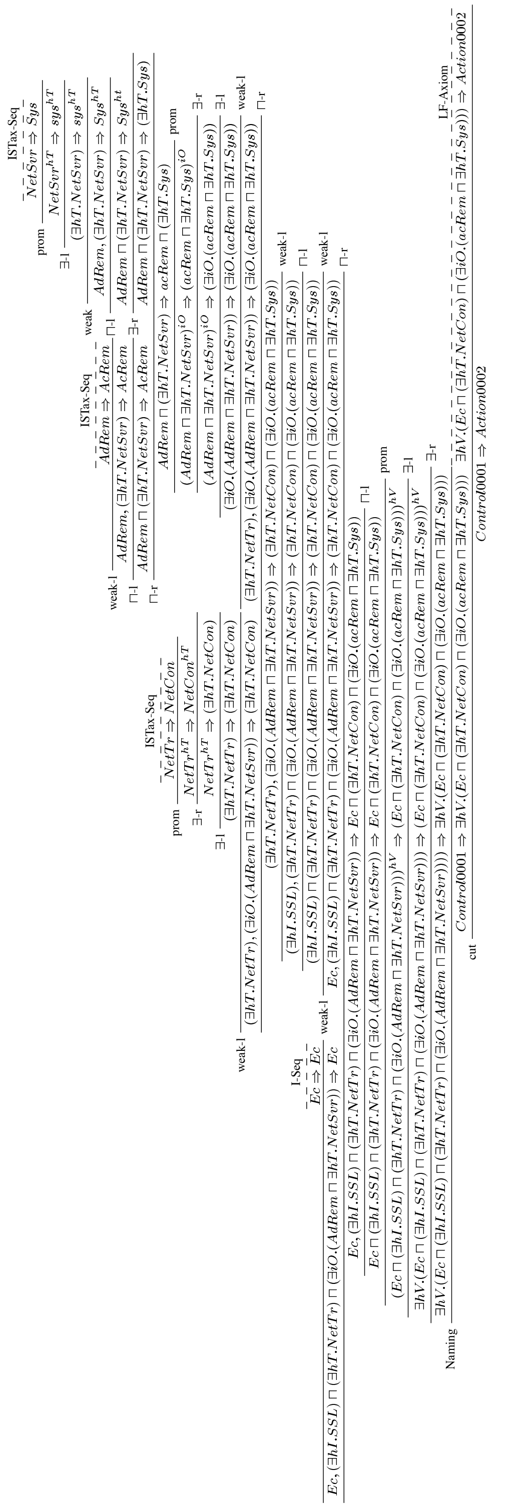
ISTax-Seq
$NetSvr \Rightarrow Sys$
prom
$NetSvr^{hT} \Rightarrow sys^{hT}$
∃-l
$(\exists hT.NetSvr) \Rightarrow sys^{hT}$
weak
$AdRem, (\exists hT.NetSvr) \Rightarrow Sys^{hT}$
⊓-l
$AdRem \sqcap (\exists hT.NetSvr) \Rightarrow Sys^{ht}$
∃-r
$AdRem \sqcap (\exists hT.NetSvr) \Rightarrow (\exists hT.Sys)$
ISTax-Seq
$AdRem \Rightarrow AcRem$
weak-l
$AdRem, (\exists hT.NetSvr) \Rightarrow AcRem$
⊓-l
$AdRem \sqcap (\exists hT.NetSvr) \Rightarrow AcRem$
⊓-r
$AdRem \sqcap (\exists hT.NetSvr) \Rightarrow acRem \sqcap (\exists hT.Sys)$
prom
$(AdRem \sqcap \exists hT.NetSvr)^{iO} \Rightarrow (acRem \sqcap \exists hT.Sys)^{iO}$
∃-r
$(AdRem \sqcap \exists hT.NetSvr)^{iO} \Rightarrow (\exists iO.(acRem \sqcap \exists hT.Sys))$
∃-l
$(\exists iO.(AdRem \sqcap \exists hT.NetSvr)) \Rightarrow (\exists iO.(acRem \sqcap \exists hT.Sys))$
weak-l
$(\exists hT.NetTr), (\exists iO.(AdRem \sqcap \exists hT.NetSvr)) \Rightarrow (\exists iO.(acRem \sqcap \exists hT.Sys))$
ISTax-Seq
$NetTr \Rightarrow NetCon$
prom
$NetTr^{hT} \Rightarrow NetCon^{hT}$
∃-r
$NetTr^{hT} \Rightarrow (\exists hT.NetCon)$
∃-l
$(\exists hT.NetTr) \Rightarrow (\exists hT.NetCon)$
weak-l
$(\exists hT.NetTr), (\exists iO.(AdRem \sqcap \exists hT.NetSvr)) \Rightarrow (\exists hT.NetCon)$
⊓-r
$(\exists hT.NetTr), (\exists iO.(AdRem \sqcap \exists hT.NetSvr)) \Rightarrow (\exists hT.NetCon) \sqcap (\exists iO.(acRem \sqcap \exists hT.Sys))$
weak-l
$(\exists hI.SSL), (\exists hT.NetTr) \sqcap (\exists iO.(AdRem \sqcap \exists hT.NetSvr)) \Rightarrow (\exists hT.NetCon) \sqcap (\exists iO.(acRem \sqcap \exists hT.Sys))$
⊓-l
$(\exists hI.SSL) \sqcap (\exists hT.NetTr) \sqcap (\exists iO.(AdRem \sqcap \exists hT.NetSvr)) \Rightarrow (\exists hT.NetCon) \sqcap (\exists iO.(acRem \sqcap \exists hT.Sys))$
weak-l
$Ec, (\exists hI.SSL) \sqcap (\exists hT.NetTr) \sqcap (\exists iO.(AdRem \sqcap \exists hT.NetSvr)) \Rightarrow (\exists hT.NetCon) \sqcap (\exists iO.(acRem \sqcap \exists hT.Sys))$
I-Seq
$Ec \Rightarrow Ec$
weak-l
$Ec, (\exists hI.SSL) \sqcap (\exists hT.NetTr) \sqcap (\exists iO.(AdRem \sqcap \exists hT.NetSvr)) \Rightarrow Ec$
⊓-r
$Ec, (\exists hI.SSL) \sqcap (\exists hT.NetTr) \sqcap (\exists iO.(AdRem \sqcap \exists hT.NetSvr)) \Rightarrow Ec \sqcap (\exists hT.NetCon) \sqcap (\exists iO.(acRem \sqcap \exists hT.Sys))$
⊓-l
$Ec \sqcap (\exists hI.SSL) \sqcap (\exists hT.NetTr) \sqcap (\exists iO.(AdRem \sqcap \exists hT.NetSvr)) \Rightarrow Ec \sqcap (\exists hT.NetCon) \sqcap (\exists iO.(acRem \sqcap \exists hT.Sys))$
prom
$(Ec \sqcap (\exists hI.SSL) \sqcap (\exists hT.NetTr) \sqcap (\exists iO.(AdRem \sqcap \exists hT.NetSvr)))^{hV} \Rightarrow (Ec \sqcap (\exists hT.NetCon) \sqcap (\exists iO.(acRem \sqcap \exists hT.Sys)))^{hV}$
∃-l
$\exists hV.(Ec \sqcap (\exists hI.SSL) \sqcap (\exists hT.NetTr) \sqcap (\exists iO.(AdRem \sqcap \exists hT.NetSvr))) \Rightarrow (Ec \sqcap (\exists hT.NetCon) \sqcap (\exists iO.(acRem \sqcap \exists hT.Sys)))^{hV}$
∃-r
$\exists hV.(Ec \sqcap (\exists hI.SSL) \sqcap (\exists hT.NetTr) \sqcap (\exists iO.(AdRem \sqcap \exists hT.NetSvr)))) \Rightarrow \exists hV.(Ec \sqcap (\exists hT.NetCon) \sqcap (\exists iO.(acRem \sqcap \exists hT.Sys)))$
Naming
$Control0001 \Rightarrow \exists hV.(Ec \sqcap (\exists hT.NetCon) \sqcap (\exists iO.(acRem \sqcap \exists hT.Sys)))$
LF-Axiom
$\exists hV.(Ec \sqcap (\exists hT.NetCon) \sqcap (\exists iO.(acRem \sqcap \exists hT.Sys))) \Rightarrow Action0002$
cut
$Control0001 \Rightarrow Action0002$

Figure 3: An example of a proof in $\mathcal{S}_{\mathcal{ALC}}$

[11] Deborah L. McGuinness. *Explaining Reasoning in Description Logics*. PhD thesis, Rutgers University, 1996.

[12] Christian Jacques Renteria, Edward Hermann Haeusler, and Paulo A. S. Veloso. NUL: Natural deduction for ultrafilter logic. *Bulletin of The Section of Logic*, 32(4):191–200, 2003.

[13] Christian Jacques RenterÃa. *Uma abordagem geral para quantificadores em deduÃ§Ã£o natural*. PhD thesis, PUC-Rio, DI, 2000.

[14] Christian Jacques RenterÃa and Edward Hermann Haeusler. A natural deduction system for keisler logic. *Eletronic Notes in Theoretical Computer Science*, 123:229–240, 2005.

[15] Klaus Schild. A correspondence theory for terminological logics: Preliminary report. Technical Report 91, Technische Universitat Berlin: IJCAI, 1991.

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