Toward Short and Structural $\mathcal{ALC}$-Reasoning Explanations: A Sequent Calculus Approach

Alexandre Rademaker and Edward Hermann Haeusler

Department of Informatics, PUC-Rio, Rio de Janeiro, Brazil
{arademaker,hermann}@inf.puc-rio.br
http://www.tecmf.inf.puc-rio.br

Abstract. This article presents labelled sequent calculi $\mathcal{S}_{\mathcal{ALC}}$ and $\mathcal{S}^{[]}_{\mathcal{ALC}}$ for the basic Description Logic (DL) $\mathcal{ALC}$. Proposing Sequent Calculus (SC) for dealing with DL reasoning aims to provide a more structural way to generated explanations, from proofs as well as counter-models, in the context of Knowledge Base and Ontologies authoring tools. The ability of providing short (Polynomial) proofs is also considered as an advantage of *SC*-based explanations with regard to the well-known Tableaux-based reasoners. Both, $\mathcal{S}_{\mathcal{ALC}}$ and $\mathcal{S}^{[]}_{\mathcal{ALC}}$ satisfy cut-elimination, while $\mathcal{S}^{[]}_{\mathcal{ALC}}$ also provides $\mathcal{ALC}$ counter-example from unsuccessful proof-trees. Some suggestions for extracting explanations from proofs in the presented systems is also discussed.

Keywords: description logics, sequent calculus, proof theory.

1 Introduction

Logic languages have been frequently used as a basis to build tools and frameworks for Knowledge Representation (KR) and artificial intelligence (AI). Automatic Theorem Provers offered the mechanical support to achieve the derivation of meaningful utterances from already known statements. One could consider First-Order Logic (FOL) as a basis for *KR*-system, however, it is undecidable and hence, would not provide answers to any query. The use of decidable fragments of FOL and other logics have been considered instead. **DATALOG** is among the logics historically considered. In general after getting to a decidable logic to KR, one starts extending it in order to accommodate more powerful features, recursive **DATALOG** is an example. Other possibilities include extending by adding temporal, deontic or even action modalities to the core logic. However, authoring a Knowledge Base (KB) using any *KR*-Logic/Framework, is far from an easy task. It is a conceptually hard task, indeed. The underlying semantics of the specific domain (SD) has to be formalized by means of linguistic components as *Predicates*, *Individual-Designators*, *Rules or Axioms*, even *Modalities* and so on. The Logic Language should not provide any glue concerning this task, since it is expected to be neutral. Anyway, two main questions raise up when one designs a KB: (1) Is the KB consistent? (2) Is the KB representing the right Knowledge at all? Again, with the help of an Automatic Proof Procedure[1] for

[1] A Sat-Solver Procedure can be also considered.

G. Zaverucha and A. Loureiro da Costa (Eds.): SBIA 2008, LNAI 5249, pp. 167–176, 2008.

the underlying logic it is possible to design authoring environments to support the KB correct definition. It is recommendable to use a decidable underlying logic, any case. The authoring environment ability on providing glues on why the current KB is inconsistent is as important as the consistency test itself. It is also important the ability to *explain* how and why a known piece of knowledge is supported (or not) by the current KB. To either ability we simply refer to *explanation*, according to our terminology.

Description Logics (DL) are quite well-established as underlying logics for KR. The core of the DL is the $\mathcal{ALC}$ description logic. In a broader sense, a KB specified in any description logic having $\mathcal{ALC}$ as core is called an Ontology. We will not take any discussion on the concept just named as well as we will not discuss also the technological concerns around Ontologies and the **Web**. A DL theory presentation is a set of axioms in the DL logical language as well as an OWL-DL file. DL have implemented reasoners, for example Pellet and Racer,[2] only to mention two. There are also quite mature Ontology Editors. Protegé is the more popular and used free editor. However, the Reasoners (theorem provers) as well as the Editors do not have a good, if any, support for explanations. As far as we know, the existing *DL*-Reasoners implement the Tableaux proof procedure first published in [1]. Concerning approaches on explanation in DL, the papers [2,3] describe methods to extract explanations from *DL*-Tableaux proofs. In [4] it is described the explanation extraction in quite few details, making impossible a feasible comparison with the approach followed in our article.

Simple Tableaux[3] cannot produce short proofs (polynomially lengthy proofs). This follows from the theorem that asserts that simple Tableaux cannot produce short proofs for the Pigeonhole Principle (PHP). PHP is easily expressed in propositional logic, and hence, is also easily expressed in $\mathcal{ALC}$. On the other hand, Sequent Calculus (SC) (with the cut rule) has short proofs for PHP. In [5,6] it is shown, distinct SC proof procedures that incorporate mechanisms that are somehow equivalent to introducing cut-rules in a proof. Anyway, both articles show how to obtain short proofs, in SC, for the PHP. We believe that super-polynomial proofs, like the ones generated by simple Tableaux, cannot be considered as good sources for text generation. The reader might want to consider that only the reading of the proof itself is a super-polynomial task regarding time complexity.

The generation of an explanatory text from a formal proof is still under investigation by the community, at least if one consider *good* explanations. The use of Endophoras[4] in producing explanations is a must. However, the produced text should not contain unstructured nesting of endophoras, a text like this is hard to follow. Some structure is need relating the endophoras. We believe that

[2] Racer is an industrial product, and currently must be bought.

[3] A simple Tableaux is not able to implement analytical cuts. The Tableaux used for $\mathcal{ALC}$ is simple.

[4] An anaphora is a linguistic reference to an antecedent piece of text. A Cataphora is a linguistic reference to a posterior piece of text in a phrase. Endophora refer to both, anaphora and cataphora.

as more structured the proof is, as easier the generation of a better text, at least concerning the use of endophoras.

In section 2 we present $\mathcal{S}_{\mathcal{ALC}}$ and its main proof-theoretical features. Section 3 shows how the structural subsumption algorithm described in [7] for a fragment of $\mathcal{ALC}$ is subsumed by $\mathcal{S}_{\mathcal{ALC}}$. Section 4 shows how to obtain a counter-model from an unsuccessful proof-tree, providing an explanatory power for negative answers similar to Tableaux, from $\mathcal{S}^{[]}_{\mathcal{ALC}}$, an implementation-driven conservative extension of $\mathcal{S}_{\mathcal{ALC}}$. Section 5 presents an example of explanation extracted from a proof in $\mathcal{S}_{\mathcal{ALC}}$ focusing on comparing our system with the well-known work described in [2] as well as more recent works on $\mathcal{ALC}$-explanation. Our main contribution, besides the proof theoretical one, is an step towards short and structured explanations in $\mathcal{ALC}$ theorem proving.

2 The $\mathcal{ALC}$ Sequent Calculus

$\mathcal{ALC}$ is a basic description language [7] and its syntax of concept descriptions is described as follows:

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

where A stands for atomic concepts and R for atomic roles.

The Sequent Calculus for $\mathcal{ALC}$ ($\mathcal{S}_{\mathcal{ALC}}$) that it is shown in Figures 1 and 2 was first presented in [8] where it was proved to be sound and complete. It is based on the extension of the language ϕ_c. Labels are lists of (possibly skolemized) role symbols. Its syntax is as follows:

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

where R stands for roles, L for list of roles and $R(L)$ is an skolemized role expression.

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

Let α be an ϕ_{lc} formula; the function $\sigma : \phi_{lc} \rightarrow \phi_c$ transforms a labeled $\mathcal{ALC}$ concept into an $\mathcal{ALC}$ concept. $\mathcal{ALC}$ sequents are expressions of the form $\Delta \Rightarrow \Gamma$ where Δ and Γ are finite *sequences* of labeled concepts. The natural interpretation of the sequent $\Delta \Rightarrow \Gamma$ is the $\mathcal{ALC}$ formula $\bigsqcap_{\delta\in\Delta} \sigma(\delta) \sqsubseteq \bigsqcup_{\gamma\in\Gamma} \sigma(\gamma)$.

In Figures 1 and 2 the lists of labels are omitted whenever it is clear that a rule does not take into account their specific form. This is the case for the structural rules. In all rules α, β stands for $\mathcal{ALC}$ concepts (formulas without labels), γ, δ stands for labeled concepts, Γ, Δ for list of labeled concepts. L, M, N stands for list of roles. All of this letters may have indexes whenever necessary for distinction. If ${}^{L_1}\alpha^{L_2}$ 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.

Consider ${}^{L_1}\alpha^{L_2}$; the notation ${}^{\frac{L_2}{L_1}}\alpha^{\frac{L_1}{L_2}}$ denotes the exchanging of the universal roles occurring in L_1 for the existential roles occurring in L_2 in a consistent way

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

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

$$\frac{\Delta, \delta, \delta \Rightarrow \Gamma}{\Delta, \delta \Rightarrow \Gamma} \text{ contraction-l} \qquad \frac{\Delta \Rightarrow \Gamma, \gamma, \gamma}{\Delta \Rightarrow \Gamma, \gamma} \text{ contraction-r}$$

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

Fig. 1. The System $\mathcal{S}_{\mathcal{ALC}}$: structural rules

such that the skolemization is dually placed. This is used to express the negation of labeled concepts. If $\beta \equiv \neg\alpha$ the formula ${}^{\frac{Q}{R}}\beta^{\frac{R}{Q(R)}}$ is the negation of ${}^{R}\alpha^{Q(R)}$.

In the rules (∀-r) and (∀-l) $R \notin \mathcal{D}(L_2)$ must hold. In rules (prom-2), (∀-r) and (∀-l), the notation L_2', N_i' means the reconstructions of the skolemized expressions on those lists regarding the modification of the lists L_1 and M_i,

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

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

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

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

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

$$\frac{{}^{L_1}\alpha^{L_2} \Rightarrow {}^{M_1}\beta_1{}^{N_1}, \ldots, {}^{M_n}\beta_n{}^{N_n}}{{}^{L_1}\alpha^{L_2,R} \Rightarrow {}^{M_1}\beta_1{}^{N_1,R}, \ldots, {}^{M_n}\beta_n{}^{N_n,R}} \text{ prom-1}$$

$$\frac{{}^{M_1}\beta_1{}^{N_1}, \ldots, {}^{M_n}\beta_n{}^{N_n} \Rightarrow {}^{L_1}\alpha^{L_2}}{{}^{R,M_1}\beta_1{}^{N_1'}, \ldots, {}^{R,M_n}\beta_n{}^{N_n'} \Rightarrow {}^{R,L_1}\alpha^{L_2'}} \text{ prom-2}$$

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

Fig. 2. The System $\mathcal{S}_{\mathcal{ALC}}$: logical rules

respectively. The restrictions in the rules (∀-r) and (∀-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.

In [9] the cut-elimination[5] theorem is proved for $\mathcal{S}_{\mathcal{ALC}}$.

3 Comparing with the Structural Subsumption Algorithm

The *structural subsumption algorithms* (SSC), presented in [7], compare the syntactic structure of two normalized concept descriptions in order to verify if the first one is subsumed by the second one. Due to lack of space, we can only say that each step taken by a bottom-up construction of a $\mathcal{S}_{\mathcal{ALC}}$ proof corresponds to a step towards this matching by means of the *SSC* algorithms. The following construction on $\mathcal{S}_{\mathcal{ALC}}$ deals with normalized concepts (the sub-language of $\mathcal{ALC}$ required by *SSC*) [7]. It would conclude the subsumption (sequent) whenever the top-sequents ensure also their respective subsumptions. This is just what the (recursive) *SSC* does. Consider:

$$\dfrac{\dfrac{\dfrac{A_1 \Rightarrow B_1}{\forall R_1.C_1, A_1 \Rightarrow B_1}}{A_1, \forall R_1.C_1 \Rightarrow B_1} \quad \dfrac{\dfrac{\dfrac{{}^{R_1}C_1 \Rightarrow {}^{S_1}D_1}{{}^{R_1}C_1 \Rightarrow \forall S_1.D_1}}{\forall R_1.C_1 \Rightarrow \forall S_1.D_1}}{A_1, \forall R_1.C_1 \Rightarrow \forall S_1.D_1}}{\dfrac{A_1, \forall R_1.C_1 \Rightarrow B_1 \sqcap \forall S_1.D_1}{A_1 \sqcap \forall R_1.C_1 \Rightarrow B_1 \sqcap \forall S_1.D_1}}$$

4 Obtaining Counter-Models from Unsuccessful Proof-Trees

The structural subsumption algorithm is restricted to a quite inexpressive language and simple Tableaux based algorithms generally fails to provide short proofs. On the other hands, the later has an useful property, it returns a counter-model from an unsuccessful proof. A counter-model, that is, an interpretation that falsifies the premise, is a quite useful artifact to a knowledge-base engineer.

In this section we show how to extend $\mathcal{S}_{\mathcal{ALC}}$ system in order to be able to construct a counter-model from unsuccessful proofs too. In this way, $\mathcal{S}_{\mathcal{ALC}}$ can be compared with Tableaux algorithms, indeed. In fact $\mathcal{S}_{\mathcal{ALC}}$ is a structural-free sequent calculus designed to provide sequent proofs without considering backtracking during the proof-construction from conclusion to axioms. Concerning $\mathcal{ALC}$ the novelty is focused on the promotion and frozen rules, shown in the sequel.

Let us consider the system $\mathcal{S}^{[]}_{\mathcal{ALC}}$, a conservative extension of $\mathcal{S}_{\mathcal{ALC}}$, presented in Figure 3. $\mathcal{S}^{[]}_{\mathcal{ALC}}$ sequents are expressions of the form $\Delta \Rightarrow \Gamma$ where Δ and Γ

[5] This theorem states that any $\mathcal{S}_{\mathcal{ALC}}$-provable sequent can be proved without the cut-rule.

are *sets* of labeled concepts (possibly frozen). A frozen formula (labeled concept) α is represented as $[\alpha]$. The notation $[\Delta]$ means that each $\delta \in \Delta$ is frozen (i.e. $\{[\delta] \mid \delta \in \Delta\}$). Frozen formulas may also be indexed as $[\alpha]^n$. The notation can also be extended to a set of formulas $[\Delta]^n = \{[\delta]^n \mid \delta \in \Delta\}$.

Due to space limitation, it is only displayed the rules of $\mathcal{S}^{[]}_{\mathcal{ALC}}$ that modify the indexes of the formulas in the sequents. The remaining formulas are those of Figure 2 except *cut* (*prom-1* and *prom-2* of Figure 3 replace theirs corresponding in Figure 2) and considering that Δ and Γ range over formulas and frozen formulas (indexed or not).

Reading bottom-up the rules *prom-1* and *prom-2* freeze all formulas that do not contain the removed label. In rules *frozen-exchange*, *prom-1* and *prom-2* $[\Delta]^n$ and $[\Gamma]^n$ contains all the indexed-frozen formulas of the set of formulas in the antecedent (resp. succedent) of the sequent. The n is taken as the greatest index among all indexed-frozen formulas present in Γ and Δ. In rule *prom-2* the notation L'_2 means the reconstructions of the skolemized expressions on list L_2 regarding the modification of the list L_1. The notation $\Gamma^{(R)}$ and ${}^{(R)}\Gamma$ denotes the addition of the Role R to the beginning of the existential and universal labels respectively. In rule *frozen-exchange* all formulas in Δ_2 and Γ_2 must be atomic.

$$\frac{}{\Delta, \alpha \Rightarrow \ \alpha, \Gamma}$$

$$\frac{[\Delta]^n, [\Delta_1], {}^{L_1}\alpha^{L_2} \Rightarrow \Gamma_1, [\Gamma_2], [\Gamma]^n}{[\Delta]^n, \Delta_1, {}^{L_1}\alpha^{L_2,R} \Rightarrow \Gamma_1{}^{(R)}, \Gamma_2, [\Gamma]^n}\ \text{prom-1}$$

$$\frac{[\Delta]^n, [\Delta_2], \Delta_1 \Rightarrow {}^{L_1}\alpha^{L_2}, [\Gamma_1], [\Gamma]^n}{[\Delta]^n, \Delta_2, {}^{(R)}\Delta_1 \Rightarrow {}^{R,L_1}\alpha^{L'_2}, \Gamma_1, [\Gamma]^n}\ \text{prom-2}$$

$$\frac{\Delta_1, [\Delta_2]^{n+1}, [\Delta]^n \Rightarrow \Gamma_1, [\Gamma_2]^{n+1}, [\Gamma]^n}{[\Delta_1], \Delta_2, [\Delta]^n \Rightarrow [\Gamma_1], \Gamma_2, [\Gamma]^n}\ \text{frozen-exchange}$$

Fig. 3. Sub-system of $\mathcal{S}^{[]}_{\mathcal{ALC}}$ containing the rules that modify indexes

In Section 2, we stated the natural interpretation of a sequent $\Delta \Rightarrow \Gamma$ in $\mathcal{S}_{\mathcal{ALC}}$ as the $\mathcal{ALC}$-formula $\bigsqcap_{\delta\in\Delta} \sigma(\delta) \sqsubseteq \bigsqcup_{\gamma\in\Gamma} \sigma(\gamma)$. Given an interpretation function $\cdot^{\mathcal{I}}$ we write $\mathcal{I} \models \Delta \Rightarrow \Gamma$, if and only if, $\bigsqcap_{\delta\in\Delta} \sigma(\delta)^{\mathcal{I}} \sqsubseteq \bigsqcup_{\gamma\in\Gamma} \sigma(\gamma)^{\mathcal{I}}$ [8]. Now we have to extend that definition to give the *semantics* of sequents with frozen and indexed-frozen formulas.

Definition 1 (Satisfability of frozen-labeled sequents). *Let $\Delta \Rightarrow \Gamma$ be a sequent with its succedent and antecedent having formulas that range over labeled concepts, frozen labeled concepts and indexed-frozen labeled concept. This sequent has the general form $\Delta_1, [\Delta_2], [\Delta_3]^1, \ldots, [\Delta_k]^{k-2} \Rightarrow \Gamma_1, [\Gamma_2], [\Gamma_3]^1, \ldots, [\Gamma_k]^{k-2}$. Let $(\mathcal{I}, \mathcal{I}', \mathcal{I}_1 \ldots, \mathcal{I}_{k-2})$ be a tuple of interpretations. We say that this tuple satisfy $\Delta \Rightarrow \Gamma$, if and only if, one of the following clauses holds: $\mathcal{I} \models \Delta_1 \Rightarrow \Gamma_1$, $\mathcal{I}' \models \Delta_2 \Rightarrow \Gamma_2$, $\mathcal{I}_1 \models \Delta_3 \Rightarrow \Gamma_3$, ..., $\mathcal{I}_{k-2} \models \Delta_k \Rightarrow \Gamma_k$.*

Obviously, $\Delta \Rightarrow \Gamma$ is not satisfiable by a tuple of interpretations, if and only if, no interpretation in the tuple satisfy the corresponding indexed (sub) sequent.

The following lemma shows that $\mathcal{S}^{[]}_{\mathcal{ALC}}$ is a conservative extension of $\mathcal{S}_{\mathcal{ALC}}$.

Lemma 1. *Consider $\Delta \Rightarrow \Gamma$ a $\mathcal{S}_{\mathcal{ALC}}$ sequent. If P is a proof of $\Delta \Rightarrow \Gamma$ in $\mathcal{S}^{[]}_{\mathcal{ALC}}$ then it is possible to construct a proof P' of $\Delta \Rightarrow \Gamma$ in $\mathcal{S}_{\mathcal{ALC}}$.*

Proof. The proof of Lemma 1 is done by simple induction over the number of applications of *prom-1* and *prom-2* occurring in a proof P. Let us consider a top most application of rule *prom-1* of the system $\mathcal{S}^{[]}_{\mathcal{ALC}}$. Due to space limitations, we present *prom-1* cases only, *prom-2* can be dealt similarly.

Below we present how we can build a new proof P' in $\mathcal{S}_{\mathcal{ALC}}$ (right) from a proof P in $\mathcal{S}^{[]}_{\mathcal{ALC}}$ (left). We have to consider two cases according to the possible outcomes from the top most application of the rule *prom-1*. The first case deals with the occurrence of a *frozen-exchange* rule application between the *prom-1* application and the initial sequent. This situation happens whenever the formulas frozen by *prom-1* rule were necessary to obtain the initial sequent, in a bottom-up construction of a $\mathcal{S}^{[]}_{\mathcal{ALC}}$ proof. Note that above the top most *prom-1* application, there might exist at most one *frozen-exchange* application.

$$\dfrac{\dfrac{\dfrac{\Delta'_1, [\Delta^*_2], \alpha \Rightarrow \alpha, [\Gamma^*_1], \Gamma'_2}{\Pi_1}}{\Delta_1, [\Delta^*_2] \Rightarrow [\Gamma^*_1], \Gamma_2}\,\text{frozen-exch}}{[\Delta_1], \Delta^*_2 \Rightarrow \Gamma^*_1, [\Gamma_2]}\,\text{prom-1}$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\alpha \Rightarrow \alpha}{\Delta'_1, \alpha \Rightarrow \alpha, \Gamma'_2}\,\text{some weak}}{\Pi'_1}}{\Delta_1 \Rightarrow \Gamma_2}}{\Delta_1, {}^{L_1}\beta^{L_2,R} \Rightarrow {\Gamma_1}^R, \Gamma_2}\,\text{some weak}$$

In the second case, the initial sequent is obtained (bottom-up building of the proof) without the application of *frozen-exchange* rule:

$$\dfrac{\dfrac{[\Delta_1], \Delta'_2, \alpha \Rightarrow \alpha, \Gamma'_1, [\Gamma_2]}{\Pi_1}}{\Delta_1, {}^{L_1}\beta^{L_2,R} \Rightarrow {\Gamma_1}^R, \Gamma_2}\,\text{prom-1}$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\alpha \Rightarrow \alpha}{\Delta'_2, \alpha \Rightarrow \alpha, \Gamma'_1}\,\text{some weak}}{\Pi'_1}}{{}^{L_1}\beta^{L_2} \Rightarrow {\Gamma_1}^R}\,\text{prom-1}}{\Delta_1, {}^{L_1}\beta^{L_2,R} \Rightarrow {\Gamma_1}^R, \Gamma_2}\,\text{some weak}$$

Applying recursively the transformations above from top to bottom we obtain a proof in $\mathcal{S}_{\mathcal{ALC}}$ from a proof in $\mathcal{S}^{[]}_{\mathcal{ALC}}$. □

A fully atomic $\mathcal{S}^{[]}_{\mathcal{ALC}}$ sequent has every (frozen and not frozen) formula in the antecedent as well as in the succedent as atomic concepts. A fully expanded proof-tree of $\Delta \Rightarrow \Gamma$ is a tree having $\Delta \Rightarrow \Gamma$ as root, each internal node is a premise of a valid $\mathcal{S}^{[]}_{\mathcal{ALC}}$ rule application, and each leaf is either a $\mathcal{S}^{[]}_{\mathcal{ALC}}$axiom (initial sequent) or a fully atomic sequent. In the following lemma we are interested in fully expanded proof-trees that are not $\mathcal{S}^{[]}_{\mathcal{ALC}}$ proofs.

Lemma 2. *Let Π be a fully expanded proof-tree having $\Delta \Rightarrow \Gamma$, a sequent of $\mathcal{S}^{[]}_{\mathcal{ALC}}$, as root. From any non-initial top-sequent, one can explicitly define a counter-model for $\Delta^* \Rightarrow \Gamma^*$, where $\Delta^* \Rightarrow \Gamma^*$ is the $\mathcal{S}_{\mathcal{ALC}}$ sequent related to $\Delta \Rightarrow \Gamma$.*

Proof. Note that the construction of the counter-model (a tuple of interpretation) is made from top to bottom, starting from any top-sequent that is no initial in the fully expanded tree.

From the natural interpretation of a sequent in $\mathcal{S}_{\mathcal{ALC}}$ as an $\mathcal{ALC}$ concepts subsumption, we know that an interpretation $\mathcal{I}$ falsifies a sequent $\Delta \Rightarrow \Gamma$ if there exists and element c such that $c \in \Delta^{\mathcal{I}}$ and $c \notin \Gamma^{\mathcal{I}}$. In $\mathcal{S}^{[]}_{\mathcal{ALC}}$ a tuple of interpretation falsifies a sequent $\Delta \Rightarrow \Gamma$ if each of its elements (interpretations) falsifies the correspondent sequent (Definition 1).

We proof Lemma 2 by cases, considering each rule of the system $\mathcal{S}^{[]}_{\mathcal{ALC}}$. For each rule, we must prove that given a tuple t that falsifies its premises, one can provide a tuple t' that falsifies its conclusion. In t' each interpretation may be an extension of its correspondent in t.

It can be easily seen that from a non-initial and fully atomic top-sequent, we are able to define a tuple falsifying it.

Since rules ($\sqcap$-r), ($\sqcap$-l), ($\sqcup$-r), ($\sqcup$-l), ($\neg$-r), ($\neg$-l), ($\exists$-r), ($\exists$-l), ($\forall$-r) and ($\forall$-l) do not touch in the frozen formulas, to verify the preservation of falsity in those rules, one can consider just the second projection of the tuple. Due the lack of space, we only show the case ($\sqcup$-l). If there exist an $\mathcal{I}$ that falsifies $\Delta, {}^{\emptyset}\alpha^{L} \Rightarrow \Gamma$ then we can state that in the domain of $\mathcal{I}$ there exist an individual c such that $c \in \sigma(\Delta)^{\mathcal{I}} \cap \sigma({}^{\emptyset}\alpha^{L})^{\mathcal{I}}$ and $c \notin \sigma(\Gamma)^{\mathcal{I}}$. From the hypothesis and basic Set Theory, we can conclude that $c \in \sigma(\Delta)^{\mathcal{I}} \cap \sigma({}^{\emptyset}\alpha^{L})^{\mathcal{I}} \cup \sigma({}^{\emptyset}\beta^{L})^{\mathcal{I}}$, which from $\exists R.C \sqcup \exists R.D \equiv \exists R.(C \sqcup D)$ is equivalent to $c \in \sigma(\Delta)^{\mathcal{I}} \cap \sigma({}^{\emptyset}\alpha \sqcup \beta^{L})^{\mathcal{I}}$, falsifying the conclusion since $c \notin \sigma(\Gamma)^{\mathcal{I}}$. The same idea holds for the right premise.

For the rule *frozen-exchange* we must consider the whole tuple of interpretations, since it manipulates the frozen formulas. Suppose a tuple such that $(\mathcal{I}, \emptyset, \mathcal{I}_1, \ldots, \mathcal{I}_{n+1}) \not\models \Delta_1, [\Delta_3]^1, \ldots, [\Delta_k]^n, [\Delta_2]^{n+1} \Rightarrow \Gamma_1, [\Gamma_3]^1, \ldots, [\Gamma_k]^n, [\Gamma_2]^{n+1}$ where the second projection of the tuple is an empty interpretation since there is no frozen formulas without index in the premise. In order to falsify the conclusion of the rule one simply considers the tuple $(\mathcal{I}_{n+1}, \mathcal{I}, \mathcal{I}_1, \ldots, \mathcal{I}_n) \not\models \Delta_2, [\Delta_1], [\Delta_3]^1, \ldots, [\Delta_k]^n \Rightarrow \Gamma_2, [\Gamma_1], [\Gamma_3]^1, \ldots, [\Gamma_k]^n$.

For rule *prom-1*, suppose that $(\mathcal{I}, \mathcal{I}', \mathcal{I}_1, \ldots, \mathcal{I}_n)$ falsifies the premise. From that, we can obtain a new tuple $t' = (\mathcal{I}'', \emptyset, \mathcal{I}_1, \ldots, \mathcal{I}_n)$ that falsifies the conclusion ${}^{L_1}\alpha^{L_2,R}, \Delta_1, [\Delta_3]^1, \ldots, [\Delta_k]^n \Rightarrow {\Gamma_1}^{(R)}, \Gamma_2, [\Gamma_3]^1, \ldots, [\Gamma_k]^n$. To construct t', the unique non trivial part is the construction of $\mathcal{I}''$. This comes from $\mathcal{I}$ and $\mathcal{I}'$ from the tuple that falsifies the premise. $\mathcal{I}''$ preserve the structure of both $\mathcal{I}$ and $\mathcal{I}'$ and also the properties that make those interpretation falsify the premise. Due space limitation, the complete procedure can not be presented here. The principal idea is that whenever a *crash* of names of individuals occurs, that is, names that occurs in both, one can consistently renames the individuals from I or I'. The reader must note that this is a perfectly adequate construct.

Starting from a tuple of interpretations defined to falsify the fully atomic top-sequent we obtain by the preservation of the falsity, provide by the $\mathcal{S}^{[]}_{\mathcal{ALC}}$ rules, the falsity of the root of the proof-tree. We can see that the counter-model construction also works for $\mathcal{S}_{\mathcal{ALC}}$ sequents when submitted to the $\mathcal{S}^{[]}_{\mathcal{ALC}}$ proof-system. □

5 Providing Explanations from Proofs

Consider the proof:

$$\begin{array}{c}
Doctor \Rightarrow Doctor \\ \hline
Doctor \Rightarrow Rich, Doctor \quad \text{weak-r} \\ \hline
Doctor \Rightarrow (Rich \sqcup Doctor) \quad \sqcup\text{-r} \\ \hline
{}^{child}Doctor \Rightarrow {}^{child}(Rich \sqcup Doctor) \quad \text{prom-2} \\ \hline
\top, {}^{child}Doctor \Rightarrow {}^{child}(Rich \sqcup Doctor) \quad \text{weak-l} \\ \hline
\top \Rightarrow \neg Doctor^{child}, {}^{child}(Rich \sqcup Doctor) \quad neg\text{-r} \\ \hline
\top \Rightarrow \neg Doctor^{child}, Lawyer^{child}, {}^{child}(Rich \sqcup Doctor) \quad \text{weak-r} \\ \hline
\top \Rightarrow \neg Doctor^{child}, \exists child.Lawyer, {}^{child}(Rich \sqcup Doctor) \quad \exists\text{-r} \\ \hline
\top \Rightarrow \exists child.\neg Doctor, \exists child.Lawyer, {}^{child}(Rich \sqcup Doctor) \quad \exists\text{-r} \\ \hline
\top \Rightarrow ((\exists child.\neg Doctor) \sqcup (\exists child.Lawyer)), {}^{child}(Rich \sqcup Doctor) \quad \sqcup\text{-r} \\ \hline
\top^{child} \Rightarrow ((\exists child.\neg Doctor) \sqcup (\exists child.Lawyer))^{child}, {}^{child}(Rich \sqcup Doctor)^{child} \quad \text{prom-1} \\ \hline
\top^{child}, {}^{child}\neg((\exists child.\neg Doctor) \sqcup (\exists child.Lawyer)) \Rightarrow {}^{child}(Rich \sqcup Doctor)^{child} \quad \neg\text{-l} \\ \hline
\top^{child}, {}^{child}\neg((\exists child.\neg Doctor) \sqcup (\exists child.Lawyer)) \Rightarrow \forall child.(Rich \sqcup Doctor)^{child} \quad \forall\text{-r} \\ \hline
\top^{child}, \forall child.\neg((\exists child.\neg Doctor) \sqcup (\exists child.Lawyer)) \Rightarrow \forall child.(Rich \sqcup Doctor)^{child} \quad \forall\text{-l} \\ \hline
\top^{child}, \forall child.\neg((\exists child.\neg Doctor) \sqcup (\exists child.Lawyer)) \Rightarrow \exists child.\forall child.(Rich \sqcup Doctor) \quad \exists\text{-r} \\ \hline
\exists child.\top, \forall child.\neg((\exists child.\neg Doctor) \sqcup (\exists child.Lawyer)) \Rightarrow \exists child.\forall child.(Rich \sqcup Doctor) \quad \exists\text{-l} \\ \hline
\exists child.\top \sqcap \forall child.\neg((\exists child.\neg Doctor) \sqcup (\exists child.Lawyer)) \Rightarrow \exists child.\forall child.(Rich \sqcup Doctor) \quad \sqcap\text{-l}
\end{array}$$

which could be explained by: "(1) Doctors are Doctors or Rich (2) So, Everyone having all children Doctors has all children Doctors or Rich. (3) Hence, everyone either has at least a child that is not a doctor or every children is a doctor or rich. (4) Moreover, everyone is of the kind above, or, alternatively, have at least one child that is a lawyer. (5) In other words, if everyone has at least one child, then it has one child that has at least one child that is a lawyer, or at least one child that is not a doctor, or have all children doctors or rich. (6) Thus, whoever has all children not having at least one child not a doctor or at least one child lawyer has at least one child having every children doctors or rich."

The above explanation was build from top to bottom (toward the conclusion of the proof), by a procedure that tries to not repeat conjunctive particles (if - then, thus, hence, henceforth, moreover etc) to put together phrases derived from each subproof. In this case, phrase (1) come from weak-r, ⊔-r; phrase (2) come from prom-2; (3) is associated to weak-l, neg-r; (4) corresponds to weak-r, the two following ∃-r and the ⊓; (5) is associated to prom-1 and finally (6) corresponds to the remaining of the proof. The reader can note the large possibility of using endophoras in the construction of texts from structured proofs as the ones obtained by either $\mathcal{S}_{\mathcal{ALC}}$ or $\mathcal{S}^{[]}_{\mathcal{ALC}}$.

6 Conclusion and Further Work

In this article it is shown two sequent calculi for $\mathcal{ALC}$. Both are cut-free, and one ($\mathcal{S}^{[]}_{\mathcal{ALC}}$) is designed for implementation without the need of any backtracking resource. Moreover $\mathcal{S}^{[]}_{\mathcal{ALC}}$ provides counter-model whenever the subsumption candidate is not valid in $\mathcal{ALC}$. We briefly suggest how to use the structural feature of sequent calculus in favour of producing explanations in natural language from proofs. As it was said at the introduction, the use of the cut-rule can provide shorter proofs. The cut-rule may not increase the complexity of the explanation, since it simply may provide more structure to the original proof. With the help of the results reported in this article one has a solid basis to build mechanisms to provide shorter and good explanation for $\mathcal{ALC}$ subsumption in the context of a KB authoring environment. The inclusion of the cut-rule, however, at the implementation level, is a hard one. Presently, there are approaches to include analytical cuts in Tableaux, as far as we know there is no research on how to extend this to $\mathcal{ALC}$ Tableaux. This puts our results in advantage when taking explanations, and the size of the proofs as well, into account. There are also other techniques, besides the use of the cut-rule, to produce short proofs in the sequent calculus, see [5] and [10], that can be used in our context.

References

1. Schmidt-Schau, M., Smolka, G.: Attributive concept descriptions with complements. Artificial Intelligence 48(1), 1–26 (1991)
2. McGuinness, D.: Explaining Reasoning in Description Logics. PhD thesis, Rutgers University (1996)
3. Liebig, T., Halfmann, M.: Explaining subsumption in $\mathcal{ALEHF}_{R^+}$ tboxes. In: Horrocks, I., Sattler, U., Wolter, F. (eds.) Proc. of the 2005 International Workshop on Description Logics - DL 2005, Edinburgh, Scotland, pp. 144–151 (July 2005)
4. Deng, X., Haarslev, V., Shiri, N.: Using patterns to explain inferences in $\mathcal{ALCHI}$. Computational Intelligence 23(3), 386–406 (2007)
5. Gordeev, L., Haeusler, E., Costa, V.: Proof compressions with circuit-structured substitutions. In: Zapiski Nauchnyh Seminarov POMI (to appear, 2008)
6. Finger, M., Gabbay, D.: Equal Rights for the Cut: Computable Non-analytic Cuts in Cut-based Proofs. Logic Journal of the IGPL 15(5–6), 553–575 (2007)
7. Baader, F.: The Description Logic Handbook: theory, implementation, and applications. Cambridge University Press, Cambridge (2003)
8. Rademaker, A., do Amaral, F., Haeusler, E.: A Sequent Calculus for $\mathcal{ALC}$. Monografias em Ciência da Computação 25/07, Departamento de Informática, PUC-Rio (2007)
9. Rademaker, A., Haeusler, E., Pereira, L.: On the proof theory of $\mathcal{ALC}$. In: XV EBL-15th Brazilian Logic Conference (to appear)
10. Finger, M.: DAG sequent proofs with a substitution rule. In: Artemov, S., Barringer, H., dAvila Garcez, A., Lamb, L., Woods, J. (eds.) We will show Them – Essays in honour of Dov Gabbays 60th birthday, vol. 1, pp. 671–686. Kings College Publications, London (2005)