A New Mapping from $\mathcal{ALCI}$ to $\mathcal{ALC}$

Yu Ding and Volker Haarslev and Jiewen Wu

Concordia University, Montreal, Quebec, Canada
{ding_yu|haarslev|w_jiewen}@cse.concordia.ca

1 Introduction

It is well known that a *correspondence theory* for description logics (DLs), propositional modal logics (MLs) and propositional dynamic logics (PDLs) was given in [Sch91] and that an *axiomatic system* for a description logic with inverse roles was presented in the same paper. Schild's paper covers what is to be discussed in this paper although his paper did not mention *nominals*[1] as well as expressive *roles* explicitly. On the other hand, reductions to eliminate *converse programs* (*inverse roles* in DLs) are known even for full PDL ($\mathcal{ALCI}_{reg}$); the original reduction from converse PDL to PDL was in [Gia96]. Moreover, a direct tableaux method for converse PDL and further discussions on the elimination of *converse programs* were given in [GM00]. Besides *graded modalities*, it was pointed out in [Gia96] that the previous reduction technique is also applicable to *nominals*.

This paper shares some viewpoints well made in [Gia96]. The proposed mapping process here is based on the simple idea to capture possible back-propagation caused by the use of inverse roles[2]. This process consists of three steps, *tagging, recording, polarisation*, which are introduced below. Concept expressions/formulae are assumed to be in *negation norm form* (NNF). For simplicity, *existential restrictions* and *universal restrictions* are called *modal constraints* somewhere. We refer to [BCM$^+$03] for usual background knowledge on *description logics* (DLs).

2 Concept Satisfiability with General Concept Inclusions

Concept expressions/formulae are in NNF. For simplicity, we consider a GCI (*general concept inclusion*) of the form $\top \sqsubseteq C$, where C is in NNF. The following shows how to use three simple steps (*tagging, recording and polarisation*) to convert a concept formula (in NNF) of a source logic with *inverse roles* to a target logic without *inverse roles*. Terminological knowledge bases with general concept inclusions (GCIs) are also considered.

[1] Such viewpoint can be found in the early literature on *hybrid logics*, which are logics extending the *propositional modal logic* with *nominals* (a.k.a. named states).

[2] The proposed mapping technique also relies on the model properties of the description logics concerned in the paper.

Definition 1. (Tagging-1) *The tagging technique introduces new concept names for modal constraints of concept expressions and axioms. The function $tag(.)$ on a concept formula x is defined as:*
(1) if x is $C \sqcap D$, then $tag(x) = tag(C) \sqcap tag(D)$;
(2) if x is $C \sqcup D$, then $tag(x) = tag(C) \sqcup tag(D)$;
(3) if x is $\exists R.C$, then $tag(x) = \exists R.(tag(C))$;
(4) if x is $\forall R.C$, then $tag(x) = Q(x) \oplus \forall R.(tag(C))$;
(5) if x is $\top \sqsubseteq C$, then $tag(x) = \top \sqsubseteq tag(C)$;
(6) otherwise $tag(x) = x$.
where $Q(x)$ is a fresh name unique for each x; C, D are subformulae; the symbol $\oplus$ represents the conjunction operator in exactly the same way as $\sqcap$.

$\mathcal{U}(R)$ denotes tagged *universal constraints* of the form $Q(x) \oplus \forall R.(tag(C))$, where R is a role and $\mathcal{U}$ are sets indexed by roles. E_0/E_1 denotes the formulae before and after tagging; $\mathcal{K}_0/\mathcal{K}_1$ denotes the GCI before and after tagging. Let $\forall R.C$ be a subformula before tagging, we have after tagging:
($\star$) for $x = \forall R.C$, there is $Q(x) \oplus \forall R.tag(C) \in \mathcal{U}(R)$;

Definition 2. (Recording-1) *Initialize $\mathcal{K}_a = \emptyset$. For each set $\mathcal{U}(R)$ indexed by each role R, and for each element $(Q(x) \oplus \forall R.(tag(C))) \in \mathcal{U}(R)$, perform the operation: $\mathcal{K}_a = \mathcal{K}_a \cup \{\top \sqsubseteq tag(C) \sqcup \forall R^-.\neg Q(x)\}$.*

Definition 3. (Polarisation-1) *$Pol(x)$ is performed on the tagged input formula E_1 to get a polarized E_2, and on $\mathcal{K}_1 \cup \mathcal{K}_a$ to get the polarized $\mathcal{K}_2$:*
(1) if x is $C \sqcap D$, then $Pol(x) = Pol(C) \sqcap Pol(D)$;
(2) if x is $C \sqcup D$, then $Pol(x) = Pol(C) \sqcup Pol(D)$;
(3) if x is $\exists R.C$, then $Pol(x) = \exists R^a.Pol(C)$;
(4) if x is $\forall R.C$, then $Pol(x) = \forall R^a.Pol(C)$;
(5) if x is $\exists R^-.C$, then $Pol(x) = \exists R^b.Pol(C)$;
(6) if x is $\forall R^-.C$, then $Pol(x) = \forall R^b.Pol(C)$;
(7) if x is $\top \sqsubseteq C$, then $Pol(x) = \top \sqsubseteq Pol(C)$;
(8) otherwise, $Pol(x) = x$.
where R^a (R^b) is a fresh role name unique for R (R^-).

3 Concept Satisfiability/Abox Consistency with Tbox

The Tbox (a.k.a. terminological box) is a set of unfoldable axioms. The notion of Tbox is related some fundamental notions such as *name unfolding* and GCI *absorption* [BCM+03]. By *descriptive semantics*, *equality axioms* like $A \equiv C$ are expressed in two *inclusion axioms* $A \sqsubseteq C$ and $\neg A \sqsubseteq \neg C$. The right-hand-sides of the axioms are in `NNF`. An acyclic Tbox of only such inclusion axioms is called *simplified* [Lut99]. In the following, we show how to use the three simple steps (i.e., *tagging, recording, polarisation*) for Tboxes.

Definition 4. (Acyclic Ordering) *The ordering relation[3] is as following:*

[3] Due to acyclicity, $ord(A) \succ ord(A)$ is not induced.

(1) for each axiom $A \sqsubseteq C$, there is $ord(A) \succ ord(C)$;
(2) $ord(C \sqcap D) \succ ord(C)$ and $ord(C \sqcap D) \succ ord(D)$;
(3) $ord(C \sqcup D) \succ ord(C)$ and $ord(C \sqcup D) \succ ord(D)$;
(4) $ord(\exists R.C) \succ ord(C)$;
(5) $ord(\forall R.C) \succ ord(C)$;

An Abox consists of *concept assertions* and *role assertions*. If an Abox has several unconnected components, each of them can be treated alike separately. We assume one individual has at most one label because $d : C$ and $d : D$ can be replaced by $d : C \sqcap D$. W.l.o.g. we consider a single-component Abox $\mathcal{A}_0$ and each individual has at most one label. We denote the label for d_i as $\mathcal{L}(d_i)$.

Definition 5. (Tagging-2) *The function $tag(x)$ is:*
(1) if x is $A \sqsubseteq C$, then $tag(x) = A \sqsubseteq tag(C)$;
(2) if x is $\exists R.C$, then $tag(x) = P(x) \oplus \exists R.(tag(C))$;
(3) if x is individual d_i with $\mathcal{L}(d_i)$, then $tag(x) = d_i : P(x) \oplus tag(\mathcal{L}(d_i))$;
(4) if x is an individual d_i of no label, then $tag(x) = d_i : P(x)$;
(5) otherwise, call Tagging-1 for x.
where $P(x)$ is a unique name for each x, and the sign $\oplus$ stands for $\sqcap$.

The original Abox/Tbox are denoted as $\mathcal{A}_0/\mathcal{T}_0$, their tagged counterparts are denoted as $\mathcal{A}_1/\mathcal{T}_1$. Notice we do not tag any *role assertions*. We also write $P(d_i)$ instead of $P(x)$ if the tag is for an individual d_i. The set of tags for all individuals of the Abox is $\mathcal{D} = \{P(d_i) | d_i \in \mathcal{A}_0\}$. Let C denote any (sub)formula:
($\star$) for $x = \exists R.C$, there is $P(x) \oplus \exists R.tag(C) \in \mathcal{E}(R)$;
($\star$) for $y = \forall R.C$, there is $P(y) \oplus \forall R.tag(C) \in \mathcal{U}(R)$;
($\star$) for $z = d_i$, there is $P(d_i) \in \mathcal{D}$;

We additionally stipulates $ord(P(d_i)) \succ ord(tag(\mathcal{L}(d_j)))$ for any individual d_i and d_j. This forces $P(d_i)$ to get a higher order than $tag(\mathcal{L}(d_j))$ (and higher than subformulae of $tag(\mathcal{L}(d_j))$ but does not introduce cycles[4].

Definition 6. (Recording-2) *For two tuples $\beta \in \mathcal{U}(*)$ and $\alpha \in (\mathcal{E}(*) \bigcup \mathcal{U}(*) \bigcup \mathcal{D})$ where $*$ denotes any role name, if the following conditions are met:*
(1) $ord(\alpha) \succ ord(\beta)$; and
(2) $\alpha = P(x) \oplus \forall R_1.tag(C)$ or
$\alpha = P(x) \oplus \exists R_1.tag(C)$ or
$\alpha = P(x)$ and x is some Abox individual d_i; and
(3) $\beta = P(y) \oplus \forall R_2.tag(D)$;
then perform the operation: $\mathcal{T}_a = \mathcal{T}_a \cup \{P(x) \sqsubseteq \forall R_2^-.\neg P(y) \sqcup tag(D)\}$.

Definition 7. (Polarisation-2) *$Pol(x)$ is performed on the tagged Abox $\mathcal{A}_1$ to get $\mathcal{A}_2$, and on the augmented Tbox $\mathcal{T}_1 \cup \mathcal{T}_a$ to get $\mathcal{T}_2$:*
(1) if x is $A \sqsubseteq C$, then $Pol(x) = A \sqsubseteq Pol(C)$;

[4] Please notice $\succ$ is transitive. The extra requirement forces $ord(P(d_i) \oplus tag(\mathcal{L}(d_i))) \succ ord(P(d_i)) \succ ord(tag(\mathcal{L}(d_i)))$. For $i \neq j$, we have: (1) $ord(tag(\mathcal{L}(d_i)))$ and $ord(tag(\mathcal{L}(d_j)))$ are incomparable; (2) $ord(P(d_i))$ and $ord(P(d_j))$ are incomparable; (3) $ord(P(d_i)) \succ ord(tag(\mathcal{L}(d_j)))$.

(2) if x is $(c,d) : R \in \mathcal{A}_1$, then $\mathcal{A}_2 = \mathcal{A}_2 \cup \{(c,d) : R^a,\ (d,c) : R^b\}$;
(3) if x is $(c,d) : R^- \in \mathcal{A}_1$, then $\mathcal{A}_2 = \mathcal{A}_2 \cup \{(d,c) : R^a,\ (c,d) : R^b\}$;
(4) if x is $d_i : P(d_i) \oplus \mathcal{L}(d_i) \in \mathcal{A}_1$, then $\mathcal{A}_2 = \mathcal{A}_2 \cup \{d_i : P(d_i) \oplus Pol(\mathcal{L}(d_i))\}$;
(5) otherwise, call Polarisation-1.
where R^a (R^b) is a fresh role name unique for R (R^-).

Though in the above only acyclic Tboxes are emphasized, the exact mapping method also applies to cyclic Tboxes by simply dropping the acyclic ordering condition prescribed at the *recording* step.

4 Experiments

We have implemented the mapping as presented in Section 1 to evaluate its practicality. All satisfiability tests were performed with RacerPro 1.9.0 on a Pentium PC with 3.5 GB memory. The tested ontologies were also converted on the same machine. Note that the expressivity of the original ontologies is $\mathcal{ALCI}$.

KB Name	Coherence Check (original Tbox)	Conversion	Coherence Check (converted Tbox)
galen-ir1-alci-new1	9.141	50.657	84.625
galen-ir2-alci-new1	9.549	52.547	76.156
uml-no-max-min-new4	timeout after 1 hour	1.156	0.110
revised-9-alci (partial)	timeout after 20 mins	17.016	3.297

Table 1. Experimental results (all times are given in seconds)

KB Name	Num. of Axioms (original/converted)	Classes/Properties (original Tbox)	Classes/Properties (converted Tbox)
galen-ir1-alci-new1	4645/5495	3107/234	3597/228
galen-ir2-alci-new1	4666/5508	3107/234	3597/228
uml-no-max-min-new4	524/739	233/213	448/213
revised-9-alci (partial)	3077/3099	2427/56	2449/37

Table 2. KBs before and after conversion (number of axioms, classes and properties)

Table 1 shows some empirical results (coherence check only), where the time indicated is the average of 5 independent runs of the conversion system. It can been seen that although more time is spent for testing Tbox coherence for the converted versions of the first two KBs, the performance is still acceptable since the KB sizes after conversion are nearly five times of the original ones. Evidently, for the UML ontology the runtime after conversion is quite impressive. Besides, we have also divided the UML ontology into two sub-ontologies, both of which, if converted, require less time to compute the satisfiability of all the concepts. Dramatic increase of performance is shown in the last case, where the ontology contains one major class extracted from ontology "revised-9-alci".

5 Discussions

The question about whether a *decision procedure* that has to work both "forward" and "backward" could be implemented to run efficiently was raised long ago in the literature, among them we mention [Gia96]. Nonetheless, the highly optimized tableau-based reasoning systems (the 3rd generation [BCM$^+$03]) are convincingly found "to behave quite well" in practice for many realistic problems. As *description logics* are widely used in diverse application domains, different "application patterns" produce a lot of realistic problems that might not be quite "tractable" as previous ones in terms of "problem size" and "problem structure". Several application domains are known to easily give "practically intractable problems", e.g., the *model checking* field, probably due to their indistinct narrative styles, i.e., extensive use of tightly constrained constraints. Recently, it was even found that some small-size ontologies are "hard" enough to kill some best tableau-based DL systems currently available. The existence of latest "intractable realistic problems" is more baffling than any indistinct narrative style that people have seen before.

Schild has provided an axiomatic system [Sch91] for $\mathcal{ALCI}$, the DL extending the basic description logic $\mathcal{ALC}$ with *inverse roles*. A *correspondence theory* for description logics, propositional modal logics and propositional dynamic logics was also given in [Sch91]. In [Gia95] and [CGR98], the "converse elimination technique" was presented for the CPDL and $\mathcal{ALCI}_{reg}$. Their technique is more general than what is presented in this paper and was extended in various aspects. Important literature on "converse elimination" and a direct tableaux approach include [Gia96] and [GM00]. Their transformation leads to target problems in the ExpTime class. Their technique could possibly lead to good implementations in practice. However, it is not very clear if there was any empirical result about their elimination of converse for the so-called "realistic problems".

A worst-case optimal tableau procedure for testing concept satisfiability w.r.t. general Tbox was given in [DM00] for $\mathcal{ALC}$ in details. Their technique of *caching* intermediate results and nogoods has deep influence on tableau-based DL systems. The belief that description logics without inverse roles lend themselves better to optimisations (e.g. the caching technique) originates from [DM00] and is well supported from practice. Lutz discussed the complexity cliff phenomenon for the problem of concept satisfiability test w.r.t. an acyclic Tbox in several logics in [Lut99]. One of the results showed is a tableaux procedure that takes a polynomial space for concept satisfiability test w.r.t. an acyclic Tbox in $\mathcal{ALC}$. The *pre-completion technique* was proposed in [DLNS94] [Hol94] for reducing the Abox consistency problem to a number[5] of concept satisfiability tests to be carried out independently. For a pre-completion technique for $\mathcal{SH}$ Abox (which strictly contains the logic $\mathcal{ALC}$) w.r.t. Tbox, see [TG99].

The proposed mapping in this paper allows tableau-based decision procedures to safely use the *global sub-tableaux caching technique*. This gives a hope that the run-time performance should not be much worse than using the *dynamic block-*

[5] It is the exact number of Abox individuals.

ing and the *pseudo-model merging* techniques, two best optimisations currently available [BCM$^+$03] to tableau-based decision procedures for DLs with inverse roles. This conjecture is supported by our first-hand experiments. Further, the *pseudo-model merging* technique coexists with the *global sub-tableaux caching* technique. Also, the new axioms introduced in the recording step can be "selectively" used to simulate the well-known *tableau expansion rules* in such a way that if the well-known tableau algorithm (that allows bi-directional propagation of constraints) constructs a pre-model for the source problem, then this construction process can be repeated to construct a pre-model for the target problem at an equal cost. The only possible disadvantage of the proposed mapping is that it introduces extra concept names and extra axioms (with one disjunction per axiom). However it should be noted that these extra names and axioms are for "simulating" the well-known tableau expansion rules that rely on the *dynamic blocking technique*. Moreover, the newly recorded axioms are not necessarily GCIs but can always be unfoldable axioms (as shown in Section 3).

We require each role has a unique inverse role. For a role R, for example, we consider R^- as the only *inverse role*. This takes a linear cost. The presented transformation is *equisatisfiability preserving*, and is fine-grained in the sense that the target problems stay in the same complexity class as the source problems. The *recording* operation descends from the C-rule (the Ramsey-Rule) which states an equivalence of $\exists R.C \sqsubseteq D$ and $C \sqsubseteq \forall R^-.D$ [Ram31]. This equivalence was rediscovered in DLs and was lately used for new *absorption techniques*, for example [HW06] and [SGP06].

This paper largely follows and extends our previous work in [DH05]. In [DH05], we proposed three different ways to deal with $\mathcal{ALCI}$. The first was a *dynamic caching* technique that extends the *dynamic blocking* technique to work on different traces and thus allows *anywhere blocking*. The second was a *reachability analysis* to guarantee the soundness of the *global sub-tableaux caching* technique through a pre-compilation of a Tbox. The third was about the equivalence mentioned above. In this paper, rather than to enrich the *absorption technique* as previously perceived, we used the equivalence in a different and novel way. Here we have presented two versions of a mapping technique to deal with GCIs and (acyclic) Tboxes[6]. It also works for nominals and expressive roles. Moreover, the mapping without *polarisation* is quite interesting in itself for it brings a *back-propagation don't-care property* for the target problems (now in DLs with inverse roles). Here is a summary conclusion of the proposed mapping technique:

- every knowledge base in fragments of $\mathcal{SHOI}$ that contain $\mathcal{SHI}$ (or $\mathcal{ALCOI}$) can be converted to a unfoldable Tbox in the corresponding fragments containing $\mathcal{SH}$ (or $\mathcal{ALCO}$);
- every (acyclic) unfoldable Tbox in $\mathcal{ALCHI}$ (or its fragments) can be converted to an (acyclic) unfoldable Tbox in $\mathcal{ALCH}$ (or corres. fragments);
- the mapping is fine-grained in the sense that the target problems stay in the same complexity class as the source problems.

[6] The transformation is presented for the Abox consistency check problem w.r.t. (acyclic) Tboxes.

It is observed in our current experiments that some (not all) very hard ontologies the coherence of which could not even be tested are able to be classified in reasonable time. For optimisations of the classification, they are beyond the satisfiability test based (tableau-based) decision procedures. It is well known that classification could even be done without resorting to any satisfiability test at all, for example [BHN+92] [TH05]. Right now, we are preparing an optimized and extended implementation. An in-depth empirical analysis is under way. For a parallel work on a worst-case ExpTime (binary coding of numbers) tableau-based decision procedure for $\mathcal{ALCQI}$, a description logic containing both *qualified number restrictions* and *inverse roles*, see [DH07].

6 Acknowledgments

The authors thank referees for giving many insightful comments so that an improvement on the paper is possible; and thank Xi Deng for proof-reading the latest version. Also sincere thanks from the first author goes to Angel for helping a draft version before the Christmas season.

References

[BCM+03] Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications.* Cambridge University Press, 2003.

[BHN+92] Franz Baader, Bernhard Hollunder, Bernhard Nebel, Hans-Jürgen Profitlich, and Enrico Franconi. An Empirical Analysis of Optimization Techniques for Terminological Representation Systems, or Making KRIS Get a Move On. In *KR*, pages 270–281, 1992.

[CGR98] Diego Calvanese, Giuseppe De Giacomo, and Riccardo Rosati. A Note on Encoding Inverse Roles and Functional Restrictions in $\mathcal{ALC}$ Knowledge Bases. *CEUR Workshop DL-98*, 1998.

[DH05] Yu Ding and Volker Haarlsev. Towards Efficient Reasoning for Description Logics with Inverse Roles. *DL-Workshop'05*, 2005.

[DH07] Yu Ding and Volker Haarslev. An ExpTime Tableau-based Decision Procedure for $\mathcal{ALCQI}$. *DL-Workshop'07*, 2007.

[DLNS94] Francesco M. Donini, Maurizio Lenzerini, Daniele Nardi, and Andrea Schaerf. Deduction in Concept Languages: From Subsumption to Instance Checking. *J. Log. Comput.*, 4(4):423–452, 1994.

[DM00] Francesco M. Donini and Fabio Massacci. EXPTIME Tableaux for $\mathcal{ALC}$. *Artifical Intelligence*, 124(1):87–138, 2000.

[FL79] Michael J. Fischer and Richard E. Ladner. Propositional Dynamic Logic of Regular Programs. *Journal of Computer and System Sciences*, 18:194–211, 1979.

[Gia95] Giuseppe De Giacomo. *Decidability of Class-based Knowledge Representation Formalisms.* PhD thesis, Universita' di Roma "La Sapienza", Roma, Italy, 1995.

[Gia96] Giuseppe De Giacomo. Eliminating "Converse" from Converse PDL. *Journal of Logic, Language and Information*, 5(2):193–208, 1996.

[GM00] Giuseppe De Giacomo and Fabio Massacci. Combining Deduction and Model Checking into Tableaux and Algorithms for Converse-PDL. *Inf. Comput.*, 162(1-2):117–137, 2000.

[Hol94] B. Hollunder. *Algorithmic Foundations of Terminological Knowledge Representation Systems*. PhD thesis, Universit, at des Saarlandes, Germany, 1994.

[HW06] Alexander K. Hudek and Grant Weddell. Binary Absorption in Tableaux-Based Reasoning for Description Logics. *DL-Workshop'06*, 2006.

[Lut99] Carsten Lutz. Complexity of Terminological Reasoning Revisited. *LPAR-1999*, 99:181–200, 1999.

[Ram31] F. Plank Ramsey. *Truth and Probability*. In Foundations of Mathematics and Other Logical Essays, R.B. Braithwaite, ed., New York., 1931.

[Sav70] Walter J. Savitch. Relationships between Nondeterministic and Deterministic Tape Complexities. *J. of Computer and System Sci.*, 4(2):177–192, 1970.

[Sch91] Klaus Schild. A Correspondence Theory for Terminological Logics: Preliminary Report. *IJCAI*, pages 466–471, 1991.

[SGP06] Evren Sirin, Bernardo Cuenca Grau, and Bijan Parsia. From wine to water: Optimizing description logic reasoning for nominals. In Patrick Doherty, John Mylopoulos, and Christopher A. Welty, editors, *KR*, pages 90–99. AAAI Press, 2006.

[Tes01] Sergio Tessaris. *Questions and Answers: Reasoning and Querying in Description Logic*. PhD thesis, Universit of Manchester, Manchester, UK, 2001.

[TG99] Sergio Tessaris and Graham Gough. Abox Reasoning with Transitive Roles and Axioms. *DL WorkShop'99*, 1999.

[TH05] Dmitry Tsarkov and Ian Horrocks. Optimised Classification for Taxonomic Knowledge Bases. In Ian Horrocks, Ulrike Sattler, and Frank Wolter, editors, *Description Logics*, volume 147 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2005.

A One Working Example

Give a concept $E_0 = \exists s^-.C_2$, and a Tbox $\mathcal{K}_0$ of the following nine general axioms in logic $\mathcal{ALCI}$:

$(a_1)\top \sqsubseteq \neg C_1 \sqcup \exists q.C_2$ $\quad$ $(a_2)\top \sqsubseteq \neg C_2 \sqcup \forall s.C_4$
$(a_3)\top \sqsubseteq \neg C_4 \sqcup \exists p.C_3$ $\quad$ $(a_4)\top \sqsubseteq \neg C_3 \sqcup \exists s^-.C_2$
$(a_5)\top \sqsubseteq \neg C_2 \sqcup \forall s.C_5$ $\quad$ $(a_6)\top \sqsubseteq \neg C_5 \sqcup \forall p^-.C_6$
$(a_7)\top \sqsubseteq \neg C_6 \sqcup \forall p^-.C_7$ $\quad$ $(a_8)\top \sqsubseteq \neg C_7 \sqcup \forall s^-.C_8$
$(a_9)\top \sqsubseteq \neg C_8 \sqcup \forall q^-.C_9$

Source Problem: the satisfiability of E_0 w.r.t. $\mathcal{K}_0$ in $\mathcal{ALCI}$.

Step-1: perform tagging. For each a_i, there is $a'_i = tag(a_i)$.
$(a'_1)\top \sqsubseteq \neg C_1 \sqcup \exists q.C_2$ (no change from a_1)
$(a'_2)\top \sqsubseteq \neg C_2 \sqcup (A_1 \oplus \forall s.C_4)$
$(a'_3)\top \sqsubseteq \neg C_4 \sqcup \exists p.C_3$ (no change from a_3)
$(a'_4)\top \sqsubseteq \neg C_3 \sqcup \exists s^-.C_2$ (no change from a_4)

$(a_5')\top \sqsubseteq \neg C_2 \sqcup (A_2 \oplus \forall s.C_5)$
$(a_6')\top \sqsubseteq \neg C_5 \sqcup (A_3 \oplus \forall p^-.C_6)$
$(a_7')\top \sqsubseteq \neg C_6 \sqcup (A_4 \oplus \forall p^-.C_7)$
$(a_8')\top \sqsubseteq \neg C_7 \sqcup (A_5 \oplus \forall s^-.C_8)$
$(a_9')\top \sqsubseteq \neg C_8 \sqcup (A_6 \oplus \forall q^-.C_9)$

The tagging operation changes nothing for axioms a_1, a_3 and a_4. A_j are newly introduced tags for occurrences of universal constraints. Now, $\mathcal{K}_1 = \{a_i'\}$; the tagged concept is $E_1 = tag(E_0) = \exists s^-.C_2$.

Step-2: perform recording. Each A_j has an axiom r_j.

$(r_1)\top \sqsubseteq \forall s^-.\neg A_1 \sqcup C_4$ $(r_2)\top \sqsubseteq \forall s^-.\neg A_2 \sqcup C_5$
$(r_3)\top \sqsubseteq \forall p.\neg A_3 \sqcup C_6$ $(r_4)\top \sqsubseteq \forall p.\neg A_4 \sqcup C_7$
$(r_5)\top \sqsubseteq \forall s.\neg A_5 \sqcup C_8$ $(r_6)\top \sqsubseteq \forall q.\neg A_6 \sqcup C_9$

Now, $\mathcal{K}_a = \{r_j\}$.

Step-3: perform polarisation. Notice $\mathcal{K}_2 = Poly(\mathcal{K}_1 \cup \mathcal{K}_a)$. Accordingly, we have $c_i = Poly(a_i')$ and $d_j = Poly(r_j)$ as following.

$(c_1)\top \sqsubseteq \neg C_1 \sqcup \exists q^a.C_2$
$(c_2)\top \sqsubseteq \neg C_2 \sqcup (A_1 \oplus \forall s^a.C_4)$
$(c_3)\top \sqsubseteq \neg C_4 \sqcup \exists p^a.C_3$
$(c_4)\top \sqsubseteq \neg C_3 \sqcup \exists s^b.C_2$
$(c_5)\top \sqsubseteq \neg C_2 \sqcup (A_2 \oplus \forall s^a.C_5)$
$(c_6)\top \sqsubseteq \neg C_5 \sqcup (A_3 \oplus \forall p^b.C_6)$
$(c_7)\top \sqsubseteq \neg C_6 \sqcup (A_4 \oplus \exists p^b.C_7)$
$(c_8)\top \sqsubseteq \neg C_7 \sqcup (A_5 \oplus \forall s^b.C_8)$
$(c_9)\top \sqsubseteq \neg C_8 \sqcup (A_6 \oplus \forall q^a.C_9)$
$(d_1)\top \sqsubseteq \forall s^b.\neg A_1 \sqcup C_4$ $(d_2)\top \sqsubseteq \forall s^b.\neg A_2 \sqcup C_5$
$(d_3)\top \sqsubseteq \forall p^a.\neg A_3 \sqcup C_6$ $(d_4)\top \sqsubseteq \forall p^a.\neg A_4 \sqcup C_7$
$(d_5)\top \sqsubseteq \forall s^a.\neg A_5 \sqcup C_8$ $(d_6)\top \sqsubseteq \forall q^a.\neg A_6 \sqcup C_9$

We get $E_2 = Poly(E_1) = Poly(\exists s^-.C_2) = \exists s^b.C_2$; and $\mathcal{K}_2 = \{c_i\} \cup \{d_j\}$ as listed above. The newly introduced roles $\{s^a, p^a, q^a, s^b, p^b, q^b\}$ replace the old roles $\{s, p, q, s^-, p^-, q^-\}$. Replace $\oplus$ with $\sqcap$ and we get the problem in $\mathcal{ALC}$.

Target problem: the satisfiability of E_2 w.r.t. $\mathcal{K}_2$ in $\mathcal{ALC}$.

B Proofs

Definition 8. (Fischer-Ladner Closure) *The Fischer-Ladner closure[FL79] $FL(H)$ of a set of formulae H is the least set of formulae which is inductively generated as follows:*
(1) H is a subset of $FL(H)$; *(2) if $C \in FL(H)$, then so is $\neg C$;*
(3) if $C \sqcap D \in FL(H)$, then so are C and D;
(4) if $C \sqcup D \in FL(H)$, then so are C and D;
(5) if $\exists R.C \in FL(H)$, then so is C; *(6) if $\forall R.C \in FL(H)$, then so is C.*

To denote a *modal constraint*, we use $\substack{\exists \\ \forall}R.tag(C)$. The $tag(.)$ operation (recursively) converts one *modal constraint* to a conjunction having two conjuncts.

In each tagged constraint we stipulate that the tag is on the left and the *modal constraint* is on the right. This excludes cases like $\overset{\exists}{\forall} R.tag(C) \sqcap Q(x)$.

Definition 9. *Let $Q(x) \sqcap \overset{\exists}{\forall} R.tag(C)$ be a tagged constraint in the Tbox $\mathcal{KT}$ and the formula E. Consider the Fischer-Ladner closure $FL(\mathcal{KT} \cup \{E\})$, a generating formula for $\overset{\exists}{\forall} R.tag(C)$ is a formulae α such that $FL(\{\alpha\}) \supset FL(\{\overset{\exists}{\forall} R.tag(C)\})$, and $\alpha \notin \{\overset{\exists}{\forall} R.tag(C), \neg(\overset{\exists}{\forall} R.tag(C))\}$.*

Definition 10. *The generating formula α for $\overset{\exists}{\forall} R.tag(C)$ is least if there exists no other generating formula β s.t. $FL(\{\beta\}) \subset FL(\{\alpha\})$.*

A few comments are necessary: (1) Each tag $Q(x)$ is unique and occurs only as a conjunct; its negation $\neg Q(x)$ occurs only as a disjunct; (2) $Q(x) \sqcap \overset{\exists}{\forall} R.tag(C)$ and $\neg(Q(x) \sqcap \overset{\exists}{\forall} R.tag(C))$ are two least generating formulae for $\overset{\exists}{\forall} R.tag(C)$. We call the former *positive* and the latter *negative*; (3) The formulae have already been converted into NNF before performing the mapping; (4) The tagging operation assigns a set of unique tags for one modal constraint and that a least generating formula for the tag $Q_i(x)$ will also be a least generating formula for the tagged modal constraint; (5) Only *positive generating formulae* need to be considered[BCM+03] when building a model[7]. These lead to the notion[8] of p-model.

Definition 11. *Given E and $\mathcal{KT}$ that are tagged formula and Tbox. Let $Q_1(x) \sqcap \overset{\exists}{\forall} R.tag(C)$, ..., $Q_m(x) \sqcap \overset{\exists}{\forall} R.tag(C)$ be tagged constraints. A p-model for E and $\mathcal{KT}$ is a model $(\Delta^{\mathcal{I}}, .^{\mathcal{I}})$ such that for any $n \in \Delta^{\mathcal{I}}$ there are*
(1) if $n \in (Q_i(x))^{\mathcal{I}}$ then $n \in (\overset{\exists}{\forall} R.tag(C))^{\mathcal{I}}$; and
(2) if $n \in (\overset{\exists}{\forall} R.tag(C))^{\mathcal{I}}$, then there is $Q_i(x)$ for $\overset{\exists}{\forall} R.tag(C)$ s.t. $n \in (Q_i(x))^{\mathcal{I}}$.

Lemma 1. *Given E and $\mathcal{KT}$ as the tagged formula and the tagged Tbox. Let $Q_i(x) \sqcap \overset{\exists}{\forall} R.tag(C)$ be tagged constraints in E and $\mathcal{KT}$, where $i = 1, ..., m$. E and $\mathcal{KT}$ is satisfiable only if it is satisfiable in a p-model.*

Proof. Only proof outline. Let $M_1 = (\Delta^{\mathcal{I}_1}, .^{\mathcal{I}_1})$ be any model for E and $\mathcal{KT}$.

We use the well-known *sub-model generating technique*[9] to get a p-model from M_1. To guide the extraction process, it is only necessary to consider the *positive generating formulae.* To be precise, for any $n \in \Delta^{\mathcal{I}_1}$, the extraction process ignores assertions like $n \in (\neg(Q_i \sqcap \overset{\exists}{\forall} R.tag(C)))^{\mathcal{I}_1}$, where $\overset{\exists}{\forall} R.tag(C)$ is a tagged constraints with a unique set of tags $\{Q_1, Q_2, ..., Q_m\}$.

Starting from a (root) node with $\mathcal{L}(n) \supseteq \{E\} \cup \mathcal{KT}$, a guided extraction process (focusing on *positive least generating formulae*) generates a sub-model

[7] This is true for $\mathcal{ALCI}$ and expressive logics without the qualified number restrictions.

[8] Consider a multi-valued function from modal constraints to tags $f(\overset{\exists}{\forall} R.tag(C)) = \{Q_1, ..., Q_m\}$ s.t. $f(x) \cap f(y) = \emptyset$ for $x \neq y$. The model of interest is such a model $(\Delta^{\mathcal{J}}, .^{\mathcal{J}})$ that $\forall n \in \Delta^{\mathcal{J}}$: (i) for all $1 \leq i \leq m$, $(Q_i)^{\mathcal{J}} \subseteq (\overset{\exists}{\forall} R.tag(C))^{\mathcal{J}}$; and (ii) if $n \in (\overset{\exists}{\forall} R.tag(C))^{\mathcal{J}}$, then $n \in Q_i^{\mathcal{J}}$ for some $1 \leq i \leq m$.

[9] It has also been extensively used in the literature in the completeness proof of certain tableau-calculus for DLs with inverse roles.

from M_1. It is verifiably a model by routinely showing/checking it is saturated and clash-free, and it meets p-model definition. So, there is a p-model $(\Delta^{\mathcal{I}_2}, .^{\mathcal{I}_2})$ provided that $(\Delta^{\mathcal{I}_1}, .^{\mathcal{I}_1})$ be a model (which is saturated and has no clash). □

Lemma 2. *Given a concept formula E' and a Tbox $\mathcal{KT}'$. Let the tagged formula and Tbox be E and $\mathcal{KT}$. E' is satisfiable w.r.t. $\mathcal{KT}'$ iff E is satisfiable w.r.t. $\mathcal{KT}$.*

Proof. The proof uses the same *sub-model generating technique* as above and a mapping from the $\overset{\exists}{\forall}R.tag(C)$ to $Q_i \sqcap \overset{\exists}{\forall}R.tag(C)$ and vice versa.

From the p-model (out of the above sub-model generating) for E and $\mathcal{KT}$, taking away (negated) tags leads to a model for E' and $\mathcal{KT}'$ (having no tags).

In the other direction, for any model of E' and $\mathcal{KT}'$, a mapping reflecting the tag operation (from the $\overset{\exists}{\forall}R.tag(C)$ to $Q_i \sqcap \overset{\exists}{\forall}R.tag(C)$ for some Q_i) will lead to a model (interpreting tags) for E and $\mathcal{KT}$. Since a p-model is a model, this concludes that the tagging operation preserves satisfiability. □

B.1 Concept Satisfiability Test with General Concept Inclusions

Lemma 3. *if E_1 and $\mathcal{K}_1$ has a p-model, then $\mathcal{K}_a$ is satisfiable in that model.*

Proof. This follows from the definition of p-model $Q(x)^{\mathcal{I}} \subseteq (\forall R.tag(C))^{\mathcal{I}}$. □

Lemma 4. *E_1 is satisfiable w.r.t. $\mathcal{K}_1 \cup \mathcal{K}_a$ iff E_1 is satisfiable w.r.t. $\mathcal{K}_1$.*

Proof. (Only If Direction) It is trivial.

(If Direction) Let $\mathcal{M}_2$ a p-model for E_1 and $\mathcal{K}_1$. According to the lemma above, $\mathcal{K}_a$ is always satisfied in the p-model for both E_1 and $\mathcal{K}_1$. It follows that $\mathcal{M}_2$ is a model for E_1 and $\mathcal{K}_1 \cup \mathcal{K}_a$. □

Lemma 5. *E_1 is satisfiable w.r.t. $\mathcal{K}_1 \cup \mathcal{K}_a$ iff E_2 is satisfiable w.r.t. $\mathcal{K}_2$.*

Proof. Note $E_2 = Pol(E_1)$ and $\mathcal{K}_2 = Pol(\mathcal{K}_1 \cup \mathcal{K}_a)$.

(If Direction) Let $\mathcal{M}_2 = (\Delta^{\mathcal{I}_2}, .^{\mathcal{I}_2})$ be a p-model (possibly non-tree) for E_2 and $\mathcal{K}_2$. For $m', n' \in \Delta^{\mathcal{I}_2}$, consider a mapping to $m, n \in \Delta^{\mathcal{I}_1}$ such that

(1) if $(m', n') \in (R^a)^{\mathcal{I}_2}$, then $(m, n) \in R^{\mathcal{I}_1}$;

(2) if $(m', n') \in (S^b)^{\mathcal{I}_2}$, then $(m, n) \in (S^-)^{\mathcal{I}_1}$;

(3) if $m', n' \in Poly(C)^{\mathcal{I}_2}$, then $m, n \in C^{\mathcal{I}_1}$.

(Only If Direction) Let $\mathcal{M}_1 = (\Delta^{\mathcal{I}_1}, .^{\mathcal{I}_1})$ be a p-model (possibly non-tree) for E_1 and $\mathcal{K}_1$. For $m, n \in \Delta^{\mathcal{I}_1}$, consider a mapping that maps them to $m', n' \in \Delta^{\mathcal{I}_2}$

(1) if $(m, n) \in R^{\mathcal{I}_1}$, then $(m', n') \in (R^a)^{\mathcal{I}_2}$ and $(n', m') \in (R^b)^{\mathcal{I}_2}$;

(2) if $(m, n) \in (S^-)^{\mathcal{I}_1}$, then $(m', n') \in (S^b)^{\mathcal{I}_2}$ and $(n', m') \in (S^a)^{\mathcal{I}_2}$;

(3) if $m, n \in C^{\mathcal{I}_1}$ then $m', n' \in Poly(C)^{\mathcal{I}_2}$.

R, S^- are roles in $\mathcal{ALCI}$; R^a, R^b, S^a, S^b are roles in $\mathcal{ALC}$.

Note the definition of p-models and the special axioms acquired by the recording operation. In both directions, at each element of the target interpretation, all constraints are satisfied both locally and w.r.t. its neighbor elements provided the given $(\Delta^{\mathcal{I}_k}, .^{\mathcal{I}_k})$ be a p-model. This concludes that the polarisation operation preserves equisatisfiability (for a tagged and recorded problem). □

Lemma 6. *$\|E_2\| + \|\mathcal{K}_2\|$ is of $O(n^2)$ where $n = \|E_0\| + \|\mathcal{K}_0\|$.*

Theorem 1. *(1) E_0 is satisfiable w.r.t. $\mathcal{K}_0$ iff E_2 is satisfiable w.r.t. $\mathcal{K}_2$. (2)The satisfiability of E_0 w.r.t. $\mathcal{K}_0$ in $\mathcal{ALCI}$ can be decided by a test of E_2 w.r.t. $\mathcal{K}_2$ in $\mathcal{ALC}$. (3) The concept satisfiability w.r.t. GCIs in $\mathcal{ALCI}$ can be decided in exponential time by the tableaux procedure in $\mathcal{ALC}$ [DM00].*

B.2 Concept Satisfiability/Abox Consistency with (Acyclic) Tbox

Regarding to acyclic Tboxes in DLs without inverse roles, we refer to [Lut99] and [Tes01] for their results. We list supporting theorems and lemmas. The proofs are similar to the ones previously done for general axioms.

Theorem 2. (Acyclic ALC Tbox[Lut99]) *The concept satisfiability w.r.t. an acyclic Tbox in $\mathcal{ALC}$ is decidable in PSPACE by a tableau-based procedure.*

Lemma 7. *For an input concept E_0 and an acyclic Tbox $\mathcal{T}_0$ in $\mathcal{ALCI}$, by tagging-recording-polarisation the concept E_2 and the acyclic Tbox $\mathcal{T}_2$ in $\mathcal{ALC}$ is of size $O(n^3)$, where $n = ||E_0|| + ||\mathcal{T}_0||$.*

Proof. (outline only) By a similar step-by-step proof[10] (as in the case of general axioms), it is able to show E_2 and $\mathcal{T}_2$ is equisatisfiable to E_0 and $\mathcal{T}_0$. The acyclicity of $\mathcal{T}_2$ is easy to verify. The number of combinations (formulae α, β s.t. $ord(\alpha) \succ ord(\beta)$) is at most n^2 and each new axiom is of size at most n. □

Lemma 8. *The concept satisfiability problem w.r.t. an acyclic Tbox in $\mathcal{ALCI}$ can be decided in PSPACE by tableau procedures as given in [Lut99].*

Lemma 9. *For Abox Consistency, $\mathcal{T}_2$ is acyclic. The conversion (a streamlined tagging, recording and polarisation operations) takes a polynomial space.*

Lemma 10. *The consistency of an Abox w.r.t. an acyclic Tbox in $\mathcal{ALCI}$ can be decided by the consistency check of an Abox w.r.t. an acyclic Tbox in $\mathcal{ALC}$.*

Theorem 3. (Precompletion[Tes01]) *A precompletion of an Abox w.r.t. a Tbox can be nondeterministically computed in a polynomial space; the size of each precompletion is polynomial bounded.*

Theorem 4. *The consistency of an Abox w.r.t. an acyclic Tbox in $\mathcal{ALCI}$ can be decided in PSPACE by tableau-based decision procedures.*

Proof. (1) Perform the conversion to get the target problem, which is of a polynomial size to the source problem. The conversion itself takes a polynomial space; (2) Nondeterministically compute one precompletion. Then individuals of the target Abox are subject to satisfiability tests by the PSPACE procedure in [Lut99] w.r.t. the target Tbox independently. If each individual of the target Abox is satisfiable, the source problem is consistent. The Abox consistency problem w.r.t. an acyclic Tbox is decidable in a nondeterministic polynomial space. Consider Savitch's theorem[Sav70]. This ends the proof[11]. □

---

[10] Proofs of equisatisfiability do not use acyclicity or properties of Aboxes in our case. These properties are however needed in proofs of particular decision procedures that are PSPACE and they are taken into account by previous work [Lut99] [Tes01]. We cite those results as theorems.

[11] Similar arguments were extensively used for PSPACE tableaux in the literature.