Unions of Conjunctive Queries in $\mathcal{SHOQ}$

Birte Glimm and **Ian Horrocks**
University of Oxford, UK

Ulrike Sattler
The University of Manchester, UK

Abstract

Conjunctive queries play an important role as an expressive query language in Description Logics (DLs). Decision procedures for expressive Description Logics are, however, only recently emerging and it is still an open question whether answering conjunctive queries is decidable for the DL $\mathcal{SHOIQ}$ that underlies the OWL DL standard. In fact, no decision procedure was known for expressive DLs that contain nominals. In this paper, we close this gap by providing a decision procedure for entailment of unions of conjunctive queries in $\mathcal{SHOQ}$. Our algorithm runs in deterministic time single exponential in the size of the knowledge base and double exponential in the size of the query, which is the same as for $\mathcal{SHIQ}$. Our procedure also shows that $\mathcal{SHOQ}$ knowledge base consistency is indeed ExpTime-complete, which was, to the best of our knowledge, always conjectured but never proved.

Introduction

Description Logics (Baader et al. 2003) are a well-established family of logic-based knowledge representation formalisms that have gained increased attention due to their usage as the logical underpinning of ontology languages such as OWL (Horrocks, Patel-Schneider, and van Harmelen 2003). A DL knowledge base consists of a TBox, which contains intensional knowledge such as concept definitions and general background knowledge, and an ABox, which contains extensional knowledge and is used to describe individuals. Using a database metaphor, the TBox corresponds to the schema, and the ABox corresponds to the data. Contrary to a typical database setting, the open world assumption is usually made in (Description) Logics, which means we have only incomplete knowledge about the modeled domain and models can be infinite.

In data-intensive applications, querying knowledge bases plays a central role. *Instance retrieval* is a basic reasoning task that involves the retrieval of all *certain* instances of a given (possibly complex) concept C, i.e., it returns all individuals from the ABox that are an instance of C in every model of the knowledge base. Technically, instance retrieval is well-understood. For the prominent DL $\mathcal{SHIQ}$, which underlies OWL Lite, it is ExpTime-complete (Tobies 2001) and for $\mathcal{SHOIQ}$, which underlies OWL DL, it is NExpTime-complete (Tobies 2001). Despite this high worst-case complexity, efficient implementations are available.[1] Instance retrieval provides, however, only limited forms of querying: concepts (roles) are used as queries, and thus we can only query for (pairs of) individual names and for tree-like structures that are invariant under (guarded) bisimulations. For this reason, *conjunctive query answering* has been suggested as a stronger form of querying, i.e., computing the certain answers to a conjunctive query over a knowledge base (Calvanese, De Giacomo, and Lenzerini 1998).

Recently, it has been shown that conjunctive query answering is decidable in $\mathcal{SHIQ}$ and, thus, in OWL Lite (Glimm et al. 2008; 2007; Calvanese, Eiter, and Ortiz 2007). To the best of our knowledge, it is, however, still an open problem whether this is also the case for $\mathcal{SHOIQ}$ and, thus, OWL DL. In this paper, we make an important step in this direction by presenting an algorithm for unions of conjunctive queries over $\mathcal{SHOQ}$, i.e., $\mathcal{SHOIQ}$ without inverse roles. More precisely, we devise a decision procedure for *entailment* of unions of conjunctive queries by a $\mathcal{SHOQ}$ knowledge base, where conjunctive query entailment is the decision problem corresponding to conjunctive query answering. It is well-known that decidability and complexity results carry over from entailment to answering (Calvanese, Eiter, and Ortiz 2007; Glimm et al. 2007; Glimm 2007). Our decision procedure is inspired by the query rewriting algorithm for $\mathcal{SHIQ}$ (Glimm et al. 2008), but is adapted to handle the more complicated relational structures that nominals can express. In the query rewriting process, we reduce a conjunctive query to (possibly several) $\mathcal{SHOQ}^\sqcap$-concepts, i.e., $\mathcal{SHOQ}$-concepts with role conjunctions. Conjunctive query entailment is then reduced to consistency checking of $\mathcal{SHOQ}^\sqcap$ knowledge bases. For this task, we devise an automata-based algorithm that runs in deterministic double exponential time in the size of the query and single exponential time in the size of the knowledge

[1] http://www.cs.man.ac.uk/~sattler/reasoners.html

base. This result concerns the combined complexity, i.e., it is measured in the size of the knowledge base and the query. For $\mathcal{SHIQ}$, the same upper bound holds and it is known to be tight (Lutz 2007).

For full proofs and more detailed definitions of the presented results we refer to (Glimm 2007).

Preliminaries

A signature is a tuple (N_C, N_R, N_I), where N_C is a set of *concept names*, N_R a set of *role names*, and N_I a set of *individual names*. The set of role names contains a subset $N_{tR} \subseteq N_R$ of *transitive role names*. A *role inclusion* is of the form $r \sqsubseteq s$ with $r, s \in N_R$. A *role hierarchy* $\mathcal{R}$ is a finite set of role inclusions. A role r is *simple* w.r.t. a role hierarchy $\mathcal{R}$ if there is no transitive role $s \in N_{tR}$ such that $s \overset{*}{\sqsubseteq}_{\mathcal{R}} r \in \mathcal{R}$, where $\overset{*}{\sqsubseteq}_{\mathcal{R}}$ denotes the reflexive transitive closure of $\sqsubseteq$ w.r.t. $\mathcal{R}$.

An *interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ consists of a non-empty set $\Delta^{\mathcal{I}}$, the *domain* of $\mathcal{I}$, and a function $\cdot^{\mathcal{I}}$, which maps every concept name $A \in N_C$ to a subset $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$, every role name $r \in N_R$ to a binary relation $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$, and every individual name $a \in N_I$ to an element $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$. An interpretation $\mathcal{I}$ *satisfies* a role inclusion $r \sqsubseteq s$ if $r^{\mathcal{I}} \subseteq s^{\mathcal{I}}$, and a role hierarchy $\mathcal{R}$ if it satisfies all role inclusions in $\mathcal{R}$.

$\mathcal{SHOQ}$-concepts (or concepts for short) are built inductively using the following grammar, where $o \in N_I$, $A \in N_C$, $n \in \mathbb{N}$, $r \in N_R$, and $s \in N_R$ is a simple role:

$$C ::= \ A \mid \{o\} \mid \neg C \mid C_1 \sqcap C_2 \mid \forall r.C \mid \geqslant n\ s.C.$$

We use the following standard abbreviations: $C_1 \sqcup C_2 \equiv \neg(\neg C_1 \sqcap \neg C_2)$, $\exists r.C \equiv \neg(\forall r.(\neg C))$, and $\leqslant n\ s.C \equiv \neg(\geqslant (n+1)\ s.C)$.

The semantics of $\mathcal{SHOQ}$-concepts is defined as usual:

$$\{o\}^{\mathcal{I}} = \{o^{\mathcal{I}}\},\ (C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}},\ (\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}},$$
$$(\forall r.C)^{\mathcal{I}} = \{d \in \Delta^{\mathcal{I}} \mid \text{ if } (d, d') \in r^{\mathcal{I}}, \text{ then } d' \in C^{\mathcal{I}}\},$$
$$(\geqslant n\ s.C)^{\mathcal{I}} = \{d \in \Delta^{\mathcal{I}} \mid \sharp(s^{\mathcal{I}}(d, C)) \geq n\}$$

where $\sharp(M)$ denotes the cardinality of the set M and $s^{\mathcal{I}}(d, C) = \{d' \in \Delta^{\mathcal{I}} \mid (d, d') \in s^{\mathcal{I}} \text{ and } d' \in C^{\mathcal{I}}\}$.

The presented algorithm reduces a conjunctive query to concepts that may also contain role conjunctions in the place of role names. A *role conjunction* is an expression $r_1 \sqcap \ldots \sqcap r_n$, where $r_1, \ldots, r_n$ are role names. The interpretation function is extended to role conjunctions as follows: $(r_1 \sqcap \ldots \sqcap r_n)^{\mathcal{I}} = {r_1}^{\mathcal{I}} \cap \ldots \cap {r_n}^{\mathcal{I}}$. A role conjunction is *simple* if each role occurring in it is simple. We can then build complex $\mathcal{SHOQ}^{\sqcap}$-concepts with the same grammar as above, just with r (s) being a (simple) role conjunction.

A *general concept inclusion* (GCI) is an expression $C \sqsubseteq D$, where both C and D are concepts. A finite set of GCIs is called a *TBox*. An *assertion* is an expression of the form $A(a), \neg A(a), r(a, b)$, $\neg r(a, b)$, or $a \not\approx b$ with $A \in N_C$, $r \in N_R$, and $a, b \in N_I$. An *ABox* is a finite set of assertions. Since, in the presence of nominals, the ABox can be internalized (e.g., $A(a)$ is equivalent to the GCI $\{a\} \sqsubseteq A, r(a, b)$ to $\{a\} \sqsubseteq \exists r.\{b\}$, etc.), we assume w.l.o.g. that a $\mathcal{SHOQ}^{\sqcap}$ knowledge base $\mathcal{K}$ is a pair $(\mathcal{T}, \mathcal{R})$ where $\mathcal{T}$ is a TBox and $\mathcal{R}$ is a role hierarchy. We use $\mathsf{rol}(\mathcal{K})$ for the set of role names used in $\mathcal{K}$ and $\mathsf{nom}(\mathcal{K})$ for the set of individual names (nominals) that occur in $\mathcal{K}$. We assume that $\mathsf{nom}(\mathcal{K})$ is non-empty. This is w.l.o.g. since we can always add an axiom $\{o\} \sqsubseteq \top$ to $\mathcal{T}$ for a fresh nominal $o \in N_I$.

An interpretation $\mathcal{I}$ *satisfies* a GCI $C \sqsubseteq D$ if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ and it *satisfies* a TBox if it satisfies each GCI in it. An interpretation $\mathcal{I}$ is a *model* of a knowledge base $\mathcal{K} = (\mathcal{T}, \mathcal{R})$, denoted as $\mathcal{I} \models \mathcal{K}$, if it satisfies $\mathcal{T}$ and $\mathcal{R}$. A knowledge base is *consistent* if it has a model.

As usual, we use $\mathsf{nnf}(C)$ to denote the negation normal form of a concept C. We define the *closure* $\mathsf{cl}(\mathcal{K})$ of $\mathcal{K}$ as the smallest set containing $\mathsf{nnf}(\neg C \sqcup D)$ if $C \sqsubseteq D \in \mathcal{T}$; D if D is a sub-concept of C and $C \in \mathsf{cl}(\mathcal{K})$; and $\mathsf{nnf}(\neg C)$ if $C \in \mathsf{cl}(\mathcal{K})$.

Let N_V be a countably infinite set of variables with $v, v' \in N_V$ and let $(N_C,\ N_R,\ N_I)$ be a signature with $A \in N_C, r \in N_R$. A Boolean *conjunctive query* q is a non-empty set of atoms, where an atom is an expression $A(v)$, $r(v, v')$, or $v \approx v'$. We refer to these three types of atoms as concept, role, and equality atoms respectively. We use $\mathsf{Vars}(q)$ to denote the set of variables occurring in q. A *sub-query* of q is simply a subset of q (including q itself). A *union of Boolean conjunctive queries* is an expression $q_1 \vee \ldots \vee q_\ell$, where each disjunct q_i is a Boolean conjunctive query.

Please note that we do not allow for constants (individual names) to occur in the position of variables. This is w.l.o.g. since our DL contains nominals: for each constant a in q, we introduce a new variable x_a, replace each occurrence of a with x_a, and add a concept atom $(\{a\})(x_a)$.

Since equality is reflexive, symmetric and transitive, we define the equivalence relation $\overset{*}{\approx}$ as the transitive, reflexive, and symmetric closure of $\approx$ over the variables in q. We define the relation $\bar{\in}$ over atoms in q as follows: $A(v)\ \bar{\in}\ q$ if there is a variable $v' \in \mathsf{Vars}(q)$ such that $v \overset{*}{\approx} v'$ and $A(v') \in q$ and similarly for role atoms.

Let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ be an interpretation. For a total function $\pi\colon \mathsf{Vars}(q) \to \Delta^{\mathcal{I}}$, we write (i) $\mathcal{I} \models^{\pi} A(v)$ if $\pi(v) \in A^{\mathcal{I}}$; (ii) $\mathcal{I} \models^{\pi} r(v, v')$ if $(\pi(v), \pi(v')) \in r^{\mathcal{I}}$; and (iii) $\mathcal{I} \models^{\pi} v \approx v'$ if $\pi(v) = \pi(v')$. If $\mathcal{I} \models^{\pi} at$ for all atoms $at \in q$, we write $\mathcal{I} \models^{\pi} q$. We say that $\mathcal{I}$ *satisfies* q and write $\mathcal{I} \models q$ if there exists a π such that $\mathcal{I} \models^{\pi} q$. We call such a π a *match* for q in $\mathcal{I}$. Let $\mathcal{K}$ be a $\mathcal{SHOQ}$ knowledge base and q a conjunctive query. If, for every interpretation $\mathcal{I}$, $\mathcal{I} \models \mathcal{K}$ implies $\mathcal{I} \models q$, we say that $\mathcal{K}$ *entails* q and write $\mathcal{K} \models q$. $\mathcal{K}$ *entails* a union of conjunctive queries $q_1 \vee \ldots \vee q_\ell$, written as $\mathcal{K} \models q_1 \vee \ldots \vee q_\ell$, if, for each model $\mathcal{I}$ of $\mathcal{K}$, there is some i with $1 \leq i \leq \ell$ such that $\mathcal{I} \models q_i$.

The *size* of a knowledge base $\mathcal{K}$ (a query q), denoted $|\mathcal{K}|$ ($|q|$), is simply the number of symbols needed to write it over the alphabet of constructors, concept, role, individual, and variable names that occur in $\mathcal{K}$ (q). We assume unary coding of numbers in number restrictions.

Canonical Models

We first show that we can restrict our attention to interpretations that have a kind of forest shape. Since $\mathcal{SHOQ}$ allows for nominals, the forest structure is not directly obvious, but we can use models with an underlying "forest backbone" called *forest base.* These forest bases are forest-shaped interpretations that interpret transitive roles in an unrestricted way, i.e., not necessarily in a transitive way. In a forest base, each element can be related to its direct successor nodes and to some root/nominal nodes. Due to the allowed relations from any element to nominal nodes, even forest bases are strictly speaking not forests.

Definition 1. *Let $\mathbb{N}^*$ be the set of all (finite) words over the alphabet $\mathbb{N}$. A* tree *T is a non-empty, prefix-closed subset of $\mathbb{N}^*$. The empty word ε is called the* root *of T. For $w, w' \in T$, we call w' a* successor *of w if $w' = w \cdot c$ for some $c \in \mathbb{N}$, where "$\cdot$" denotes concatenation. The* branching degree *$d(w)$ of a node $w \in T$ is the number of its successors. If there is a k such that $d(w) \leq k$ for each $w \in T$, we say that T has branching degree k. Given a set of elements $\rho = \{o_1, \ldots, o_n\}$, a* forest *$F$* w.r.t. *$\rho$ is a subset of $\rho \times \mathbb{N}^*$ such that, for each $o_i \in \rho$, $(o_i, \varepsilon) \in F$ and the set $\{w \mid (o_i, w) \in F\}$ is a tree.*

Let $\mathcal{K}$ be a $\mathcal{SHOQ}$ knowledge base. A forest base *for $\mathcal{K}$ is an interpretation $\mathcal{J} = (\Delta^{\mathcal{J}}, \cdot^{\mathcal{J}})$ that interprets transitive roles in an unrestricted (i.e., not necessarily transitive) way and, additionally, satisfies the following conditions:*

F1 *$\Delta^{\mathcal{J}}$ is a forest w.r.t. $\mathsf{nom}(\mathcal{K})$;*

F2 *if $((o, w), (o', w')) \in r^{\mathcal{J}}$, then either $w' = \varepsilon$ or $o = o'$ and w' is a successor of w;*

F3 *for each $o \in \mathsf{nom}(\mathcal{K})$, $o^{\mathcal{J}} = (o, \varepsilon)$.*

An interpretation $\mathcal{I}$ is canonical *for $\mathcal{K}$ if there exists a forest base $\mathcal{J}$ for $\mathcal{K}$ such that $\mathcal{I}$ is identical to $\mathcal{J}$ except that, for all non-simple roles r, we have*

$$r^{\mathcal{I}} = r^{\mathcal{J}} \cup \bigcup_{s \,\sqsubseteq^*_{\mathcal{R}} r,\; s \in N_{tR}} (s^{\mathcal{J}})^+,$$

where $^+$ denotes the transitive closure operator. In this case, we say that $\mathcal{J}$ is a forest base *for $\mathcal{I}$ and, if $\mathcal{I} \models \mathcal{K}$, we say that $\mathcal{I}$ is a* canonical model *for $\mathcal{K}$. If, for each $o \in \mathsf{nom}(\mathcal{K})$, the branching degree of the tree $\{w \mid (o, w) \in \Delta^{\mathcal{I}}\}$ is bounded by some k, we say that $\mathcal{I}$ has branching degree k.*

For simplicity we make the unique name assumption (UNA) in the above definition (cf. 1), i.e., for each $a, b \in N_I$ such that $a \neq b$ and each interpretation $\mathcal{I}$, we assume that $a^{\mathcal{I}} \neq b^{\mathcal{I}}$. This is w.l.o.g. as we can guess an appropriate partition among the individual names and replace the individual names in each partition with one representative individual name from that partition. Hence, our decision procedure does not rely on the UNA and this assumption does not affect the complexity results.

We use the following running example throughout this paper.

Example 2. *Let $\mathcal{K} = (\mathcal{T}, \mathcal{R})$ be a $\mathcal{SHOQ}$ knowledge base with $t, t' \in N_{tR}$*

$$\begin{aligned} \mathcal{T} = \{ \; & \{o\} \sqsubseteq \exists t.(C \sqcap \exists r.(\exists r.(D \sqcap \exists t.(\{o\})))) \\ & \{o'\} \sqsubseteq \exists s.\top \sqcap \exists s.(\{o\})\} \\ \mathcal{R} = \{ \; & \;\; r \;\; \sqsubseteq t'\} \end{aligned}$$

and $q = \{C(x), D(z), t'(x, z), t(z, x), r(x, y), r(y, z)\}$ with $\mathsf{Vars}(q) = \{x, y, z\}$.

Figure 1 shows a graphical representation of a canonical model for $\mathcal{K}$. Without the dashed lines, the figure would show the forest base for this canonical model. In what follows, we sometimes informally use the term *shortcut.* A shortcut is a generalization of the case where a forest base $\mathcal{J}$ and its canonical model $\mathcal{I}$ differ, i.e., where $(a, b) \in r^{\mathcal{I}}$ but $(a, b) \notin r^{\mathcal{J}}$.

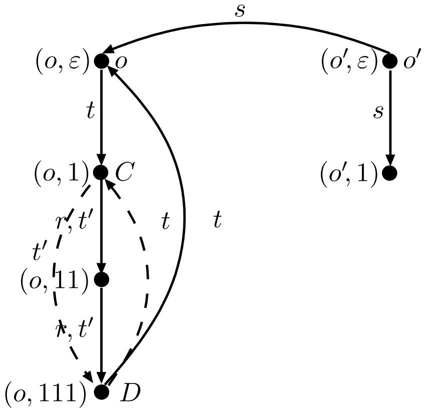

Figure 1: A representation of a canonical interpretation $\mathcal{I}$ for $\mathcal{K}$. The transitive shortcuts are shown as dashed lines; without them, the figure would show a representation of a forest base for $\mathcal{I}$.

Figure 2 shows a representation of the query from our running example together with a match for the canonical model. We have that $\mathcal{I} \models^{\pi} q$ for $\pi \colon \{x \mapsto (o, 1), y \mapsto (o, 11), z \mapsto (o, 111)\}$ and that also $\mathcal{K} \models q$.

The following lemma justifies our focus on canonical models. The bound on the branching degree is important for our automata based decision procedure.

Lemma 3. *Let q be a union of Boolean conjunctive queries, $\mathcal{K}$ a $\mathcal{SHOQ}$ knowledge base with $|\mathcal{K}| = m$, n_{max} the maximal number occurring in number restrictions, and $k = m \cdot n_{max}$. $\mathcal{K} \not\models q$ iff there is some canonical model $\mathcal{I}$ of $\mathcal{K}$ such that $\mathcal{I} \not\models q$ and $\mathcal{I}$ has branching degree k.*

Informally, for the only if direction, we can take an arbitrary counter-model for the query, which exists by assumption, and "unravel" all non-tree structures. During the unravelling process, we use a choice function that chooses, for each existential and each at-least number restriction occurring in the closure of $\mathcal{K}$, the required successors. This guarantees that the branching degree is indeed k. Since, during the unraveling process, we only replace cycles in the model with infinite paths

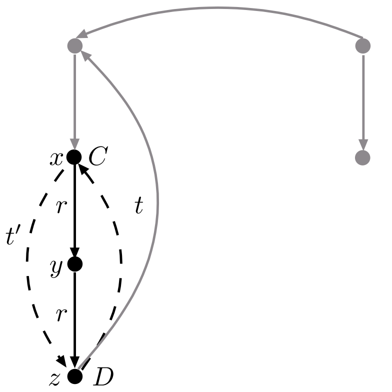

Figure 2: A graphical representation of the query q with a match in the canonical model (shown in grey, and without labels).

and leave the interpretation of concepts from the closure unchanged, the query is still not satisfied in the unravelled canonical model. The if direction of the proof holds vacuously.

Query Rewriting

In order to decide whether a union of conjunctive queries q is entailed by a $\mathcal{SHOQ}$ knowledge base $\mathcal{K}$, we transform each disjunct of q in a four stage process into a set of $\mathcal{SHOQ}^{\sqcap}$-concepts. We then show that we can use these concepts to build extensions of $\mathcal{K}$ in such a way that $\mathcal{K}$ entails q iff all the extensions are inconsistent.

In the first rewriting step, called *collapsing*, we can identify variables. Consider, for example, the cyclic query $q = \{r(x, y), r(x, y'), s(y, z), s(y', z)\}$, which can be transformed into a tree-shaped one by adding the equality atom $y \approx y'$. The collapsing step is not new and already required for simpler logics (see, e.g., (Horrocks et al. 2000)). With $\mathsf{co}(q)$ we denote the set of all collapsings that can be obtained from q.

A common property of the next two rewriting steps is that they allow for substituting (implicit) shortcut edges with (explicit) paths that imply the shortcut. The steps aim at different cases in which these shortcuts can occur. The second step is called *nominal rewriting* and we can replace role atoms of the form $r(v, v')$ for which r is non-simple with two role atoms by possibly introducing a fresh variable. This allows for explicating all (transitive relations) that bypass a nominal. In this step, we also "guess", for each of the rewritten queries, which variables correspond to nominals. In the third step, called *shortcut rewriting*, we explicate shortcuts within a tree by replacing a role atom with a non-simple role with up to $\sharp(q)$ role atoms that use a transitive sub-role. In the fourth step, we filter out those queries that still cannot be expressed as $\mathcal{SHOQ}^{\sqcap}$ concepts. These queries are trivially false since their structure cannot be mapped to the canonical models of the knowledge base. The remaining queries are transformed into concepts by applying the rolling-up technique (Calvanese, De Giacomo, and Lenzerini 1998; Tessaris 2001). Finally, we use the obtained concepts to reduce conjunctive query entailment to knowledge base consistency checking.

Since the first query rewriting step, collapsing, is not new our running example focuses on the other rewriting steps. For the nominal rewriting step, we can choose to replace the conjunct $t(z, x)$, which bypasses the nominal node (o, ε) for the given mapping π and canonical model $\mathcal{I}$, with $t(z, x_r), t(x_r, x)$ for $x_r \in N_V$ a fresh variable. We call this nominal rewriting q_{nr} (see Figure 3).

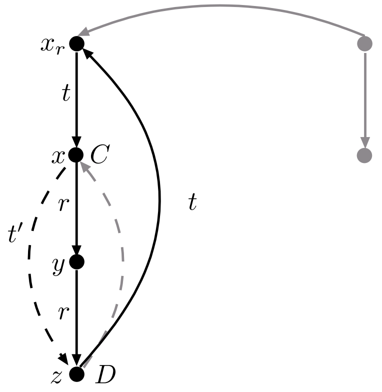

Figure 3: A graphical representation of the nominal rewriting q_{nr} for q. The canonical model $\mathcal{I}$ is shown in grey (without labels).

Since t is transitive, $\mathcal{I} \models^{\pi_{nr}} q_{nr}$ where π_{nr} is the extension of π that maps x_r to (o, ε). It is not hard to check that $\mathcal{K} \models q_{nr}$ implies $\mathcal{K} \models q$ since the role used in the rewriting must be transitive. At the end of the nominal rewriting step, we also "guess", for each of the rewritten queries, which variables correspond to nominals. With $\mathsf{nr}_{\mathcal{K}}(q)$, we denote the set of all pairs (q_{nr}, ρ) such that q_{nr} is a nominal rewriting of some collapsing $q_{co} \in \mathsf{co}(q)$ and $\rho \subseteq \mathsf{Vars}(q_{nr})$. The pair $(q_{nr}, \{x_r\})$ would, for example, belong to the set $\mathsf{nr}_{\mathcal{K}}(q)$.

In the shortcut rewriting step, we explicate shortcuts within a tree, such as $t'(x, z)$ in our running example, by replacing role atoms with a non-simple role with up to $\sharp(q)$ role atoms that use a transitive sub-role. In our running example, we can replace $t'(x, z)$ with $t'(x, y), t'(y, z)$ and we refer to the resulting query as q_{sr}. The mapping $\pi_{sr} = \pi_{nr}$ for q_{sr} no longer uses any shortcuts in $\mathcal{I}$ (see Figure 4) and we consider this query as forest-shaped w.r.t. the root choice $\{x_r\}$.

In the fourth and last step, we transform the obtained forest-shaped queries into concepts by applying the rolling-up technique. For example, the tuple $(q_{sr}, \{x_r\}, \tau)$ with a grounding τ that maps x_r to o results in the concept:

$$\{o\} \sqcap \exists t.(C \sqcap \exists(r \sqcap t').(\exists(r \sqcap t').(D \sqcap \exists t.\{o\})))$$

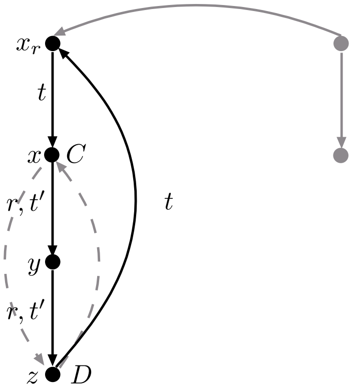

Figure 4: A graphical representation of the shortcut rewriting q_{sr} for q_{nr}. The canonical model $\mathcal{I}$ is shown in grey (without labels).

Finally, we use the concepts from all possible rewritings and reduce the task of deciding query entailment to the task of deciding knowledge base consistency by augmenting $\mathcal{K}$ with an axiom of the form $\top \sqsubseteq \neg C_q$ for each obtained concept C_q. If both $\mathcal{K}$ and the extended knowledge base are satisfiable, then $\mathcal{K} \not\models q$ because there is a model $\mathcal{I}$ of $\mathcal{K}$ that does not satisfy q. If both the extended knowledge base is unsatisfiable, then $\mathcal{K} \models q$ because every model $\mathcal{I}$ of $\mathcal{K}$ satisfies q and, thus, violates one of the axioms $\top \sqsubseteq \neg C_q$ in the extended knowledge base.

We now give a precise definition of the rewriting steps. Please note that we assume that q is a conjunctive query and not a union of conjunctive queries since we apply the rewriting steps to each disjunct separately.

Definition 4. *Let $\mathcal{K}$ be a $\mathcal{SHOQ}$ knowledge base and q a Boolean conjunctive query. A* collapsing *q_{co} of q is obtained by adding zero or more equality atoms of the form $v \approx v'$ for $v, v' \in \mathsf{Vars}(q)$ to q. We use $\mathsf{co}(q)$ to denote the set of all queries that are a collapsing of q.*

A nominal rewriting *q_{nr} of q and $\mathcal{K}$ is obtained by choosing, for each role atom $r(v, v') \in q$ such that there is a role $s \in N_{tR}$ and $s \sqsubseteq^*_{\mathcal{R}} r$ to either do nothing or to replace $r(v, v')$ with $s(v, v'')$ and $s(v'', v')$ for $v'' \in N_V$ a possibly fresh variable. We use $\mathsf{nr}_{\mathcal{K}}(q)$ to denote the set of pairs (q_{nr}, ρ) such that q_{nr} is a nominal rewriting of a query $q_{co} \in \mathsf{co}(q)$ and ρ is a subset of $\mathsf{Vars}(q_{nr})$. We call ρ a* root choice *for q_{nr}.*

A shortcut rewriting *of q and $\mathcal{K}$ is obtained from q by replacing each role atom $t(v_1, v_n)$ from q for which there is a sequence $r_1(v_1, v_2), \ldots, r_{n-1}(v_{n-1}, v_n) \mathrel{\bar{\in}} q$ and a role $s \in N_{tR}$ such that $s \sqsubseteq^*_{\mathcal{R}} t$ with $n-1$ role atoms $s(v_1, v_2), \ldots, s(v_{n-1}, v_n)$. We use $\mathsf{sr}_{\mathcal{K}}(q)$ to denote the set of all pairs (q_{sr}, ρ) such that there is a pair $(q_{nr}, \rho) \in \mathsf{nr}_{\mathcal{K}}(q)$ and q_{sr} is a shortcut rewriting of q_{nr}.*

We assume that $\mathsf{nr}_{\mathcal{K}}(q)$ contains no isomorphic queries, which are queries that differ only in newly introduced variable names.

A conjunctive query q is *tree-shaped* if there is a total function f from $\mathsf{Vars}(q)$ to a tree such that f is bijective modulo $\overset{*}{\approx}$ and $r(x, y) \mathrel{\bar{\in}} q$ implies that $f(y)$ is a successor of $f(x)$. A conjunctive query q is *forest-shaped w.r.t. a root choice ρ* if there is a total function f from $\mathsf{Vars}(q)$ to a forest F w.r.t. ρ such that f is bijective modulo $\overset{*}{\approx}$ and, for each $r(x, y) \in q$ with $f(x) = (x_r, w)$, either $f(y) = (y_r, \varepsilon)$ with $y_r \in \rho$ or $f(y) = (x_r, w \cdot c)$ for $c \in \mathbb{N}$. We set $\mathsf{fr}_{\mathcal{K}}(q) = \{(q_{fr}, \rho) \mid (q_{fr}, \rho) \in \mathsf{sr}_{\mathcal{K}}(q)$ and either $\rho = \emptyset$ and q_{fr} is tree-shaped or q_{fr} is forest-shaped w.r.t. $\rho\}$.

We now build a query that consists only of concept atoms for queries in $\mathsf{fr}_{\mathcal{K}}(q)$ by replacing the variables from the root choice ρ with nominals from $\mathsf{nom}(\mathcal{K})$ and applying the rolling-up technique. Please note that, although we say concept atoms, we actually allow queries in this intermediate step that contain complex concepts.

Definition 5. *Let $(q_{fr}, \rho) \in \mathsf{fr}_{\mathcal{K}}(q)$. A* grounding *for q_{fr} w.r.t. ρ is a total function $\tau\colon \rho \to \mathsf{nom}(\mathcal{K})$ such that, for all $v, v' \in \rho$, $\tau(v) = \tau(v')$ iff $v \overset{*}{\approx} v'$. We build $\mathsf{con}(q_{fr}, \rho, \tau)$ as follows:*

1. *For each $r(v, v_r) \mathrel{\bar{\in}} q_{fr}$ with $v_r \in \rho$, replace $r(v, v_r)$ with $(\exists r.\{\tau(v_r)\})(v)$.*
2. *For each $v_r \in \rho$, add a concept atom $(\{\tau(v_r)\})(v_r)$ to q_{fr}.*
3. *Let q be the result of applying 1 and 2.*
4. *We now inductively assign, to each $v \in \mathsf{Vars}(q)$ a concept $\mathsf{con}(v)$ as follows:*
   - *if there is no role atom $r(v, v') \mathrel{\bar{\in}} q$, then $\mathsf{con}(v) = \bigsqcap_{C(v) \mathrel{\bar{\in}} q} C$,*
   - *otherwise, let $r(v, v_1), \ldots, r(v, v_k)$ be all role atoms of the form $r(v, x) \mathrel{\bar{\in}} q$, then*

$$\mathsf{con}(v) = \bigsqcap_{C(v) \mathrel{\bar{\in}} q} C \sqcap \bigsqcap_{1 \le i \le k} \exists (\bigsqcap_{r(v, v_i) \mathrel{\bar{\in}} q} r).\mathsf{con}(v_i).$$

5. *Finally, $\mathsf{con}(q_{fr}, \rho, \tau) = \{(\mathsf{con}(v))(v) \mid v \in \mathsf{Vars}(q)$ and there is no role atom $r(v', v) \mathrel{\bar{\in}} q\}$.*

We use $\mathsf{con}_{\mathcal{K}}(q)$ for the set $\{\mathsf{con}(q_{fr}, \rho, \tau) \mid (q_{fr}, \rho) \in \mathsf{fr}_{\mathcal{K}}(q)$ and τ is a grounding w.r.t. $\rho\}$.

Please note that, after the first step, the resulting query consists of a set of unconnected components such that each component is a tree-shaped query with a distinguished root variable. This root variable need not necessarily belong to the root choice ρ. In Step 4, we collect all query concepts for these root variables in the set $\mathsf{con}(q_{fr}, \rho, \tau)$. Hence $\mathsf{con}(q_{fr}, \rho, \tau)$ is a conjunctive query of the form $\{C_1(v_1), \ldots, C_n(v_n)\}$ with $v_i \neq v_j$ for $1 \le i < j \le n$ and each C_i is a $\mathcal{SHOQ}^{\sqcap}$-concept. For a union of conjunctive queries $q = q_1 \vee \ldots \vee q_\ell$, we define $\mathsf{con}_{\mathcal{K}}(q)$ as $\mathsf{con}_{\mathcal{K}}(q_1) \cup \ldots \cup \mathsf{con}_{\mathcal{K}}(q_\ell)$. The following theorem shows that the queries in $\mathsf{con}_{\mathcal{K}}(q)$ can be used to decide entailment of q and that there is a bound on the cardinality of this set. We can then use the standard methods for deciding entailment of tree-shaped conjunctive queries in order to decide entailment of arbitrary conjunctive queries in $\mathcal{SHOQ}$.

Theorem 6. *Let q be a union of Boolean conjunctive queries, $\mathcal{K} = (\mathcal{T}, \mathcal{R})$ a $\mathcal{SHOQ}$ knowledge base, and $\mathsf{con}_{\mathcal{K}}(q) = \{q_1, \ldots, q_\ell\}$. Then (1) $\mathcal{K} \models q$ iff $\mathcal{K} \models q_1 \vee \ldots \vee q_\ell$.*

Due to space limitations, we just give a sketch of the proof for the above theorem. For the if direction: for the collapsing step, it is clear that, if a model $\mathcal{I}$ of $\mathcal{K}$ entails the collapsing, then it entails the query. Apart from the collapsing step, we replace only role atoms over non-simple roles with sequences of role atoms that use one of the transitive sub-roles. It is, therefore, not hard to see that, if a model $\mathcal{I}$ of $\mathcal{K}$ satisfies a rewritten query, then it satisfies the original query. In the rolling-up step, the use of nominal concepts (e.g., in concepts of the form $\exists r.(\{o\})$) enforces the required co-references for the links back to nominal nodes. The remaining tree-shaped parts can then straightforwardly be expressed as concepts. As for $\mathcal{SHIQ}$, we can then show that a model of the knowledge base that satisfies such a rolled-up query concept, also satisfies the query. For the only if direction: we can, by Lemma 3, restrict our attention to the canonical models of the knowledge base. We can then use any of the canonical models as a "guide" for the rewriting process as we used the given canonical model in our running example. If the canonical model satisfies the query, the rewriting steps can be applied in such a way that we obtain a forest-shaped query as required.

By carefully analysing the definition of the rewriting steps, we get the following bounds on the size of the rewritten queries and the number of rewritings.

Lemma 7. *Let q be a Boolean union of conjunctive queries, $\mathcal{K} = (\mathcal{T}, \mathcal{R})$ a $\mathcal{SHOQ}$ knowledge base, and $\mathsf{con}_{\mathcal{K}}(q) = \{q_1, \ldots, q_\ell\}$. Then the size of each $q_i \in \mathsf{con}_{\mathcal{K}}(q)$ is polynomial in $|q|$ and the cardinality of $\mathsf{con}_{\mathcal{K}}(q)$ is at most polynomial in $|\mathcal{K}|$ and exponential in $|q|$.*

Please note that each query $q_i \in \mathsf{con}_{\mathcal{K}}(q)$ is a set of concept atoms of the form $\{C_i^1(x_1), \ldots, C_i^n(x_n)\}$, i.e., each q_i contains n unconnected components. By transforming the disjunction $q_1 \vee \ldots \vee q_\ell$ of queries in $\mathsf{con}_{\mathcal{K}}(q)$ into conjunctive normal form (cf. (Tessaris 2001, 7.3.2)), we can reduce the problem of deciding whether $\mathcal{K} \models q_1 \vee \ldots \vee q_\ell$ to deciding whether $\mathcal{K}$ entails each union of connected conjunctive queries $\{at_1\} \vee \ldots \vee \{at_\ell\}$ where each at_i is a concept atom from q_i. Let $\mathsf{con}_{\mathcal{K}}(q) = \{q_1, \ldots, q_\ell\}$. We use $\mathsf{cnf}(\mathsf{con}_{\mathcal{K}}(q))$ for the conjunctive normal form of $q_1 \vee \ldots \vee q_\ell$. We now show how we can decide entailment of unions of conjunctive queries, where each atom consists of one concept atom only. This suffices to decide conjunctive query entailment for $\mathcal{SHOQ}$.

Definition 8. *Let $\mathcal{K} = (\mathcal{T}, \mathcal{R})$ be a $\mathcal{SHOQ}$ knowledge base, q a union of Boolean conjunctive queries, and $C_1(v_1) \vee \ldots \vee C_\ell(v_\ell)$ a query from $\mathsf{cnf}(\mathsf{con}_{\mathcal{K}}(q))$. An* extended knowledge base w.r.t. $\mathcal{K}$ and q *is a pair $(\mathcal{T} \cup \mathcal{T}_q, \mathcal{R})$ such that $\mathcal{T}_q = \{\top \sqsubseteq \neg C_i\}$ with $1 \leq i \leq \ell$.*

We can now use the extended knowledge bases in order to decide conjunctive query entailment:

Theorem 9. *Let $\mathcal{K}$ be a $\mathcal{SHOQ}$ knowledge base and q a union of Boolean conjunctive queries. Then $\mathcal{K} \models q$ iff each extended knowledge base $\mathcal{K}_q$ w.r.t. $\mathcal{K}$ and q is inconsistent.*

By again carefully analysing the definition of the extended knowledge bases and by using the results from Lemma 7, we get the following bounds on the size of and the number of extended knowledge bases.

Lemma 10. *Let $\mathcal{K}$ be a $\mathcal{SHOQ}$ knowledge base and q a union of Boolean conjunctive queries. The size of each extended knowledge base is at most polynomial in $|\mathcal{K}|$ and exponential in $|q|$ and the number of extended knowledge bases w.r.t. $\mathcal{K}$ and q is at most exponential in $|\mathcal{K}|$ and double exponential in $|q|$.*

$\mathcal{SHOQ}^\sqcap$ Knowledge Base Consistency

We now present our automata based decision procedure for $\mathcal{SHOQ}^\sqcap$ knowledge base consistency. By Theorem 9 this gives us a decision procedure for entailment of unions of conjunctive queries in $\mathcal{SHOQ}$.

Eliminating Transitivity

Since automata cannot directly handle transitive roles, we first transform a $\mathcal{SHOQ}^\sqcap$ knowledge base $\mathcal{K}$ into an equisatisfiable $\mathcal{ALCHOQ}^\sqcap$ knowledge base $\mathsf{et}(\mathcal{K})$. In the presence of role conjunctions *and* nominals, it does not suffice to extend the standard encoding of transitivity (see, e.g., (Kazakov and Motik 2006)) in a trivial way. Such a naive extension would result in an $\mathcal{ALCHOQ}^\sqcap$ knowledge base $\mathsf{et}(\mathcal{K})$ that is obtained from $\mathcal{K}$ by treating all transitive roles as non-transitive and by adding an axiom

$$\forall(r_1 \sqcap \ldots \sqcap r_n).C \sqsubseteq \forall(t_1 \sqcap \ldots \sqcap t_n).(\forall(t_1 \sqcap \ldots \sqcap t_n).C)$$

for each concept $\forall(r_1 \sqcap \ldots \sqcap r_n).C \in \mathsf{cl}(\mathcal{K})$ and roles $t_1, \ldots, t_n \in N_{tR}$ such that $t_i \sqsubseteq^*_{\mathcal{R}} r_i$ for $1 \leq i \leq n$. The following example shows that such an encoding does not yield an equisatisfiable knowledge base. Let $\mathcal{K} = (\mathcal{T}, \mathcal{R})$ be a $\mathcal{SHOQ}$ knowledge base with

$$\begin{aligned} \mathcal{T} = \{&\{o\} \sqsubseteq \exists t.(\exists t.(\exists t.(\{o'\}))), \\ &\{o\} \sqsubseteq \exists r.(\{o'\})\}, \end{aligned}$$

$\mathcal{R} = \emptyset$, and t a transitive role. Figure 5 shows a representation of a model for $\mathcal{K}$, where the grey edge represents the role r and the black edges represent the role t. The dashed black lines represent implicit (due to transitivity) instances of t. It is not hard to check that adding the axiom $\{o\} \sqsubseteq \forall(r \sqcap t).(\neg\{o'\})$ makes the knowledge base inconsistent.

The trivial encoding $\mathsf{et}(\mathcal{K})$ of $\mathcal{K}$ contains (among others) the additional axiom $\forall t.(\neg\{o'\}) \sqsubseteq \forall t.(\forall t.(\neg\{o'\}))$ since $\exists t.(\{o'\}) \in \mathsf{cl}(\mathcal{K})$ and, thus, $\forall t.(\neg\{o'\}) \in \mathsf{cl}(\mathcal{K})$. Adding the axiom $\{o\} \sqsubseteq \forall(r \sqcap t).(\neg\{o'\})$ to $\mathcal{K}$ does not yield any further axioms in $\mathsf{et}(\mathcal{K})$ since r is a simple role. Since none of the added axioms explicates the

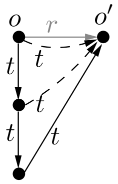

Figure 5: A representation of a model for $\mathcal{K}$.

implicit t relation between the nominals o and o', extending $\mathcal{K}$ with the axiom $\{o\} \sqsubseteq \forall(r \sqcap t).(\neg\{o'\})$ yields a consistent knowledge base after applying the translation, contradicting our assumption that a knowledge base $\mathcal{K}$ is consistent iff $\mathsf{et}(\mathcal{K})$ is consistent.

Intuitively, this problem arises since we can have arbitrary relations between nominals. This can lead to situations where, as in the above example, we have an explicit relationship between two nominals, but only together with the implicit transitive shortcut can the universal quantifier over the role conjunction be applied. Hence, the above described encoding does not suffice. We propose, therefore, a more involved encoding that explicates all transitive shortcuts between nominals.

Definition 11. *Let $\mathcal{K} = (\mathcal{T}, \mathcal{R})$ be a $\mathcal{SHOQ}^\sqcap$ knowledge base. The function $\mathsf{et}(\mathcal{K})$ yields the $\mathcal{ALCHOQ}^\sqcap$ knowledge base obtained from $\mathcal{K}$ as follows:*

1. *for each transitive role t and nominal $o \in \mathsf{nom}(\mathcal{K})$, add an axiom $\exists t.(\exists t.(\{o\})) \sqsubseteq \exists t.(\{o\})$,*
2. *for each concept $\forall R.C \in \mathsf{cl}(\mathcal{K})$ with $R = r_1 \sqcap \ldots \sqcap r_n$ and transitive roles $t_1, \ldots, t_n$ such that $t_i \sqsubseteq^*_{\mathcal{R}} r_i$ for each $1 \leq i \leq n$, add an axiom $\forall R.C \sqsubseteq \forall T.(\forall T.C)$, where $T = t_1 \sqcap \ldots \sqcap t_n$, and*
3. *all roles in $\mathsf{et}(\mathcal{K})$ are non-transitive.*

With the above definition, $\mathsf{et}(\mathcal{K})$ contains, additionally, the axiom $\exists t.(\exists t.(\{o\})) \sqsubseteq \exists t.(\{o\})$, which ensures that the implicit t-edges to nominals (the dashed lines in Figure 5) are made explicit. As a consequence, adding the axiom $\{o\} \sqsubseteq \forall(r \sqcap t).(\neg\{o'\})$ indeed results in an inconsistent knowledge base.

Due to role conjunctions over non-simple roles, the encoding from Definition 11 yields not necessarily a knowledge base whose size is polynomial in the size of the input knowledge base. The number of transitive sub-roles for a role r_i that occurs in a role conjunction is bounded by m. Hence, we can use up to m transitive sub-roles for each of the at most n role conjuncts in the second step of the encoding, which results in at most m^n additional axioms in $\mathsf{et}(\mathcal{K})$. Since, in general, the length of the longest role conjunction can also only be bound by m, the encoding is exponential in m. To the best of our knowledge, it is unknown if this blow-up can be avoided.

Lemma 12. *Let $\mathcal{K}$ be a $\mathcal{SHOQ}^\sqcap$ knowledge base with $|\mathcal{K}| = m$ and where the length of the longest role conjunction occurring in $\mathcal{K}$ is n. Then $\mathcal{K}$ is consistent iff $\mathsf{et}(\mathcal{K})$ is consistent and the size of $\mathsf{et}(\mathcal{K})$ is polynomial in m and exponential in n.*

In the remainder we assume w.l.o.g. that $\mathcal{K} = (\mathcal{T}, \mathcal{R})$ is an $\mathcal{ALCHOQ}^\sqcap$ knowledge base and that existential and universal restrictions in $\mathcal{K}$ are expressed using number restrictions.

Alternating Automata

In this section, we show how we can use (one-way) alternating automata (Muller and Schupp 1987) to decide the consistency of an $\mathcal{ALCHOQ}^\sqcap$ knowledge base. Alternating automata have the power of making both universal and existential choices. Informally, this means that in the transition function, we can create copies of the automaton, send them to successor nodes, and require that either some (existential) or all (universal) of them are accepting. We use, as usual, positive Boolean formulae as defined below in the specification of the transition function.

Definition 13. *A* labeled tree *over an alphabet Σ is a pair $(T, \mathcal{L})$, where T is a tree and $\mathcal{L} \colon T \to \Sigma$ maps each node in T to an element of Σ.*

Let X be a set of atoms. The set $\mathcal{B}^+(X)$ of positive Boolean formulae *is built over atoms from X,* true, *and* false *using only the connectives $\wedge$ and $\vee$. Let $X^\top$ be a subset of X. We say that $X^\top$* satisfies *a formula $\phi \in \mathcal{B}^+(X)$ if assigning* true *to all atoms in $X^\top$ and false to all atoms in $X \setminus X^\top$ makes ϕ true.*

Let $[k] = \{0, 1, \ldots, k\}$. An alternating looping tree automaton *on k-ary Σ-labeled trees is a tuple $\mathbf{A} = (\Sigma, Q, \delta, q_0)$, where Q is a finite set of states, $q_0 \in Q$ is the initial state, and $\delta \colon Q \times \Sigma \to \mathcal{B}^+([k] \times Q)$ is the transition function.*

A run *of $\mathbf{A}$ on a Σ-labeled k-ary tree $(T, \mathcal{L})$ is a $(T \times Q)$-labeled tree $(T_r, \mathcal{L}_r)$ that satisfies the following conditions:*

- *$\mathcal{L}_r(\varepsilon) = (\varepsilon, q_0)$,*
- *if $y \in T_r$ with $\mathcal{L}_r(y) = (x, q)$ and $\delta(q, \mathcal{L}(x)) = \phi$, then there is a (possibly empty) set $S \subseteq [k] \times Q$ that satisfies ϕ such that, for each $(c, q') \in S$, y has a successor $y{\cdot}i$ in T_r with $i \in \mathbb{N}$ and $\mathcal{L}_r(y{\cdot}i) = (x{\cdot}c, q')$.*

An automaton $\mathbf{A}$ accepts *an input tree T if there exists a run of $\mathbf{A}$ on T. The language accepted by $\mathbf{A}$, $\mathsf{lang}(\mathbf{A})$, is the set of all trees accepted by $\mathbf{A}$.*

For alternating automata, the *non-emptiness problem* is the following: given an alternating automaton $\mathbf{A}$, is there a tree $(T, \mathcal{L})$ such that $\mathbf{A}$ has an accepting run on $(T, \mathcal{L})$?

Please note that, since we use looping automata, we do not impose any acceptance conditions and each run is accepting, i.e., we require only that the conditions imposed on a run are satisfied.

Tree Relaxations

In this section, we show how we can obtain labeled k-ary trees from a canonical model for an $\mathcal{ALCHOQ}^\sqcap$ knowledge base that can be thought of as input for our automaton. Since the labeled trees that an automaton takes as input cannot have labeled edges, we additionally store, in the label of a node, with which roles it is related to its predecessor. Unfortunately, this does not work for the nominal nodes since a nominal node can be the successor of arbitrary elements and does not necessarily have a unique predecessor. In a first step, we build, therefore, a *relaxation* for a canonical model where, for each relationship between a node and a nominal node, we create a dummy node that is a representative of the nominal node. The label of the representative node is the extension of the label for the nominal node with rep and the roles via which it is related to the nominal.

In this section, we use the equisatisfiable $\mathcal{ALCHOQ}^{\sqcap}$ version $\mathsf{et}(\mathcal{K})$ of the running example, which also contains the axioms $\exists t.(\exists t.(\{o\})) \sqsubseteq \exists t.(\{o\})$ and $\exists t'.(\exists t'.(\{o\})) \sqsubseteq \exists t'.(\{o\})$. Without the dashed lines Figure 1 would represent a canonical model and forest base for $\mathsf{et}(\mathcal{K})$ and Figure 6 shows a relaxation for $\mathsf{et}(\mathcal{K})$.

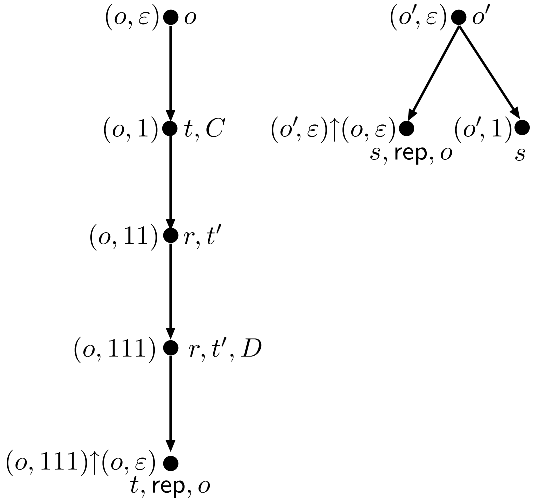

Figure 6: A graphical representation of a relaxation for $\mathcal{K}$.

Definition 14. *A set $H \subseteq \mathsf{cl}(\mathcal{K})$ is called a* Hintikka set *for $\mathcal{K}$ if the following conditions are satisfied:*

1. *For each $C \sqsubseteq D \in \mathcal{T}$, $\mathsf{nnf}(\neg C \sqcup D) \in H$.*
2. *If $C \sqcap D \in H$, then $\{C, D\} \subseteq H$.*
3. *If $C \sqcup D \in H$, then $\{C, D\} \cap H \neq \emptyset$.*
4. *For all $C \in \mathsf{cl}(\mathcal{K})$, either $C \in H$ or $\mathsf{nnf}(\neg C) \in H$.*

We use $H(\mathcal{K})$ to denote the set of all Hintikka sets for $\mathcal{K}$. A relaxation *$R = (\Delta, \mathcal{L})$ for $\mathcal{K}$ with $\mathcal{L} \colon \Delta \to 2^{\mathsf{cl}(\mathcal{K}) \cup \mathsf{rol}(\mathcal{K}) \cup \{\mathsf{rep}\}}$ satisfies the following properties:*

(R1) *Let $D = \mathsf{nom}(\mathcal{K}) \times \mathbb{N}^*$ and $B = \{d \uparrow d' \mid d \in D$ and $d' \in \mathsf{nom}(\mathcal{K}) \times \{\varepsilon\}\}$, then $\Delta \subseteq D \cup B$.*

(R2) *For each $o \in \mathsf{nom}(\mathcal{K}), (o, \varepsilon) \in \Delta$.*

(R3) *Each set $\{w \mid (o, w) \in \Delta \cap D$ is a tree$\}$.*

(R4) *If $d \uparrow d' \in \Delta$, then $\mathcal{L}(d \uparrow d') \cap \mathsf{cl}(\mathcal{K}) = \mathcal{L}(d') \cap \mathsf{cl}(\mathcal{K})$, and $\mathsf{rep} \in \mathcal{L}(d \uparrow d')$.*

(R5) *For each $d \in \Delta, \mathcal{L}(d) \cap \mathsf{cl}(\mathcal{K}) \in H(\mathcal{K})$.*

(R6) *For each $d \in \Delta$, if $r \sqsubseteq s \in \mathcal{R}$ and $r \in \mathcal{L}(d)$, then $s \in \mathcal{L}(d)$.*

(R7) *For each $(o, \varepsilon) \in \Delta, \mathcal{L}(o, \varepsilon) \cap \mathsf{rol}(\mathcal{K}) = \emptyset$.*

(R8) *If $d = (o, w) \in \Delta$ and $(\geqslant n\ (r_1 \sqcap \ldots \sqcap r_k).C) \in \mathcal{L}(d)$, then there are n distinct elements $d_1, \ldots, d_n \in \Delta$ such that, for each i with $1 \leq i \leq n, \{r_1, \ldots, r_k, C\} \subseteq \mathcal{L}(d_i)$ and either $d_i = (o, w \cdot c)$ with $c \in \mathbb{N}$ or $d_i = d \uparrow d' \in \Delta$.*

(R9) *If $d = (o, w) \in \Delta$ and $(\leqslant n\ (r_1 \sqcap \ldots \sqcap r_k).C) \in \mathcal{L}(d)$, then $\sharp(\{d' \in \Delta \mid d' = (o, w \cdot c)$ for some $c \in \mathbb{N}$ or $d' = d \uparrow d_o \in \Delta$ and $\{r_1, \ldots, r_k, C\} \subseteq \mathcal{L}(d')\}) \leq n$.*

Given the above properties of relaxations and the results from Lemma 3, we can show the following:

Lemma 15. *$\mathcal{K}$ has a relaxation iff $\mathcal{K}$ is consistent.*

In a second step, we build a tree relaxation, which is a labeled tree, from the relaxation. For this, we additionally add a dummy root node labeled with root that has all nominal nodes as successors, and we require that the domain is a tree.

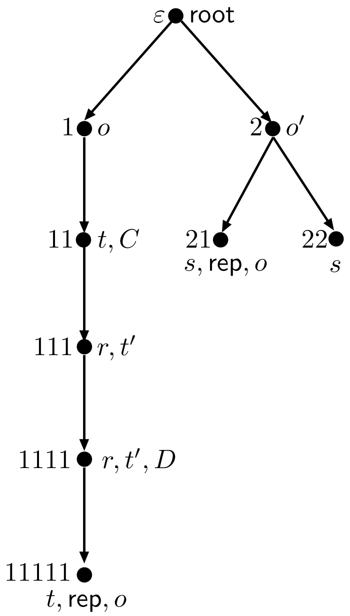

Figure 7: A tree relaxation for the relaxation from Figure 6.

Figure 7 shows a representation of a tree relaxation built from the relaxation for our running example. The tree relaxation can, additionally, have dummy nodes labelled with #, but we do not show any dummy nodes in the figure. For ease of presentation, we assume in the remainder that all tree relaxations are full trees, i.e., all non-leaf nodes have the same number of successors, and we add dummy nodes labelled with # where necessary.

Definition 16. *A* tree relaxation *for $\mathcal{K}$ is a labeled tree $(T, \mathcal{L})$ with $\mathcal{L} \colon T \to 2^{\mathsf{cl}(\mathcal{K}) \cup \mathsf{rol}(\mathcal{K}) \cup \{\mathsf{rep}, \#, \mathsf{root}\}}$ that satisfies the following conditions:*

(T1) $\mathcal{L}(\varepsilon) = \{\mathsf{root}\}$ *and, for each* $w \in \mathbb{N}^+, \mathcal{L}(w) \cap \{\mathsf{root}\} = \emptyset$.

(T2) *For each* $o \in \mathsf{nom}(\mathcal{K})$*, there is a unique* $c \in \mathbb{N} \cap T$ *with* $o \in \mathcal{L}(c)$ *and* $\{\mathsf{rep}, \#, \mathsf{rol}(\mathcal{K})\} \cap \mathcal{L}(c) = \emptyset$.

(T3) *If* $c \in \mathbb{N} \cap T$ *and* $\mathsf{nom}(\mathcal{K}) \cap \mathcal{L}(c) = \emptyset$*, then* $\mathcal{L}(c) = \{\#\}$.

(T4) *For each* $w \in \mathbb{N}^+ \cap T, \sharp(\mathcal{L}(w) \cap \mathsf{nom}(\mathcal{K})) \leq 1$.

(T5) *For each* $w = w' \cdot c \in T$ *with* $w' \in \mathbb{N}^+$ *and* $c \in \mathbb{N}$*, if* $\mathcal{L}(w) \cap \mathsf{nom}(\mathcal{K}) \neq \emptyset$*, then* $\mathsf{rep} \in \mathcal{L}(w)$.

(T6) *For each* $w, w' \in T$ *and* $o \in \mathsf{nom}(\mathcal{K})$*, if* $o \in \mathcal{L}(w) \cap \mathcal{L}(w')$*, then* $\mathsf{cl}(\mathcal{K}) \cap \mathcal{L}(w) = \mathsf{cl}(\mathcal{K}) \cap \mathcal{L}(w')$.

(T7) *For each* $w \in \mathbb{N}^+ \cap T$*, if* $\{\mathsf{rep}, \#\} \cap \mathcal{L}(w) \neq \emptyset$*, then, for each successor* w' *of* w*,* $\# \in \mathcal{L}(w')$.

(T8) *For each* $w \in T$*, if* $\mathcal{L}(w) \cap \{\#, \mathsf{root}\} = \emptyset$*, then* $\mathcal{L}(w) \cap \mathsf{cl}(\mathcal{K}) \in H(\mathcal{K})$.

(T9) *For each* $w \in T$ *and* $r \sqsubseteq s \in \mathcal{R}$*, if* $r \in \mathcal{L}(w)$*, then* $s \in \mathcal{L}(w)$.

(T10) *For each* $w \in T$*, if* $(\geqslant n\ (r_1 \sqcap \ldots \sqcap r_m).C) \in \mathcal{L}(w)$*, then there are at least* n *distinct successors* $w_1, \ldots, w_n$ *of* w *with* $\{r_1, \ldots, r_m, C\} \subseteq \mathcal{L}(w_i)$*, for each* i *with* $1 \leq i \leq n$.

(T11) *For each* $w \in T$*, if* $(\leqslant n\ (r_1 \sqcap \ldots \sqcap r_m).C) \in \mathcal{L}(w)$*, then there are at most* n *distinct successors* $w_1, \ldots, w_n$ *of* w *with* $\{r_1, \ldots, r_m, C\} \subseteq \mathcal{L}(w_i)$*, for each* i *with* $1 \leq i \leq n$.

If T *has branching degree* k*, then we say that* $(T, \mathcal{L})$ *is a* k-ary tree relaxation for $\mathcal{K}$.

By using the above defined properties of tree relaxations and the results about the bounded branching degree of canonical models from Lemma 3, we get the following:

Lemma 17. *Let* n_{max} *be the maximal number occurring in a number restriction in* $\mathcal{K}$*, and* $k = n_{max} \cdot |\mathcal{K}| + \sharp(\mathsf{nom}(\mathcal{K}))$*.* $\mathcal{K}$ *has a* k*-ary tree relaxation iff* $\mathcal{K}$ *is consistent.*

The Decision Procedure

It remains to devise a procedure that decides whether $\mathcal{K}$ has a tree relaxation. For this, we define an alternating automaton that accepts exactly the tree relaxations of $\mathcal{K}$. Our automaton combines and extends ideas from (Sattler and Vardi 2001) and (Calvanese, Eiter, and Ortiz 2007), where (two-way) alternating automata have been used for the hybrid μ-calculus (Sattler and Vardi 2001) and for answering regular path queries in $\mathcal{ALCQIb}_{reg}$. We first define two alternating automata $\bar{\mathcal{A}}_{\mathcal{K}}$ and $\mathcal{A}_{\mathcal{K}}$, and then define an automaton $\mathcal{B}_{\mathcal{K}}$ as their intersection. Informally, the automaton $\bar{\mathcal{A}}_{\mathcal{K}}$ just checks that the input tree has a structure as required whereas the automaton $\mathcal{A}_{\mathcal{K}}$ checks that the input is indeed a tree relaxation for $\mathcal{K}$. For alternating automata, intersection is simple: we introduce a new initial state q_0 and set the transition function for q_0 and each letter σ from the input alphabet Σ to $\delta(q_0, \sigma) = (0, q_{(0,1)}) \wedge (0, q_{(0,2)})$, where $q_{(0,1)}$ and $q_{(0,2)}$ are the initial states of $\bar{\mathcal{A}}_{\mathcal{K}}$ and $\mathcal{A}_{\mathcal{K}}$ respectively. The size of the resulting automaton is the sum of the sizes of $\bar{\mathcal{A}}_{\mathcal{K}}$ and $\mathcal{A}_{\mathcal{K}}$.

The automaton $\bar{\mathcal{A}}_{\mathcal{K}}$ is relatively straightforward and helps to keep the definition of the automaton $\mathcal{A}_{\mathcal{K}}$, where we do the real work, transparent. Informally, it guarantees the following:

- We distinguish root (state q_r), nominal (state q_o), nominal representative (state q_{rep}), dummy (state $q_\#$), and normal nodes (state q_n).
- The label root is only found in the root node.
- The level one nodes are either “real” nominal nodes (i.e., they are not marked as representatives with rep) with exactly one nominal and no roles in their label, or they are dummy nodes labelled with # only.
- The level one nominal nodes have either normal, nominal representative, or dummy nodes as successors.
- Nominal representative nodes are marked with rep, and have exactly one nominal in their label.
- Dummy nodes have only dummy nodes as successors.

More precisely, let n_{max} be the maximal number occurring in number restrictions in $\mathcal{K}$, and $k = n_{max} \cdot |\mathsf{cl}(\mathcal{K})| + \sharp(\mathsf{nom}(\mathcal{K}))$. The alphabet Σ for both automata $\bar{\mathcal{A}}_{\mathcal{K}}$ and $\mathcal{A}_{\mathcal{K}}$ is

$$2^{\{\mathsf{rep}, \#, \mathsf{root}\} \cup \mathsf{cl}(\mathcal{K}) \cup \mathsf{rol}(\mathcal{K}) \cup \mathsf{nom}(\mathcal{K})}.$$

We define $\bar{\mathcal{A}}_{\mathcal{K}}$ as $(\Sigma, \{q_r, q_o, q_n, q_{\mathsf{rep}}, q_\#\}, \bar{\delta}, q_r)$. The transition function $\bar{\delta}$ for each $\sigma \in \Sigma$ is as follows:

$$\bar{\delta}(q_r, \sigma) = \bigwedge_{i=1}^{k} (i, q_o) \vee (i, q_\#)$$

if $\sigma = \{\mathsf{root}\}$ and it is false otherwise.

$$\bar{\delta}(q_o, \sigma) = \bigwedge_{i=1}^{k} ((i, q_n) \vee (i, q_{\mathsf{rep}}) \vee (i, q_\#))$$

if $\sharp(\mathsf{nom}(\mathcal{K}) \cap \sigma) = 1$ and $\{\mathsf{root}, \mathsf{rep}, \#, \mathsf{rol}(\mathcal{K})\} \cap \sigma = \emptyset$ and it is false otherwise.

$$\bar{\delta}(q_n, \sigma) = \bigwedge_{i=1}^{k} ((i, q_n) \vee (i, q_{\mathsf{rep}}) \vee (i, q_\#))$$

if $\{\mathsf{root}, \mathsf{rep}, \#, \mathsf{nom}(\mathcal{K})\} \cap \sigma = \emptyset$ and it is false otherwise.

$$\bar{\delta}(q_{\mathsf{rep}}, \sigma) = \bigwedge_{i=1}^{k} (i, q_\#)$$

if $\sharp(\mathsf{nom}(\mathcal{K}) \cap \sigma) = 1, \mathsf{rep} \in \sigma$, and $\{\mathsf{root}, \#\} \cap \sigma = \emptyset$ and it is false otherwise.

$$\bar{\delta}(q_\#, \sigma) = \bigwedge_{i=1}^{k} (i, q_\#)$$

if $\{\#\} = \sigma$ and it is false otherwise.

The automaton $\mathcal{A}_{\mathcal{K}}$ mainly checks the formulae occurring in the labels of the input. Hence, most of the states correspond to formulae in $\mathsf{cl}(\mathcal{K})$ and the transition function is more or less determined by the semantics and we only sketch its realization. For number restrictions, we use the same technique as (Calvanese, Eiter, and Ortiz 2007) that involves states that count how many successors have been checked and how many of the checked ones fulfill the requirements of the number restriction. In the root node, we additionally make a non-deterministic choice, for each nominal and each atomic concept, whether the concept or its negation holds at the nominal node. This choice is propagated downwards in the tree in order to ensure that the nominal representatives agree with their corresponding real nominal nodes on all atomic concepts. This also enables us to simply count over the successors of a node for the qualified number restrictions. We propagate the concepts via a kind of universal role and we assume that u is a symbol that does not occur in $\mathsf{cl}(\mathcal{K})$ or $\mathsf{rol}(\mathcal{K})$. We define, therefore, the following set of auxiliary states

$$\begin{aligned} Q_{\mathsf{rep}} = & \{\neg\{o\} \sqcup A \mid o \in \mathsf{nom}(\mathcal{K}) \text{ and } A \in N_C \cap \mathsf{cl}(\mathcal{K})\} \cup \\ & \{\neg\{o\} \sqcup \neg A \mid o \in \mathsf{nom}(\mathcal{K}) \text{ and } A \in N_C \cap \mathsf{cl}(\mathcal{K})\}. \end{aligned}$$

We then define $\mathcal{A}_{\mathcal{K}}$ as (Σ, Q, δ, q_0), where q_0 is the initial state and the set Q of states is

$$\begin{aligned} & \{q_0\} \cup \mathsf{cl}(\mathcal{K}) \cup \mathsf{rol}(\mathcal{K}) \cup \{\neg r \mid r \in \mathsf{rol}(\mathcal{K})\} \cup \{q_{\mathcal{T}}, q_{\mathcal{R}}\} \cup \\ & \{\langle \bowtie nR.C, i, j\rangle \mid \bowtie \in \{\leqslant, \geqslant\}, \bowtie nR.C \in \mathsf{cl}(\mathcal{K}), \text{ and} \\ & 0 \leq i, j \leq k\} \cup Q_{\mathsf{rep}} \cup \{\forall u.C \mid C \in Q_{\mathsf{rep}}\}, \end{aligned}$$

where states of the form $\langle \bowtie\ n\ R.C, i, j\rangle$ are used to check that the number restrictions are satisfied.

In the following, we give the definition of the transition function for each $\sigma \in \Sigma$ together with an explanation for each of the different types of states.

At the root node, we are in the initial state q_0 which has the following tasks: (a) we make the non-deterministic guesses for all atomic concepts, (b) we check that there is exactly one nominal node for each of the nominals in $\mathsf{nom}(\mathcal{K})$, and (c) we make sure that the axioms in $\mathcal{T}$ and $\mathcal{R}$ are satisfied in all non-dummy descendants. Let $\ell = \sharp(\mathsf{nom}(\mathcal{K}))$.

$$\begin{aligned} & \delta(q_0, \sigma) = \bigwedge_{A \in \sigma \cap N_C} \bigwedge_{i=1}^{\ell} \\ & ((0, \forall u.(\neg\{o_i\} \sqcup A)) \vee (0, \forall u.(\neg\{o_i\} \sqcup \neg A)))\wedge \\ & \bigwedge_{i=1}^{\ell} \bigvee_{j=1}^{k} (j, \{o_i\}) \wedge \bigwedge_{1 \leq i < j \leq k} (\bigwedge_{o \in \mathsf{nom}(\mathcal{K})} (i, \neg\{o\}) \vee (j, \neg\{o\})) \\ & \bigwedge_{i=1}^{k} (i, q_{\mathcal{T}}) \wedge (i, q_{\mathcal{R}}) \end{aligned}$$

Whenever we are in a state that is used to propagate information downwards through the whole tree via the "universal role" and we are not at a dummy node, we check that the required concept holds at the current node and also check all successors. More precisely, for each $C \in Q_{\mathsf{rep}}$,

$$\delta(\forall u.C, \sigma) = (0, C) \wedge \bigwedge_{i=1}^{k} (i, \forall u.C)$$

if $\#, \mathsf{root} \notin \sigma$,

$$\delta(\forall u.C, \sigma) = \bigwedge_{i=1}^{k} (i, \forall u.C)$$

if $\mathsf{root} \in \sigma$, and it is true otherwise.

All non-dummy descendants of the root nodes must satisfy the TBox and RBox axioms.

$$\delta(q_{\mathcal{T}}, \sigma) = \bigwedge_{C \sqsubseteq D \in \mathcal{T}} ((0, \mathsf{nnf}(\neg C)) \vee (0, D)) \wedge \bigwedge_{i=1}^{k} (i, q_{\mathcal{T}})$$

if $\# \notin \sigma$ and it is true otherwise.

$$\delta(q_{\mathcal{R}}, \sigma) = \bigwedge_{r \sqsubseteq s \in \mathcal{R}} ((0, \neg r) \vee (0, s)) \wedge \bigwedge_{i=1}^{k} (i, q_{\mathcal{R}})$$

if $\# \notin \sigma$ and it is true otherwise.

The concepts that are used as states are inductively decomposed according to the semantics. We start by defining the base cases. For each $\alpha \in (N_C \cap \mathsf{cl}(\mathcal{K})) \cup \mathsf{rol}(\mathcal{K}) \cup \mathsf{nom}(\mathcal{K})$: $\delta(\alpha, \sigma) = \mathsf{true}$ if $\alpha \in \sigma$ and it is false otherwise; $\delta(\neg\alpha, \sigma) = \mathsf{true}$ if $\alpha \notin \sigma$ and it is false otherwise. Since we use constructors for nominals, they are not handled as atomic concepts and we set, for each $o \in \mathsf{nom}(\mathcal{K})$, $\delta(\{o\}, \sigma) = (0, o)$; $\delta(\neg\{o\}, \sigma) = (0, \neg o)$.

Conjunction and disjunction are handled in the straightforward way. For each $C_1 \sqcap C_2 \in \mathsf{cl}(\mathcal{K})$, $\delta(C_1 \sqcap C_2, \sigma) = (0, C_1) \wedge (0, C_2)$; for each $C_1 \sqcup C_2 \in \mathsf{cl}(\mathcal{K})$, $\delta(C_1 \sqcup C_2, \sigma) = (0, C_1) \vee (0, C_2)$.

For number restrictions, we have to use a more sophisticated technique that involves states that count how many successors have been checked and how many of the checked ones fulfill the requirements of the number restriction. This technique was introduced by Calvanese, De Giacomo, and Lenzerini (2002). More precisely, for each concept of the form $(\geqslant n\ R.C) \in \mathsf{cl}(\mathcal{K})$ with $R = r_1 \sqcap \ldots \sqcap r_m$,

$$\delta(\geqslant n\ R.C, \sigma) = (0, \langle \geqslant n\ R.C, 0, 0\rangle)$$

if $\mathsf{rep} \notin \sigma$ and it is true otherwise. For $1 \leq i \leq k$ and $1 \leq j \leq n, \delta(\langle \geqslant n\ R.C, i, j\rangle, \sigma) =$

$$\begin{aligned} & (((i, \neg r_1) \vee \ldots \vee (i, \neg r_m) \vee (i, \mathsf{nnf}(\neg C)))\wedge \\ & (0, \langle \geqslant n\ R.C, i+1, j\rangle))\vee \\ & ((i, r_1) \wedge \ldots \wedge (i, r_m) \wedge (i, C)\wedge \\ & (0, \langle \geqslant n\ R.C, i+1, j+1\rangle)) \end{aligned}$$

For $1 \leq i \leq k, \delta(\langle \geqslant n\ R.C, i, n\rangle, \sigma) = \mathsf{true}$. For $1 \leq j < n, \delta(\langle \geqslant n\ R.C, k, j\rangle, \sigma) = \mathsf{false}$.

Informally, we use the counter i to count how many of the k successors have already been checked and j is increased for each successor that fulfills the requirements of the number restriction. The atmost number restrictions are handled similarly:

$$\delta(\leqslant n\ R.C, \sigma) = (0, \langle \leqslant n\ R.C, 0, 0\rangle)$$

if $\mathsf{rep} \notin \sigma$ and it is true otherwise. For $1 \leq i \leq k$ and $1 \leq j \leq n, \delta(\langle \leqslant n\ R.C, i, j\rangle, \sigma) =$

$$\begin{aligned} & (((i, \neg r_1) \vee \ldots \vee (i, \neg r_m) \vee (i, \mathsf{nnf}(\neg C)))\wedge \\ & (0, \langle \leqslant n\ R.C, i+1, j\rangle))\vee \\ & ((i, r_1) \wedge \ldots \wedge (i, r_m) \wedge (i, C)\wedge \\ & (0, \langle \leqslant n\ R.C, i+1, j+1\rangle)) \end{aligned}$$

For $1 \leq i \leq k, \delta(\langle \leqslant n\ R.C, i, n+1 \rangle, \sigma) = \mathsf{false}$. For $1 \leq j < n, \delta(\langle \leqslant n\ R.C, k, j \rangle, \sigma) = \mathsf{true}$.

We can now check whether the language accepted by the automaton $\mathcal{B}_{\mathcal{K}}$ is empty, which is enough to decide consistency of $\mathcal{K}$:

Theorem 18. *Let $\mathcal{B}_{\mathcal{K}}$ an alternating automaton as defined above. Then $\mathcal{K}$ is consistent iff the language accepted by $\mathcal{B}_{\mathcal{K}}$ is non-empty.*

Since looping alternating tree automata are a special case of alternating Büchi tree automata, we can use the result that, for an alternating Büchi automaton $\mathbf{A}$ with n states and input alphabet with ℓ elements, non-emptiness of the language accepted by $\mathbf{A}$ is decidable in time exponential in n and polynomial in ℓ (Vardi 1995).

For the states, we mainly use formulae from $\mathsf{cl}(\mathcal{K})$ and the size of the closure is linear in $|\mathcal{K}|$ and, overall, the number of states for our automaton is polynomial in $|\mathcal{K}|$. The input alphabet $2^{\{\mathsf{rep},\#,\mathsf{root}\} \cup \mathsf{cl}(\mathcal{K}) \cup \mathsf{rol}(\mathcal{K}) \cup \mathsf{nom}(\mathcal{K})}$ is exponential in $|\mathcal{K}|$. Hence, we get the following result:

Theorem 19. *Checking the consistency of an $\mathcal{ALCHOQ}^{\sqcap}$ knowledge base $\mathcal{K}$ can be done in deterministic time single exponential in the size of $\mathcal{K}$ assuming unary coding of numbers in number restrictions.*

By Lemma 12, we obtain the following upper bound on deciding consistency of $\mathcal{SHOQ}^{\sqcap}$ knowledge bases.

Theorem 20. *Let $\mathcal{K}$ be a $\mathcal{SHOQ}^{\sqcap}$ knowledge base where $|\mathcal{K}| = m$ and the length of the longest role conjunction occurring in $\mathcal{K}$ is n. Deciding the consistency of $\mathcal{K}$ can be done in deterministic time single exponential in m and double exponential in n assuming unary coding of numbers in number restrictions.*

The above result also shows that $\mathcal{SHOQ}$, i.e., when no role conjunctions are used, is indeed ExpTime-complete. This was always conjectured but, to the best of our knowledge, never proved.

Conclusions and Future Work

Although the presented upper bound for query entailment in $\mathcal{SHOQ}$ agrees with the one for $\mathcal{SHIQ}$ (Lutz 2007) it is still an open whether this bound is tight. The hardness proof for $\mathcal{SHIQ}$ very much relies on the ability to propagate information upwards via inverse roles, which nominals can only simulate to some extend.

For $\mathcal{SHOQ}^{\sqcap}$, our algorithm runs in deterministic time double exponential in the size of the input knowledge base and, as for $\mathcal{SHIQ}^{\sqcap}$, it is still an open question whether this blow-up due to role conjunctions over non-simple roles can be avoided.

References

Baader, F.; Calvanese, D.; McGuinness, D. L.; Nardi, D.; and Patel-Schneider, P. F., eds. 2003. *The Description Logic Handbook.* Cambridge University Press.

Calvanese, D.; De Giacomo, G.; and Lenzerini, M. 1998. On the decidability of query containment under constraints. In *Proc. of the Seventeenth ACM SIGACT SIGMOD Sym. on Principles of Database Systems (PODS-98).* ACM Press and Addison Wesley.

Calvanese, D.; De Giacomo, G.; and Lenzerini, M. 2002. 2ATAs make DLs easy. In *Proc. of the 2002 Description Logic Workshop (DL 2002)*, volume 53. CEUR.

Calvanese, D.; Eiter, T.; and Ortiz, M. 2007. Answering regular path queries in expressive description logics: An automata-theoretic approach. In *Proc. of the Nat. Conf. on Artificial Intelligence (AAAI-07).*

Glimm, B.; Horrocks, I.; Lutz, C.; and Sattler, U. 2007. Conjunctive query answering in the description logic $\mathcal{SHIQ}$. In *Proc. of the 20th Int. Joint Conf. on Artificial Intelligence (IJCAI-07).*

Glimm, B.; Horrocks, I.; Lutz, C.; and Sattler, U. 2008. Conjunctive query answering for the description logic $\mathcal{SHIQ}$. *J. of Artificial Intelligence Research* 31:151–198.

Glimm, B. 2007. *Querying Description Logic Knowledge Bases.* Ph.D. Dissertation, The University of Manchester, Manchester, United Kingdom. `http://www.cs.man.ac.uk/~glimmbx/download/Glim07a.pdf`.

Horrocks, I.; Sattler, U.; Tessaris, S.; and Tobies, S. 2000. How to decide query containment under constraints using a description logic. In *Proc. of the 7th Int. Conf. on Logic for Programming and Automated Reasoning (LPAR 2000)*, Lecture Notes in Artificial Intelligence. Springer.

Horrocks, I.; Patel-Schneider, P. F.; and van Harmelen, F. 2003. From SHIQ and RDF to OWL: The making of a web ontology language. *John Wiley & Sons* 1(1):7–26.

Kazakov, Y., and Motik, B. 2006. A resolution-based decision procedure for $\mathcal{SHOIQ}$. In *Proc. of the Int. Joint Conf. on Automated Reasoning (IJCAR-06)*, volume 4130, 662–667. Springer.

Lutz, C. 2007. Inverse roles make conjunctive queries hard. In *Proc. of the 2007 Description Logic Workshop (DL 2007).*

Muller, D. E., and Schupp, P. E. 1987. Alternating automata on infinite trees. *Theoretical Computer Science* 54(2-3):267–276.

Sattler, U., and Vardi, M. Y. 2001. The hybrid μ-calculus. In *Proc. of the Int. Joint Conf. on Automated Reasoning (IJCAR-01).* Springer.

Tessaris, S. 2001. *Questions and answers: reasoning and querying in Description Logic.* PhD thesis, University of Manchester.

Tobies, S. 2001. *Complexity Results and Practical Algorithms for Logics in Knowledge Representation.* PhD thesis, RWTH Aachen.

Vardi, M. Y. 1995. Alternating automata and program verification. In *Computer Science Today*, volume 1000 of *Lecture Notes in Computer Science.* Springer.