Exhaustive Query Answering via Referring Expressions

David Toman and Grant Weddell

Cheriton School of Computer Science
University of Waterloo, Canada
{david,gweddell}@uwaterloo.ca

Abstract. Earlier work has considered how concepts can replace individual names as referring expressions in both instance retrieval and in more general query answering over knowledge bases with an underlying description logic. This earlier work, however, relied on this logic being able to express functionality, and also relied on syntactic typing restrictions on referring expressions to ensure that the number of query answers was finite. Here, we introduce a variety of referring expression concepts that extend query answering with respect to Horn-$\mathcal{ALC}$ and $\mathcal{EL}^{\perp}$, description logics that are not able to express functionality, and techniques necessary to finitely describe *all* entailed answers to queries expressed in terms of such concepts.

1 Introduction

Usually, individual names occurring in a knowledge base $\mathcal{K}$ expressed in terms of an underlying DL serve the role of *referring expressions* in query answering. However, earlier work has considered how concepts in the DL can replace individual names as referring expressions in instance retrieval [6] and, more recently, in the case of conjunctive queries [2, 7]. The more recent work, however, relied on two things. First, the underlying DL needed to be able to express functionality in order to ensure a concept serving the role of a referring expression satisfied a strong *singularity* property. This property required the denotation of the concept to be a singleton set for *any* interpretation of $\mathcal{K}$. And second, this recent work relied on syntactic typing restrictions on referring expressions to ensure that the number of query answers was finite.

In this paper, we introduce a variety of referring expression concepts that extend query answering with respect to Horn-$\mathcal{ALC}$ and $\mathcal{EL}^{\perp}$, description logics that are not able to express functionality, and techniques necessary to finitely describe *all* entailed answers to queries expressed in terms of such concepts. Fundamentally, this involves weakening the strong notion of singularity, requiring instead, that the denotation of a referring concept is a singleton set *for some tree interpretation* of $\mathcal{K}$. Also, our preliminary focus on Horn-$\mathcal{ALC}$ and $\mathcal{EL}^{\perp}$ enables a more transparent development since a knowledge base over these dialects will have a *unique* tree interpretation.

The remainder of the paper is organized as follows: Section 2 gives the necessary general definitions, Section 3 studies the problem of *instance retrieval* in Horn-$\mathcal{ALC}$ and $\mathcal{EL}^{\perp}$, and Section 3.3 discusses a finite representation of sets of answers. The development for instance retrieval is then extended, in Section 4.2, to conjunctive queries. Section 5 summarizes and outlines directions for further research.

2 Background and Definitions

We begin by defining a space of concept descriptions for the function free DL dialects that will concern us, including the concept descriptions that replace individual names in the role of *referring expressions* in query answering:

Definition 1 (Concept Language)
Let R, PC and IN be disjoint sets of role names, primitive concept names and individual names respectively. Derived *concept descriptions* and their *semantics* are defined as follows:

$$
\begin{array}{lll r}
 & \textsc{Syntax} & \textsc{Semantics: Defn of } \text{“}\cdot^{\mathcal{I}}\text{”} & \\
C ::= & A & A^{\mathcal{I}} \subseteq \triangle & \text{(primitive concept; } A \in \mathsf{PC}) \\
\mid & C_1 \sqcap C_2 & C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}} & \text{(conjunction)} \\
\mid & \perp & \{\} & \text{(bottom)} \\
\mid & \forall R.C & \{x \mid \forall y : (x,y) \in R^{\mathcal{I}} \to y \in C^{\mathcal{I}}\} & \text{(value restriction; } R \in \mathsf{R}) \\
\mid & \exists R.C & \{x \mid \exists y : (x,y) \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\} & \text{(existential restriction; } R \in \mathsf{R}) \\
\mid & \exists R^{-}.C & \{x \mid \exists y : (y,x) \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\} & \text{(inverse existential restriction)} \\
\mid & \{a\} & \{a^{\mathcal{I}}\} & \text{(nominal; } a \in \mathsf{IN})
\end{array}
$$

The semantics is with respect to a structure $\mathcal{I} = (\triangle, \cdot^{\mathcal{I}})$ in which $\triangle$ is a domain of “objects” and $\cdot^{\mathcal{I}}$ an interpretation function seeded by fixing the interpretations of primitive concept names A to be subsets of $\triangle$ (as indicated), role names R to be subsets of $\triangle \times \triangle$, and individual names a to be elements of $\triangle$ (and is extended to derived concept descriptions C as also indicated). $\square$

The DL dialects $\mathcal{EL}^{\perp}$ and Horn-$\mathcal{ALC}$ are given as follows:

Definition 2 (Horn-$\mathcal{ALC}$ and $\mathcal{EL}^{\perp}$ TBoxes and Knowledge Bases)
A Horn-$\mathcal{ALC}$ or $\mathcal{EL}^{\perp}$ *knowledge base* $\mathcal{K}$ consists of a *TBox* $\mathcal{T}$ and *ABox* $\mathcal{A}$, where $\mathcal{T}$ consists of a finite set of *subsumptions* of the form $C \sqsubseteq D$ in which

- C is a conjunction of primitive concepts A and existential restrictions of the form $\exists R.A$, and
- D is one of $\perp$, A, $\exists R.A$, and, in the case of Horn-$\mathcal{ALC}$, $\forall R.A$,

and where $\mathcal{A}$ consists of a finite set of *assertions* of the form $a : A$ and $R(a,b)$. An interpretation $\mathcal{I}$ is called a *model* of $\mathcal{K}$ if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ for all $C \sqsubseteq D \in \mathcal{T}$, $a^{\mathcal{I}} \in C^{\mathcal{I}}$ for all $a : C \in \mathcal{A}$, and $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$ for all $R(a,b) \in \mathcal{A}$.

Consistency, logical implication, and other reasoning problems are defined in the standard way. □

Observe that we require TBoxes to be in a simple normal form. For more general but expressively equivalent syntax, see [3].

Tree Models. Hereon, we rely on the fact that DL knowledge bases will usually possess the *tree model property*: with the exception of the explicit ABox, satisfiable knowledge bases have a tree-like model in which all anonymous objects form a role-connected forest rooted by ABox individuals. Moreover, in the tree parts of this model, no individuals are made equal unless forced to do so by TBox assertions.

For Horn logics, one can also show that there is a unique tree-like model commonly called the minimal or universal model that captures all the facts implied by the knowledge base. Thus, many reasoning tasks, in particular, *instance retrieval*, reduce to inspecting this model.

Queries and Referring Expressions. In the classical setting, instance retrieval (resp. query answering) with respect to a knowledge base $\mathcal{K}$ and a concept C (resp. query Q) is the task that determines for which *individual names appearing in* $\mathcal{K}$ it holds that $\mathcal{K} \models a : C$ (resp. $\mathcal{K} \models Q(a_1, \ldots, a_k)$). Here, constant names serve the role of referring expressions, and our concern is with replacing such expressions by more general concept descriptions:

Definition 3 (Referring Expressions)
Referring expressions are simply concepts in (a subset of) the above concept language. In the following, we use concept descriptions of the form

$$C_1 \sqcap \exists R_1^-.(C_2 \sqcap \exists R_2^-.(\ldots \exists R_k^-.\{a\}))$$

where C_i are (conjunctions) of primitive concepts. □

The intuition behind this choice of referring expressions lies in the tree model property of our logics: every anonymous object can be reached by a role path from an ABox individual. (Indeed, unreachable objects that may exist in some models of our knowledge bases should not be considered since they fail to qualify as certain answers.)

In order to use *referring expressions* in place of constant symbols, one should ensure that they describe a single (certain) answer. Also note that, to account for various DL dialects, both knowledge base subsumptions/assertions and referring expressions will be restricted to appropriate subsets of the concept language in Definition 1. The following definition of a *singularity property* of concepts serving the role of referring expressions in instance checking, however, is independent of the choice of DL dialect:

Definition 4 (Singular Certain Answers)
Let $\mathcal{K}$ be a knowledge base, D an instance query (i.e., a concept expression), and C a referring expression. We say that C is a *singular certain answer* to D if

1. (certainty) $\mathcal{K} \models C \sqsubseteq D$ and $|C^{\mathcal{I}}| > 0$ for all models $\mathcal{I}$ of $\mathcal{K}$, and
2. (singularity) $|C^{\mathcal{I}}| = 1$ for at least one *tree* model $\mathcal{I}$ of $\mathcal{K}$. □

This constitutes a *weakening* of the singularity property defined in [2] in which a referring expression was required to denote a singleton set in *all* models of the knowledge base. Indeed, this is essential since DL dialects such as Horn-$\mathcal{ALC}$ and $\mathcal{EL}^{\bot}$ are not sufficiently expressive to enforce the stronger requirement. In these logics, it is always possible to replicate identical successors of objects in a model without invalidating any TBox subsumptions. Doing this leads immediately to a violation of the singularity property of [2]. However, in the setting of *certain answers*, the weaker requirement seems sufficient: it guarantees that it is never the case that the referring expression describes more than one answer *in every* model of $\mathcal{K}$. To illustrate, consider the following:

Example 5
$\mathcal{T} = \{A \sqsubseteq \exists R.C \sqcap \exists R.D, A \sqsubseteq \forall R.B\}$ and $\mathcal{A} = \{a : A\}$. Then $C \sqcap \exists R^{-}.\{a\}$ and $D \sqcap \exists R^{-}.\{a\}$ are singular certain answers for the instance query $B(x)$, but $\exists R^{-}.\{a\}$ is not since it fails the singularity requirement. (The description contains at least two objects in every tree model of $\mathcal{K}$.)

This seems to be in agreement with the usual entailment style of semantics for certain answers in the database community that thinks of the results of a query as an "intersection over all models". The benefit of this weaker definition is that results can now apply to logics that are unable to express functionality, such as Horn-$\mathcal{ALC}$ or $\mathcal{EL}^{\bot}$, which were excluded from consideration in [2].

Conversely, we require the weaker singularity condition to hold in a *tree model* of the knowledge base. This avoids models that equate objects without a need to do so. Allowing such models in our definition of singularity would incorrectly allow for concepts to be considered referring expressions for singular certain answers even though there could be two or more referring expressions that also describe singular answers and imply the expression in question. The following illustrates this case:

Example 6
Consider again the situation in Example 5. Without restricting the singularity requirement to *tree models*, we could use an interpretation $\mathcal{I}$ that maps the R successors of $a^{\mathcal{I}}$ to the same anonymous object. That interpretation is a model of $\mathcal{K}$ since C and D are not mandated to be disjoint. Hence, $\exists R^{-}.\{a\}$ would be incorrectly considered to be a singular certain answer even though two distinct singular certain answers referring to two distinct objects (as in Example 5) are subsumed by this description. Hence, this expression should not be considered singular.

Note that the above example presents a situation in which $\exists R^{-}.\{a\}$ refers to two distinct certain answers. However, note that aliases, that is, alternative referring expressions that refer to the same *single* answer, are still possible. This is natural and similar to standard approaches in which distinct constants may be interpreted as the same individual.

3 Instance Retrieval and Unit ABox

We first consider the problem of generalized instance retrieval. In the classical setting, this task deals only with ABox individuals. However, in our setting, referring expressions can describe certain answers that can be *arbitrarily far* from ABox individuals denoted by constant symbols.

The two DLs that we consider, Horn-$\mathcal{ALC}$ and $\mathcal{EL}^{\perp}$, do not possess the capability of expressing the functionality of roles. A slightly surprising result is that Horn-$\mathcal{ALC}$ or $\mathcal{EL}^{\perp}$ TBoxes are *not* able to enforce the existence of objects that are indistinguishable by appropriate referring expressions. Hence, *all* possible answers can in principle be described by such expressions as *singular certain answers*.

To simplify the exposition and focus on the issues connected with referring expressions, we first assume that the ABox in a knowledge base contains a single assertion $a : A$. (We relax this restriction later.) Initially, we only consider instance retrieval queries of the form $B(x)$ for B a primitive concept; instance queries for more complex concepts can be reduced to this case by introducing appropriate subsumptions in the TBox.

3.1 The Horn-$\mathcal{ALC}$ Case

In principle, testing whether a concept is an answer to an instance query reduces to a simple logical implication problem (perhaps in an extension of Horn-$\mathcal{ALC}$). The main questions we answer here are *what* concepts should qualify as referring expressions, and *how* one guarantees singularity for these expressions.

To answer these questions, we utilize a construction similar to the standard construction of a *tree automaton* for recognizing tree models of the knowledge base.[1] We generate a transition relation from our instance checking problem as follows:

Definition 7
Let $\mathcal{K} = (\mathcal{T}, \{a : A\})$ be a Horn-$\mathcal{ALC}$ knowledge base (in normal form) and $\mathsf{Concepts}(\mathcal{K})$ the set of all concepts and subconcepts appearing in $\mathcal{K}$.

We define $\mathsf{Implied}(S) = \{C \in \mathsf{Concepts}(\mathcal{K}) \mid \mathcal{T} \models \bigsqcap_{A \in S} A \sqsubseteq C\}$, where S is a set of primitive concepts, and define $\mathcal{S}_{\mathcal{K}} = \{S \mid S \subseteq \mathsf{PC} \cap \mathsf{Concepts}(\mathcal{K})\}$.

We say that an existential restriction $\exists R.C \in \mathsf{Implied}(S)$ is *independent* if it is minimal among existential restrictions in $\mathsf{Implied}(S)$ with respect to subsumption. For mutually equivalent restrictions, we chose one representative to be independent.

A *matching tuple for* $S \in \mathcal{S}_{\mathcal{K}}$ is a tuple

$$(S, \{C_0, D_{0,0}, \ldots, D_{0,k_0}\}, \ldots, \{C_k, D_{k,0}, \ldots, D_{k,k_k}\})$$

[1] In the standard construction, the Hintikka sets are generated syntactically by analyzing concepts present in a TBox. Here, to simplify the presentation, we rely on logical implication algorithms already developed for the underlying logics.

where $\exists R_0.C_0, \ldots, \exists R_k.C_k$ are all independent existential restrictions that appear in $\mathsf{Implied}(S)$ and $\forall R_0.D_{0,0}, \ldots, \forall R_0.D_{0,k_0}, \ldots, \forall R_k.D_{k,0}, \ldots, \forall R_k.D_{k,k_k}$ are all value restrictions that appear in $\mathsf{Implied}(S)$. We say that $\{C_i, D_{i,0}, \ldots, D_{i,k_i}\}$ belongs to S's matching tuple *for the existential restriction* $\exists R_i.C_i$.

This construction is similar to the looping automaton construction for $\mathcal{K}$ with an initial state $\{A\}$. However, note that the transitions are deterministic for Horn-$\mathcal{ALC}$. A similar construction also yields an optimal EXPTIME upper bound for satisfiability of Horn-$\mathcal{ALC}$ knowledge bases since the number of the sets in the construction is at most exponential in $|\mathcal{K}|$ (as is the size of the tree automaton), and testing for the emptiness of a looping tree automaton can be done in time polynomial in the number of states as follows:

Set $S \in \mathcal{S}_\mathcal{K}$ is *feasible* if
1. $\perp \notin \mathsf{Implied}(S)$, and
2. for the matching tuple $(S, S_0, \ldots, S_k)$ all S_i are feasible.

Otherwise, S is infeasible.

It is easy to see that the above definition of (in)feasible states can be implemented by an algorithm that marks all infeasible states in $|\mathcal{S}_\mathcal{K}|$ rounds. Consequently, $\mathcal{K}$ is satisfiable if and only if the initial state $\{A\}$ is feasible since the structure finitely encodes the universal (minimal) model of $\mathcal{K}$: the model corresponds to the unfolding of the structure starting from $\{A\}$ (i.e., a run of the automaton). We use the feasible states and the structure defined over them by the matching tuples (a.k.a., the transition relation of the looping automaton) to define referring expressions that will serve as our singular certain answers:

Definition 8 (Certain Paths and Referring Expressions)
A *certain path for a query $B(x)$ and knowledge base $\mathcal{K}$* is a sequence of role and concept pairs $R_1A_1 \ldots R_kA_k$ such that there are feasible $S_0, \ldots, S_k \in \mathcal{S}_\mathcal{K}$ and

1. $S_0 = \{A\}$,
2. $B \in \mathsf{Implied}(S_k)$, and
3. S_{i+1} belongs to S_i's matching tuple for the existential restriction $\exists R_i.A_i$.

Observe that we consider *all such paths* in the above (i.e., not just paths that are simple). Also note that, unlike satisfiability, we need to make certain that the referring expression concept *works* in all models of $\mathcal{K}$. Here we again take advantage of the logic being Horn and rely on the (universal) tree model captured by the above construction.

Theorem 9
Every certain path $R_1A_1 \ldots R_kA_k$ for B and $\mathcal{K}$ corresponds to a singular certain answer $A_k \sqcap \exists R_k^-.(\ldots A_1 \sqcap \exists R_1^-.\{a\})$. Moreover, every B object common to all models of $\mathcal{K}$ will be reached by a certain path and will be returned as an answer.

Proof (sketch): The construction guarantees that the referring expressions constructed from certain paths satisfy the certainty condition of our definition: the end object of every certain path for $B(x)$ and $\mathcal{K}$ is in the interpretation of the B concept in the minimal model and thus in all models of $\mathcal{K}$. The objects at the ends of these paths are referred to by the referring expression concept constructed from such paths.

Requiring only independent existential restrictions to be parts of matching tuples guarantees singularity of the certain answers witnessed by the tree model of $\mathcal{K}$.

3.2 The $\mathcal{EL}^\perp$ Case

We use the same construction. However, in the absence of value restrictions, observe that only the sets $\mathsf{Implied}(\{A\})$, where $A \in \mathsf{PC}$, are needed. There are only polynomially many of these, all of which can now be constructed in PTIME.[2]

3.3 Finite Representation of Answers

Our focus so far has been on problems of determining *if a referring expression is a singular certain answer* to an instance query $B(x)$ over a knowledge base $\mathcal{K}$. However, in practical information systems, one is often faced with the task of reporting *all* certain answers. This is easy in the standard case: we simply consider the available constant symbols one-by-one. The following examples show that this is not so simple for referring expressions.

In the case of acyclic TBoxes (even in $\mathcal{EL}^\perp$), the number of singular certain answers can be easily exponential (and doubly exponential in the case of Horn-$\mathcal{ALC}$):

Example 10
Consider a knowledge base with unit ABox and an $\mathcal{EL}^\perp$ TBox of the form

$$\mathcal{T} = \{A \sqsubseteq \exists R.B_0 \sqcap \exists S.B_0, \ldots, B_{k-1} \sqsubseteq \exists R.B_k \sqcap \exists S.B_k\}$$

for $k > 0$. Our construction gives k matching tuples $(\{B_i\}, \{B_{i+1}\}, \{B_{i+1}\})$ (plus a tuple $(\{B_k\})$). This, however, leads to exponentially many certain paths that are witnessed by the *tree model* of this TBox that contains 2^k leaves.

The situation is even worse in the case of Horn-$\mathcal{ALC}$ since one can force paths of exponential length using value restrictions and auxiliary concepts that stand for counters. Hence, one can force 2^{2^k} certain paths (and in turn singular certain answers).

For cyclic TBoxes, it is easy to construct examples in which the number of singular certain answers is infinite:

Example 11
Let $T = \{A \sqsubseteq \exists R.A\}$, and $\mathcal{A} = \{a : A\}$. Then $\{a\}$, $\exists R^-.\{a\}$, $\exists R^-.\exists R^-.\{a\}$, $\exists R^-.\exists R^-.\exists R^-.\{a\}$, etc., are all singular certain answers to $A(x)$.

[2] This construction is essentially the same as the construction of the so called *canonical model* for $\mathcal{EL}^\perp$ [1, 5].

One can *represent* all these answers as simple regular expression-based extensions of our language of referring expressions, stating that the singular certain answers can be reached, for example, by $R_1 \dots R_{i-1}[R_i \dots R_k]*$ paths. When transformed to the concept language embellished by a Kleene star-like construct, such a referring expression would appear as follows:

$$[C_k \sqcap \exists R_k^-.(\dots C_i \sqcap \exists R_i^-.()^* C_{i-1} \sqcap \exists R_{i-1}^-.(\dots \exists R_1^-.\{a\}))).$$

Note that the regular-like concept description corresponds to the certain path written backward, hence the cycle is syntactically at the beginning of this expression. Such expressions can be *extracted* from our construction of matching tuples as concatenations of simple paths from A to B followed by B to B cycles. However, while this solves our problems with the finiteness of (the presentation of) all answers, issues connected with the number of answers raised in Example 10 remain. Similarly, the number of distinct simple cycles can be bounded by a factorial function from below. The representations consisting of sets of matching tuples (essentially the transition relation of a tree automaton) are vastly more succinct, but may not be appropriate as an end user feedback. Indeed, a succinct and user-friendly representation remains a topic for further research.

4 Extensions

This section considers relaxing the various restrictions that we have assumed in addressing the problem of exhaustive query answering via referring expressions (restrictions that enabled a simpler exposition of what we believe are the principle issues).

4.1 General ABoxes

One can use a standard approach to extend an explicitly given ABox to a tree model (represented again using matching tuples). One issue that needs to be addressed is guaranteeing the singularity of answers. This is not an issue for the ABox individuals, but roles in the ABox can make certain existential restrictions redundant and break our *independence* requirement, as illustrated in the following:

Example 12
Consider knowledge base $\mathcal{K}$ with an ABox $\mathcal{A} = \{A(a), R(a,b), B(b)\}$ and a TBox $\mathcal{T} = \{A \sqsubseteq \exists R.B\}$. Considering the TBox alone, we generate a matching tuple $(\{A\}, \{B\})$ that is used to generate (anonymous) R successors for As. However, were this tuple used for the a object, it would lead to a certain answer $\exists R^-.\{a\}$ no longer being singular (in the constructed model). Extending the *independence* requirement to eliminate redundant existential restrictions by generating additional matching tuples for ABox objects solves this problem. In this particular case, one would generate a tuple $(\{A, \{a\}\})$ with no successors.

The above approach can be applied to all ABox objects leading to at most $|\mathcal{A}|$ increase in the number of matching tuples and hence preserving our complexity bounds.

4.2 Conjunctive Queries

One simply reduces conjunctive query (CQ) answering to existing approaches that deal with queries whose answer variables match ABox individuals [4, 5]. Note that such approaches require slight extensions to account for anonymous objects in the $|CQ|$-*neighborhood* of the ABox that can now be described by referring expressions. What remains is to introduce additional cases for a CQ that can be *folded to concept descriptions* for which our instance retrieval approach can then be applied, in particular, when:

1. the whole query *matches* in the knowledge base's ABox,
2. part of the query matches in the ABox and part in the implied part of models, and
3. the whole query (folding) matches in the implied part (in all models).

Observe with the standard setting for open queries that only cases (1) and (2), where all answer variables match in the ABox, will apply. This is no longer the case when referring expressions can be used: in cases (2) and (3) referring expressions can provide *bindings* for variables that match outside of an ABox. However, such matches only apply to those portions of the given conjunctive query that can be folded to a tree-shaped concept description since models of description logic knowledge bases have the tree model property for the anonymous parts. This implies in turn that matches for inherently non-tree-shaped conjunctive queries are not possible. However, whenever referring expressions are used as parts of the answer tuples, they will always satisfy our singularity condition jointly (i.e., *in the same model* of the knowledge base) as witnessed by the tree model.

The latter two cases need to make certain allowances for free variables of the queries that can now match anonymous objects referred to by our referring expressions in the answer tuples. This requires simple, but slightly tedious housekeeping to be added to the process to track the variable matches, in particular in case (3).

4.3 Logics with Number Restrictions

When quantified role restrictions of the form $(\geq 2\ R.C)$ are present in the language, it may not be possible to describe all answers as singular certain answers since such *at-least* restrictions can force multiple certain answers that cannot be distinguished by referring expressions (without the loss of singularity). Note, however, that genuine at-least restrictions can be modeled by existential restrictions and auxiliary disjoint primitive concepts. Then, however, those concepts will guarantee singularity in the tree model.

Results are better with only functionality or at-most restrictions, although there remains some dependence on the way such restrictions are realized in the TBox or concept language, for example, as (**func** R) constraints or as $(\leq 1\ R.C)$ concepts. Indeed, negations in the latter case can lead to *at-least* restrictions and non-singularity of certain answers.

4.4 Non-Horn Logics

The situation for non-Horn logics is even more complex: we can certainly extend our construction to full $\mathcal{ALC}$, but we face the following issue in the presence of disjunctions, in particular, when such disjunctions are allowed in referring expressions:

Example 13
According to our definition of singularity, given a TBox $\{A \sqsubseteq \exists R.B \sqcup \exists S.B\}$, an ABox $\{a : A\}$, and a query $B(x)$, a singular certain answer could be $\exists R^{-}.\{a\} \sqcup \exists S^{-}.\{a\}$ since the two (minimal) tree models will contain $\{R(a, o), B(o)\}$ and $\{S(a, o), B(o)\}$.
Even worse, where the TBox given instead as $\{A \sqsubseteq (\exists R.B \sqcap \exists S.B) \sqcup \exists T.B\}$, one could have *two* certain answers, both singular: $\exists R^{-}.\{a\} \sqcup \exists T^{-}.\{a\}$ and $\exists S^{-}.\{a\} \sqcup \exists T^{-}.\{a\}$, that seem to *reuse* the second part of the disjunction. This not only leads to combinatorial problems but also renders answers that are unintuitive.

Also, observe in the first case that the anonymous object o, indeed *the answer we are trying to refer to*, need not be the same object in the two models. However, this isn't too different from interpreting a constant symbol by varying domain elements in different models of a knowledge base. The downside of this arrangement is that such a system will be reporting answers that contain (possibly large numbers of) disjunctions may not be what users of such a system would expect. Limiting what referring expressions are used in answers has been considered in [2] where the idea of referring expression types was introduced.

5 Summary and Open Problems

We have presented an extension to instance retrieval and query answering tasks that, with the help of referring expressions, allows one to return *all* singular certain answers in Horn-$\mathcal{ALC}$ and $\mathcal{EL}^{\perp}$ knowledge bases. We have also shown that this is no longer the case for logics endowed with *at-least* number restrictions, even though additional answers are still possible when referring expressions can be arbitrary concepts.

There are many directions for further research, in particular:

– Issues related to a more compact representation of answers; this direction is related to discovering "small" regular expressions or devising other ways to present all the singular certain answers over a knowledge base.

– Extensions to more powerful Horn description logics: what concept constructors can be supported while maintaining the ability to report all answers? What to do with at-least restrictions and (unlike functionality) do we really need them?
– Extensions to non-Horn Description Logics: can the techniques be extended to DLs with concept disjunction (see the discussion in Section 4.4)?

References

1. Baader, F., Brandt, S., Lutz, C.: Pushing the $\mathcal{EL}$ Envelope. In: Proc. Int. Joint Conf. on Artificial Intelligence (IJCAI). pp. 364–369 (2005)
2. Borgida, A., Toman, D., Weddell, G.: On referring expressions in query answering over first order knowledge bases. In: Proc. KR. pp. 319–328 (2016)
3. Hustadt, U., Motik, B., Sattler, U.: Data complexity of reasoning in very expressive description logics. In: Proc. Int. Joint Conf. on Artificial Intelligence (IJCAI). pp. 466–471. Professional Book Center (2005)
4. Lutz, C., Seylan, I., Toman, D., Wolter, F.: The combined approach to OBDA: Taming role hierarchies using filters. In: ISWC (1). pp. 314–330 (2013)
5. Lutz, C., Toman, D., Wolter, F.: Conjunctive query answering in the description logic EL using a relational database system. In: Proc. IJCAI. pp. 2070–2075 (2009)
6. Pound, J., Toman, D., Weddell, G.E., Wu, J.: Concept projection in algebras for computing certain answer descriptions. In: Grau, B.C., Horrocks, I., Motik, B., Sattler, U. (eds.) Proceedings of the 22nd International Workshop on Description Logics (DL 2009), Oxford, UK, July 27-30, 2009. CEUR Workshop Proceedings, vol. 477. CEUR-WS.org (2009), `http://ceur-ws.org/Vol-477/paper_44.pdf`
7. Toman, D., Weddell, G.E.: Identity resolution in conjunctive querying over dl-based knowledge bases. In: Ortiz, M., Schneider, T. (eds.) Proceedings of the 31st International Workshop on Description Logics co-located with 16th International Conference on Principles of Knowledge Representation and Reasoning (KR 2018), Tempe, Arizona, US, October 27th - to - 29th, 2018. CEUR Workshop Proceedings, vol. 2211. CEUR-WS.org (2018), `http://ceur-ws.org/Vol-2211/paper-34.pdf`