Charting the Potential of Description Logic for the Generation of Referring Expressions

Yuan Ren and **Kees van Deemter** and **Jeff Z. Pan**
Department of Computing Science
University of Aberdeen
Aberdeen, UK

Abstract

The generation of referring expressions (GRE), an important subtask of Natural Language Generation (NLG) is to generate phrases that uniquely identify domain entities. Until recently, many GRE algorithms were developed using only simple formalisms, which were taylor made for the task. Following the fast development of ontology-based systems, reinterpretations of GRE in terms of description logic (DL) have recently started to be studied. However, the expressive power of these DL-based algorithms is still limited, not exceeding that of older GRE approaches. In this paper, we propose a DL-based approach to GRE that exploits the full power of OWL2. Unlike existing approaches, the potential of reasoning in GRE is explored.

1 GRE and KR: the story so far

Generation of Referring Expressions (GRE) is the subtask of Natural Language Generation (NLG) that focuses on identifying objects in natural language. For example, Fig.1 depicts the relations between several women, dogs and cats. In such a scenario, a GRE algorithm might identify $d1$ as "the dog that loves a cat", singling out $d1$ from the five other objects in the domain. Reference has long been a key issue in theoretical linguistics and psycholinguistics, and GRE is a crucial component of almost every practical NLG system. In the years following seminal publications such as (Dale and Reiter, 1995), GRE has become one of the most intensively studied areas of NLG, with links to many other areas of Cognitive Science. After plan-based contributions (e.g., (Appelt, 1985)), recent work increasingly stresses the human-likeness of the expressions generated in simple situations, culminating in two evaluation campaigns in which dozens of GRE algorithms were compared to human-generated expressions (Belz and Gatt, 2008; Gatt et al., 2009).

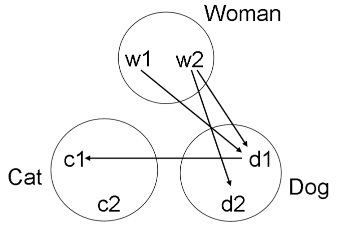

Figure 1: An example in which edges from women to dogs denote $feed$ relations, from dogs to cats denote $love$ relations.

Traditional GRE algorithms are usually based on very elementary, custom-made, forms of Knowledge Representation (KR), which allow little else than atomic facts (with negation of atomic facts left implicit), often using a simple $\langle Attribute : Value\rangle$ format, e.g $\langle Type : Dog\rangle$. This is justifiable as long as the properties expressed by these algorithms are simple one-place predicates (e.g., being a dog), but when logically more complex descriptions are involved, the potential advantages of "serious" KR become overwhelming. (This point will become clearer in later sections.) This realisation is now motivating a modest new line of research which stresses logical and computational issues, asking what properties a KR framework needs to make it suitable to generate all the referring expressions that people can produce (and to generate them in reasonable time). In this new line of work, which is proceeding in tandem with the more empirically oriented work mentioned above, issues of human-likeness are temporarily put on the backburner. These and other empirical issues will be brought to bear once it is better understood what types of KR system are best suitable for GRE, and what is the best way to pursue GRE in them.

A few proposals have started to combine GRE with KR. Following on from work based on labelled directed graphs (cf. (Krahmer et al., 2003)) – a well-understood mathematical formalism that offers no reasoning support – (Croitoru and van Deemter, 2007) analysed GRE as a *projection* problem in Conceptual Graphs. More recently, (Areces et al., 2008) analysed GRE as a problem in Description Logic (DL), a formalism which, like Conceptual Graphs, is specifically designed for representing and reasoning with potentially complex information. The idea is to produce a formula such as $Dog \sqcap \exists love.Cat$ (the set of dogs intersected with the set of objects that love at least one cat); this is, of course, a successful reference if there exists *exactly one* dog who loves at least one cat. This approach forms the starting point for the present paper, which aims to show that when a principled, logic based approach is chosen, it becomes possible to refer to objects which no existing approach to GRE (including that of Areces et al.) has been able to refer to. To do this, we deviate substantially from these earlier approaches. For example, while Areces et al. use one finite interpretation for model checking, we consider arbitrary (possibly infinite) interpretations, hence reasoning support becomes necessary.

We shall follow many researchers in focussing on the semantic core of the GRE problem: we shall generate descriptions of semantic content, leaving the decision of what words to use for expressing this content (e.g., 'the ancient dog', or 'the dog which is old') to later stages in the NLG pipeline. Furthermore, we assume that all domain objects are equally salient (Krahmer and Theune, 2002). As explained above, we do not consider here the important matter of the naturalness or efficacy of the descriptions generated. We shall be content producing uniquely referring expressions whenever such expressions are possible, leaving the choice of the *optimal* referring expression in each given situation for later.

In what follows, we start by explaining how DL has been applied in GRE before (Sec. 2) , pointing out the limitations of existing work. In Sec.3 we discuss which kinds of additional expressivity are required and how they can be achieved through modern DL. In Sec.4 we present a generic algorithm to compute these expressive REs. Sec.5 concludes the paper by comparing its aims and achievements with current practise in GRE.

2 DL for GRE

2.1 Description Logics

Description Logic (DLs) come in different flavours, based on decidable fragments of first-order logic. A DL-based KB represents the domain with descriptions of concepts, relations, and their instances. DLs underpin the Web Ontology Language (OWL), whose latest version, OWL2 (Motik et al., 2008), is based on DL $\mathcal{SROIQ}$ (Horrocks et al., 2006).

An $\mathcal{SROIQ}$ ontology Σ usually consists of a TBox $\mathcal{T}$ and an ABox $\mathcal{A}$. $\mathcal{T}$ contains a set of concept inclusion axioms of the form $C \sqsubseteq D$, relation inclusion axioms such as $R \sqsubseteq S$ (the relation R is contained in the relation S), $R_1 \circ \ldots \circ R_n \sqsubseteq S$, and possibly more complex information, such as the fact that a particular relation is functional, or symmetric; $\mathcal{A}$ contains axioms about individuals, e.g. $a : C$ (a is an instance of C), $(a, b) : R$ (a has an R relation with b).

Given a set of atomic concepts, the entire set of concepts expressible by $\mathcal{SROIQ}$ is defined recursively. First, all atomic concepts are concepts. Furthermore, if C and D are concepts, then so are $\top \mid \bot \mid \neg C \mid C \sqcap D \mid C \sqcup D \mid \exists R.C \mid \forall R.C \mid \leq nR.C \mid \geq nR.C \mid \exists R.Self \mid \{a_1, \ldots, a_n\}$, where $\top$ is the top concept, $\bot$ the bottom concept, n a non-negative integer number, $\exists R.Self$ the self-restriction ((i.e., the set of those x such that $(x, x) : R$ holds)), a_i individual names and R a relation which can either be an atomic relation or the inverse of another relation (R^-). We call a set of individual names $\{a_1, \ldots, a_n\}$ a *nominal*, and use CN, RN and IN to denote the set of atomic concept names, relation names and individual names, respectively.

An *interpretation* $\mathcal{I}$ is a pair $\langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ where $\Delta^{\mathcal{I}}$ is a non-empty set and $\cdot^{\mathcal{I}}$ is a function that maps atomic concept A to $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$, atomic role r to $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ and individual a to $a^{I} \in \Delta^{\mathcal{I}}$. The interpretation of complex concepts and axioms can be defined inductively based on their semantics, e.g. $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$, etc.

$\mathcal{I}$ is a *model* of Σ, written $\mathcal{I} \models \Sigma$, *iff* all the axioms in Σ are satisfied in $\mathcal{I}$. It should be noted that one Σ can have multiple models. For example when $\mathcal{T} = \emptyset, \mathcal{A} = \{a : A \sqcup B\}$, there can be a model $\mathcal{I}_1$ s.t. $\Delta^{\mathcal{I}_1} = \{a\}$, $a^{\mathcal{I}_1} = a, A^{\mathcal{I}_1} = \{a\}, B^{\mathcal{I}_1} = \emptyset$, and another model $\mathcal{I}_2$ s.t. $\Delta^{\mathcal{I}_2} = \{a\}$, $a^{\mathcal{I}_2} = a, B^{\mathcal{I}_2} = \{a\}, A^{\mathcal{I}_2} = \emptyset$. In other words, the world is open. For details, see

(Horrocks et al., 2006).

The possibly multiple models indicate that an ontology is describing an open world. In GRE, researchers usually impose a closed world. From the DL point of view, people can (partially) close the ontology with a DBox $\mathcal{D}$ (Seylan et al., 2009), which is syntactically similar to the ABox, except that $\mathcal{D}$ contains only atomic formulas. Furthermore, every concept or relation appearing in $\mathcal{D}$ is closed. Its extension is exactly defined by the contents of $\mathcal{D}$, i.e. if $D \not\models a : A$ then $a : \neg A$, thus is the same in all the models. The concepts and relations not appearing in $\mathcal{D}$ can still remain open. DL reasoning can be exploited to infer implicit information from ontologies. For example, given $\mathcal{T} = \{Dog \sqsubseteq \exists feed^{-}.Woman\}$ (*every dog is fed by some woman*) and $\mathcal{A} = \{d1 : Dog, w1 : Woman\}$, we know that there must be some $Woman$ who feeds $d1$. When the domain is closed as $\mathcal{D} = \mathcal{A}$ we can further infer that this $Woman$ is $w1$ although there is no explicit relation between $w1$ and $d1$. Note that the domain $\Delta^{\mathcal{I}}$ in an interpretation of $\mathcal{D}$ is not fixed, but it includes all the DBox individuals.

However, closing ontologies by means of the DBox can restrict the usage of implicit knowledge (from $\mathcal{T}$). More precisely, the interpretations of the concepts and relations appearing in $\mathcal{D}$ are fixed therefore no implicit knowledge can be inferred. To address this issue, we introduce the notion of NBox to support Negation as Failure (NAF): Under NAF, an ontology is a triple $\mathcal{O} = (\mathcal{T}, \mathcal{A}, \mathcal{N})$, where $\mathcal{T}$ is a TBox, $\mathcal{A}$ an ABox and $\mathcal{N}$ is a subset of $CNorRN$. We call $\mathcal{N}$ an NBox. NAF requires that $\mathcal{O}$ satisfy the following conditions:

1. Let $x \in IN$ and $A \in \mathcal{N} \sqcap CN$. Then $(\mathcal{T}, \mathcal{A}) \not\models x : A$ implies $O \models x : \neg A$.
2. Let $x, y \in IN$ and $r \in \mathcal{N} \sqcap RN$. Then $(\mathcal{T}, \mathcal{A}) \not\models (x, y) : r$ implies $O \models (x, y) : \neg r$.

Like the DBox approach, the NBox $\mathcal{N}$ defines conditions in which "unknown" should be treated as "failure". But, instead of hard-coding this, it specifies a vocabulary on which such treatment should be applied. Different from the DBox approach, inferences on this NAF vocabulary is still possible. An example of inferring implicit knowledge with NAF will be shown in later sections.

2.2 Background Assumptions

When applying DL to GRE, people usually impose the following assumptions.

- Individual names are not used in REs. For example, "the Woman who feeds $d1$" would be invalid, because $d1$ is a name. Names are typically outlawed in GRE because, in many applications, many objects do not have names that readers/hearers would be familiar with.
- *Closed World Assumption (CWA)*: GRE researchers usually assume a closed world, without defining what this means. As explained above, DL allows different interpretations of the CWA. Our solution does not depend on a specific definition of CWA. In what follows, however, we use NAF to illustrate our idea. Furthermore, the domain is usually considered to be finite and consists of only individuals appearing in $\mathcal{A}$.
- *Unique Name Assumption (UNA)*: Different names denote different individuals. If, for example, $w1$ and $w2$ may potentially be the same woman, then we can not distinguish one from the other.

We follow these assumptions when discussing existing works and presenting our approach. In addition, we consider the entire KB, including $\mathcal{A}$, $\mathcal{T}$ and $\mathcal{N}$. It is also worth mentioning that, in the syntax of $\mathcal{SROIQ}$, negation of relations are not allowed in concept expressions, e.g. you cannot compose a concept $\exists\neg feed.Dog$. However, if $feed \in \mathcal{N}$, then we can interpret $(\neg feed)^{\mathcal{I}} = \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \setminus feed^{\mathcal{I}}$. In the rest of the paper, we use this as syntactic sugar.

2.3 Motivation: DL Reasoning and GRE

Every DL concept can be interpreted as a set. If the KB allows one to prove that this set is a singleton then the concept is a referring expression. It is this idea (Gardent and Striegnitz, 2007) that (Areces et al., 2008) explored. In doing so, they say little about the TBox, appearing to consider only the ABox, which contains only axioms about instances of atomic concepts and relations. For example, the domain in Fig.1 can be described as

KB1: $\mathcal{T}_1 = \emptyset$, $\mathcal{A}_1 = \{w1 : Woman$, $w2 : Woman$, $d1 : Dog$, $d2 : Dog$, $c1 : Cat$, $c2 : Cat$, $(w1, d1) : feed$,

$(w2, d1) : feed$, $(w2, d2) : feed$, $(d1, c1) : love\}$

Assuming that this represents a Closed World, Areces et al. propose an algorithm that is able to generate descriptions by partitioning the domain.[1] More precisely, the algorithm first finds out which objects are describable through increasingly large conjunctions of (possibly negated) atomic concepts, then tries to extend these conjunctions with complex concepts of the form $(\neg)\exists R1.Concept$, then with concepts of the form $(\neg)\exists R2.(Concept \sqcap (\neg)\exists R1.Concept)$, and so on. At each stage, only those concepts that have been acceptable through earlier stages are used. Consider, for instance, KB1 above. Regardless of what the intended referent is, the algorithm starts partitioning the domain stage by stage as follows. Each stage makes use of all previous stages. During stage (3), e.g., the object $w2$ could only be identified because $d2$ was identified in stage (2):

1. $Dog = \{d1, d2\}$,
   $\neg Dog \sqcap Woman = \{w1, w2\}$,
   $\neg Dog \sqcap \neg Woman = \{c1, c2\}$.
2. $Dog \sqcap \exists love.(\neg Dog \sqcap \neg Woman) = \{d1\}$,
   $Dog \sqcap \neg\exists love.(\neg Dog \sqcap \neg Woman) = \{d2\}$.
3. $(\neg Dog \sqcap Woman) \sqcap \exists feed.(Dog \sqcap \neg\exists love.(\neg Dog \sqcap \neg Woman)) = \{w2\}$,
   $(\neg Dog \sqcap Woman) \sqcap \neg\exists feed.(Dog \sqcap \neg\exists love.(\neg Dog \sqcap \neg Woman)) = \{w1\}$.

As before, we disregard the important question of the quality of the descriptions generated, other than whether they do or do not identify a given referent uniquely. Other aspects of quality depend in part on details, such as the question in which order atomic concepts are combined during phase (1), and analogously during later phases.

However this approach does not extend the expressive power of GRE. This is not because of some specific lapse on the part of the authors: it seems to have escaped the GRE community as a whole that relations can enter REs in a variety of alternative ways.

Furthermore, the above algorithm considers only the ABox, therefore background information will not be used. It follows that the domain always has a fixed single interpretation/model. Consequently the algorithm essentially uses model-checking, rather than full reasoning. We will show that when background information is involved, reasoning has to be taken into account. For example, suppose we extend Fig.1 with background (i.e., TBox) knowledge saying that *one should always feed any animal loved by an animal whom one is feeding*, while also adding a love edge (Fig.2) between $d2$ and $c2$:

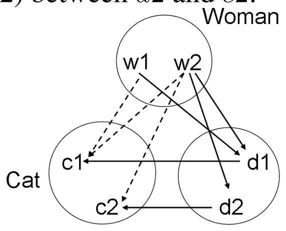

Figure 2: An extended example of Fig.1. Edges from women to cats denote $feed$ relations. Dashed edges denote implicit relations.

If we close the domain with NAF, the ontology can be described as follows:

KB2: $\mathcal{T}_2 = \{feed \circ love \sqsubseteq feed\}$, $\mathcal{A}_2 = \mathcal{A}_1 \cup \{(d2, c2) : love\}$, $\mathcal{N}_2 = \{Dog, Woman, feed, love\}$

The TBox axiom enables the inference of implicit facts: the facts $(w1, c2) : feed$, $(w2, c1) : feed$, and $(w2, c2) : feed$ can be inferred using DL reasoning under the above NBox $\mathcal{N}_2$. Axioms of this kind allow a much more natural, insightful and concise representation of information than would otherwise be possible.

Continuing to focus on the materialised KB2, we note another limitation of existing works: if only existential quantifiers are used then some objects are unidentifiable (i.e., it is not possible to distinguish them uniquely). These objects would become identifiable if other quantifiers and inverse relations were allowed. For example,

- The cat which is fed by *at least 2* women = $Cat \sqcap \geq 2 feed^{-}.Woman = \{c1\}$,
- The woman feeding *only* those fed by *at least 2* women = $Woman \sqcap \forall feed. \geq 2.feed^{-}.Woman = \{w1\}$,
- The woman who feeds *all* the dogs = $\{w2\}$.

It thus raises the question: which quantifiers would it be natural to use in GRE, and how might DL realise them?

[1] Areces et al. (Areces et al., 2008) consider several DL fragments (e.g., $\mathcal{ALC}$ and $\mathcal{EL}$). Which referring expressions are expressible, in their framework, depends on which DL fragment is chosen. Existential quantification, however, is the only quantifier that was used, and inverse relations are not considered.

3 Beyond Existential Descriptions

In this section, we show how more expressive DLs can make objects referable that were previously unreferable. This will amount to a substantial reformulation which allows the approach based on DL reasoning to move well beyond other GRE algorithms in its expressive power.

3.1 Expressing Quantifiers in OWL2

Because the proposal in (Areces et al., 2008) uses only existential quantification, it fails to identify *any* individual in Fig.2. Before filling this gap, we pause to ask what level of expressivity ought to be achieved. In doing so, we make use of a conceptual apparatus developed in an area of formal semantics and mathematical logic known as the theory of Generalized Quantifiers (GQ), where quantifiers other than *all* and *some* are studied (Mostowski, 1957). The most general format for REs that involves a relation `R` is, informally, `the N1 who R Q N2′s`, where `N1` and `N2` denote sets, R denotes a relation, and `Q` a generalized quantifier. (Thus for example `the women who feed SOME dogs`.) An expression of this form is a unique identifying expression if it corresponds to exactly one domain element. Using a set-theoretic notation, this means that the following set has a cardinality of 1:

$$\{y \in N1 : Qx \in N2 \mid Ryx\}$$

where Q is a generalized quantifier. For example, if Q is the existential quantifier, while $N1$ denotes the set of women, $N2$ the set of dogs, and R the relation of feeding, then this says that the number of women who feed SOME dog is one. If Q is the quantifier *at least two*, then it says that the number of women who feed at least two dogs is one. It will be convenient to write the formula above in the standard GQ format where quantifiers are cast as relations between sets of domain objects A, B. Using the universal quantifier as an example, instead of writing $\forall x \in A \mid x \in B$, we write $\forall(AB)$. Thus, the formula above is written

$$\{y \in N1 : Q(N2\{z : Ryz)\}\}.$$

Instantiating this as before, we get $\{y \in Woman : \exists(Dog\{z : Feed\ yz)\}\}$, or "women who feed a dog", where Q is $\exists$, $A = Dog$ and $B = \{z : Feed\ yz\}$ for some y.

Mathematically characterising the class of *all* quantifiers that can be expressed in referring expressions is a complex research programme to which we do not intend to contribute here, partly because this class includes quantifiers that are computationally problematic; for example, a quantifiers such as *most* (in the sense of more than 50%), which is not first-order expressible, as is well known.

To make transparent which quantifiers are expressible in the logic that we are using, let us think of quantifiers in terms of simple quantitative constraints on the sizes of the sets $A \cap B$, $A - B$, and $B - A$, as is often done in GQ theory, asking what types of constraints can be expressed in referring expressions based on $\mathcal{SROIQ}$. The findings are summarised in Tab.1. OWL2 can express any of the following types of descriptions, plus disjunctions and conjunctions of anything it can express.

Table 1: Expressing GQ in DL

	QAB	DL
1	$\geq nN2\{z : Ryz\}$	$y :\geq nR.N2$
2	$\geq nN2\neg\{z : Ryz\}$	$y :\geq n\neg R.N2$
3	$\geq n\neg N2\{z : Ryz\}$	$y :\geq nR.\neg N2$
4	$\geq n\neg N2\neg\{z : Ryz\}$	$y :\geq n\neg R.\neg N2$
5	$\leq nN2\{z : Ryz\}$	$y :\leq nR.N2$
6	$\leq nN2\neg\{z : Ryz\}$	$y :\leq n\neg R.N2$
7	$\leq n\neg N2\{z : Ryz\}$	$y :\leq nR.\neg N2$
8	$\leq n\neg N2\neg\{z : Ryz\}$	$y :\leq n\neg R.\neg N2$

When $n = 1$, for example, type 1 becomes $\exists R.N2$, i.e. the *existential* quantifier. When $n = 0$ type 7 becomes $\forall R.N2$, i.e. the quantifier *only*. When $n = 0$ type 6 becomes $\forall\neg R.\neg N2$, i.e. the quantifier *all*. In types 2, 4, 6 and 8, negation of a relation is used. This is not directly supported in $\mathcal{SROIQ}$ but, as we indicated earlier, given $R \in \mathcal{N}$, $\neg R$ can be used in concepts.

Together, this allows the expression of a description such as "women who feed at least one but at most 7 dogs", by conjoining type 1 (with $n = 1$) with type 5 (with $n = 7$). Using negation, it can say "women who do not feed all dogs and who feed at least one non-dog" ($Woman \sqcap \neg\forall\neg Feed.\neg Dog \sqcap \exists Feed.\neg Dog$). In addition to Tab.1, $\mathcal{SROIQ}$ can even represent reflexive relation such as "the dog who loves itself" by $Dog \sqcap \exists love.Self$, which was regarded infeasible in (Gardent and Striegnitz, 2007).

Comparing the quantifiers that become expressible through OWL2's apparatus with classes of quantifiers studied in the theory of GQ, it is clear that OWL2 is highly expressive: it does not only include quantifiers expressible in the binary tree of numbers, e.g. (van Benthem, 1986) – which is generally regarded as highly general – but much else besides. Even wider classes of referring expressions can certainly be conceived, but these are not likely to have overwhelming practical utility in today's NLG applications.

4 Generating $\mathcal{SROIQ}$-enabled REs

In this section, we present an algorithm that computes the descriptions discussed in sect.3. A GRE algorithm should have the following behaviour: if an entity is distinguishable from all the others, the algorithm should find a unique description; otherwise, the algorithm should say there exists no unique description. In this paper, we follow Areces et al.'s strategy of generating REs for all objects simultaneously, but we apply it to a much larger search space, because many more constructs are taken into account.

4.1 GROWL: an algorithm for Generating Referring expressions using OWL-2.

In this section we show how the ideas of previous sections can be implemented. To do this, we sketch an algorithm scheme called GROWL. GROWL applies a generate-and-test strategy that composes increasingly complicated descriptions and uses DL reasoning to test whether a description denotes a singleton w.r.t. the KB. To avoid considering unnecessarily complicated descriptions, the algorithm makes use of the (syntactic) depth of a description, defined as follows:

Definition 1 ***(Depth)*** *Given a description d, its depth $|d|$ is calculated as follows:*

1. $|d| = 1$ *for* $d := \top|\bot|A|\neg A$, *where* A *is atomic.*
2. $|d \sqcap d'| = |d \sqcup d'| = max(|d|, |d'|) + 1$.
3. $|\exists r.d| = |\forall r.d| = |\leq nr.d| = |\geq nr.d| = |= nr.d| = |d| + 1$.

Different descriptions can mean the same of course, e.g. $\neg\forall R.A \equiv \exists R.\neg A$. We do not know which syntactic variant should be used but focus, for simplicity, on generating their unique *negated normal form* (NNF). The NNF of a formula ϕ can be obtained by pushing all the $\neg$ inward until only before atomic concepts (including $\top$ and $\bot$), atomic relations, nominals or self restrictions (e.g. $\exists r.Self$). Without loss of generality, in what follows we assume all the formulas are in their NNF. To avoid confusion, the NNF of negation of a formula ϕ is denoted by $\sim\phi$ instead of $\neg\phi$. For example $\sim(A \sqcup B) = \neg A \sqcap \neg B$ if A and B are atomic. Obviously, $\sim(\sim A) = A$, $\sim(\sim R) = R$, $(R^-)^- = R$, and $(\sim R)^- = \sim R^-$. The use of NNF substantially reduces the redundancies generated by the algorithm. For example, we won't generate both $\neg\forall R.A$ and $\exists R.\neg A$ but only the later.

Given an ontology Σ, we initialise GROWL with the following sets:

1. The relation name set RN is the minimal set satisfying:
   - if R is an atomic relation in Σ, then $R \in RN$;
   - if $R \in RN$, then $\sim R \in RN$;
   - if $R \in RN$, then $R^- \in RN$;
2. The concept name set CN is the minimal set satisfying:
   - $\top \in CN$;
   - if A is an atomic concept in Σ, then $A \in CN$;
   - if $R \in RN$, then $\exists R.Self \in CN$;
   - if $A \in CN$, then $\sim A \in CN$;
3. The natural number set N contains $1, 2, \ldots, n$ where n is the number of individuals in Σ.
4. The construct set S contains all the constructs supported by a particular language. For $\mathcal{SROIQ}$, $S = \{\neg, \sqcap, \sqcup, \exists, \forall, \leq, \geq, =\}$. We assume here that nominals are disallowed (cf. sect.2).

Algorithm GROWL:
$Construct - description(\Sigma, CN, RN, N, S)$
INPUT: Σ, CN, RN, N, S
OUTPUT: Description Queue D

```
1: $D := \emptyset$
2: for $e \in CN$ do
3:     $D := Add(D, e)$
4: for $d = fetch(D)$ do
5:     for each $s \in S$ do
6:         if $s = \sqcap$ or $s = \sqcup$ then
7:             for each $d' \in D$ do
8:                 $D := Add(D, d\ s\ d')$
9:         if $s = \exists$ or $s = \forall$ then
```

```
10:         for each $r \in RN$ do
11:             $D := Add(D, s\ r.d)$
12:     if $s = \leq$ or $s = \geq$ or $s\ is\ =$ then
13:         for each $r \in RN$, each $k \in N$ do
14:             $D := Add(D, s\ k\ r.d)$
15: return $D$
```

Algorithm ADD:$Add(D, e)$
INPUT: D, e
OUTPUT: (Extended) Description Queue D

```
1: for $d \in D$ do
2:     if $|d| < |e|$ and $d \sqsubseteq_\Sigma e$ then
3:         return $D$
4:     else if $|d| = |e|$ and $d \sqsubseteq_\Sigma e$ and $e \sqcap \neg d$ is satisfiable then
5:         return $D$
6: if $e$ is satisfiable in $\Sigma$ then
7:     $D := D \cup \{e\}$
8: return $D$
```

GROWL takes an ontology Σ as its input and output a queue D of descriptions by adding increasingly complex concepts e to D, using the function $Add(D, e)$, which is implemented as the algorithm ADD. Because of the centrality of ADD we start by explaining how this function works.

In the simple algorithm we are proposing in this paper – which represents only one amongst many possibilities – addition is governed by the heuristic that *more complex descriptions should have smaller extensions.* To this end, a candidate description e is compared with each existing description $d \in D$. Step 2 ensures that if there exists a simpler description d ($|d| < |e|$) whose extension is no larger than e ($d \sqsubseteq_\Sigma$ e), then e is not added into D (because the role of e can be taken by the simpler description d). Similarly, step 4 ensures that if there exists d with same depth ($|d| = |e|$) but smaller extension ($d \sqsubseteq_\Sigma e$ and $e \sqcap \neg d$ is satisfiable), then e should not be added into D. The subsumption checking in Step 2 and 4, and the instance retrieval in Step 6, must be realised by DL reasoning, in which TBox, ABox and NBox must all be taken into account. ADD guarantees that when the complexity of descriptions increases, their extensions are getting smaller.

We now turn to the main algorithm, GROWL. In Step 1 of this algorithm, D is initialised to $\emptyset$. Steps 2 to 3 add all satisfiable elements of CN to D. From Steps 4 to 14, we recursively "process" elements of D one by one, by which we mean that the constructors in S are employed to combine these elements with other elements of D (e.g., an element is intersected with all other elements, and so on). We use $fetch(D)$ to retrieve the first unprocessed element of D. New elements are added to the end of D. Thus D is a first-come-first-served queue (note that processed elements are not removed from D).

To see in more detail how elements of D are processed, consider Steps 5-14 once again. For each element d of D, Step 5 uses a construct s to extend it:

1. If s is $\sqcap$ or $\sqcup$, in Step 7 and 8, we extend d with each element of D and add new descriptions to D.

2. If s is $\exists$ or $\forall$, in Step 10 and 11, we extend d with all relations of RN and add new descriptions to D. In Areces et el.'s work, $\forall$ is also available when using $\neg$ and $\exists$ together, however due to their algorithm they can never generates descriptions like $\forall r.A$.

3. If s is $\leq, \geq$ or $=$, in Step 13 and 14, we extend d with all relations in RN and all numbers in N, and add new descriptions to D.

   Because the $=$ construct can be equivalently substituted by the combination of $\leq, \geq$ and $\sqcap$ constructs ($= kr.d$ is semantically equivalent to $\geq kr.d \sqcap \leq kr.d$), it is a modelling choice to use either $\leq, \geq$, or only $=$, or all of them. In this algorithm we use them all.

Because we compute only the NNF and we disallow the use of individual identifiers, negation $\neg$ appears only in front of atomic concept names. For this reason, processing does not consider $s = \neg$. Note that GROWL leaves some important issues open. In particular, the order in which constructs, relations, integers and conjuncts/disjuncts are chosen is left unspecified. Note that D, RN, N, S are all assumed to be finite, hence Steps 5 to 14 terminate for a given $d \in D$. Because Steps 5 to 14 generate descriptions whose depth increases with one constructor at a time, there are finitely many $d \in D$ such that $|d| = n$ (for a given n).

GROWL extends the algorithm presented by Areces et al. The example in Fig.2 shows that many referring expressions generated by our algorithm cannot be generated by our predecessors; in fact, some objects that are not referable for them are referable by GROWL. For example, if we apply the algorithm to the KB in Fig.2, a possible solution is as follows:

1. $\{w1\} = Woman \sqcap \exists \neg feed.Cat$, the woman that does not feed all cats.
2. $\{w2\} = \leq 0 \neg feed.Cat$, the woman that feeds all cats.
3. $\{d1\} = Dog \sqcap \leq 0 \neg feed^{-}.Woman$, the dog that is fed by all women.
4. $\{d2\} = Dog \sqcap \exists \neg feed^{-}.Woman$, the dog that is not fed by all women.
5. $\{c1\} = Cat \sqcap \leq 0 \neg feed^{-}.Woman$, the cat that is fed by all women.
6. $\{c2\} = Cat \sqcap \exists \neg feed^{-}.Woman$, the cat that is not fed by all women.

It is worth reiterating here that our algorithm focusses on finding uniquely referring expressions, leaving aside which of all the possible ways in which an object can be referred to is "best". For this reason, empirical validation of our algorithm – a very sizable enterprise in itself, which should probably be based on descriptions elicited by human speakers – is not yet in order.

4.2 Discussion

Let us revisit the basic assumptions of Sec.2.2, to see what can be achieved if they are abandoned.

1. In natural language, people do using names, e.g. "the husband of Marie Curie". To allow REs of this kind, we can extend our Algorithm A-1 by including singleton classes such as $\{Maria\_Curie\}$ in CN.
2. Traditional GRE approaches have always assumed a single model with complete knowledge. Without this assumption, our approach can still find interesting REs. For example, if a man's nationality is unknown, but he is known to be the Chinese or Japanese, we can refer to him/her as $Chinese \sqcup Japanese$. However, models should be finite to guarantee that N is finite.
3. Individuals with multiple names. DL imposes the UNA by explicitly asserting the inequality of each two individuals. Without UNA, reasoning can still infer some results, e.g. $\{Woman \sqcap Man \sqsubseteq \bot, David : Man, May : Woman\} \models David \neq May$. Thus we can refer to David as "the man" if the domain is closed.

5 Widening the remit of GRE

This paper has shown some of the benefits that arise when the power of KR is brought to bear on an important problem in NLG, namely the generation of referring expressions (GRE). We have done this by using DL as a representation and reasoning formalism, extending previous work in GRE in two ways. First, we have extended GRE by allowing the generation of REs that involve quantifiers other than $\exists$. By relating our algorithm to the theory of Generalised Quantifiers, we were able to formally characterise the set of quantifiers supported by our algorithm, making exact how much expressive power we have gained. Secondly, we have demonstrated the benefits of *implicit* knowledge through inferences that exploit TBox-information, thereby allowing facts to be represented more efficiently and elegantly, and allowing GRE to tap into kinds of generic (as opposed to atomic) knowledge that it had so far left aside, except for hints in (Gardent and Striegnitz, 2007) and in (Croitoru and van Deemter, 2007). Thirdly, we have allowed GRE to utilise incomplete knowledge, as when we refer to someone as "the man of Japanese or Chinese nationality".

Current work on reference is overwhelmingly characterised by an emphasis on empirical accuracy, often focussing on very simple referring expressions, which are constituted by conjunctions of 1-place relations (as in "the grey poodle"), and asking which of these conjunctions are most likely to be used by human speakers (or which of these would be most useful to a human hearer). The present work stresses different concerns: we have focussed on questions of expressive power, focussing on relatively complex descriptions, asking what referring expressions are possible when relations between domain objects are used. We believe that, at the present stage of work in GRE, it is of crucial importance to gain insight into questions of this kind, since this will tell us what types of reference are possible in principle. Once such insight, we hope to explore how the newly gained expressive power can be put to practical use.

References

Douglas Appelt. 1985. *Planning English Sentences*. Cambridge University Press, Cambridge, UK.

Carlos Areces, Alexander Koller, and Kristina Striegnitz. 2008. Referring expressions as formulas of description logic. In *Proceedings of the 5th INLG*, Salt Fork, Ohio.

Anja Belz and Albert Gatt. 2008. Intrinsic vs. extrinsic evaluation measures for referring expression generation. In *HLT '08: Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies*, pages 197–200.

Madalina Croitoru and Kees van Deemter. 2007. A conceptual graph approach to the generation of referring expressions. In *Proceedings of the 20th IJCAI*.

Robert Dale and Ehud Reiter. 1995. Computational interpretations of the gricean maxims in the generation of referring expressions. *CoRR*, cmp-lg/9504020.

Claire Gardent and Kristina Striegnitz. 2007. Generating bridging definite descriptions. *Computing Meaning*, 3:369–396.

Albert Gatt, Anja Belz, and Eric Kow. 2009. The TUNA-REG Challenge 2009: Overview and evaluation results. In *Proceedings of the 12th ENLG (ENLG 2009)*, pages 174–182, Athens, Greece, March. Association for Computational Linguistics.

Ian Horrocks, Oliver Kutz, and Ulrike Sattler. 2006. The Even More Irresistible SROIQ. In *KR 2006*.

Emiel Krahmer and Mariet Theune. 2002. Efficient context-sensitive generation of descriptions in context. *Information Sharing: Givenness and Newness in Language*, pages 223–264.

Emiel Krahmer, Sebastiaan van Erk, and Andr Verleg. 2003. Graph-based generation of referring expressions. *Computational Linguistics*, 29(1):53–72.

A Mostowski. 1957. On a generalization of quantifiers. *Fund. Math.*, 44:235–273.

Boris Motik, Bernardo Cuenca Grau, Ian Horrocks, Zhe Wu, Achille Fokoue, and Carsten Lutz. 2008. Owl 2 web ontology language: Profiles. W3c working draft, W3C, October.

Inanç Seylan, Enrico Franconi, and Jos de Bruijn. 2009. Effective query rewriting with ontologies over dboxes. In *IJCAI 2009*.

Johan van Benthem. 1986. Essays in Logical Semantics. Reidel.