Revisiting Grounded Circumscription in Description Logics

Stathis Delivorias[2(✉)] and Sebastian Rudolph[1]

[1] Theoretical Computer Science, TU Dresden, Dresden, Germany
Sebastian.Rudolph@tu-dresden.de
[2] University of Montpellier, LIRMM, Montpellier, France
Delivorias@lirmm.fr

Abstract. Circumscription is a paradigm of non-monotonic logic meant to formalize the common-sense understanding that, among competing theories that represent phenomena equally well, the one with the fewest "abnormal" assumptions should be selected. Several papers have considered ways of adding circumscription to Description Logics. One of the proposals with good computational properties is Grounded Circumscription, introduced by Sengupta, Krishnadi and Hitzler in 2011. Our paper builds on their general idea, but identifies some problems with the original semantics definition, which gives rise to counter-intuitive consequences and renders the proposed tableau algorithm incorrect. We give an example that makes the problem explicit and propose a modification of the semantics that remedies this issue. On the algorithmic side, we show that a big part of the reasoning can actually be transferred to standard Description Logics, for which tools and results already exist.

1 Introduction

Circumscription is a paradigm of non-monotonic logic introduced by John McCarthy in 1980 [5]. The main idea is to formalize the common sense understanding that among competing theories that predict equally well, the one with the fewest assumptions should be selected. This is basically an application of the principle known as "Occam's razor" to logic. It is also similar to the closed world assumption, where what is not known to be true is taken to be false. In its original first-order logic formulation, circumscription minimizes the extension of some predicates, where the extension of a predicate is the set of tuples of values the predicate is true on.

Description Logics (DLs) are knowledge representation formalisms designed to describe and reason about qualitative properties and conceptual aspects of a system [1,6]. Ontology languages based on DLs have been widely adopted in a large class of application areas. One of the most prominent applications of DLs is to provide the underlying logical basis of the web ontology language OWL 2, which is the current recommendation of the World Wide Web Consortium (W3C) [4,8]. Therein DLs are used to represent the intended meaning of Web resources and establish powerful reasoning tools, so as to facilitate machine

© Springer International Publishing Switzerland 2016
M. Ortiz and S. Schlobach (Eds.): RR 2016, LNCS 9898, pp. 46–60, 2016.
DOI: 10.1007/978-3-319-45276-0_5

understandability of Web pages. From a more scholarly perspective, DLs are decidable fragments of first order logic.

Description Logics traditionally operate within the monotonic realm, namely the addition of more assertions to a knowledge base does not negate previously inferred information. But in many prevalent application domains, such as common sense reasoning, this property does not hold. Conclusions might need to be revised in the light of new information. Hence it is quite intriguing to try to develop a DL framework where reasoning would be non-monotonic. There have been notable efforts to define circumscription for DLs, albeit with rather high complexity or even undecidable if roles are circumscribed [2].

In this essay we aim to fuse Description Logics, with a restricted version of Circumscription, called Grounded Circumscription. The work is based on a 2011 publication by K. Sengupta, A.A. Krisnadhi and P. Hitzler, which throughout this work we will refer to as "the original paper" [7]. In ground circumscription, some of the predicates in our language (which in DL can only be unary or binary) are chosen to be grounded and minimized. Grounded means that their interpretations must include only named individuals, i.e. elements of the domain that correspond to one of the constants that appear in our knowledge base. Moreover, those predicates are minimized in the sense that we accept only models which assign as few individuals as possible to them, so that there cannot be a model whose extensions of these predicates are subsets of the respective extensions in the minimal model.

In the original paper, the main idea of grounded circumscription is given along with algorithms for certain decision problems. We have optimized and modified these ideas. The optimization was our initial aim, in particular we wanted (and largely achieved) to transfer a big part of the reasoning to standard DLs, for which there already exist tools and available results. But in the process we uncovered some insufficiencies in the notion of minimality as introduced in the original paper, to the discussion of which Subsect. 3.1 is devoted. Hence we have modified the main definition to one that is more effective and more intuitive.

After introducing the particular DL formalism and terminology that we work on (Sect. 2), we specify the basic notions and proceed to present an algorithm for satisfiability (Sect. 3) which is predominantly in the monotonic sphere. We then introduce important notions which are put to use in the algorithm for entailment of facts (Sect. 4). Following are supplementary results that further develop the theory of grounded circumscription in DLs (Sect. 5) and finally we give an overview of the contribution of this endeavor and discuss prospects of further research (Sect. 6).

All proofs can be found in the original master thesis [3].

2 Preliminaries

In this section, we give a brief introduction to our formalism and the main terminology and ideas around it.

In the original paper, decidability of ground circumscription is proven using rather complex and non-standard languages which feature concept products, role

hierarchies and role disjunctions. Then, independently, algorithms which apply only to $\mathcal{ALC}$ are given. Our work is entirely based on the standard DL $\mathcal{ALCO}$ but it can trivially be extended to any more complex formalism that subsumes $\mathcal{ALCO}$.

$\mathcal{ALCO}$ Syntax. Let N_C, N_r and N_I be mutually disjoint sets of *concept-, role-* and *individual names*, respectively. Concepts C in $\mathcal{ALCO}$ are built using the grammar rule:

$$C ::= \top \mid A \mid \{a\} \mid \neg C \mid C \sqcap C \mid \exists r.C$$

where $A \in N_C$, $r \in N_r$, and $a \in N_I$. We employ the usual abbreviations: $\bot = \neg\top$, $C \sqcup D = \neg(\neg C \sqcap \neg D)$, and $\forall r.C = \neg\exists r.\neg C$.

An expression of the form $C \sqsubseteq D$, where C and D are concepts, is called a *general concept inclusion* (GCI). A finite set of GCIs is a *TBox*. An expression of the form $C(a)$, where C is a concept and $a \in N_I$, is called a *concept assertion*. For $r \in N_R$ and $a, b \in N_I$, an expression of the form $r(a, b)$ is called a *role assertion*. A finite set of concept and role assertions is called an *ABox*.

A pair $K = (\mathcal{T}, \mathcal{A})$ consisting of a TBox $\mathcal{T}$ and an ABox $\mathcal{A}$ is called a *knowledge base* (abbreviated frequently as KB). For ease of presentation, in this study we will usually understand a knowledge base as a single set of axioms, which would formally be expressed as $K = \mathcal{T} \cup \mathcal{A}$. We will not refer to Aboxes and Tboxes individually, rather we will handle the knowledge base as a whole.

$\mathcal{ALCO}$ Semantics. An *interpretation* is a pair $\mathcal{I} = (\Delta, \cdot^{\mathcal{I}})$, where Δ is a non-empty *domain* and $\cdot^{\mathcal{I}}$ is a function that maps every $a \in N_I$ to $a^{\mathcal{I}} \in \Delta$, every $A \in N_C$ to $A^{\mathcal{I}} \subseteq \Delta$, and every $r \in N_r$ to $r^{\mathcal{I}} \subseteq \Delta \times \Delta$. The mapping $\cdot^{\mathcal{I}}$ is naturally extended to all concepts by setting

$$\begin{aligned}
\top^{\mathcal{I}} &= \Delta, \\
(\neg C)^{\mathcal{I}} &= \Delta \setminus C^{\mathcal{I}}, \\
(C \sqcap D)^{\mathcal{I}} &= C^{\mathcal{I}} \cap D^{\mathcal{I}}, \\
\{a\}^{\mathcal{I}} &= \{a^{\mathcal{I}}\}, \\
(\exists r.C)^{\mathcal{I}} &= \{x \in \Delta \mid \exists y \in \Delta.\, (x, y) \in r^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}.
\end{aligned}$$

An interpretation $\mathcal{I}$ *satisfies*

- a concept inclusion $C \sqsubseteq D$ if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$,
- a concept assertion $C(a)$ if $a^{\mathcal{I}} \in C^{\mathcal{I}}$ and
- a role assertion $r(a, b)$ if $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in r^{\mathcal{I}}$.

We say that $\mathcal{I}$ is a *model* of a TBox $\mathcal{T}$ or an ABox $\mathcal{A}$ if it satisfies every concept inclusion in $\mathcal{T}$ or every assertion in $\mathcal{A}$, respectively. $\mathcal{I}$ is a model of a knowledge base $K = (\mathcal{T}, \mathcal{A})$ if $\mathcal{I}$ is a model of both $\mathcal{T}$ and $\mathcal{A}$. If there exists a model of a knowledge base K, then K is a *satisfiable* KB. If every model of K satisfies $C(a)$ (or $r(a, b)$, respectively), we say that $C(a)$ (or $r(a, b)$, respectively) is *entailed* by K. If every model of K satisfies $C \sqsubseteq D$, then C is *subsumed* by D with respect to K.

3 Grounded Circumscription

In this section we formally define the basic notions of ground circumscription. The definition of minimality is reestablished in solid grounds, which can prove a useful framework for further development of this theory.

Ground Extension. A central notion in this study is that of ground extension of a predicate with respect to a certain interpretation, which is the set of individual names or pairs of individual names (depending on whether the predicate is a concept or a role), whose interpretations belong to the interpretation of this predicate. Given a knowledge base K, the set of individual names that appear in K are symbolized $Ind(K)$.

Definition 1. Let K be an $\mathcal{ALCO}$ knowledge base and $\mathcal{I}$ an interpretation. The *ground extension* wrt $\mathcal{I}$ of a predicate $W \in N_C \cup N_r$, is the following set:

$$Ext^{\mathcal{I}}(W) := \begin{cases} \{a \in Ind(K) | a^{\mathcal{I}} \in W^{\mathcal{I}}\} & if\, W \in N_C, \\ \{(a,b) \in Ind(K) \times Ind(K) | (a^{\mathcal{I}}, b^{\mathcal{I}}) \in W^{\mathcal{I}}\} & if\, W \in N_r. \end{cases} \dashv$$

The key role that ground extension plays, is evident by its frequent presence throughout the rest of this work. $Ext^{\mathcal{I}}(\cdot)$ can be naturally extended to be applicable to any concept description: if C is a concept, then $Ext^{\mathcal{I}}(C) := \{a \in Ind(K) | a^{\mathcal{I}} \in C^{\mathcal{I}}\}$. Since nominals are valid concept constructors in our language, we can ultimately view $Ext^{\mathcal{I}}(C)$ as a concept description, provided that C is a concept as well. One more important property of ground extension, which is easy to verify, is that it is monotonic with respect to set inclusion, i.e. if $A^{\mathcal{I}} \subseteq B^{\mathcal{I}}$ then $Ext^{\mathcal{I}}(A) \subseteq Ext^{\mathcal{I}}(B)$.

Groundedness and Minimality. The main idea in grounded circumscription is to select some predicates (concept and role names), and demand that for every model their interpretation is grounded, i.e. it includes only named individuals, and that it is minimized, in the sense that there cannot be an interpretation that assigns fewer individuals to those predicates and still is a model of our given knowledge base.

Definition 2. Let K be a knowledge base and $M \subseteq N_C \cup N_r$. A model $\mathcal{I}$ of K is called *grounded* wrt M if

(i) $C^{\mathcal{I}} \subseteq \{b^{\mathcal{I}} | b \in Ind(K)\}$ for every $C \in M \cap N_C$.
(ii) $r^{\mathcal{I}} \subseteq \{(a^{\mathcal{I}}, b^{\mathcal{I}}) | a, b \in Ind(K)\}$ for every $r \in M \cap N_r$. $\dashv$

Definition 3. A *GC-$\mathcal{ALCO}$-KB* is a pair (K, M) where K is an $\mathcal{ALCO}$ knowledge base and $M \subseteq N_C \cup N_r$. Every $W \in M$ is said to be *closed* wrt K. Let $\prec_M$ denote a *"smaller than"* relation which is a partial order on the set of all interpretations for K. An interpretation $\mathcal{I}$ is a *GC-model* of (K, M) if it is a grounded model of K wrt M and $\mathcal{I}$ is minimal wrt $\prec_M$, i.e. there is no grounded model $\mathcal{J}$ of K such that $\mathcal{J} \prec_M \mathcal{I}$. (K, M) is *satisfiable* if it has a GC-model. A statement ϕ is a *logical consequence* of (K, M) if every GC-model of (K, M) satisfies ϕ. We then say that (K, M) *entails* ϕ. $\dashv$

Note that ϕ in the above definition could be a GCI, a concept assertion or a role assertion. Obviously, the precise semantics depends on the concrete choice of the "smaller than" relation $\prec_M$, which will be discussed in the next paragraph. Henceforth we will frequently substitute the term GC-model, with *minimal grounded model* or simply *minimal model*.

3.1 Discussion on the Modification of the GC-Definition

The original paper employed the following definition for the "smaller-than" relation.

Definition 4. Let (K, M) be a *GC-$\mathcal{ALCO}$-KB*. If $\mathcal{I}$ and $\mathcal{J}$ are interpretations of K then the *"smaller than"* relation is defined in the following way:
$\mathcal{I} \prec_M^{\text{orig}} \mathcal{J}$ if

(i) $\Delta^{\mathcal{I}} = \Delta^{\mathcal{J}}$ and $a^{\mathcal{I}} = a^{\mathcal{J}}$ for every $a \in Ind(K)$,
(ii) $W^{\mathcal{I}} \subseteq W^{\mathcal{J}}$ for every $W \in M$ and
(iii) there is a $W \in M$ such that $W^{\mathcal{I}} \subset W^{\mathcal{J}}$. ⊣

The first condition for the minimality relation in the original paper requires that two interpretations have equal domains in order for them to be comparable. Firstly, we argue that this is counter-intuitive. When we say that a model has fewer assumptions than another model, this does not imply any similarity of their domains, it rather requires that those predicates which are of importance to us are of smaller extension. Furthermore, the original definition imposes algorithmic problems as one will have to look for a minimal model for every possible domain cardinality. This can make devising correct procedures for expressive languages significantly more difficult.

In particular, in the original paper the proposed algorithm for instance checking in $\mathcal{ALC}$ does not take the definition fully into account. We present here an example that demonstrates the potential counter-intuitive results of the above definition, as well as its disagreement with the proposed algorithm.

Consider the following knowledge base:

$\mathtt{GoodPerson} \sqcap \mathtt{Murderer} \sqsubseteq \mathtt{Abnormal}$
(a good person that is a murderer is an abnormal person)
$\top \sqsubseteq \exists \mathtt{r}_1.(\neg\mathtt{GoodPerson} \sqcap \neg\mathtt{Murderer})$
(there exists someone who is not a good person nor a murderer)
$\top \sqsubseteq \exists \mathtt{r}_2.\mathtt{GoodPerson}$ *(there exists a good person)*
$\top \sqsubseteq \exists \mathtt{r}_3.\mathtt{Murderer}$ *(there exists a murderer)*
$\mathtt{GoodPerson}(\mathtt{Sam})$ *(Sam is a good person)*

Assume the set of closed predicates is $M = \{\mathtt{Abnormal}\}$. Then an expected consequence under the grounded circumscription semantics would be $\neg\mathtt{Murderer}(\mathtt{Sam})$.

Yet, we observe that the following interpretation $\mathcal{I}$ is a GC-model according to the original definition (and hence prevents the expected consequence):

$\Delta^{\mathcal{I}} = \{1, 2\}$
$\texttt{Sam}^{\mathcal{I}} = 1$
$\texttt{GoodPerson}^{\mathcal{I}} = \texttt{Murderer}^{\mathcal{I}} = \texttt{Abnormal}^{\mathcal{I}} = \{1\}$
$\texttt{r}_1{}^{\mathcal{I}} = \{(1, 2), (2, 2)\}$
$\texttt{r}_2{}^{\mathcal{I}} = \texttt{r}_3{}^{\mathcal{I}} = \{(1, 1), (2, 1)\}$

The only reason why $\mathcal{I}$ is a GC-model is because it is minimal among all models of cardinality 2. For models of greater domain sizes, `Sam` would never be included in `Murderer`, since he is in `GoodPerson` and we want to minimize `Abnormal`. Moreover, we note that this model is not produced by the *GC-model-finder* algorithm, given in the original paper, hence contradicting their definition. We believe that this is because the notion of GC-model was not meant to include an interpretation like $\mathcal{I}$.

We propose to overcome the problems with the original definition of the grounded circumscription semantics by modifying the notion of minimality in the following way.

Definition 5. If $\mathcal{I}$ and $\mathcal{J}$ are models of K then the *"smaller than"* relation is defined in the following way: $\mathcal{I} \prec_M^{\text{new}} \mathcal{J}$ if

(i) $Ext^{\mathcal{I}}(\{a\}) = Ext^{\mathcal{J}}(\{a\})$ for every $a \in Ind(K)$,
(ii) $Ext^{\mathcal{I}}(W) \subseteq Ext^{\mathcal{J}}(W)$ for every $W \in M$ and
(iii) there is a $W \in M$ such that $Ext^{\mathcal{I}}(W) \subset Ext^{\mathcal{J}}(W)$. ⊣

Our definition of minimality, which subsumes the one in the original paper (i.e. $\mathcal{I} \prec_M^{\text{orig}} \mathcal{J}$ implies $\mathcal{I} \prec_M^{\text{new}} \mathcal{J}$), is more intuitive in that it directly involves the assignment of individuals to concepts and roles. This is anyway at the heart of the tableau method used by the authors of the original paper when providing algorithms for the reasoning tasks in ground circumscription. Apart from being more intuitive, this is also the more realistic approach. Comparison between two models can simply be done on the basis of the mapping of individuals to concepts and roles. Although we acknowledge that requiring same domains in order to allow comparison between two models has been the default approach to circumscription in DLs so far, it seems to be an unnecessary specialization.

Furthermore, we will present in the following how we can significantly improve the inferencing algorithms in comparison to the original paper. Keeping the old definition would have hindered this development. Hence we believe that our reformulation of the notion of minimality is an improvement compared to the previous work. It is more efficient in producing results by avoiding to interfere as much with the actual semantics, whilst capturing the essense of the idea of ground circumscription in more satisfactory way.

3.2 Satisfiability of a GC-Knowledge Base

We present now a direct and complexitywise cheap way of determining whether a GC-$\mathcal{ALCO}$ knowledge base is satisfiable. To this end, we first enhance the

KB with axioms that ensure grounding of the closed predicates and then we take advantage of the following lemma, which effectively says that if a grounded model exists, then a minimal grounded model must exist as well.

Lemma 1. Both relations $\prec_M^{\text{orig}}$ and $\prec_M^{\text{new}}$ are well-founded on the class of grounded models of a knowledge base K wrt to M. ⊣

Definition 6. Let (K, M) be a *GC-$\mathcal{ALCO}$-KB*, where $M \cap N_C = \{A_1, ..., A_n\}$ and $M \cap N_r = \{r_1, ..., r_m\}$. We define K_M as the $\mathcal{ALCO}$-*KB* which consists of all the axioms that are included in K as well as the following ones:

- $P \equiv \{x | x \in Ind(K)\}$ where P is a fresh concept name,
- $A_i \sqsubseteq P$ for every $i \in \{1, ..., n\}$,
- $\exists r_j.\top \sqsubseteq P$ for every $j \in \{1, ..., m\}$,
- $\top \sqsubseteq \forall r_j.P$ for every $j \in \{1, ..., m\}$.

K_M is then called a *grounded* $\mathcal{ALCO}$ knowledge base. ⊣

We do not give an explicit algorithm for determining satisfiability of a GC knowledge base, because we will show that this decision problem can be reduced to the satisfiability checking of a (standard) $\mathcal{ALCO}$ knowledge base. To solve the reasoning tasks of ground circumscription with the use of the already developed monotonic DL reasoning tools was our aim, and as the next proposition shows, in this case it is proven to be achieved quite ideally.

Proposition 1. Let (K, M) be a *GC-$\mathcal{ALCO}$-KB*. (K, M) is satisfiable (under the grounded circumscription semantics, both w.r.t. $\prec_M^{\text{orig}}$ and $\prec_M^{\text{new}}$) if and only if the $\mathcal{ALCO}$ knowledge base K_M is (classically) satisfiable. ⊣

One observation worth mentioning here is that grounding, although defined at a semantic level, can be internalized in the syntax and expressed as a particular class of knowledge bases. And through this grounding, a localization of the non-monotonicity is achieved, such that for the principal task of deciding satisfiability, we do not even need to expand reasoning beyond the already known algorithms that exist for standard DLs.

4 Instance Checking

For the task of determining whether or not a concept assertion (also referred to as 'fact') is entailed by a GC-$\mathcal{ALCO}$ knowledge base, knowing that a minimal model exists is not enough. We have to be able to find this model or at least to negate the possibility of a grounded model being minimal. What seems to be more efficient is a bottom-up approach, where the grounded models found first are definitely minimal. From now on we are working only on our definition of minimality and we will write $\prec_M$ instead of $\prec_M^{\text{new}}$. Also in the following $Part(X)$ will denote the set of all partitions of a set X and $\mathbb{Z}_2^* = \{0, 1\}^*$ will denote the set of all finite binary words.

Specification of the Configuration Space. The idea of defining independently what is essentially the search space of our algorithm, a space of possible choices of extensions to the closed predicates, is inspired by the original paper, where a similar set is specified. However, and this is one more clue which points to the divergence between the intended meaning of ground circumscription and what was initially defined, in the original paper the domain and the possible interpretations of the individuals over it are not taken into consideration when defining this space. Having improved the definition, we still need to add a dimension to the search space which will correspond to the possible interpretations of the individual names.

Given an interpretation $\mathcal{I}$, the *individual allocation* of $\mathcal{I}$ is the set $\mathsf{AL}(\mathcal{I}) \in Part(Ind(K))$ such that every $X \in \mathsf{AL}(\mathcal{I})$ has the property: for every $a \in X$ and $b \in Ind(K)$ holds that $a^{\mathcal{I}} = b^{\mathcal{I}}$ if and only if $b \in X$.

Suppose that $I \in Part(Ind(K))$ and $a, b \in Ind(K)$. We call a and b *I-invariant* and write $a \simeq_I b$ if there is an $X \in I$ such that $a, b \in X$. A set $Z \subseteq Ind(K)$ is called *I-complete* if $a \in Z$ and $a \simeq_I b$ imply $b \in Z$. Similarly a set $V \in Ind^2(K)$ is called *I-complete* if $(a, b) \in V$ and $a \simeq_I a'$ and $b \simeq_I b'$ imply $(a', b') \in V$. For the sake of conciseness in the next definition, we define the following sets:

$$Cmp_I(K) = \{X \subseteq Ind(K) | X \text{ is } I\text{-complete}\}$$
$$Cmp_I^2(K) = \{Y \subseteq Ind^2(K) | Y \text{ is } I\text{-complete}\}$$

We can now employ the above notions to specify the search space of our algorithm:

Definition 7. Let (K, M) be a *GC-$\mathcal{ALCO}$-KB*, where $M \cap N_C = \{A_1, ..., A_n\}$ and $M \cap N_r = \{r_1, ..., r_m\}$. Then the set

$$\mathcal{G}_{(K,M)} = \Big\{(X_1, ..., X_n, Y_1, ..., Y_m, I) \Big| X_i \subseteq Cmp_I(K), Y_j \subseteq Cmp_I^2(K), I \in Part(Ind(K))\Big\}$$

is called *configuration space* of (K, M). ⊣

$\mathcal{G}_{(K,M)}$ is obviously a finite set. Every grounded model $\mathcal{I}$ of K wrt. to M corresponds to a point in the configuration space. In particular we will call the tuple

$$\Big(Ext^{\mathcal{I}}(A_1), ..., Ext^{\mathcal{I}}(A_n), Ext^{\mathcal{I}}(r_1), ..., Ext^{\mathcal{I}}(r_m), \mathsf{AL}(\mathcal{I})\Big)$$

the *assignment* of $\mathcal{I}$.

Let $G_1, G_2 \in \mathcal{G}_{(K,M)}$ with $G_1 = (Z_1, ..., Z_{n+m}, I)$ and $G_2 = (V_1, ..., V_{n+m}, I)$. We say that G_1 is *smaller than* G_2 and we write $G_1 \prec G_2$ if it holds that $Z_i \subseteq V_i$ for all $i \in \{1, ..., n+m\}$ and there exists $i \in \{1, ..., n+m\}$ such that $Z_i \subset V_i$. The following result then holds trivially:

Lemma 2. Let $\mathcal{I}, \mathcal{J}$ be grounded models of a knowledge base K wrt M and let $G_1, G_2 \in \mathcal{G}_{(K,M)}$ be their respective assignments. Then it holds that $\mathcal{I} \prec_M \mathcal{J}$ if and only if $G_1 \prec G_2$. ⊣

Binary Encoding and Linear Order. Let $\mathcal{G}_{(K,M)}$ be the configuration space of a *GC-$\mathcal{ALCO}$-KB*. Let $Ind(K) = \{a_1, ..., a_\mu\}$. For the purposes of the algorithm presented in the next section, we want to order $\mathcal{G}_{(K,M)}$ linearly. We achieve that by using a binary encoding for every $G \in \mathcal{G}_{(K,M)}$ and the lexicographical order. We first introduce an encoding $s : Part(Ind(K)) \to \mathbb{Z}_2^*$. Every partition of $Ind(K)$ can be specified by indicating which couples of individual names (that appear in the knowledge base) belong to the same block of the partition. This is easily percieved with the following visualization:

	a_1	a_2	a_3	$\dots$	a_μ
a_1	-	$s_{(1,2)}$	$s_{(1,3)}$	$\dots$	$s_{(1,\mu)}$
a_2	-	-	$s_{(2,3)}$	$\dots$	$s_{(2,\mu)}$
$\vdots$	-	-	-	$\ddots$	$\vdots$
$a_{\mu-1}$	-	-	-	-	$s_{(\mu-1,\mu)}$
a_μ	-	-	-	-	-

In accordance with the above table we define

$$s(I) = s_{(1,2)}s_{(1,3)}...s_{(1,\mu)}s_{(2,3)}s_{(2,4)}...s_{(2,\mu)}...s_{(\mu-1,\mu)}$$

where $s_{(i,j)} = 1$ if there exists $Z \in I$ with $a_i, a_j \in Z$, otherwise $s_{(i,j)} = 0$. We can now proceed to define the complete binary encoding of the points of the configuration space.

Let $\sigma : \mathscr{P}\Big(Ind(K)\Big) \cup \mathscr{P}\Big(Ind^2(K)\Big) \cup Part\Big(Ind(K)\Big) \to \mathbb{Z}_2^*$, with

$$\sigma(X) = \begin{cases} z_1 z_2 \dots z_\mu & \text{if } X \subseteq Ind(K) \text{ where } z_\kappa = \begin{cases} 1 & \text{if } a_\kappa \in X \\ 0 & \text{if } a_\kappa \notin X \end{cases} \\ z_1 z_2 \dots z_{\mu^2} & \text{if } X \subseteq Ind^2(K) \text{ where } z_{\kappa\mu+\lambda} = \begin{cases} 1 & \text{if } (a_\kappa, a_\lambda) \in X \\ 0 & \text{if } (a_\kappa, a_\lambda) \notin X \end{cases} \\ s(X) & \text{if } X \in Part(Ind(K)) \end{cases}$$

We can view words over $\mathbb{Z}_2$ as natural numbers encoded in the binary system. If $w_1, w_2 \in \mathbb{Z}_2^*$ are words of the same length, we write $w_1 < w_2$ if this relation holds for the respective natural numbers. We can now define a total order '$<$' on $\mathcal{G}_{(K,M)}$.

Definition 8. Let $G_1 = (Z_1, ..., Z_k)$ and $G_2 = (V_1, ..., V_k)$ be two points in the configuration space of a *GC-$\mathcal{ALCO}$-KB* (K, M). G_1 *precedes* G_2 and we write $G_1 < G_2$ if there exists an $i \leq k$ such that $\sigma(Z_i) < \sigma(V_i)$ and for all $j < i$ holds $\sigma(Z_j) = \sigma(V_j)$. ⊣

For efficiency purposes, it is important here that the order defined above is induced by the partial order of minimality, so that the algorithm will discover the minimal model first and discard searching in large sections of the configuration space. The following lemma ensures that this is indeed the case.

Lemma 3. Let $\mathcal{G}_{(K,M)}$ be the configuration space of a *GC-$\mathcal{ALCO}$-KB*. For every $G_1, G_2 \in \mathcal{G}_{(K,M)}$ holds that $G_1 \prec G_2$ implies $G_1 < G_2$. ⊣

Navigation Within the Configuration Space. It is critical, given a point in the configuration space, to be able to construct a grounded model with such an assignment, if one exists. This is accomplished by adding the axioms specified in the next definition.

Definition 9. Let (K, M) be a *GC-$\mathcal{ALCO}$-KB*, where $M \cap N_C = \{A_1, ..., A_n\}$ and $M \cap N_r = \{r_1, ..., r_m\}$. Let $G = (X_1, ..., X_n, Y_1, ..., Y_m, I)$ be a point in the configuration space of (K, M). We define K_G as the $\mathcal{ALCO}-KB$ which consists of all the axioms that are included in K_M as well as the following ones:

- $\{a\} \equiv \{b\}$ for all $a, b \in Ind(K)$ with $a \simeq_I b$,
- $\{a\} \sqsubseteq \neg\{b\}$ for all $a, b \in Ind(K)$ with $a \not\simeq_I b$,
- $A_i \equiv X_i$ for every $i \in \{1, ..., n\}$,
- $N_{(a,j)} \equiv \{c \in Ind(K) | (c, a) \in Y_j\}$ for every $a \in Ind(K)$ and $j \in \{1, ..., m\}$,
- $\exists r_j.\{a\} \equiv N_{(a,j)}$ for every $a \in Ind(K)$ and $j \in \{1, ..., m\}$.

K_G is then called a *pointwise restriction* of (K, M). ⊣

Lemma 4. Let K_G be a pointwise restriction of a *GC-$\mathcal{ALCO}$* knowledge base (K, M). The following statements hold:

(i) If $\mathcal{I}$ is a model of K_G, then G is the assignment of $\mathcal{I}$.
(ii) If there exists a model of K_M with assignment G, then K_G is satisfiable. ⊣

The Algorithm. We can now specify an algorithm for deciding whether or not a GC knowledge base entails an assertion. We will also give an example and discuss complexity and possibility of further use and development.

Let (K, M) be a *GC-$\mathcal{ALCO}$-KB*, where $M \cap N_C = \{A_1, ..., A_n\}$ and $M \cap N_r = \{r_1, ..., r_m\}$. We want to check if an assertion $B(a)$ is a logical consequence of (K, M). Such a reasoning task is commonly refered to as *instance checking*, hence the title of this section. If $a \notin Ind(K)$ the answer is trivial, so for the rest it assumed that $a \in Ind(K)$. We split the decision procedure in two cases, the first of which will prove to be solvable in a much more simple way, by only once calling the "oracle" $\mathcal{ALCO}$ reasoner. In the following, given a knowledge base K_0, we use the notation $K_0^+ := K_0 \cup \{\neg B(a)\}$ to refer to K_0 augmented with the negation of the assertion we are checking for entailment.

Case 1: $B \in M$

Proposition 2. K_M^+ is unsatisfiable if and only if (K, M) entails $B(a)$. ⊣

Case 2: $B \notin M$

We want to determine if every GC-model of (K, M) entails $B(a)$. To achieve that, we navigate bottom up in the configuration space which is essentially the space of possible individual allocations and ground extensions to the predicates in M. Let $\mathcal{G}_{(K,M)} = \{G_1, ..., G_\lambda\}$, where $G_1 < G_2 < ... < G_\lambda$.

IC Algorithm:

1] Initiate `Stack` := $\mathcal{G}_{(K,M)}$.
2] for $i = 1$ to λ
3] If $G_i \in$ `Stack`:
4] Check K_{G_i} for satisfiability.
5] If YES:
6] Check $K^+_{G_i}$ for satisfiability.
7] If YES return FALSE.
8] Else remove all $G_j \succ G_i$ from `Stack`.
9] return TRUE.

That the above algorithm terminates is obvious, because there is only one loop. Moreover the command in line 9, outside of the loop, guarantees that it will return either TRUE or FALSE.

Proposition 3. The IC algorithm returns TRUE if and only if (K, M) entails $B(a)$. ⊣

To demonstrate how this whole procedure works, we give a simple example.

Example. Let K be a knowledge base consisting of the following axioms:

$$B(a), \neg B(b), r(b, c), \rho(a, b), \rho(a, c), \exists r.\neg A \sqsubseteq A$$

Let $M = \{A\}$. Then $Part(Ind(K)) = \{I_1, I_2, I_3, I_4, I_5\}$ where

$$I_1 = \{\{a\}, \{b\}, \{c\}\}$$
$$I_2 = \{\{a, b\}, \{c\}\}$$
$$I_3 = \{\{a, c\}, \{b\}\}$$
$$I_4 = \{\{a\}, \{b, c\}\}$$
$$I_5 = \{\{a, b, c\}\}.$$

Figure 1 is a visualization of our configuration space. Each possible individual allocation corresponds to a lattice of possible ground extensions for the closed predicates, which in our case consists of just A. The restriction of the search space to I-complete sets of possible extensions, with respect to an individual allocation I, is portrayed by the apparent reduction of points in I_2-I_5.

Suppose that we want to check the assertion $\neg(A \sqcap \forall\rho.A)(a)$ for entailment. This basically means that not all individuals can be interpreted as members of the extension of A. Then the IC algorithm will look bottom-up for grounded models of K wrt M. If a model is found, then an augmented knowledge base will be built, consisting of the current pointwise restriction and the negation of the given assertion, which in our case is just $(A \sqcap \forall.\rho A)(a)$. In case this augmented KB is found to be satisfiable, the algorithm will halt, giving FALSE as an answer. Otherwise it will remove from the `Stack` all points which are above, hence reducing further the remaining exploration. For this particular instance,

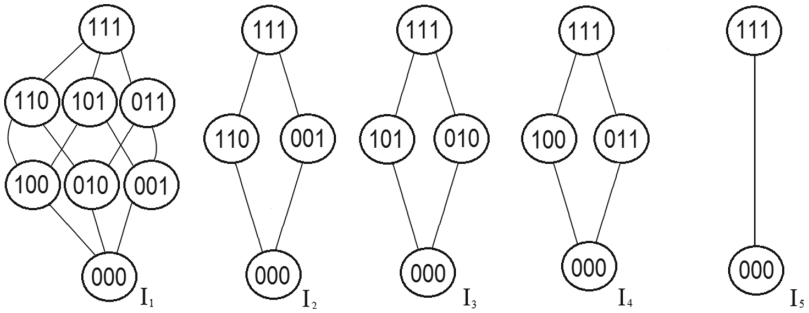

Fig. 1. The configuration space of (K, M).

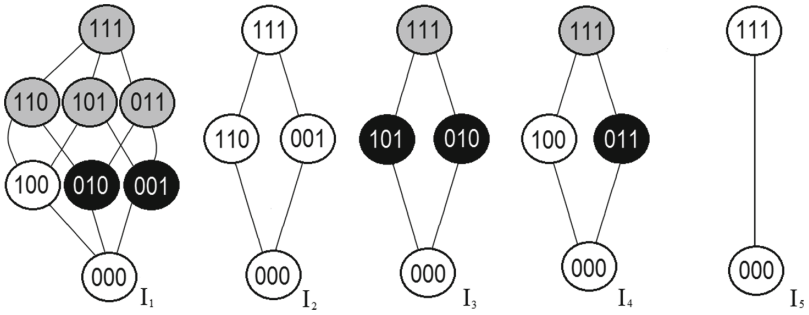

Fig. 2. The distribution of GC-models of (K, M) in the configuration space.

(K, M) entails $\neg(A \sqcap \forall.\rho A)(a)$, so no minimal model which satisfies $(A \sqcap \forall.\rho A)(a)$ can be found, and so the algorithm will return TRUE.

Note that the entailment holds exactly because of the minimality, i.e. there are grounded models where all individuals belong to A. Figure 2 gives an account of the distribution of grounded models and GC-models of (K, M) over the configuration space. Points in white are those that do not correspond to any model of K_M, points in black correspond to GC-models and points in grey to the rest of the grounded models. All the points in grey are exactly those that will be never "visited", i.e. at some step they will be removed from the `Stack`.

Complexity and Optimization Considerations. Considering that the presented algorithm requires at most exponentially many calls of the $\mathcal{ALCO}$ reasoner, each of which requires exponential time, we get that the overall complexity is still in ExpTime. The lower bound is ExpTime as well as in the case $M = \emptyset$ the inference problem turns into standard reasoning in $\mathcal{ALCO}$ which is known to be ExpTime-complete. For more expressive description logics, the complexity of the black-box reasoning part will dominate and hence determine the overall complexity.

Regarding the practical runtime behavior, we expect a significant improvement through the removal of points that results from the command in line 8.

That is because the algorithm, in accordance with the defined linear order, will try smaller points of the configuration space first and once a model is found, the algorithm will stop looking at the rest of the branch.

Of course there is room for optimization of this algorithm. Notably from the example we can see how two out of the five lattices should have been rejected from the start, since they represent individual allocations which are incompatible with the given knowledge base. More thoroughly, one could remove points which correspond to assignments which are not consistent with the axioms in the knowledge base.

On the other hand, the results we have acquired so far can be directly extended to more complex languages. That follows from the fact that in none of the proofs supporting this study did we rely on the limitations of $\mathcal{ALCO}$. In effect, we have used the constructive capabilities of our language, in creating new knowledge bases that represent the notion of grounding and different points in the configuration space. But we have not appealed to any restrictions imposed by the specific syntax of $\mathcal{ALCO}$, with the exception of course of the property of decidabilty, which is implicit wherever a decision procedure is regarded.

5 Minimality Checking: A Non-standard Reasoning Task

In this section we present a solution to the task of determining whether a specific grounded model is minimal by calling the standard DL reasoner just once. It can be of use in devising algorithms for other reasoning problems in grounded circumscription, but also maybe in some optimized variant of the IC algorithm presented previously.

Definition 10. Let K_M be a grounded KB where $M \cap N_C = \{A_1, ..., A_n\}$ and $M \cap N_r = \{r_1, ..., r_m\}$ and let $\mathcal{I}$ be a model of K_M. We call *down-the-chain axioms* with respect to $\mathcal{I}$, the following set of GCIs:

I. $\{a\} \equiv Ext^{\mathcal{I}}(\{a\})$ for every $a \in Ind(K)$,
II. $Ext^{\mathcal{I}}(\neg\{a\}) \sqsubseteq \neg\{a\}$ for every $a \in Ind(K)$,
III. $A_i \sqsubseteq Ext^{\mathcal{I}}(A_i)$ for every $i \in \{1, ..., n\}$,
IV. $B_{(a,j)} \equiv \{c \in Ind(K) | (a^{\mathcal{I}}, c^{\mathcal{I}}) \in r_j^{\mathcal{I}}\}$ for every $a \in Ind(K)$ and $j \in \{1, ..., m\}$,
V. $\{a\} \sqsubseteq \forall r_j.B_{(a,j)}$ for every $a \in Ind(K)$ and $j \in \{1, ..., m\}$,
VI. $$\top \sqsubseteq \exists r.\left(\left(\bigsqcup_{i\in\{1,...,n\}} \left(Ext^{\mathcal{I}}(A_i) \sqcap \neg A_i\right)\right) \sqcup \left(\bigsqcup_{\substack{j\in\{1,...,m\} \\ a\in Ind(K) \\ c\in B_{(a,j)}}} \left(\{a\} \sqcap \forall r_j.\neg\{c\}\right)\right)\right),$$

where r is a fresh role, i.e. it does not appear in K. K_M augmented with the down-the-chain axioms with respect to a model is called a *confining* of K_M and symbolized $K_M^{\mathcal{I}-}$, where $\mathcal{I}$ is the respective model. ⊣

Notice that the number of axioms in each of the categories I-V depends on M whereas VI is one single axiom. The next lemma shows how we can find a smaller grounded model than a given one, if there exists one. Intuitively this is like going down in the lattice of possible grounded models, hence the terminology.

Lemma 5. Let (K, M) be a *GC-$\mathcal{ALCO}$-KB* and let $\mathcal{I}$ be a model of K_M. There exists a model $\mathcal{J}$ of K_M such that $\mathcal{J} \prec_M \mathcal{I}$ if and only if $K_M^{\mathcal{I}-}$ is satisfiable. ⊣

For direct practical use, the above lemma is more conveniently expressed in the following form:

Corollary 1. *(Minimality Check)* Let (K, M) be a *GC-$\mathcal{ALCO}$-KB* and let $\mathcal{I}$ be a model of K_M. If $K_M^{\mathcal{I}-}$ is unsatisfiable, then $\mathcal{I}$ is a GC-model of (K, M). ⊣

6 Conclusions

In our paper we have refined and rectified the foundational definition of grounded circumscription and have produced some first results as a basis for further research. Starting from a definition that is more accurate in incorporating the intuition behind grounded circuscription, we have an improved solution to the satisfiability task which now does not require any non-standard description logic and can be solved by a single call to an off-the-shelf DL reasoner. Moreover, we have provided an algorithm for instance checking, which was only insufficiently covered in the original paper on grounded circumscription.

Apart from the algorithm itself, the theory provided gives a well-founded understanding of the general potential of grounded circumscription, as redefined here. The configuration space can prove to be a useful notion for devising other non-standard reasoning algorithms. The down-the-chain axioms and minimality check as a sub-task could contribute to solving other reasoning tasks within grounded circuscription as well.

As mentioned earlier, an advantage of our approach is that all our results hold if $\mathcal{ALCO}$ is replaced by a more complex language, as long as it is decidable. Certainly there is a lot of space for further development of grounded circumscription. It remains to be seen whether the IC algorithm performs well in practice and/or can be sufficiently optimized further.

One of our main aims was to reduce as much of the reasoning as possible to standard DL reasoning. This is achieved, in our opinion to the largest extent possible. With this feature, our theory is implementation-friendly, and one main future objective is to create a reasoner for grounded circumscription, which will of course be working on top of an efficient standard DL reasoner.

References

1. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F. (eds.): The Description Logic Handbook: Theory Implementation and Applications. Cambridge University Press, New York (2003)
2. Bonatti, P.A., Lutz, C., Wolter, F.: The Complexity of Circumscription in DLs (2014). CoRR abs/1401.3476
3. Delivorias, S., Rudolph., S.: Grounded Circumscription in Description Logics (2015)
4. Hitzler, P., Krötzsch, M., Rudolph, S.: Foundations of Semantic Web Technologies. Chapman & Hall/CRC, Boca Raton (2009)

5. McCarthy, J.: Circumscription – a form of non-monotonic reasoning. Artif. Intell. **13**, 27–39 (1980)
6. Rudolph, S.: Foundations of description logics. In: Polleres, A., d'Amato, C., Arenas, M., Handschuh, S., Kroner, P., Ossowski, S., Patel-Schneider, P. (eds.) Reasoning Web 2011. LNCS, vol. 6848, pp. 76–136. Springer, Heidelberg (2011)
7. Sengupta, K., Krisnadhi, A.A., Hitzler, P.: Local closed world semantics: grounded circumscription for OWL. In: Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N., Blomqvist, E. (eds.) ISWC 2011, Part I. LNCS, vol. 7031, pp. 617–632. Springer, Heidelberg (2011)
8. W3C OWL Working Group. OWL 2 Web Ontology Language: Document Overview (2009). http://www.w3.org/TR/owl2-overview/