TECHNICAL REPORT

INSTITUT FÜR INFORMATIONSSYSTEME

ABTEILUNG DATENBANKEN UND ARTIFICIAL INTELLIGENCE

dRDF: Entailment for Domain-Restricted RDF

DBAI-TR-2008-59

Reinhard Pichler Axel Polleres Fang Wei
Stefan Woltran

Institut für Informationssysteme
Abteilung Datenbanken und
Artificial Intelligence
Technische Universität Wien
Favoritenstr. 9
A-1040 Vienna, Austria
Tel: +43-1-58801-18403
Fax: +43-1-58801-18492
sekret@dbai.tuwien.ac.at
www.dbai.tuwien.ac.at

DBAI TECHNICAL REPORT
2008

TECHNISCHE UNIVERSITÄT WIEN

DBAI TECHNICAL REPORT

DBAI TECHNICAL REPORT DBAI-TR-2008-59, 2008

dRDF: Entailment for Domain-Restricted RDF

Reinhard Pichler[1] **Axel Polleres** [2] **Fang Wei**[3]
Stefan Woltran[4]

Abstract. We introduce domain-restricted RDF (dRDF) which allows to associate an RDF graph with a fixed, finite domain that interpretations for it may range over. We show that dRDF is a real extension of RDF and discuss impacts on the complexity of entailment in dRDF. The entailment problem represents the key reasoning task for RDF and is well known to be NP-complete. Remarkably, we show that the restriction of domains in dRDF raises the complexity of entailment from NP- to Π_2^P-completeness. In order to lower complexity of entailment for both domain-restricted and unrestricted graphs, we take a closer look at the graph structure. For cases where the structure of RDF graphs is restricted via the concept of bounded treewidth, we prove that the entailment is tractable for unrestricted graphs and coNP-complete for domain-restricted graphs.

[1] Institute for Information Systems 184/2, Technische Universität Wien, Favoritenstrasse 9-11, 1040 Vienna, Austria. E-mail: pichler@dbai.tuwien.ac.at

[2] Digital Enterprise Research Institute (DERI), National University of Ireland, Galway. E-mail: axel.polleres@deri.org

[3] Institute for Information Systems 184/2, Technische Universität Wien, Favoritenstrasse 9-11, 1040 Vienna, Austria. E-mail: wei@dbai.tuwien.ac.at

[4] Institute for Information Systems 184/2, Technische Universität Wien, Favoritenstrasse 9-11, 1040 Vienna, Austria. E-mail: woltran@dbai.tuwien.ac.at

1 Introduction

The Resource Description Framework [18] provides means to publish and share metadata on the Web in a machine readable form. One of the features of RDF is to express incomplete metadata by so-called blank nodes, which allow to make statements about unknown resources, such as "I know *somebody* called 'Tim Berners-Lee' (but I don't know the URI identifying him)". In a sense, blank nodes can be viewed as existential variables in the data. In certain circumstances however, it is conceivable that more refined statements could be made about this "*somebody*". Normally, an RDF graph is interpreted over an infinite set of resources. However, one often has a concrete set of resources in mind when writing RDFs. For instance, we want to be able to say: "I don't know the URI identifying Tim, but I know that it is one of the URI's listed at: `http://www.example.org/w3c-people`", i.e. we want to assign blank nodes only to certain URI's from a restricted, finite set, but we just do not know which one.

In this paper, we introduce and investigate so-called *domain-restricted RDF (dRDF)* graphs which allow to define exactly such restrictions. Domain-restricted RDF graphs are graphs for which interpretations are bound to a fixed, finite domain.

Example 1.1 *The RDF graphs in Fig. 1 model collaboration links between various people. In the figure and subsequent examples, we use* $\_{:}b_1, \_{:}b_2, ..., \_{:}b_n$ *to denote blank nodes, quoted strings for literals of* L*, and colon separated pairs of alphanumeric strings where the prefix may be empty for QNames/URIs.*[1] *Graphs are sets of triples, as usual. The two fictitious graphs* G_1 *and* G_2 *describe metadata we assume to be published by two of the authors of this paper working at TU Vienna, Fang and Stefan. Fang's graph only talks about current employees of TU Vienna, Stefan's graph talks about current and past employees of TU Vienna, whereas* G_3 *denotes collaboration links of Stefan Decker, who talks in his graph only about current DERI employees. Even if we assume that lists of URIs to denote these domains*[2] *are published at some Web referenceable address, current RDF does not provide means to allow the respective publishers of the graphs* $G_1 - G_3$ *to express or reference the domain they are talking about. dRDF fills exactly this gap.* □

The key reasoning task for RDF is deciding whether the information in one RDF graph is subsumed by what is said by another RDF graph – the RDF entailment problem. Entailment should intuitively be affected by restricting the domain of a graph. For instance, the graph G_3 is subsumed by G_2 modulo blank node renaming. Nevertheless, since these graphs talk about different domains, a reasoning engine aware of these domain restrictions should not conclude entailment here.

It is well known that blank nodes raise the complexity of the entailment problem to NP-completeness [14]. A major goal of this work is to search for realistic restrictions which might

[1] We use QNames in the sense of RDF notations such as Turtle [2], where e.g. `foaf:name`, `:axel`, or `:worksWith` stand for full URIs, but we leave out the actual namespace prefixes here, as they do not matter for illustration.

[2] Complete lists of URIs denoting all employees of TU Vienna, DERI, etc. should be easy to obtain. Institutes typically already do publish this data, see e.g. `http://www.deri.ie/about/team/` or `http://www.dbai.tuwien.ac.at/staff/`. It would be easy to write e.g. a GRDDL [9] transformation for those pages which creates lists of unique identifiers for their respective team members.

G_1	G_2	G_3
(_:b_1,foaf:name,"Fang"), (_:b_2,foaf:name,"Stefan"), (_:b_3,foaf:name,"Reini"), (_:b_1,:worksWith,_:b_2), (_:b_2,:worksWith,_:b_3)	(_:b_1,foaf:name,"Stefan"), (_:b_2,foaf:name,"Reini"), (_:b_3,foaf:name,"Fang"), (_:b_1,:worksWith,_:b_2), (_:b_3,:worksWith,_:b_1), (_:b_1,:worksWith,_:b_3), (_:b_4,foaf:name,"Axel"), (_:b_1,:worksWith, _:b_4)	(_:b_2,foaf:name,"Stefan"), (_:b_1,foaf:name,"Axel"), (_:b_2,:worksWith,_:b_1)

Figure 1: Fictitious collaboration graphs published by Fang, Stefan W. and Stefan D.

ensure tractability of the entailment problem. We thus study two kinds of restrictions: one is the restriction to a fixed, finite domain (i.e., dRDF) mentioned above. The other one is the restriction of the graph structure of the (RDF or dRDF) graphs. More precisely, we investigate the entailment problem for graphs having *bounded treewidth*, which can be thought of as a generalization of acyclicity. It has been successfully applied to graph-related problems in many areas [3, 8] where otherwise intractable problems have been proved to become tractable if the underlying graph structure has bounded treewidth.

One may expect that both kinds of restrictions decrease the complexity of the entailment problem. Somewhat surprisingly, we will show that the restriction to finite domains does not help at all. In contrast, it even increases the complexity of entailment up to the second level of the polynomial complexity hierarchy, viz. to Π_2^P-completeness. On the other hand, we will show that the restriction to RDF graphs of bounded treewidth indeed makes the entailment problem tractable. We will present a polynomial-time algorithm for this case. Actually, also for dRDF graphs, bounded treewidth decreases the complexity of entailment by one level in the polynomial hierarchy; we thus end up with coNP-completeness rather than Π_2^P-completeness.

Our complexity results are summarized as follows. Note that the case of infinite resources and no restriction on the treewidth is well known to be NP-complete [14].

	finite domain-restricted graphs	unrestricted graphs
bounded treewidth	**coNP-complete**	**in P**
unbounded treewidth	$\mathbf{\Pi_2^P}$**-complete**	NP-complete

The remainder of this paper is organized as follows. In Section 2 we will first review the formal definitions of RDF's syntax and semantics and introduce *domain-restricted RDF (dRDF)* along the way. In this section we will also prove some important theoretical properties concerning general RDF entailment vs. dRDF entailment. The complexity of the entailment problem in case of domain-restricted RDF is dealt with in Section 3. The effect of bounded treewidth without or with domain-restriction is investigated in Section 4 and Section 5, respectively. We wrap up the paper with an outlook to related and future works and draw conclusions in Sections 6 and 7. Appendix A gives a detailled proof of a central lemma.

2 Preliminaries

In this paper, we exclusively deal with *simple* RDF entailment, i.e., without giving any special semantics to the RDF(S) vocabulary. For short, we shall therefore use the term "RDF entailment" throughout this paper in order to refer to "simple RDF entailment". For the definition of the syntax and semantics of RDF graphs, we find the notation given in [14] more convenient than the one used for defining the standard semantics in [10]. It should be noted that for *simple* interpretations which we consider here both approaches are equivalent, apart from the fact that plain literals are ignored in [14]. It can be easily verified that our complexity results also hold if we stick literally to the definitions in [10].

2.1 RDF graphs and domain-restricted RDF graphs

We consider an infinite set U (RDF URI references), an infinite set B (blank nodes, also referred to as variables), and an infinite set L (RDF literals). An *RDF triple* is a triple of the form $(v_1, v_2, v_3) \in (U \cup B) \times U \times (U \cup B \cup L)$. In such a triple, v_1 is called the *subject*, v_2 the *predicate*, and v_3 the *object*. The union of the sets U and L is often denoted by UL, and likewise, $U \cup B \cup L$ is often denoted by UBL.

An *RDF graph* (or simply a *graph*) is a set of RDF triples. A subgraph is a subset of a graph. The *vocabulary* of a graph G, denoted by UL_G, is the set of elements of UL occurring in triples of G. A graph is ground if it has no blank nodes. RDF graphs are often represented as edge-labeled, directed graphs where a triple (a, b, c) is represented by $a \xrightarrow{b} c$.

A *map* is a function $\mu\colon UBL \to UBL$ preserving URIs and literals, i.e., $\mu(v) = v$ for all $v \in UL$. We define $\mu(G) := \{(\mu(s), \mu(p), \mu(o)) \mid (s, p, o) \in G\}$. A graph G' is an instance of G if there exists a map μ with $G' = \mu(G)$. With some slight ambiguity we say that there exists a map $\mu\colon G_1 \to G_2$ if there is a map $\mu\colon UBL \to UBL$, such that $\mu(G_1)$ is a subgraph of G_2. Let G_1 and G_2 be graphs. The *union* $G_1 \cup G_2$ is the set-theoretical union of their sets of triples.

Let $D \subseteq UL$ be a non-empty set of URI references and literals and G be an RDF graph. A *domain-restricted RDF graph (dRDF graph)* is a pair $\langle G, D\rangle$. Graphs such that $|D| = n$ is finite are also called *finitely restricted* (or simply *restricted* for short); graphs with $D = UL$ are also called *unrestricted* graphs. Sightly abusing notation, instead of $\langle G, UL\rangle$ we also write G to denote unrestricted graphs.

2.2 Semantics of (domain-restricted) RDF graphs

A simple *interpretation* $I = (Res, Prop, Lit, \varepsilon, IS, IL)$[3] of an RDF graph G over vocabulary UL_G is defined by (1) a non-empty set of resources Res (also called the domain of I) and of properties $Prop$, (2) a distinguished subset $Lit \subseteq Res$, (3) an extension $\varepsilon(pr) \subseteq Res \times Res$ for every property $pr \in Prop$, and (4) mappings $IS\colon U_G \to Res \cup Prop$ and $IL\colon L \to Lit$.

[3] As mentioned above, we are following the notation from [14]. Clearly, $Res, Prop, Lit, \varepsilon, IS$, and IL correspond to $IR, IP, LV, IEXT, IS$, and IL, respectively, in [10].

We write $I(.)$ to denote the valuation under the interpretation I. We have $I(u) := IS(u)$ for a URI u and, $I(l) := IL(l)$ for a literal l. A triple (s, p, o) has the value "true" in I if $IS(p) \in Prop$ and $(I(s), I(o)) \in \varepsilon(IS(p))$; otherwise (s, p, o) has the value "false". For a ground graph G, we have $I(G) =$ "true" if every triple of G is true in I.

Blank nodes in non-ground graphs are interpreted as existentially quantified variables. Let $A\colon B \to Res$ be a *blank node assignment* (or an *assignment*, for short), and let I be an interpretation. Then we write $[I + A]$ to denote the interpretation I extended by the blank node assignment A. Clearly, $[I + A](b) = A(b)$ for blank nodes $b \in B$, while $[I + A](a) = I(a)$ for $a \in UL$. A non-ground graph G is true in I, if there exists an assignment $A'\colon B \to Res$, s.t. every triple of G is true in $[I + A']$. If a graph G is true in an interpretation I, then we say that I *is a model of* G or I *satisfies* G.

We say that an RDF graph G_1 *entails* the graph G_2, if every interpretation I which satisfies G_1 also satisfies G_2. If this is the case, we write $G_1 \models G_2$. This leads us to the *RDF entailment problem*: Given two RDF graphs G_1, G_2, does $G_1 \models G_2$ hold? This problem is well known to be NP-complete [14]. We may assume w.l.o.g. that $UL_{G_2} \subseteq UL_{G_1}$, since otherwise $G_1 \not\models G_2$ clearly holds (i.e., we can easily construct an interpretation I which satisfies G_1 but not G_2).

Interpretations for a dRDF graph $\langle G, D\rangle$ restrict general RDF interpretations in the following sense. Given an interpretation $I = (Res, Prop, Lit, \varepsilon, IS, IL)$ and a set $D \subseteq UL$ we call the interpretation $I = (Res \cap D, Prop, Lit \cap D, \varepsilon, IS', IL')$ with $IS' = IS_{Res \cap D}$ and $IL' = IL_{Res \cap D}$ the D-restriction of I, also written I_D. Note that we do not restrict the domain of $Prop$ in I_D. Since the purpose of domain-restrictions is mainly to restrict the values which blank nodes may take, we do not need to restrict properties—blank nodes are not allowed in property position in RDF anyway.

We define *d-models* as before with the only difference that for any interpretation I its D-restriction is considered. I.e., given an interpretation I and a dRDF graph $\langle G, D\rangle$, if G is true in I_D, then we say that I *is a d-model of* $\langle G, D\rangle$ or I *d-satisfies* $\langle G, D\rangle$.

Finally, we say that a dRDF graph $\langle G_1, D_1\rangle$ *d-entails* $\langle G_2, D_2\rangle$ (by overloading $\models$ we write $\langle G_1, D_1\rangle \models \langle G_2, D_2\rangle$), if for any interpretation I s.t. I_{D_1} satisfies G_1, I_{D_2} also satisfies G_2. Obviously, if D_1 contains an element not existing in D_2, then this condition can never be fulfilled. Indeed, if $c \in D_1 \setminus D_2$, then we can easily construct a D_1-model of G_1 (where every URI in G_1 is mapped to c) which is not a D_2-model of G_2. Conversely, if D_2 contains elements not existing in D_1, then these elements play no role in a d-entailment test. More precisely, we clearly have that $\langle G_1, D_1\rangle \models \langle G_2, D_2\rangle$ iff $\langle G_1, D_1\rangle \models \langle G_2, D_1 \cap D_2\rangle$. Therefore, in the sequel, we shall restrict ourselves w.l.o.g. to the case $D_1 = D_2$ when we investigate complexity results and other properties of d-entailment.

Example 2.1 (Example 1.1 cont'd) *Getting back to the graphs in Fig. 1, it is easy to see that $G_2 \models G_3$ and that $G_1 \models G'_2$, where G'_2 is the graph obtained from G_2 by removing the last three statements of G_2. As mentioned earlier, Fang's graph G_1 talks only about people working at TU Vienna, i.e., it is restricted to the fixed domain $D_1 = \{$`"Fang"`, `"Stefan"`, `"Reini"`$\} \cup D_{TUV}$ where D_{TUV} is a fixed, finite list of URIs which gives identifiers to all current TU Vienna employees and contains for instance the URIs `:fangwei`, `:stefanwoltran`, and `:reinhardpichler`. This list may be huge and instead of looking up all the real identifiers there, Fang still uses blank* *nodes as in the example for publishing her metadata. But in order to indicate the fact that her graph talks about a finite domain she publishes the dRDF graph $\langle G_1, D_1 \rangle$. Likewise, Stefan publishes his collaboration links as graph $\langle G_2, D_2 \rangle$. Stefan's graph is restricted to $D_2 = D_1 \cup D_{TUVold}$ where D_{TUVold} is a finite list of identifiers of former TU Vienna members that also contains the URI* `:axelpolleres`*, for example. Both $\langle G_1, D_1 \rangle \models \langle G_2', D_2 \rangle$ and $\langle G_2, D_2 \rangle \models G_3$ hold. However, G_3 is in fact none of the authors' but Stefan Decker's collaboration graph at DERI and restricted to the domain $D_3 = \{$* `"Stefan"`*,* `"Axel"`*$\} \cup D_{DERI}$ where D_{DERI} is the (again finite) list of identifiers of DERI employees that contains among others the URIs* `:axelpolleres` *and* `:stefandecker`*, but none of the other previously mentioned URIs. Obviously, $\langle G_2, D_2 \rangle \not\models \langle G_3, D_3 \rangle$ despite the fact $\langle G_2, D_2 \rangle \models G_3$.* □

2.3 Properties of (domain-restricted) entailment

Before we have a closer look at the complexity of this restricted form of the entailment problem, let us discuss some fundamental properties of (domain-restricted) entailment.

Proposition 2.2 *Let G_1, G_2 be graphs and D a finite domain. Then $G_1 \models G_2$ implies $\langle G_1, D \rangle \models \langle G_2, D \rangle$ while the converse is, in general, not true.*

Proof. Clearly, entailment implies d-entailment, since every d-model is also a model. To see that the converse is, in general, not true, consider the following counter-example: Let $G_1 = \{(a, p, b), (a, p, c), (b, p, c)\}$ and $G_2 = \{(x, p, x)\}$ where $a, b, c, p \in U$ and $x \in B$. Moreover, let $D = \{d_1, d_2\}$. Then $\langle G_1, D \rangle \models \langle G_2, D \rangle$ holds: Indeed, with $|D| = 2$, any d-model I of $\langle G_1, D \rangle$ assigns the same value d_i (for some $i \in \{1, 2\}$) to two URIs out of $\{a, b, c\}$. Hence, G_2 is true in $[I + A]$ with $A(x) = d_i$. □

Proposition 2.3 *Let G_1, G_2 be graphs and D a finite domain with $|D| \geq |UL_{G_1 \cup G_2}|$. Then $G_1 \models G_2$ iff $\langle G_1, D \rangle \models \langle G_2, D \rangle$.*

Proof. The "only if" direction immediately follows from Proposition 2.2. The basic idea of the 'if"-direction is that, for any interpretation, only the "active domain" (i.e, the elements in Res which are actually used for interpreting the elements in $UL_{G_1 \cup G_2}$) is relevant. More precisely, suppose that G_1 is true in some interpretation $I = (Res, Prop, Lit, \varepsilon, IS, IL)$ on $UL_{G_1 \cup G_2}$. Then G_1 is also satisfied by the restriction $I' = (Res', Prop, Lit', \varepsilon, IS, IL)$ with $Res' = Res \cap IS(I)$ and $Lit' = Lit \cap IS(I)$. Since the size of the active domain Res' is restricted by $UL_{G_1 \cup G_2}$, there exists a bijective mapping $b : Res' \rightarrow D'$ for some subset $D' \subseteq D$. From this, we can construct a D'-model J of G_1 by composing ε, IS, and IL with b. By assumption, $\langle G_1, D \rangle \models \langle G_2, D \rangle$ holds. Hence, J d-satisfies $\langle G_2, D \rangle$ and also $\langle G_2, D' \rangle$, i.e., there exists an assignment A on the blank nodes in G_2, s.t. G_2 is true in $[J + A]$. But then G_2 is also true in $[I' + A \circ b^{-1}]$ and in $[I + A \circ b^{-1}]$, where b^{-1} denotes the inverse function of b. □

Intuitively, Proposition 2.3 states that entailment and d-entailment coincide for a sufficiently large domain D.

We conclude this section by showing that w.l.o.g. several simplified assumptions may be made, both for the entailment problem and the d-entailment problem.

A *Skolemization* of a graph G is a ground instance of G which maps every blank node in G to some "fresh" URI reference. These fresh URI references are called the Skolem vocabulary. The Skolemization of G is denoted as $sk(G)$. The usefulness of Skolemizations is due to the following property:

Lemma 2.4 *Let G_1, G_2 be graphs and let $sk(G_1)$ be a Skolemization of G_1, s.t. the Skolem vocabulary is disjoint from both G_1 and G_2. Moreover, let D be a finite domain. Then the following equivalences hold: $G_1 \models G_2 \Leftrightarrow sk(G_1) \models G_2$ and $\langle G_1, D\rangle \models \langle G_2, D\rangle \Leftrightarrow \langle sk(G_1), D\rangle \models \langle G_2, D\rangle$.*

Proof. The correctness of this lemma in case of ordinary entailment is shown in [10]. The case of d-entailment can be shown by exactly the same arguments. □

In other words, for both ordinary entailment and d-entailment, we may assume w.l.o.g. that the graph G_1 is ground. After having restricted the syntax, we show that also the set of models to be inspected by an (ordinary or d-) entailment test can be significantly restricted. In [10], entailment testing is reduced to *Herbrand models*. However, in case of domain-restricted graphs, we can of course not be sure that the Herbrand universe is contained in the finite domain D. We thus have to generalize the idea of Herbrand models to *minimal* models.

Definition 2.5 *We call a model I of an RDF graph G (resp. a dRDF graph $\langle G, D\rangle$) a* minimal model *of G (resp. $\langle G, D\rangle$), if the extensions $\varepsilon(pr)$ in I are chosen minimal (for every $pr \in Prop$) s.t. G is true in I. In other words, for every property $pr \in Prop$, a minimal model is characterized by the following relation*

$$\varepsilon(pr) = \{(I(s), I(o)) \mid (s, p, o) \in G_1 \textit{ and } IS(p) = pr\}.$$

Clearly, every Herbrand model is a minimal model while the converse is, in general, not true. The following lemma states that, for (d-) entailment testing, we may restrict ourselves to minimal models of G_1.

Lemma 2.6 *Let G_1, G_2 be graphs, s.t. $UL_{G_2} \subseteq UL_{G1}$ and G_1 is ground. Moreover, let D denote a finite domain. Then the following equivalences hold:*
(a) $G_1 \models G_2$ iff every minimal model I of G_1 satisfies G_2.
(b) $\langle G_1, D\rangle \models \langle G_2, D\rangle$ iff every minimal model I of G_1 with $Res \subseteq D$ satisfies G_2.

Proof. The restriction to minimal models of G_1 (resp. $\langle G_1, D\rangle$) is based on the following observation: Suppose that G_1 (or $\langle G_1, D\rangle$) is true in some interpretation I. Then it remains of course true if we restrict ε to ε' with

$$\varepsilon'(pr) = \varepsilon(pr) \cap \{(I(s), I(o)) \mid (s, p, o) \in G_1 \text{ and } IS(p) = pr\}$$

In case (b), the restriction to interpretations I with $Res \subseteq D$ is obvious since, in a d-interpretation, Res is restricted to a subset of D anyway. □

3 Complexity of d-Entailment

We are now ready to investigate the complexity of d-entailment testing. It turns out that it is one level higher in the polynomial hierarchy than without domain restrictions.

The Π_2^P upper bound is easily established via the lemmas from Section 2.3.

Lemma 3.1 *The d-entailment problem is in* Π_2^P.

Proof. Recall that, by Lemma 2.4, we may assume w.l.o.g. that G_1 is ground. Then the complementary problem "$\langle G_1, D\rangle$ does not d-entail $\langle G_2, D\rangle$" can be decided by the following Σ_2^P-algorithm.

1. Guess an interpretation I over the vocabulary of $G_1 \cup G_2$, s.t. G_1 is true in I.
2. Check that for all assignments A for the blank nodes in G_2, the graph G_2 is false in $[I + A]$. Clearly, this check can be done by a coNP-oracle. □

The proof of the Π_2^P lower bound is much more involved. Due to space limitations, we can only give a rough sketch here. For details, see Appendix A. The proof goes by a reduction from a restricted form of the so-called *H-subsumption problem*. H-subsumption was introduced in the area of automated deduction as a powerful technique of redundancy elimination (cf. [12]). Given two clauses C, C', and a Herbrand universe H, $C \leq_{ss}^H C'$ holds, iff, for each substitution ϑ of the variables in C' to H, there exists a substitution μ of the variables in C to H, such that $C\mu \subseteq C'\vartheta$. In this paper we are only interested in the case that H is a *finite* domain of constants. In [21], it was shown that the H-subsumption problem is Π_2^P-complete even if C and C' consist of unnegated atoms only. However, we need a strongly restricted version of H-subsumption: In particular, we have to restrict the H-subsumption problem to the setting, where no constants are allowed to occur in the clauses and where all predicates are binary. We call such problems total, binary H-subsumption problems (TBH-subsumption, for short). Of course, it is a priori by no means clear that TBH-subsumption is still Π_2^P-hard. Hence, the Π_2^P-hardness proof essentially consists of two parts: the problem reduction from TBH-subsumption to d-entailment and the Π_2^P-hardness proof of TBH-subsumption.

Lemma 3.2 *The TBH-subsumption problem can be reduced in polynomial time to the d-entailment problem.*

Proof. Consider an instance $C \leq_{ss}^H C'$ of the TBH-problem over some finite universe H. In C, C', all predicates are binary and all arguments of the atoms in C and C' are first-order variables. W.l.o.g., the clauses C and C' have no variables in common. Moreover, all predicates in C also occur in C' (since otherwise $C \not\leq_{ss}^H C'$ trivially holds) and all predicates in C' also occur in C (since literals in C with a predicate symbol not occurring in C' play no role at all in the H-subsumption test—they can never be matched by literals in C). Let $Pred = \{p_1, \dots, p_\ell\}$ denote the predicates in C and C'. We define the dRDF graphs $\langle G_1, D\rangle$ and $\langle G_2, D\rangle$ with $D = H$, $G_1 = \{(s, p, o) \mid p(s, o) \in C'\}$, and $G_2 = \{(s, p, o) \mid p(s, o) \in C\}$, s.t. the vocabulary of $G_1 \cup G_2$

is given as $U := \{s, p, o \mid p(s, o) \in C'\}$ and $L = \emptyset$. Moreover, we have $B = \{s, o \mid p(s, o) \in C\}$. In other words, G_1 is ground while G_2 contains only blank nodes. Clearly, this reduction is feasible in polynomial time. For the correctness, we have to show the equivalence $C \leq^H_{ss} C' \Leftrightarrow \langle G_1, D\rangle \models \langle G_2, D\rangle$).

"$\Leftarrow$" Assume $\langle G_1, D\rangle \models \langle G_2, D\rangle$ and let ϑ be an arbitrary ground substitution on the variables in C'. We have to show that there exists a substitution μ with $C\mu \subseteq C'\vartheta$.

From ϑ (defined on the first-order variables in C', which correspond to URIs in G_1) we define the interpretation I as follows: $Res = D$, $Prop = \{pr_1, \ldots, pr_\ell\}$, $IS(a) = a\vartheta$, and $IS(p_j) = pr_j$. Finally, for every j, we set $\varepsilon(pr_j) = \{(IS(a), IS(b)) \mid \exists(a, p_j, b) \in G_1\}$, i.e., I is a minimal model of G_1.

By assumption, there exists an assignment A on the blank nodes in G_2, s.t. G_2 is true in $[I + A]$. We thus define the substitution μ on the variables z in C (which correspond to the blank nodes in G_2) as $z\mu = A(z)$. Now let $p_j(x, y)$ be an arbitrary atom in C. We have to show that $p_j(x, y)\mu$ is in $C'\vartheta$. By construction, (x, p_j, y) is a triple in G_2. Since G_2 is true in $[I + A]$, we have $(A(x), A(y)) \in \varepsilon(pr_j)$. Hence, there exists a triple $(a, p_j, b) \in G_1$, s.t. $A(x) = IS(a)$ and $A(y) = IS(b)$. But then also $p_j(x, y)\mu = p_j(a, b)\vartheta \in C'\vartheta$ holds.

"$\Rightarrow$" Assume $C \leq^H_{ss} C'$ and let I be an arbitrary minimal model of G_1. We have to show that G_2 is true in $[I + A]$ for some blank node assignment A.

Let $(a, p_j, b) \in G_1$. Since G_1 is true in I, we can be sure that $IS(a), IS(b) \in Res \subseteq D$ and $IS(p_j) \in Prop$. By construction, the variables in C' correspond to those URIs in G_2, which occur in subject or object position in G_1. We may thus define ϑ as the substitution $z\vartheta = IS(z)$. By assumption, $C \leq^H_{ss} C'$. Hence, there exists a ground substitution μ, s.t. $C\mu \subseteq C'\vartheta$. From μ we define the assignment A on the blank nodes z in G_2 (which correspond to the variables in C) as $A(z) = z\mu$. It can be easily verified that $[I + A]$ is a model of G_2. Indeed, let (x, p_j, y) be a triple in G_2. Then $(A(x), A(y)) = (x\mu, y\mu)$. By $C\mu \subseteq C'\vartheta$, we have $p_j(x, y)\mu = p_j(a, b)\vartheta$ for some atom $p_j(a, b) \in C'$. Hence, there exists a triple (a, p_j, b) in G_1 with $IS(a) = a\vartheta$ and $IS(b) = b\vartheta$. Thus, since we are assuming that G_1 is true in I, we may conclude that $(IS(a), IS(b))$ is in the extension $\varepsilon(IS(p_j))$, i.e., $(A(x), A(y))$ is true in $[I + A]$. □

Lemma 3.3 *The TBH-subsumption problem is Π^P_2-hard.*

Proof. The proof is highly involved and very technical. It proceeds in three steps: First, Π^P_2-hardness is shown for clauses using only ternary predicates over a 2-element Herbrand universe $H = \{a, b\}$. This result is then extended to an arbitrary, finite H with $|H| \geq 2$. Finally, it is shown that Π^P_2-hardness still holds even if the clauses are built up from binary predicates only and $|H| \geq 4$; details are fully worked out in Appendix A. □

Putting the Lemmas 3.1–3.3 together, we immediately get the following result.

Theorem 3.4 *The d-entailment problem is Π^P_2-complete.*

In other words, the complexity of entailment increases from NP- to Π^P_2-completeness if we restrict the domain. This unexpected effect can be explained as follows. RDF entailment with unrestricted domain admits a syntactical characterization: $G_1 \models G_2$ iff there exists a map from G_2 to G_1. The proof of the “only if” direction of this equivalence crucially depends on an argument via Herbrand interpretations of G_1 (see the interpolation lemma and its proof in [10]). Of course, this argument is no longer valid for dRDF-graphs if the domain D is smaller than the Herbrand universe (note that the counter-example given in the proof of Proposition 2.2 is based on a similar idea).

4 Efficient Entailment through Bounded Treewidth

In this section we first define the treewidth of an RDF graph, then show that the entailment problem of $G_1 \models G_2$ can be solved in polynomial time, if G_2 has bounded treewidth. Recall that an RDF triple has the form $(v_1, v_2, v_3) \in (U \cup B) \times U \times (U \cup B \cup L)$. Let us denote those triples (v_1, v_2, v_3) where v_1 and v_3 are two distinct variables as *blank triples*. Moreover, we speak of *semi-blank triples* if only one of v_1, v_3 is a variable or if v_1 and v_3 are identical variables.

It is interesting to observe that the intractability of the RDF entailment problem $G_1 \models G_2$ depends *only* on the blank triples in G_2. To see this, consider the ground and semi-blank triples in G_2: finding a map of any ground triple is merely an existence test of the triple in G_1. Thus all the ground triples can be tested independently from each other. Now let us assume that G_2 contains only semi-blank triples with k distinct variables. Assume further that $|G_1| = m$ and $|G_2| = n$. To test $G_1 \models G_2$, we first partition all the triples of G_2 into k disjoint sub-graphs $P_1, \ldots, P_k$, s.t. two triples belong to the same sub-graph if and only if they contain the same variable. For each i, let n_i denote the cardinality $|P_i|$ of P_i. Clearly, $n_1 + \cdots + n_k = n$. We can then check the entailment of the sub-graphs one by one. For each $P_{i(1 \leq i \leq k)}$, the variable in P_i can be mapped to m possible values. Because there is only one variable in P_i, for each map μ, we have to execute the existence test $\mu(P_i) \subseteq G_1$, which takes maximum mn_i steps. Thus in summary, the total cost of the entailment test is $\mathcal{O}(m^2 n)$.

However, if the graph G_2 contains blank triples, it is possible that the variables are intertwined s.t. no variable can be tested independently, thus the number of possible maps is exponential in the size of the variables occurring in blank triples. Treewidth is a well-known metric on graphs that measures how tree-like a graph is. Many intractable problems become tractable, if the treewidth of the underlying structure is bounded.

We shall now show that the entailment problem $G_1 \models G_2$ becomes tractable if the graph G_2 has bounded treewidth. Recall the syntactical characterization of entailment [10, 14]: $G_1 \models G_2$ iff there exists a map from G_2 to G_1. Hence, the entailment problem for unrestricted RDF graphs comes down to a special case of conjunctive query containment where all predicates are binary. Hence, the notion of treewidth and the tractability of conjunctive query containment in case of bounded treewidth (see e.g. [5]) naturally carry over to RDF graphs and the entailment problem. However, we prefer to give a *native* definition of tree decomposition for RDF graphs here, so that the RDF intuition is preserved. Likewise, we explicitly present an entailment algorithm in terms of the RDF terminology rather than by just referring to conjunctive queries.

We start by giving the definitions of tree decomposition and treewidth for an RDF graph. By

G_1	G_2
(_:b_1,:worksWith,_:b_2), (_:b_2,:worksWith,_:b_3), (_:b_1,:worksWith,_:b_3), (_:b_2,:worksWith,_:b_5), (_:b_1,:worksWith,_:b_5), (_:b_3,:worksWith,_:b_6), (_:b_3,:worksIn,"TUV"), (_:b_5,:worksIn,"DERI")	(_:b_1,:worksWith,_:b_2), (_:b_2,:worksWith,_:b_3), (_:b_1,:worksWith,_:b_3), (_:b_2,:worksWith,_:b_4), (_:b_1,:worksWith,_:b_4), (_:b_2,:worksWith,_:b_5), (_:b_4,:worksWith,_:b_6), (_:b_3,:worksIn,"TUV"), (_:b_5,:worksIn,_:b_7)

Figure 2: RDF graphs for Example 4.2

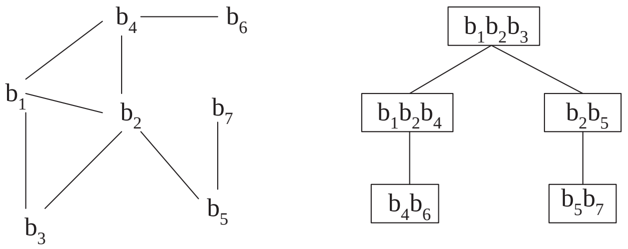

Figure 3: Undirected graph of G_2 from Fig. 2 and the tree decomposition of G_2

the above considerations, we assume that the RDF graph does not contain any ground triple. We denote all the variables occurring in G as B_G.

Definition 4.1 *A* tree decomposition $\mathcal{T}$ *of an RDF graph G is defined as $\langle T, (B_i)_{i \in T} \rangle$ where T is a tree and each B_i is a subset of B_G with the following properties:*

1. *Every $b \in B_G$ is contained in some B_i.*

2. *For every blank triple $(v_1, v_2, v_3) \in G$, there exists an $i \in T$ with $\{v_1, v_3\} \subseteq B_i$.*

3. *For every $b \in B_G$, the set $\{i \mid b \in B_i\}$ induces a subtree of T.*

The third condition is usually referred to as the *connectedness condition*. The sets B_i are called the *blocks* of $\mathcal{T}$. The *width* of the tree decomposition $\langle T, (B_i)_{i \in T} \rangle$ is defined as $\max\{|B_i| \mid i \in T\} - 1$. The *treewidth* of an RDF graph G (denoted as $tw(G)$) is the minimal width of all tree decompositions of G. For a given $w \geq 1$, it can be decided in linear time whether some graph has treewidth $\leq w$. Moreover, in case of a positive answer, a tree decomposition of width w can be computed in linear time [4].

Example 4.2 *Consider the graph G_2 given in Fig. 2. The undirected graph and the tree decomposition are depicted in Fig. 3. The treewidth of G_2 is 2.* □

Below, we describe an algorithm which, given the tree decomposition of G_2, tests $G_1 \models G_2$ *in polynomial time*. The intuition behind the algorithm is as follows: we first construct *partial* maps from the nodes on the tree decomposition into G_1 (denoted as M_i in the algorithm below),

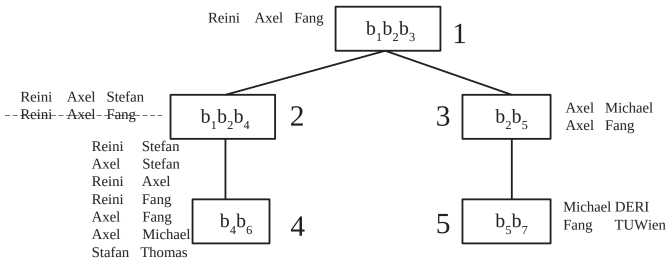

Figure 4: Bottom up processing on the tree decomposition

then successively merge those partial maps which are consistent with each other. If at last the merging succeeds, $G_1 \models G_2$ holds, otherwise not. Note that the connectedness property of the tree decomposition allows us to merge such partial maps in a bottom up manner on the tree (by using the semi-join operation of relational algebra), in polynomial time. We thus carry over ideas proposed in [5] for testing conjunctive query containment to the RDF entailment problem.

Polynomial Time Algorithm. Let $\langle T, (B_i)_{i \in T} \rangle$ be the tree decomposition of the RDF graph G_2 with treewidth k. Given a node i in T, S_i is denoted as the union of all the blocks in the sub-tree rooted at i. The induced sub-graph $G[S_i]$ contains all the triples (v_1, v_2, v_3) in G_2, such that either v_1 or v_3 belongs to S_i. We maintain for each node i in T a relation M_i. In the algorithm below, $\ltimes$ is the natural semi-join operator.

The *Polycheck* algorithm for checking $G_1 \models G_2$ consists of the following steps:

1. For each node i in T, generate the sub-graph G'_i which contains all the triples (v_1, v_2, v_3) such that $\{v_1, v_3\} \subseteq B_i \cup UL_{G_2}$ and $\{v_1, v_3\} \cap B_i \neq \emptyset$.

2. Initialize the relation M_i as follows: for each map μ from G'_i to G_1, the tuple $\mu(B_i)$ is in M_i.

3. Process the tree nodes bottom-up as follows: Suppose i is a tree node in T all of whose children have been processed. For each child j of i, we set $M_i := M_i \ltimes M_j$.

4. Let r be the root of T. Then $G_1 \models G_2$ if and only if M_r is not empty.

Example 4.3 *Let us continue with Example 4.2. With the given tree decomposition of G_2, we illustrate in Fig. 4 how the* Polycheck *algorithm works when testing $G_1 \models G_2$.*

Step 1: We need to generate the sub-graphs $G'_1, \ldots, G'_5$ for the nodes 1–5 of the tree decomposition. For instance, G'_4 is the sub-graph consisting of only one triple $(\_{:}b_4, :workWith, \_{:}b_6)$.

Step 2: Next we generate the partial *maps $M_{i(1 \leq i \leq 5)}$, which are given as the tables beside the tree nodes. Note that following the convention of relational databases, the variable names at each block give the relation schema for that block and every row of a table is called a tuple. For the time being, let us ignore the dotted lines drawn over the tuples. Now consider M_4. For every map μ with $\mu((\_{:}b_4, :worksWith, \_{:}b_6) \in G_1$, we insert the tuple $\mu(\_{:}b_4, \_{:}b_6)$ into M_4. It is easy to verify that there are six distinct maps from G'_4 to G_1, thus M_4 consists of six tuples.*

Step 3: We execute semi-joins along the bottom-up traversal of the tree decomposition. The tables at the leaf nodes remain unchanged. Let us consider the semi-join operation $M_2 \ltimes M_4$. By definition, the result of the semi-join is the set of those tuples t in M_2 for which there is a tuple t' in M_4, s.t. t and t' coincide on their common attributes (in our case b_4). Such a t' is called a partner *of t. Now let us consider the first tuple (b_1, b_2, b_3) in M_2. In this case, b_3 is the value for the common attribute b_4. A partner tuple (b_3, b_6) in M_4 is found, thus the tuple (b_1, b_2, b_3) remains in M_2. However, for the second tuple (b_1, b_2, b_5), there does not exist any partner tuple in M_4. Therefore the tuple (b_1, b_2, b_5) is deleted by the semi-join operation.*

Step 4: Finally, the only tuple in M_1 remains after the semi-join operation with both M_2 and M_3, thus the entailment test succeeds. □

Theorem 4.4 *The algorithm* Polycheck *correctly decides whether $G_1 \models G_2$.*

Proof. We use induction on the number of nodes processed in the tree, with the following hypothesis: After node i is processed, tuple $t \in M_i$ if and only if there is a map μ from $G[S_i]$ to G_1 such that $\mu(B_i) = t$. Thus when the root r has been processed, M_r is non-empty if and only if there is a map μ from $G[S_r]$ to G_1. Because $G[S_r]$ is G_2, we can therefore conclude that $G_1 \models G_2$.

The induction hypothesis holds for the leaves, because of step 2, and the induced sub-graph $G[S_l]$ of any leaf node l is the the graph G'_l we defined in the step 1.

For the induction, assume that we have processed all the children $j_1, \ldots, j_r$ of node i. Suppose that $t \in M_i$ holds *before* the processing of node i. Let ϕ be the map of G'_i to G_1 s.t. $\phi(B_i) = t$. If $t \in M_i$ still holds *after* the processing of i (i.e., the semi-join operations with all the child nodes), then for each $j_{k(1 \leq k \leq r)}$, there is a tuple $t_k \in M_{j_k}$, that agrees with t on the variables in $B_i \cap B_{j_k}$. By the induction hypothesis, there is a map ϕ_k from $G[S_{j_k}]$ to G_1, such that $\phi_k(B_{j_k}) = t_k$.

It remains to show that the maps ϕ, $\phi_1, \ldots, \phi_r$ are consistent. Assume that v occurs in the blocks B_{j_α} and B_{j_β} of two children of i. According to the connectedness condition, v occurs in B_i too. Since ϕ, ϕ_{j_α} and ϕ_{j_β} agree on the common variables, ϕ, ϕ_{j_α} and ϕ_{j_β} are consistent. Let $\mu := \phi \cup \phi_1 \cup \ldots \cup \phi_r$. Then μ is clearly a map from $G[S_i]$ to G_1 such that $\mu(B_i) = t$.

Conversely, assume there is a map μ from $G[S_i]$ to G_1 such that $\mu(B_i) = t$, we show that t is in M_i after the processing of i. Clearly t is in M_i before the processing of i, because G'_i as defined in step 1 is a sub-graph of $G[S_i]$. Let $\phi_1, \ldots, \phi_r$ be the projection μ onto the variables in $S_{j_1}, \ldots, S_{j_r}$, and let ϕ be the projection of μ onto the variables in B_i. By the induction hypothesis, there is a tuple $t_i \in M_{j_k}$, where $1 \leq k \leq r$, such that $\phi_k(B_{j_k}) = t_k$. After step 2, there is a tuple $t \in M_i$, such that $\phi(B_i) = t$. Since t agrees with the tuples $t_1, \ldots, t_r$ on all common attributes, t is in M_i after the processing of i. □

Theorem 4.5 *The entailment problem of $G_1 \models G_2$ can be decided in polynomial time if G_2 has bounded treewidth.*

Proof. Suppose that $|G_1| = n$, $|G_2| = m$ and $tw(G_2) = k - 1$. Step (1): For each tree node i, we need to scan all the triples in G_2 to generate the subgraph G'_i of G_2. Since the size of the tree decomposition of G_2 is not more than m, we have an m^2 upper bound. Step (2): For block B_i with size k, there are n^k possible tuples to be checked. For each tuple t, we generate the map μ from B_i

to t. If $\mu(G_i) \subseteq G_1$, then t is added to M_i. Thus, the cost for the initialization of all the nodes is mn^k. Step (3): Each semi-join operation of two k-ary relations takes n^{2k} (using primitive nested loops), thus the total cost is mn^{2k}. In summary, we get the upper bound $\mathcal{O}(m^2 + mn^{2k})$ on the time complexity of the algorithm *Polycheck*. □

To summarize, the entailment problem $G_1 \models G_2$ is intractable, only if the blank triples in G_2 are cyclic. We note that, in practice, an RDF graph contains rarely blank nodes, and even less blank triples. Hence, most of the real RDF graphs are acyclic or have low treewidth such as 2, and the entailment can be tested efficiently with the above algorithm. For instance, all the graphs in Fig. 1 are acyclic and thus have $tw \leq 1$.

5 Bounded Treewidth and d-Entailment

In the previous section, we have seen for RDF graphs that bounded treewidth significantly decreases the complexity of entailment. We shall now prove a similar result for d-entailment, where bounded treewidth again has a positive impact on the complexity.

Lemma 5.1 *The d-entailment problem of $\langle G_1, D\rangle \models \langle G_2, D\rangle$ is in coNP if G_2 has bounded treewidth.*

Proof. Suppose that $tw(G_2)$ is bounded by some constant. Recall that, by Lemma 2.4, we may assume w.l.o.g. that G_1 is ground. Then the complementary problem of testing $G_1 \not\models G_2$ can be decided by the following NP-algorithm:

1. Guess an interpretation I over the vocabulary of $G_1 \cup G_2$, s.t. G_1 is true in I.

2. Check that there exists no assignments A for the blank nodes in G_2, s.t. the graph G_2 is true in $[I + A]$.

The check in step 2 comes down to an ordinary entailment test $G_1' \not\models G_2'$ with $G_1' := \{(I(s), I(p), I(o)) \mid (s, p, o) \in G_1\}$ and $G_2' := \{(I(s), I(p), I(o)) \mid (s, p, o) \in G_2\}$, where we stipulate $I(z) = z$ for the variables z in G_2. We clearly, have $tw(G_2) = tw(G_2')$. Hence, by Theorem 4.5, the check $G_1' \not\models G_2'$ is feasible in polynomial time. □

Lemma 5.2 *The d-entailment problem of $\langle G_1, D\rangle \models \langle G_2, D\rangle$ is coNP-hard for bounded treewidth of G_2. It remains coNP-hard even if $tw(G_2) = 0$ (i.e., the graph induced by the blank nodes consists of isolated nodes only).*

Proof. We prove the coNP-hardness by reducing the well-known NP-complete problem of graph ℓ-colorability with $\ell \geq 3$ to the complementary problem $\langle G_1, D\rangle \not\models \langle G_2, D\rangle$.

Let $G = (V, E)$ be a graph with vertices V and edges E. We define two RDF graphs G_1 and G_2 as $G_1 := \{(u, e, v) \mid (u, v)$ is an edge in $E\}$ and $G_2 := \{(x, e, x)\}$ for some blank node x. Clearly, $tw(G_2) = 0$. Moreover, this reduction is feasible in polynomial time. It remains to show the correctness of this reduction, which can be seen as follows: By definition, G is ℓ-colorable iff there exists a mapping ϑ, assigning different colors to any two adjacent vertices $u, v \in V$. Obviously, such an assignment exists iff there exists an interpretation I sending all triples (u, e, v) in G_1 to values $(I(u), I(e), I(v))$ with $I(u) \neq I(v)$. This, in turn, is the case iff there exists no blank node assignment A, s.t. $(A(x), A(x)) \in \varepsilon(I(e))$. □

In summary, we thus have the following exact complexity classification.

Theorem 5.3 *The d-entailment problem of $\langle G_1, D\rangle \models \langle G_2, D\rangle$ is coNP-complete if G_2 has bounded treewidth.*

6 Related and Future Work

Our results touch upon many related issues on RDF reasoning and Semantic Web reasoning in general. First of all, we point out that the peculiarities of reasoning with open and closed domains raised by dRDF are closely related to similar issues discussed in the context of reasoning with rules and ontologies [6]. In this paper, we have only discussed *simple* (d)RDF entailment. As for future works, it will be interesting to see, which implications restrictions on the domain have, when higher entailment regimes such as RDF entailment, RDFS entailment, or entailments in OWL variants are considered. We remark here that on the one hand RDF(S) and some non-standard fragments of OWL entailment can be reduced to sets of Datalog rules [7, 13, 17, 27]. Note however, that subsumption of arbitrary Datalog programs is undecidable [26]. Standard fragments of OWL on the other hand are well-known to be syntactic variants of decidable Description Logics [1], i.e. OWL Light is reducible to SHIF(D) and OWL DL is reducible to SHOIN(D) [16]. We plan to investigate how (finite) domain-restrictions on the data affect the complexity of entailment in these languages, see also [1, Chapter 5].

Issues get even more involved, when (non-monotonic) rule languages are added on top of these languages (see [11, 15, 25] and references therein) since in the unrestricted case, the satisfiability for rule-extended ontologies problem becomes undecidable. However, here domain-restrictions may turn out to be actually a good thing, since those cases become decidable for finite domains, as well-known, although a complete investigation of complexity classes such combinations would fall in is still missing (and thus on our agenda). In this context, let us mention that restricting the domain of interpretations is also closely related to restricting the scope of negation in such non-monotonic rule languages for the Web to closed sets of rules, see [23] for further details.

Alternatively to finitly restricting the domain of interpretations for the whole graph it seems that restricting the blank nodes in an RDF graph to a finite, enumerated class (using OWL's `oneOf` constructor) could have similar effects, when we extend our considerations towards OWL. We are currently investigating respective generalizations of the definition of dRDF graphs.

As for related results on finding tractable fragments of RDF, Muñoz et al. [19] define a syntactic subclass of RDFS with $\mathcal{O}(n \log n)$ bounds for entailment (without blank nodes though), which our results complement.

Deciding whether a SPARQL [24] query has an answer is an extension of simple RDF entailment which is PSPACE complete in general but also NP-complete in many cases [20]. We expect that our results propagate to tractable fragments of SPARQL over unrestricted RDF as well as over dRDF graphs, which to define is on our agenda.

Bounded treewidth is a well-established method for identifying tractable subclasses of otherwise intractable problems. It has been successfully applied to a great variety of graph-related problems like network reliability, job scheduling, compiler optimization, model checking, etc. (see e.g., [3, 8]). To the best of our knowledge though, bounded treewidth has not yet been considered in the context of Semantic Web reasoning.

7 Conclusions

Entailment checking is the key reasoning task for RDF. In this work, we have investigated how the complexity of deciding entailment in RDF is affected by two restrictions. Firstly, we introduced dRDF, a variant of RDF which allows to associate an RDF graph with a fixed, finite domain that interpretations for it may range over. We have demonstrated that such restrictions are useful in environments where someone wants to make RDF statements over closed contexts such as enterprises or institutions. Secondly, we investigated restrictions of the graph structure of (d)RDF graphs. Particularly, we investigated the effect of restricting the structure of RDF graphs to bounded treewidth, which considerably lowered the complexity of entailment checking. As related works show, there are many promising directions for applying our results, such as finding further tractable algorithms for fragments of SPARQL, or applying respective restrictions beyond simple RDF entailment.

References

[1] F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider (eds.). *The Description Logic Handbook.* Cambridge, 2003.

[2] D. Beckett. Turtle - Terse RDF Triple Language, Nov. 2007. Available at `http://www.dajobe.org/2004/01/turtle/`.

[3] H. L. Bodlaender. A tourist guide through treewidth. *Acta Cybern.*, 11(1-2):1–22, 1993.

[4] H. L. Bodlaender. A linear-time algorithm for finding tree-decompositions of small treewidth. *SIAM J. Comput.*, 25(6):1305–1317, 1996.

[5] C. Chekuri and A. Rajaraman. Conjunctive query containment revisited. *Theor. Comput. Sci.*, 239(2):211–229, 2000.

[6] J. de Bruijn, T. Eiter, A. Polleres, and H. Tompits. On representational issues about combinations of classical theories with nonmonotonic rules. In *Proc. KSEM'06*, Springer.

[7] J. de Bruijn and S. Heymans. RDF and logic: Reasoning and extension. In *Proc. WebS'07*, IEEE Computer Society.

[8] R. G. Downey and M. R. Fellows. *Parameterized Complexity*. Springer, 1999.

[9] D. Conolly (ed.). Gleaning Resource Descriptions from Dialects of Languages (GRDDL). W3C recommendation, Sep. 2007.

[10] P. Hayes (ed.). RDF Semantics. W3C Recommendation, Feb. 2004.

[11] T. Eiter, G. Ianni, A. Polleres, R. Schindlauer, and H. Tompits. Reasoning with rules and ontologies. In *Reasoning Web 2006,* Springer.

[12] C. G. Fermüller and A. Leitsch. Hyperresolution and automated model building. *J. Log. Comput.*, 6(2):173–203, 1996.

[13] B. Grosof, I. Horrocks, R. Volz, and S. Decker. Description logic programs: Combining logic programs with description logic. In *Proc. WWW'03,* ACM.

[14] C. Gutiérrez, C. A. Hurtado, and A. O. Mendelzon. Foundations of semantic web databases. In *Proc. PODS'04*, ACM.

[15] S. Heymans, J. de Bruijn, L. Predoiu, C. Feier, and D. Van Nieuwenborgh. Guarded hybrid knowledge bases. *Theory and Practice of Logic Programming*, 2008. To appear.

[16] I. Horrocks and P. F. Patel-Schneider. Reducing OWL entailment to description logic satisfiability. In *Proc. ISWC'03*, Springer.

[17] G. Ianni, A. Martello, C. Panetta, and G. Terracina. Faithful and effective querying of RDF ontologies using DLV$^{\mathrm{DB}}$. In *Proc. ASP'07*.

[18] G. Klyne and J. J. Carroll (eds.). Resource Description Framework (RDF): Concepts and Abstract Syntax. W3C Recommendation, Feb. 2004.

[19] S. Muñoz, J. Pérez, C. Gutiérrez. Minimal deductive systems for RDF. In *Proc. ESWC'07*, Springer.

[20] J. Pérez, M. Arenas, and C. Gutierrez. Semantics and complexity of SPARQL. In *Proc. ISWC'06,* Springer.

[21] R. Pichler. On the complexity of H-subsumption. In *Proc. CSL'98*, Springer.

[22] R. Pichler, A. Polleres, F. Wei, and S. Woltran. Entailment for domain-restricted RDF (ext. version). Tech. Report DBAI-TR-2008-59. Available at `http://www.dbai.tuwien.ac.at/research/report/dbai-tr-2008-59.pdf`.

[23] A. Polleres, C. Feier, and A. Harth. Rules with contextually scoped negation. In *Proc. ESWC'06*, Springer.

[24] E. Prud'hommeaux and A. Seaborne (eds.). SPARQL Query Language for RDF. W3C Proposed Recommendation, Nov. 2007.

[25] R. Rosati. $\mathcal{DL}+log$: Tight integration of description logics and disjunctive datalog. In *Proc. KR'06*, AAAI Press.

[26] O. Shmueli. Decidability and expressiveness aspects of logic queries. In *Proc. PODS'87*, ACM.

[27] H. J. ter Horst. Completeness, decidability and complexity of entailment for RDF Schema and a semantic extension involving the OWL vocabulary. *JWS*, 3(2–3):79–115, 2005.

A Appendix

We first show that the problem of H-subsumption remains Π_2^P-complete for the restriction to total problems, even if we restrict ourselves to predicates of arity ≤ 3. Then, we show an according hardness result for binary predicates. The number of predicate symbols remains unbounded. For the sake of presentation, we first show the result for a two-element universe $H = \{0, 1\}$, and then generalize our construction to an arbitrary finite set H.

Theorem A.1 *Total H-subsumption is Π_2^P-hard for any universe H with $|H| = 2$ even for predicates of arity ≤ 3.*

Proof. Let $H = \{0, 1\}$. We reduce the Π_2^P-complete decision problem of $\forall\exists$-QSAT to testing whether $C \leq_{ss}^H D$ holds. To this end, let $\Phi = \forall X_1 \dots \forall X_k \exists X_{k+1} \dots \exists X_m \phi$, where $\phi = \bigwedge_{i=1}^n l_{i,1} \vee l_{i,2} \vee l_{i,3}$. The sets C and D are defined as follows:

$$\begin{aligned}
D &= \{v_i(X_i, T), v_i(X_i, F), v_i(T, F) \mid 1 \leq i \leq k\} \cup \\
&\quad \{n(T, F), n(F, T)\} \cup \\
&\quad \{c(T, T, T), c(T, T, F), c(T, F, T), c(T, F, F), c(F, T, T), c(F, T, F), c(F, F, T)\}; \\
C &= \{v_i(X_i, X_i) \mid 1 \ \leq i \leq k\} \cup \\
&\quad \{n(X_j, \bar{X}_j) \mid 1 \leq j \leq m\} \cup \\
&\quad \{c(l_{i,1}^*, l_{i,2}^*, l_{i,3}^*) \mid 1 \leq i \leq n\};
\end{aligned}$$

where $l^* = X$ if $l = X$, and $l^* = \bar{X}$ if $l = \neg X$ with $\bar{X}_1, \dots, \bar{X}_m$ being new variables. Also T and F are additional variables, where intuitively T should be mapped to a constant representing "true", and F should be mapped to a constant which represents "false". Note that in total H-subsumption problems we cannot force a constant, say "1", to be the representative for "true". Next, variables of form $\bar{X}$ play the role of the negation of X. Finally, the intuitive meaning of the predicates is as follows: $c(.,.,.)$-atoms are used to indicate that a clause c_i is "true", the $n(.,.)$-atoms are used to establish a consistent assignment, i.e., to assign different truth values to variables and their negation, and the atoms $v_i(.,.)$ are, roughly speaking, used to force an assignment to the variables $X_1, \dots, X_n$ in C according to their current assignment in D.

We show that Φ is true $\Leftrightarrow C \leq_{ss}^H D$. Note that the latter problem is total and in polynomial time constructible from Φ.

"$\Rightarrow$" Suppose that Φ is true and let ϑ be an arbitrary ground substitution on the variables $\{X_1, \dots, X_k, T, F\}$ in D. We have to show that there exists a substitution μ on the variables in C, s.t. $C\mu \subseteq D\vartheta$. To this end, we distinguish two cases:

Case 1. $T\vartheta = F\vartheta$. W.l.o.g., let $T\vartheta = F\vartheta = 0$. Then,

$$D\vartheta \ = \ \{v_i(X_i\vartheta, 0), v_i(0, 0) \mid 1 \leq i \leq k\} \cup \{n(0, 0), c(0, 0, 0)\},$$

and it is easy to see that μ which assigns 0 to all variables X_i and $\bar{X}_i$ in C yields $C\mu \subseteq D\vartheta$.

Case 2. $T\vartheta \neq F\vartheta$. W.l.o.g., let $F\vartheta = 0$ and $T\vartheta = 1$. Then,

$$\begin{aligned} D\vartheta &= \{v_i(X_i\vartheta, 1), v_i(X_i\vartheta, 0), v_i(1, 0) \mid 1 \leq i \leq k\} \cup \\ &\quad \{n(1, 0), n(0, 1)\} \cup \\ &\quad \{c(1, 1, 1), c(1, 1, 0), c(1, 0, 1), c(1, 0, 0), c(0, 1, 1), c(0, 1, 0), c(0, 0, 1)\}. \end{aligned}$$

With respect to the QBF, we define a truth assignment I on $\{X_1, \ldots, X_k\}$ with $I(X_i)$ = false if $X_i\vartheta = F\vartheta$ and $I(X_i)$ = true otherwise, i.e. if $X_i\vartheta = T\vartheta$. By assumption, Φ is true. Hence, there exists an extension J of I to $\{X_{k+1}, \ldots, X_m\}$, such that ϕ is true in J. From J, we define the ground substitution μ as follows, for any $1 \leq i \leq m$:

$$X_i\mu = \begin{cases} T\vartheta & \text{if } X_i \text{ is true in } J \\ F\vartheta & \text{if } X_i \text{ is false in } J \end{cases} \qquad \bar{X}_i\mu = \begin{cases} F\vartheta & \text{if } X_i \text{ is true in } J \\ T\vartheta & \text{if } X_i \text{ is false in } J \end{cases}$$

It remains to show that $C\mu \subseteq D\vartheta$. For every $i \leq k$, we have $X_i\mu = X_i\vartheta$ by construction. Hence, every atom $v_i(X_i, X_i)\mu$ in $C\mu$ is contained in $D\vartheta$, since $X_i\vartheta \in \{0, 1\}$, and thus $v_i(X_i\vartheta, X_i\vartheta)$ is in fact in $D\vartheta$. Moreover, by construction, for every $j \in \{1, \ldots, m\}$, the inequality $X_j\mu \neq \bar{X}_j\mu$ holds. Hence, every atom $n(X_j, \bar{X}_j)\mu$ is either of the form $n(1, 0)$ or $n(0, 1)$, and thus contained in $D\vartheta$. Finally, ϕ is true in J, i.e. in all clauses of ϕ, at least one literal is true in J. Hence, by construction, for each i, at least one of the first order variables $l^*_{i,1}, l^*_{i,2}, l^*_{i,3}$ is instantiated to the constant $T\vartheta = 1$. Thus, all atoms $c(l^*_{i,1}, l^*_{i,2}, l^*_{i,3})\mu$ are different from $c(F\vartheta, F\vartheta, F\vartheta) = c(0, 0, 0)$, and therefore, contained in $D\vartheta$.

"$\Leftarrow$" Suppose that $C \leq^H_{ss} D$ holds. To show that then Φ is true, we consider an arbitrary assignment I on $\{X_1, \ldots, X_k\}$ and show that there exists an extension J of I to $\{X_{k+1}, \ldots, X_m\}$, such that ϕ is true in J. Consider now a ϑ with $T\vartheta = 1$, $F\vartheta = 0$, and for $1 \leq i \leq k$, $X_i\vartheta = 1$ iff X_i is true in I, otherwise $X_i\vartheta = 0$. By assumption, $C \leq^H_{ss} D$. Hence, there exists a substitution μ on the variables in C, such that $C\mu \subseteq D\vartheta$. Then, for $1 \leq i \leq k$, the atom $v_i(X_i, X_i)\mu$ in $C\mu$ must be contained in $D\vartheta$. Hence, $X_i\mu = X_i\vartheta$ has to hold. Moreover, for $1 \leq i \leq m$, every atom $n(X_j, \bar{X}_j)\mu$ is either of the form $n(1, 0)$ or $n(0, 1)$. In particular, $X_i\mu \neq \bar{X}_i\mu$ holds for $1 \leq i \leq m$.

From μ, we define the truth assignment J on $\{X_1, \ldots, X_m\}$ with $J(X_i)$ = true iff $X_i\mu = 1$. Clearly, by $X_i\mu \neq \bar{X}_i\mu$, we thus also have that $J(\neg X_i)$ = true iff $\bar{X}_i\mu = 1$. By $C\mu \subseteq D\vartheta$, we have for each i, that at least one of the first order variables $l^*_{i,1}, l^*_{i,2}, l^*_{i,3}$ is instantiated to the constant $T\vartheta = 1$. Hence, for every clause, at least one of the literals $l_{i,1}, l_{i,2}, l_{i,3}$ is true in J. □

Theorem A.2 *Total H-subsumption is Π^P_2-hard for any finite universe H with $|H| \geq 2$ even for predicates of arity ≤ 3.*

Proof. Let $H = \{0, \ldots, \ell\}$ with $\ell \geq 1$. We extend our construction from the previous theorem by using variables $T_0, T_1, \ldots, T_\ell$ instead of F, T. Intuitively, T_0 plays the role of F, and the other T_i's play the role of T. Let Φ and C be as above, and

$$\begin{aligned} D &= \{v_i(X_i, T_\alpha) \mid 1 \leq i \leq k, 0 \leq \alpha \leq \ell\} \cup \\ &\quad \{v_i(T_\alpha, T_\beta) \mid 1 \leq i \leq k, 0 \leq \alpha < \beta \leq \ell\} \cup \\ &\quad \{n(T_\alpha, T_0), n(T_0, T_\alpha)\} \mid 1 \leq \alpha \leq \ell\} \cup \\ &\quad \{c(T_\alpha, T_\beta, T_\gamma) \mid 0 \leq \alpha, \beta, \gamma \leq \ell, (\alpha, \beta, \gamma) \neq (0, 0, 0)\}. \end{aligned}$$

and let C as above. We show that Φ is true $\Leftrightarrow C \leq_{ss}^{H} D$. Note that the latter problem is total and in polynomial time constructible from Φ.

"$\Rightarrow$" Suppose that Φ is true and let ϑ be an arbitrary ground substitution on the variables $\{X_1, \dots, X_k, T_0, \dots, T_\ell\}$ in D. We have to show that there exists a substitution μ on the variables in C, such that $C\mu \subseteq D\vartheta$. To this end, we distinguish three cases:

Case 1. There exists $\alpha \in \{1, \dots, \ell\}$ with $T_0\vartheta = T_\alpha\vartheta$. W.l.o.g., let $T_\alpha\vartheta = T_0\vartheta = 0$. Then, $D\vartheta$ contains the atoms $\{v_i(0,0) \mid 1 \leq i \leq k\} \cup \{n(0,0), c(0,0,0)\}$. Thus, it is easy to see that μ which assigns 0 to all variables X_i and $\bar{X}_i$ in C yields $C\mu \subseteq D\vartheta$.

Case 2. For all $\gamma \in \{1, \dots, \ell\}$, $T_0\vartheta \neq T_\gamma\vartheta$ but there exist $\alpha \neq \beta$ in $\{1, \dots, \ell\}$ with $T_\alpha\vartheta = T_\beta\vartheta$. W.l.o.g., let $T_0\vartheta = 0$ and $T_\alpha\vartheta = T_\beta\vartheta = \ell$. Then, $D\vartheta$ contains (among others) the following atoms:

$v_i(T_\alpha, T_\beta)\vartheta = v_i(\ell, \ell)$, for $1 \leq i \leq k$;
$n(T_\alpha, T_0)\vartheta = n(\ell, 0)$ and $n(T_0, T_\alpha)\vartheta = n(0, \ell)$;
$c(T_\alpha, T_\alpha, T_\alpha)\vartheta = c(\ell, \ell, \ell)$;
$c(T_\alpha, T_\alpha, T_0)\vartheta = c(\ell, \ell, 0)$;
$\vdots$
$c(T_0, T_0, T_\alpha)\vartheta = c(0, 0, \ell)$.

With respect to the QBF Φ, consider the assignment I on $\{X_1, \dots, X_k\}$ with $I(X_i)$ = true for all $i \in \{1, \dots, k\}$. By assumption, Φ is true. Hence, there exists an extension J of I to $\{X_k + 1, \dots, X_m\}$, such that ϕ is true in J. From J, we define the ground substitution μ as follows, for any $1 \leq i \leq m$:

$$X_i\mu = \begin{cases} \ell & \text{if } X_i \text{ is true in } J \\ 0 & \text{if } X_i \text{ is false in } J \end{cases} \qquad \bar{X}_i\mu = \begin{cases} 0 & \text{if } X_i \text{ is true in } J \\ \ell & \text{if } X_i \text{ is false in } J \end{cases}$$

It remains to show that $C\mu \subseteq D\vartheta$. For every $i \leq k$, we have $v_i(X_i, X_i)\mu = v_i(\ell, \ell)$, which is contained in $D\vartheta$ by the above considerations. Moreover, by construction, for every $j \in \{1, \dots, m\}$, μ instantiates exactly one of the variables X_j and $\bar{X}_j$ to 0 and one to ℓ. Hence, every atom $n(X_j, \bar{X}_j)\mu$ is either of the form $n(0, \ell)$ or $n(\ell, 0)$, and thus contained in $D\vartheta$. Finally, μ instantiates the variables $l_{i,1}^*, l_{i,2}^*, l_{i,3}^*$ either to 0 or to ℓ. Moreover, since ϕ is true in J, in all clauses of ϕ, at least one literal is true in J. Hence, by construction, at least one of the first order variables $l_{i,1}^*, l_{i,2}^*, l_{i,3}^*$ is instantiated to the constant ℓ. Thus, all atoms $c(l_{i,1}^*, l_{i,2}^*, l_{i,3}^*)\mu$ are different from $c(0,0,0)$ and are, therefore, contained in $D\vartheta$.

Case 3. For all $\alpha \neq \beta$ in $\{0, \dots, \ell\}$, $T_\alpha\vartheta \neq T_\beta\vartheta$ holds. W.l.o.g., let $T_\alpha\vartheta = \alpha$ for all $\alpha \in \{0, \dots, \ell\}$. Note that then $D\vartheta$ contains all atoms of the form $c(\alpha, \beta, \gamma)$ with $0 \leq \alpha, \beta, \gamma \leq \ell$ and $(\alpha, \beta, \gamma) \neq (0,0,0)$.

Consider the truth assignment I on $\{X_1, \dots, X_k\}$ with $I(X_i)$ = false if $X_i\vartheta = 0$ and $I(X_i)$ = true otherwise, i.e. if $X_i\vartheta = \alpha = T_\alpha\vartheta$ for some $\alpha \geq 1$. By assumption, Φ is true. Hence, there exists an extension J of I to $\{X_{k+1}, \dots, X_m\}$, such that ϕ is true in J. From J, we define the ground substitution μ as follows, for any $1 \leq i \leq m$:

$$X_i\mu = \begin{cases} \ell & \text{if } X_i \text{ is true in } J \text{ and } i > k \\ X_i\vartheta & \text{if } X_i \text{ is true in } J \text{ and } i \leq k \\ 0 & \text{if } X_i \text{ is false in } J \end{cases} \qquad \bar{X}_i\mu = \begin{cases} 0 & \text{if } X_i \text{ is true in } J \\ \ell & \text{if } X_i \text{ is false in } J \end{cases}$$

It remains to show that $C\mu \subseteq D\vartheta$. For every $i \leq k$, we have $X_i\mu = X_i\vartheta$ by construction. Hence, every atom $v_i(X_i, X_i)\mu$ is equal to $v_i(X_i, X_i)\vartheta$ and therefore contained in $D\vartheta$. Moreover, by construction, for every $i \in \{1, \ldots, m\}$ exactly one of the variables X_i and $\bar{X}_i$ is instantiated to 0 by μ. Hence, every atom $n(X_i, \bar{X}_i)\mu$ is either of the form $n(\alpha, 0)$ or $n(0, \alpha)$ for some $\alpha \geq 1$. Thus, every atom $n(X_j, \bar{X}_j)\mu$ is contained in $D\vartheta$. Finally, ϕ is true in J, i.e. in all clauses of ϕ, at least one literal is true in J. Hence, by construction, for each $i \in \{1, \ldots, n\}$, at least one of the first order variables $l^*_{i,1}, l^*_{i,2}, l^*_{i,3}$ is instantiated to a constant different from 0 by μ. Thus, all atoms $c(l^*_{i,1}, l^*_{i,2}, l^*_{i,3})\mu$ are different from $c(0, 0, 0)$ and are, therefore, contained in $D\vartheta$.
"$\Leftarrow$" Suppose that $C \leq^H_{ss} D$ holds, and, for the QBF Φ, consider an arbitrary assignment I on $\{X_1, \ldots, X_k\}$. We have to show that there exists an extension J of I to $\{X_{k+1}, \ldots, X_m\}$, such that ϕ is true in J. Hence, let ϑ be defined as $T_\alpha\vartheta = \alpha$ for all $\alpha \in \{0, \ldots, \ell\}$. Moreover, for $1 \leq i \leq k$, we set $X_i\vartheta = \ell$ iff X_i is true in I, otherwise $X_i\vartheta = 0$. By assumption, $C \leq^H_{ss} D$. Hence, there exists a substitution μ on the variables in C, such that $C\mu \subseteq D\vartheta$. Then, for $1 \leq i \leq k$, the atom $v_i(X_i, X_i)\mu$ in $C\mu$ must be contained in $D\vartheta$. Hence, $X_i\mu = X_i\vartheta$ has to hold. Moreover, for $1 \leq i \leq m$, every atom $n(X_j, \bar{X}_j)\mu$ is either of the form $n(\alpha, 0)$ or $n(0, \alpha)$ with $\alpha \geq 1$. In particular, exactly one of the variables X_i and $\bar{X}_i$ with $1 \leq i \leq m$ is instantiated to 0 by μ.

From μ, we define the truth assignment J on $\{X_1, \ldots, X_m\}$ with $J(X_i) =$ true iff $X_i\mu \neq 0$. Clearly, we thus also have that $J(\neg X_i) =$ true iff $\bar{X}_i\mu \neq 0$. By $C\mu \subseteq D\vartheta$, we have for each $i \in \{1, \ldots, n\}$, that at least one of the first order variables $l^*_{i,1}, l^*_{i,2}, l^*_{i,3}$ is instantiated to a constant different from 0. Hence, for every clause, at least one of the literals $l_{i,1}, l_{i,2}, l_{i,3}$ is true in J. □

Theorem A.3 *Total H-subsumption over binary predicates is Π^P_2-hard for any finite universe H with $|H| \geq 4$.*

Proof. We reduce a variant of $\forall\exists$-QSAT to testing whether $C \leq^H_{ss} D$ holds. In fact, deciding the truth of QBFs $\Phi = \forall X_1 \ldots \forall X_k \exists X_{k+1} \ldots \exists X_m \phi$, where $\phi = \bigwedge_{i=1}^n = l_{i,1} \vee l_{i,2} \vee l_{i,3}$ and each clause (i.e., each conjunct) in ϕ contains at most one universal variable, remains Π^P_2-complete. This can be seen as follows. Let l be a universal literal in a QBF Φ (of above form but without the additional restriction on universal literals per clause). Then, the following rewriting does not change the semantics of Φ: replace l by a label (i.e., a new variable) L_l; add $\exists L_l$ to the block of existential quantifiers; and add clauses $l \vee \neg L_l \vee \neg L_l$, $\neg l \vee L_l \vee L_l$ to ϕ. Note these two clauses represent $l \equiv L_l$ which explains why this transformation does not change the truth of Φ. Moreover, if we treat each universal literal that way, we obtain in polynomial time an equivalent QBF of the desired form, i.e., with at most one universal quantified literal per clause.

We now give a reduction from such QBFs into problems $C \leq^H_{ss} D$. Let $H = \{0, 1, ..., \ell\}$ with $\ell \geq 3$. The general intuition is as in the previous proofs but to deal with the restriction to binary predicates we melt any pair of *existential* variables X_i, X_j into a single variable $[X_i, X_j]$. In fact, we use four different variables $[X_i, X_j]$, $[\bar{X}_i, X_j]$, $[X_i, \bar{X}_j]$, $[\bar{X}_i, \bar{X}_j]$ to represent the 4 different pairs of literals over X_i and X_j. This will allow us to represent 2 literals in 1 argument of a binary predicate in an H-subsumption problem. As before, we consider variables for truth values $T_0, T_1, \ldots, T_\ell$. Again T_0 is intuitively used to refer to "false" and the others to "true". However, $T_0 \ldots T_3$ are used in a bit more subtle way to deal with the new type of variables introduced. To illustrate the basic idea, consider a substitution ϑ which maps $T_0, \ldots, T_3$ to different elements

$H' \subseteq H$, and a substitution μ which maps each $[X_i, X_j]$ to an element from H'. Then we interpret the substitution of variables $[X_i, X_j]$ as follows:

$$\begin{array}{lll}
[X_i, X_j]\mu = T_3\vartheta & \ldots & \text{both } X_i \text{ and } X_j \text{ are true;} \\
[X_i, X_j]\mu = T_2\vartheta & \ldots & X_i \text{ is true, and } X_j \text{ is false;} \\
[X_i, X_j]\mu = T_1\vartheta & \ldots & X_i \text{ is false, and } X_j \text{ is true;} \\
[X_i, X_j]\mu = T_0\vartheta & \ldots & \text{both } X_i \text{ and } X_j \text{ are false.}
\end{array}$$

Now, to establish consistent assignments, we need to set the different "literals" $[X_i, X_j]$, $[X_i, \bar{X}_j]$, $[\bar{X}_i, X_j]$, $[\bar{X}_i, \bar{X}_j]$ into the correct relation. For instance, if $[X_i, X_j]\mu = T_3\vartheta$ then, we have to guarantee that $[X_i, \bar{X}_j]\mu = T_2\vartheta$, $[\bar{X}_i, X_j]\mu = T_1\vartheta$, and $[\bar{X}_i, \bar{X}_j]\mu = T_0\vartheta$. Moreover, variables $X_{i,j}$ and $X_{j,l}$ have to be linked accordingly. For instance, if $[X_i, X_j]\mu = T_3\vartheta$, then $[X_j, X_l]\mu \in \{T_2\vartheta, T_3\vartheta\}$. This is also used to ensure that variables $[X_i, X_i]\mu$ are either mapped to to $T_0\vartheta$ or to $T_3\vartheta$, as expected. Finally, we want to relate each variable X_i to variables $[X_i, X_j]$. For instance, if $X_i\mu = T_0\vartheta$, then we need $[X_i, X_j]\mu \in \{T_0\vartheta, T_1\vartheta\}$, and likewise, if $X_i\mu \neq T_0\vartheta$, then we need $[X_i, X_j]\mu \in \{T_2\vartheta, T_3\vartheta\}$. For those purposes, we will use below the following binary predicates: r (right complement), l (left complement), b (both complement), s (successor), and e (extraction).

We construct D as follows (observe that the first three lines are exactly as in previous constructions).

$$\begin{array}{rcl}
D & = & \{v_i(X_i, T_\alpha) \mid 1 \leq i \leq k, 0 \leq \alpha \leq \ell\} \cup \\
& & \{v_i(T_\alpha, T_\beta) \mid 1 \leq i \leq k, 0 \leq \alpha < \beta \leq \ell\} \cup \\
& & \{n(T_\alpha, T_0), n(T_0, T_\alpha)\} \mid 1 \leq \alpha \leq \ell\} \cup \\
& & \{c(T_\alpha, T_\beta) \mid 0 \leq \alpha \leq \ell; 0 \leq \beta \leq 3; (\alpha, \beta) \neq (0, 0)\} \cup \\
& & E; \quad \text{where} \\
E & = & \{r(T_3, T_2), r(T_2, T_3), r(T_1, T_0), r(T_0, T_1)\} \cup \\
& & \{l(T_3, T_1), l(T_2, T_0), l(T_1, T_3), l(T_0, T_2)\} \cup \\
& & \{b(T_3, T_0), b(T_2, T_1), b(T_1, T_2), b(T_0, T_3)\} \cup \\
& & \{s(T_0, T_0), s(T_0, T_1), s(T_2, T_0), s(T_2, T_1), \\
& & \ s(T_1, T_2), s(T_1, T_3), s(T_3, T_2), s(T_3, T_3)\} \cup \\
& & \{e(T_0, T_0), e(T_0, T_1)\} \cup \{e(T_\alpha, T_2), e(T_\alpha, T_3) \mid 1 \leq \alpha \leq \ell\}.
\end{array}$$

For the construction of C, we use l^* as before, i.e., $l^* = X$ if $l = X$, and $l^* = \bar{X}$ if $l = \neg X$. We use this notation also to obtain the required "melted variable" as follows: For instance, if the i-th clause of ϕ is of form $\neg x_1 \vee \neg x_2 \vee x_3$, we get $l^*_{i,1} = \bar{X}_1$ and $[l^*_{i,2}, l^*_{i,3}]$ yields the variable $[\bar{X}_2, X_3]$. Let us also w.l.o.g. assume that in each clause of ϕ, the variable in first place is either universal or existential and that the variable in the second place (which is thus always existential) has a lower or equal index than the variable in the third place (which is also always existential). We thus need new variables $[X_i, X_j]$, $[X_i, \bar{X}_j]$, $[\bar{X}_i, X_j]$, $[\bar{X}_i, \bar{X}_j]$ with $k < i \leq j \leq m$. We construct C as follows:

$$\begin{array}{rcl} C & = & \{v_i(X_i, X_i) \mid 1 \ \leq i \leq k\} \cup \\ & & \{n(X_j, \bar{X}_j) \mid 1 \leq j \leq m\} \cup \\ & & \{c(l^*_{i,1}, [l^*_{i,2}, l^*_{i,3}])\} \mid 1 \leq i \leq n\} \cup \\ & & F \quad \text{where} \\ F & = & \{r([X_i, X_j], [X_i, \bar{X}_j]) \mid k < i < j \leq m\} \cup \\ & & \{l([X_i, X_j], [\bar{X}_i, X_j]) \mid k < i < j \leq m\} \cup \\ & & \{b([X_i, X_j], [\bar{X}_i, \bar{X}_j]) \mid k < i \leq j \leq m\} \cup \\ & & \{s([X_i, X_j], [X_j, X_l]) \mid k < i \leq j \leq l \leq m\} \cup \\ & & \{e(X_i, [X_i, X_j]) \mid k < i \leq j \leq m\}. \end{array}$$

Recall that each variable $[X_i, X_j]$ refers to a pair of existential variables; moreover, observe that variables $[X_i, X_i]$ are not used in the predicates for left and right complement.

We show that Φ is true $\Leftrightarrow$ $C \leq^H_{ss} D$. Note that the latter problem is total and polynomial constructible from Φ. As well it contains only binary predicates.

"$\Rightarrow$" Suppose that Φ is true and let ϑ be an arbitrary ground substitution on the variables $\{X_1, \ldots, X_k, T_0, \ldots, T_\ell\}$ in D. We have to show that there exists a substitution μ on the variables in C, such that $C\mu \subseteq D\vartheta$. As before, we distinguish three cases:

Case 1. There exists $\alpha \in \{1, \ldots, \ell\}$ with $T_0\vartheta = T_\alpha\vartheta$. W.l.o.g., let $T_\alpha\vartheta = T_0\vartheta = 0$. We then have

$$\begin{array}{rcl} D\vartheta & \supseteq & \{v_i(0, 0) \mid 1 \leq i \leq k\} \cup \\ & & \{n(0, 0), c(0, 0), r(0, T_1\vartheta), l(0, T_2\vartheta), b(0, T_3\vartheta), s(0, 0), e(0, 0)\}. \end{array}$$

Consider μ as follows: $X_i\mu = 0$ and $\bar{X}_i\mu = 0$, for $1 \leq i \leq m$; as well as

$$[X_i, X_j]\mu = 0, \quad [X_i, \bar{X}_j]\mu = T_1\vartheta, \quad [\bar{X}_i, X_j]\mu = T_2\vartheta, \quad [\bar{X}_i, \bar{X}_j]\mu = T_3\vartheta,$$

for $k < i \leq j \leq m$. It can be checked, that $C\mu \subseteq D\vartheta$ holds.

Case 2. For all $\gamma \in \{1, \ldots, \ell\}$, $T_0\vartheta \neq T_\gamma\vartheta$ but there exist $\alpha \neq \beta$ in $\{1, \ldots, \ell\}$ with $T_\alpha\vartheta = T_\beta\vartheta$. W.l.o.g., let $T_0\vartheta = 0$ and $T_\alpha\vartheta = T_\beta\vartheta = \ell$. Then,

$$\begin{array}{rcl} D\vartheta & \supseteq & \{v_i(\ell, \ell) \mid 1 \leq i \leq k\} \cup \\ & & \{n(\ell, 0), n(0, \ell)\} \cup \\ & & \{c(0, T_1\vartheta), c(0, T_2\vartheta), c(0, T_3\vartheta), c(\ell, T_0\vartheta), c(\ell, T_1\vartheta), c(\ell, T_2\vartheta), c(\ell, T_3\vartheta)\} \end{array}$$

With respect to the QBF Φ, consider now the assignment I on $\{X_1, \ldots, X_k\}$ with $I(X_i)$ = true for all $i \in \{1, \ldots, k\}$. By assumption, Φ is true. Hence, there exists an extension J of I to $\{X_{k+1}, \ldots, X_m\}$, such that ϕ is true in J. From J, we define the ground substitution μ as follows: for any $1 \leq i \leq m$,

$$X_i\mu = \begin{cases} \ell & \text{if } X_i \text{ is true in } J \\ 0 & \text{if } X_i \text{ is false in } J \end{cases} \qquad \bar{X}_i\mu = \begin{cases} 0 & \text{if } X_i \text{ is true in } J \\ \ell & \text{if } X_i \text{ is false in } J \end{cases}$$

and for any $k < i \leq j \leq m$, $[X_i, X_j]$, $[X_i, \bar{X}_j]$, $[\bar{X}_i, X_j]$, $[\bar{X}_i, \bar{X}_j]$ are assigned by μ according to the following table:

$[X_i, X_j]\mu$	$[X_i, \bar{X}_j]\mu$	$[\bar{X}_i, X_j]\mu$	$[\bar{X}_i, \bar{X}_j]\mu$	in J
$T_3\vartheta$	$T_2\vartheta$	$T_1\vartheta$	$T_0\vartheta$	X_i and X_j are true
$T_2\vartheta$	$T_3\vartheta$	$T_0\vartheta$	$T_1\vartheta$	X_i is true and X_j is false
$T_1\vartheta$	$T_0\vartheta$	$T_3\vartheta$	$T_2\vartheta$	X_i is false and X_j is true
$T_0\vartheta$	$T_1\vartheta$	$T_2\vartheta$	$T_3\vartheta$	X_i and X_j are false

We show $C\mu \subseteq D\vartheta$. First of all we show, $F\mu \subseteq E\vartheta$. Consider, for instance predicate r. Here, $E\vartheta$ contains the following predicates

$$r(T_3\vartheta, T_2\vartheta), r(T_2\vartheta, T_3\vartheta), r(T_1\vartheta, T_0\vartheta), r(T_0\vartheta, T_1\vartheta);$$

while $F\mu$ contains

$$r([X_i, X_j]\mu, [X_i, \bar{X}_j]\mu),$$

for each $k < i \leq j \leq m$. By inspecting the first two columns of the table it is easy see that any $r([X_i, X_j]\mu, [X_i, \bar{X}_j]\mu)$ matches one of the four predicates in $E\vartheta$. By similar investigations (i.e., by inspecting the first and third, (resp., the first and fourth) column of the table, one can show that this also holds for the predicates $l(\cdot, \cdot)$ (resp., for $b(\cdot, \cdot)$)). For the $s(\cdot, \cdot)$ predicates we have the following observation. Consider any atom $s([X_i, X_j]\mu, [X_j, X_l])\mu$ from $F\mu$. By definition we have that in case X_j is true in J, $[X_i, X_j]\mu \in \{T_3\vartheta, T_1\vartheta\}$ and $[X_j, X_l]\mu \in \{T_3\vartheta, T_2\vartheta\}$. Since all those combinations, i.e.,

$$s(T_1\vartheta, T_2\vartheta), s(T_1\vartheta, T_3\vartheta), s(T_3\vartheta, T_2\vartheta), s(T_3\vartheta, T_3\vartheta)$$

are contained in $E\vartheta$, that case is well captured and $s([X_i, X_j], [X_j, X_l])\mu \in E\vartheta$ holds. For the other case, i.e., X_j is false in J, we have by definition, $[X_i, X_j]\mu \in \{T_2\vartheta, T_0\vartheta\}$ and $[X_j, X_l]\mu \in \{T_1\vartheta, T_0\vartheta\}$. Now the other elements

$$s(T_0\vartheta, T_0\vartheta), s(T_0\vartheta, T_1\vartheta), s(T_2\vartheta, T_0\vartheta), s(T_2\vartheta, T_1\vartheta)$$

in $E\vartheta$ are doing the job and we have $s([X_i, X_j], [X_j, X_l])\mu \in E\vartheta$ also in the second case. Finally, consider predicates $e(X_i\mu, [X_i, X_j]\mu)$ in $F\mu$. If X_i is true under J, we have $X_i\mu = \ell$ and $[X_i, X_j]\mu \in \{T_3\vartheta, T_2\vartheta\}$ by construction. We have $\{e(T_\alpha\vartheta, T_2\vartheta), e(T_\alpha\vartheta, T_3\vartheta) \mid 1 \leq \alpha \leq \ell\} \subseteq E\vartheta$ and thus, in particular, $\{e(\ell, T_2\vartheta), e(\ell, T_3\vartheta)\} \subseteq E\vartheta$, since we assumed that at least one $T_\alpha\vartheta$ reduces to ℓ. Thus for the case X_i is true under J, $e(X_i\mu, [X_i, X_j]\mu)$ is contained in ϑ. For the remaining case, i.e., X_i is false under J, we have $X_i\mu = T_0\vartheta = 0$ and $[X_i, X_j]\mu \in \{T_1\vartheta, T_0\vartheta\}$ by construction. But, also $\{e(T_0\vartheta, T_0\vartheta), e(T_0\vartheta, T_1\vartheta)\} \subseteq E\vartheta$. Thus also for this case, $e(X_i\mu, [X_i, X_j]\mu)$ is contained in $E\vartheta$. This shows $F\mu \subseteq E\vartheta$.

We proceed with the remaining predicates in C and D. For every $i \leq k$, we have $v_i(X_i, X_i)\mu = v_i(\ell, \ell)$, which is contained in $D\vartheta$ by the above considerations. Moreover, by construction, for every $j \in \{1, \ldots, m\}$, μ instantiates exactly one of the variables X_j and $\bar{X}_j$ to 0 and one to ℓ. Hence, every atom $n(X_j, \bar{X}_j)\mu$ is either of the form $n(0, \ell)$ or $n(\ell, 0)$, and thus contained in $D\vartheta$. Since ϕ is true in J, in all clauses of ϕ, at least one literal is true in J. Hence, by construction, at least one of the first order variables $l^*_{\iota,1}, l^*_{\iota,2}, l^*_{\iota,3}$ is instantiated to the constant ℓ, for each $1 \leq \iota \leq n$. If this is the case for $l^*_{\iota,1}$, we are done since $c(l^*_{\iota,1}, [l^*_{\iota,2}, l^*_{\iota,3}])\mu$ then is given by $c(\ell, T_\beta\vartheta)$ with $\beta \in \{0, \ldots, 3\}$ and $c(\ell, T_\beta\vartheta) \in D\vartheta$, for each such β. Otherwise, we have 4 cases:

- $l^*_{\iota,2}\mu = \ell$, where $l^*_{\iota,2} = X_i$; then X_i was true in J and by inspecting the table $[l^*_{\iota,2}, l^*_{\iota,3}]\mu \in \{T_3\vartheta, T_2\vartheta\}$, and thus $c(l^*_{\iota,1}, [l^*_{\iota,2}, l^*_{\iota,3}])\mu$ is either $c(0, T_3\vartheta)$ or $c(0, T_2\vartheta)$, which are both contained in $D\vartheta$.

- $l^*_{\iota,2}\mu = \ell$, where $l^*_{\iota,2} = \bar{X}_i$; then X_i was false in J and by inspecting the table, we again have $[l^*_{\iota,2}, l^*_{\iota,3}]\mu \in \{T_3\vartheta, T_2\vartheta\}$, showing $c(l^*_{\iota,1}, [l^*_{\iota,2}, l^*_{\iota,3}])\mu \in D\vartheta$.

- $l^*_{\iota,3}\mu = \ell$, where $l^*_{\iota,3} = X_j$; then X_j was true in J and by inspecting the table $[l^*_{\iota,2}, l^*_{\iota,3}]\mu \in \{T_3\vartheta, T_1\vartheta\}$, and thus $c(l^*_{\iota,1}, [l^*_{\iota,2}, l^*_{\iota,3}])\mu$ is either $c(0, T_3\vartheta)$ or $c(0, T_1\vartheta)$, which are both contained in $D\vartheta$.

- $l^*_{\iota,3}\mu = \ell$, where $l^*_{\iota,3} = \bar{X}_j$; as before, this shows that X_j is false in J and inspecting the table gives evidence that $c(l^*_{\iota,1}, [l^*_{\iota,2}, l^*_{\iota,3}])\mu \in D\vartheta$.

Case 3. For all $\alpha \neq \beta$ in $\{0, \dots, \ell\}$, $T_\alpha\vartheta \neq T_\beta\vartheta$ holds. W.l.o.g., let $T_\alpha\vartheta = \alpha$ for all $\alpha \in \{0, \dots, \ell\}$. For the QBF Φ, consider the truth assignment I on $\{X_1, \dots, X_k\}$ with $I(X_i)$ = false if $X_i\vartheta = 0$ and $I(X_i)$ = true otherwise, i.e. if $X_i\vartheta = \alpha = T_\alpha\vartheta$ for some $\alpha \geq 1$. By assumption, Φ is true. Hence, there exists an extension J of I to $\{X_{k+1}, \dots, X_m\}$, such that ϕ is true in J. From J, we define the ground substitution μ as follows, for any $1 \leq i \leq m$:

$$X_i\mu = \begin{cases} 0 & \text{if } X_i \text{ is false in } J \\ X_i\vartheta & \text{if } X_i \text{ is true in } J \text{ and } i \leq k \\ \ell & \text{if } X_i \text{ is true in } J \text{ and } i > k \end{cases} \qquad \bar{X}_i\mu = \begin{cases} 0 & \text{if } X_i \text{ is true in } J \\ \ell & \text{if } X_i \text{ is false in } J \end{cases}$$

and for any $k < i \leq j \leq m$, $[X_i, X_j]$, $[X_i, \bar{X}_j]$, $[\bar{X}_i, X_j]$, $[\bar{X}_i, \bar{X}_j]$ are assigned by μ as before (see table). We show $C\mu \subseteq D\vartheta$. In fact, showing $F\mu \subseteq E\vartheta$ is as before, except that for the predicates $e(\cdot, \cdot)$ we now make use of the assumption that $T_\alpha\vartheta = \alpha$ and thus each $e(X_i, [X_i, X_j])\mu$ has a matching element $e(\cdot, \cdot)$ in $F\vartheta$ as well.

Now for the remaining parts of the clauses, we proceed as follows. For every $i \leq k$, we have $X_i\mu = X_i\vartheta$ by construction. Hence, every atom $v_i(X_i, X_i)\mu$ is equal to that $v_i(X_i, T_\alpha)\vartheta$ where $X_i\vartheta = T_\alpha\vartheta$. Note that such a correspondence has to exist since $\{T_0\vartheta, \dots, T_\ell\vartheta\} = H$. Therefore each $v_i(X_i, X_i)\mu$ is contained in $D\vartheta$. Moreover, as before, we have by construction that, for every $i \in \{1, \dots, m\}$ exactly one of the variables X_i and $\bar{X}_i$ is instantiated to 0 by μ. Hence, every atom $n(X_i, \bar{X}_i)\mu$ is either of the form $n(\alpha, 0)$ or $n(0, \alpha)$ for some $\alpha \geq 1$. Thus, every atom $n(X_j, \bar{X}_j)\mu$ is contained in $D\vartheta$. Finally, ϕ is true in J, i.e. in all clauses of ϕ, at least one literal is true in J. As in the previous case, one can show that all atoms $c(l^*_{i,1}, [l^*_{i,2}, l^*_{i,3}])\mu$ are thus contained in $D\vartheta$, as well.

"$\Leftarrow$" Suppose that $C \leq^H_{ss} D$ holds, and consider an arbitrary assignment I on $\{X_1, \dots, X_k\}$. We have to show that there exists an extension J of I to $\{X_{k+1}, \dots, X_m\}$, such that ϕ is true in J. Hence, let ϑ be defined as $T_\alpha\vartheta = \alpha$ for all $\alpha \in \{0, \dots, \ell\}$. By assumption, $C \leq^H_{ss} D$. Hence, there exists a substitution μ on the variables in C, such that $C\mu \subseteq D\vartheta$. Then, for $1 \leq i \leq k$, the atom $v_i(X_i, X_i)\mu$ in $C\mu$ must be contained in $D\vartheta$. Hence, $X_i\mu = X_i\vartheta$ has to hold. Moreover, for $1 \leq i \leq m$, every atom $n(X_j, \bar{X}_j)\mu$ is either of the form $n(\alpha, 0)$ or $n(0, \alpha)$ with $\alpha \geq 1$. In particular, exactly one of the variables X_i and $\bar{X}_i$ with $1 \leq i \leq m$ is instantiated to 0 by

μ. Now for any variable of form $[X_i, X_i]$ we have either $[X_i, X_i]\mu = T_3\vartheta$ or $[X_i, X_i]\mu = T_0\vartheta$, since $s([X_i, X_i], [X_i, X_i])\mu$ is in $D\vartheta$. Moreover, since $e(X_i, [X_i, X_i])\mu \in D\vartheta$, we obtain that $[X_i, X_i]\mu = T_3\vartheta$ iff $X_i\mu \neq T_0\alpha$ and $[X_i, X_i]\mu = X_i\mu = T_0\vartheta$, otherwise. Hence, we have the required correspondence between variables $[X_i, X_i]$ and X_i, for all i. Thus, the variables $[X_i, X_j]$ have the desired truth-value by the definition of the $s(\cdot, \cdot)$ predicates, and so have the "literals", $[X_i, X_j]$, $[X_i, \bar{X}_j]$, $[\bar{X}_i, X_j]$, $[\bar{X}_i, \bar{X}_j]$, for any $k < i \leq j \leq m$,

From μ, we define the truth assignment J on $\{X_1, \ldots, X_m\}$ with $J(X_i) =$ true iff $X_i\mu \neq 0$. Clearly, we thus also have that $J(\neg X_i) =$ true iff $\bar{X}_i\mu \neq 0$. By $C\mu \subseteq D\vartheta$, we have for each $i \in \{1, \ldots, n\}$, that either $l^*_{i,1}$ or $[l^*_{i,2}, l^*_{i,3}]$ is instantiated to a constant different from 0. Hence, for every clause, at least one of the literals $l_{i,1}, l_{i,2}, l_{i,3}$ is true in J. □