Semantics of SPARQL under OWL 2 Entailment Regimes

Egor V. Kostylev and Bernardo Cuenca Grau

Department of Computer Science, University of Oxford

Abstract. We study the semantics of SPARQL queries with optional matching features under entailment regimes. We argue that the normative semantics may lead to answers that are in conflict with the intuitive meaning of optional matching, where unbound variables naturally represent unknown information. We propose an extension of the SPARQL algebra that addresses these issues and is compatible with any entailment regime satisfying the minimal requirements given in the normative specification. We then study the complexity of query evaluation and show that our extension comes at no cost for regimes with an entailment relation of reasonable complexity. Finally, we show that our semantics preserves the known properties of optional matching that are commonly exploited for static analysis and optimisation.

1 Introduction

SPARQL became the standard language for querying RDF in 2008 [1]. Since then, the theoretical properties of SPARQL have been the subject of intensive research efforts and are by now relatively well-understood [2–7]. At the same time, SPARQL has become a core technology in practice, and most RDF-based applications rely on SPARQL endpoints for query formulation and processing.

The functionality of many such applications is enhanced by OWL 2 ontologies [8], which are used to provide background knowledge about the application domain, and to enrich query answers with implicit information. A new version of SPARQL, called SPARQL 1.1, was released in 2013 [9]. This new version captures the capabilities of OWL 2 by means of the so-called *entailment regimes* [10]: a flexible mechanism for extending SPARQL query answering to the W3C standards layered on top of RDF. A regime specifies which RDF graphs and SPARQL queries are legal (i.e., admissible) for the regime, as well as an entailment relation that unambiguously defines query answers for all legal queries and graphs.

The semantics of SPARQL under entailment regimes is specified for the conjunctive fragment, where queries are represented as *basic graph patterns* (i.e., sets of RDF triples with variables) and query answers are directly provided by the entailment relation of the regime. Roughly speaking, to check whether a mapping from variables of the query to nodes in the RDF graph is an answer to the query, one first transforms the query itself into an RDF graph by substituting each variable with the corresponding value, and then checks whether this graph is entailed in the regime by the original data graph [10–12].

When one goes beyond the basic fragment of SPARQL the language becomes considerably more complicated, but the effect of entailment regimes on the query semantics remains circumscribed to basic graph patterns [13, 14]. To evaluate a query one must first evaluate its component basic patterns using the relevant regime and then compose the results by means of the SPARQL algebra operations.

Of particular interest from both a theoretical and a practical perspective is the extension of the basic fragment of SPARQL with the *optional matching feature*, which is realised in the language by means of the `OPTIONAL` operator (abbreviated by OPT in this paper). This feature allows the optional information to be added to query answers only when the information is available in the RDF data graph: if the optional part of the query does not match the data, then the relevant variables are left unbounded in query answers.

One of the main motivations behind optional matching in SPARQL is to deal with the "lack of regular, complete structures in RDF graphs" (see [9] Section 6) and hence with the inherent incompleteness of information in RDF data sources where only partial information about the relevant Web resources is typically available. In this setting, an unbound variable in an answer mapping is naturally interpreted as a "null" value, meaning that there might exist a binding for this variable if we consider other information elsewhere on the Web, but none is currently available in the RDF graph at hand. Another (and slightly different) motivation for optional matching was to introduce a mechanism for "not rejecting solutions because some part of the query pattern does not match" [1]; in this sense, one would naturally expect optional matching to either extend solutions with the optional information, or to leave solutions unchanged. Both readings of optional matching coincide if we focus just on RDF, and they are faithfully captured by the normative semantics. In this paper we argue that they naturally diverge once we consider more sophisticated entailment regimes. Furthermore, the differences that arise can have a major impact on expected answers.

To make this discussion concrete, let us briefly discuss a simple example of an RDF graph representing the direct train lines between UK cities as well as ferry boat transfers from UK cities to international destinations. Let this graph be exhaustive in its description of rail connections, but much less so in what concerns ferry transfers. We may exploit optional matching to retrieve all direct train connections between cities X and Y, extended with ferry transfers from Y to other cities Z whenever possible. Under the normative semantics of SPARQL we may obtain answers $(\mathit{London}, \mathit{Oxford}, -)$ and $(\mathit{London}, \mathit{Holyhead}, -)$ provided the graph has information about direct train lines from *London* to both *Oxford* and *Holyhead*, but no matching can be found in the graph for ferry connections starting from *Oxford* or *Holyhead* to other cities. Suppose next that the data graph is extended to a graph corresponding to an OWL 2 ontology in which it is stated that inland cities do not have ferry connections, and that *Oxford* is an inland city. The ontology establishes a clear distinction between *Oxford* and *Holyhead*: whereas the former is inland and cannot have ferry connections, the latter may still well be (and indeed is) a coastal city offering a number of transfers to international destinations. The normative OWL 2 direct semantics

entailment regime, however, does not distinguish between the case of *Holyhead* (where the information about ferry connections is still unknown) and *Oxford* (where the information is certain), and both answers would be returned. In this way, the normative semantics adopts the reading of optional matching where the optional information is used to complete (but never discard) query answers. In contrast, under the reading of unbounded variables as placeholders for unknown information, one would naturally expect the answer on *Oxford* to be ruled out. Indeed, if our goal were to find rail to ferry transfers starting from *London* and terminating in *Dublin* by first querying this graph and then looking for the missing information elsewhere on the Web, discarding cities like *Oxford* on the first stage would significantly facilitate our task.

In this paper, we propose an alternative semantics for the OPT operator which adopts the aforementioned reading of optional matching as an incomplete "null". We call our semantics *strict*, which reflects the fact that it rules out those answers in which unbound variables in the optional part cannot be matched to *any* consistent extension of the input graph. Our semantics is given as an extension of the SPARQL algebra and hence satisfies the expected compositionality properties of algebraic query languages. Furthermore, it is backwards-compatible with the normative semantics for regimes in which all legal graphs are consistent, such as the RDF regime [10]. We also study the complexity of query evaluation and show that our extension comes at no cost for regimes in which entailment is not harder than query evaluation under normative semantics for the RDF regime. Finally, we show that our semantics preserves the known properties of optional matching that are commonly exploited for static analysis and optimisation.

This paper is an updated version of the work [15].

2 SPARQL 1.1 under Entailment Regimes

In this section, we formalise the syntax and normative semantics of a core fragment of SPARQL 1.1 with optional matching under entailment regimes. Our formalisation is based on the normative specification documents [9–11] and builds on the well-known foundational works on SPARQL [2, 3, 6].

2.1 Syntax

Let $\mathbf{I}$, $\mathbf{L}$, and $\mathbf{B}$ be infinite sets of *IRIs*, *literals*, and *blank nodes*, respectively. The set of *RDF terms* $\mathbf{T}$ is $\mathbf{I} \cup \mathbf{L} \cup \mathbf{B}$. An *RDF triple* is a triple $(s\ p\ o)$ from $\mathbf{T} \times \mathbf{I} \times \mathbf{T}$, with s called *subject*, p *predicate*, and o *object*. An *(RDF) graph* is a finite set of RDF triples. Assume additionally the existence of a countably infinite set $\mathbf{V}$ of variables disjoint from $\mathbf{T}$. A *triple pattern* is a tuple from $(\mathbf{T} \cup \mathbf{V}) \times (\mathbf{I} \cup \mathbf{V}) \times (\mathbf{T} \cup \mathbf{V})$. A *basic graph pattern (BGP)* is a finite set of triple patterns. *Built-in conditions* are conditions of the form $bound(?X)$, $?X = c$, and $?X = ?Y$ for $?X, ?Y \in \mathbf{V}$ and $c \in \mathbf{T}$, and their Boolean combinations.

Complex graph patterns are constructed from BGPs using a range of available operators that are applicable to graph patterns and built-in conditions. We

focus on the AND-OPT-FILTER fragment (i.e., we consider neither union nor projection), which is widely accepted to be the fundamental core of SPARQL [2]. In this setting, *graph patterns* are inductively defined as follows (e.g., see [11]):

1. every BGP is a graph pattern;
2. if P_1 and P_2 are graph patterns that share no blank nodes then $(P_1 \,\mathsf{AND}\, P_2)$ and $(P_1 \,\mathsf{OPT}\, P_2)$ are graph patterns (called AND and OPT *patterns*); and
3. if P is a graph pattern and R is a built-in condition, then $(P \,\mathsf{FILTER}\, R)$ is a graph pattern (called FILTER *pattern*).

In what follows $\mathsf{vars}(P)$ (respectively $\mathsf{triples}(P)$) denotes all the variables from $\mathbf{V}$ (respectively all triple patterns) that appear in a graph pattern P.

We conclude with the definition of a special class of graph patterns with intuitive behaviour [2]. A graph pattern is *well-designed* if and only if for each of its OPT subpatterns $(P_1 \,\mathsf{OPT}\, P_2)$ the pattern P_1 mentions all the variables of P_2 which appear outside this subpattern. Note that all graph patterns in the examples of this paper are well-designed.

2.2 Semantics of BGPs under Entailment Regimes

The semantics of graph patterns is defined in terms of *mappings*; that is, partial functions from variables $\mathbf{V}$ to terms $\mathbf{T}$. The *domain* $\mathsf{dom}(\mu)$ of a mapping μ is the set of variables on which μ is defined. The set of triples obtained from a BGP P by replacing each $?X$ from $\mathsf{dom}(\mu)$ by $\mu(?X)$ is denoted by $\mu(P)$.

Two mappings μ_1 and μ_2 are *compatible* (written as $\mu_1 \sim \mu_2$) if $\mu_1(?X) = \mu_2(?X)$ for all variables $?X$ which are in both $\mathsf{dom}(\mu_1)$ and $\mathsf{dom}(\mu_2)$. If $\mu_1 \sim \mu_2$, then we write $\mu_1 \cup \mu_2$ for the mapping obtained by extending μ_1 with μ_2 on variables undefined in μ_1. A mapping μ_1 is *subsumed* by a mapping μ_2 (written $\mu_1 \sqsubseteq \mu_2$) iff $\mu_1 \sim \mu_2$ and $\mathsf{dom}(\mu_1) \subseteq \mathsf{dom}(\mu_2)$. Finally, a set of mappings Ω_1 is *subsumed* by a set of mappings Ω_2 (written $\Omega_1 \sqsubseteq \Omega_2$) iff for each $\mu_1 \in \Omega_1$ there exists $\mu_2 \in \Omega_2$ such that $\mu_1 \sqsubseteq \mu_2$.

Based on [10], an *(entailment) regime* $\mathfrak{R}$ is a tuple $(\mathbf{R}, \mathcal{G}, \mathcal{P}, \mathcal{C}, [\![\cdot]\!])$, where

1. $\mathbf{R}$ is a set of *reserved* IRIs from $\mathbf{I}$;
2. $\mathcal{G}$ is the set of *legal* graphs;
3. $\mathcal{P}$ is the set of *legal* BGPs;
4. $\mathcal{C}$ is the set of *consistent* graphs, such that $\mathcal{C} \subseteq \mathcal{G}$; and
5. $[\![\cdot]\!]$ is the *query answering* function, that takes a graph G from $\mathcal{G}$ and a BGP P from $\mathcal{P}$ and returns either a set $[\![P]\!]_G$ of mappings μ such that $\mathsf{dom}(\mu) = \mathsf{vars}(P)$, if $P \in \mathcal{C}$; or Err, otherwise.

As in most theoretical works on SPARQL [2, 3, 6, 16], we assume that the query answering function returns a set of mappings, rather than a multiset.

The definitions of query answering and consistency in a regime are based on an entailment relation [10], which is also specified as part of the regime. We do not model the entailment relation explicitly, but assume two conditions that capture the effects of any reasonable entailment relation on legality and consistency. All regimes mentioned in the normative specification satisfy these properties and in this paper we consider only regimes that do so.

(C1) If graphs G, G_1 and G_2 are legal, and there is $h : \mathbf{T} \to \mathbf{T}$, preserving $\mathbf{R}$, such that $h(G_1 \cup G_2) \subseteq G$ then $G_1 \cup G_2$ is legal; if, in addition, G is in $\mathcal{C}$ then $G_1 \cup G_2$ is also in $\mathcal{C}$.

(C2) If a BGP P is in $\mathcal{P}$ then $\mu(P)$ is in $\mathcal{G}$ for any (total) $\mu : \mathbf{V} \to (\mathbf{T} \setminus \mathbf{R})$, such that $\mu(P)$ is a graph; if also $\mu(P)$ is in $\mathcal{C}$ then $\mu \in [\![P]\!]_{\mu(P)}$.

Condition (C1) formalises (a weak form of) the monotonicity of legality and consistency: an illegal graph that is a union of legal ones cannot be made legal by identifying and renaming of non-reserved terms or adding triples to it; moreover, a similar property holds for consistency. Condition (C2) guarantees, that "freezing" variables of a legal BGP to non-reserved terms gives us a legal graph, and, moreover, if such a graph is consistent, then the answer of the BGP on this graph contains the mapping corresponding to the "freezing".

The notions introduced in the remainder of this paper are parameterised with a regime $\mathfrak{R}$, which is not mentioned explicitly for brevity.

2.3 Normative Semantics under Entailment Regimes

Following [2], now we show how the query answering function $[\![\cdot]\!]$ extends to complex graph patterns (we refer to [2] for details). A mapping μ *satisfies* a built-in condition R, denoted $\mu \models R$, if one of the following holds:

1. R is $bound(?X)$ and $?X \in \mathsf{dom}(\mu)$; or
2. R is $?X = c$, $?X \in \mathsf{dom}(\mu)$, and $\mu(?X) = c$; or
3. R is $?X = ?Y$, $?X \in \mathsf{dom}(\mu)$, $?Y \in \mathsf{dom}(\mu)$, and $\mu(?X) = \mu(?Y)$; or
4. R is an evaluating to true Boolean combination of other built-in conditions.

The *join*, *difference*, and *left outer join* of sets of mappings Ω_1, Ω_2 are as follows:

$$\Omega_1 \bowtie \Omega_2 = \{\mu_1 \cup \mu_2 \mid \mu_1 \in \Omega_1 \text{ and } \mu_2 \in \Omega_2 \text{ such that } \mu_1 \sim \mu_2\},$$
$$\Omega_1 \setminus \Omega_2 = \{\mu_1 \mid \mu_1 \in \Omega_1, \text{ there is no } \mu_2 \in \Omega_2 \text{ such that } \mu_1 \sim \mu_2\},$$
$$\Omega_1 \mathbin{⟕} \Omega_2 = (\Omega_1 \bowtie \Omega_2) \cup (\Omega_1 \setminus \Omega_2).$$

A graph pattern is *legal* for a regime $\mathfrak{R}$ if all the BGPs it contains are legal. The *normative query answering* function $[\![\cdot]\!]^n$ is inductively defined for all legal graph patterns P on the base of $[\![\cdot]\!]$ as follows. For graphs G from $\mathcal{C}$ we have:

1. if P is a BGP then $[\![P]\!]^n_G = [\![P]\!]_G$;
2. if P is $(P_1 \ \mathsf{AND}\ P_2)$ then $[\![P]\!]^n_G = [\![P_1]\!]^n_G \bowtie [\![P_2]\!]^n_G$;
3. if P is $(P_1 \ \mathsf{OPT}\ P_2)$ then $[\![P]\!]^n_G = [\![P_1]\!]^n_G \mathbin{⟕} [\![P_2]\!]^n_G$; and
4. if P is $(P' \ \mathsf{FILTER}\ R)$ then $[\![P]\!]^n_G = \{\mu \mid \mu \in [\![P']\!]^n_G \text{ and } \mu \models R\}$.

If $G \notin \mathcal{C}$ then $[\![P]\!]^n_G = Err$ for any graph pattern P (which again coincides with $[\![P]\!]_G$ when P is a BGP). Note, that by these definitions $\mu \in [\![P]\!]^n_G$ implies that $\mathsf{dom}(\mu) \subseteq \mathsf{vars}(P)$, but this inclusion may be strict if P contains OPT operator.

Two legal patterns P_1 and P_2 are *equivalent (under normative semantics)*, denoted by $P_1 \equiv^n P_2$, if $[\![P_1]\!]^n_G = [\![P_2]\!]^n_G$ for every RDF graph $G \in \mathcal{G}$.

3 On Optional Matching Under the Normative Semantics

One of the main motivations for optional matching in SPARQL was to deal with the "lack of regular, complete structures in RDF graphs" [9]. Indeed, RDF data

is loosely structured, and in many applications it is not satisfactory to reject an answer if some relevant information is missing. For example, if we are interested in retrieving the names, emails, and websites of employees, we may not want to discard a partial answer involving the name and email address of an employee merely because the information on the employee's website is not available in the graph. The normative semantics was designed to deal with such situations: the optional information is included in query answers only when the information is available; otherwise, the relevant variables are left unbounded. An unbound variable in an answer is thus a manifestation of inherent incompleteness of RDF data sources, and the missing information is interpreted as unknown.

This natural interpretation of query results, however, no longer holds if the query is evaluated under certain entailment regimes, as we illustrate next by means of examples. In these and all other examples given later on, we focus on the OWL 2 direct semantics regime. In order for an RDF graph to be legal for this regime, it must correspond to an OWL 2 ontology; similarly, legal BGPs must correspond to an extended ontology in which variables are allowed [10]. Thus, in the examples we express RDF graphs and BGPs in (extended) OWL 2 functional syntax, and use words "ontology" and "graph" interchangeably (we also omit declaration axioms in ontologies and BGPs to avoid clatter).

Example 1. Consider the OWL 2 ontology $\mathcal{O}_1$ consisting of the axioms

$$\mathsf{ClassAssertion}(\mathit{InlandCity}\ \mathit{Oxford}),\quad \mathsf{PropertyAssertion}(\mathit{train}\ \mathit{London}\ \mathit{Oxford}),$$
$$\mathsf{ClassAssertion}(\mathit{CoastalCity}\ \mathit{Holyhead}),\mathsf{PropertyAssertion}(\mathit{train}\ \mathit{London}\ \mathit{Holyhead}),$$
$$\mathsf{PropertyDomain}(\mathit{ferry}\ \mathit{CoastalCity}),\quad \mathsf{DisjointClasses}(\mathit{CoastalCity}\ \mathit{InlandCity}).$$

Consider also the following graph pattern P_1, which we wish to evaluate over $\mathcal{O}_1$:

$$\mathsf{PropertyAssertion}(\mathit{train}\quad ?X\quad ?Y)\ \mathsf{OPT}\ \mathsf{PropertyAssertion}(\mathit{ferry}\quad ?Y\quad ?Z).$$

Intuitively, solutions to P_1 provide direct train lines from city X to city Y as well as, optionally, the ferry transfers from Y to other cities Z. Under the normative semantics, the BGPs in P_1 are evaluated separately. In particular, the optional BGP is evaluated to the empty set, and $[\![P_1]\!]^n_{\mathcal{O}_1} = \{\mu_1, \mu_2\}$, where

$$\mu_1 = \{?X \mapsto \mathit{London}, ?Y \mapsto \mathit{Oxford}\} \text{ and } \mu_2 = \{?X \mapsto \mathit{London}, ?Y \mapsto \mathit{Holyhead}\}.$$

In both answers, variable $?Z$ is unbounded and hence we conclude that $\mathcal{O}_1$ contains no relevant information about ferry connections starting from *Oxford* or *Holyhead*. However, the nature of the lack of such information is fundamentally different. On the one hand, the connections from *Holyhead* (e.g., to *Dublin*) are missing from $\mathcal{O}_1$ just by the incompleteness of the information in the graph, which is usual in (and also a feature of) Semantic Web applications. On the other hand, *Oxford* cannot have a ferry connection because it is a landlocked city, and hence the information about its (lack of) ferry connections is certain. Thus, the normative semantics cannot distinguish between unknown and non-existent ferry connections. However, if we adhere to the reading of unbounded variables as incomplete information or "nulls", then μ_1 should not be returned as an answer.

The issues described in this example become even more apparent in cases where the optional part alone cannot be satisfied, as in the following example.

Example 2. Consider the ontology $\mathcal{O}_2$ with the axioms

$$\mathsf{ClassAssertion}(\mathit{Person}\ \mathit{Peter}), \quad \mathsf{DisjointProperties}(\mathit{hasFather}\ \mathit{hasMother}).$$

Furthermore, consider the following pattern P_2:

$$\mathsf{ClassAssertion}(\mathit{Person}\ ?X)\ \mathsf{OPT}\ (\{ \\ \mathsf{PropertyAssertion}(\mathit{hasFather}\ ?X\ ?Y), \mathsf{PropertyAssertion}(\mathit{hasMother}\ ?X\ ?Y)\}).$$

The optional BGP does not match to anything, so $[\![P_2]\!]^n_{\mathcal{O}_2}$ consists of $\{?X \mapsto \mathit{Peter}\}$. However, this BGP is in contradiction with the disjointness axiom: under the OWL 2 regime, no solution to P_2 exists for any ontology with this axiom.

As these examples suggest, if we interpret unbound variables in answers to queries with optional parts as an indication of unknown information in the data graph, then the normative semantics may yield counter-intuitive answers. At the core of this issue is the inability of the normative semantics to distinguish between answers in which it is possible to assign values to the missing optional part (a natural reflection of incompleteness in the data), and those where this is impossible (a reflection that the missing information is incompatible with the answer). This distinction is immaterial for regimes without inconsistencies, but it becomes apparent in more sophisticated regimes, such as those based on OWL 2.

4 Semantics of Strict Optional Matching

In this section, we propose our novel semantics for optional matching under regimes. In a nutshell, our semantics addresses the issues described in Section 3 by ruling out those answer mappings where unbound variables in the optional part cannot be matched to any consistent extension of the input graph. Our semantics is therefore *strict*, in the sense that only answers in which unbound variables are genuine manifestations of incompleteness in the data are returned.

4.1 Definition of Strict Semantics

We start by introducing the notion of a frozen RDF graph for a pattern P and a mapping μ. Roughly speaking, this graph is obtained by taking all the triple patterns in P and transforming them into RDF triples by applying the extension of μ where unbounded variables are "frozen" to arbitrary fresh constants.

Definition 1. *Let $\mathfrak{R} = (\mathbf{R}, \mathcal{G}, \mathcal{P}, \mathcal{C}, [\![\cdot]\!])$ be an entailment regime. Let P be a legal graph pattern, and let μ be a mapping from variables $\mathbf{V}$ to RDF terms $\mathbf{T}$. Then, the* freezing *G^P_μ of P under μ is the RDF graph $\bar{\mu}(\mathsf{triples}(P))$, where $\bar{\mu}$ is the mapping that extends μ by assigning each variable in $\mathsf{vars}(P)$, which is not in $\mathsf{dom}(\mu)$, to a globally fresh IRI from $\mathbf{I}$ not belonging to $\mathbf{R}$.*

The freezing G^P_μ depends only on the candidate mapping μ and the triple patterns occurring in P; thus, it does not depend either on the specific operators used in P, or on the RDF graph over which the query pattern is to be evaluated.

Example 3. For pattern P_1 and mappings μ_1, μ_2 from Example 1 the freezings $G^{P_1}_{\mu_1}$ and $G^{P_1}_{\mu_2}$ have the following form, for fresh IRIs k and ℓ:

$\{\mathsf{PropertyAssertion}(\mathit{train\ London\ Oxford}), \mathsf{PropertyAssertion}(\mathit{ferry\ Oxford\ k})\}$,
$\{\mathsf{PropertyAssertion}(\mathit{train\ London\ Holyhead}), \mathsf{PropertyAssertion}(\mathit{ferry\ Holyhead\ \ell})\}$.

Intuitively, the freezing represents the simplest and most general RDF graph over which all the undefined variables in a given solution mapping could be bounded to concrete values. Thus, if G^P_μ together with the input graph G is not a consistent graph for the relevant regime, we can conclude, using condition (C1) of the regime, that the undefined variables in μ will never be matched to concrete values in any consistent extension of G and hence μ should be ruled out as an answer. On the other hand, if $G \cup G^P_\mu$ is consistent, then such an extension exists and, by condition (C2), the undefined variables can be mapped in this extension.

Definition 2. *Let $\mathfrak{R} = (\mathbf{R}, \mathcal{G}, \mathcal{P}, \mathcal{C}, [\![\cdot]\!])$ be an entailment regime. A mapping μ is* $\mathfrak{R}$-admissible *for a graph $G \in \mathcal{C}$ and legal graph pattern P if $G \cup G^P_\mu$ is a graph belonging to $\mathcal{C}$. The set of all $\mathfrak{R}$-admissible mappings for a consistent graph G and a legal graph pattern P is denoted as $Adm(G, P)$.*

Example 4. Clearly, $\mathcal{O}_1 \cup G^{P_1}_{\mu_1}$ is inconsistent since ferries only depart from coastal cities, but Oxford is an inland city. In contrast, $\mathcal{O}_1 \cup G^{P_1}_{\mu_2}$ is consistent. Thus, we have $\mu_1 \notin Adm(\mathcal{O}_1, P)$, but $\mu_2 \in Adm(\mathcal{O}_1, P)$.

We are now ready to formalise our semantics.

Definition 3. *Given an entailment regime $\mathfrak{R} = (\mathbf{R}, \mathcal{G}, \mathcal{P}, \mathcal{C}, [\![\cdot]\!])$, the* strict query answering *function $[\![\cdot]\!]^s$ is defined for legal graph patterns P and $G \in \mathcal{C}$ as follows:*

1. *if P is a BGP then $[\![P]\!]^s_G = [\![P]\!]_G$;*
2. *if P is P_1 AND P_2 then $[\![P]\!]^s_G = ([\![P_1]\!]^s_G \bowtie [\![P_2]\!]^s_G) \cap Adm(G, P)$;*
3. *if P is P_1 OPT P_2 then $[\![P]\!]^s_G = ([\![P_1]\!]^s_G \bowtie [\![P_2]\!]^s_G) \cap Adm(G, P)$; and*
4. *if P is P' FILTER R then $[\![P]\!]^s_G = \{\mu \mid \mu \in [\![P']\!]^s_G \text{ and } \mu \models R\}$,*

where $\cap$ denotes the standard set-theoretic intersection. If $G \notin \mathcal{C}$ then $[\![P]\!]^s_G = Err$ for any graph pattern P. Finally, legal patterns P_1 and P_2 are equivalent (under strict semantics)*, written $P_1 \equiv^s P_2$, if $[\![P_1]\!]^s_G = [\![P_2]\!]^s_G$ for any legal G.*

Example 5. The strict semantics behaves as expected for our examples: $[\![P_1]\!]^s_{\mathcal{O}_1} = \{\mu_1\}$ holds for $\mathcal{O}_1$ and P_1 from Example 1, while $[\![P_2]\!]^s_{\mathcal{O}_2} = \emptyset$ holds for Example 2.

The strict and normative semantics coincide in two limit cases. First, if the entailment regime does not allow for inconsistent graphs (i.e., if $\mathcal{C} = \mathcal{G}$) as is the case for the RDF regime [10], then $[\![P]\!]^s_G = [\![P]\!]^n_G$ for every legal pattern P and graph G. Second, if the relevant pattern P is OPT-free then the freezing for

every candidate answer mapping contains no fresh IRIs and is $\mathfrak{R}$-entailed by G; thus, we again have $[\![P]\!]^s_G = [\![P]\!]^n_G$ for every legal graph G.

Thus, the difference between the normative semantics $[\![\cdot]\!]^n$ and strict semantics $[\![\cdot]\!]^s$ manifests only for regimes that admit inconsistency, and is circumscribed to the presence of OPT in graph patterns, where non-admissible mappings are excluded in the case of the strict semantics. Note, however, that even if a mapping μ_1 (respectively μ_2) is admissible for a subpattern P_1 (respectively P_2) containing OPT, it is possible for $\mu_1 \cup \mu_2$ not to be admissible for the joined pattern $P = P_1$ AND P_2. Thus, the admissibility restriction is also explicitly reflected in the semantics of AND given in Definition 3. This is illustrated in the example given next.

Example 6. Consider ontology $\mathcal{O}_3$, consisting of the axioms

SubClassOf(
  IntersectionOf(SomeValuesFrom(*husband* Thing) SomeValuesFrom(*wife* Thing))
  Nothing),
ClassAssertion(*Person Mary*).

The first axiom establishes that a person cannot have both a husband and a wife. Consider also the following well-designed graph pattern P_3:

(ClassAssertion(*Person* ?*X*) OPT (PropertyAssertion(*husband* ?*X* ?*Y*))) AND
(ClassAssertion(*Person* ?*X*) OPT (PropertyAssertion(*wife* ?*X* ?*Z*))).

Clearly, $\mu = \{?X \mapsto \mathit{Mary}\}$ belongs to the strict answer to each of the OPT subpatterns of P_3 since each of them independently can match to a consistent extension of $\mathcal{O}_3$. However, μ is not admissible for P_3 since *Mary* has both a husband and a wife in $G^{P_3}_\mu$, and hence $\mathcal{O}_3 \cup G^{P_3}_\mu$ is inconsistent. Thus, $[\![P_3]\!]^s_{\mathcal{O}_3} = \emptyset$.

4.2 Comparing the Normative and Strict Semantics

Our previous examples support the expected behaviour of our semantics, namely that its effect is circumscribed to filtering out problematic answers returned under the normative semantics. We next formally show that our semantics behaves as expected *in general*, provided that we restrict ourselves to well-designed patterns and negation-free FILTER expressions (which are rather mild restrictions).

It is known that patterns which are not well-designed easily lead to unexpected answers, even under the normative semantics (we refer to [2] for a detailed discussion). Therefore, it comes at no surprise that the intuitive behaviour of our semantics is only guaranteed under this assumption.

Theorem 1. *Let $\mathfrak{R} = (\mathbf{R}, \mathcal{G}, \mathcal{P}, \mathcal{C}, [\![\cdot]\!])$ be an entailment regime. The inclusion $[\![P]\!]^s_G \sqsubseteq [\![P]\!]^n_G$ holds for any graph G from $\mathcal{C}$ and any legal well-designed graph pattern P which does not use negation in* FILTER *expressions.*

Note that Theorem 1 is formulated in terms of subsumption, instead of set-theoretic containment. The rationale behind this formulation is clarified next.

Example 7. Consider the ontology $\mathcal{O}_1'$, which is obtained from $\mathcal{O}_1$ in Example 1 by removing all axioms involving *Holyhead*, and adding the axiom

$$\mathsf{PropertyAssertion}(\mathit{bus}\ \mathit{Canterbury}\ \mathit{London}).$$

Consider also the following graph pattern P_1':

$$\mathsf{PropertyAssertion}(\mathit{bus}\ ?U\ ?X)\ \mathsf{OPT}$$
$$(\mathsf{PropertyAssertion}(\mathit{train}\ ?X\ ?Y)\ \mathsf{OPT}\ \mathsf{PropertyAssertion}(\mathit{ferry}\ ?Y\ ?Z)).$$

The mapping $\mu = \{?U \mapsto \mathit{Canterbury}, ?X \mapsto \mathit{London}, ?Y \mapsto \mathit{Oxford}\}$ is returned by the normative semantics. As already discussed, Oxford is an inland city and hence cannot have ferry connections; thus, μ is not returned under strict semantics. However, it may be possible to reach a ferry connection from London (although none is given), and hence the answer $\mu' = \{?U \mapsto \mathit{Canterbury}, ?X \mapsto \mathit{London}\}$ is returned instead of μ under strict semantics. Clearly, μ' is not a normative answer and $[\![P_1']\!]^s_{\mathcal{O}_1'} \not\subseteq [\![P_1']\!]^n_{\mathcal{O}_1'}$; however, $\mu' \sqsubseteq \mu$ and $[\![P_1']\!]^s_{\mathcal{O}_1'} \sqsubseteq [\![P_1']\!]^n_{\mathcal{O}_1'}$.

5 Computational Properties and Static Optimisation

In this section we first study the computational properties of our semantics. We show that the complexity of graph pattern evaluation under strict and normative semantics coincide, provided that consistency checking is feasible in PSPACE for the regime at hand. Then we focus on static query analysis, and in particular on pattern equivalence. We show that the key equivalence-preserving transformation rules that have been proposed for static optimisation of SPARQL queries continue to hold if we consider equivalence under strict semantics.

5.1 Complexity of Strict Graph Pattern Evaluation

Recall that the graph pattern evaluation is the key problem in SPARQL. In the context of entailment regimes, it is defined as follows, where x is either n or s.

Graph Pattern Evaluation	
Input:	Regime $\mathfrak{R}$, legal graph G, legal graph pattern P, and mapping μ.
Question:	Is $\mu \in [\![P]\!]^x_G$ under the regime $\mathfrak{R}$?

Here, when we say that regime $\mathfrak{R}$ is a part of the input, we mean that it includes two oracle functions checking consistency of legal graphs and evaluating legal BGPs over legal graphs, respectively. In what follows, we refer to the problem as Normative if $x = n$, and as Strict if $x = s$.

It is known that the normative graph pattern evaluation problem is in PSPACE for the RDF regime [2]. We next argue that membership in PSPACE holds in general for any regime satisfying the basic properties discussed in Section 2 and for both normative and strict versions of the problem, provided that the complexity of both oracles of the regime is in PSPACE.

Theorem 2. NORMATIVE *and* STRICT GRAPH PATTERN EVALUATION *problems are in* PSPACE*, provided the oracles associated to input regimes are in* PSPACE*.*

Consequently, the use of our strict semantics does not increase the computational complexity for reasonable regimes. In particular, it follows directly from Theorem 2 that the evaluation problem is in PSPACE under both semantics for the tractable entailment regimes associated to the OWL 2 profiles [17].

It is also known that graph pattern evaluation under normative semantics is PSPACE-hard for the RDF regime [2]. To formulate a general hardness result that holds for *any* regime we would need to require additional properties for a regime to qualify as "reasonable". In order not to unnecessarily complicate the presentation, we simply point out that PSPACE-hardness holds for all the regimes in the specification under both normative and strict semantics [10].

5.2 Static Analysis and Optimisation

Static analysis and optimisation of SPARQL queries has received significant attention in recent years [4, 6, 18–20]. A key ingredient for optimisation is the availability of a comprehensive catalog of equivalence-preserving transformation rules for SPARQL patterns. A rich set of such equivalences for normative semantics and RDF regime is established in [2] and [4]. Some of these equivalences, such as idempotence, commutativity, and associativity of the AND operator, hold without any restrictions (for our core fragment of SPARQL). However, those that involve OPT are more intricate and hold only for well-designed patterns. The claim of this section is that these equivalences continue to hold for any entailment regime, under both normative and strict semantics.

Theorem 3. *The following equivalences hold for any entailment regime, provided the graph patterns on both sides are legal and well-designed, for* $x \in \{n, s\}$*:*

$$(P_1 \,\mathsf{OPT}\, P_2)\,\mathsf{FILTER}\, R \equiv^x (P_1 \,\mathsf{FILTER}\, R)\,\mathsf{OPT}\, P_2,$$
$$P_1 \,\mathsf{AND}\, (P_2 \,\mathsf{OPT}\, P_3) \equiv^x (P_1 \,\mathsf{AND}\, P_2)\,\mathsf{OPT}\, P_3,$$
$$(P_1 \,\mathsf{OPT}\, P_2)\,\mathsf{OPT}\, P_3 \equiv^x (P_1 \,\mathsf{OPT}\, P_3)\,\mathsf{OPT}\, P_2.$$

6 Conclusion

In this paper, we have proposed a novel semantics for optional matching in SPARQL under entailment regimes where unbound variables in answer mappings are naturally interpreted as "null" values. Our *strict* semantics has been designed to deal in a faithful way with the "lack of regular, complete structures in RDF graphs" and hence with the fundamental incompleteness of information on the Semantic Web [1]. We believe that both strict and normative semantics are valid, but one may be more appropriate than the other in certain applications. Both semantics are compatible at a fundamental level and it would be possible to exploit them in the same application by letting users commit to one or the other explicitly when posing queries. Integrating them in a clean way from a syntactic point of view is more tricky, and it is something we leave for future investigation.

References

1. Prud'hommeaux, E., Seaborne, A.: SPARQL query language for RDF. W3C Recommendation (2008) Available at http://www.w3.org/TR/rdf-sparql-query/.
2. Pérez, J., Arenas, M., Gutierrez, C.: Semantics and complexity of SPARQL. ACM Trans. Database Syst. **34**(3) (2009)
3. Angles, R., Gutierrez, C.: The expressive power of SPARQL. In: ISWC. (2008) 114–129
4. Schmidt, M., Meier, M., Lausen, G.: Foundations of SPARQL query optimization. In: ICDT. (2010) 4–33
5. Arenas, M., Pérez, J.: Querying semantic web data with SPARQL. In: PODS. (2011) 305–316
6. Letelier, A., Pérez, J., Pichler, R., Skritek, S.: Static analysis and optimization of semantic web queries. ACM Trans. Database Syst. **38**(4) (2013) 25
7. Polleres, A.: From SPARQL to rules (and back). In: WWW. (2007) 787–796
8. Motik, B., Patel-Schneider, P.F., Parsia, B.: OWL 2 Web Ontology Language Structural Specification and Functional-style Syntax. W3C Recommendation (2012) Available at http://www.w3.org/TR/owl2-syntax/.
9. W3C SPARQL Working Group: SPARQL 1.1 Query language. W3C Recommendation (2013) Available at http://www.w3.org/TR/sparql11-query/.
10. Glimm, B., Ogbuji, C.: SPARQL 1.1 Entailment Regimes. W3C Recommendation (2013) Available at http://www.w3.org/TR/sparql11-entailment/.
11. Glimm, B., Krötzsch, M.: SPARQL beyond subgraph matching. In: ISWC. (2010) 241–256
12. Kollia, I., Glimm, B.: Optimizing SPARQL query answering over OWL ontologies. J. Artif. Intell. Res. **48** (2013) 253–303
13. Kontchakov, R., Rezk, M., Rodriguez-Muro, M., Xiao, G., Zakharyaschev, M.: Answering SPARQL queries over databases under OWL 2 QL entailment regime. In: ISWC. (2014) 552–567
14. Arenas, M., Gottlob, G., Pieris, A.: Expressive languages for querying the semantic web. In: PODS. (2014) 14–26
15. Kostylev, E.V., Cuenca Grau, B.: On the semantics of SPARQL queries with optional matching under entailment regimes. In: ISWC. (2014)
16. Pérez, J., Arenas, M., Gutierrez, C.: Semantics and complexity of SPARQL. In: ISWC. (2006) 30–43
17. Motik, B., Cuenca Grau, B., Horrocks, I., Wu, Z., Fokoue, A., Lutz, C.: OWL 2 Web Ontology Language: Profiles (27 October 2009) Available at http://www.w3.org/TR/owl2-profiles/.
18. Chekol, M.W., Euzenat, J., Genevès, P., Layaïda, N.: SPARQL query containment under RDFS entailment regime. In: IJCAR. (2012) 134–148
19. Chekol, M.W., Euzenat, J., Genevès, P., Layaïda, N.: SPARQL query containment under $\mathcal{SHI}$ axioms. In: AAAI. (2012)
20. Chekol, M.W., Euzenat, J., Genevès, P., Layaïda, N.: Evaluating and benchmarking SPARQL query containment solvers. In: ISWC. (2013) 408–423