Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence

On Completeness Classes for Query Evaluation on Linked Data

Andreas Harth and **Sebastian Speiser**
Institute AIFB
Karlsruhe Institute of Technology (KIT)
76128 Karlsruhe, Germany

Abstract

The advent of the Web of Data kindled interest in link-traversal (or lookup-based) query processing methods, with which queries are answered via dereferencing a potentially large number of small, interlinked sources. While several algorithms for query evaluation have been proposed, there exists no notion of completeness for results of so-evaluated queries. In this paper, we motivate the need for clearly-defined completeness classes and present several notions of completeness for queries over Linked Data, based on the idea of authoritativeness of sources, and show the relation between the different completeness classes.

1 Introduction

A tenet in work on query evaluation and reasoning on the Semantic Web is the open world assumption (OWA): given the size and decentralised nature of the web, it is impossible to achieve complete results. Thus, an answer to a query or reasoning task is therefore always a subset of all possible answers. To what degree that subset is complete is left open.

In this paper we define more fine-grained completeness classes for query answers. We do so in the context of Linked Data, a set of principles detailing how to publish graph-structured data on the web. Recently developed query evaluation algorithms traverse the Web of Data and at the same time record answers to a query (Hartig, Bizer, and Freytag 2009; Harth et al. 2010; Ladwig and Tran 2010; Haase, Mathäß, and Ziller 2010; Umbrich, Hogan, and Polleres 2011). These algorithms, however, lack a clear specification of result completeness.

Thus, we present several completeness classes that rigorously define which sources may contribute to an answer to Linked Data queries. Doing so has a number of benefits; with a clear specification of complete answers:

- users know what to expect from a query evaluation algorithm;
- different algorithms become comparable;
- algorithms can have crisp termination criteria;
- developers can devise optimised algorithms that exclude irrelevant sources;
- systems can implement operations that rely on checking for the absence of results, such as negation-as-failure; and
- certain statements can be restricted to trustworthy sources.

Our specific contributions are:

- We extend and generalise the idea of authoritative sources from (Hogan, Harth, and Polleres 2009); based on authority, we define the notion of completeness for sources.
- We define three completeness classes for triple patterns and conjunctive queries: one that considers the entire web, one that considers documents in the surrounding of sources derived from the query and one that considers documents based on the query execution.
- We show how the completeness classes related to each other.

Please note that our results apply to both web and intranet environments, as long as data providers follow Linked Data principles. Our results also apply to Dataspaces (Franklin, Halevy, and Maier 2005) without central registries.

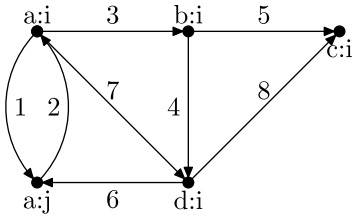
The remainder of the paper is organised as follows: Section 2 provides an example. Section 3 introduces necessary notation and definitions. Section 4 presents the idea of authoritative documents. Section 5 explains how query parts can be answered completely, while Section 6 considers entire queries under three completeness types. Section 7 explains the relation between the completeness classes. Section 8 presents related work, and Section 9 concludes.

2 Example

We begin with an example of an RDF graph and a query over that graph.

Example 1. *Figure 1 shows an example RDF graph. We use labels* `a:i`, `a:j`, `b:i`, `c:i`, `d:i`, `p:i` *to denote resources, and numbers* $1 \ldots 8$ *to denote triples. Now, assume the query* Q_{ex} *depicted in Figure 2. The overall goal is to find bindings* μ *to the variables in the query.*

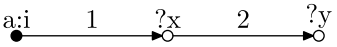
A system with access to the entire graph in Figure 1 could evaluate the query using standard query processing techniques. However, on the Linked Data web, the graph is distributed across multiple sources in form of web-accessible RDF files (henceforth called documents).



No	Triple
1	a:i p:i a:j .
2	a:j p:i a:i .
3	a:i p:i b:i .
4	b:i p:i d:i .
5	b:i p:i c:i .
6	d:i p:i a:j .
7	a:i p:i d:i .
8	d:i p:i c:i .

Document	Triple
a	1, 2, 3
b	4
c	5
d	6, 7
e	8
p	-

Figure 1: An example RDF graph with six IRIs and eight triples. Numbers denote triples.



No	Triple pattern
1	a:i p:i ?x .
2	?x p:i ?y .

Figure 2: An acyclic query consisting of two triple patterns.

Assume a, b, c, d, e, p are documents; the right table in Figure 1 lists the six documents and the triples they contain. Please note that the assignment is rather arbitrary and can differ, as maintainers of documents are free to decide which triples they host. One thing we can assume, though, is that identifiers are associated with documents (as mandated by the Linked Data principles (Berners-Lee 2006)). Thus, we can assume that we get some triples with identifer a:i when looking up the corresponding document a (and similarly, a:j for a and b:i for b, c:i for c).

Table 1: Bindings for variables in Q_{ex}, including which triples and documents contributed to bindings.

μ	μ(?x)	μ(?y)	Triple	Document
μ_1	a:j	a:i	1, 2	a
μ_2	b:i	d:i	3, 4	a , b
μ_3	d:i	a:j	7, 6	d

Now, to answer the query, we perform a lookup on a, which results in triples 1 - 3 from which we can derive bindings a:j and b:i for ?x, and a:i for ?y. Next, we perform a lookup on b which returns triple 4, from which we can derive d:i for ?y. We also perform a lookup on d which returns triple 7 and 6, from which we can derive d:i for ?x and a:j for ?y. As a result, we arrive at the bindings as depicted in Table 1. Please note that bindings $\{?x \mapsto b:i, ?y \mapsto c:i\}$ (via sources a and c) and bindings $\{?x \mapsto a:i, ?y \mapsto c:i\}$ (via sources d and e) cannot be reached via link traversal.

The example illustrates a couple of issues: first, a link-traversal algorithm cannot discover documents which are not referenced in any already known document. Second, assuming a larger graph, the link traversal process could actually go on for a long time, as more and more new documents are discovered and accessed. In the rest of the paper we show how to decide which subset of documents should be accessed to derive answers to queries.

3 Preliminaries

We introduce basic notation to clarify our understanding of RDF, Linked Data and queries. We stay close to similar definitions as found in (Pérez, Arenas, and Gutierrez 2009; Umbrich, Hogan, and Polleres 2011).

Definition 1 (RDF Terms, Triple, Graph). *The set of RDF terms consists of the set of IRIs $\mathcal{I}$, the set of blank nodes $\mathcal{B}$ and the set of literals $\mathcal{L}$. A triple $(s, p, o) \in \mathcal{T} = (\mathcal{I} \cup \mathcal{B}) \times \mathcal{I} \times (\mathcal{I} \cup \mathcal{B} \cup \mathcal{L})$ is called an RDF triple, where s is the subject, p is the predicate and o is the object. We denote by $\mathsf{s}(t)$ the subject, $\mathsf{p}(t)$ the predicate and $\mathsf{o}(t)$ the object of a triple t. We denote by $\mathsf{iris}(t)$ all IRIs from a triple t, and by $\mathsf{terms}(t)$ all RDF terms. A set of triples is called RDF graph; $\mathcal{G} = 2^{\mathcal{T}}$ is the set of all graphs.*

Next, we define ways for accessing RDF graphs published on the web as Linked Data. A key characteristic of Linked Data is the correspondence between an identifier and a source; i.e., the name for a thing (non-information resource) is associated with the document where one can find related information (information resource).

Definition 2 (Information Resource, Lookup). *Let $\mathcal{I}_{\mathcal{I}} \subseteq \mathcal{I}$ be the set of all information resources. The set of all non-information resources is defined as $\mathcal{I}_{\mathcal{N}} = \mathcal{I} \setminus \mathcal{I}_{\mathcal{I}}$. The function $\mathsf{deref} : \mathcal{I}_{\mathcal{I}} \mapsto \mathcal{G}$ models a Linked Data lookup and returns the graph represented in a document, or the empty set if none found, e.g., if there is a timeout or the document returns non-RDF content.*

We use the terms information resource and document interchangably. To be able to model the assocation between non-information Resources and information resources we introduce the concept of correspondence.

Definition 3 (Correspondence). *The function $\mathsf{co} : \mathcal{I} \mapsto \mathcal{I}_{\mathcal{I}}$ associates to a resource its information resource. For inputs from $\mathcal{I}_{\mathcal{I}}$, co behaves as the identity function.*

Determining the kind of an IRI is not always possible from the outset; a HTTP lookup clarifies the kind of IRI. We define a high-level function which provide abstractions on low-level functionality pertaining to protocol-level issues. Thus, in co we abstract away the following cases:

1. remove the local identifier from an IRI (i.e., strip everything after the # symbol);
2. dereference the IRI and follow redirects (HTTP status codes 30x);

3. dereference the IRI and parse the `Content-Location` header to yield the canonical name;
4. no-op: do nothing if the IRI is an information resource.

Options 1-3 may be called never or repreatedly, to ultimately arrive at 4. The co function may never return due to infitite redirects; in practice, one sets a limit on how often co can be applied.

Definition 4 (Variable, Triple Pattern). *Let $\mathcal{V}$ be a set of variables; variables bind to RDF terms from $\mathcal{I} \cup \mathcal{B} \cup \mathcal{L}$. A triple $p \in (\mathcal{I} \cup \mathcal{V}) \times (\mathcal{I} \cup \mathcal{V}) \times (\mathcal{I} \cup \mathcal{L} \cup \mathcal{V})$ is called triple pattern. We omit blank nodes from triple patterns for ease of exposition. $\mathcal{P}$ is the set of all triple patterns. We denote by $\mathsf{vars}(p)$ all variables from a triple pattern p.*

Definition 5 (Variable Binding). *Let $\mathcal{M}$ be the set of all partial functions $\mu\colon \mathcal{V} \mapsto \mathcal{I} \cup \mathcal{B} \cup \mathcal{L}$. A function $\mu \in \mathcal{M}$ is called a variable binding.*

Definition 6 (Basic Graph Pattern (BGP)). *A BGP (or just query) is a set $Q \subset \mathcal{P}$. The set of all queries is $\mathcal{Q} = 2^{\mathcal{P}}$.*

BGP queries are important as they present a large subset of SPARQL. Previous work also focussed on such queries.

Definition 7 (Query Binding). *The bindings of a query $Q \in \mathcal{Q}$ on an RDF graph $G \in \mathcal{G}$ consisting of the triples available at a set $I \subset \mathcal{I}_\mathcal{I}$, denoted as $\mathsf{bindings}\colon \mathcal{Q} \times 2^{\mathcal{I}_\mathcal{I}} \mapsto 2^{\mathcal{M}}$, is the set of minimal variable bindings which map Q to a subgraph of G: $\mathsf{bindings}(Q, I) = \{\mu \in \mathcal{M} \mid \mathsf{dom}(\mu) = \mathsf{vars}(Q) \wedge \forall p \in Q.\mu(p) \in \cup_{u \in I} \mathsf{deref}(u)\}$.*

4 Authoritative Documents

We introduce the notion of authoritative document for an identifier, that is, we define which information resource can talk authoritatively about a specific identifier. In other words, we restrict the documents which can make statements containing certain identifiers. Our notion is an extension and generalisation of the idea of authoritative source from (Hogan, Harth, and Polleres 2009).

The notion of authoritativeness is important on the web, which consists of a motley collection of data sources, some of which may provide questionable information. Also, we use authoritativeness to specify which information resources are necessary to have complete information about an identifier.

Definition 8 (Authoritative Document). *Document u talks with authority about a triple t if there is a correspondence between u and any identifier from t, i.e., $\mathsf{co}(s(t)) = u$, $co(p(t)) = u$ or $co(o(t)) = u$. We call a document u to be subject-authoritative for t if $\mathsf{co}(s(t)) = u$ (s-auth in short). Analogously, p-auth and o-auth relate a document to the identifier of a predicate or object.*

Example 2. *Consider the triples and documents from Figure 1. Document `a` talks with authority about triples 1-3, namely s-auth for 1-3 and o-auth for triples 1 and 2. Document `e` contains triple 8 using identifiers (`d:i`, `p:i`, `c:i`) without authority, as there is no connection in* co *between any of the identifiers and `e`.*

Definition 9 (Authority Types). *We can have atomtic authority types s, p, or o denoting whether a triple has been stated with authority regarding its subject, predicate or object. We can combine atomic authority types using conjunction and disjunction to arrive at the set of possible authority types $\mathcal{A} = \{\bot, s, p, o, s \vee p, s \vee o, p \vee o, s \vee p \vee o, s \wedge p, s \wedge o, p \wedge o, s \wedge p \wedge o\}$. Note that $\bot$ denotes no authority.*

Example 3. *In the following, we explain two examplaric authority types:*

- *A triple t is stated $s \wedge o$-auth, if both $t \in \mathsf{deref}(\mathsf{co}(s(t)))$ and $t \in \mathsf{deref}(\mathsf{co}(o(t)))$.*
- *A triple t is stated $s \vee p \vee o$-auth, if $t \in \mathsf{deref}(\mathsf{co}(s(t)))$ or $t \in \mathsf{deref}(\mathsf{co}(p(t)))$ or $t \in \mathsf{deref}(\mathsf{co}(o(t)))$.*

Based on the notion of authority types we introduce a modified deref function, the derefa function, which only selects triples that satisfy specified authority types.

Definition 10 (Authoritative Lookup). *The function $\mathsf{derefa}\colon \mathcal{I}_\mathcal{I} \times \mathcal{A} \mapsto \mathcal{G}$ models a Linked Data lookup and returns the graph represented in an information resource, while applying the specified authority types, i.e., filtering the triples which do not adhere to the authority criteria. Please note that* derefa *might perform additional lookups if those are required for clarifying the authoritativeness of a triple.*

Example 4. *The function $\mathsf{derefa}(b, s \wedge o)$ involves $\mathsf{deref}(b)$, yielding the triple (`b:i` , `p:i` , `d:i`) and subsequently requiring also a $\mathsf{deref}(\mathsf{co}(d\texttt{:}i))$ to verify that the triple also occurs in d.*

In case $a = \bot$ the results for deref and derefa coincide.

The different authority types specify the documents that can contribute certain triples to query results, thus paving the way towards defining completeness.

5 Authoritative Documents for Triple Patterns

We now show which documents are relevant to a triple pattern p under a specified authority type a. If we assure that these relevant documents are dereferenced with the derefa function, we can state that p has been completely answered under a. Based on complete answers to single triple patterns, we define complete answers to Basic Graph Pattern queries in Section 6.

Consider a triple pattern p for which we want to get bindings. In Linked Data query evaluation, the query processor has to dereference (lookup) IRIs which yields data, which in turn is matched with the triple pattern to ultimately yield bindings.

Thus, to get all possible bindings on the web, we would need to get all $\mathcal{I}_\mathcal{I}$ and match the resulting graphs to the triple pattern p. However, based on the notion of authoritative source, we can answer a triple pattern p completely, given a defined authority type.

Example 5. *Consider the triple pattern $p_1 = (\texttt{a:i}, \texttt{p:i}, \texttt{?x})$. If we restrict the answers to be derived from s-auth triples, we are sure to get all those triple if we perform a lookup on `a:i`, that is, $\mathsf{derefa}(\mathsf{co}(\texttt{a:i})), s)$. Thus, we have answered p_1 completely under s-auth assumption.*

We now use the definition of $\mathcal{A}$ to derive, given a triple pattern and authority specification, the subset of $\mathcal{I}$ we have to dereference to find the complete set of bindings for the pattern.

Definition 11 (Completely Sufficient Documents). *We define* $\mathsf{csuff}\colon \mathcal{P} \times \mathcal{A} \mapsto 2^{2^{\mathcal{I}\times\mathcal{V}}}$*, which, given a pattern and an authority type, returns a set of alternative documents sets, each of which is sufficient to completely answer the triple pattern.*

The use of variables becomes clear in Section 6.

$$\mathsf{csuff}(t,a) = \begin{cases} \{\{\mathcal{I}_\mathcal{I}\}, & \text{if } a = \bot \\ \{\{\mathsf{s}(t)\}\}, & \text{if } a = s \\ \{\{\mathsf{p}(t)\}\}, & \text{if } a = p \\ \{\{\mathsf{o}(t)\}\}, & \text{if } a = o \\ \{\{\mathsf{s}(t)\},\{\mathsf{p}(t)\}\}, & \text{if } a = s \wedge p \\ \{\{\mathsf{s}(t)\},\{\mathsf{o}(t)\}\}, & \text{if } a = s \wedge o \\ \{\{\mathsf{p}(t)\},\{\mathsf{o}(t)\}\}, & \text{if } a = p \wedge o \\ \{\{\mathsf{s}(t)\},\{\mathsf{p}(t)\},\{p(o)\}\}, & \text{if } a = s \wedge p \wedge o \\ \{\{\mathsf{s}(t),\mathsf{p}(t)\}\}, & \text{if } a = s \vee p \\ \{\{\mathsf{s}(t),\mathsf{o}(t)\}\}, & \text{if } a = s \vee o \\ \{\{\mathsf{p}(t),\mathsf{o}(t)\}\}, & \text{if } a = p \vee o \\ \{\{\mathsf{s}(t),\mathsf{p}(t),\mathsf{o}(t)\}\}, & \text{if } a = s \vee p \vee o \end{cases}$$

Note that when no authority type is given ($a = \bot$), we would need to retrieve the set of all documents to arrive at complete answers. There can be several alternatives that are sufficient for completely answering a pattern (see $s \wedge o$ authority), and each alternative can require more than one position (see $s \vee o$ authority).

If we know that a triple t exists in the documents $\mathcal{I}_\mathcal{I}$ under an authority type a, we can infer that t exists in the corresponding document of one IRI of each alternatively sufficient IRI set (denoted as L for aLternative):

$$\begin{aligned} & t \in \mathsf{derefa}(\mathcal{I}_\mathcal{I}, a) \\ \rightarrow & \forall L \in \mathsf{csuff}(t,a).\exists l \in L.t \in \mathsf{derefa}(\mathsf{co}(l), a). \end{aligned}$$

The fact that the triple must be contained in all alternatives may not sound intuitive at first, but every alternative is sufficient to determine whether the triple exists.

Example 6. *Consider the triple pattern* $p_1 = (\texttt{a:i}, \texttt{p:i}, \texttt{?x})$*. We illustrate the complete answers for* p_1 *under different authority types:*

- *s-auth:* $\mathsf{csuff}(p_1, s) = \{\{\texttt{a:i}\}\}$*, so there is only one alternative for completely answering* p_1 *by finding all bindings* $\mu \in \mathcal{M}$*, such that* $\mu(p_1) \in \mathsf{derefa}(\texttt{a:i}, s)$*, which would result in the bindings* $\mu_1 = \{\texttt{?x} \mapsto \texttt{a:i}\}$*, and* $\mu_2 = \{\texttt{?x} \mapsto \texttt{b:i}\}$*.*
- $s \wedge p$*-auth:* $\mathsf{csuff}(p_1, s \wedge p) = \{\{\texttt{a:i}\},\{\texttt{p:i}\}\}$*, so it would be sufficient to retrieve either the graph* $\mathsf{derefa}(\texttt{a:i}, s \wedge p)$ *or the graph* $\mathsf{derefa}(\texttt{p:i}, s \wedge p)$ *to find all bindings for* p_1*. However, both graphs are empty, as there is no triple in* $\mathsf{co}(\texttt{p:i}) = \texttt{p}$ *and thus none of the triples in* $\mathsf{co}(\texttt{a:i}) = \texttt{a}$ *is "confirmed", as required by* $s \wedge p$ *authority. Please note that the invocation of* derefa *may involve additional lookups to ensure that triples adhere to a given authority type. These additional lookups only invalidate existing results but never contribute new ones.*
- $s \vee o$*-auth:* $\mathsf{csuff}(p_1, s \vee o) = \{\{\texttt{a:i}, \texttt{?x}\}\}$*, so we cannot answer* p_1 *completely, because there is only one alternative, which would require a binding for* ?x*.*

One complication arises when all alternatives returned by csuff contain variables instead of IRIs. In this case, the pattern cannot be completely answered under the authority scheme. However, if we have conjunctions of several triple patterns, another pattern may be used to find complete bindings for the variables in a sufficient alternative, thus making the conjunction completely answerable. We define completeness for such conjunctions in the next section.

6 Completeness of Basic Graph Patterns

In the following, we address the problem of answering queries consisting of several patterns (so-called Basic Graph Patterns). A query Q consisting of several patterns $tp_0, tp_1, \ldots tp_n$ can be completely answered if the corresponding required positions of a triple pattern are bound either by a constant or by a variable in another completely answerable pattern in the query. We thus define a mapping for assigning a required authority to every pattern in a query.

Definition 12 (Authority Mapping). *We define a mapping* $\alpha\colon Q \mapsto \mathcal{A}$ *that assigns triple patterns in* Q *to different authority types. The set of all such mappings is denoted as* $\mathcal{AU}$*.*

Definition 13 (Authoritative Query Bindings). *We extend the* $\mathsf{bindings}\colon \mathcal{Q} \times 2^\mathcal{I} \mapsto 2^\mathcal{M}$ *function to return only bindings satisfying an authority mapping* α*:* $\mathsf{bindings}^\alpha(Q, I) = \{\mu \in \mathcal{M} \mid \mathsf{dom}(\mu) = \mathsf{vars}(Q) \wedge \forall p \in Q.\mu(p) \in \cup_{u\in I}\mathsf{derefa}(u, \alpha(p))\}$.

We define completeness via a set s of documents that have to be retrieved to completely answer a query Q, i.e., a set s is complete for an authority mapping α, if $\mathsf{bindings}^\alpha(Q, s)$ contains all desired query results. A natural requirement for such a set s of documents is that it holds the same results for Q as the entire Linked Data web, i.e. $\mathsf{bindings}^\alpha(Q, s) = \mathsf{bindings}^\alpha(Q, \mathcal{I}_\mathcal{I})$.

As it is infeasible to materialise the entire Linked Data web, i.e., $\mathsf{deref}(\mathcal{I}_\mathcal{I})$, and thus instead we are searching for a subset $s \subset \mathcal{I}_\mathcal{I}$, where $|s| \ll |\mathcal{I}_\mathcal{I}|$, which can be accessed at query time and so that $\mathsf{deref}(s)$ contains sufficient information to answer the query Q.

Thus, we define that a set s of documents is complete for query Q given an authority mapping α, if $\mathsf{complete}(Q, \alpha) \subseteq s$, where complete is one of the different completeness classes introduced in the following:

- web-complete $\mathsf{wc}\colon \mathcal{Q} \times \mathcal{AU} \mapsto 2^{\mathcal{I}_\mathcal{I}}$ which is mainly of theoretical interest when considering the web, but possibly applicable to controlled environments such as intranets;
- seed-complete $\mathsf{sc}\colon \mathcal{Q} \times \mathcal{AU} \mapsto 2^{\mathcal{I}_\mathcal{I}}$ which is practical and pragmatic solution, if no authority restrictions are given;

- query-reachable-complete $\mathsf{qrc}\colon \mathcal{Q} \times \mathcal{AU} \mapsto 2^{\mathcal{I}_\mathcal{I}}$ which defines complete results under given authority types for a certain class of queries.

We now formally define the three different completeness classes and then discuss the relationships between the different notions in Section 7.

6.1 Web-complete Set

The web-complete set gives the results of the query, when it is evaluated over the whole Linked Data web, i.e. $\mathcal{I}_\mathcal{I}$. However it is sufficient to evaluate over every document that helps to produce a result binding, (could also be a duplicate of a binding that can be produced without it). Without authority restrictions, every document can contain arbitrary triples, thus there is no other way of determining the set than accessing every $u \in \mathcal{I}_\mathcal{I}$ or having some form of index structure, which has accessed every such u before.

Definition 14. *With authority restriction, we define web-complete as the set of documents that contain a triple which is part of a result when evaluating Q over $\mathcal{I}_\mathcal{I}$.*

$$\mathsf{wc}(Q,\alpha) = \{u \in \mathcal{I}_\mathcal{I} \mid \exists \mu \in \mathsf{bindings}^\alpha(Q,\mathcal{I}_\mathcal{I}). \exists p \in Q.\ \mu(p) \in \mathsf{derefa}(u,\alpha(p))\}.$$

Example 7. *Considering our example of query $\mathrm{Q_{ex}}$ and assuming two authority mappings α_1 and α_2 we get:*

- *Let $\alpha_1(p) = \bot$, for $p \in \mathrm{Q_{ex}}$:* $\mathsf{wc}(\mathrm{Q_{ex}},\alpha_1) = \{a,b,c,d,e\}$.
- *Let $\alpha_2(p) = s$, for $p \in \mathrm{Q_{ex}}$:* $\mathsf{wc}(\mathrm{Q_{ex}},\alpha_2) = \{a,b\}$.

6.2 Seed-complete Set

The seed-complete set consists of all documents that can be reached via following triple paths of maximum length of the query beginning from triples in the documents identified by the IRIs in the query. The intuition is a traversal of $\mathcal{I}_\mathcal{I}$ to get the documents that are up to n hops away.

In the size-restricted seed-complete set, we fix n to the query size $|Q|$. In an alternative, length-restricted seed-complete set (which we leave open for future work), we can fix n to the depth of the query, i.e., the length of longest path in the query, starting from a constant.

As there can be several different IRIs in the query and from each IRI there can start several paths of triples, we possibly end up with forests, consisting of several trees starting in different triples.

Definition 15 (Forest). *A triple forest grounded in a set of seed IRIs is a list of triples, where each triple is either in the seed IRIs, or in the corresponding document of a resource occuring in a previous triple in the list. The function* $\mathsf{forests}\colon 2^{\mathcal{I}_\mathcal{I}} \times \mathbb{N} \mapsto 2^{\mathcal{T}^*}$ *returns all forests of triples of size up to n, starting with the triples in the seed IRIs $s_0 = \mathsf{co}(\mathsf{iris}(Q))$:*

$$\mathsf{forests}(s_0,n) = \bigcup_{j\in[1..n]} \{(t_1,\ldots,t_j) \in \mathcal{T}^j \mid \forall i \in [1..j].t_i \in \cup_{u\in s_0}\mathsf{deref}(u) \vee \exists k \in [1..i-1].t_i \in \cup_{u\in\mathsf{iris}(t_k)}\mathsf{deref}(u)\}$$

Definition 16. *We define the seed completeness set to contain all documents corresponding to IRIs in the forests grounded in the query's IRIs:*

$$\mathsf{sc}(Q,\alpha) = \mathsf{co}(\mathsf{iris}(\mathsf{forests}(\mathsf{iris}(Q),|Q|))).$$

Example 8. *Considering our example of query $\mathrm{Q_{ex}}$, we get for $\mathsf{iris}(\mathrm{Q_{ex}}) = \{\texttt{a:i}, \texttt{p:i}\}$, and $|\mathrm{Q_{ex}}| = 2$:*

$$\begin{aligned}\mathsf{forests}(\mathsf{iris}(\mathrm{Q_{ex}}),|\mathrm{Q_{ex}}|) = \{&(t_1),(t_1,t_1),(t_1,t_2),(t_1,t_3),\\ &(t_2),(t_2,t_1),(t_2,t_2),(t_2,t_3),\\ &(t_3),(t_3,t_1),(t_3,t_2),(t_3,t_3),\\ &(t_3,t_4)\}.\end{aligned}$$
$$\mathsf{sc}(\mathrm{Q_{ex}},\alpha) = \{a,p,b,d\},$$

where t_i stands for triple number i from the runnnig example (see Section 2).

6.3 Query-reachable-complete Set

We first define the notion of completely answerable queries for a given authority mapping α. Then, we specify the set of documents required to answer such a query completely in the sense of obtaining the same results as if the query would be evaluated over the web-complete set. The equivalence of the result sets is shown in Section 7.

Definition 17 (Completely-answerable Query). *A query is completely answerable if the triple patterns can be brought into an order, such that for each triple pattern p, there exists a set of RDF terms sufficient to completely answer p, where each term is either an IRI or a variable occuring in a previous pattern. The predicate caq^α defines the completely answerable property of a query under an authority mapping α:*

$$\begin{aligned}&\mathsf{caq}^\alpha(Q) \leftrightarrow\\ &(Q = \{p\} \wedge \exists L \in \mathsf{csuff}(p,\alpha(p)).\forall l \in L.l \in \mathcal{I}_\mathcal{N}) \vee\\ &(|Q| > 1 \wedge \exists Q_n, Q_1.Q_n \cup Q_1 = Q \wedge Q_n \cap Q_1 = \emptyset \wedge\\ &\qquad \mathsf{caq}^\alpha(Q_n) \wedge Q_1 = \{p\} \wedge\\ &\qquad \exists L \in \mathsf{csuff}(p,\alpha(p)).\forall l \in L.l \in \mathcal{I} \vee l \in \mathsf{vars}(Q_n)).\end{aligned}$$

In other (recursive) words: a query Q is completely answerable if either Q is of size 1 and there exists a set of sufficient terms which are all IRIs in Q, or one can remove a pattern p from the query, such that the resulting query Q_n is completely answerable, and p has a set of required terms which are either IRIs or variables bound by query Q_n.

An IRI must be in the query-reachable-complete set if it occurs in a forest, starting in the IRIs of the query, which is a result for a completely answerable subquery of the original query.

Definition 18 (Completely Answerable Subqueries). *The function* $\mathsf{csq}^\alpha\colon \mathcal{Q} \mapsto 2^\mathcal{Q}$ *returns all completely-answerable subqueries of a query:*

$$\mathsf{csq}^\alpha(Q) = \{Q' \subseteq Q \mid \mathsf{caq}^\alpha(Q')\}.$$

The forests, which are results for a completely answerable subquery of Q are defined by the function $\mathsf{qforests}^\alpha$ as

a subset of all forests starting in the IRIs contained in the query.

$$\mathsf{qforests}^\alpha(Q) = \{F \in \mathsf{forests}(\mathsf{co}(\mathsf{iris}(Q)), |Q|) \mid \exists Q' \in \mathsf{csq}^\alpha(Q) \wedge \exists \mu \in \mathcal{M}.\mu(Q') = F\}$$

Definition 19. *We define the query-reachable-complete set, to contain all documents corresponding to IRIs in the forests that produce bindings for a completely answerable subquery of Q:*

$$\mathsf{qrc}(Q, \alpha) = \mathsf{co}(\mathsf{iris}(\mathsf{qforests}^\alpha(Q))).$$

Example 9. *Considering our example of query $\mathrm{Q_{ex}}$, we get for α, where $\alpha(p) = s$, for all $p \in \mathrm{Q_{ex}}$:*

$$\begin{aligned}\mathsf{csq}^\alpha(\mathrm{Q_{ex}}) &= \{\mathrm{Q_{ex}}, \{(\texttt{a:i}, \texttt{p:i}, \texttt{?x})\}\}\\ \mathsf{qforests}^\alpha(\mathrm{Q_{ex}}) &= \{(t_1), (t_1, t_2), (t_3), (t_3, t_4)\}\\ \mathsf{qrc}(\mathrm{Q_{ex}}, \alpha) &= \{\mathtt{a}, \mathtt{p}, \mathtt{b}, \mathtt{d}\},\end{aligned}$$

where t_i stands for triple number i from the runnnig example (see Section 2). We can see that $\mathsf{wc}(\mathrm{Q_{ex}}, \alpha) \subset \mathsf{qrc}(\mathrm{Q_{ex}}, \alpha)$, meaning that the query reachable set produces all bindings available in the web under the authority mapping α. In Section 7 we show this in general for all completely answerable queries.

7 Relations Between Completeness Classes

In the following, we show the relation between the different completeness classes.

Theorem 1. *QRC results are a subset of (size-restriced) SC results:* $\mathsf{qrc}(Q, \alpha) \subseteq \mathsf{sc}(Q, \alpha)$.

Proof. Theorem 1 is obvious from the definition, as $\mathsf{qforests}^\alpha(Q) \subseteq \mathsf{forests}(\mathsf{co}(\mathsf{iris}(Q), |Q|))$. □

Theorem 2. *SC query results are a subset of WC results:* $\mathsf{bindings}^\alpha(Q, \mathsf{sc}(Q, \alpha)) \subseteq \mathsf{bindings}^\alpha(Q, \mathsf{wc}(Q, \alpha))$.

Proof. Theorem 2 is obvious from the definition, as web complete is defined to contain all bindings, and thus seed complete can not contain more bindings. □

Theorem 3. *For a query Q that is completely answerable under an authority mapping α, the bindings for query reachable complete and web complete coincide:* $\mathsf{bindings}^\alpha(Q, \mathsf{qrc}(Q, \alpha)) = \mathsf{bindings}^\alpha(Q, \mathsf{wc}(Q, \alpha))$.

Proof. We prove the equivalence of the sets, by showing their mutual containment:
(1) $\mathsf{bindings}^\alpha(Q, \mathsf{qrc}(Q, \alpha)) \subseteq \mathsf{bindings}^\alpha(Q, \mathsf{wc}(Q, \alpha))$ follows from the definition, that web completeness means that every (in this case α-authoritative) result is found.
(2) $\mathsf{bindings}^\alpha(Q, \mathsf{wc}(Q, \alpha)) \subseteq \mathsf{bindings}^\alpha(Q, \mathsf{qrc}(Q, \alpha))$ is shown by induction on the query size. As bindings is monotonic over the set of documents, we reduce this case to showing that $\mathsf{wc}(Q, \alpha) \subseteq \mathsf{qrc}(Q, \alpha)$, if $\mathsf{caq}^\alpha(Q)$.
Induction start for a query Q of size 1:
from $\mathsf{caq}^\alpha(Q) \wedge |Q| = 1 \wedge u \in \mathsf{wc}(Q, \alpha)$ follows that there exists a binding μ for Q over $\mathcal{I}_\mathcal{I}$, which maps the single triple pattern $p \in Q$ to a triple from u: $\exists \mu \in \mathsf{bindings}^\alpha(Q, \mathcal{I}_\mathcal{I}).\mu(p) \in \mathsf{derefa}(u, \alpha(p))$. This implies that $u \in \mathsf{co}(\mathsf{iris}(\mu(p)))$, for $\alpha(p) \neq \bot$, which is ruled out by the definition of caq. Furthermore, we know that $(\mu(p)) \in \mathsf{forests}(\mathsf{co}(\mathsf{iris}(Q)), 1)$, as $\mu(p)$ is a query answer to $Q = \{p\}$, and p must be completely answerable, given that $\mathsf{caq}^\alpha(Q)$, it follows, that $(\mu(p)) \in \mathsf{qforests}^\alpha(Q)$ and thus: $u \in \mathsf{qrc}(Q, \alpha)$.
We form the induction hypothesis:

$$\mathsf{caq}^\alpha(Q) \wedge u \in \mathsf{wc}(Q, \alpha) \wedge |Q| = n \rightarrow u \in \mathsf{qrc}(Q, \alpha).$$

The inductive step: given $\mathsf{caq}^\alpha(Q) \wedge u \in \mathsf{wc}(Q, \alpha) \wedge |Q| = n + 1$, we can split Q into Q_n and Q_1, such that $Q = Q_n \cup Q_n \wedge Q_n \cap Q_1 = \emptyset \wedge |Q_1| = 1 \wedge \mathsf{caq}^\alpha(Q_n)$ (follows from $\mathsf{caq}^\alpha(Q)$). Accordingly our argument can be split into two cases:
Case (2.1): u is also in the web complete set of Q_n: $\mathsf{caq}^\alpha(Q_n) \wedge |Q_n| = n \wedge u \in \mathsf{wc}(Q_n, \alpha)$. We can use the induction hypothesis and conclude $u \in \mathsf{qrc}(Q_n, \alpha)$ and because the reachable completeness set is monotonic (a larger query still has the smaller query as a subquery), we conclude: $u \in \mathsf{qrc}(Q, \alpha)$.
Case (2.2): u is not in the web complete set of Q_n, thus it must be contributed by a variable binding or a constant in Q_1. Evaluating Q_1 has to be done only for the bindings of Q_n, other results that do not join with the results for Q_n cannot contribute an result for Q. As $\mathsf{caq}^\alpha(Q)$, we know that there exists a set of terms sufficient for completely answering Q_1, in which all terms are either constants or variables already occuring in Q_n. Therefore, we can reduce the case to considering only those $\mu(Q_1)$, where μ is a result binding for Q_n. Thus, $\mu(Q_1)$ is completely answerable, and we can use the induction start:

$$\mathsf{caq}^\alpha(\mu(Q_1)) \wedge |Q_1| = 1 \rightarrow u \in \mathsf{qrc}(Q_1, \alpha) \rightarrow u \in \mathsf{qrc}(Q, \alpha).$$

□

8 Related Work

Early work on queries over the web graph include (Mendelzon and Milo 1997) and (Abiteboul and Vianu 2000). (Hartig, Bizer, and Freytag 2009) introduced Linked Data query processing via link traversal. Subsequent work (Hartig, Bizer, and Freytag 2009; Harth et al. 2010; Ladwig and Tran 2010; Haase, Mathäß, and Ziller 2010; Umbrich, Hogan, and Polleres 2011) lack a rigorous specification of termination criteria (some use heuristics). (Fionda, Gutierrez, and Pirrò 2011) introduce a navigation language, which has different characteristics than our query language based on BGPs. (Hartig 2012) analyses the computability of SPARQL queries over Linked Data under different semantics. The notion of semantics in (Hartig 2012) roughly corresponds to our notion of completeness: the full-web semantics is similar to our web-completeness, whereas the reachability-based semantics can be considered as an abstract concept, while we provide two actual completeness classes. While (Hartig 2012) shows that in the general case

the full-web semantics is not computable, we show that under certain authority constraints it is possible to achieve results equivalent to web-completeness.

Several other papers propose definitions for semantics in distributed settings. The Local Relational Model (Serafini et al. 2003) uses model-theoretic means to specify the semantics of a federation of relational databases. Context OWL (Bouquet et al. 2003) considers description logic inference and connects models via bridge rules, whereas our definitions are on the RDF graph level with shared use of identifiers. Our definitions have an operational aspect and are tied to a view that assumes only local knowledge, in contrast to the global view taken by typical model-theoric approaches. Finally, (Polleres, Feier, and Harth 2006) use a form of a local closed world model to specify the semantics of queries with negation as failure under a modified OWA.

9 Conclusion

We have provided a general notion of authoritativeness and defined three completeness classes for Linked Data query evaluation. While the seed-complete class is straightforward to implement, the query-reachable-complete class requires less lookups while yielding a well-defined and useful set of results, based on a specified authority assignment for each query. However, implementing query-reachable is more intricate.

Future work includes extending the authority types with negation, thus allowing for a negation-as-failure semantics, an algorithm for enumerating possible authority assignments for queries, and an investigation of query result completeness when allowing sources with query capabilities.

Acknowledgements

We thank Sebastian Rudolph for insightful comments, and acknowledge the support of the European Commission's Seventh Framework Programme FP7/2007-2013 (PlanetData, Grant 257641).

References

Abiteboul, S., and Vianu, V. 2000. Queries and computation on the web. *Theoretical Computer Science* 239:231–255.

Berners-Lee, T. 2006. Linked Data. http://www.w3.org/DesignIssues/LinkedData.

Bouquet, P.; Giunchiglia, F.; van Harmelen, F.; Serafini, L.; and Stuckenschmidt, H. 2003. C-OWL: Contextualizing ontologies. In *Second International Semantic Web Conference*, number 2870 in Lecture Notes in Computer Science, 164–179. Springer.

Fionda, V.; Gutierrez, C.; and Pirrò, G. 2011. Semantic navigation on the web of data: Specification of routes, web fragments and actions. *CoRR* abs/1111.4316.

Franklin, M.; Halevy, A.; and Maier, D. 2005. From databases to dataspaces: a new abstraction for information management. *SIGMOD Rec.* 34:27–33.

Haase, P.; Mathäß, T.; and Ziller, M. 2010. An evaluation of approaches to federated query processing over linked data. In *6th International Conference on Semantic Systems*, I-SEMANTICS 2010. ACM.

Harth, A.; Hose, K.; Karnstedt, M.; Polleres, A.; Sattler, K.-U.; and Umbrich, J. 2010. Data summaries for on-demand queries over linked data. In *Proceedings of the 19th International Conference on World Wide Web*, 411–420. ACM.

Hartig, O.; Bizer, C.; and Freytag, J.-C. 2009. Executing sparql queries over the web of linked data. In *Eight International Semantic Web Conference*, volume 5823 of *Lecture Notes in Computer Science*, 293–309. Springer.

Hartig, O. 2012. Sparql for a web of linked data: Semantics and computability (extended version). *CoRR* abs/1203.1569.

Hogan, A.; Harth, A.; and Polleres, A. 2009. Scalable authoritative owl reasoning for the web. *International Journal on Semantic Web Information Systems* 5(2):49–90.

Ladwig, G., and Tran, T. 2010. Linked data query processing strategies. In *Ninth International Semantic Web Conference*, volume 6496 of *Lecture Notes in Computer Science*. Springer. 453–469.

Mendelzon, A. O., and Milo, T. 1997. Formal models of web queries. In *Sixteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, PODS '97, 134–143. ACM.

Pérez, J.; Arenas, M.; and Gutierrez, C. 2009. Semantics and complexity of sparql. *ACM Transactions on Database Systems* 34:16:1–16:45.

Polleres, A.; Feier, C.; and Harth, A. 2006. Rules with contextually scoped negation. In *Third European Semantic Web Conference*, number 4011 in Lecture Notes in Computer Science, 332–347. Springer.

Serafini, L.; Giunchiglia, F.; Mylopoulos, J.; and Bernstein, P. 2003. Local relational model: a logical formalization of database coordination. In *Proceedings of the 4th International and Interdisciplinary Conference on Modeling and Using Context*, 286–299. Springer.

Umbrich, J.; Hogan, A.; and Polleres, A. 2011. Improving the recall of decentralised linked data querying through implicit knowledge. *CoRR* abs/1109.0181.