A logical characterisation of SPARQL federation

Editor(s): Name Surname, University, Country
Solicited review(s): Name Surname, University, Country
Open review(s): Name Surname, University, Country

Audun Stolpe,
Norwegian Defence Research Establishment (FFI), Postboks 25, 2027 Kjeller, Norway
E-mail: audun.stolpe@ffi.no

Abstract. The paper provides a logical characterisation of distributed processing of SPARQL conjunctive queries. The principal notion analysed is that of a distribution scheme, which is a pair consisting of an evaluation rule and a distribution function. Different choices of evaluation rules and distribution functions give different federation schemes that are proved to be sound and complete wrt. different sets of clusters. A cluster is here understood as a finite set of RDF sources. Three distribution functions are singled out for particular attention, two of which are implicit in several examples from the literature. These are named the even-, standard- and prudent distribution and yield sound and complete federation schemes when paired with an evaluation rule named the collect-and-combine rule. These completeness results are next compared with two systems that have been proposed in the literature. They indicate that the level of comparison facilitated by a formal framework can be useful. In particular, certain sources of incompleteness turn out to recur in the literature without being afforded much attention as such. Finally, the concept of a distribution scheme is related to the heuristic notion of a join-ordering to yield sound and complete execution plans for the aforementioned federation schemes.

Keywords: SPARQL federation, distributed query processing, logical foundation, completeness, distribution scheme

1. Introduction

That the exploitation and dissemination of Semantic Web data requires powerful federation engines, is a claim that hardly needs an argument. Unedited and decentralized, the Semantic Web is also a singularly fluid computational environment. Therefore any federation engine that aims to be generic must be capable of operating in a dynamic network topology where sources may be discovered at run time, and little information about the structure and content of the sources is in general to be had.

SPARQL federation—the decomposition and distribution of SPARQL queries—is one approach that seems both principled and promising. Its emerging importance is mirrored by the number of publications devoted to this topic in the last five to ten years. Yet, attempts to formulate a theoretical framework in which to explore and compare different approaches mathematically are largely absent. This remains true even particular query languages are abstracted away so as to include the comparatively rich literature on distributed databases. Thus, whilst the 150+ references in Kossman's excellent "The State of the Art in Distributed Query Processing" [19] together with the 150+ references in Hose et al.'s "Database foundations for scalable RDF processing" [16] span more than 30 years of research, none of the listed references gives a general logical characterisation of distributed query answering.

The present paper provides some tentative results that indicate how the working out of such a framework may be both an interesting and worthwhile pursuit. Although the concepts introduced ought to be of more general validity, focus will be placed on federation in a zero-knowledge environment, that is, on federation performed in the absence of specific information about the structure and content of contributing sources (apart from their signatures—a concept to be defined). Also, only the fragment of SPARQL that consists of simple conjunctive queries aka. basic graph patterns will be considered. The operative assumption is that these restrictions form a natural base-line for a theory, from which more complex cases can be derived.

The present paper does not advocate ‘an approach’ to SPARQL federation. Rather, it is more aptly seen as giving formal expression to properties and observations that are already part of the ‘folklore’ of the Semantic Web community. None of these proofs are particularly deep or difficult. It is hoped that they will be perceived as natural and useful.

After a review of related work in section 2, section 3 recalls a few essential notions of SPARQL semantics, fixes notation and declares this paper’s theoretical assumptions. Section 4 introduces the principal unit of analysis in this paper, which is that of a *federation* scheme understood as a pair of a *distribution function* and an *evaluation* rule. Federation schemes will be said to be sound and/or complete wrt. sets of *clusters*. A set of *clusters* is a finite set of RDF graphs which will serve as the semantical relatum with respect to which soundness and completeness is measured. Section 5 applies federation schemes and clusters to to the analysis of federation in a zero-knowledge environment. The set of all clusters—which is the most unconstrained set of permissible constellations of contributing sources—is pushed in to service as a model for the case where no assumptions are made about the content and structure of the contributing sources. A number of federation schemes are then defined that are provably sound and complete with respect to it. These schemes are all obtained by combining a particularly natural evaluation rule, called the collect-and-combine-rule, with three different distribution functions, called the *even-*, *standard-* and *prudent distribution* respectively. Section 6 consists of two case studies of systems that have been proposed in the literature. The purpose is to show that the framework developed in the present paper is a natural and useful idiom in which to state the properties one intends such a system to have, and to explore the ramifications of these assumption. Finally, since no federation engine can be expected to perform well without a carefully crafted optimizer, the paper is rounded off in section 7 showing how federation schemes fit together with join-orderings. This section can be considered as a corollary to the preceding ones.

2. Related work

Fairly recent surveys of the state of the art wrt. RDF federation can be found in Betz et al. [5], Görlitz and Staab [10] and Hose et al. [16]. Based on these sources, it is possible to distinguish between at least three major trends in RDF federation, namely lookup-based federation, warehousing and distributed query processing. Lookup-based federation, which may also be called federation-by-traversal, iteratively downloads and evaluates data based on links (usually as prescribed by the Linked Data principles [7]) from one dataset into another. The answer to a query is composed by cumulatively adding answers from incoming data during the traversal process. This approach is exemplified by e.g. [15,14] and [20]. Approaches based on warehousing, on the other hand, collect all data in advance and combines it into a central triple store against which queries are evaluated. Recent efforts in this direction focus on the use of cluster technology such as MapReduce and hadoop, e.g. [17,8,18] and [25]. Finally, approaches based on distributed query processing rely on analysing a query to identify a set of relevant sources—with or without the aid of statistics—to which subqueries can be assigned. Like federation-by-traversal and unlike warehousing, queries are evaluated directly at the remote sources. Recent example of this approach include [2,26,4,24] and [27].

The present paper is most comfortably subsumed under the latter category, although it ought to be of relevance to the other two as well. It is to the best of our knowledge the first foundational study of distributed query processing that approaches the discipline from a purely logical point of view. There do exist a few formal studies of what can be considered the more general topic of querying the Web, for instance Abiteboul and Vianu [1] and Bouquet et al. [6]. The former is concerned with the computability of queries over the Web expressed in Datalog, and predates the SPARQL query language. The latter describes three different ways in which a query can be answered and characterises their answers semantically in complete abstraction from query languages and distribution algorithms. These references have little in common with the present study, apart from being formal and about queries on the Semantic Web.

3. Preliminaries

Let U, L and B denote pairwise disjoint sets of URIs, literals and blank nodes respectively and denote by $\mathcal{T}$ the set of RDF terms $U \cup L \cup B$. An RDF *triple* is an element of $\mathcal{T} \times U \times \mathcal{T}$ and an RDF *graph* is a finite set of triples. RDF graphs are denoted R_i, but the index and the index set will be omitted when idle. RDF graphs will also be referred to as sources, as this terminology is more suggestive in the context of federation.

Let V be a countably infinite set of variables disjoint from $\mathcal{T}$. A triple pattern is an element of $(\mathcal{T} \cup V) \times (U \cup V) \times (\mathcal{T} \cup V)$. Triple patterns will be denoted by lowercase letters from the end of the alphabet, possibly with subscripts $t_i, \dots, t_j$. In order to avoid tedious limiting cases in proofs, it will be assumed that the set of triple patterns is disjoint from the set of RDF triples, that is, a triple pattern contains at least one variable.

In analogy to the common practice in database theory, the distinction between the syntax and semantics of queries will be deliberately blurred by defining a basic graph pattern (henceforth a BGP) as a *set* of triple patterns. A SPARQL `SELECT` query is thus a pair $(P, \vec{x})$, where P (or S or T) is a *set* of triple patterns and $\vec{x}$ a vector of elements of variables in which no element is repeated. The order of elements in $\vec{x}$ is of no consequence for the developments in this paper, whence $\vec{x}$ will not be distinguished from the underlying set x. This licenses talk about 'the' vector of variables of P, which will be denoted $var(P)$. Moreover, since all queries are assumed to be of form $Q := (P, var(P))$, the query Q will frequently be identified with the graph pattern P.

Familiarity with SPARQL semantics is assumed. The reader who would like a recap is referred to [3], from whence much of the formal nomenclature in the present paper derives. Thus, the evaluation of a BGP P over a graph R is written $[\![P]\!]_R$ and contains partial functions $\mu_1, \mu_2, \dots$ from V to U, which may or may not be pairwise compatible in the sense the two maps agree on the value assigned to variables.

Considered from an abstract or logical point of view, the problem of characterising SPARQL federation is essentially that of generalizing the evaluation of BGPs to multiple sources. There are only two ways of doing so that will be utilized in the present paper—the relationship between them will be elaborated in subsequent sections.:

Definition 3.1 *Let $\mathscr{R}$ be any set of sources and P a BGP. Then:*

1. $[\![P]\!]_{\mathscr{R}} =_{df} [\![P]\!]_{\bigcup\mathscr{R}}$
2. $[P]_{\mathscr{R}} =_{df} \bigcup_{R_i \in \mathscr{R}} [\![P]\!]_{R_i}$

Here, $\mathscr{R}$ denotes a cluster—that is, a set of finite sets of RDF graphs—and $\bigcup\mathscr{R}$ degnotes the union of its elements. It is assumed that blank nodes are never shared between sources. This is no real restriction as one can always rename blank nodes without altering the information content of a source.

In order to lighten the notation somewhat, curly braces will be omitted from singletons in set-theoretic expressions and applications of functions is if no confusion is likely to ensue, e.g. $P \cup t$ instead of $P \cup \{t\}$ and $f(t)$ instead of $f(\{t\})$. Also, when f is a function and A is a set, $f(A)$ is the image of f under A, that is, if $f : A \to B$ then $f(A) = \{b \in B : f(a) = b, a \in A\}$.

4. Federation schemes

In general, the problem of answering a query Q over a set of sources $\mathscr{R}$ reduces to two interrelated subproblems: A) how to assign to members of $\mathscr{R}$ the proper subquery of Q, and B) how to combine the answers to these subqueries into an answer to Q itself. Varying either of these parameters may change the final answer. That is, the outcome of the federation process will in general be sensitive to how the contributing sources are selected, how subqueries of the global query are allocated to these sources and how the partial results are subsequently combined.

Therefore any *general* property of distributed query processing must be similarly parametrised, that is, it should be formulated in terms of a *decomposition- or distribution function*, an *evaluation rule* and some notion capturing the permissible ways of selecting contributing sources.

One way of representing the concept of a permissible constellation of sources, is as family of sets of RDF graphs: a single element of the family comes to represents one admissible constellation, whereas the sense in which it is permitted can be expressed by placing requirements on the family of sets as such. Initially one may wish to restrict attention to *finite* sets of RDF graphs, as distributing a query over an infinite set of sources is not a notion that makes a whole lot of sense. Introducing a terminology that is a bit more suggestive than 'set', such constellations will henceforth be referred to as *clusters*.

In order to distribute a (global) query among the elements of a cluster, the query must be decomposed into subqueries and each subquery must be routed to the sources that are relevant for answering it. In this paper, subqueries are assigned to sources by comparing the signature of the subquery with the signature of the sources—this is a common approach in the literature: If P is a BGP the signature of P written $sig(P)$ is $sig(S) =_{df} S^2 \setminus V$, where S^2 denotes the projection of S onto its second coordinates. If the signature of P is not contained by the signature of a source R, then P can not be assigned to R as R can only ever provide an empty set of answers to P. The converse does not in general hold, but depends on the choice of distribution strategy as we shall see.

Given a set of repositories $\mathscr{R}$ and a BGP P let $\mathscr{R}$ do dual service as an *operator* that maps P to the set of sources in $\mathscr{R}$ that are relevant for answering P:

Definition 4.1 *The sources in $\mathscr{R}$ that are relevant for answering P are denoted $\mathscr{R}(P)$ and is defined as*

$$\mathscr{R}(P) =_{df} \{R \in \mathscr{R} : sig(P) \subseteq sig(R)\}$$

if P is non-empty, and as $\mathscr{R}(P) =_{df} \emptyset$ otherwise.

Distribution functions can now be defined as follows:

Definition 4.2 (Distribution function) *A distribution function is a binary function δ that takes a set of sources $\mathscr{R} := \{R_i\}_{i \in I}$ and a BGP P and returns a set of pairs (S, J) such $S \neq \emptyset$, $J \subseteq \mathscr{R}$ and*

1. $\mathscr{R}(S) = J$
2. $\bigcup_{(S,J) \in \delta(\mathscr{R},P)} J = \mathscr{R}$
3. $\bigcup_{(S,J) \in \delta(\mathscr{R},P)} S = P$

Clause (3) of this definition is a necessary condition for soundness: δ distributes P over $\mathscr{R}$ only if P is decomposed into subqueries that are jointly exhaustive of P. Clause (2), on the other hand, requires that each source in $\mathscr{R}$ at least potentially contribute to the result. This is not a real restriction since if there is a source in $R \in \mathscr{R}$ s.t. $sig(R) \cap sig(P) = \emptyset$ one may simply remove R and define the distribution function as operating on the reduced set. Finally, clause (1) prevents a distribution function from behaving erratically when it comes to assigning subqueries to sources. It has two important consequences: first in the presence of (3) a distribution function becomes a partial function, that is, (1) and (3) can not be satisfied for arbitrary choices of $\mathscr{R}$ and P, reflecting the fact that the accumulated signature of the sources may not cover the signature of the global query. In the opposite case δ will be said to be defined *at* $\mathscr{R}$ and P. Secondly, the relation of assigning subqueries to sources becomes a function:

Theorem 4.1 *Let $\mathscr{R}$ be any cluster, P any BGP and δ any distribution function. If δ is defined at $\mathscr{R}$ and P, then $\delta(\mathscr{R}, P)$ is a function.*

Proof Suppose $(S, J), (T, K) \in \delta(\mathscr{R}, P)(S, J)$ and that $S = T$. Then $J = \mathscr{R}(S) = \mathscr{R}(T) = K$, by definition 4.2(1), so $J = K$.

Thus $\delta(\mathscr{R}, P)$ is in fact an indexed set, and the reader should feel free to substitute S_J for (S, J) if he or she prefers it.

In order to present a few concrete examples of distribution functions, let the notion of an *exclusive subset* of P pertaining to a given source be defined as follows:

Definition 4.3 *The subpattern of BGP P that is exclusive to a source R is the set:*

$$R^e(P) =_{df} \{t \in P : \mathscr{R}(t) = \{R\}\}$$

A triple pattern that is not an element of an exclusive group is called a non-exclusive triple.

In this form, this definition goes back to [27].

Definition 4.4 *Let $\mathscr{R}$ be any cluster and P any BGP. Stipulate that for S and J such that $J = \mathscr{R}(S)$. Then the following conditions all define distribution functions:*

1. *Even distribution: $(S, J) \in \delta(\mathscr{R}, P)$ iff S is a singleton $t \in P$*
2. *Standard distribution: $(S, J) \in \delta(\mathscr{R}, P)$ iff either*
   a) *if S is a non-exclusive triple pattern, or*
   b) *$S = R^e_i(P)$ for some $R_i \in \mathscr{R}$*
3. *Prudent distribution: $(S, J) \in \delta(\mathscr{R}, P)$ iff either*
   a) *S is a non-exclusive triple pattern, or*
   b) *S is a join-connected component of $R^e_i(P)$ for some $R_i \in \mathscr{R}$*

Anticipating subsequents developments, these distribution functions are all suitable for the zero-knowledge case insofar as they yield sound and complete federation schemes (wrt. the set of all clusters) when paired with the collect-and-combine evaluation rule.

The standard distribution in particular occurs frequently in the literature (hence the name), although it has proved difficult to find a precise statement of it. For instance, SPLENDID [11], DARQ [26], FedX [27] and HERMES [28] are all examples of federators that decompose queries in a manner which is essentially the standard distribution (although theses systems are not necessarily comparable on other counts).

All three distribution functions can be said to heed the fact that one source may be able to generate new results when joining with *partial* results from another [26, p. 120]. Stated differently, a join may span two or more sources, in which case the evaluation of a non-singleton graph pattern over a given source may come to suppress intermediate results thereby excluding cross-site joins. The even distribution averts this risk by breaking a graph pattern into its singleton constituents, but this may be considered to be overly cautious. The standard distribution, in contrast, carves a query at its natural joints in the sense that whenever the pattern in question is exclusive to a source, then it is assigned to that source as-is. This is safe too, since grouping exclusive patterns cannot incur a loss of information [11,27]. Finally, the prudent distribution, which does not seem to have been mentioned in the literature yet, is halfway between the other two. Like the even distribution it treats non-exclusive triples as singletons, and like the standard distribution it groups together (some) exclusive triples. However, unlike the standard distribution, the prudent distribution splits exclusive groups into maximal join-connected components, each of which is assigned to the source for which it is exclusive.

The relative merits of these three distribution schemes is not really visible unless looked at from the perspective of join-order heuristics—a topic which is otherwise postponed until section 7. If a join-ordering is perceived as a partial order over the power set of a graph pattern P then the even distribution is the finest ordering of P. In computational terms, the even distribution makes each triple pattern in P executable independently of every other, and therefore leaves maximum control to query planning on the side of the federator. The standard distribution, on the other hand, strikes a balance between query optimization on the side of the federator and on the side of the contributing sources. That is, exclusive patterns effectively 'bracket' joins, thus suspending optimization and delegating it to the remote source in question. The prudent distribution, being halfway between the other two, is more discerning than the even distribution, but, one might say, less dogmatic about the heuristic value of exclusive subpatterns than the standard distribution. Specifically, if an exclusive subpattern can be split into several components without severing joins, then the prudent distribution hands responsibility for preferring between these components back to the federator in order, possibly, to avoid requesting cartesian products from the remote source.

Turning now to the notion of an evaluation rule, the second proposed component of federation schemes, it suffices for now to define it in complete abstraction simply as a function that produces a set of answers from a distribution:

Definition 4.5 (Evaluation rule) *Let Δ be any distribution. An evaluation rule is a function $\mathbb{E}$ such that $\mathbb{E}(\Delta)$ is a set of variable mappings $\mu : V \to T$.*

Obviously, this definition hides great variety. Next:

Definition 4.6 *A federation scheme is a pair $(\mathbb{E}, \delta)$ consisting of an evaluation rule and a distribution function.*

The central claim of the present paper, as already mentioned, is that a federation scheme is the proper unit of analysis for distributed query processing—or at any rate, for the purposes of logical analysis—and it is the relation between federation schemes and sets of cluster that can be described as sound and/or complete:

Definition 4.7 *Let $\mathscr{C}$ be a set of clusters. A federation scheme $(\mathbb{E}, \delta)$ is* sound *wrt. $\mathscr{C}$ if for any BGP P and any $\mathscr{R} \in \mathscr{C}$ we have that $\mathbb{E}(\delta(\mathscr{R}, P)) \subseteq [\![P]\!]_{\mathscr{R}}$. It is complete wrt. $\mathscr{C}$ if the converse inclusion holds.*

Bordering on circularity, one might say that a set of clusters represents a way of selecting RDF sources that keeps a federation scheme correct and exhaustive. Thus, soundness and completeness theorems will be made to act like chopsticks to pin down a federation scheme and a set of clusters that is perfectly matched, in effect guaranteeing that the federation scheme adequately captures the permissible clusters.

5. Completeness in the zero-knowledge case

As mentioned in section 3, the problem of giving a logical characterisation of SPARQL federation can be construed as the problem of generalizing the semantics of conjunctive queries to multiple sources: standard SPARQL semantics defines the evaluation $[\![S]\!]_G$ of a BGP S over an RDF graph G in terms of partial functions or maps from variables to RDF terms, and it defines the conjunction of different sets of answers $[\![S]\!]_G \bowtie [\![T]\!]_H$ in terms of the union of pairs of compatible maps, where compatibility means agreement on shared variables. However, the standard semantics requires that G and H are singletons and that $G = H$.

A semantics for distributed query processing requires that these restrictions be relaxed. In light of definition 3.1, this involves working towards an account of combinations

$$f(S)_J \mathbin{\overline{\wedge}} g(T)_K \tag{1}$$

where f and g is either of the evaluation functions $[\cdot]$ and $[\![\cdot]\!]$ (not necessarily different), and $\overline{\wedge}$ is $\cup$ or $\bowtie$. Of course, not all such combinations are of interest, only those for which $f(S)_J \mathbin{\overline{\wedge}} g(T)_K = [\![S \cup T]\!]_{J\cup K}$.

Of course,this equation places severe restrictions on eligible pairs of distribution functions and evaluation rules. This section focuses on one particularly natural set of such pairs, namely those formed from distributions functions that satisfy the condition of granularity—which includes all distribution functions introduced in the preceding section—together with the collect-and-combine evaluation rule (both to be defined). However, an example of a non-granular federation scheme is presented towards the end, which shows that the set of eligible federation schemes is much richer.

A few simple properties first:

Lemma 5.1 (Cluster monotony) $[\![S]\!]_J \subseteq [\![T]\!]_K$ *whenever* $J \subseteq K$.

Proof Immediate from the assumption that no RDF graphs share blank nodes.

Although cluster monotony does require the non-trivial property of disjointness for every pair of elements of a cluster wrt. blank nodes, this is not a real restriction, since renaming of blank nodes does not change the semantics of an RDF graph. Hence, one can always assume that blank nodes have been standardized apart in a preprocessing step.

The next lemma gives the case of formula (1) above for $f = g = [\![\cdot]\!]$ and $\overline{\wedge} = \bowtie$:

Lemma 5.2 *Let* $\mathscr{R}(t_1) = J$ *and* $\mathscr{R}(t_2) = K$ *for some cluster* $\mathscr{R}$. *Then* $[\![t_1 \cup t_2]\!]_{(J\cup K)} = [\![t_1]\!]_J \bowtie [\![t_2]\!]_K$.

Proof For the left-to-right direction suppose $\mu \in [\![t_1 \cup t_2]\!]_{(J\cup K)}$. Then $\mu \in [\![t_1]\!]_{(J\cup K)} \bowtie [\![t_2]\!]_{(J\cup K)}$, whence $\mu = \mu_1 \cup \mu_2$ for $\mu_1 \in [\![t_1]\!]_{(J\cup K)}$ and $\mu_2 \in [\![t_2]\!]_{(J\cup K)}$. Since t_1 and t_2 are singleton triple patterns, and μ_1 and μ_2 are functional, there is exactly one triple $g_1 \in R_i$ s.t. $\mu_1(t_1) = g_1$ and exactly one triple $g_2 \in R_j$ s.t. $\mu_2(t_2) = g_2$ for some $i, j \in J \cup K$. Clearly $sig(g_1) = sig(t_1) \subseteq sig(R_i)$ and $sig(g_2) = sig(t_2) \subseteq sig(R_j)$. Therefore, since by the supposition of the lemma $\mathscr{R}(t_1) = J$ and $\mathscr{R}(t_2) = K$ it follows that $R_i \in J$ and $R_j \in K$. Hence $\mu_1 \in [\![t_1]\!]_J$ and $\mu_2 \in [\![t_2]\!]_K$, so $\mu \in [\![t_1]\!]_J \bowtie [\![t_2]\!]_K$ as desired.

The restriction on the clusters J and K is essential. That is, lemma 5.2 does not hold for arbitrary pairs (t_1, J) and (t_2, K), which is as one would expect.

Lemma 5.3 $[S]_J \bowtie [T]_K \subseteq [\![S \cup T]\!]_{(J\cup K)}$

Proof Suppose $\mu \in [S]_J \bowtie [T]_K$. Then there are $\mu_1 \in [S]_J$ and $\mu_2 \in [T]_K$ s.t. $\mu = \mu_1 \cup \mu_2$. By definition 3.1 (2) there are $R_m \in J$ and $R_n \in K$ such that $\mu_1 \in [\![S]\!]_{R_i}$ and $\mu_2 \in [\![T]\!]_{R_n}$. Thus, applying lemma 5.1 gives $\mu_1 \in [\![S]\!]_{(J\cup K)}$ and $\mu_2 \in [\![T]\!]_{(J\cup K)}$, whence $\mu \in [\![S \cup T]\!]_{(J\cup K)}$ as desired.

The converse does not hold in general, but holds under a proviso that reflects clause 1 of definition 4:

Lemma 5.4 *Let* $\mathscr{R}(S) = J$ *and* $\mathscr{R}(T) = K$ *for some cluster* $\mathscr{R}$ *and suppose* $[\![S]\!]_J = [S]_J$ *and* $[\![T]\!]_K = [T]_K$. *Then* $[\![S \cup T]\!]_{(J\cup K)} \subseteq [S]_J \bowtie [T]_K$.

Proof It follows by definition 3.1(1) that $[\![S\cup T]\!]_{(J\cup K)} = [\![S]\!]_{(J\cup K)} \bowtie [\![T]\!]_{(J\cup K)}$. By the supposition of the lemma $\mathscr{R}(S) = J$ and $\mathscr{R}(T) = K$ so $[\![S]\!]_{(J\cup K)} \bowtie [\![T]\!]_{(J\cup K)} = [\![S]\!]_J \bowtie [\![T]\!]_K$. Since $[\![S]\!]_J = [S]_J$ and $[\![T]\!]_K = [T]_k$, it follows that $[\![S \cup T]\!]_{(J\cup K)} = [\![S]\!]_J \bowtie [\![T]\!]_K = [S]_J \bowtie [T]_K$, so $[\![S \cup T]\!]_{(J\cup K)} = [S]_J \bowtie [T]_K$ which is clearly sufficient.

We shall say that a distribution function that satisfies both directions is *agnostic*:

Definition 5.1 (Agnosticism) *A distribution function* δ *is agnostic if for every cluster* $\mathscr{R}$, *every BGP* P, *and every* $(S, J), (T, K) \in \delta(\mathscr{R}, P)$ *we have that* $[S]_J \bowtie [T]_K = [\![S \cup T]\!]_{(J\cup K)}$

As an example of a distribution function that is not agnostic, consider the trivial distribution defined by putting $\delta(\mathscr{R}, P) = \{(P, \mathscr{R})\}$.

Turning now to the announced condition of granularity, it is defined as:

Definition 5.2 (Granularity) *A distribution function δ is granular iff for every $(S, J) \in \delta(\mathscr{R}, P)$, either S or J is a singleton.*

It is evident by inspection of definition 4.2 that the even-, standard- and prudent distribution are all granular in this sense. Stretching the terminology somewhat, a federation scheme will also be called granular if its distribution function is.

Granularity and agnosticism are closely related. For one, we have implication from let to right:

Theorem 5.5 *If δ is granular then δ is agnostic.*

Proof Let $(S, J), (T, K) \in \delta(\mathscr{R}, P)$ and assume that δ granular. We show that $[S]_J \bowtie [T]_K = [\![S \cup T]\!]_{(J \cup K)}$. By lemma 5.3 we have the left to right direction, so it suffices to prove the converse. By definition 4.2 (1) we have that $\mathscr{R}(S) = J$ and $\mathscr{R}(T) = K$. Since δ is granular we also have that $[\![S]\!]_J = [S]_J$ and $[\![T]\!]_K = [T]_K$. Thus, the conditions of lemma 5.4 are fulfilled, whence $[\![S \cup T]\!]_{(J \cup K)} \subseteq [S]_J \bowtie [T]_K$ as desired.

As a corollary to this we have

Corollary 5.6 *The even- and prudent distribution are agnostic.*

Granular distribution functions have the important property that they make sets $[P]_J$ and $[\![P]\!]_J$ coincide:

Lemma 5.7 *Suppose δ is granular and that $(S, J) \in \delta(\mathscr{R}, P)$. Then $[\![S]\!]_J = [S]_J$.*

Proof If J is a singleton we have $[S]_J = \bigcup_{R_i \in J} [\![S]\!]_{R_i} = [\![S]\!]_{R_i} = [\![S]\!]_J$. Suppose $\mu \in [\![t]\!]_J$. For the second case suppose S is a singleton t. Then there is an $R_i \in J$ such that $\mu(t) \in R_i$ whence $\mu \in [\![t]\!]_{R_i}$. Now, $[\![t]\!]_{R_i} \subseteq \bigcup_{R_k \in J} [\![t]\!]_{R_k} = [t]_J$ so $\mu \in [t]_J$. For the converse, suppose $\mu \in [t]_J = \bigcup_{R_k \in J} [\![t]\!]_{R_k}$ then for some $R_i \in J$ we have $\mu \in [\![t]\!]_{R_i} \subseteq [\![t]\!]_J$, by answer set monotony.

The reader should note, that even when $[\cdot]$ and $[\![\cdot]\!]$ are identical *functions*, they are still different considered as evaluation *rules*, for recall that $[S]_J$ assembles the result of evaluating S over *each* $j \in J$, whereas $[\![S]\!]_J$ evaluates S directly over the *union* of J. Lemma 5.7 shows is that in the context of a granular distribution function these rules generate the same answers.

Turning finally to the announced collect-and-combine rule, this rule must surely have been noticed somewhere in the literature, though it has proved difficult to find an explicit statement of it:

Definition 5.3 (Collect-and-combine rule) *Put $\Delta := \delta(\mathscr{R}, P)$. If δ is not defined at $\mathscr{R}$ and P we put $\mathbb{E}(\Delta) =_{df} \emptyset$, otherwise*

$$\mathbb{E}(\Delta) =_{df} \bowtie \{[S]_J : (S, J) \in \Delta\}$$

The rule may be motivated follows: To produce a result set from a distribution, each cell in the distribution has to be evaluated against its designated sources and the partial answers that are returned have to be combined in order to yield an answer to the global query. The question, is how to unpack 'combine'. Note that just, fixating on one operator will not do. Consider for instance, the case where $\mu_i \in [\![t]\!]_{R_i}$ and $\mu_j \in [\![t]\!]_{R_j}$ for incompatible μ_i and μ_j (i.e. suppose t contains a variable $?x$ such that $\mu_i(?x) \neq \mu_j(?x)$). Then neither μ_i or μ_j is in $[\![t]\!]_{R_i} \bowtie [\![t]\!]_{R_j}$, even though they are both answers to the query t. Or, suppose instead that we form the union of partial answers, and consider the case where $\mu = \mu_i \cup \mu_j$ for compatible $\mu_i \in [\![S]\!]_{R_i}$ and $\mu_j \in [\![T]\!]_{R_j}$. Then μ need not be in $[\![S]\!]_{R_i} \cup [\![T]\!]_{R_j}$, whereas μ_i and μ_j always are, despite the fact that μ but not μ_i and μ_j may be an answer to $S \cup T$. The collect-and-combine rule is designed to balance the need for both operations.

Theorem 5.8 *If δ is agnostic then $(\mathbb{E}, \delta)$ is sound and complete wrt. to the set of all clusters.*

Proof Let $\mathscr{R}$ be any cluster and P any BGP. It suffices to show that $[\![P]\!]_{\mathscr{R}} = \bowtie \{[S]_J : (S, J) \in \delta(\mathscr{R}, P)\}$. This follows immediately from agnosticism together with clause 2 and 3 of definition 4.2.

As a simple corollary we have

Corollary 5.9 *Suppose δ is the even-, standard or prudent distribution. Then $(\mathbb{E}, \delta)$ is sound and complete wrt. the set of all clusters.*

5.1. Non-granular federation scheme

There are many examples of non-granular federation schemes that are sound and complete wrt. the set of all clusters. The following property can be taken to demarcates an entire set of such:

Theorem 5.10 *Let P be a BGP with no repeated occurences of variables. Then then $[\![P]\!]_J$ is equal to the k-th direct product of $[P]_J$ where $k = |P|$. That is $[\![P]\!]_J = ([P]_J)^{|P|}$.*

Proof By induction on the complexity of P. The case where P is a singleton t is trivial, so let $P = P_1 \cup P_2$ and assume as the induction hypothesis that the property holds for P_1 and P_2. Then $[\![P_1]\!]_J = ([P_1]_J)^{|P_1|}$ and $[\![P_2]\!]_J = ([P_2]_J)^{|P_2|}$. We have

$$\begin{aligned} [\![P]\!]_J &= [\![P_1]\!]_J \bowtie [\![P_2]\!]_J \\ &= ([P_1]_J)^{|P_1|} \bowtie ([P_2]_J)^{|P_2|} \end{aligned}$$

Now, since P does not have repeated occurrences of variables it follows that P_1 and P_2 do not have a variable in common. Hence $([P_1]_J)^{|P_1|} \bowtie ([P_2]_J)^{|P_2|} = ([P_1]_J)^{|P_1|} \times ([P_2]_J)^{|P_2|}$, which suffices to prove the theorem.

Theorem 5.10 suggest the following alternative evaluation rule:

Definition 5.4 *Put $\Delta := \delta(\mathscr{R}, P)$. If δ is not defined at $\mathscr{R}$ and P we put $\mathbb{E}^{\times}(\Delta) =_{df} \emptyset$, otherwise*

$$\mathbb{E}^{\times}(\Delta) =_{df} \bowtie \{([S]_J)^{|S|} : (S, J) \in \Delta\}$$

Given the conditions on the definition of a distribution function, it is not difficult to show that:

Theorem 5.11 *Let δ be any distribution function satsifying the following condition: For any cluster $\mathscr{R}$ and any BGP P, if $(S, J) \in \delta(\mathscr{R}, P)$ then S do not have repeated occurences of variables. Then the federation scheme $(\mathbb{E}^{\times}, \delta)$ is sound and complete wrt. the set of all clusters.*

The proof is a simple rerun of the proof of theorem 5.8, with minor adjustments made at obvious places. It has been omitted for reasons of economy.

Further exploration of the set of sound and complete federation schemes and their uses is left as a topic for future research.

6. An analysis of two examples from the literature

The case studies that will be presented in this section have been selected for two reasons: First, they were originally stated with sufficient precision to be amenable to an analysis using the concepts and techniques developed in the preceding sections. Secondly, they are mutually illuminating: one tells us about a potential source of incompleteness in the zero-knowledge case, the other about a potential source of incompleteness in the more specific case where all sources are assumed to be typed (explanation pending). It is, moreover, easy to see a shared pattern as the cause of the incompleteness: the property of granularity fails for the set of clusters that is assumed by each approach.

It should be stressed that neither of the proposals discussed in this section emphasises completeness as an important property to be attained. Sometimes it legitimately isn't, e.g. traversal-based federation [14,15,13,20] or top-k query processing [30,23,31], or any other approach based on the idea of probing the network in an exploratory fashion, usually dispenses with the concept of completeness entirely since it is usually either not clear what completeness is completeness with respect to, or it is simply not feasible to attain it. What follows should therefore not be taken as an argument to the effect that either of the approaches being analysed here is inherently flawed.

The first case study is of the approach underlying the ADERIS system described in Lynden et al. [22]. ADERIS is an acronym for *Adaptive Distributed Endpoint RDF Integration System*, which should serve to indicate its focus: the emphasis of the study is placed on query processing techniques which allow join-orderings to change during query execution as a means of continual refinement and optimization of the execution plan.

Lynden et al. [22] make a point of not relying on meta-data or statistics, whence it should not be too unfair to analyse the ADERIS approach as a zero-knowledge approach in the present sense of the term. Completeness of query answering, on the other hand, does not seem to be a primary concern.

However, ADERIS *is* a distributed query processing system in the sense described in section 2, which means that it relies on a decomposition algorithm to distribute subqueries to contributing sources. Lynden et al. [22] give the following concise description of this algorithm: "For each data source, a source query is generated that contains all of the triple patterns from the federated query that could possibly match the triples in the given data source". The quote describes a distribution function that can be recognized as having a familiar mathematical structure: define $P^{\rhd} =_{df} \mathscr{R}(P)$ and $\mathscr{R}^{\lhd} =_{df} \{t \in P : \forall R \in \mathscr{R}.sig(t) \in sig(R)\}$. As indicated by the quantifiers 'each' and 'all' in the quote above, an element of an ADERIS distribution is a pair (S, J) such that $S^{\rhd} = J$ and $J^{\lhd} = S$. That is,

S is a set of triple patterns *each* of which is covered by *all* sources in J, and conversely J is a set of sources such that the signature of *each* subsumes the signature of all elements in S. In lattice-theoretic terms, the pair of maps $(\rhd, \lhd)$ forms a Galois connection between P and $\mathscr{R}$, and an ADERIS distribution contains a set of formal concepts induced by this connection (cf. [9, chp. 4]). The distribution does not necessarily contain all such concepts as there is not necessarily a single source that covers the signature of all triple patterns in P, nor a single triple pattern that all sources cover. That is, the distribution may lack the top and bottom elements of the lattice, but is otherwise identical.

This reconstruction finds support in the following example of query decomposition from Lynden et al. [22]: The global query is (prefixes and filters excluded):

```
SELECT ?name ?url ?img WHERE {
    ?p foaf:name ?name.
    ?p foaf:homepage ?url.
    ?p foaf:depiction ?img.
    ?p dbpedia:occupation ?occupation.}
```

It is distributed over a set of sources that includes a pair, call it J, that cover the signature elements `foaf:name` and `foaf:homepage` and no other signature elements from the query. The ADERIS decomposition algorithm then assigns the same subpattern, call it S, to both of these sources:

```
    ?p foaf:homepage ?o1.
    ?p foaf:name ?o2.
```

In this example $S^{\rhd}$ is precisely J and $J^{\lhd} = S$.

In light of lemma 5.5 and theorem 5.8, one may suspect that the behaviour and abstract properties of the ADERIS distribution function will be dominated by the fact that it is not granular: in the example above we have a pair (S, J) where neither S nor J is a singleton. Lemma 5.5 says that granularity implies agnosticism and theorem 5.8 that agnosticism is sufficient for completeness. Hence, if the converses are also true, then it follows that the ADERIS federation engine may happen to produce incomplete answers when contributing sources are selected without knowledge of their content or structure beyond their signature. Since ADERIS appears to be designed precisely for zero-knowledge federation, the failure of completeness would stand in need of some justification explaining what it is that it is traded in for.

However, as it turns out the converses of lemma 5.5 and theorem 5.8 do not hold. That is, not unless we also postulate that the distribution function in question is principled and well-behaved in a couple of key senses. First, it needs to be *signature-based*:

Definition 6.1 *A distribution function δ is* signature-based *if $(S, J) \in \delta(\mathscr{R}, P)$ implies $(S, J') \in \delta(\mathscr{R}', P)$ whenever there is a $J' \subseteq J$ such that $sig(J) = sig(J')$*

All the distribution functions from definition 4.4 are signature-based in this sense.

Theorem 6.1 *If δ is agnostic and signature-based then it is granular.*

Proof Proof proceeds by contraposition. Suppose δ is not granular. Then there is an $(S, J) \in \delta(\mathscr{R}, P)$, for some $\mathscr{R}$ and P, such that neither S nor J is a singleton. Our strategy is to use S and J to construct a new cluster $\mathscr{R}'$ from $\mathscr{R}$ such that $(S, J') \in \delta(\mathscr{R}', P)$ for $J' \subseteq \mathscr{R}'$ and such that $[S]_{J'} \neq [\![S]\!]_{J'}$. Since $[S]_{J'} \bowtie [S]_{J'} = [S]_{J'}$ and $[\![S \cup S]\!]_{J' \cup J'} = [\![S]\!]_{J'}$ this suffices to show that δ is not agnostic. Therefore, partition S into two sets S^- and $\{t\}$, it doesn't matter how t is chosen. We have assumed that BGPs do not contain RDF triples so t contains a variable. Assume wlog. that $t := (?x, p, o)$, where p and o may or may not be variables too. Based on S^- and $\{t\}$ we construct four sets of RDF triples $G_1, G_2, \{g_1\}, \{g_2\}$. We make sure that these sets satisfy the following constraints:

1. $g_1 := (s_1, p', o')$, $g_2 := (s_2, p'', o'')$ where p', p'' match p and o', o'' match o.
2. G_1 matches S^-
3. G_1 contains s_1 and in place of $?x$ (if $?x$ occurs S^-).
4. There is a μ such that $\mu(S) = G_1 \cup g_1$
5. g_1 and g_2 do not occur in $\bigcup \mathscr{R}$

To see that this construction is well defined, note first that step 1, 5 and 3 require no justification. Step 2 is warranted by the assumption that S is not a singleton whence $G_1 \neq \emptyset$, and step 4 is warranted by step 3.

Now, partition J into two non-empty sets J_1 and J_2. This is possible since J is not a singleton. We construct a new set $J' := J_1' \cup J_2'$ by putting $J_1' := \{R \cup g_2 \cup G_1 : R \in J_1\}$ and $J_2' := \{R \cup g_1 : R \in J_2\}$. Let $\mathscr{R}'$ be exactly like $\mathscr{R}$ except that J is replaced by $J' = J_1' \cup J_2'$. By 1 and 2 above $sig(J) = sig(J')$ whence $(S, J') \in \delta(\mathscr{R}', P)$, by the assumption that δ is signature-based. By 4 $\mu(S) = G_1 \cup g_1$, and by 5 and the construction of J', $G_1 \cup g_1$ does not occur in any $R' \in J'$. It follows that $\mu \notin [S]_{J'}$. Yet, since $g_1 \cup G_1 \in \bigcup J'$ we have that $\mu \in [\![S]\!]_{J'}$ so the proof is complete.

Secondly, it needs to be *uniform*:

Definition 6.2 (Uniformity) *Let $\Delta \subseteq \delta(\mathscr{R}, P)$. Put $\mathscr{R}^- := \bigcup_{(S,J)\in\Delta} J$ and $P^- := \bigcup_{(S,J)\in\Delta} S$. Then $\Delta = \delta(\mathscr{R}^-, P^-)$*

This too is a property shared by all the distribution functions from definition 4.4. Note that the fact that theorem 6.1 does not by itself entail zero-knowledge incompleteness, means that a distribution scheme may fail agnosticism and still be zero-knowledge complete as the following example shows:

Example Let δ be any distribution function defined at $\mathscr{R}$ and P such that $(\mathbb{E}, \delta)$ is sound and complete wrt. to the set of all clusters. Let f and g be functions that select subsets of P such that $f(P) \cap g(P) \neq \emptyset$ and $f(P) \cup g(P) = P$. To keep notation less verbose put $A := f(P)$ and $B := g(B)$, and define δ^* as follows:

$$\delta^*(\mathscr{R}, P) =_{df} \delta(\mathscr{R}, P) \cup \{(A, \mathscr{R}(A)), (B, \mathscr{R}(B))\}$$

To show that $\mathscr{R}, P$, f and g can be chosen in such a way that $[A]_{\mathscr{R}(A)} \bowtie [B]_{\mathscr{R}(B)} \neq [\![A \cup B]\!]_{\mathscr{R}(A)\cup\mathscr{R}(B)}$, it suffices to put $A = P = B$ and rig an example similar to that employed in the proof of theorem 6.1. It remains only to show that $(\mathbb{E}, \delta^*)$ is complete wrt. the set of all clusters: Put $D := \{(A, \mathscr{R}(A)), (B, \mathscr{R}(B))\}$. Since 1) $[A]_{\mathscr{R}(A)} \bowtie [B]_{\mathscr{R}(B)} \subseteq [\![A \cup B]\!]_{\mathscr{R}(A)\cup\mathscr{R}(B)} \subseteq [\![A \cup B]\!]_{\mathscr{R}} = [\![P]\!]_{\mathscr{R}}$, by lemma 5.1 and the supposition that $A \cup B = P$, we have:

$\mathbb{E}(\delta^*(\mathscr{R}, P)) =$

$\bowtie \{[S]_j : (S, J) \in \delta(\mathscr{R}, P) \cup D\}$

by definition of $\mathbb{E}$ and δ^*

$= \bowtie \{[S]_j : (S, J) \in \delta(\mathscr{R}, P)\} \bowtie (\bowtie \{[T]_K : (T, K) \in D\})$

by the associativity of $\bowtie$

$= \bowtie \{[S]_j : (S, J) \in \delta(\mathscr{R}, P)\} \bowtie ([A]_{\mathscr{R}(A)} \bowtie [B]_{\mathscr{R}(B)})$

resolving D

$= [\![P]\!]_{\mathscr{R}} \bowtie ([A]_{\mathscr{R}(A)} \bowtie [B]_{\mathscr{R}(B)})$

by theorem 5.8

$= [\![P]\!]_{\mathscr{R}}$

by (1) above

Therefore $\delta(\mathscr{R}, P) = \delta^*(\mathscr{R}, P)$, so $(\mathbb{E}, \delta^*)$ is sound and complete wrt. the set of all clusters, although δ^* is not agnostic.

On the other hand:

Theorem 6.2 *If δ is uniform but not agnostic then $(\mathbb{E}, \delta)$ is incomplete wrt. the set of all clusters.*

Proof Suppose that δ is not agnostic. Then there is a cluster $\mathscr{R}$ and a BGP P such that there are $(S, J), (T, K) \in \delta(\mathscr{R}, P)$ with $S, T \subseteq P$ and $J, K \subseteq R$ and $[S]_J \bowtie [T]_K \neq [\![S \cup T]\!]_{(J\cup K)}$. Since δ is uniform we have$\delta(S \cup T, J \cup K) = \{(S, J), (T, K)\}$. Hence $\mathbb{E}(\delta(S \cup T, J \cup K)) = [S]_J \bowtie [T]_K \neq [\![S \cup T]\!]_{(J\cup K)}$. Thus $(\mathbb{E}, \delta)$ is incomplete wrt. the set of all clusters by lemma 5.3.

As a simple corollary we have:

Corollary 6.3 *If δ is uniform and signature-based but not granular, then $(\mathbb{E}, \delta)$ is incomplete wrt. to the set of all clusters.*

Taking stock, the status wrt. the ADERIS approach, then, is the following: the distribution function, on a fair interpretation of it, is not granular. Therefore, it is either not uniform or not signature-based or else it yields an incomplete federation scheme when paired with the collect-and-combine rule. Of course, as theorem 5.11 shows there other evaluation rules that yield complete federation schemes for certain choices of distribution functions. Note however, that the ADERIS distribution function is not an eligible choice for the particular rule given in definition 5.4 since that rule requires that S in (S, J) have no repeated occurrences of variables. As the example from Lynden et al. [22] shows, this is not a property of the ADERIS distribution function.

Turning now to the second announced case study, another approach that can fruitfully be analysed in terms of federation schemes and sets of clusters is the one from Langegger et al. [21] which describes the SemWIQ system. SemWIQ is based on the very interesting notion of *concept-based* integration. The principal idea behind concept-based integration is to require a global query to contain an explicit type assertion for each subject variable that occurs in it. The SemWIQ SPARQL federator uses these types to constrain the subqueries that are assigned to the contributing sources, requiring all retrieved instances to be similarly typed.

Clearly, concept-based integration is not compatible with a granular distribution function: when the global query is decomposed into subqueries, each subquery must contain type assertions for all of its subject variables, which means that all subqueries, including those that are routed to more than one source, will contain at least two triple patterns and so violate granularity

Yet, concept-based integration is equally clearly not designed to be a complete federation strategy for *all* ways of selecting sources, since it anyway requires of eligible clusters that they have, in a sense to be made precise, a predictable type system. The possibility remains, therefore, that a concept-based federation scheme can be proved to be sound and complete wrt. such a reduced set of clusters. An argument to this effect is here provided in outline:

Call a BGP P *typed* iff for every $t_1 \in P$ with a variable in subject position there is a $t_2 \in P$ that specifies the required type of that subject. It is convenient to assume that types are always asserted with the same predicate, typically `rdf:type`. Call a cluster $\mathscr{R}$ *consistently typed* if a resource is qualified by the same set of type assertions in each of the sources in which it occurs, given that it occurs in subject position. We have:

Lemma 6.4 $[\![t_1 \cup t_2]\!]_J = [t_1 \cup t_2]_J$ *whenever* t_2 *types the subject of* t_1 *and* J *is a consistently typed cluster.*

Proof It suffices to prove the left-to-right direction. Suppose $\mu \in [\![t_1 \cup t_2]\!]_J = [\![t_1]\!]_J \bowtie [\![t_2]\!]_J$. Then $\mu = \mu_1 \cup \mu_2$ for some $\mu_1 \in [\![t_1]\!]_J$ and $\mu_2 \in [\![t_2]\!]_J$. Since t_1 is a singleton, there is an $R_i \in J$ such that $\mu_1(t_1) \in R_j$ whence $\mu_1 \in [\![t_1]\!]_{R_i}$. Therefore, since J is a well-typed cluster, it follows that $\mu_2(t_2) \in R_i$ as well. Hence $\mu \in [\![t_1]\!]_{R_i} \bowtie [\![t_2]\!]_{R_i} = [\![t_1 \cup t_2]\!]_{R_i} \subseteq \bigcup_{R_k \in J}[\![t_1 \cup t_2]\!]_{R_k} = [t_1 \cup t_2]_J$.

It follows from this simple result that it is possible to give a sound and complete federation scheme for concept-based integration under a slight generalization of definition 4.2. The generalization involves sorting RDF properties into two categories, say the *domain specific* and *generic* ones, and to count only the domain specific ones as signature elements. One can then add 'logical' vocabulary such as `rdf:type` to any BGP S in any dsitribution element (S, J) without interfering with the relationship $\mathscr{R}(S) = J$.

As a concrete example of such a distribution function, put $\mathscr{F} := (\mathbb{E}, \delta^t)$ where $\mathbb{E}$ is the collect-and-combine rule. Let δ be the standard distribution and define δ^t as follows: For typed P if $(S, J) \in \delta(\mathscr{R}, P)$ then $(S \cup T, J) \in \delta^t(\mathscr{R}, P)$ where T is the set of type assertions for subjects occurring in S. Now, if J is a singleton in (S, J) then it remains so in $(S \cup T, J)$, since the type property is a generic element, and if $T = \emptyset$ then $(S \cup T, J) = (S, J)$. For these cases, lemma 5.7 tells us that $[\![\cdot]\!]$ and $[\cdot]$ coincide. The case where T is nonempty and J contains more than one element, is covered by lemma 6.4. The agnosticism of δ^t now follows immediately from the assumption that the contributing sources are consistently typed, whence the federation scheme $(\mathbb{E}, \delta^t)$ is complete wrt. to the same set.

This shows that the idea of concept-based federation is perfectly compatible with soundness and completeness, given that the latter notions are qualified wrt. the set of consistently typed clusters. However, the particular distribution function that SemWIQ employs is, somewhat surprisingly, not characterised by this qualification. Witness the following sample query from [21] (it was reportedly executed against sunspot observations recorded at Kanzelhöhe Solar Observatory, which should give some intuitive explanation for the vocabulary):

```
SELECT ?dt ?groups ?spots ?r ?fn ?ln WHERE {
    ?s rdf:type sobs:SunspotRelativeNumbers.
    ?s sobs:dateTime ?dt.
    ?s sobs:groups ?groups.
    ?s sobs:spots ?spots.
    ?s sobs:rValue ?r.
    ?obs rdf:type obs:Observer.
    ?obs person:firstName ?fn.
    ?obs person:lasttName ?ln.}
```

The setup of the experiment is such that there are two source that share all the signature elements prefixed by `sobs:` and contain none prefixed by `person:`. Both sources are assigned the following subpattern:

```
    ?s rdf:type sobs:SunspotRelativeNumbers.
    ?s sobs:dateTime ?dt.
    ?s sobs:groups ?groups.
    ?s sobs:spots ?spots.
    ?s sobs:rValue ?r.
    ?obs rdf:type obs:Observer.
```

Remove the type assertions and this indicates the same distribution function as that employed by the ADERIS system, i.e. a distribution function based on the Galois connection between sources and signature elements. Whether or not this is indeed the case cannot conclusively be decided on the basis of the text. It is nevertheless clear that, whatever the distribution function is it is not granular wrt. the set of consistently typed clusters. The reasoning applied to the case of the ADERIS distribution can therefore be rerun with a restriction to the aforementioned set of clusters, to conclude that SemWIQ, like ADERIS, is either not uniform or not signature-based, or else it yields an incomplete federation scheme when paired with the collect-and-combine rule—even when incompleteness is measured wrt. the restricted set of consistently typed clusters.

7. A corollary wrt. join-order heuristics

As a way to tie up loose ends, the present section shows how to fit a federation scheme into a join-ordering in order to obtain an execution plan for a given distribution. Attention is restricted to federation schemes based on the collect-and-combine rule from section 5.

A join ordering will be represented as usual as an operator tree, based on the following definitions from [12]:

Definition 7.1 (Tree domain) *A tree domain D is a non-empty subset of the set of all strings $\mathbb{N}^*$ over the natural numbers, satisfying the following conditions:*

1. *For each $u \in D$ every prefix of u is also in D.*
2. *For each $u \in D$ and $i \in \mathbb{N}^*$, if $ui \in D$ then for every j, $1 \leq j \leq i$, $uj \in D$.*

Given a tree domain D the relation $x \leq y$ denotes the prefix relation on strings of D. The covering relation $x \prec y$ is defined by stipulating that if $x \prec y$ and $x \leq z \leq y$ then either $x = z$ or $z = y$.

Definition 7.2 *Given a set Σ of symbols, a tree over Σ is a function $d : D \longrightarrow \Sigma$ where D is a tree domain.*

Definition 7.3 (Subtrees) *Given a tree d and a node u in $dom(d)$, the* subtree rooted at u *is the tree d/u whose domain is the set $\{v : uv \in dom(d)\}$ and such that $(d/u)(v) = d(uv)$ for all v in $dom(d/u)$.*

Every element of a domain D of a tree d will be called a node of d. The *outdegree* $out(u)$ of a node u is the cardinality of the set $\{i : ui \in dom(d)\}$. A node u with $out(u) = 0$ is called a *leaf*. The leaves of d are denoted $L(d)$. Indegrees are defined as $in(u) = \{i : iu \in dom(d)\}$. Of course all nodes in a tree have indegree equal to 1 except one, the root, which has indegree zero. If d is a tree, $r(d)$ denote the root of d.

Definition 7.4 (Ordered distribution) *Let Δ be a distribution. A join-ordering of Δ is a tree d over $\{\bowtie\} \cup \Delta$ such that:*

1. *d is a bijection between $L(d)$ and Δ*
2. *if $out(x) > 1$ then $d(x) = \bowtie$*

Definition 7.4 represents a generalisation of the usual notion of the join-ordering of a BGP. It is a generalization in the sense that the carrier set of the poset is not a set of triples but a distribution, that is, itself a set of BGPs.

In computational terms it is natural to see this generalization as an adaptation in which the join-ordering also distributes the responsibility for executing joins: a distribution assigns a subquery of a global query to a set of contributing sources. If the subquery in question contains more than one triple pattern—which will be the case if it is e.g. an exclusive group—then the subquery is handed over to the contributing sources along with the responsibility for executing the joins involved.

It should be fairly obvious how to relate definition 7.4 to the completeness results from section 5:

Definition 7.5 (Evaluation) *The evaluation of an ordered distribution d, written $[\![d]\!]$, is*

1. *$[S]_J$ if $dom(d)$ contains a single leaf x such that $d(x) = (S, J)$*
2. *otherwise it is the application of $r(d)$ to $[\![d/y_1]\!]$ and $[\![d/y_2]\!]$ for $i \in [1, 2]$*

We have:

Corollary 7.1 *Let δ be either of the even-, standard- or prudent distributions and suppose it is defined at $\mathscr{R}$ and P. Let d be a join-ordering of $\delta(\mathscr{R}, P)$. Then $[\![d]\!] = \mathbb{E}(\delta(\mathscr{R}, P))$.*

Proof By definition 7.4(2) and definition 7.4(2) we have that $[\![d]\!] = \bowtie (d(L(D))$. Therefore, since we have $d(L(d)) = \delta(\mathscr{R}, P)$ by definition 7.4(2) it follows that $[\![d]\!] = \mathbb{E}(\delta(\mathscr{R}, P))$ as desired.

None of this is surprising, which is why it is presented as a corollary. Nevertheless there are a couple of things to note: First, the unit of heuristic significance in a distributed setting is the subpattern, not the triple pattern. Moreover, all subpatterns are by default equally privileged. That is, there is no constraint on the shape of a tree that dictates that e.g. exclusive groups be processed first. This seems to contrast with for instance the FedX system [27], where exclusive groups appear to be given priority based on the assumption that they are more constrained than singleton triple patterns.[1] Yet, this is is not necessarily the case. Heuristic rules that do not rely on statistics will usually determine a priority ordering between patterns by analysing their lexical form. Typically the heuristic value of a sin-

[1] Again, this is not a matter that can be conclusively decided based on the text of [27]

gle triple will be determined by assessing how selective that triple pattern is relative to others. For instance a pattern of form `(?x, p, o)` can plausibly taken to be more constrained than `(?x, p, ?y)` and so forth, whereas the heuristic value of a join $t_1 \bowtie t_2$ will be determined by ranking the position and number of shared variables involved (cf. [29]).There is no *a priori* reason why such a procedure, generalised from triple patterns to BGPs in the obvious manner, should assign a higher accumulated heuristic value to an exclusive group, say. On the contrary it is not too difficult to come up with intuitive counterexamples that involve exclusive groups with a high variable to term ratio and a small number of joins being ranked below a single triple pattern with an instantiated subject and object.

The second thing to note, is the generality of the collect-and-combine rule which yields sound and complete heuristics for each of the even-, standard and prudent distributions in one fell swoop. Indeed, one may speculate that there is little reason for employing a different evaluation rule with these distributions functions—or, at any rate, at least in the zero knowledge case. If there were, then it would be for heuristic reasons. However, in the absence of detailed knowledge about the contributing sources there seems to be no reason to think that some other particular evaluation order would do better than the collect-and-combine rule. Whilst evaluating the two pairs $(\{t_1, t_2\}, J)$ and (t_3, K), say, by distributing joins over unions $([t_1]_J \bowtie [t_3]_K) \cup ([t_2]_J \bowtie [t_3]_K)$ would preserve completeness, the lexical form of the patterns involved does not lend support to one over the other.

8. Conclusion

This foundational study has provided some first steps towards a logical framework for reasoning about distributed query processing in SPARQL in a formal manner. The notion of a federation scheme consisting of an evaluation rule and a distribution function, turns out to be a quite versatile tool for characterising different approaches to federation, as well as for proving meta-properties such as soundness and completeness wrt. different sets of clusters, representing different ways of selecting contributing sources.

In particular, the combination of the property of granularity with the collect-and-combine rule has emerged as the shared trait that induces completeness in a zero-knowledge computational environment for several natural examples of distribution functions, including the standard distribution, so-called, which occurs frequently in the literature.

By analysing two proposed systems, the present paper can also be taken to have shown how the manner of analysis based on federation schemes and sets of clusters can also be fruitfully applied to existing systems as a means of articulating unclarities and ambiguities in the underlying definitions.

Federation schemes have a simple and transparent interface towards the heuristic notion of a join ordering of a query. This interface connects execution plans to soundness and completeness results in a manner that makes it evident that whole classes of execution plans preserve the aforementioned properties.

A natural line of future research is to extend the present account to the full SPARQL language. Also, it would be interesting to explore further the set of federation schemes that is sound and complete wrt. the set of all clusters, in particular the set of non-granular ones that were introduced in subsection 5.1.

References

[1] Abiteboul, S. and Vianu, V. [1997], Queries and computation on the web, *in* F. Afrati and P. Kolaitis, eds, 'Database Theory âĂŤ ICDT '97', Vol. 1186 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 262–275.

[2] Acosta, M., Vidal, M.-E., Lampo, T., Castillo, J. and Ruckhaus, E. [2011], Anapsid: An adaptive query processing engine for sparql endpoints., *in* 'Proceedings of the 10th International Semantic Web Conference (ISWC 2011)'.

[3] Arenas, M., Gutierrez, C. and PÃľrez, J. [2009], Foundations of rdf databases, *in* S. Tessaris, E. Franconi, T. Eiter, C. Gutierrez, S. Handschuh, M.-C. Rousset and R. Schmidt, eds, 'Reasoning Web. Semantic Technologies for Information Systems', Vol. 5689 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 158–204.

[4] Basca, C. and Bernstein, A. [2010], Avalanche: Putting the spirit of the web back into semantic web querying, *in* 'ISWC Posters&Demos'.

[5] Betz, H., Gropengießer, F., Hose, K. and Sattler, K.-U. [2012], Learning from the history of distributed query processing - a heretic view on linked data management, *in* 'COLD'.

[6] Bouquet, P., Ghidini, C. and Serafini, L. [2009], Querying the web of data: A formal approach, *in* A. GÃşmez-PÃľrez, Y. Yu and Y. Ding, eds, 'The Semantic Web', Vol. 5926 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 291–305.

[7] Bizer, C., Heath, T. and Berners-Lee, T. [2009], 'Linked data - the story so far', *Int. J. Semantic Web Inf. Syst.* **5**(3), 1–22.

[8] Choi, H., Son, J., Cho, Y., Sung, M. K. and Chung, Y. D. [2009], SPIDER: a system for scalable, parallel / distributed evaluation of large-scale RDF data, *in* 'Proceedings of the 18th ACM conference on Information and knowledge management', CIKM '09, ACM, New York, NY, USA, pp. 2087–2088.

[9] Davey, B. A. and Priestley, H. A. [2002], *Introduction to Lattices and Order*, Cambridge University Press.

[10] Görlitz, O. and Staab, S. [2011], Federated data management and query optimization for linked open data., *in* A. Vakali and L. C. Jain, eds, 'New Directions in Web Data Management 1', Vol. 331 of *Studies in Computational Intelligence*, pp. 109–137.

[11] Görlitz, O. and Staab, S. [2011], SPLENDID: SPARQL Endpoint Federation Exploiting VOID Descriptions, *in* 'Proceedings of the 2nd International Workshop on Consuming Linked Data (COLD 2011)'.

[12] Gorn, S. [1967], 'Explicit definitions and linguistic dominoes', *Journal of Computer and System Sciences* .

[13] Hartig, O. [2012], Sparql for a web of linked data: Semantics and computability, *in* 'Proceedings of the 9th International Conference on The Semantic Web: Research and Applications', ESWC'12, Springer-Verlag, Berlin, Heidelberg, pp. 8–23.

[14] Hartig, O. [2011], Zero-knowledge query planning for an iterator implementation of link traversal based query execution, *in* 'Proceedings of the 8th Extended Semantic Web Conference on The Semantic Web: Research and Applications - Volume Part I', ESWC'11, Springer-Verlag, Berlin, Heidelberg, pp. 154–169.

[15] Hartig, O., Bizer, C. and Freytag, J.-C. [2009], Executing sparql queries over the web of linked data, *in* 'Proceedings of the 8th International Semantic Web Conference', ISWC '09, Springer-Verlag, Berlin, Heidelberg, pp. 293–309.

[16] Hose, K., Schenkel, R., Theobald, M. and Weikum, G. [2011], Database foundations for scalable rdf processing, *in* 'Proceedings of the 7th international conference on Reasoning web: semantic technologies for the web of data', RW'11, Springer-Verlag, Berlin, Heidelberg, pp. 202–249.

[17] Huang, J., Abadi, D. J. and Ren, K. [2011], 'Scalable sparql querying of large rdf graphs', *PVLDB* **4**(11), 1123–1134.

[18] Husain, M. F., McGlothlin, J., Masud, M. M., Khan, L. R. and Thuraisingham, B. [2011], 'Heuristics-based query processing for large rdf graphs using cloud computing', *IEEE Transactions on Knowledge and Data Engineering* **23**(9), 1312–1327.

[19] Kossmann, D. [2000], 'The state of the art in distributed query processing', *ACM Comput. Surv.* **32**(4), 422–469.

[20] Ladwig, G. and Tran, T. [2011], Sihjoin: Querying remote and local linked data, *in* G. Antoniou, M. Grobelnik, E. Simperl, B. Parsia, D. Plexousakis, P. Leenheer and J. Pan, eds, 'The Semantic Web: Research and Applications', Vol. 6643 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 139–153.

[21] Langegger, A., Wöß, W. and Blöchl, M. [2008], A semantic web middleware for virtual data integration on the web, *in* 'Proceedings of the 5th European semantic web conference on The semantic web: research and applications', ESWC'08, Springer-Verlag, Berlin, Heidelberg, pp. 493–507.

[22] Lynden, S., Kojima, I., Matono, A. and Tanimura, Y. [2010], 'Adaptive integration of distributed semantic web data', pp. 174–193.

[23] Magliacane, S., Bozzon, A. and Della Valle, E. [2012], Efficient execution of top-k sparql queries, *in* P. CudrÃl'-Mauroux, J. Heflin, E. Sirin, T. Tudorache, J. Euzenat, M. Hauswirth, J. Parreira, J. Hendler, G. Schreiber, A. Bernstein and E. Blomqvist, eds, 'The Semantic Web âĂŞ ISWC 2012', Vol. 7649 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 344–360.

[24] Montoya, G., Vidal, M.-E. and Acosta, M. [2012], A heuristic-based approach for planning federated sparql queries., *in* J. Sequeda, A. Harth and O. Hartig, eds, 'COLD', Vol. 905 of *CEUR Workshop Proceedings*, CEUR-WS.org.

[25] Papailiou, N., Konstantinou, I., Tsoumakos, D. and Koziris, N. [2012], H2rdf: adaptive query processing on rdf data in the cloud., *in* A. Mille, F. L. Gandon, J. Misselis, M. Rabinovich and S. Staab, eds, 'WWW (Companion Volume)', ACM, pp. 397–400.

[26] Quilitz, B. and Leser, U. [2008], Querying Distributed RDF Data Sources with SPARQL, *in* 'Proc. ESWC' 08'.

[27] Schwarte, A., Haase, P., Hose, K., Schenkel, R. and Schmidt, M. [2011], Fedx: optimization techniques for federated query processing on linked data, *in* 'Proceedings of the 10th international conference on The semantic web - Volume Part I', ISWC'11, Springer-Verlag, Berlin, Heidelberg, pp. 601–616.

[28] Tran, T., Wang, H. and Haase, P. [2009], Hermes: Data Web search on a pay-as-you-go integration infrastructure. Web Semantics: Science, Services and Agents on the World Wide Web, vol. 7, no. 3. Springer 2009, 189–203.

[29] Tsialiamanis, P., Sidirourgos, L., Fundulaki, I., Christophides, V. and Boncz, P. [2012], Heuristics-based query optimisation for sparql, *in* 'Proceedings of the 15th International Conference on Extending Database Technology', EDBT '12, ACM, New York, NY, USA, pp. 324–335.

[30] Valle, E. D., Schlobach, S., KrÃűtzsch, M., Bozzon, A., Ceri, S. and Horrocks, I. [2013], 'Order matters! harnessing a world of orderings for reasoning over massive data.', *Semantic Web* **4**(2), 219–231.

[31] Wagner, A., Duc, T. T., Ladwig, G., Harth, A. and Studer, R. [2012], Top-k linked data query processing, *in* 'Proceedings of the 9th international conference on The Semantic Web: research and applications', ESWC'12, Springer-Verlag, Berlin, Heidelberg, pp. 56–71.