RC25278 (WAT1204-035) April 12, 2012
Computer Science

IBM Research Report

Querying Linked Ontological Data through Distributed Summarization

Achille Fokoue
IBM Research Division
Thomas J. Watson Research Center
P.O. Box 704
Yorktown Heights, NY 10598
USA

Felipe Meneguzzi
Carnegie Mellon University

Murat Sensoy, Jeff Z. Pan
University of Aberdeen

Querying Linked Ontological Data through Distributed Summarization *

Achille Fokoue,
IBM Watson Research Center
achille@us.ibm.com

Felipe Meneguzzi,
Carnegie Mellon University
felipe.meneguzzi@gmail.com

Murat Sensoy,
Jeff Z. Pan
University of Aberdeen
m.sensoy,jeff.z.pan@abdn.ac.uk

Abstract

As the semantic web expands, ontological data becomes distributed over a large network of data sources on the Web. Consequently, evaluating queries that aim to tap into this distributed semantic database necessitates the ability to consult multiple data sources efficiently. In this paper, we propose methods and heuristics to efficiently query distributed ontological data based on a series of properties of summarized data. In our approach, each source summarizes its data as another RDF graph, and relevant section of these summaries are merged and analyzed at query evaluation time. We show how the analysis of these summaries enables more efficient source selection, query pruning and transformation of expensive distributed joins into local joins.

Introduction

Linked Data is extending the Web into a global space of RDF data, with more than 30 billion RDF statements being online, contributed not only by government entities (*e.g.,* data.gov) and scientific communities (*e.g.,* the Bio-medical community), but also by companies (*e.g.,* BestBuy) and community driven efforts (*e.g.,* DBpedia). Ontological vocabulary is used to annotate RDF data; such vocabulary can be defined in an ontology using the OWL Web Ontology Language. The use of ontological vocabulary necessitates the use of reasoning.

Query processing over distributed and large numbers of data sources becomes one of the key challenges for the semantic web. Even without reasoning, this is a challenging task. Querying all sources in the network would be inefficient because only a small number of sources might be relevant for a given query. Furthermore, querying a semantic data network may amount to querying each relevant source for partial answers and joining the partial answers over the network. Hartig *et al.* proposed to leverage the correspondence between source addresses and identifiers used in the queries (Hartig, Bizer, and Freytag 2009). Konrath et al. used extracted schemas for source selection (Konrath, Gottron, and Scherp 2011). Harth *et al.* proposed a data summarization approach for querying Linked Data (Harth et al. 2010), without considering reasoning.

In this paper, we investigate the use of distributed summarization so as to enable scalable reasoning for querying linked ontological data. Prior work (Fokoue et al. 2006) has shown that practical scalable query answering and reasoning can be achieved on large local knowledge bases through the construction of a local summary $\mathcal{A}'$ of an ABox $\mathcal{A}$ (the data part of an ontology). The summary $\mathcal{A}'$ captures the patterns of relationships between individuals in $\mathcal{A}$. However, efficient aggregation of local summaries into a global summary remains as a significant challenge in distributed settings.

Our key contributions in this paper are threefold. First, we propose an alternative summarization technique which can be efficiently built in a decentralized fashion. By exploiting the principles of Linked Data, each source summarizes its data as another RDF graph, which is significantly smaller in size and hides details of the actual data by containing only the *patterns* repeating in the data. Relevant (w.r.t. input queries) sections of these summaries are merged and analyzed at query evaluation time. Second, we show how the analysis of these summaries enables more efficient source selection, query pruning and transformation of expensive distributed joins into local joins. Finally, through experiments over real and synthetic data, we show that our approach provides significant performance gains over state-of-the-art federated query engines for distributed semantic data.

Preliminaries

In the rest of the paper, we use the term ontology and knowledge base interchangeably. Description Log-

*Research was sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defence and was accomplished under Agreement Number W911NF-06-3-0001. The views and conclusions contained in this document are those of the author(s) and should not be interpreted as representing the offcial policies, ei- ther expressed or implied, of the U.S. Army Research Laboratory, the U.S. Government, the U.K. Ministry of Defence or the U.K. Government. The U.S. and U.K. Govern- ments are authorised to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

ics (DLs) are the formal underpinning of the OWL standard. Due to limited space, we refer the reader to (Baader et al. 2002) for details of DLs.

Conjunctive Query

Let $\mathcal{N}_I$ be a set of individuals, $\mathcal{N}_R$ a set of roles, and $\mathcal{N}_C$ a set of concepts in knowledge base (*i.e.,* ontology) $\mathcal{K} = (\mathcal{A}, \mathcal{T})$, where $\mathcal{A}$ is the ABox and $\mathcal{T}$ is the TBox (schema part of $\mathcal{K}$). Let $\mathcal{N}_V$ be set of variables. We assume that these four sets are mutually disjoint. A conjunctive query q is of the form $(x_1, ..., x_n) \leftarrow t_1 \wedge ... \wedge t_m$ where, for $1 \leq i \leq n$, $x_i \in \mathcal{N}_V$ and, for $1 \leq j \leq m$, t_j is a query term. A query term t is of the form $C(x)$ or $R(x, y)$ where x and y are either variables in $\mathcal{N}_V$ or individuals in $\mathcal{N}_I$, C is an atomic concept and R is an atomic role. We consider the evaluation of a conjunctive query w.r.t. a $\mathsf{DL\text{-}Lite}_{\mathcal{R}}$ $\mathcal{T}$ using the standard first order semantics presented in (Calvanese et al. 2007). A construct conjunctive query q is of the form $\mathsf{construct}[c_1, ..., c_n] \leftarrow t_1 \wedge ... \wedge t_m \wedge \mathsf{filter}(\alpha_1) \wedge ... \wedge \mathsf{filter}(\alpha_p)$ where, c_i and t_i have the same form as terms in a conjunctive query and α_i is a boolean expression. Furthermore, variables in c_i are restricted to those appearing in t_i or α_i. Construct conjunctive queries are evaluated only w.r.t an empty $\mathcal{T}$. The semantics is similar to SPARQL construct[1] query semantics; that is, the evaluation of q builds the $\mathcal{A}$ obtained by instantiating patterns c_i for each variable binding that satisfies all the constraints $t_1 \wedge ... \wedge t_m \wedge \alpha_1 \wedge ... \wedge \alpha_p$.

Centralized Summary ABox

Prior work (Fokoue et al. 2006) has demonstrated that practical scalable query answering and reasoning -even in very expressive description logics- can be achieved on large knowledge bases through the construction of a summary ABox $\mathcal{A}'$ corresponding to the ABox $\mathcal{A}$. Intuitively, an individual in $\mathcal{A}'$ represents a set of individuals in $\mathcal{A}$ which have some common semantically relevant properties (*e.g.,* they are members of the same explicit concepts). Formally, an ABox $\mathcal{A}'$ is a summary ABox of ABox $\mathcal{A}$ if there is a mapping function $\mathbf{f}$ that satisfies the following constraints:

(1) if $C(a) \in \mathcal{A}$ then $C(\mathbf{f}(a)) \in \mathcal{A}'$

(2) if $R(a, b) \in \mathcal{A}$ then $R(\mathbf{f}(a), \mathbf{f}(b)) \in \mathcal{A}'$ [2]

(3) if $a \dot{\neq} b \in \mathcal{A}$ then $\mathbf{f}(a) \dot{\neq} \mathbf{f}(b) \in \mathcal{A}'$

For a subset $\mathcal{S}$ of a summary ABox $\mathcal{A}'$, we define the preimage of $\mathcal{S}$, denoted $\mathbf{f}^{-1}[\mathcal{S}]$, as the following subset of $\mathcal{A}$: $\mathbf{f}^{-1}[\mathcal{S}] = \mathcal{A} \bigcap (\{R(a, b) | R(m, n) \in \mathcal{S} \wedge m = \mathbf{f}(a) \wedge n = \mathbf{f}(b)\} \cup \{C(a) | C(m) \in \mathcal{S} \wedge m = \mathbf{f}(a)\} \cup \{a \dot{\neq} b | m \dot{\neq} n \in \mathcal{S} \wedge m = \mathbf{f}(a) \wedge n = \mathbf{f}(b)\})$.

If the summary ABox $\mathcal{A}'$ obtained by applying the mapping function $\mathbf{f}$ to $\mathcal{A}$ is consistent w.r.t. a TBox $\mathcal{T}$, then $\mathcal{A}$ is consistent w.r.t. $\mathcal{T}$. However, the converse does not hold. (Dolby et al. 2007) introduced a refinement method to deal with an inconsistent summary.

[1] http://www.w3.org/TR/rdf-sparql-query/#construct

[2] This also applies to the equality relation (=)

Let $\mathcal{L}$ be a mapping from each individual in $\mathcal{A}$ to a set of concepts, such that $C(a) \in \mathcal{A}$ iff $C \in \mathcal{L}(a)$. We call $\mathcal{L}(a)$ the *concept set* of a. In a centralized setting, a summary ABox that satisfies properties (1)-(3) can be effeciently built by (a) mapping individuals with the same concept set CS to the same summary individual n whose concept set is CS and (b) adding relations between n and other summary individuals to satisfy (2) and (3). In practice, such summary ABox $\mathcal{A}'$ is dramatically smaller than the original ABox $\mathcal{A}$. It can be constructed efficiently from $\mathcal{A}$ using conventional relational database queries. It only needs to be computed once, persisted, and then reused in subsequent queries. It is easily updated incrementally and is thus resilient to changes in $\mathcal{A}$. Unfortunately, this simple summarization technique is impractical in a decentralized setting since it requires complete explicit information for each individual.

Distributed Summary

In a decentralized setting, an abox $\mathcal{A}$ is spread to n sources s_1 to s_n, with $\mathcal{A}_i$ denoting the portion of the abox $\mathcal{A}$ located in s_i, and $\mathcal{A} = \bigcup_{1 \leq i \leq n} \mathcal{A}_i$. Conceptually, the construction of the global summary ABox $\mathcal{A}'$ of a distributed ABox $\mathcal{A}$ consists of two steps: (a) construction of local summaries and (b) the merge of the local summaries. In what follows, we assume the existence of a global hash function $\mathbf{h}$ which maps an individual a in a source s_i to its hash value $\mathbf{h}(a)$, as well as an owner function $\mathbf{Ow}$ that maps a hash value hv computed by $\mathbf{h}$ to a source s_i considered as the unique 'owner' of individuals whose hash value is hv. Specific implementations of $\mathbf{h}$ and $\mathbf{Ow}$ are defined on Page 4.

Constructing Local Summaries At each source s_i, a local summary ABox $\mathcal{A}'_i$ of $\mathcal{A}_i$ is built by mapping individuals with the same concept set CS in $\mathcal{A}_i$ and the same hash value hv to the same local summary node n in $\mathcal{A}'_i$ whose concept set $\mathcal{L}(n) = CS$ and hash value $\mathbf{h}(n) = hv$ (if such a node n does not exist in $\mathcal{A}'_i$, a new node n is first created in $\mathcal{A}'_i$ with $\mathcal{L}(n) = CS$ and hash value $\mathbf{h}(n) = hv$). As in the traditional summary construction described in the previous section, n is then related to other summary nodes in $\mathcal{A}'_i$ to preserve role, equality and different-from assertions (*i.e.,* satisfy properties (2) and (3) of a summary ABox). Let $\mathbf{f}_i$ denote the summary function of the summary ABox $\mathcal{A}'_i$.

Merging Local Summaries The main challenge of the merger phase is illustrated by a simple example: if an individual a is present in the ABox $\mathcal{A}_1$ of source s_1, where it is mapped to the node n_1 in the local summary $\mathcal{A}'_1$, and in the ABox $\mathcal{A}_2$ of the source s_2, where it is mapped to the node n_2 in $\mathcal{A}'_2$, the merger phase must ensure that, in the final merged summary, a is mapped to a single summary node which has the same role, concept, equality and different-from assertions as both n_1 and n_2. To achieve this goal, we rely on the hash function $\mathbf{h}$ to identify summary nodes in

Algorithm 1 Building the global summary.

1: **procedure** BUILDGLOBALSUMMARY(S)
2: $N \leftarrow \emptyset$
3: **for all** $s_i \in S$ **do**
4: $\mathcal{A}'_i \leftarrow$ BUILDLOCALSUMMARY(s_i)
5: **for all** $n \in \mathcal{A}'_i | \mathbf{Ow}(\mathbf{h}(n)) \neq s_i$ **do**
6: $s_{ow} \leftarrow \mathcal{A}'_{ow} | s_{ow} = \mathbf{Ow}(\mathbf{h}(n))$
7: $N \leftarrow N \cup \langle n, s_i, s_{ow} \rangle$
8: **for all** $\langle n, s_i, s_{ow} \rangle \in N$ **do**
9: $M \leftarrow \{m | m \in \mathcal{A}'_{ow} \wedge \mathbf{h}(n) = \mathbf{h}(m)\}$
10: **for all** $(\alpha \in \mathcal{A}'_i | n$ appears in $\alpha) \wedge m \in M$ **do**
11: $\mathcal{A}'_i \leftarrow (\mathcal{A}'_i / \{\alpha\}) \cup \{\alpha[n \rightarrow m]\}$
$\alpha[n \rightarrow m]$ is the assertion obtained after replacing n by m in α
12: **return** $\bigcup_{1 \leq i \leq n} \mathcal{A}'_i$

different local summaries whose set of corresponding ABox individuals might overlap (*e.g.,*, n_1 and n_2 in the previous example, which, by construction of local summaries, will have the same hash value). Moreover, we use the **Ow** function to ensure a unique representative of each ABox individual in the final merged summary. Although an individual a may be mentioned in various sources, the intuition is that its most authoritative description is provided by the source which owns it (*i.e.,* $\mathbf{Ow}(\mathbf{h}(a))$). The distributed global summary ABox $\mathcal{A}'$ is built from the local summaries according to Algorithm 1. This algorithm takes as a parameter the set of individual sources S, and assumes a function that builds local summaries according to the previous section. Informally, the distributed summary ABox $\mathcal{A}'$ is built from retrieved local summaries $\mathcal{A}'_i$ by replacing, in each $\mathcal{A}'_i$, a 'foreign' summary node n (*i.e.,* a node such that $\mathbf{Ow}(\mathbf{h}(n)) \neq s_i$) by all summary nodes m in the local summary of the source $\mathbf{Ow}(\mathbf{h}(n))$ having the same hash value as n (*i.e.,* m is a node in the local summary of $\mathbf{Ow}(\mathbf{h}(n))$ and $\mathbf{h}(n) = \mathbf{h}(m)$ [3]).

Distributed Summary Function For a source s_i, let $\mathcal{A}''_i$ denote the summary obtained after replacing all "foreign" nodes in the local summary retrieved from s_i (*i.e.,* $\mathcal{A}''_i$ is the value of $\mathcal{A}'_i$ at line 12 in Algorithm 1). The summary function $\mathbf{f}$ of $\mathcal{A}'$ maps an individual a in $\mathcal{A}$ to a node n in $\mathcal{A}''_i$ such that a is owned by source s_i (*i.e.,* $s_i = \mathbf{Ow}(\mathbf{h}(a))$) and $\mathbf{h}(n) = \mathbf{h}(a)$ (by construction, at least one such n exists. In case of multiple n, if a is in $\mathcal{A}_i$, then $\mathbf{f}_i(a)$ is chosen; otherwise, any such n can be chosen). In practice, $\mathbf{f}$ is not computed as it is not needed for any optimization technique described in the paper. It can easily be shown that this summarization scheme satisfies the three fundamental properties of a summary ABox.

Summary Filtering

For a given query or reasoning task, only a small fraction of the distributed summary is typically relevant. As a result, in practice, we never perform the merger step of the distributed summarization directly on the local summaries, but rather on their relevant subsets. In the remainder of this section, we consider filtering w.r.t. conjunctive query answering with a shared aligned DL-Lite$_{\mathcal{R}}$ TBox. Since conjunctive query answering w.r.t. a DL-Lite$_{\mathcal{R}}$ TBox can be reduced to answering a union of conjunctive queries w.r.t. an empty TBox, we present only summary filering w.r.t. a conjunctive query and an empty TBox.

We assume that all the local summaries have been pulled in a central location. For a conjunctive query q, the filtering is done by relaxing q to account for the fact that joins in q are not necessarily local to each local summary. For instance, the query $q = (x) \leftarrow R(x, y) \wedge S(x, a) \wedge C(x)$ (where R and S are roles, C is a concept, a is an individual, and x and y are variables) is transformed into the relaxed construct query $[q]_r$:

$$\mathsf{construct}[R(x_1, y_1), S(x_2, val_{a_1}), C(x_3)] \leftarrow$$
$$R(x_1, y_1) \wedge S(x_2, val_{a_1}) \wedge C(x_3) \wedge$$
$$\mathbf{hr}(x_1, hv_x) \wedge \mathbf{hr}(x_2, hv_x) \wedge \mathbf{hr}(x_3, hv_x) \wedge \mathbf{hr}(val_{a_1}, \mathbf{h}(a)) \wedge$$
$$\mathbf{src}(x_1, s_{x_1}) \wedge \mathbf{src}(x_2, s_{x_2}) \wedge \mathbf{src}(x_3, s_{x_3}) \wedge \mathbf{src}(y_1, s_{y_1}) \wedge$$
$$\mathsf{filter}(s_{x_1} \neq s_{x_2} \vee x_1 = x_2) \wedge \mathsf{filter}(s_{x_2} \neq s_{x_3} \vee x_2 = x_3) \wedge$$
$$\mathsf{filter}(s_{x_1} \neq s_{x_3} \vee x_1 = x_3)$$

Evaluating $[q]_r$ on the union of local summaries retrieves only relevant statements from each local summary. We make three important observations about $[q]_r$ in this example. First, each join variable x in q (*i.e.,* a variable that occurs at least twice in the body of the query) is replaced by new variables x_i for each occurrence of x. Second, these new variables x_i must have the same hash value hv_x to allow joins across local summaries as long as the summary nodes share the same hash value (as they potentially represent the same ABox individual). These constraints appear in the second line of the body of the query. **hr** in $\mathbf{hr}(x_1, hv_x)$ corresponds to a role that indicates the hash value of an individual. However, if x_i and x_j come from the same source, then they must also be identical since an individual in a local ABox cannot be mapped to two distinct nodes in the local summary. This constraint is enforced by the boolean filter tests in $[q]_r$ (**src** is a role that indicates the source of a node). Finally, each occurrence of a constant (*e.g.,* a) is replaced by a new variable (*e.g.,* val_{a_1}), referred to as a constant variable. This new variable is then constrained to have the same hash value as the constant (*e.g.,* $\mathbf{hr}(val_{a_1}, \mathbf{h}(a))$).

Formally, the relaxation operator $[.]_r$ is defined inductively as follows (we use the auxillary function **con** to specify constraints on the hash value of nodes bound to variables in $[q]_r$):

- For the k^{th} occurrence of a variable x, denoted x_k (hereafter referred to as an occurrence-annotated variable): $[x_k]_r = x_k$ and $\mathbf{con}(x_k) = \{\mathbf{hr}(x_k, hv_x)\}$. hv_x is a variable whose values represent the hash values of summary nodes bound to the variable x.

[3] if no such m exists in $\mathbf{Ow}(\mathbf{h}(n))$, a new one is created with $\mathbf{h}(m)$ set to $\mathbf{h}(n)$

- For the k^{th} occurrence of a constant a, denoted a_k $[a_k]_r = val_{a_k}$ and $\mathbf{con}(a_k) = \{\mathbf{hr}([a_k]_r, \mathbf{h}(a))\}$. val_{a_k} is a variable, called constant variable, representing the k^{th} occurrence of a.
- For a term t of the form $R(v, w)$ where R is a role, v and w are occurrence-annotated variables or constants, we define auxillary functions: $[t]^b_r$ to represent its contribution to the body clause of the final query, and $[t]^c_r$ for its contribution to the construct clause:
  - $[R(v,w)]^b_r = \{R([v]_r, [w]_r)\} \cup \mathbf{con}(v) \cup \mathbf{con}(w) \cup \{\mathbf{src}([v]_r, s_{[v]_r}), \mathbf{src}([w]_r, s_{[w]_r})\}$
  - $[R(v,w)]^c_r = [R(v,w)]^b_r \cup \{\mathbf{var}([v]_r, \text{“}[v]_r\text{”}), \mathbf{var}([w]_r, \text{“}[w]_r\text{”})\}$. **src** is a role that indicates the source of a summary node. It is specified for each summary node at summary construction. $s_{[v]_r}$ (*e.g.,* s_{x_1}) is a new variable for the source of the summary node bound to the new variable $[v]_r$ (*e.g.,* x_1). Finally, **var** is a role in the filtered summary that indicates the occurrence-annotated or constant variable to which each filtered summary node is bound. It plays a key role in the subsequent analysis of the filtered summary (see page 5).
- For a term t of the form $C(v)$, it is handled in a similar fashion as the previous case (*i.e.,* $R(u, v)$).
- Finally, for a conjunctive query q of the form $x_1, ..., x_n \leftarrow t_1 \wedge ... \wedge t_m$, the relaxed query $[q]_r$ used to construct the filtered local summaries is :

  $$\mathsf{construct}[\, c_1. \dots c_p.\,] \leftarrow b_1 \wedge ... \wedge b_q \wedge fl_1 \wedge ... \wedge fl_r$$

  where $c_i \in \bigcup_{1\le j\le m}[t'_j]^c_r$, $b_i \in \bigcup_{1\le j\le m}[t'_j]^b_r$ (t'_j is the term obtained after replacing variables occurring in t_i by their corresponding occurrence-annotated variable), and $fl_i \in \bigcup_{x\in \mathsf{joinVar}(q)}\{\mathsf{filter}(s_{x_j} \neq s_{x_k} \vee x_j = x_k) | 1 \le j < k \le \mathsf{occ}(x,q)\}$ or $fl_i \in \bigcup_{c\in \mathsf{joinConst}(q)}\{\mathsf{filter}(s_{val_{c_j}} \neq s_{val_{c_k}} \vee val_{c_j} = val_{c_k}) | 1 \le j < k \le \mathsf{occ}(c,q)\}$ ($\mathsf{joinVar}(q)$ is the set of join variables in q, $\mathsf{joinConst}(q)$ is the set of constants appearing at least twice in q, and $\mathsf{occ}(u,q)$ is the number of occurrence of u in the body of q) .

The correctness of filtering relies on the following Theorem whose proof is provided in the appendix:

Theorem 1 *Let $\mathcal{A}'_i$ be the local summary of a local ABox $\mathcal{A}_i$ located at source s_i with summary function $\mathbf{f}_i$, for $1 \le i \le n$ and $n > 0$. For a conjunctive query q, let $\mathbf{filter}(\mathcal{A}'_i, q)$ denote the largest subset of $\mathcal{A}'_i$ contained in the result of evaluating $[q]_r$ on $\bigcup_{1\le i\le n}\mathcal{A}'_i$. Then, the evaluation of q w.r.t. an empty TBox produces the same results on $\mathcal{A} = \bigcup_{1\le i\le n}\mathcal{A}_i$ and on $\bigcup_{1\le i\le n}{\mathbf{f}_i}^{-1}[\mathbf{filter}(\mathcal{A}'_i, q)]$*

Furthermore, if q is minimal w.r.t. the number of its terms (*i.e.,* if a conjunctive query q' is equivalent to q w.r.t. the empty TBox, then either q' has more terms than q or there are syntactically identical after variable renaming and term reordering), then the filtering performed by evaluating $[q]_r$ is optimal. This is formalized by the Theorem 2 whose proof is in Appendix.

Notation 1 *For a summary ABox $\mathcal{S}$, $\overline{\mathcal{S}}$ denotes the subset of $\mathcal{S}$ obtained after removing relations involving metadata roles (i.e.,* **src**, **hr**, **var***).*

Theorem 2 *Let $\mathcal{A}'_i$ be the local summary ABox located at source s_i, for $1 \le i \le n$ and $n > 0$. Let $\mathcal{S}'$ be a strict subset of $\overline{\bigcup_{1\le i\le n}\mathbf{filter}(\mathcal{A}'_i, q)}$. If q is minimal w.r.t. the number of its terms, then there exists n ABoxes $\mathcal{S}_i$ ($1 \le i \le n$) such that (1) $\mathcal{A}'_i$ is a valid summary ABox for $\mathcal{S}_i$ with summary function $\mathbf{g}_i$ and (2) the evaluation of q w.r.t. an empty TBox returns different set of results on $\bigcup_{1\le i\le n}{\mathbf{g}_i}^{-1}[\mathcal{S}' \cap \mathcal{A}'_i]$ and on $\bigcup_{1\le i\le n}\mathcal{S}_i$*

In practice, for performance reasons and since access to some remote local summaries may be restricted to a SPARQL end point, before issuing the query $[q]_r$ against $\bigcup_{1\le i\le n}\mathcal{A}'_i$, we start by retrieving, for each term t in q, the relevant portion of each local summary $\mathcal{A}'_i$. This is achieved by evaluating $[() \leftarrow t']_r$ against $\mathcal{A}'_i$, where t' is the term obtained after replacing variables occurring in t by their corresponding occurrence-annotated variable.

A Meaningful Global Hash for Linked Data

The global hash function **h** and the owner function **Ow** play a key role in the distributed summary. We first briefly describe some important desiderata on **h** and **Ow**. The number of buckets created by the hashing obviously controls the trade-off between precision of the global distributed summary and its size. However, the decision on the number of buckets should not be made upfront and at a global level (Desideratum-1). Instead, this decision should be made, without coordination, by local sources as they build their local summary. For example, one source with a large and very complex data set might require a more aggressive hashing (fewer buckets), while another source might choose a less aggressive hashing because it still produces a relatively small summary of its ABox. On the other hand, a normalization mechanism is needed when merging the relevant sections of the local summaries to compensate for the difference in “aggressiveness” of hashing between sources (Desideratum-2). Finally, the owner function **Ow** should be able to interpret hash value of an individual a to identify the source which is more likely to have the most authoritative information about a (Desideratum-3).

If the identifiers of individuals and the distribution of data in the network of sources are completely random, it is unclear how such **h** and **Ow** can be designed to satisfy the previous three desirata. Fortunately, this is not how the Linked Open Data is organized. The first three of its four principles outlined by Tim Berners-Lee[4] can be exploited to design

[4] http://www.w3.org/DesignIssues/LinkedData

h and **Ow**: "(i) Use URIs to identify things; (ii) Use HTTP URIs so that these things can be referred to and looked up by people and user agents; (iii) Provide useful information about the thing when its URI is dereferenced". These principles clearly entail a notion of ownership of each individual; namely, the domain (or host name) of the HTTP URI of the individual is its owner. A HTTP URI has syntax *http://P_0/.../P_n/name* where P_0 is domain, $\{P_1, \ldots, P_n\}$ are path elements, and *name* is local name. We can leverage URI syntax to create desirable **h** and **Ow** functions. Function **h** can be defined by removing some elements of URIs. That is, given the level of abstraction l, **h**(*http://P_0/.../P_n/name*) = *http://P_0/.../P_x*, where $x = max(n - l, 0)$. Here, the hash value of a URI serves as a bucket of URIs with common domain and path elements. The size of the bucket increases and hashing becomes more aggressive as l increases. For instance, **h**(*http://dbpedia.org/resource/Bill_Clinton*) is *http://dbpedia.org/resource* when $l = 0$, but it becomes *http://dbpedia.org* when $l = 1$. Similarly, following Linked Open Data principles, **Ow** can be defined as **Ow**(*http://P_0/.../P_x*)= P_0.

To hash URIs in a specific domain, two different sources s_i and s_j may select different l values, such as l_i and l_j. This means that the same URI will be represented by different hash values in their local summaries. Therefore, in the meta-data of each local summary, sources state l values they used for each domain. While aggregating local summaries, hash values are normalised easily by taking the maximum level of abstraction. For instance, if $l_i = 0$ and $l_j = 1$ for *dbpedia.org* domain, then normalized value for **h**(*http://dbpedia.org/resource/Bill_Clinton*) would be *http://dbpedia.org* in the global summary.

Summary-based Optimizations

The filtered and annotated (with **src**, **hr**, **var**) local summaries enable three important types of optimizations: query pruning, efficient source selection, and transformation of distributed joins into local joins (even when multiple sources are selected).

Query Pruning

For a conjunctive query q, if the evaluation of $[q]_r$ on the local summaries $\mathcal{A}'_i$ of $\mathcal{A}_i$ $(1 \leq i \leq n)$ results in an empty summary, then, by Theorem 1, the evaluation of q w.r.t. the empty Tbox on $\mathcal{A} = \bigcup_{1 \leq i \leq n} \mathcal{A}_i$ is guaranteed to also return an empty result set. Query pruning is particularly important for conjunctive queries evaluated w.r.t. to a DL-Lite$_\mathcal{R}$ Tbox because, as shown in the experimental evaluation, many generated conjunctive queries can be discarded - including queries that cannot be pruned based only on the unsatisfiability of one of their terms. Furthermore, the optimality of the filtering (see Theorem 2) makes our approach more likely to detect queries with empty result set.

Source Selection

Source selection consists in assigning to each term t in a query q, the set of sources, denoted **srcsel**(t), that need to be contacted in order to answer t. Assuming that t is of the form $R(v, w)$ ($C(v)$ is treated in a similar fashion), by definition of $[.]_r$, $[v]_r$ is always a variable x (of the form val_v if v is a constant; otherwise, it is of the form x_k). In our approach, source selection is performed by simply evaluating the following conjunctive query on the filtered summary:

$$(s) \leftarrow \mathbf{var}(u, \text{“}x\text{”}) \wedge \mathbf{src}(u, s)$$

This query selects the sources of all the filtered summary nodes which have a variable annotation (**var**) to the variable x ($x = [v]_r$). The correctness of our source selection stems from the fact that the construct query used to build the filtered summary adds the triple **var**(n, "x") for all constructed nodes n bound to x.

Distributed Join Elimination

We now present two techniques to transform distributed joins into local joins. The first technique is applicable to any system with some source selection capability, whereas the second is unique to our approach.

Exclusive source-based technique Given a conjunctive query q and a join variable x appearing in n terms of q $(n > 1)$ such that m $(1 < m \leq n)$ of these terms have the same unique source s, an expensive distributed join can be avoided by performing the m join locally on the source s. This simple technique works fairly well as discussed in (Schwarte et al. 2011) when the sources use different vocabularies or ontologies. However, it is less effective when they have the same ontology and most subjects and objects in terms of q are variables. In such situations, as illustrated on the UOBM 30 dataset in the experimental evaluation, almost all sources are selected.

Variable Locality-based technique In distributed settings where a common vocabulary or ontology is shared by many sources, some roles exhibit a local behavior; that is, in each source s, they are only involved in statements where the subject a (or the object a) is an individual owned by s (*i.e.,* $\mathbf{Ow}(\mathbf{h}(a)) = s$)). For example, in the distributed data network of a multinational coorporation where each data source contains information for each country, the role "salary" and "position" would appear in each source. However, in each source s, the subject of a statement with such a role is always an employee (*e.g.,* $http : //xyz.com/France/HR/Jean\_Dupond$) working in the country corresponding to s . Therefore, the conjunctive query $q = (x, p, s) \leftarrow salary(x, s) \wedge position(x, p) \wedge s > 200K$ can more efficiently be evaluated by computing it locally at each source and returning the union of the local results - thus avoiding an expensive distributed join.

Annotations in the filtered summary allow us to detect this locality. Let x be a join variable which appears in n ($n > 1$) terms of a query q. For a subset $S = \{t_1, ..., t_m\}$ (with $1 < m \leq n$), x can be identified as local for the join (t_1, ..., t_m) if all filtered summary nodes with the same hash value bound to any occurrence of x in a t_i come from the same source. Formally, for $1 \leq i \leq m$, if terms t_i satisfy the following property (Var-Locality), then the join on x for the all the terms t_i ($1 \leq i \leq m$) can safely be performed locally in each source: (**Var-Locality**) If two distinct nodes α and β in the filtered summary are such that $\mathbf{var}(\alpha, \text{“}x_i\text{”})$, $\mathbf{var}(\beta, \text{“}x_j\text{”})$, $\mathbf{hr}(\alpha, hv)$ and $\mathbf{hr}(\beta, hv)$ are statements in the filtered summary (where x_i and x_j are any occurrence-annotated variables corresponding to occurrences of x in $\{t_1, ..., t_m\}$, and hv denotes the common hash value of α and β), then α and β must come from the same source (*i.e.,* $\mathbf{src}(\alpha, sc)$ and $\mathbf{var}(\beta, sc)$ must be in the filtered summary, where sc is the value of the common source).

Experimental Evaluation

To evaluate the effectiveness of our approach, we developed two experiments to assess the efficiency gains over existing state-of-the-art distributed query answering engines, such as Alibaba and FedX (Schwarte et al. 2011), using our summary-based optimizations. [5]

The first set of experiments consists of a subset of the FedBench (Schmidt et al. 2011) benchmark and aims to compare our approach to the FedX and Alibaba. FedBench benchmarks consist of sets of queries issued to a variety of collections of data sources, with queries spanning multiple individual sources to require a federated query answering mechanism. For example, the “Cross Domain” collection of datasets includes DBpedia (Bizer et al. 2009), the New York Times ontology, LinkedMDB (Consens 2008), Jamendo, Geonames and the SW Dog ontology (an ontology describing academic conferences). In our experiments, we use only the conjunctive queries from FedBench for both the “Life Sciences” dataset (LifeSci), as well as a modified version of the Cross Domain dataset called “Open Domain” (OpenDom). Within OpenDom the SW Dog ontology's ABox is broken down into multiple sources, one for each individual conference, resulting in 36 different sources rather than the six sources from FedBench. Moreover, the set of queries for OpenDom is divided into cross-domain and linked data (OpenDom-LD).

The second set of experiments consists of a series of federated queries over the UOBM benchmark (Ma et al. 2006).UOBM is an improvement over the popular LUBM (Guo, Pan, and Heflin 2005) adding the ability to scale the size of the benchmark almost indefinitely, but critically, it adds multiple links between universities. Since we use one data source per university, inter-university links lead to some federated queries requiring joins across data sources. In our benchmark, we have transformed the UOBM ontology into $\mathsf{DL\text{-}Lite}_\mathcal{R}$ expressivity, and generated two datasets with, respectively, five (UOBM5) and 30 universities (UOBM30). Using the $\mathsf{DL\text{-}Lite}_\mathcal{R}$ query compilation described in (Rosati and Almatelli 2010) [6], the UOBM queries were translated into 483 conjunctive queries w.r.t an empty TBox (the compilation step also prunes out generated queries with concept or role not present in the sources).

Table 1: Number of assertions in summary and number of sources by domain — ‡ indicates small-sized sources.

Domain	LifeSci	OpenDom	UOBM5	UOBM30
Summary Size	33506	60041	97956	702206
Number of Sources	4	6+30‡	5	30
$\frac{\text{Summary Size}}{\text{Source Data Size}}$	0.063 %	0.037%	8.8%	9.2 %

The federation engines were run on a laptop with a dual-core 2.4Ghz Intel CPU and 3GB of RAM (1GB maximum heap size for the Java Virtual Machine) running Windows XP. The federation engines connected to a remote HTTP Sesame RDF server with four 2.33GHz 64-bit Intel CPUs and 25 GB of RAM running Linux. The size ratio of summary to data sources varied significantly, depending on the number of similar concepts in each source, ranging from 0.037% for OpenDom to 9.2% for UOBM30. The size of the summaries and the number of sources for each dataset is shown in Table 1.

The results of these experiments are summarized in Table 2, which shows runtime statistics for each combination of dataset and federation engine (with and without our summary-based optimizations (SO)). The results show that the cost of analyzing the summary is amortized by the gains obtained from more efficient query plans in all datasets and engines, except for FedX on LifeSci, which has the smallest number of sources. Moreover, the improvements for UOBM are much more dramatic as the number of sources increases due to our unique “variable locality-based technique” to eliminate expensive distributed joins, a more efficient source selection and query pruning (13% of the queries are pruned without contacting any source). For UOBM30, we observe a four times average speed gain on FedX (Alibaba without our optimizations on UOBM30 did not complete after 2.5 days).

Conclusions

In this paper, we developed a novel distributed summarization approach for efficiently querying Linked Open Data. Our approach exploits the main principles of Linked Data for indexing distributed data. We performed extensive experiments over real and synthetic data and show that our approach improves state-of-the-art query engines for distributed semantic data in $\mathsf{DL\text{-}Lite}_\mathcal{R}$ ontologies. In future work, we will extend our approach for more expressive ontology languages.

[5] See (Schmidt et al. 2011; Dolby et al. 2007) for additional information on the datasets used.

[6] This compilation technique produces a non recursive datalog which we translate into a set a conjunctive queries

Table 2: Querying Times (sec) — † did not fully complete.

Engine	Dataset	Average	St. Dev	Range
FedX	UOBM5	8.362	10.67	0-179.89
FedX+SO	UOBM5	5.11	7.06	0.02-51.14
Alibaba †	UOBM30	days	-	-
Alibaba+SO	UOBM30	51.30	141.82	0-943.80
FedX	UOBM30	114.22	120.84	0-679.97
FedX+SO	UOBM30	25.39	57.11	0-603.13
Alibaba †	OpenDom	213.50	205.43	44.89-510.92
Alibaba+SO	OpenDom	76.36	73.29	18.73-186.58
FedX	OpenDom	26.70	14.80	9.17-44.67
FedX+SO	OpenDom	10.08	8.34	3.45-25.84
Alibaba †	OpenDom-LD	520.99	351.96	37.69-940.72
Alibaba+SO	OpenDom-LD	420.47	441.48	12.38-930.19
FedX	OpenDom-LD	18.34	13.12	7.74-46.72
FedX+SO	OpenDom-LD	15.08	13.24	0.02-45.28
Alibaba	LifeSci	1040.73	99.38	882.84-1125.08
Alibaba+SO	LifeSci	576.66	367.14	125.53-930.64
FedX	LifeSci	17.64	16.91	2.97-45.81
FedX+SO	LifeSci	19.23	23.03	3.67-59.23

References

Aroyo, L.; Welty, C.; Alani, H.; Taylor, J.; Bernstein, A.; Kagal, L.; Noy, N. F.; and Blomqvist, E., eds. 2011. *The Semantic Web - ISWC 2011 - 10th International Semantic Web Conference, Bonn, Germany, October 23-27, 2011, Proceedings, Part I*, volume 7031 of *Lecture Notes in Computer Science*. Springer.

Baader, F.; McGuiness, D. L.; Nardi, D.; and Patel-Schneider, P., eds. 2002. *Description Logic Handbook: Theory, implementation and applications*. Cambridge University Press.

Bizer, C.; Lehmann, J.; Kobilarov, G.; Auer, S.; Becker, C.; Cyganiak, R.; and Hellmann, S. 2009. Dbpedia - a crystallization point for the web of data. *Web Semant.* 7:154–165.

Calvanese, D.; Giacomo, G.; Lembo, D.; Lenzerini, M.; and Rosati, R. 2007. Tractable reasoning and efficient query answering in description logics: The dl-lite family. *J. Autom. Reason.* 39:385–429.

Consens, M. P. 2008. Managing linked data on the web: The linkedmdb showcase. In Baeza-Yates, R. A.; Jr., W. M.; and Santos, L. A. O., eds., *LA-WEB*, 1–2. IEEE Computer Society.

Dolby, J.; Fokoue, A.; Kalyanpur, A.; Kershenbaum, A.; Schonberg, E.; Srinivas, K.; and Ma, L. 2007. Scalable semantic retrieval through summarization and refinement. In *Proceedings of the 22nd AAAI Conference on Artificial intelligence*.

Fokoue, A.; Kershenbaum, A.; Ma, L.; Schonberg, E.; and Srinivas, K. 2006. The summary abox: Cutting ontologies down to size. In *Proceedings of the International Semantic Web Conference (ISWC)*, 343–356.

Guo, Y.; Pan, Z.; and Heflin, J. 2005. Lubm: A benchmark for owl knowledge base systems. *Web Semant.* 3:158–182.

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.

Hartig, O.; Bizer, C.; and Freytag, J.-C. 2009. Executing sparql queries over the web of linked data. In *Proceedings of the International Semantic Web Conference (ISWC)*, 293–309.

Konrath, M.; Gottron, T.; and Scherp, A. 2011. Schemexweb-scale indexed schema extraction. In *Proceedings of the International Semantic Web Conference (ISWC) - Winner of the Billion Triple Challenge.*

Ma, L.; Yang, Y.; Qiu, Z.; Xie, G.; and Pan, Y. 2006. Towards a complete owl ontology benchmark. In *Proc. of the third European Semantic Web Conf.(ESWC 2006)*, 124–139.

Rosati, R., and Almatelli, A. 2010. Improving query answering over dl-lite ontologies. In *KR*.

Schmidt, M.; Görlitz, O.; Haase, P.; Ladwig, G.; Schwarte, A.; and Tran, T. 2011. Fedbench: A benchmark suite for federated semantic data query processing. In Aroyo et al. (2011), 585–600.

Schwarte, A.; Haase, P.; Hose, K.; Schenkel, R.; and Schmidt, M. 2011. Fedx: Optimization techniques for federated query processing on linked data. In Aroyo et al. (2011), 601–616.

Appendix

Preliminaries

Let q be a conjunctive query of the form

$$q = (x^1, ..., x^r) \leftarrow t_1 \wedge ... \wedge t_m$$

Let $X = (x^1, ..., x^r)$ denote the r-vector made of result variables of q. Let $Y = (y^1, ..., y^p)$ denote the p-vector of non-result variables (*i.e.,* variables occuring in the body of q and different from all x^i). By definition (see (Calvanese et al. 2007)), an answer π to q w.r.t. the empty TBox on an ABox $\mathcal{A}$ is a mapping from variables x^i ($1 \leq i \leq r$) to individuals in $\mathcal{A}$ such that there exists a mapping π' from variables y^j ($1 \leq j \leq p$) to individuals in $\mathcal{A}$ such that $\{\pi[\pi'[t_j]] | 1 \leq j \leq m\} \subseteq \mathcal{A}$, where $\pi[\alpha]$ (resp. $\pi'[\alpha]$) denotes the term obtained from α after replacing each variable x^i (resp. y^j) by $\pi(x^i)$ (resp. $\pi'(y^j)$).

Proof of Theorem 1

Let $\mathcal{A}'_i$ be the local summary of a local ABox $\mathcal{A}_i$ located at source s_i with summary function $\mathbf{f}_i$, for $1 \leq i \leq n$ and $n > 0$.

By definition of the preimage (see preliminaries section on page 2) and $\mathbf{filter}(\mathcal{A}'_i, q)$ (see Theorem 1), $\mathcal{S} = \bigcup_{1 \leq i \leq n} \mathbf{f}_i{}^{-1}[\mathbf{filter}(\mathcal{A}'_i, q)]$ is a subset of $\mathcal{A} = \bigcup_{1 \leq i \leq n} \mathcal{A}_i$. Therefore, if a mapping π is an answer to q w.r.t. the empty TBox on the ABox $\mathcal{S}$, then it is obviously also an answer to q w.r.t. the empty TBox on $\mathcal{A}$. Thus, to prove Theorem 1, we only need to establish that if a mapping π is an answer to q on $\mathcal{A}$, then it is also an answer to q on $\mathcal{S}$.

Notation 2 *Let $\mathcal{A}'$ be the summary ABox of an ABox $\mathcal{A}$ with summary function* $\mathbf{f}$*. Let $\mathcal{U}$ be a subset of an ABox $\mathcal{A}$. The image of $\mathcal{U}$ in $\mathcal{A}'$, denoted $\mathbf{f}[\mathcal{U}]$, is defined as the following set:*

$$\mathbf{f}[\mathcal{U}] = \{R(\mathbf{f}(a), \mathbf{f}(b)) | R(a, b) \in \mathcal{U}\} \cup \{C(\mathbf{f}(a)) | C(a) \in \mathcal{U}\}$$
$$\cup \{\mathbf{f}(a) \dot{\neq} \mathbf{f}(b) | a \dot{\neq} b \in \mathcal{U}\}$$

It follows directly from the three key properties of a summary Abox that $\mathbf{f}[\mathcal{U}] \subseteq \mathcal{A}'$. From the definition of preimage, it follows that $\mathcal{U} \subseteq \mathbf{f}^{-1}[\mathbf{f}[\mathcal{U}]]$.

Let π be an answer to q on $\mathcal{A}$. Since π is an answer, there exists π' a mapping from y^i ($1 \leq i \leq p$) to individuals in $\mathcal{A}$ such that $\mathcal{J} = \{\pi[\pi'[t_j]] | 1 \leq j \leq m\} \subseteq \mathcal{A}$. We define by induction n subsets of $\mathcal{J}$ which constitues a partition of $\mathcal{J}$ as follows:

- $\mathcal{J}_1 = \mathcal{J} \cap \mathcal{A}_1$
- for $1 < i \leq n$, $\mathcal{J}_i = (\mathcal{J} \cap \mathcal{A}_i) - \bigcup_{1 \leq j < i} \mathcal{J}_j$

We now show that, for $1 \leq i \leq n$, $\mathbf{f}_i[\mathcal{J}_i] \subseteq \mathbf{filter}(\mathcal{A}'_i, q)$. This is enough to prove Theorem 1 because it implies $\mathcal{J}_i \subseteq \mathbf{f}_i^{-1}[\mathbf{filter}(\mathcal{A}'_i, q)]$ as a result of (a) $\mathbf{f}_i^{-1}[\mathbf{f}_i[\mathcal{J}_i]] \subseteq \mathbf{f}_i^{-1}[\mathbf{filter}(\mathcal{A}'_i, q)]$ (by monotonicity of the preimage), and (b) $\mathcal{J}_i \subseteq \mathbf{f}_i^{-1}[\mathbf{f}_i[\mathcal{J}_i]]$ (by definition of image and preimage).

Let q' be the conjunctive query whose set of result variables are all variables occurring in $[q]_r$ and whose body is made of terms in the body of $[q]_r$ except filter tests. The proof proceeds with the following steps:

1. We construct a mapping Π from variables of q' to individuals in $\mathcal{Y} = \bigcup_{1 \leq i \leq n} \mathcal{Y}_i$
$$\mathcal{Y}_i = \mathbf{f}_i[\mathcal{J}_i] \cup \{\mathbf{src}(n, s_i) | n \in \mathsf{indSet}(\mathbf{f}_i[\mathcal{J}_i])\}$$
$$\cup \{\mathbf{hr}(n, \mathbf{h}(n)) | n \in \mathsf{indSet}(\mathbf{f}_i[\mathcal{J}_i])\}$$
$\mathsf{indSet}(\mathcal{U})$ denotes the set of individuals in a subset $\mathcal{U}$ of an ABox or a summary Abox . $\mathcal{Y}_i$ adds to $\mathbf{f}_i[\mathcal{J}_i]$ metadata relations (**src** and **hr**) for each individual in $\mathbf{f}_i[\mathcal{J}_i]$.
2. We show that Π is an answer to q' on $\mathcal{Y}$ (*i.e.,* $\bigcup_{t \in \mathsf{body}(q')} \Pi[t] \subseteq \mathcal{Y}$, where $\mathsf{body}(q')$ denotes the set of terms in the body of the query q'). Furthermore, Π also satisfies all the filter test conditions in the body of $[q]_r$.
3. We show that $\bigcup_{t \in \mathsf{body}(q')} \Pi[t] = \mathcal{Y}$
4. Finally, assuming the previous three results have been established, we conclude the proof as follows. Since $\mathcal{Y}$ is clearly a subset of $\mathcal{A}' = \bigcup_{1 \leq i \leq n} \mathcal{A}'_i$ and Π also satisfies all the filter tests in the body of $[q]_r$, it follows that the result of the evaluation of $[q]_r$ on $\mathcal{A}'$ contains $\mathcal{Y}$ (because the construct part of $[q]_r$ is a super set of $\mathsf{body}(q')$ and $\mathsf{body}([q]_r) = \mathsf{body}(q') \cup \mathsf{filters\ in}\ [q]_r)$, which implies $\bigcup_{1 \leq i \leq n} \mathbf{f}_i[\mathcal{J}_i] \subseteq \bigcup_{1 \leq i \leq n} \mathbf{filter}(\mathcal{A}'_i, q)$. This result combines with the fact that, for $1 \leq i \leq n$, $\mathbf{f}_i(\mathcal{J}_i)$ (resp. $\mathbf{filter}(\mathcal{A}'_i, q)$) are mutually disjoint and $\mathbf{f}_i(\mathcal{J}_i) \subseteq \mathcal{A}'_i$ (resp. $\mathbf{filter}(\mathcal{A}'_i, q) \subseteq \mathcal{A}'_i$) establishes $\mathbf{f}_i[\mathcal{J}_i] \subseteq \mathbf{filter}(\mathcal{A}'_i, q)$.

Step 1: Specification of Π Variables in q' are in one of five forms;

- x^l_k for the k^{th} occurrence of the result variable x^l in the body of original query q, or
- y^l_k for the k^{th} occurrence of the non-result variable y^l in the body of original query q, or
- $val_{c^l_k}$ for the k^{th} occurrence of the constant c^l in the body the original query q
- s_u for the source of the nodes bound to variable u (where u is in one of the first three forms).
- hv_u for the hash value of a variable u. u is either a result variable $u = x^l$ of q or a non result variable $u = y^l$ of q.

Let z be a variable in q'. If z is in one of the first three forms (*i.e.,* x^l_k, y^l_k or $val_{c^l_k}$), then there exists a single term t_j ($1 \leq j \leq m$) in the original query q such that z appears in the term t'_j obtained from t_j after replacing variables by their corresponding occurrence-annotated variables and constants by their corresponding constant variables. By construction of $(\mathcal{J}_v)_{1 \leq v \leq n}$, there is a unique i ($1 \leq i \leq n$) such that of $\pi[\pi'[t_j]] \in \mathcal{J}_i$.

- If z is of the form x^l_k, then $\Pi(z) = \mathbf{f}_i(\pi(x^l))$
- If z is of the form y^l_k, then $\Pi(z) = \mathbf{f}_i(\pi'(y^l))$
- If z is of the form $val_{c^l_k}$, then $\Pi(z) = \mathbf{f}_i(c^l)$
- If z is of the form s_u (where u is a variable in one of the previous 3 forms with $\Pi(u) = \mathbf{f}_i(a)$ and a is an individual in $\mathcal{A}_i$), $\Pi(z) = s_i$ (*i.e.,* the source of the variable u is the i^{th} source s_i)
- If z is of the hv_{x^l} (resp. hv_{y^l}) with x^l a result variable in q (resp. y^l a non-result variable in q), $\Pi(z) = \mathbf{h}(\pi(x^l))$ (resp. $\Pi(z) = \mathbf{h}(\pi'(y^l))$)

Π is obviously a mapping from variables in q' to individuals in $\mathcal{Y}$.

Step 2: Π is an answer to q' on $\mathcal{Y}$ and satisfies filter tests in $[q]_r$ We now show that Π is an answer to q' on $\mathcal{Y}$ by establishing that for each term $t \in \mathsf{body}(q')$, $\Pi[t] \in \mathcal{Y}$. Let t be term in the body of q'. We consider all the forms in which t can be:

- *Case 1:* There exists j ($1 \leq j \leq m$) such that $t = t'_j$ where t'_j is the term obtained from the j^{th} term in the body of the original query q after replacing variables and constants by their corresponding occurrence-annotated variables and constant variables. By construction of $(\mathcal{J}_v)_{1 \leq v \leq n}$, there exists a unique i ($1 \leq i \leq n$) such that $\pi[\pi'[t_j]] \in \mathcal{J}_i$. By definition of Π, the following holds:
$$\Pi[t'_j] \in \mathbf{f}_i[\{\pi[\pi'[t_j]]\}] \quad (I)$$
(*i.e.,* $\Pi[t'_j]$ is the single element in $\mathbf{f}_i[\{\pi[\pi'[t_j]]\}]$). This implies that $\Pi[t'_j] \in \mathbf{f}_i[\mathcal{J}_i]$. So, $\Pi[t] \in \mathcal{Y}$.

- *Case 2:* $t = \mathbf{hr}(x^l_k, hv_{x^l})$. By definition of Π, $\Pi(hv_{x^l}) = \mathbf{h}(\pi(x^l))$ and there exists i $(1 \leq i \leq n)$ such that $\Pi(x^l_k) = \mathbf{f}_i(\pi(x^l))$ with $\pi(x^l) \in \mathcal{J}_i$. Hence, $\Pi[t] = \mathbf{hr}(\mathbf{f}_i(\pi(x^l)), \mathbf{h}(\pi(x^l)))$. Since $\pi(x^l) \in \mathcal{J}_i$ and $\mathbf{h}(\pi(x^l)) = \mathbf{h}(\mathbf{f}_i(\pi(x^l)))$ (by construction of local summaries), it follows that $\Pi[t] \in \{\mathbf{hr}(n, \mathbf{h}(n)) | n \in \mathsf{indSet}(\mathbf{f}_i[\mathcal{J}_i])\}$, which directly implies $\Pi[t] \in \mathcal{Y}$
- *Case 3:* $t = \mathbf{hr}(y^l_k, hv_{y^l})$. Same proof as Case 2.
- *Case 4:* $t = \mathbf{hr}(val_{c^l_k}, \mathbf{h}(c^l))$. Similar to Case 2
- *Case 5:* $t = \mathbf{src}(x^l_k, s_{x^l_k})$. By definition of Π, there exists i $(1 \leq i \leq n)$ such that $\Pi(x^l_k) = \mathbf{f}_i(\pi(x^l))$ with $\pi(x^l) \in \mathcal{J}_i$ and $\Pi(s_{x^l_k})) = s_i$. Hence, $\Pi[t] = \mathbf{src}(\mathbf{f}_i(\pi(x^l)), s_i)$, which implies $\Pi[t] \in \mathcal{Y}$
- *Case 6:* $t = \mathbf{src}(y^l_k, s_{y^l_k})$. Similar to Case 5
- *Case 7:* $t = \mathbf{src}(val_{c^l_k}, s_{val_{c^l_k}})$. Similar to Case 5

Now, we show that Π satisfies all filter tests in $[q]_r$. Let fl be a filter test in $[q]_r$.

- *Case 1:* $fl = \mathsf{filter}(s_{x^l_j} \neq s_{x^l_k} \vee x^l_j = x^l_k)$ for $1 \leq j < k \leq \mathsf{occ}(x^l, q)$. By definition of Π, there exists i_0 and i_1 between 1 and n such that $\Pi(x^l_j) = \mathbf{f}_{i_0}(\pi(x^l))$, $\Pi(x^l_k) = \mathbf{f}_{i_1}(\pi(x^l))$, $\Pi(s_{x^l_j})) = s_{i_0}$, and $\Pi(s_{x^l_k})) = s_{i_1}$. Hence, $\Pi[t] = \mathsf{filter}(s_{i_0} \neq s_{i_1} \vee \mathbf{f}_{i_0}(\pi(x^l)) = \mathbf{f}_{i_1}(\pi(x^l))$, which is trivially true as $i_0 = i_1$ implies $\mathbf{f}_{i_0}(\pi(x^l)) = \mathbf{f}_{i_1}(\pi(x^l))$
- *Case 2:* $fl = \mathsf{filter}(s_{y^l_j} \neq s_{y^l_k} \vee y^l_j = y^l_k)$ for $1 \leq j < k \leq \mathsf{occ}(y^l, q)$. Same as Case 1
- *Case 3:* $fl = \mathsf{filter}(s_{val_{c^l_j}} \neq s_{val_{c^l_k}} \vee val_{c^l_j} = val_{c^l_k})$ for $1 \leq j < k \leq \mathsf{occ}(c^l, q)$. Same as Case 1

Step 3: $\bigcup_{t \in \mathbf{body}(q')} \Pi[t] = \mathcal{Y}$ We have already shown $\bigcup_{t \in \mathsf{body}(q')} \Pi[t] \subseteq \mathcal{Y}$. $\mathcal{Y} \subseteq \bigcup_{t \in \mathsf{body}(q')} \Pi[t]$ is a direct consequence of the definition of Π and $\mathcal{Y}$ (in particular of the fact (I) established in *Case 1* of the proof that the proposition "Π is an answer to q' on $\mathcal{Y}$" in Step 2).

Proof of Theorem 2

Let $\mathcal{A}'_i$ be the local summary ABox located at source s_i, for $1 \leq i \leq n$ and $n > 0$. Let $\mathcal{S}'$ be a strict subset of $\bigcup_{1 \leq i \leq n} \overline{\mathbf{filter}(\mathcal{A}'_i, q)}$. Let q be a conjunctive query that is minimal w.r.t. its number of terms.

We consider the ABox $\mathcal{S}$ whose individuals are variables and constants in q and whose assertions are terms in the body of q. Formally, $\mathcal{S} = \{t | t \in \mathsf{body}(q)\}$. Now, we construct a distribution of of $\mathcal{S}$ in n distributed ABoxes $\mathcal{S}_i$ that satisfies the conditions (1) and (2) of Theorem 2.

Since $\mathcal{S}'$ is a strict subset of $\bigcup_{1 \leq i \leq n} \overline{\mathbf{filter}(\mathcal{A}'_i, q)}$, there exists α and i_0 $(1 \leq i_0 \leq n)$ such that $\alpha \in \overline{\mathbf{filter}(\mathcal{A}'_{i_0}, q)}$ and $\alpha \notin \mathcal{S}'$ (note: $(\overline{\mathbf{filter}(\mathcal{A}'_i, q)})_{1 \leq i \leq n}$ are mutually disjoint) .

As in the proof of Theorem 1, we introduce q' the conjunctive query whose set of result variables are all variables occurring in $[q]_r$ and whose body is made of terms in the body of $[q]_r$ except filter tests. Since $\alpha \in \overline{\mathbf{filter}(\mathcal{A}'_{i_0}, q)}$, there must exist (a) an answer Π to q' on $\mathcal{A}'$ that satisfies all the filter tests in the body to $[q]_r$, and (b) a term t_{j_0} in the body of q such that $\Pi[t'_{j_0}] = \alpha$ (t'_{j_0} denotes the term obtained from t_{j_0} after replacing variables and constants by their corresponding occurrence-annotated variables and constant variables).

Informally, $\mathcal{S}_i$ is defined as the subset of $\mathcal{S}$ made of terms t such that $\Pi[t']$ is in the summary $\mathcal{A}'_i$ of source s_i (as usual, t' denotes the term obtained from t after replacing variables and constants by their corresponding occurrence-annotated variables and constant variables). Formally, for $1 \leq i \leq n$, $\mathcal{S}_i$ is defined as follows:

$$\mathcal{S}_i = \{t | t \in \mathsf{body}(q) \wedge u \text{ appears in } t' \wedge \Pi(s_u) = s_i\}$$

$(\mathcal{S}_i)_{1 \leq i \leq n}$ are mutually disjoint because if a term t' has two variables u and v (either occurrence annotated variables (*e.g.,* x^l_k) or constant variables (*e.g.,* $val_{c^l_k}$)), then $\Pi(s_u) = \Pi(s_v)$ (by construction, a local summary contains assertions involving only its nodes). Hence is $(\mathcal{S}_i)_{1 \leq i \leq n}$ is a partition of $\mathcal{S}$.

Now, we define the summary function $\mathbf{g}_i$ which maps individual in $\mathcal{S}_i$ to a node in their summary $\mathcal{A}'_i$. Let z be an individual in $\mathcal{S}_i$. We consider all the forms in which z can be:

- *Case 1:* $z = x^l$. By definition of $\mathcal{S}_i$, there exists a term t in the body of q and k $(1 \leq k \leq \mathsf{occ}(x^l, q))$ such that x^l_k appears in t' and $\Pi(s_{x^l_k}) = s_i$, which implies that $\Pi(x^l_k) \in \mathcal{A}'_i$. $\mathbf{g}_i(x^l) = \Pi(x^l_k)$. Note that this is a proper definition of $\mathbf{g}_i(x^l)$ because if there is another qualifying k, say k', since $\Pi(s_{x^l_{k'}}) = s_i$ and Π satisfies all filter tests in $[q]_r$, we must have $\Pi(x^l_k) = \Pi(x^l_{k'})$.
- *Case 2:* $z = y^l$. Same as case 1.
- *Case 3:* $z = c^l$. Same as case 1. Important note: In the submitted version of the paper, we forgot to also include filter tests for constant variables representing the same constant. Their inclusion (as presented in this technical report) is key to ensure optimality of the filtering.

By construction of $\mathcal{S}$ (which is made of terms in the body q), the mapping π from result variables of q to individuals of $\mathcal{S}$ defined as follows is an obvious answer to q on $\mathcal{S}$: for $1 \leq l \leq r$, $\pi(x^l) = x^l$. Now, we show that q does not have any answer in $\bigcup_{1 \leq i \leq n} \mathbf{g}_i{}^{-1}[\mathcal{S}' \cap \mathcal{A}'_i]$.

Since $\alpha = \Pi[t'_{j_0}] \in \overline{\mathbf{filter}(\mathcal{A}'_{i_0}, q)}$, by definition of $(\mathcal{S}_v)_{1 \leq v \leq n}$, if there is i between 1 and n such that $t_{j_0} \in \mathcal{S}_i$, it must be $i = i_0$. Furthermore, $\alpha \notin \mathcal{S}'$ implies $t_{j_0} \notin \mathbf{g}_{i_0}{}^{-1}[\mathcal{S}' \cap \mathcal{A}'_{i_0}]$. Therefore, $\bigcup_{1 \leq i \leq n} \mathbf{g}_i{}^{-1}[\mathcal{S}' \cap \mathcal{A}'_i] \subseteq \mathcal{S}$-$\{t_{j_0}\}$. q does not have any answer in $\mathcal{S}$-$\{t_{j_0}\}$ because q is minimal w.r.t. its number of terms and $\mathcal{S}$ is made of terms of q (including t_{j_0}). Therefore, q does not have any answer in $\bigcup_{1\leq i\leq n} \mathbf{g}_i{}^{-1}[\mathcal{S}' \cap \mathcal{A}'_i]$, which concludes the proof.