A. Salatino et al. (Eds.)

doi:10.3233/SSW240026

ORAQL: An Overlap and Reliability Aware Query Processing Layer for Federations of Triple Fragment Interfaces

Tobias ZEIMETZ [a,1], Katja HOSE [b] and Ralf SCHENKEL [a]
[a] *Trier University, Germany*
[b] *Technische Universität Wien, Austria*

Abstract. The increasing numbers of available data sources have led to increased data redundancy and hence novel challenges for federations. Typically, federation engines query all endpoints that provide relevant data for a given query. However, considering the overlap, a subset of these sources might already be sufficient to obtain a complete answer. Further, we deliberately might not wish to include all sources in the evaluation and make a decision based the reliability of a source. We therefore present ORAQL (an **O**verlap and **R**eliability **A**ware **Q**uery Processing **L**ayer), an approach that exploits statistics capturing the overlap between sources to choose a subset of the available sources in the federation to compute a *complete* answer while *minimizing redundant answers*. Moreover, a user-provided reliability goal is taken into account. Hence, we propose an approach based on a majority vote over multiple sources to increase the reliability of the query result. For this work, we focus on TPF interfaces, since they are the least expressive interfaces and hence our approach can be adopted for more expressive interfaces, e.g. SPARQL endpoints. The presented methods to capture the overlap between sources of a federation have shown to generate useful overlap profiles with a maximum deviation of less than five percent. Even if the identification of redundant data is NP-hard we presented an approximation with a significant reduction in requested endpoints. Further, we have shown that ORAQL is granularly tunable towards reliability and can beat a state-of-the-art baseline system in terms of coverage and reliability.

Keywords. Overlapping Data Sources, Reliability, TPF Interface, Federation

1. Introduction

Several different Web interfaces, denoted as Linked Data Fragment (LDF) interfaces, for querying RDF graphs have been developed. These interfaces differ in their capabilities (e.g., SPARQL expressiveness) and their query costs. While classic SPARQL endpoints have the most expressiveness, they can be very costly for data providers. On the other hand, providing access to data dumps is cheap but has a clear lack of expressiveness.

The interface that generates the least server load (besides dumps) is the Triple Pattern Fragment (TPF) interface. It only has a low expressiveness, as it can receive only a single triple pattern at a time, but hence, only a comparingly small server load is gener-

[1] Corresponding Author: Tobias Zeimetz, zeimetz@uni-trier.de

ated. This facilitates the distribution and usage of TPF interfaces as shown by Hartig et al. [1]. However, the increasing numbers of available data sources and interfaces has led to increased overlap between the sources and hence novel challenges for federations.

While several works already tackle the challenges arising with LDF interface federations [1,2,3,4], the problem of increasing numbers of available data sources and thus increasing data volume is only dealt with to a limited extent. Typically, federation engines query all endpoints that provide relevant data for a query. However, considering the overlap, a subset of sources might already be sufficient to obtain a complete answer.

Another challenge that arises with an increasing number of possible endpoints is the reliability of the corresponding endpoints. Reliability plays a key role in query processing over federations. With access to more data on the Web, it is becoming increasingly important to evaluate the reliability of the data. Since different data providers have different levels of reliability regarding their data, not all sources can be trusted equally. Hence, the question often arises, what data from which sources can be trusted. However, estimating reliability is often neglected or dealt with in a trivial manner [5,6,7,8,9] by simply excluding non-reliable data sources.

Contributions: This work presents a query processing layer denoted as ORAQL (**O**verlap and **R**eliability **A**ware **Q**uery Processing **L**ayer). ORAQL consists of three main contributions: **(1)** We introduce a profile feature that provides information about the overlap between all data sources of a federation. **(2)** This overlap information is afterwards used to remove endpoints that are covered by other members of the federation. to reduce redundancy in data sources.**(3)** A user-provided reliability goal is taken into account during query processing. To this end, we extended the approach of Zeimetz et al. [10] to TPF interfaces, which uses an hierarchical agglomerative clustering to simulate a majority vote over all selected data sources. In this work, we focus on TPF interfaces, since they are the least expressive interfaces and hence, from a query processor perspective, provide more restrictions that need to be overcome.

2. Motivational Example

In the following, we illustrate the challenges and concepts behind our contributions and provide a motivating example based on a small federation and a simple SPARQL query. Note that in real-world scenarios larger federations and more complex queries are likely to be encountered but for the sake of this work we only consider simple basic graph pattern queries without filter, union, service or, other more complex expressions.

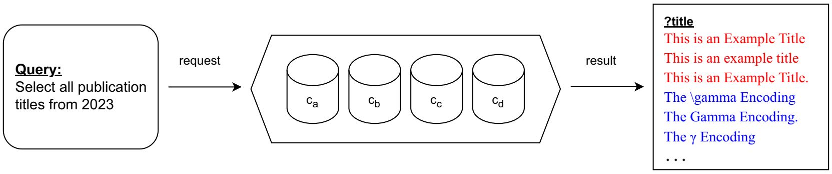


Figure 1. Example of Redundant Data in a Federation.

Scenario 1: (Redundant Data) Figure 1 shows an example federation that consists of four TPF interfaces of the scholarly domain (i.e., meta data about scientific publications). The query requests all publication titles published in 2023. Since all four interfaces are of the same domain they are likely to overlap in their data [11,12]. A traditional (TPF) query processor will filter out all interfaces that are not able to answer any triple pattern of a query. In case all four interfaces can deliver a result , hence, all four interfaces will be used during query processing. This is done regardless of whether querying only three sources would yield the same query result, leading to many queries that are not contributing new results to the query result. Hence, it makes sense to further restrict the source selection so that there is less redundancy in the query result leading not only to a decrease query time but also reduce the load for a query processor.

Scenario 2: (Overlap Degree) To accomplish this, we exploit the well-known observation [11,12] that sources for the same domain(s) overlap in their information and that those overlaps can be used for better source selection. Figure 2, part (a) presents the overlap degree of the data sources from Figure 1 used in the first scenario; since $a \cap b \cap c \cap d \neq \emptyset$ there is at least one publication whose title is included in all four sources. However, depending on the type of data (e.g., publications and authors) the data sources may overlap in different degrees. Part (b) and (c) of Figure 2 show an example for this. In the case of the class `Publication` and property `title`, a different overlap appears than for `Author` and `name`. This may be due to the fact that some data sources focus more on authors and others on publications. Further, other data sources may only store certain information, e.g., a title, an author's name and a publication year but not the name of the conference the publication was published in.

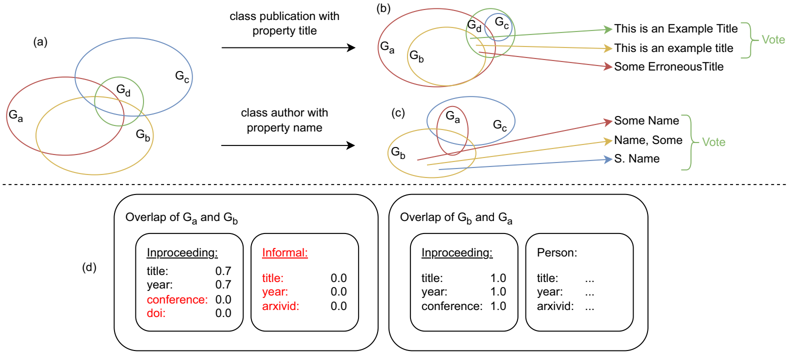


Figure 2. Example of overlapping data sources.

Based on this observation, we therefore need a fine grained overlap index that captures separate overlap information for the different data properties, such as titles and author names, and for the different classes, such as publications and authors. This information can later be used during query planning to enable a better source selection with less redundancy. For the query from Figure 1, we see that we do not have to query all four interfaces but, as shown on part (a) of Figure 2, the interfaces providing access to G_a and G_d are sufficient to obtain a complete result with minimal redundancy. If the author names are queried, the interfaces providing G_b and G_c are sufficient.

Scenario 3: (Adjustable Reliability) Another challenge is the selection of reliable sources and the degree of reliability desired by a user. It is evident that with access to more data on the Web, it is becoming increasingly important to evaluate the reliability of the data [10,9,8]. Since different data providers have different levels of reliability regarding their data, not all sources can be trusted equally. Hence, the question often arises, what data from which sources can be trusted and if it needs to be verified, for example by cross-comparison. Hence, it may be desirable in some cases to have redundancy in the data to compare them with each other and thereby perform a majority vote to increase the degree of reliability in the query result.

Further, not all users have the same reliability goal for their query result [10,6]. For example, many tasks in the digital humanities focus on processing historical data with questionable reliability, e.g., collecting data of medieval ethnic communities [13]. For such cases, the reliability may not be important to a user at first, but it is important to get a large number of results for a query (e.g., to see how an ethnic population spreads, individual errors are not important). In other cases, users might need a high degree of reliability in their data and would accept to wait longer for the query result.

3. Related Work

Source Selection: The combination of different LDF interfaces to answer SPARQL queries is focus of many publications [14,15,16,17,18,19,20]. With the development and distribution of further LDF interfaces (e.g. TPF interfaces or brTPF interfaces), new challenges arise. One challenge when using TPF interfaces is the limited expressiveness and the higher load generated on the client side. Therefore, different source selection strategies have to be developed for federations of TPF interfaces.

Heling et al. [21] proposed an approach that addresses the challenges of SPARQL query processing over federations with heterogeneous LDF interfaces (i.e., TPF interfaces, brTPF interfaces, SPARQL endpoints, etc.). Cheng et al. [1,22,2,3,4] followed a more formal approach and generalizeable approach since they cover a wider range of LDF interfaces. Additionally, they proposed a first formalization to model LDF federations and a corresponding cost-model. Even though these approaches are promising and deal above all with the wide range of possibilities of the various interfaces (i.e., level of expressiveness), they differ from our ORAQL approach in two aspects. Firstly, they do not attempt to remove redundant data sources, but only exclude data sources whose data is not included in the final query result. Secondly, these approaches do not consider the reliability of the individual data sources and the resulting query result reliability.

Furthermore, approaches for traditional federated systems [17,19,18], consisting of SPARQL endpoints, also focus on the selection of relevant sources. The system BBQ [17] focuses on selecting only relevant sources to enable an efficient query processing. They proposed an overlap-aware strategy for selecting sources for each triple pattern of a query using extended ASK operations that result in summaries in the form of Bloom filters. The goal is to achieve the same recall as an existing federation while querying fewer data sources. HiBISCuS is a system proposed by Saleem and Ngomo [19] which uses a novel type of data summaries for SPARQL endpoints that relies on the authority fragment of URIs and ASK queries. While those systems produce good results for SPARQL federations, they cannot be used for TPF interfaces since they do not support ASK queries and other aspects. Additionally, ORAQL focuses next to source selection also on reliability issues.

In addition to the source selection approaches mentioned above, some works [18,23] additionally consider characteristic sets or propose them as an extension. However, the biggest challenge with characteristic sets is the creation of the indices. Heling et al. [24]

proposed an approach based on the work of Neumann and Moerkotte [25] that estimates accurate statistical profile features based on characteristic sets based on a random sampling approach of the original dataset. They proved the usability of characteristic sets in federated systems by proposing a federated query planning which leveraged feature estimations based on characteristic sets to improve source selection. However, while characteristic sets have proven excellent for grouping triple patterns and thus querying selected endpoints, they cannot be used to predict the degree of overlap between the data sets.

As stated in many publications [11,12,26,27,28], endpoints from the same domains often overlap in their data, resulting in many redundant responses being collected. The foundation for using overlap information to implement a better source selection was already laid around 1997 by the works of Florescu et al. [11] and Vassalos et al. [12]. Florescu et al. have shown each source is categorized into one or more domains, that sources for the same domain(s) overlap in their information and that those overlaps can be used for better source selection. In contrast, Vassalos et al. outlined the challenges in utilizing overlap statistics for query answering and better source selection.

The approach of Salloum et al. [26] is based on the fact that for queries with a large number of sources it is not always possible or takes a very long time to crawl all sources. To avoid this problem, the sources are sorted according to their coverage, cost, and overlaps. This approach is more generalized, as it considers a wide variety of sources.

Reliability Computation: As baseline system we use an approach proposed by Heling and Acosta [6]. The work focuses on taking various utility aspects into account during source selection, which include aspects such as the reliability or latency of an endpoint. Their selection process considers only endpoints that satisfy the user's reliability requirements. A downside of this is that excluding unreliable data sources may not always be possible as no other sources may be available.

Next to the base line system [6], we used in our evaluation, most works considering data quality aspects like reliability as optimization goal are based in the data lake domain. Some works [5,7] consider several data-oriented quality aspects for query processing and the usage of RESTful Web APIs. Alili et al. [5] show how data lakes can be leveraged to answer user queries, taking into account the quality of the services and respecting the (time and monetary) budget set by the user. The quality of service computation is based on Zeng et al. [7], where the service selection considers multiple criteria, such as price reliability, availability, etc. and is solved using efficient linear programming methods. Such solutions only select individual sources and compositions to fill information gaps in the query result. This makes the reliability calculations considerably easier since they assume that the data provided is complete.

The only works combining LDF interfaces and Web APIs that focus on reliability are the work of Preda et al. [8,9] and Zeimetz et al. [10]. Preda et al. have developed a framework (ANGIE) for generating queries to encapsulate RESTful Web APIs during query execution. Their query generator composes sequences of requests to APIs and integrates this information into the query result. Their method aims to reduce the number of requests to retrieve results with sufficient recall. While ANGIE prioritizes fast and promising API calls, there is no guarantee that all requests will be answered.

In contrast, Zeimetz et al. proposed a query engine that is able to combine RESTful Web APIs and local RDF graphs in the form of triple stores while tuning its (query) plans towards user preferences. Erroneous information from Web APIs is detected using hierarchical agglomerative clustering. As baseline system they used the approach of Preda at al. and could show that their approach is less vulnerable to erroneous information, even in settings where only unreliable sources are available.

4. Preliminaries

We build on the definitions of Linked Data Fragment interfaces as presented by Heling et al. [21]. Yet, for the scope of this paper we only consider TPF interfaces since they have the least expressiveness and, hence, provide more restrictions than more expressive interfaces, e.g., SPARQL endpoints or brTPF interfaces. Further, we assume the data is stored in form of an RDF graph and is accessible via Linked Data Fragment interfaces.

Let the sets of RDF terms U, B, and L be pairwise disjoint sets of URIs, blank nodes, and literals, and V be a set of variables disjoint from U, B, and L. A triple $(s,p,o) \in (U \cup B) \times U \times (U \cup B \cup L)$ is called an RDF triple. A set of RDF triples is an RDF graph G, and the universe of RDF graphs is denoted as $\mathbb{G}$. A triple pattern t is a 3-tuple with $t \in (U \cup V) \times (U \cup V) \times (U \cup L \cup V)$.

A query Q is a SPARQL expression that is constructed of triple patterns in addition to operators like *AND*, *UNION*, and others. A set of only triple patterns is denoted a basic graph pattern (BGP). To narrow the scope of this paper we focus only on BGP queries. Further, $[\![Q]\!]_G$ and $[\![t]\!]_G$ denote the evaluation of a query Q or a triple pattern t over an RDF graph G.

Definition 1 (TPF Interface)**.** *A Triple Pattern Fragment (TPF) interface $u \in U$ is a Web interface that supports the evaluation of a single triple pattern t. The corresponding function $ep : U \rightarrow \mathbb{G}$ maps each TPF interface u to its default RDF graph $ep(u) = G$. Further, a TPF interface provides metadata about the number of triples $|[\![t]\!]_G|$ that match the triple pattern t in G. Additionally, $P_{rel}(u)$ denotes the reliability of the interface.*

TPF interfaces should follow a paginated approach[2] to avoid overly large responses. The provided metadata about the triples can be used to estimate the number of pages, and all result pages need to subsequently be requested to collect all results.

Definition 2 (Query Evaluation)**.** *The evaluation of a BGP query Q consisting of several triple patterns $t_0, t_1, ..., t_n$ ($n \in \mathbb{N}$) over a TPF interface u is given as*

$$[\![Q]\!]_u = \bigcup_{\forall t_i \in Q} [\![t_i]\!]_{ep(u)}.$$

To evaluate a query Q against a TPF interface u, each individual triple pattern $t_i \in Q$ must be evaluated against u. In order to evaluate a query against a federation of TPF interfaces we need to first define the notion of a federation.

Definition 3 (TPF Federation)**.** *A federation of TPF interfaces $F \subset U$ is a set of URIs of TPF interfaces. The corresponding function ep maps each TPF interface u_i to the RDF graph G_i available at that interface.*

We denote the evaluation of a query over a federation of TPF interfaces as $[\![\cdot]\!]_F$ and define its semantics as follows.

[2] https://linkeddatafragments.org/specification/triple-pattern-fragments/#paging

Definition 4. *Given a BGP query Q and federation F, the result of Q over F is given as*

$$[\![Q]\!]_F = [\![Q]\!]_G \text{ with } G = \bigcup_{\forall u \in F} ep(u)$$

In the following we will discuss the three main contributions of ORAQL. First, it receives as input a BGP query Q, together with a user-specified constraint on a minimum reliability r_{min}. Then an overlap aware profile is created, which provides tuple-wise information about the overlap degree between all pairs of data sources of a federation. Next, the previously computed overlap profile is used to remove all redundant interfaces from a federation F if a query Q is executed. Lastly, a user-provided reliability goal r_{min} is taken into account during query processing. Hence, a method to compute the estimated reliability of a query result against a federation is introduced. To guarantee the user's reliability goal, TPF interfaces may have to be added to the selected sources again.

5. Precomputed Profiles of TPF Interfaces

A well known index structure used for query processing are characteristic sets. It was introduced for RDF graphs by Neumann et al. [25] and denotes a set of different characteristics that describe a graph. Characteristic sets are often used for query planning as they capture the co-occurrences of properties in RDF graphs. Heling et al. [24] extended the work of Neumann et al. so that accurate statistical profile features based on characteristic sets are estimated, relying only on samples of the original dataset.

Even though this approach works well for a more fine-grained source selection, it does not consider the problem of overlapping sources (see Scenario 1 and 2). Further, it is not possible for TPF interfaces to group triple patterns of a query by sources, as TPF interfaces can only process one triple pattern at a time. In the following we present an index that can be leveraged for source selection with less redundant data sources.

The core idea of Heling et al. to create a characteristic set is to randomly select triples from the RDF graph: entities with higher out-degrees have a higher probability of being chosen with this approach, as they appear more frequently in the triples. This is done until a large enough sample size (given by the user) is collected.

We first request a sample from each TPF interface in F. All requested samples are cached so that afterwards the cached triples are used to generate an initial overlap aware profile (O-profile) for all combinations of endpoints. It is important to emphasize that the O-profiles $O(G_a, G_b)$ and $O(G_b, G_a)$ of two RDF Graphs G_a and G_b can differ significantly since, as shown in Figure 2, part (b), endpoint G_a contains all the information from endpoint G_b with regard to the titles of a publication. However, since this is not the case the other way around, the two O-profiles $O(G_a, G_b)$ and $O(G_b, G_a)$ are very different from each other (see Figure 2, part (d)). Formally, an O-profile is defined as follows:

Definition 5 (O-Profile)**.** *Given two RDF graphs G and G', their O-profile $O(G, G')$ is a 3-tuple $(\mathscr{C}, \mathscr{P}, o)$ where $\mathscr{C}$ is the set of all classes of G, $\mathscr{P}$ denotes the set of all property sets where $P_c \in \mathscr{P}$ with $c \in \mathscr{C}$ denotes the set of properties used by entities of class c. Further, $o_{G,G'} : \mathscr{C} \times U \rightarrow [0,1]$ describes the overlap function that returns for a property $p \in P_c$ for class $c \in \mathscr{C}$ the overlap between G and G'. The overlap measures the fraction of entities of class c with property p in graph G that also occur in G'.*

To make this definition easier to grasp, the O-profile of G_a and G_b is shown in Figure 2, part (d). Firstly, in the example for $O(G_a, G_b)$ the set of classes and the associated properties are shown. For example, the class `Inproceedings` $\in \mathscr{C}$ is described by the properties $\{\texttt{title, year, conference, doi}\} \in \mathscr{P}$. The overlap function o_{G_a,G_b} returns the degree of overlap for a class and predicate pair, for example $o_{G_a,G_b}(\texttt{Inproceedings,title}) = 0.7$ and $o_{G_a,G_b}(\texttt{Inproceedings,conference}) = 0.0$ since G_b does not store any conference names.

To create the O-profile of G_a and G_b, the cached samples named $\tilde{G}_a$ and $\tilde{G}_b$ are used. For all classes in $\tilde{G}_a$ it is first iterated over all corresponding entities. Since a predicate can have different overlap degrees for different classes, as shown in the example before, we chose to iterate over the classes. For each entity $e \in \tilde{G}_a$ it is checked whether it (with the same properties, e.g. title, year, etc.) is also present in $\tilde{G}_b$, i.e., for each property of a class, the number of times two entities have used the `title` property, for example, is counted. This allows an overlap value to be created for each property of a class.

However, this alone is not sufficient, as the drawn samples $\tilde{G}_a$ and $\tilde{G}_b$ can be too small to cover all aspects of G_a and G_b, so that the estimated overlap differs significantly from the true overlap. One possibility is to drastically increase the sample size, but this would mean a significantly longer runtime. A better performing way is to draw entities of $\tilde{G}_a$ that that could not be found in $\tilde{G}_b$ and request the information of additional entities. The number of additional requests is given like the sample size by a user.

This process is the most time-consuming step, as individual entities are requested due to the limited expressiveness. For all other types of endpoints (e.g., SPARQL endpoints or brTPF interfaces), this part is significantly easier since they provide more options like querying entities in bulk or sorting entities which makes it possible to avoid additional requests, as better samples can be collected. After this procedure has been executed for each class and property in G_a, a final O-profile $O(G_a, G_b)$ is created based on the sample graphs $\tilde{G}_a$ and $\tilde{G}_b$. The O-profile can also be used to make assumptions for entities without classes by calculating the average overlap for a predicate over all classes.

6. Removal of Redundant Sources

The problem we motivated in section 2 is based on the idea of removing redundant endpoints from F that are covered by other sources in F, e.g., for the example shown in Figure 2(b), G_b and G_c can be removed from F since G_a and G_d cover both endpoints. Hence, the size of the federation can be reduced by 50 percent while still covering 100 percent of the federation data. The optimization problem we focus in this paper is also known as set cover problem [29] which is NP-hard and can be formalized as follows:

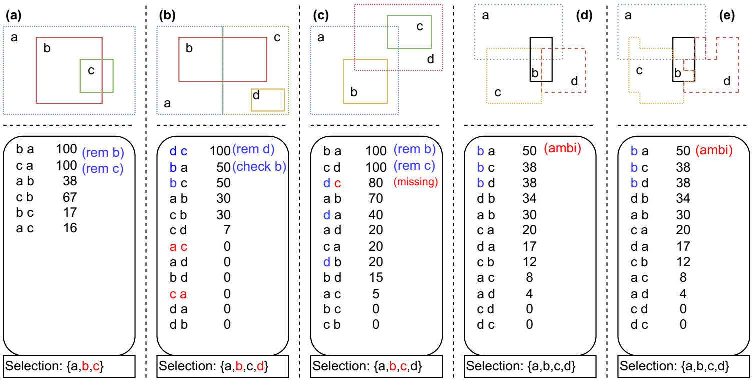
Definition 6 (Minimum Federation). *Given a query Q, a federation $F = \{u_1, u_2, ..., u_n\}$ and $S = 2^F$ a minimum covering of F regarding a query Q is defined as*

$$\arg\min_{S_i \in S} \; s.t. \; |[\![Q]\!]_{S_i}| = |[\![Q]\!]_F|.$$

Informally, we aim to obtain for a query Q a minimum federation $F_{min} \subseteq F$ that covers all information of F for a query Q but with no redundancy. Since the problem is NP-hard there are only a few greedy algorithms that provide a practical approximation for

F_{min} that removes (some) redundant interfaces from F. However, this requires complete access to $G = ep(u)$, which is practically not possible for TPF interfaces.

Next, we present an approach that leverages the O-profile to identify TPF interfaces in a federation F that are covered by other interfaces in F and therefore can be removed.



b a	100	(rem b)
c a	100	(rem c)
a b	38	
c b	67	
b c	17	
a c	16	

Selection: {a,b,c}

d c	100	(rem d)
b a	50	(check b)
b c	50	
a b	30	
c b	30	
c d	7	
a c	0	
a d	0	
b d	0	
c a	0	
d a	0	
d b	0	

Selection: {a,b,c,d}

b a	100	(rem b)
c d	100	(rem c)
d c	80	(missing)
a b	70	
d a	40	
a d	20	
c a	20	
d b	20	
b d	15	
a c	5	
b c	0	
c b	0	

Selection: {a,b,c,d}

b a	50	(ambi)
b c	38	
b d	38	
d b	34	
a b	30	
c a	20	
d a	17	
c b	12	
a c	8	
a d	4	
c d	0	
d c	0	

Selection: {a,b,c,d}

b a	50	(ambi)
b c	38	
b d	38	
d b	34	
a b	30	
c a	20	
d a	17	
c b	12	
a c	8	
a d	4	
c d	0	
d c	0	

Selection: {a,b,c,d}

Figure 3. Different overlap scenarios including overlap information.

To discuss this problem in detail, Figure 3 shows some federation examples with different overlap degrees. Assuming a query Q requests the titles of all publications, the first example shows that the sources a, b and c can provide a title. This information is obtained by an initial probing of all TPF interfaces with the corresponding triple pattern. The result is metadata about the result size. It can also be seen that the sources b and c are contained in a and hence, only a needs to be queried.

The O-profile can be used to determine redundant sources and remove them. It provides information about the overlap degree between two sources (e.g., b and a) for a selected class and property. For the sake of simplicity, the class is ignored in the following. The overlap of b and a for a title is $o_{b,a}(title) = 1.0$ since a contains b. But $o_{a,b}(title) = 0.38$ as only approx. 38 percent of a is covered by b. Based on the O-profile for the first case (shown in the lower part of part (a)), it is easy to see that b and c can be deleted from the federation F (for this query), as b and c are covered by a.

The federation F shown in the second example (part (b)) consists of a, b, c and d. Considering the information belonging to the O-profile, it can be determined that $o_{d,c}(title) = 1.0$ and hence d is removed from F, since c covers all titles from d. Next, we can see that $o_{b,a}(title) + o_{b,c}(title) = 1.0$. Since a and c do not overlap (i.e., $o_{a,c}(title) = 0$), it can be assumed that b is covered by a and c and b is removed from F.

The third example (part (c)) shows that the O-profile cannot be used for a precise coverage computation. First, all TPF interfaces in F that are covered by another (single) source (i.e., b and c) are removed. Next, it is checked whether d is covered by multiple other sources. Hence, we calculate $o_{d,a}(title) + o_{d,b}(title) = 0.6$. $o_{d,c}(title)$ is ignored since c is already been removed from the federation. However, since d is not covered by other sources, as well as a, the corresponding interfaces remain in F.

In the example before we ignored that a and b do not cover 60 percent of d in reality, but only 40 percent. This is since we have to subtract the intersection $a \cup b \cup d$ according to the inclusion-exclusion principle. This means that for values above ≥ 1.0 only in special cases a decision can be made whether the interface is covered. Since only tuple-wise overlap information is stored overlap information about the intersection of more than two interfaces is missing and it is not known whether $a \cup b \cup d = \emptyset$ applies or not. In the second example (part (b)), b could only be removed because a and c have no intersection and therefore $a \cup b \cup c = \emptyset$ can be derived.

This problem can be discussed in more detail, considering the last examples (part (d) and (e)). The degree of overlap shown in the O-profile is the same for both examples. However, the associated graphs overlap differently and it is undecidable by only using the O-profile whether b is covered by other sources and hence, no source can be excluded. Hence it can be concluded that overlap variations exists that make it impossible to remove all redundant sources since the information given by the O-profile is not sufficient.

Algorithm 1 Overlap Based Source Removal.

```
Require: Query $Q$, Federation $F$ and O-profile $O_F$
$F_{min} \leftarrow \emptyset$
for all $t \in Q$ do                                          ▷ iterate over all triple patterns
    $F_t \leftarrow F$
    Let $p_t$ denote the property used in the triple pattern $t$
    for all $u \in F_t$ do
        Let $U' = \{u' : u' \in F_{min} \wedge \exists O(ep(u), ep(u'))\}$
        if $\exists u' \in U'$ s.t. $o_{u,u'}(p_t) = 1.0$ then                    ▷ First Phase
            $F_t = F_t \setminus \{u\}$
        else if $\Sigma_{u' \in U'} o_{u,u'}(p_t) = 1.0 \wedge \forall i, j \in U' : o_{i,j}(p_t) = 0$ then   ▷ Second Phase
            $F_t = F_t \setminus \{u\}$
        end if
    end for
    $F_{min} \leftarrow F_{min} \cup (t, F_t)$
end for
return $F_{min}$
```

The informal approach described above is presented as pseudo code in Algorithm 1. The approach is based on two phases: (1) remove all sources from F that are covered by another (single) source. (2) If it is true for a source u that multiple other sources $u' \in U'$ cover u and the intersection between these sources is empty, u can be removed from F. The result F_{min} stores a source selection for each triple of a query $t \in Q$ (i.e., $(t, F_t) \in F_{min}$.

7. Improvement of Query Result Reliability

Next, the framework has to determine if the selected sources provide the user's required minimum reliability. The correct result is determined by a majority vote. As shown by Zeimetz et al. [10], this approach works well to determine a correct value in a "federation" of Web APIs. Hence, we extend this approach for TPF interfaces in the following.

The Poisson binomial distribution (PBD) is used in literature [10,30] to estimate this precisely. It describes the probability distribution of the number of successes (successful votes) in a collection of n independent yes/no experiments with individual success probabilities $p_1, \ldots, p_n$. The corresponding probability mass function is defined as follows

$$P_{PBD}(F,k) = \sum_{A \in B_k} \prod_{i \in A} P_{res}(u_i) \prod_{j \in A^c} (1 - P_{res}(u_j))$$

where B_k is the set of all subsets of k endpoints for a relation (e.g., a publication title) that can be retrieved via $F = \{u_1, \ldots, u_n\}$. If $n = 3$ and we want to determine $P_{PBD}(F, 2)$, then

B_2 denotes TPF interfaces that deliver a correct answer $\{\{u_1,u_2\},\{u_1,u_3\},\{u_2,u_3\}\}$. A^C denotes the complement of A, i.e., incorrect answers.

Next, we extend the notion of the evaluation of a query $[\![Q]\!]_{F_{min}}$ for an overlap based federation F_{min} as

$$[\![Q]\!]_{F_{min}} = \bigcup_{\forall (t,F_t)\in F_{min}} [\![Q]\!]_{F_t}$$

so that by using P_{PBD}, the reliability of a majority decision can be calculated als follows:

Definition 7 (Reliability)**.** *The reliability based on an overlap based source selection of a federation F_{min} for a query result $[\![Q]\!]_{F_{min}}$ is defined as*

$$R(F,[\![Q]\!]_{F_{min}}) = \arg \min_{(t,F_t)\in F_{min}} R(t,F_t) \textit{ and}$$

$$R(t,F_t) = \begin{cases} P_{rel}(u), & \textit{if } |F_t| = 1 \textit{ and } u \in F_t \\ \max(P_{rel}(u),P_{rel}(u')), & \textit{if } |F_t| = 2 \textit{ and } u,u' \in F_t \\ \sum_{k=\lceil \frac{|F_t|}{2} \rceil}^{|F_t|} P_{PBD}(F_t,k), & \textit{otherwise} \end{cases}$$

The formula distinguishes between the case where only one, two or multiple TPF interfaces are requested. In case only one TPF interface is contained in F_t the reliability of the corresponding interface $P_{rel}(u)$ is used. If $|F_t| = 2$ the interface with the higher reliability is used. Therefore, the max function is used in this case. The probability mass function P_{PBD} is only used in cases where a majority decision can occur (i.e., $|F_t| > 3$).

Next, we present an algorithm that leverages the introduced reliability estimation to increase the reliability of a query result $[\![Q]\!]_{F_{min}}$ over a federation F_{min} such that it meets the reliability r_{min} demanded by a user. Therefore, sources are potentially added to F_{min} that were previously omitted due to a high degree of overlap. A high degree of overlap is desired, as this provides more information that can be used in a majority vote.

Algorithm 2 Reliability Based Source Extension.

```
Require: Query $Q$, Federation $F$ with O-Profile $O_F$ and a minimum reliability $r_{min}$
$F_{min} \leftarrow Algorithm1(Q,F,C_F,O_F)$          ▷ apply overlap based source removal
$F_{rel} \leftarrow F_{min}$                          ▷ init federation
for all $(t,F_t) \in F_{rel}$ do
    while $R(t,F_t) < r_{min}$ do
        $F' \leftarrow F \setminus F_t$               ▷ determine all unused endpoints for triple $t$
        $F' \leftarrow sort_{asc}(F')$                ▷ according to overlap $o(p_t)$
        $F_t \leftarrow F_t \cup F'.pop()$            ▷ add first endpoint in $F'$ to $F_t$
    end while
    $F_{rel} \leftarrow F_{rel} \cup (t,F_t)$         ▷ Update source selection for federation
end for
return $F_{rel}$
```

As input we require a query Q, a federation F with O-profile $O_F = (\mathscr{C},\mathscr{P},o)$ and a minimum reliability r_{min} demanded by a user. The first step of Algorithm 2 is to call Algorithm 1 and compute F_{min}. Further, F_{rel} is initialized with the empty set.

Next, the algorithm iterates over each tuple $(t,F_t) \in F_{min}$ and computes the reliability of $R(t,F_t)$ in order to check if the reliability of the selected sources F_t for triple pattern t

of query Q is not yet sufficient for the user (i.e.,$R(t, F_t) < r_{min}$). As long as the required reliability cannot be met by F_t more endpoints (TPF interfaces) are added to F_t. Therefore, first a temporary set F' is created which contains all endpoints that are not included in F_t. Next, F' is sorted, but in contrast to Algorithm 1, it is sorted ascending according to the overlap $o(p_t)$ and query coverage $cov(Q)$. In order to increase the reliability, a high amount of overlap is needed, so that as many triples can be double checked by a majority vote. Lastly, F_{rel} is updated with the new selection of endpoints F_t.

If the desired reliability is achieved we follow the same approach as Zeimetz et al. [10], i.e., the plan is executed and the data is extracted from the API responses. If more than two results (e.g., titles or author names) from different TPF interfaces are available, a hierarchical agglomerative clustering (HAC) is used to divide the results into clusters of equal information. HAC brings the advantage that the number of clusters does not have to be known. Besides, the *two grams overlap* [31] method was used as dissimilarity method, as it is relatively flexible and can handle minor typing errors, different sequences of names or abbreviations well. The largest cluster is considered as winner.

8. Evaluation

In this section, we present the evaluation of ORAQL[3]. To simulate TPF interfaces and ensure a reproducibility we used the newest version of the ETARA [32] benchmark system[4] since it can be used to simulate TPF interfaces.

Data Sources: As data sources we used different scholarly data sets based on dblp with data on publications between 2015 and 2020. To ensure reproducibility, all data sources[5] and used queries are publicly available. We created seventeen data sets, each with different degrees of overlap. In addition, the datasets used for the reliability evaluation also contain different errors for titles and author names.

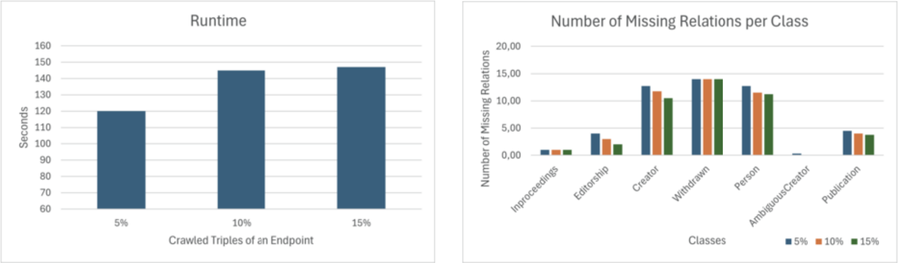
Baseline System: As baseline system we use the approach proposed by Heling and Acosta [6]. It focuses on taking various utility aspects into account during source selection, which include aspects such as the reliability or latency of an endpoint. The original approach only considers so called ALTERNATIVE SERVICE expressions which are the union of SERVICE requests around the same expression. The authors then devise a utility-aware semantics for them, in which quality information like the reliability of a federation is leveraged to potentially reduce the number of endpoints to be contacted. Their selection process considers only endpoints that satisfy the user's reliability requirements. A downside of this is that excluding unreliable data sources may not always be possible as no other sources may be available. Further, multiple endpoints in a federation with a low reliability, providing the same information, can be more reliable than only a single (reliable) endpoint. For this work, we have adopted the approach and modified it for BGP queries. At the beginning, a union of SERVICE requests is created for all possible endpoints for each triple pattern of the query. Subsequently, the baseline is applied.

We created three experiments to evaluate ORAQL's performance. The first one is used to analyze the quality of the created O-profiles. In the second experiment the source selection determined by ORAQL is analyzed and lastly the performance of the state-of-the-art system proposed by Heling and Acosta [6] is compared to ORAQL's.

[3] https://github.com/dbis-trier-university/ORAQL

[4] https://github.com/ETARA-Benchmark-System

[5] https://doi.org/10.5281/zenodo.12634642



(a) Comparison of runtime.

(b) Number of missing relations.

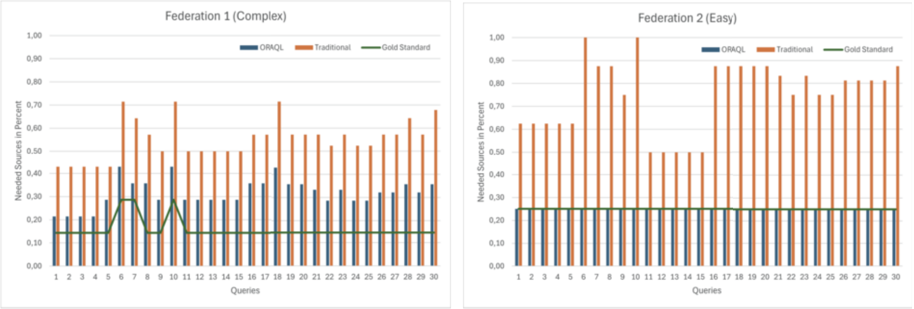
Figure 4. Runtime and missing relations of O-profile.

Experiment 1: (Overlap Index) The first experiment focuses on analyzing the determined O-profiles of a federation of five TPF interfaces. Thereby, we focus on the influence of the sample size and the runtime.

We retrieved five, ten and fifteen percent of an endpoints triples. The number of additional requests was always set to 100 since a broad test series of various sample size and additional requests would have gone beyond the scope of the work. The results presented in Figure 4 indicate that larger sample size increase the chances to capture overlap information of rarely used classes and properties since less relations for each class are missing. Further, Figure 4a shows that the runtime does not drastically increase by raising the sample size. The step between five and ten percent results from the fact that more classes and relations were found for which additional probing requests are performed. However, an increase from ten to fifteen percent has almost no effect, since only few more relations can be found. The most frequent relations and classes are found in all cases. The maximal overlap deviation compared to the real overlap, determined by accessing the full data sources and creating a comprehensive gold standard overlap profile, is around five percent. As shown in the results of the following experiments, this precision is sufficient to achieve good results. However, classes and relations that occur in less than ten percent of the triples could not be found. This problem is also evident in other studies focusing on the creation of additional index profiles [24]. However, as described in section 6, all TPF interfaces of a federation are initially probed. Hence, even for predicates that are missing in an O-profile, metadata about how often they occur is retrieved. Therefore, no coverage is lost since we trivially assume an overlap of 0.

Experiment 2: (Source Selection) The aim of the second experiment is to analyze ORAQL's selected TPF interfaces of a federation compared to traditional federations, removing only sources that deliver no results for any triple pattern. Further, the number of selected interfaces is compared to an optimal selection denoted as gold standard.

Hence, we have created a set of 30 queries covering a result size between one and 200 triples to analyze a wide range of cases. The queries were executed against two federations. The first federation consists of seven TPF interfaces and has a high degree of overlap. There are several combinations of three or more interfaces that have an intersection. This is particularly challenging since for performance reasons the overlap is only captured tuple-wise and as described in Section 6 there are cases where sources cannot be excluded because we lack information. The second federation consists of four TPF interfaces and, in contrast to the first federation, there is no case where more than two TPF interfaces partially overlap (similar to Figure 3 (a) and (b)).



(a) (b)

Figure 5. Comparison of the number of selected sources for two different federations.

Figure 5 shows the results for our proposed approach that removes redundant data sources, a traditional federated query engine and an optimal solution, denoted as gold standard. For the first federation, which has a complex degree of overlap between the individual data sources, it is evident that traditional query engines are quite capable of reducing the number of data sources. In the case of TPF interfaces, this is mainly due to the usual procedure of sampling each interface for the first time. Since TPF interfaces follow a paginated approach to avoid overly large responses they are first exploited as a simplified ask query. The returned query result page contains metadata about the result size and hence interfaces with result size of zero can be removed before all queries are send. And nevertheless, many sources still remain that provide redundant data and can therefore be removed. Figure 5 shows that ORAQL is able to significantly reduce the number of selected sources compared to traditional federated query engines while maintaining complete coverage (no information is lost). In some cases, ORAQL is even close to the gold standard. Since the second federation has significantly less complex overlap ratios, the optimum can even be achieved for such clear cases. However, the first federation is the more common real-world federation.

Experiment 3: (Result Reliability) To evaluate the performance and tunability of the baseline system and ORAQL for different reliability thresholds (0.7 to 0.9) we used ten queries with a result size between 100 and 2000, resulting in 30 query-reliability combinations. We have restricted the queries to a result set of over 100 triples to ensure more stable reliability results during the evaluation. Queries that are answered with only a few triples would otherwise have too strong an effect on the average reliability.

The queries were executed against two federations. The first federation will from now on be denoted as a “trustful” federation and consists of three TPF interfaces with a reliability of 0.75 and a fourth TPF interface with a reliability of only 0.66. The second federation is a less trustful federation and consists of two TPF interfaces with a reliability of 0.75 and a further two TPF interface with a reliability of only 0.66.

Figure 6 shows the reliability and coverage results for the “trustful” federation. It is easy to see that ORAQL can always produce a query result with the required reliability for thresholds between 0.7 and 0.9. The baseline system could only retrieve results for a reliability of 0.7, as it removes all endpoints with a lower reliability from the federation. This has the disadvantage that the maximum reliable results that can be found are those provided by the most reliable interface. In addition, ORAQL performs a majority vote

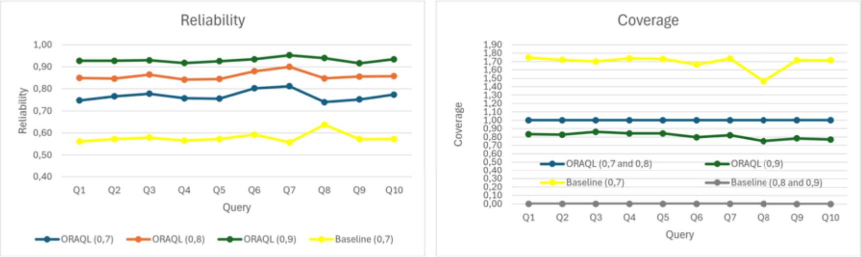


Figure 6. Reliability and coverage of a "trustful" federation.

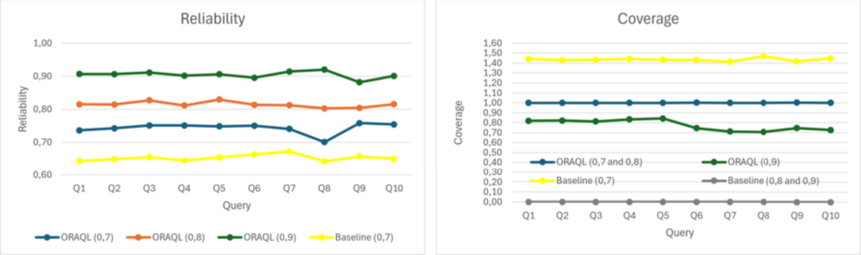


Figure 7. Reliability and coverage of a less trustful federation.

compared to the baseline system, which is why results can be found even for high reliability values such as 0.9. Heling and Acosta's approach does not check the requested triples and therefore contains incorrect data in the query result. Moreover, the baseline system queries all three TPF interfaces with a reliability greater than 0.7 and hence collects the incorrect data of all three interfaces, resulting in a low reliability.

Considering the coverage, it is noticeable that the baseline system delivers significantly more query results than required since contradictory triples, e.g., three different titles for the same publication are queried and collected. In comparison, ORAQL does not query too many results and identifies correct triples via a majority vote. The drawback of this approach is that for reliability values above 0.9 some coverage is lost. This is because in some cases there is no majority since all interfaces deliver contradicting values. Clearly, this is an extreme case, but but it shows that ORAQL is able to deliver the required reliability even in extreme situations.

The results for the second federation are shown in Figure 7. Even if the achieved reliability has fallen slightly, the overall result is similar. Hence we can conclude that ORAQL achieves good results even for less trustful federations.

9. Conclusion

The presented methods to capture tuple-wise overlap information of a federation have shown to generate useful overlap profiles with a maximum deviation of less than five percent. Even if the identification of redundant data (set cover problem) is NP-hard we presented an approximation with a significant reduction in requested endpoints without losing results (recall). Further, we have shown that ORAQL is granularly tunable towards reliability and can beat the baseline system in terms of coverage and reliability.

References

[1] Hartig O, Letter I, Pérez J. A Formal Framework for Comparing Linked Data Fragments. In: The Semantic Web - ISWC 2017 - 16th International Semantic Web Conference, Vienna, Austria, October 21-25, 2017, Proceedings, Part I. vol. 10587 of Lecture Notes in Computer Science. Springer; 2017. p. 364-82. Available from: https://doi.org/10.1007/978-3-319-68288-4_22.

[2] Cheng S, Hartig O. Source Selection for SPARQL Endpoints: Fit for Heterogeneous Federations of RDF Data Sources? In: Saleem M, Ngomo AN, editors. Proceedings of the QuWeDa 2022: 6th Workshop on Storing, Querying and Benchmarking Knowledge Graphs co-located with 21st International Semantic Web Conference (ISWC 2022), Hangzhou, China, 23-27 October 2022. vol. 3279 of CEUR Workshop Proceedings. CEUR-WS.org; 2022. p. 5-16. Available from: https://ceur-ws.org/Vol-3279/paper1.pdf.

[3] Cheng S, Hartig O. Towards Query Processing over Heterogeneous Federations of RDF Data Sources. In: The Semantic Web: ESWC 2022 Satellite Events - Hersonissos, Crete, Greece, May 29 - June 2, 2022, Proceedings. vol. 13384 of Lecture Notes in Computer Science. Springer; 2022. p. 57-62. Available from: https://doi.org/10.1007/978-3-031-11609-4_11.

[4] Cheng S, Hartig O. A Cost Model to Optimize Queries over Heterogeneous Federations of RDF Data Sources. In: Joint Proceedings of the ESWC 2023 Workshops and Tutorials co-located with 20th European Semantic Web Conference (ESWC 2023), Hersonissos, Greece, May 28-29, 2023. vol. 3443 of CEUR Workshop Proceedings. CEUR-WS.org; 2023. Available from: https://ceur-ws.org/Vol-3443/ESWC_2023_DMKG_paper_7042.pdf.

[5] Alili H, Belhajjame K, Drira R, Grigori D, Ghézala H. Quality Based Data Integration for Enriching User Data Sources in Service Lakes. In: Proc. ICWS; 2018. p. 163-70. Available from: https://doi.org/10.1109/ICWS.2018.00028.

[6] Heling L, Acosta M. Utility-aware Semantics for Alternative Service Expressions in Federated SPARQL Queries. In: Proc. ICWS; 2022. p. 208-18. Available from: https://doi.org/10.1109/ICWS55610.2022.00042.

[7] Zeng L, Benatallah B, Dumas M, Kalagnanam J, Sheng Q. Quality driven web services composition. In: Proc. WWW; 2003. p. 411-21. Available from: https://doi.org/10.1145/775152.775211.

[8] Preda N, Kasneci G, Suchanek F, Neumann T, Yuan W, Weikum G. Active knowledge: dynamically enriching RDF knowledge bases by web services. In: Proc. SIGMOD; 2010. p. 399-410. Available from: https://doi.org/10.1145/1807167.1807212.

[9] Preda N, Suchanek F, Kasneci G, Neumann T, Ramanath M, Weikum G. ANGIE: Active Knowledge for Interactive Exploration. Proc VLDB Endow. 2009;2(2):1570-3. Available from: http://www.vldb.org/pvldb/vol2/vldb09-411.pdf.

[10] Zeimetz T, Hose K, Schenkel R. Tunable Query Optimizer for Web APIs and User Preferences. In: Venable KB, Garijo D, Jalaian B, editors. Proceedings of the 12th Knowledge Capture Conference 2023, K-CAP 2023, Pensacola, FL, USA, December 5-7, 2023. ACM; 2023. p. 92-100. Available from: https://doi.org/10.1145/3587259.3627542.

[11] Florescu D, Koller D, Levy AY. Using Probabilistic Information in Data Integration. In: VLDB'97, Proceedings of 23rd International Conference on Very Large Data Bases, August 25-29, 1997, Athens, Greece. Morgan Kaufmann; 1997. p. 216-25. Available from: http://www.vldb.org/conf/1997/P216.PDF.

[12] Vassalos V, Papakonstantinou Y. Using knowledge of redundancy for query optimization in mediators. In: In AAAI Workshop on AI and Info. Integration; 1998. .

[13] Assaf L. Names, Identifications, and Social Change : Naming Practices and the (Re-)Shaping of Identities and Relationships within German Jewish Communities in the Late Middle Ages; 2016.

[14] Abdelaziz I, Mansour E, Ouzzani M, Aboulnaga A, Kalnis P. Lusail: A System for Querying Linked Data at Scale. PVLDB. 2017;11(4):485-98. Available from: http://www.vldb.org/pvldb/vol11/p485-abdelaziz.pdf.

[15] Charalambidis A, Troumpoukis A, Konstantopoulos S. SemaGrow: optimizing federated SPARQL queries. In: Proc. SEMANTiCS; 2015. p. 121-8. Available from: https://doi.org/10.1145/2814864.2814886.

[16] Görlitz O, Staab S. SPLENDID: SPARQL Endpoint Federation Exploiting VOID Descriptions. In: Proc. COLD2011; 2011. Available from: http://ceur-ws.org/Vol-782/GoerlitzAndStaab_COLD2011.pdf.

[17] Hose K, Schenkel R. Towards benefit-based RDF source selection for SPARQL queries. In: Pro. SWIM; 2012. p. 2. Available from: https://doi.org/10.1145/2237867.2237869.

[18] Montoya G, Skaf-Molli H, Hose K. The Odyssey Approach for Optimizing Federated SPARQL Queries. In: Proc. ISWC; 2017. p. 471-89. Available from: https://doi.org/10.1007/978-3-319-68288-4_28.

[19] Saleem M, Ngomo A. HiBISCuS: Hypergraph-Based Source Selection for SPARQL Endpoint Federation. In: Proc. ESWC; 2014. p. 176-91. Available from: https://doi.org/10.1007/978-3-319-07443-6_13.

[20] Schwarte A, Haase P, Hose K, Schenkel R, Schmidt M. FedX: Optimization Techniques for Federated Query Processing on Linked Data. In: Proc. ISWC; 2011. p. 601-16. Available from: https://doi.org/10.1007/978-3-642-25073-6_38.

[21] Heling L, Acosta M. Federated SPARQL Query Processing over Heterogeneous Linked Data Fragments. In: Proc. WWW; 2022. p. 1047-57. Available from: https://doi.org/10.1145/3485447.3511947.

[22] Cheng S, Hartig O. FedQPL: A Language for Logical Query Plans over Heterogeneous Federations of RDF Data Sources. In: iiWAS '20: The 22nd International Conference on Information Integration and Web-based Applications & Services, Virtual Event / Chiang Mai, Thailand, November 30 - December 2, 2020. ACM; 2020. p. 436-45. Available from: https://doi.org/10.1145/3428757.3429120.

[23] Azzam A, Polleres A, D Fernández J, Acosta M. smart-KG: Partition-Based Linked Data Fragments for querying knowledge graphs. Semantic Web. 2022;(Preprint):1-45.

[24] Heling L, Acosta M. Characteristic sets profile features: Estimation and application to SPARQL query planning. Semantic Web. 2023;14(3):491-526. Available from: https://doi.org/10.3233/SW-222903.

[25] Neumann T, Moerkotte G. Characteristic sets: Accurate cardinality estimation for RDF queries with multiple joins. In: Proceedings of the 27th International Conference on Data Engineering, ICDE 2011, April 11-16, 2011, Hannover, Germany. IEEE Computer Society; 2011. p. 984-94. Available from: https://doi.org/10.1109/ICDE.2011.5767868.

[26] Salloum M, Dong XL, Srivastava D, Tsotras VJ. Online Ordering of Overlapping Data Sources. Proc VLDB Endow. 2013;7(3):133-44. Available from: http://www.vldb.org/pvldb/vol7/p133-salloum.pdf.

[27] Roth A, Naumann F. System P: Completeness-driven Query Answering in Peer Data Management Systems. In: Datenbanksysteme in Business, Technologie und Web (BTW 2007), 12. Fachtagung des GI-Fachbereichs "Datenbanken und Informationssysteme" (DBIS), Proceedings, 7.-9. März 2007, Aachen, Germany. vol. P-103 of LNI. GI; 2007. p. 625-8. Available from: http://subs.emis.de/LNI/Proceedings/Proceedings103/article1431.html.

[28] Bleiholder J, Khuller S, Naumann F, Raschid L, Wu Y. Query Planning in the Presence of Overlapping Sources. In: Advances in Database Technology - EDBT 2006, 10th International Conference on Extending Database Technology, Munich, Germany, March 26-31, 2006, Proceedings. vol. 3896 of Lecture Notes in Computer Science. Springer; 2006. p. 811-28. Available from: https://doi.org/10.1007/11687238_48.

[29] Balas E, Padberg MW. On the Set-Covering Problem. Oper Res. 1972;20(6):1152-61. Available from: https://doi.org/10.1287/opre.20.6.1152.

[30] Wang YH. On the number of Successes In Independent Trials. Statistica Sinica. 1993;3(2):295-312. Available from: http://www.jstor.org/stable/24304959.

[31] Baltes S, Dumani L, Treude C, Diehl S. SOTorrent: reconstructing and analyzing the evolution of stack overflow posts. In: Proc. MSR; 2018. p. 319-30. Available from: https://doi.org/10.1145/3196398.3196430.

[32] Zeimetz T, Büsching M, Birringer F, Otter C, Zeiler D, Schenkel R. Evaluation toolkit for API and RDF alignment. In: OM@ISWC; 2023. Available from: http://disi.unitn.it/~pavel/om2023/papers/om2023_LTpaper5.pdf.