

# Characteristic Sets Profile Features: Estimation and Application to SPARQL Query Planning

Lars Heling<sup>a,\*</sup> and Maribel Acosta<sup>b</sup>

<sup>a</sup>*AIFB, Karlsruhe Institute of Technology, Germany*

*E-mail: heling@kit.edu*

<sup>b</sup>*Center of Computer Science, Ruhr University Bochum, Germany*

*E-mail: maribel.acosta@rub.de*

**Abstract.** RDF dataset profiling is the task of extracting a formal representation of a dataset’s features. Such features may cover various aspects of the RDF dataset ranging from information on licensing and provenance to statistical descriptors of the data distribution and its semantics. In this work, we focus on the characteristic sets profile features that capture both structural and semantic information of an RDF dataset, making them a valuable resource for different downstream applications. While previous research demonstrated the benefits of characteristic sets in centralized and federated query processing, access to these fine-grained statistics is taken for granted. However, especially in federated query processing, computing this profile feature is challenging as it can be difficult and/or costly to access and process the entire data from all federation members. We address this shortcoming by introducing the concept of a profile feature estimation and propose a sampling-based approach to generate estimations for the characteristic sets profile feature. In addition, we showcase the applicability of these feature estimations in federated querying by proposing a query planning approach that is specifically designed to leverage these feature estimations. In our first experimental study, we intrinsically evaluate our approach on the representativeness of the feature estimation. The results show that even small samples of just 0.5% of the original graph’s entities allow for estimating both structural and statistical properties of the characteristic sets profile features. Our second experimental study extrinsically evaluates the estimations by investigating their applicability in our query planner using the well-known FedBench benchmark. The results of the experiments show that the estimated profile features allow for obtaining efficient query plans.

Keywords: RDF dataset profiling, Sampling, Query Processing, Federations

## 1. Introduction

In recent years, the amount of information published as Linked Open Data has steadily increased as new RDF datasets are published and existing datasets are growing<sup>1</sup>. Simultaneously, Semantic Web technologies are more and more used in companies to integrate data from heterogeneous sources using Knowledge Graphs and Semantic Data Lakes [1–3]. In order to support applications that leverage these large corpora of semantic data, we require means to extract high-level informa-

tion on the content, the quality, and the structure of the data provided by these sources. To this end, a variety of approaches have been proposed to summarize semantic graphs [4], assess their quality [5] and extract various features in dataset profiles [6]. Similar to the field of databases, where “Data profiling refers to the activity of creating small but informative summaries of a database” [7], RDF dataset profiling aims to formally represent a set of features of an RDF dataset to aid downstream tasks [8]. RDF dataset profiles typically consist of several profile features that capture information that ranges from licensing, provenance to statistical characteristics of the dataset [6]. The cost to acquire statistical features depend on their granularity as well

\*Corresponding author. E-mail: heling@kit.edu.

<sup>1</sup><https://lod-cloud.net/>

as the size of and the means to access the RDF dataset. Dataset profiling techniques typically assume local access to the entire datasets such that the features can be computed more efficiently. Upon creation, RDF dataset profiles support applications such as entity linking, entity retrieval, distributed search and (federated) query processing [6]. In particular, centralized and federated query engines rely on fine-grained dataset profiles to obtain efficient query plans [9–16]. For instance, characteristic sets are often used to estimate join cardinalities in query plan optimization [14–17].

In this work, we focus on the *Characteristic Sets Profile Feature* (CSPF) [18], which is a statistical feature of RDF datasets that includes the characteristic sets, their occurrence distribution, and the multiplicity of their predicates. We focus on this particular profile features as an informative statistical characterization of RDF datasets due to the following reasons. First, CSPFs implicitly captures structural features of the data, such as the mean out-degree, distinct number of subjects, and the set of predicates including their counts. Second, characteristic sets implicitly reflect the schema of an RDF dataset as they capture semantic information on its entities. Finally, CSPFs are well suited to be leveraged by (decentralized) query planning approaches for cardinality estimations and other downstream tasks as they provide insights into the predicate co-occurrence. While the CSPFs provide rich information that is beneficial for various applications, obtaining the profile feature can be a challenging task. For instance, the federated query planning approach Odyssey [16] exploits information on the characteristic sets and how they are linked across datasets to estimate intermediate results when optimizing query plans. Yet, especially in federated query processing, it can be difficult and/or costly to access the entire RDF datasets of all members in the federation and, subsequently, compute these fine-grained statistics. First, data dumps are not always available in federated querying as data publisher may choose different Linked Data Fragments (LDFs) to publish their data [19]. For example, the datasets may only be partially accessed via SPARQL endpoints or Triple Pattern Fragment servers. Moreover, in the case of public resources, the statistics need to be obtained while respecting the fair use policies of the publisher [20]. **Second, to build the characteristic sets for an RDF dataset requires scanning the entire dataset.** This computational effort can be an additional restriction, especially for very large and evolving datasets, such as

DBpedia<sup>2</sup> or Wikidata<sup>3</sup> with more than a billion triples that are updated frequently

Addressing these limitations, we propose an approach that estimates accurate statistical profile features based on characteristic sets and that relies only on a small sample of the original dataset. Our sample-based approach alleviates the necessity of having access to the entire dataset and also reduces the cost of computing the profile feature. Given an RDF dataset, our approach employs an entity-entire sampling method and computes the characteristic sets of the entities to build the CSPF of the sample. Then, we apply a projection function to extrapolate the statistics observed in the sample to estimate the original dataset’s CSPF. As the statistics of the characteristic sets are sensitive to the structure of the datasets and the sample, this extrapolation step can be challenging. Take for example the following characteristic sets  $S_1$ ,  $S_2$  and  $S_3$  from the LinkedMDB datasets and the number of associated subjects (*count*)<sup>4</sup>:

	Characteristic Set	<i>count</i>
$S_1$	<code>fowl:sameAs, dbp:hasPhotoCollection</code>	9838
$S_2$	<code>fowl:sameAs, dbp:hasPhotoCollection, lmdb:relatedBook</code>	5
$S_3$	<code>fowl:sameAs, dbp:hasPhotoCollection, foaf:page</code>	697

We observe that the characteristic set  $S_1$  differs only by a single predicate from both  $S_2$  and  $S_3$ . However, the count values differ significantly as  $S_1$  occurs almost 2000 times more often than  $S_2$  but only about 14 times more often than  $S_3$ . The example shows the challenge of estimating the count statistic accurately, as the semi-structured nature of RDF datasets can lead to situations where small differences between characteristic sets result in large changes in their occurrence. Therefore, we investigate which sampling and extrapolation approaches are less prone to potential estimation errors. In addition, we want to understand to which extent the estimated CSPF can be leveraged in federated query processing. We, therefore, propose a federated query planning approach that is specifically designed to make use of estimated CSPF to obtain query plans. To this end, we investigate the following research questions.

<sup>2</sup><https://www.dbpedia.org/>

<sup>3</sup><https://www.wikidata.org/>

<sup>4</sup>In the remainder of this work, we assume prefixes as given in <https://prefix.cc>

*RQ 1 Which steps are sufficient for accurately estimating RDF dataset profile features?* First, we want to investigate how we can estimate profile features without requiring access to the entire dataset. In particular, we study which sampling methods are suitable to obtain a representative sample for the specific characteristic sets profile feature. Based on samples, we investigate how the information captured in the sample can be extrapolated to the original dataset. The goal of this research question is, therefore, to understand the impact of different sampling methods, sample sizes, and extrapolation techniques on the profile feature estimation.

*RQ 2 How can we assess the effectiveness of estimated profile features?* Given the original profile feature and an estimation, we need means to assess the accuracy of the estimation. As we focus on the characteristic sets profile features that capture various characteristics of the dataset, we investigate how the representativeness of an estimation with respect to those characteristics can be measured. Moreover, with this question, we also want to understand which characteristics are more difficult to estimate and how the structure of the original dataset impacts the estimations.

*RQ 3 How can we leverage estimated profile features to support downstream applications such as federated query processing?* This question aims to investigate how estimated profile features still support applications that rely on these features. As we focus on the characteristic sets profile feature, we consider applications that leverage this information and where the computation of the complete statistics may not be feasible. Therefore, we investigate how a federated query planning approach can leverage the estimated profile features. Specifically, we study how source selection, query decomposition, and join ordering is affected by different estimations. Finally, we want to understand to which extent the effectiveness of the estimation (c.f. RQ 2) is reflected in the performance of the query plans.

*Contributions* This paper builds on a previous work of ours [18], which focuses on a sampling-based approach for estimating characteristic sets for RDF dataset profiles. We extend this work regarding two major aspects. First, we expand our experimental evaluation about CSPF estimations by studying additional datasets and providing a more fine-grained analysis of the results. Second, we address a central aspect of the profile feature estimations, which is their application in downstream tasks. Particularly, we propose a novel federated query planning approach based on the profile

feature estimations. Finally, we evaluate this approach in an experimental study. In summary, the novel contributions of this work are as follows.

- C 1** An extensive experimental study of our Characteristic Sets Profile Feature estimation approach on eight new datasets,
- C 2** a fine-grained analysis of the results to understand the effectiveness of our CSPF estimation approach and its components,
- C 3** a new estimations-based query planner to showcase the application of the estimations in federated query processing,
- C 4** an experimental study of our query planner on the well-known FedBench benchmark, and
- C 5** a detailed evaluation of the results to understand the benefits and limitations of the query planner.

*Structure of this paper* The remainder of this work is organized as follows. In Section 2, we introduce RDF dataset profiling and present our problem definition for estimating profile features. We define characteristic sets profile features and present our sampling-based approach to estimate them in Section 3. In Section 4, we present an application of the estimated profile features in federated query planning. We evaluate our profile feature estimation and estimation-based query approaches in Section 5. Related work on statistical profiling, graph sampling, and federated query processing is discussed in Section 6. Finally, we conclude our work with remarks on future work in Section 7.

## 2. RDF Dataset Profiling

The Resource Description Framework (RDF) is a graph-based data model. The atomic elements in RDF are triples, which are statements represented as 3-tuple of RDF terms:  $t = \langle s, p, o \rangle$  with  $s$  the subject,  $p$  the predicate, and  $o$  the object of the triple. Each term of an RDF triple comes from one of the pairwise disjoint sets of Internationalized Resource Identifiers (IRIs)  $I$ , blank nodes  $B$ , or literals  $L$ . A set of triples allows for constructing a directed labeled graph, where the triples' subjects and objects are nodes connected by the predicates as directed labeled edges. A set of RDF triples is called an RDF graph  $G$  and we denote the universe of RDF graphs by  $\mathcal{G}$ . The set of subjects in an RDF graph is often referred to as its *entities*. According to the RDF 1.1 specification of

the W3C<sup>5</sup>, a collection of RDF graphs is called an RDF dataset where all but one graph are named graphs associated with a graph name (IRI or blank node), and the unnamed default graph. For the sake of simplicity, in the remainder of this work, we consider, RDF datasets with just the default graph and use the terms RDF graph and RDF dataset interchangeably.

### 2.1. Profile Features

RDF datasets are summarized in *RDF dataset profiles* that consist of profile features describing the characteristics and entities of the dataset. In the area of traditional relational database theory, a statistical profile is defined as a “complex object composed of quantitative descriptors” that “summarizes the instances of a database” [21]. These quantitative descriptors can cover different characteristics of the instances, for example:

- central tendency (e.g., mean),
- dispersion, (e.g., coefficient of variation),
- size (e.g., number of instances), and
- frequency distribution (e.g., uniformity).

The specific characteristics covered in such profiles depend on the downstream application, for example, the statistical features required by a query optimizer that uses the profile to devise efficient query plans. Similar to relational databases, statistic profiles are also commonly used for a variety of applications in the realm of RDF datasets. These applications range from centralized RDF data management solutions [22] and RDF graph compression [23] to query optimization for both centralized and federated query processing [16, 24, 25]. For instance, a common application of such statistical profiles in query optimization is the estimation of join cardinalities for subqueries.

The terminology for such statistical profiles varies according to their application [4–6]. In this work, we follow the terminology introduced by Ellefi et al. [6]: An *RDF dataset profile* is a formal representation of a set of *dataset profile features*. Moreover, we focus on *statistical profile features* and define them as follows.

**Definition 2.1** (Profile Feature [18]). Given an RDF graph  $G$ , a profile feature  $F^1 G^\circ$  is defined as a characteristic describing a statistical feature  $F$  of graph  $G$ .

As previously mentioned, computing these profile features can be challenging because accessing and processing the entire datasets is difficult and/or costly.

<sup>5</sup><https://www.w3.org/TR/rdf11-datasets/>

Therefore, we investigate profile feature estimation without requiring access to the complete RDF graph.

### 2.2. Profile Feature Estimation

Addressing the limitation of requiring access to the entire RDF dataset for computing statistical profile features, we propose the concept of *Profile Feature Estimation*. Profile feature estimations aim to estimate a statistical profile features of RDF datasets using a subset of the data only. The goal is to generate a profile feature estimation that is *as similar as possible* to the original profile feature while requiring access to a subset of the dataset only. In particular, we propose an approach that relies on a sample from the original RDF graph in combination with a projection function to extrapolate the true profile feature. Consequently, we define a profile feature estimation as follows.

**Definition 2.2** (Profile Feature Estimation [18]). Given an RDF graph  $G$ , a projection function  $\phi$ , a subgraph  $H \subseteq G$ , and the profile feature  $F^1 G^\circ$ , a profile feature estimation  $\hat{F}^1 G^\circ$  for  $G$  is defined as

$$\hat{F}^1 G^\circ := \phi^1 F^1 H^{\circ\circ}$$

In an ideal situation, the estimated profile feature is identical to the true original profile feature (computed over the complete RDF graph). However, the profile feature to be estimated and the properties of both the subgraph  $H$  and the projection function  $\phi$  may affect the differences between the true and the estimated feature. For example, given a larger subgraph, the estimation might be more accurate than for a smaller subgraph, as the larger subgraph potentially covers more characteristics of the original graph. The problem is finding an estimation for the profile feature which maximizes the similarity to the profile feature of the original RDF graph. To this aim, we introduce the concept of a similarity function  $\delta$  that maps two profile features to a similarity value. The profile feature estimation problem is given as finding a profile feature estimation that maximizes the similarity to the original profile feature.

**Definition 2.3** (Profile Feature Estimation Problem [18]). Given an RDF graph  $G$  and a profile feature  $F^1 G^\circ$ , the problem of profile feature estimation is defined as follows. Determine a profile feature estimation  $\hat{F}^1 G^\circ$ , such that  $\hat{F}^1 G^\circ = \phi^1 F^1 H^{\circ\circ}$  and

$$\max_{\hat{F}^1 G^\circ} \delta^1 F^1 G^\circ, \hat{F}^1 G^{\circ\circ}$$

1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51

1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51

with  $|H| = |G|$ ,  $\hat{\cdot}$  the universe of profile feature estimations, and  $\delta$  a function assessing the similarity of two profile features.

In order to maximize the similarity, the sampling algorithm for obtaining  $H$  and the projection function  $\phi$  need to be chosen appropriately. The particular method to measure the similarity  $\delta$  depends on the actual profile feature to be estimated with potentially several similarity functions for the same profile feature. Take for example a profile feature  $F^1G^o$  that counts all literal values in an RDF dataset and  $\hat{F}^1G^o$  estimating the count based on a sample of  $G$ . The similarity to determine how well the count is estimated could be calculated as the absolute difference or the relative difference between the true count and the estimated count. In other words, the similarity function measures the *representativeness* of the profile feature estimation  $\hat{F}$  with respect to  $F$ . In network theory, the representativeness of a sample is commonly assessed by how well it captures the structural properties of the original graph [26–28]. As we focus on a more comprehensive profile feature based on the characteristic sets – which capture structural and semantic features of the graph’s entities– considering only structural features of the sample and the graph may not be sufficient to assess how representative a sample is and, thus, a feature estimation for an RDF graph is. Note that in the remainder of this work, we only focus on this specific profile feature and investigate how the estimated profile feature can be leveraged in federated query processing. Nonetheless, the presented approach can be adapted and applied to other statistical profile features potentially supporting other applications as well.

### 3. Characteristic Sets Profile Feature Estimation

We now present our sampling-based approach to estimate profile features based on the characteristic sets of RDF graphs. In this section, we first introduce the concept of characteristic sets and define the corresponding Characteristic Sets Profile Feature. Thereafter, we present our approach that combines RDF graph sampling with projection functions to estimate the Characteristic Sets Profile Feature. Finally, we propose structural and statistical similarity measures to assess the representativeness of these estimations.

#### 3.1. Characteristic Sets Profile Feature

The concept of characteristic sets for RDF graphs was introduced by Neumann et al. [14]. In their work, the authors used characteristic sets for query planning as they capture the co-occurrences of predicates in RDF graphs. The idea of characteristic sets is describing semantically similar entities by grouping them according to the set of predicates the entities share. Furthermore, the number of entities in each group are counted as well as the average usage of their predicates. As a result, the characteristic sets capture both statistical information on the data distribution as well as semantic information on the entities in an RDF graph.

**Definition 3.1** (Characteristic Sets [14]). The characteristic set of an entity  $s$  in an RDF graph  $G$  is given by:  $S_C^1s^o := \{p \mid \exists o : \langle s, p, o \rangle \in G\}$ . Furthermore, for a given RDF graph  $G$ , the set of characteristic sets is given by  $\mathcal{C}^1G^o := \{S_C^1s^o \mid \exists p, o : \langle s, p, o \rangle \in G\}$

In an RDF graph, the statistical information determined for the characteristic sets typically includes the number of occurrences (count) of a given characteristic set as well as the multiplicities of the predicates within each characteristic set. These statistics are used in both centralized triple stores and federated query engines because they allow to determine exact join cardinalities for specific DISTINCT queries and to compute cardinality estimations for non-distinct queries [14, 16, 24, 25]. Similar to Neumann et al. [14], we define the number of occurrences  $count^1S^o$  of a characteristic set  $S = \{p_1, p_2, \dots\}$  in an RDF graph  $G$  as

$$|\{s \mid \exists p, o : \langle s, p, o \rangle \in G \wedge S_C^1s^o = S\}| \quad (1)$$

In addition, in this work, we focus on the occurrences of predicates in characteristic sets by considering their mean multiplicity. The mean multiplicity  $multiplicity^1p_i, S^o$  for  $p_i$  in  $S$  is given by

$$\frac{|\{s, p_i, o \mid \langle s, p_i, o \rangle \in G \wedge S_C^1s^o = S\}|}{count^1S^o} \quad (2)$$

In other words, for a given characteristic set, the multiplicity measures how often each predicate occurs on average. Combining the count and mean multiplicity statistics of a characteristic set, we can compute the number of triples that are *covered* by it. The relative

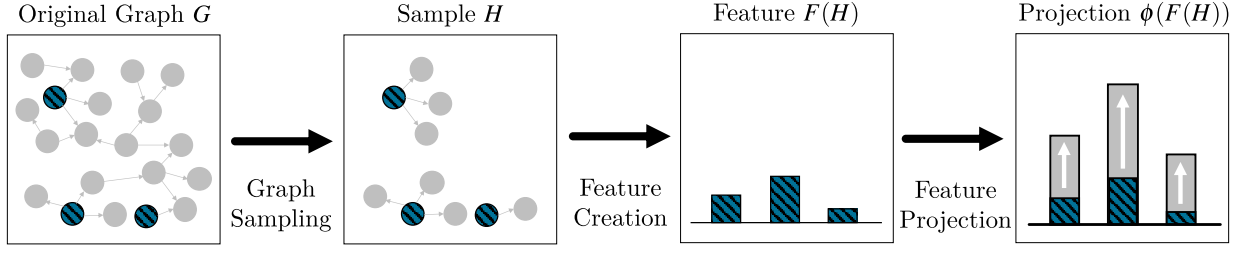


Fig. 1. Overview of the approach: A sample  $H$  is taken from the original graph  $G$ . The profile feature  $F(H)$  is created and projected to obtain the profile feature estimation  $\phi(F(H))$  (based on [18])

coverage  $coverage(S, G)$  of a characteristic set  $S$  in an RDF graph  $G$  is computed as

$$\frac{\sum_{p \in S} \text{multiplicity}(p, S) \cdot \text{count}(S)}{|G|} \quad (3)$$

For example, consider the characteristic set  $S_1 = \{ \text{lmbd:hasPhotoCollection}, \text{owl:sameAs} \}$  from our introductory example taken from LinkedMDB with

- $\text{count}(S_1) = 9838$ ,
- $\text{multiplicity}(\text{lmbd:hasPhotoCollection}, S_1) = 1$ ,
- $\text{multiplicity}(\text{owl:sameAs}, S_1) = 2$ ,
- $\text{coverage}(S_1, \text{LinkedMDB}) = 0.5\%$ .

The statistics indicate that 9838 entities in the LinkedMDB graph belong to  $S_1$  and each of those entities has *exactly* one `lmbd:hasPhotoCollection` and *on average* two `owl:sameAs` predicates. Moreover, the coverage indicates that 0.5% of the triples in LinkedMDB belong to entities with  $S_1$  as their characteristic set.

Finally, we introduce the notion of *exclusive* characteristic sets. An exclusive characteristic set occurs only once in an RDF graph and is defined as the following.

**Definition 3.2** (Exclusive Characteristic Sets [18]). Given the characteristic sets  $\mathcal{C}(G)$  of graph  $G$ . The set of exclusive characteristic sets  $\mathcal{E}(G)$  is defined as

$$\mathcal{E}(G) := \{ S \in \mathcal{C}(G) \mid \text{count}(S) = 1 \}$$

For example, in the LinkedMDB graph there exists only one entity with the predicates `lmbd:relatedBook` and `lmbd:language`. Therefore, the characteristic set  $S_4 = \{ \text{lmbd:relatedBook}, \text{lmbd:language} \}$  is an exclusive characteristic set.

Based on these definitions, the statistical aspects of characteristic sets can be combined to formally define the characteristic sets profile feature (CSPF) as follows.

**Definition 3.3** (Characteristic Sets Profile Feature (CSPF) [18]). Given a RDF graph  $G$ , the characteristic sets profile feature  $F(G)$  is a 3-tuple  $(\mathcal{C}, \mathcal{E}, m)$  with:

- $\mathcal{C} = \mathcal{C}(G)$ , the set of characteristic sets in  $G$ ,
- $c : \mathcal{C} \rightarrow \mathbb{N}$ , a function for the *count* as defined in Equation 1, and
- $m : \mathcal{E} \rightarrow \mathbb{R}^+$ , a function for the *multiplicity* as defined in Equation 2.

We now present the individual steps of our approach for estimating the CSPF of a given RDF graph.

*Overview of the approach* We propose a sampling-based approach that is shown in Figure 1. Given a graph  $G$ , the approach generates a sample  $H \subseteq G$  using an RDF graph sampling method. We present the sampling methods in Section 3.2. Thereafter, the approach computes the CSPF  $F(H)$  for the sample  $H$  (Section 3.3). Finally, a projection function  $\phi$  is applied to extrapolate the feature observed in  $H$  to an estimation  $\phi(F(H))$  of  $F(G)$  of the original graph. We propose different projection functions tailored to the CSPF in Section 3.4. Moreover, to assess the representativeness of the feature estimations, we propose a selection of structural and statistical similarity measures in Section 3.6.

### 3.2. Graph Sampling

The first step of our approach is obtaining a representative sample of the original RDF graph that provides a suitable foundation for estimating the profile feature. Before collecting data from a population in a sample, an appropriate sampling method needs to be chosen. Accordingly, we focus on sampling methods suitable for estimating the CSPF. Since characteristic sets capture the attributes of the entities in an RDF graph, we focus on entity-centered sampling methods in the following. Each entity (i.e., subject) is associated with exactly one characteristic set, thus, we define the population to be sampled from as the set of entities in the given graph:  $E := \{ s \mid \exists s, p, o \in G \}$ . The input of the sampling method is an RDF graph  $G$  and a sample size  $n$  and its output is a subgraph  $H$  induced by  $n$  entities of  $G$ . Let  $E' \subseteq E$  be the set of sampled

entities with  $|E^0| = n^0$ , then the sample  $H$  is given as  $H := \{s, p, o \mid s, p, o \in G \wedge s \in E^0\}$ . We present three entity-centered sampling methods differing in the probabilities of an entity being sampled. Thus, they allow for exploring different parts of the search space of possible characteristic sets during sampling.

*Unweighted Sampling* The unweighted sampling method selects  $n^0$  entities with equal probability from the population  $E$ . Thus, the probability  $Pr^1 e^0$  of  $e \in E$  being a part of the sample is independent of any other property of the entity with

$$Pr^1 e^0 = \frac{1}{|E|}.$$

*Weighted Sampling* The weighted sampling method is a biased sampling method, where the probability of an entity to be part of the sample is proportional to its out-degree. As a result, the more central an entity is in the graph with respect to its out-degree centrality [29], the more likely it is part of the sample. In this way, entities that appear in many triples of the graph in the subject position, have a higher probability of being selected. The out-degree of each entity  $e$  given by  $deg^1 e^0 := |\{s, p, o \mid s, p, o \in G\}|$ . The probability  $Pr^1 e^0$  of  $e \in E$  being sampled is given as

$$Pr^1 e^0 = \frac{deg^1 e^0}{|G|}.$$

*Hybrid Sampling* The hybrid sampling method combines the previous two sampling methods. From the original graph,  $\beta \cdot n^0$  entities are selected using the unweighted method and  $(1 - \beta) \cdot n^0$  entities using the weighted method. Accordingly, the probability  $Pr^1 e^0$  of entity  $e$  being selected is given as

$$Pr^1 e^0 = \beta \cdot \frac{1}{|E|} + (1 - \beta) \cdot \frac{deg^1 e^0}{|G|}, \quad \beta \in [0, 1].$$

The  $\beta$  parameter allows for favoring either the weighted or the unweighted method.

### 3.3. Profile Feature Creation

After obtaining a subgraph  $H$  using a sampling method, the characteristic sets profile feature  $F^1 H^0$  is computed. Similar to [25], we compute  $F^1 H^0$  by first sorting the triples in  $H$  by subjects and then iterating

all triples. While iterating the triples, we determine the characteristic set for each subject in  $H$  and compute the count and multiplicity values for them. [For details, please refer to Appendix A.](#)

### 3.4. Profile Feature Projection

Given the profile feature  $F^1 H^0$ , the goal of a projection function is to extrapolate the statistical properties observed in sample  $H$  to the target population, which is the original graph  $G$ . Given the profile feature  $F^1 H^0 = \langle c, m \rangle$ , the projection profile feature is applied to  $F^1 H^0$  to obtain an estimation of the original profile feature  $\hat{F}^1 G^0 = \phi^1 F^1 H^0$ . In the case of the characteristic sets profile feature (CSPF), the projection function aims to extrapolate the counts  $c$  of the observed characteristic sets to estimate the counts in the original graph. The multiplicity statistic  $m$  does not need to be extrapolated by the projection function, as it is a relative measure that captures the *mean* occurrence of a predicate in a characteristic set.

We now present two classes of projection functions for the count values of the characteristic sets in the sample. The first class, which we denote *basic projection functions*, only relies on information contained within the sample and the size of the original graph. The second class, which we denote as *statistics-enhanced projection functions* also relies on high-level information on the original dataset in addition to the information contained in the sample.

*Basic Projection Function* The basic projection function extrapolates the count values for the given characteristic sets profile feature  $F^1 H^0$  based on the relative size of the sample. The relative sample size  $r_t$  is given as the ratio of the number of triples in the original graphs and in the sample  $r_t := \frac{|G|}{|H|}$ . We define the projection function  $\phi_1$  as

$$\phi_1^1 F^1 H^0 := \langle r_t \cdot c, m \rangle$$

The assumption of the basic projection function is that the characteristic sets observed in a sample occur proportionally more often in the original graph. However, as exemplified in the introduction, the distributions of the counts may be potentially skewed which is not considered by this projection function. Moreover, this function neglects the fact that some characteristic sets might not have been sampled and merely *distributes* the proportional counts across the charac-



teristic sets in the sample. As a result, it is likely that the counts for the characteristic sets is overestimated, especially if just a small portion of characteristic sets from the original graph are captured. To address this shortcoming, we introduce two statistic-enhanced projection functions that leverage high-level information to reduce the probability of overestimating the counts.

**Statistics-enhanced Projection Functions** The second class of projection functions incorporates additional high-level information about the original graph. In particular, we consider the number of triples per predicate in the original graph as this high-level statistic. The rationale for this particular statistic is that it potentially yields a good trade-off between the effort to obtain the statistic and the benefit it provides for the projection function. The number of triples for predicate  $p^0$  is given by  $t^1 p^{00} := \text{count}(s, p^0, o^0)$ .

The first statistic-enhanced projection function,  $\phi_2$  applies a true upper bound for the counts of the characteristic sets in the following way.

$$\phi_2^1 F^1 H^{00} := \text{count}(s, p^0, o^0), \text{ with}$$

$$e^1 S_C^0 := \min_t r_t \cdot c^1 S_C^0, \min_{p^0 \in S_C} t^1 p^{00}$$

The idea of this projection function is that the predicate occurrence statistics of the original graph allows for limiting the estimated counts for characteristic sets containing that predicate. If a predicate  $p^0$  is part of a characteristic set  $S_C$ , then  $S_C$  may occur at most  $t^1 p^{00}$  times and consequently  $c^1 S_C^0 \leq t^1 p^{00}$ . This reduces the likelihood of overestimating counts without increasing the likelihood of underestimating them. The effectiveness of this upper bound, however, strongly depends on the predicates that occur in a given characteristic set. For instance, in the case that a characteristic set contains common predicates, such as `rdf:type` or `rdfs:label`, the upper bound determined by  $\phi_2$  might be rather high, even though the predicates potentially occur in a number of other characteristic sets as well. Therefore, we propose a second statistics-enhanced projection function  $\phi_3$  which *distributes* the upper bound for a predicate  $p^0$  by considering the sum of counts of the characteristic sets in which the predicate  $p^0$  occurs. To achieve this, we first sum up the counts of all characteristic sets which  $p^0$  occurs in as

$$S_C^{p^0} := \sum_{S_C^0 \in \tilde{S}_C^0} c^1 S_C^0$$

The projection function  $\phi_3$  then adjusts the upper bound by multiplying  $t^1 p^{00}$  with the ratio of the count  $c^1 S_C^0$  of the given characteristic set  $S_C$  and the sum of counts for all characteristic sets  $p^0$  occurs in  $S_C^{p^0}$ .

$$\phi_3^1 F^1 H^{00} := \text{count}(s, p^0, o^0), \text{ with}$$

$$e^1 S_C^0 := \min_t r_t \cdot c^1 S_C^0, \min_{p^0 \in S_C} t^1 p^{00} \frac{c^1 S_C^0}{S_C^{p^0}}$$

In contrast to  $\phi_2$ , this approach applies a stricter upper bound by considering all characteristic sets a predicate occurs in and reducing the upper bound accordingly. However, a drawback is that the projection function  $\phi_3$  increases the likelihood of underestimating the count of characteristic sets by assuming an equal distribution of the predicates.

Concluding this section on the projection functions, we want to note that further functions may be applied to potentially improve the estimation of the profile feature. For example, the characteristic sets sizes or additional statistics about the predicate distribution in the sample could be considered. However, we chose not to include them since they are likely to just produce accurate estimation under certain conditions and, therefore, do not generalize well for other datasets. Moreover, further improvements in the estimations do not necessarily imply the same improvements in the downstream applications that use the feature estimations.

### 3.5. Sampling RDF Graphs in Practice

The presented sampling approaches define the process for selecting entities to be sampled from the original graph but they do not encode *how* the samples are generated in practice. The implementation of the sampling approaches depends on the access methods to the given RDF graph. In the case of centralized RDF graphs, the sampling approach has access to the entire dataset and could potentially exploit existing data structures (e.g., indices) to efficiently access the entities in the graph. However, in decentralized scenarios as federations, the sampling methods have only access to parts of the RDF graphs and their implementation depends on the expressivity, capabilities, and configuration of the access interfaces. While it is out of the scope of this work to present specific implementations for each access interface, in the following, we provide insights into the considerations when implementing these sampling methods over remote SPARQL endpoints and Triple Pattern Fragments (TPF) servers.



*SPARQL Endpoints* The expressivity of endpoints allows for selecting the entities to be sampled and then obtaining the subgraph induced by those entities.

The unweighted sampling method can be implemented over a SPARQL endpoint as follows. (i) Query all unique subjects (i.e., with `SELECT` and `DISTINCT`). If the number of entities exceeds the number of answers allowed per response, the implementation can retrieve all the entities by ‘paginating’ the results by ordering them with `ORDER BY` and using `LIMIT` and `OFFSET`. (ii) Select  $n^0$  subjects from the resulting set of entities. In the case that the endpoint supports the `RAND`<sup>6</sup> function, it would suffice to obtain only  $n^0$  subjects in the query using the `LIMIT` keyword that are randomly ordered (`ORDER BY RAND()`).

In the weighted sampling method, the out-degree of an entity determines its probability to be sampled. To avoid computing the out-degree for each entity, it is possible to implement a simple heuristic that relies on the following assumption: The higher an entity’s out-degree, the more triples with it in the subject positions are present in the graph. Therefore, to obtain a weighted sample, the approach can randomly select triples from the graph and obtain the subjects of those triples. This can be achieved in endpoints using a query that matches any triple in the graph, a `LIMIT` of 1, and a random `OFFSET` value between 0 and the size of the graph  $|G|$ . This should be repeated until collecting  $n^0$  entities.

Finally, after the entities have been selected, the induced subgraph of the entities can be obtained using either a `CONSTRUCT` query or a `SELECT` query. That is, all triples where the entities are in the subject position will be constructed. This can be achieved with a `FILTER` or, if supported, with the `VALUES` keyword with the list of selected entities from the previous step to instantiate the entities in the query.

*Triple Pattern Fragments (TPF)* TPF servers support triple pattern-based querying of RDF graphs. Since TPF servers do not provide the same expressivity as SPARQL endpoint for accessing the RDF graphs, the sampling methods need to be implemented accordingly.

Because TPF servers do not support `SELECT` or `DISTINCT`, the implementation of the unweighted sampling method over TPF servers should be realized differently from the one discussed for SPARQL endpoints. To obtain entities in the RDF graph, the method could re-

quest individuals of classes using the `rdf:type` predicate. Thereafter, duplicate entities can be removed locally and the sample of size  $n^0$  can be drawn from these results. While this method disregards entities that do not belong to classes, it still allows for easily obtaining a subset of entities for the sample from TPF servers.

For implementing the weighted sampling method, a similar heuristic to the one presented for SPARQL endpoints can be followed. Instead of computing the out-degrees of each entity, the approach can 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. Random selection of triples is implemented over TPF servers by randomly selecting a page from the range 1 to the number of pages in the fragment, which can be obtained from the metadata `void:triples` and `hydra:itemPerPage` provided by the server. Then, from the triples in the selected page, randomly select one triple to obtain the entity in the subject position. This process should be repeated until obtaining  $n^0$  entities.

Finally, the induced subgraph of the entities can be obtained by querying all the triples where the subject corresponds to each selected entity.

### 3.6. Similarity Measures for Characteristic Set Profile Features

Finally, we define similarity measures that quantify the similarity between the estimated profile feature and the feature of the original graph. A higher similarity value indicates a more *representative* feature estimation. According to our problem definition (Def. 2.3) in Section 2.2, the goal is obtaining an estimator  $\hat{F}^1 G^0 = \phi^1 F^1 H^{0^0}$  for the characteristic  $F$  which maximizes the similarity  $\delta$  between the estimated and the original profile feature. As previously mentioned, methods to quantify the similarity depend on the particular profile feature. We now propose measures that are tailored to the characteristic sets profile feature (CSPF). Due to the diverse nature of the graph’s characteristics captured in the CSPF, we propose multiple measures to determine the similarity  $\delta^1 F^1 G^0, \hat{F}^1 G^{0^0}$  between the original CSPF  $F^1 G^0 = \langle c, m^0 \rangle$  and an estimated CSPF  $\hat{F}^1 G^0 = \langle \hat{c}, \hat{m}^0 \rangle$ . The proposed similarity measures capture both structural as well as statistical aspects of the CSPF. All measures take values in  $]0, 1[$  with larger values indicating a higher similarity. An overview of all similarity measures is given in Table 1.

<sup>6</sup><https://www.w3.org/TR/sparql11-query/#func-numeric>

Table 1

Overview of the similarity measures: The structural similarity measure capture the degree to which structural features of the original graph are represented. The statistical similarity measures focus on the *count* and *multiplicity* statistics of the characteristic sets.

Structural Similarity Measures				Statistical Similarity Measures	
Out-degree $\delta^{od}$ (4)	Predicate coverage $\delta^{pc}$ (5)	Absolute set coverage $\delta^{ac}$ (6)	Relative set coverage $\delta^{rc}$ (7)	Count similarity $\delta_{S_C}^{count}$ (8)	Multiplicity similarity $\delta_{S_C}^{multiplicity}$ (9)

**Structural Similarity Measures** The first category of similarity measures focuses on the structural information captured in the characteristic sets. We propose four measures that consider the mean out-degree, the predicates covered, and the absolute and relative number of covered characteristics sets. The *out-degree* similarity measure  $\delta^{od}$  determines how well the mean out-degree of the original graph is captured in the estimated profile feature. The mean out-degree  $\overline{deg}$  can be computed for a CSPF  $F^1G^0$  in the following way.

$$\overline{deg}^1 F^1 G^0 := \frac{|G|}{|S_C| \cdot |S_C^0|}$$

Note that  $\overline{deg}^1 \hat{F}^1 G^{00}$  is computed analogously for the estimated features using  $H$ ,  $\hat{\cdot}$ , and  $\hat{c}$  instead. The out-degree similarity measure is computed as

$$\delta^{od}^1 F^1 G^0, \hat{F}^1 G^{00} := 1 - \frac{|\overline{deg}^1 F^1 G^0 - \overline{deg}^1 \hat{F}^1 G^{00}|}{\max\{\overline{deg}^1 F^1 G^0, \overline{deg}^1 \hat{F}^1 G^{00}\}} \quad (4)$$

The second structural measure focuses on the predicates captured in the characteristic sets of the CSPF. The *predicate coverage* similarity  $\delta^{pc}$  is calculated as the ratio of the number of predicates covered in the estimation and the number of predicates in the original profile feature.

$$\delta^{pc}^1 F^1 G^0, \hat{F}^1 G^{00} := \frac{|\{p \in P \mid p \in S_C \wedge S_C^0 \wedge g\}|}{|\{p \in P \mid p \in S_C \wedge S_C^0 \wedge g\}|} \quad (5)$$

A higher ratio indicates that a larger portion of the terminology used to describe the entities in the graph is captured in the estimated CSPF.

The final two structural measures address characteristic sets themselves and determine how many of them are covered in the CSPF  $\hat{F}^1 H^0$ . First, the *absolute set coverage* similarity  $\delta^{ac}$  is computed as the ratio of the

number of characteristic sets in the estimation to those in the original statistic profile:

$$\delta^{ac}^1 F^1 G^0, \hat{F}^1 G^{00} := \frac{|\hat{G}|}{|G|} \quad (6)$$

However, this measure does not reflect the *importance* of the characteristic sets captured in estimated CSPF with respect to their *coverage*. A characteristic set  $S_C$  can be considered to be more important if it represents a larger portion of a given graph  $G$ . Therefore, the *relative set coverage* similarity  $\delta^{rc}$  of  $\hat{F}^1 G^0$  is calculated as the number of triples induced in the original graph with all characteristic sets in  $\hat{\cdot}$  of the estimation.

$$\delta^{rc}^1 F^1 G^0, \hat{F}^1 G^{00} := \frac{|\{S_C^0 \wedge coverage^1 S_C, G^0\}|}{|G|} \quad (7)$$

Note that the characteristic sets of the estimation  $\hat{\cdot}$  are taken as the basis for computing the measure, while their coverage (c.f. Equation (3)) is computed based on the original graph, i.e.  $p \in S_C \wedge m^1 p, S_C^0 \wedge c^1 S_C^0$ . In this way,  $\delta^{rc}$  reflects the importance of the characteristic sets captured in the sample. For example, consider an RDF graph  $G$  with two characteristic sets  $S_1$  and  $S_2$ , where  $S_1$  covers 90% and  $S_2$  10% of all triples in  $G$ . Assume that we obtain two estimations with  $\hat{1} = \{S_1\}$  and  $\hat{2} = \{S_2\}$ . Both  $\hat{1}$  and  $\hat{2}$  have the same absolute set coverage  $\delta^{ac} = 0.5$  because they both contain one characteristic set. If we take the relative set coverage into consideration, we can determine that  $\hat{1}$  should be considered more representative for the original graph  $G$  as its characteristic set  $S_1$  covers 90% of its triples.

**Statistical Similarity Measures** The second category of similarity measures focuses on the statistical aspects covered in the feature estimations, namely the *counts* of the characteristic sets and the *multiplicity* of predicates. Depending on the distribution of the characteristics sets in the original graph, an accurate estimation of both counts and multiplicity values can be challenging. As a consequence, for some characteristic sets, the estima-

tions can be accurate, while for others large estimation errors can occur. We propose statistical similarity measures that capture the estimation accuracy on the level of individual characteristics sets. To better understand the estimation error distribution across all characteristic sets in a sample, different aggregation functions, such as mean or median, may be applied afterward.

We propose a similarity measure for the count values that is the inverse of the count estimation q-errors. The q-error [17] is the maximum of the ratios between true and estimated count and vice versa. Given a profile feature  $F^1G^0$  and an estimation  $\hat{F}^1G^0$ , the count value q-errors are computed for all  $S_C$  as

$$\text{q-error}_{c^1S_C^0} := \max \left\{ \frac{c^1S_C^0}{\hat{c}^1S_C^0}, \frac{\hat{c}^1S_C^0}{c^1S_C^0} \right\}.$$

Higher q-errors indicate a higher discrepancy between the true value and the estimation, and a q-error of 1 indicates that the estimation is correct. We take the inverse of the q-error to map it to the similarity measure range  $]\!]\!], 1\llcorner$

$$\delta_{S_C}^{\text{count}^1F^1G^0, \hat{F}^1G^0} := \frac{1}{\text{q-error}_{c^1S_C^0}}. \quad (8)$$

Note that the q-error only measures the magnitude of the estimation error but does not indicate whether values are over- or underestimated. As a result, when aggregating the similarity measures across all characteristic sets of a given feature estimation  $\hat{\phantom{F}}^1G^0$ , this property avoids overestimated counts to cancel out underestimated counts and vice versa.

The proposed similarity measure for the predicate multiplicities is also based on the q-error and is defined on the level of characteristic sets. Given a profile feature  $F^1G^0$ , an estimation  $\hat{F}^1G^0$  and a characteristic set  $S_C$  in  $\hat{F}^1G^0$ , this q-error is computed as the mean of the multiplicity q-errors of all predicates in  $S_C$

$$\text{q-error}_{m^1S_C^0} := \frac{1}{|S_C|} \bigcirc_{p \in S_C} \max \left\{ \frac{\hat{m}^1p, S_C^0}{m^1p, S_C^0}, \frac{m^1p, S_C^0}{\hat{m}^1p, S_C^0} \right\}.$$

Analogously, to the count values, the similarity measure for the multiplicities is the inverse of the q-error

$$\delta_{S_C}^{\text{multiplicity}^1F^1G^0, \hat{F}^1G^0} := \frac{1}{\text{q-error}_{m^1S_C^0}}. \quad (9)$$

In summary, the characteristic sets in a CSPF captures various characteristics of RDF graphs, ranging from out-degree distribution to predicate co-occurrence. As a result, a combination of similarity measures needs to be taken into account to assess the representativeness of an estimation of a CSPF. To this end, we have proposed four structural and two statistical similarity measures. The importance of the individual measures depends on the application. In the following section, we present one potential application of the CSPF estimations in federated query planning.

#### 4. Federated Query Planning using Characteristic Sets Profile Feature Estimations

We investigate the applicability and effectiveness of the characteristic set profile feature (CSPF) estimations in a downstream application. In particular, we focus on federated SPARQL query planning motivated by previous works, which studied how characteristic sets can be exploited in query plan optimization in both centralized and decentralized scenarios. In decentralized query processing, it has been shown how characteristic sets can be leveraged to obtain efficient federated query execution plans that outperform existing state-of-the-art federated query plan optimizers in terms of query execution time and data transfer [16]. However, the computation of the complete characteristics sets profile feature can be challenging in such decentralized scenarios as it requires obtaining the entire datasets of all members in the federation and, subsequently, computing the profile feature. Consequently, we study how federated query planning approaches can benefit from the proposed characteristic sets profile feature estimations that are based on samples from the original graphs.

We propose a novel heuristic for federated query planning to obtain efficient federated query plans based on estimated CSPFs. We cannot directly apply an existing approach, such as Odyssey [16], due to the following reason. First, Odyssey relies on having access to the entire datasets to obtain the complete characteristic sets and the query planner assumes complete information. Hence, the planner relies on complete statistics for query decomposition, join ordering, and cardinality estimation and it does not handle potentially missing statistics in those steps. Second, the Odyssey approach additionally leverages both the characteristic pair (CP) and the federated characteristic pair (FCP) statistics to capture how characteristic sets are linked within and

1 entities are linked across the datasets of the members in  
 2 the federation [16]. In this work, we focus only on char-  
 3 acteristic sets and assume estimated statistics. There-  
 4 fore, we propose a novel query planner based on these  
 5 restrictions. Still, our query planner is based on exist-  
 6 ing approaches for characteristic sets-based cardinality  
 7 estimation and query planning [14, 16, 17].

8 We begin by introducing the necessary concepts and  
 9 notation from SPARQL, which is the recommended  
 10 query language for RDF graphs. Following the notation  
 11 of Schmidt et al. [30], where  $V$  is the set of variables that  
 12 is disjoint with  $I$ ,  $B$ , and  $L$ , we can define a SPARQL  
 13 expression as follows.

14 **Definition 4.1** (SPARQL Expression [30]). A SPARQL  
 15 expression is an expression that is build recursively as  
 16 follows.

- 17
- 18 (1) A triple pattern  $tp \ 2 \ ^1I \ [ \ V^0 \ ^1I \ [ \ V^0 \ ^1I \ [ \ V \ [ \ L^0$   
 19 is an expression,
  - 20 (2) If  $P_1, P_2$  are expressions and  $R$  is a filter condition,  
 21 then  $P_1 \ \text{FILTER} \ R, P_1 \ \text{UNION} \ P_2, P_1 \ \text{OPT} \ P_2,$  and  
 22  $P_1 \ \text{AND} \ P_2$  are expressions.

23 In our query planning approach, we focus on Basic  
 24 Graph Patterns which we define as follows.

25 **Definition 4.2** (Basic Graph Pattern (BGP)). Let  $P_1$   
 26 and  $P_2$  be SPARQL expressions. A basic graph pat-  
 27 tern  $P$  is defined recursively as an expression of  
 28 the form  $P_1 \ \text{AND} \ P_2$ , where  $P_1$  and  $P_2$  are either a  
 29 triple pattern or conjunctions of triple patterns (i.e.,  
 30  $tp_1 \ \text{AND} \ \dots \ \text{AND} \ tp_n$ ).

31 Next, we denote the set of predicates in a BGP  $P$  as  
 32  $\text{preds}^1 P^0$ . For simplicity, we extend the notation and  
 33 also consider a BGP  $P = ^1tp_1 \ \text{AND} \ \dots \ \text{AND} \ tp_j^0$  as the  
 34 set of triple patterns it consists of  $P = \{tp_1, \dots, tp_j\}$ .

35 Moreover, we denote the evaluation of an expression  
 36  $P$  over a graph  $G$  by  $\mathcal{J}P^0_G$ . The evaluation of a SPARQL  
 37 expression over an RDF graph yields a set of solution  
 38 mappings  $\mu$ , where  $\mu$  is a partial function  $\mu : V \ !$   
 39  $I \ [ \ B \ [ \ L$  that maps a subset of variables  $V$  to RDF terms  
 40 [31]. Finally, we consider set semantics for evaluating  
 41 SPARQL expression over RDF graphs and federations  
 42 of RDF graphs [30, 32].

#### 43 4.1. CSPF Estimation-based Federated Query 44 Planner

45 We propose a heuristic-based query planner to deter-  
 46 mine efficient query plans based on CSPF estimations.  
 47 We adapt the notation from [32] and define a federa-

1 tion of SPARQL endpoints (sources) as  $F = ^1EP, g^0$ ,  
 2 where  $EP = \{ep_1, \dots, ep_k\}$   $I$  is a set of IRIs that  
 3 identify the SPARQL endpoints, and  $g : I \ !$  a func-  
 4 tion mapping each endpoint  $ep_i$  to its (default) RDF  
 5 graph. Given a basic graph pattern  $P$  and a federation  
 6  $F$ , the federated query planner performs source selec-  
 7 tion, query decomposition, and join ordering to obtain  
 8 a query plan. We now detail how the planner leverages  
 9 the estimated CSPF in all three steps for basic graph  
 10 patterns (BGPs).

11 *Source Selection and Query Decomposition* The  
 12 source selection and query decomposition step deter-  
 13 mine which subexpressions of a given BGP  $P$  should  
 14 be evaluated at which members of the federation  $F$ .  
 15 The output of this process is a query decomposition,  
 16 which we define as follows.

17 **Definition 4.3** (Query Decomposition). Given a BGP  
 18  $P$  and a federation  $F = ^1EP, g^0$ , a query decomposition  
 19  $D = \{d_1, \dots, d_m\}$  is a set of tuples  $d_i = ^1SE_i, R_i^0$   
 20 where

- 21
- 22 –  $SE_i$  is a subexpression of  $P$  such that  $\bigoplus_{i=1, \dots, m} SE_i =$   
 23  $P$ , and
  - 24 –  $R_i \ EP$  is a non-empty set of sources where  $SE_i$   
 25 is evaluated.

26 The challenge of query decomposition based on es-  
 27 timated statistics is the fact that the planner does not  
 28 know what is not captured in the estimations. For ex-  
 29 ample, this is the case when determining the relevant  
 30 sources. In the following, we assume a source  $ep_i$  to  
 31 be relevant, if there is a subexpression  $P^0$  in the BGP  
 32  $P$  such that the evaluation of  $P^0$  over  $g^1 ep_i^0$  produces  
 33 solution mappings. That is,  $\exists P^0 \ P : \mathcal{J}P^0_{g^1 ep_i^0} < ; .$

34 Our query planner leverages the estimated CSPF to  
 35 determine these relevant sources. To this end, it follows  
 36 a conservative strategy to distinguish between relevant  
 37 and non-relevant sources. It assumes all sources to be  
 38 relevant, in case none of them is relevant according to  
 39 the estimated (potentially incomplete) statistics.

40 The source selection and query decomposition pro-  
 41 cess are shown in Algorithm 1. The goal of the ap-  
 42 proach is to merge multiple triple patterns into larger  
 43 subexpressions to reduce the number of subexpres-  
 44 sions to be evaluated over the sources. Our planner  
 45 focuses on subject-based star-shaped basic graph pat-  
 46 tern. That is, BGPs where all triple patterns share the  
 47 same RDF term or variable in the subject position  
 48 [33]. For each such pattern  $SSP$  of the given BGP  $P$   
 49

(Line 2), the decomposer leverages the characteristic sets to determine whether any subset  $SSP^0$  of  $q$  triple patterns can be evaluated jointly at any source (Line 6). This is the case, if for a source  $ep \in EP$  with  $\hat{F}^1 g^1 ep^{oo} = \langle \hat{C}, \hat{c}, \hat{m}^o \rangle$ , there exists at least one characteristic set  $\hat{S} \subseteq \hat{C}$  such that all predicates of  $SSP^0$  are contained within  $\hat{S}$ :  $\text{preds}^1 SSP^0 \subseteq \hat{S}$  (Line 10). In this case, the subquery  $SSP^0$  is added to the decomposition with the corresponding relevant sources  $R^0$  and the process continues with the remaining triple patterns of the original star-shaped BGP:  $SSP \setminus SSP^0$  (Line 15). If  $SSP^0$  only consists of a single triple pattern and no relevant source could be determined (Line 17), i.e.  $R = \emptyset$ , all sources need to be considered as relevant  $R = EP$ . This is the case, if either the predicate  $p$  in the triple pattern does not occur in a characteristic set of any source or if  $p$  is a variable, i.e.  $p \in V$ . However, for the cardinality estimation in the next step, we need to be able to distinguish this case from the case where all sources are *known* to be relevant according to the estimated CSPF. For example, all sources could be *known* to be relevant for common terminology, such as `rdf:type`. Therefore, if the estimated statistics do not allow for determining which sources are relevant, we set the relevant sources in the decomposition to be the empty set (Line 18). Note that because potentially not all predicates are covered in the sample from which the estimated CSPF is derived, we cannot assume that none of the sources is relevant.<sup>7</sup>

**Cardinality Estimation** Given a query decomposition, the query planner estimates the cardinalities of all subexpressions in the decomposition. Given these estimations, the planner determines an efficient join order for the subexpressions. To this end, the subexpressions are ordered by increasing cardinality to minimize the number of intermediate results to be transferred during query execution.

For a given subexpression  $SE_i$  in a decomposition  $D$ , the cardinality estimation distinguishes two cases:

- (1)  $R_i < \emptyset$ : There exists a set of relevant sources according to the estimated statistics, or
- (2)  $R_i = \emptyset$ : All sources need to be considered relevant according to the estimated statistics.

For the first case, we apply the STARJOINCARDINALITY algorithm proposed by Neumann and Moeroette

<sup>7</sup>Alternatively, the planner could use ASK queries to determine which source is relevant. However, as we want to investigate the effectiveness of the estimated statistics, the proposed planner only relies on the estimated CSPF.

---

### Algorithm 1: Query Decomposer

---

```

Input: BGP  $P$ , Federation
           $F = \langle EP = \{ep_1, \dots, ep_k\}, d^o \rangle$ 
Output: Decomposition  $D$  for the BGP  $P$ , annotated with
          the set of relevant sources from  $F$ 
1  $D \leftarrow \emptyset$ ;
  // Iterate all star-shaped subqueries in  $P$ 
2 for  $SSP \in \text{getSubjectStars}^1 P^o$  do
3    $q \leftarrow jSSP$ 
4    $todo \leftarrow SSP$ 
5   while  $|todo| > 0$  do
6     for  $SSP^0 \in \text{subsets}^1 SSP, q^o$  do
7        $R^0 \leftarrow \emptyset$ ;
8       for  $ep \in EP$  do
9          $\langle \hat{C}, \hat{c}, \hat{m}^o \rangle = \hat{F}^1 g^1 ep^{oo}$ 
10        if  $\exists \hat{S} \subseteq \hat{C} : \text{preds}^1 SSP^0 \subseteq \hat{S}$  then
11           $R^0 \leftarrow R^0 \cup \{ep\}$ 
12        end
13        if  $|R^0| > 0$  then
14           $D \leftarrow D \cup \langle f^1 SSP^0, R^{0o} \rangle$ 
15           $todo \leftarrow todo \setminus SSP^0$ 
16           $q \leftarrow jtodo$ 
17        if  $|R^0| = 0 \wedge q = 1$  then
18           $D \leftarrow D \cup \langle f^1 SSP^0, \emptyset^o \rangle$ 
19        end
20       $q \leftarrow q \cup 1$ 
21    end
22 end
23 return  $D$ 

```

---

[14] to estimate the cardinality for each source  $ep$  according to the estimated CSPF  $\hat{F}^1 g^1 ep^{oo}$ . As suggested in [14], we also use the absolute predicate multiplicity to compute the conditional selectivity in the case of a bound object as

$$\text{sel}^1 ?o = x \text{ } j?p = p^o = \frac{1}{\text{count}^1 S_C^o \cdot \text{multiplicity}^1 p, S_C^o}$$

We then sum up the estimations to aggregate the cardinalities across all sources:

$$\text{card}^1 SE_i, R_i^o = \bigoplus_{ep \in R_i} \text{STARJOINCARDINALITY}^1 \hat{F}^1 g^1 ep^{oo}, SE_i^o$$

In the second case ( $R_i = \emptyset$ ), the subexpression consists of a single triple pattern according to the decomposition algorithm and all sources need to be considered relevant for that triple pattern. In this case, we leverage the information from the estimated CSPF to estimate the expected number of solutions produced by the triple pattern. We distinguish the following types of triple patterns to estimate their cardinalities:



Type I:  $^1s, p, ?o^0$

If both subject and predicate are bound in the triple pattern, we estimate the cardinality as the mean multiplicity values for all predicates and characteristic sets in all sources of the federation  $F$ .

Type II:  $^1s, ?p, ?o^0$

If just the subject is bound, we estimate the cardinality as the mean of the mean out-degrees  $\overline{deg}_s$  of all sources:

$$\frac{1}{|EP|} \sum_{ep \in EP} \overline{deg}_s \cdot \hat{F}^1 g^1 e p^{000}$$

Type III:  $?s, p, ?o^0$

If just the predicate is bound, we assume that the predicate occurs less frequently than the least frequent predicate in all estimated CSPF. The rationale for this is, if predicate  $p$  would occur more frequently, it would be very likely to be sampled. Thus, it can be assumed to be less frequent than any predicate that was sampled.

Type IV:  $(?s, ?p, ?o)$

The estimated cardinality, is the sum of the sizes of the graphs in the federation:

$$\sum_{ep \in EP} |g^1 e p^0|$$

For all remaining types of triple patterns, we estimate the cardinality to be 1.

**Join Ordering and Query Plan Generation** After estimating the cardinalities of all subexpressions in the decomposition, the query planner iteratively builds a left-deep plan by joining subexpressions ordered by increasing cardinality. In this query plan, each tuple  $^1SE_i, R_i^0$  of the query decomposition is evaluated as the union of  $SE_i$  over all sources in  $R_i$ . The resulting solution mappings are joined for all tuples in the decomposition. This corresponds to the evaluation of the following algebraic expression.

$$\bowtie_{^1SE_i, R_i^0} \text{D} \left[ \bigcup_{r_j \in R_i} \text{J} SE_i \right]_{r_j^0}$$

**Limitations** Note that the proposed query planner does not guarantee to obtain query plans that produce complete answers. This is due to two main reasons.

First, the planner may fail to identify a source as relevant in the case that the relevant characteristic set is missing in the CSPF estimation. While the planner assumes all sources to be relevant in the case that none of the estimations include a matching predicate, it might be the case that the a CSPF estimation for one source is missing the relevant information and for another sources it does not.

Second, the greedy nature of our query decomposition approach presented in Algorithm 1 may produce query decompositions that yield incomplete results. For example, the decomposer may find a star-shaped sub-query  $SSP^0$  of  $SSP$  with size  $q$  whose predicates are in the characteristic set of one source. In this case,  $|R^0| > 0$  and the decomposer continues with subset of size  $q^0 = |todo \cap SSP^0|$  (Line 15 and Line 16), even though there might be a subset  $|SSP^{00}|$  of  $|SSP^0|$  with  $|SSP^{00}| = q$  and  $q^0 < q < q$  for which there is a relevant source. In this case, the decomposer does not identify this relevant source. This greedy approach reduces the computational effort as fewer subsets need to be considered. Yet, the algorithm may be adjusted to consider all true subsets of  $SSP$  if necessary.

## 5. Evaluation

We empirically evaluate our contributions in two experimental studies. The first study focuses on analyzing the different components of our proposed approach to estimate the Characteristic Sets Profile Feature (CSFP) presented in Section 3. The second study aims to evaluate our CSPF estimations-based federated query planner which we presented in Section 4. We use the well-known federated querying benchmark FedBench [31] to investigate the effectiveness of our query planner and therefore, investigate the RDF graphs from FedBench in the first part of the evaluation as well. All raw experimental results are provided online under the DOI 10.5281/zenodo.4507242. The source code is provided on GitHub: (i) the sampling implementation<sup>8</sup>, and (ii) the query planner<sup>9</sup>.

<sup>8</sup>[https://github.com/Lars-H/hdt\\_sampler](https://github.com/Lars-H/hdt_sampler)

<sup>9</sup>[https://github.com/Lars-H/federated\\_crop](https://github.com/Lars-H/federated_crop)



Table 2  
 Characterization of the FedBench RDF graphs studied in the experiments.

	RDF Graph	General Properties						Characteristic Sets			
		# Triples	# Subj.	# Pred.	# Obj.	$\overline{deg}$	$deg_{std}$	Sj	$\frac{ Sj }{\#Subjects}$	$\frac{ S^1_j }{ Sj }$	AUC
Cross Domain	DBpedia	42 855 253	9 495 865	1063	13 636 604	4.51	6.44	160 062	1.69 %	66.31 %	95.23 %
	GeoNames	107 950 085	7 479 714	26	35 799 392	14.43	2.81	673	0.01 %	19.76 %	99.07 %
	Jamendo	1 049 647	335 925	26	440 686	3.12	4.31	42	0.01 %	11.9 %	92.38 %
	LinkedMDB	6 147 996	694 400	222	2 052 952	8.85	5.31	8459	1.22 %	62.64 %	97.65 %
	NY Times	335 197	21 666	36	191 537	15.47	28.73	47	0.22 %	19.15 %	91.14 %
	SW Dog Food	103 465	9948	118	35 520	10.40	10.24	569	5.72 %	37.96 %	88.43 %
Life Science	ChEBI	4 772 706	50 477	28	772 138	94.55	1853.09	978	1.94 %	39.57 %	96.71 %
	Drugbank	517 023	19 693	119	276 142	26.26	30.33	3419	17.36 %	79.7 %	81.3 %
	KEGG	1 090 830	34 260	21	939 258	31.84	165.88	67	0.2 %	11.94 %	92.83 %

### 5.1. Characteristic Sets Profile Feature Estimation Evaluation

We first empirically evaluate the different components of our approach to estimate the Characteristic Sets Profile Feature (CSPF). In the evaluation, we focus on the following core questions:

- Q1** How do different sampling sizes influence the similarity measures?
- Q2** What is the impact of different sampling methods on the similarity measures?
- Q3** What are the effects of leveraging additional statistics in the projection functions?
- Q4** How do different characteristics of the RDF graph influence the estimation?

We first present the setup of our experiments and analyze the results in Sections 5.1.1 and 5.1.2. Based on our findings, we answer the core questions in the final discussion in Section 5.1.3.

**RDF Graphs** We select the six cross-domain and three Life Science RDF graphs from the FedBench benchmark [31] for our evaluation. An overview of the graphs' properties is given in Table 2. The graphs differ in their general properties: the number of triples, the number of distinct subjects, predicates, and objects as well as the out-degree distribution. Moreover, the graphs differ with respect to their characteristic sets. The number of characteristic sets ranges from 42 to more than 160 000. Theoretically, the number of potential characteristic sets in a graph is bound by the power set of its predicates. In practice, however, the number of distinct subjects in the graph is a stricter upper bound. Therefore, we provide the ratio of characteristic sets

and distinct subjects ( $|Sj| \cdot \#Subjects \gg \%$ ) as a measure of the characteristic sets' diversity (higher values indicating a higher diversity). Additionally, the graphs differ in the proportion of exclusive characteristic sets. These exclusive characteristic sets, which only occur once in the graph, introduce further challenges when estimating the CSPF. If an exclusive characteristic set is sampled, the projection function likely overestimates its counts. In principle, it is unlikely to sample exclusive characteristic sets as they only occur once. However, in graphs with a large portion of those sets, the likelihood of sampling and overestimating those exclusive sets increases. Depending on the application of the estimations, it is potentially less important to correctly project the exclusive characteristic sets due to their lower coverage in comparison to other characteristic sets.

Figure 2 shows the cumulative coverage curve of the characteristic sets for all graphs. The curve plots the cumulative coverage of the characteristics sets in a graph sorted by decreasing coverage (c.f. Equation (3)).

$$f^1 x^0 = \overset{\circ}{\circ} \text{coverage}^1 S_C, G^0, \text{ with}$$

$$S_C \overset{x}{2} \overset{x}{c}$$

$$x := f S_C \overset{1}{2} \overset{1}{c} G^0 \text{ j rank}^1 S_C^0 \overset{1}{c} xg,$$

where  $\text{rank}^1 S_C^0$  computes the relative rank of the characteristic sets by decreasing coverage. The curve allows for visualizing the distribution of characteristic sets in a graph. A diagonal line indicates all characteristic sets covering the same number of triples. The stronger the curve is bent towards the upper left corner the more unevenly is the coverage of the characteristic sets, i.e., a few characteristic sets covering a large portion of the graph and many covering a small fraction

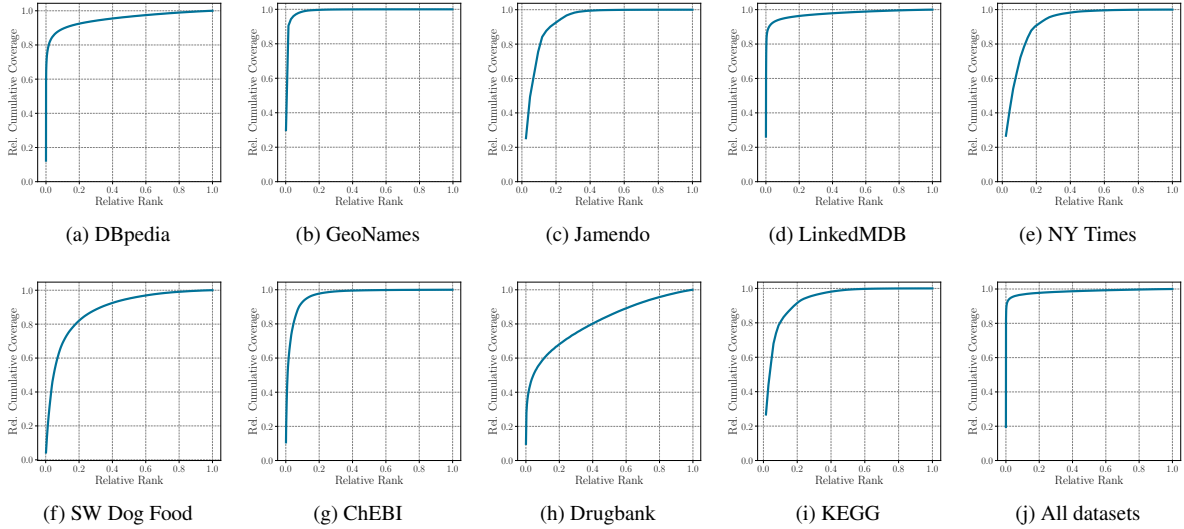


Fig. 2. The cumulative relative coverage curve shows the ratio of triples covered with respect to the characteristic sets ordered by decreasing relative coverage.

of the graph. For instance, comparing GeoNames in Fig. 2b and NY Times in Fig. 2e, we observe that in both graphs the characteristic set with the highest coverage covers 30% of all triples in the graph. However, the distribution of characteristic sets differs as in GeoNames 20% of the characteristics sets cover 90% of all triples, while in NY Times they only cover 80%. We use the area under the cumulative coverage  $AUC$  to quantify this property (cf. Table 4).

**Sampling Methods** We study the presented unweighted, weighted, and hybrid sampling methods. We chose  $\beta = 0.5$  for the hybrid sampling method equally balancing the weighted and unweighted method. We investigate four different sample sizes defined relative to the total number of entities  $|E|$  in the graph. We chose  $n^0 = \{0.1\%, 0.5\%, 1\%, 5\%\} |E|$  (Note:  $10\% = 1\%$ ). We generate 30 samples per RDF graph, sampling method, and sample size resulting in a total of  $30 \cdot 9 \cdot 3 \cdot 4 = 3240$  samples. For the sake of comparability, we reuse the samples and CSPF estimations created in this evaluation as the basis for the evaluation of our federated query planner.

### 5.1.1. Structural Similarity Measures

Table 3 shows the structural similarity measures out-degree  $\delta^{od}$ , predicate coverage  $\delta^{pc}$ , absolute set coverage  $\delta^{ac}$ , and relative set coverage  $\delta^{rc}$  for the different sampling methods and sample sizes. As descriptive statistics of the samples, we additionally computed the ratios of sampled triples and triples in the origi-

nal graph  $|H| \cdot |G|$  [%]. Comparing the different sample sizes (**Q1**), the results show an improvement in the similarity measures with an increasing sample size in the majority of cases. More importantly, we observed that the magnitude of the improvements depends on the properties of the original graphs. For example, the overall smallest improvements are achieved for GeoNames, where the similarity w.r.t  $\delta^{od}$ ,  $\delta^{pc}$ , and  $\delta^{rc}$  only slightly improves with larger sample sizes. However, for GeoNames the similarity measures are on average already on a high level in comparison to the other graphs, showing the best overall results for  $\delta^{od}$  and  $\delta^{rc}$ . Combining the results with the distribution of characteristic sets in the original graph, we find that GeoNames shows the highest  $AUC$ , the smallest characteristic set diversity, and a low exclusive set ratio. In contrast, the lowest relative set coverage  $\delta^{rc}$  values are observed for Drugbank, which also has the smallest  $AUC$  and largest characteristic sets diversity.

Interestingly, there are also few cases where a higher sample size yields worse similarity results. For instance, the out-degree similarity  $\delta^{od}$  decreases for KEGG and does not improve for ChEBI with the weighted sampling method. Again, the properties of the original graphs help explain those results (**Q4**). In particular, the dispersion of the out-degree distribution provides further insights. We can measure the dispersion with the coefficient of variation:

Table 3

Mean similarity values  $\delta^{od}$ ,  $\delta^{pc}$ ,  $\delta^{ac}$ ,  $\delta^{rc}$  (higher is better) and mean sampled triples ratio  $|H| \cdot |G|$  in per mille (‰) by sample size and sampling method. Best values per RDF graph and similarity measure are indicate in **bold**.

		0.1 ‰			0.5 ‰			1.0 ‰			5.0 ‰		
		unweighted	hybrid	weighted	unweighted	hybrid	weighted	unweighted	hybrid	weighted	unweighted	hybrid	weighted
Sample size $ H  \cdot  G $ [‰]	DBpedia	0.10	0.20	0.30	0.50	1.01	1.52	1.00	2.01	3.04	5.00	10.05	15.09
	GeoNames	0.10	0.10	0.10	0.50	0.51	0.52	1.00	1.02	1.04	5.00	5.09	5.19
	Jamendo	0.10	0.18	0.27	0.50	0.95	1.38	0.99	1.90	2.91	4.97	9.56	14.09
	LinkedMDB	0.10	0.12	0.13	0.50	0.59	0.69	0.99	1.19	1.37	4.99	5.89	6.78
	NY Times	0.10	0.17	0.53	0.38	1.45	2.28	0.99	2.33	3.93	5.24	13.15	22.18
	SW Dog Food	0.08	0.09	0.24	0.36	0.63	0.67	0.92	1.26	1.60	4.76	6.72	8.93
	ChEBI	0.03	13.59	31.58	0.30	74.45	141.75	1.15	136.84	214.70	4.23	351.39	488.45
	Drugbank	0.03	0.06	0.13	0.47	0.68	1.00	0.92	1.55	2.31	4.93	8.24	11.69
	KEGG	0.06	1.18	1.97	0.61	6.58	15.28	0.94	15.28	27.97	5.39	61.19	106.77
Out-degree $\delta^{od}$	DBpedia	0.96	0.50	0.33	0.98	0.50	0.33	<b>1.00</b>	0.50	0.33	<b>1.00</b>	0.50	0.33
	GeoNames	0.99	0.98	0.96	<b>1.00</b>	0.98	0.96	<b>1.00</b>	0.98	0.97	<b>1.00</b>	0.98	0.96
	Jamendo	0.86	0.61	0.38	0.93	0.54	0.36	0.95	0.53	0.35	<b>0.98</b>	0.52	0.36
	LinkedMDB	0.94	0.87	0.74	0.98	0.85	0.73	0.98	0.84	0.73	<b>0.99</b>	0.85	0.74
	NY Times	0.64	0.56	0.39	0.72	0.43	0.24	0.74	0.48	0.28	<b>0.86</b>	0.39	0.23
	SW Dog Food	0.55	0.57	0.54	0.69	0.68	0.65	0.81	0.73	0.62	<b>0.91</b>	0.75	0.56
	ChEBI	0.23	0.04	0.01	0.35	0.01	0.00	0.35	0.01	0.00	<b>0.51</b>	0.01	0.01
	Drugbank	0.38	0.41	0.50	0.81	0.70	0.48	0.79	0.66	0.43	<b>0.90</b>	0.61	0.43
	KEGG	0.44	0.24	0.13	0.56	0.10	0.04	0.57	0.08	0.04	<b>0.79</b>	0.08	0.05
Predicate coverage $\delta^{pc}$	DBpedia	0.20	0.30	0.35	0.37	0.50	0.55	0.46	0.58	0.63	0.65	0.76	<b>0.80</b>
	GeoNames	0.96	0.97	0.98	0.97	0.99	0.99	0.97	0.99	<b>1.00</b>	0.99	<b>1.00</b>	<b>1.00</b>
	Jamendo	0.58	0.73	0.75	0.87	0.95	0.96	0.95	0.97	0.99	<b>1.00</b>	<b>1.00</b>	<b>1.00</b>
	LinkedMDB	0.32	0.34	0.34	0.46	0.47	0.46	0.53	0.54	0.53	<b>0.73</b>	<b>0.73</b>	0.71
	NY Times	0.44	0.49	0.55	0.73	0.91	0.95	0.87	0.96	0.98	0.98	<b>1.00</b>	<b>1.00</b>
	SW Dog Food	0.05	0.06	0.09	0.15	0.19	0.20	0.27	0.27	0.29	0.48	0.47	<b>0.49</b>
	ChEBI	0.61	0.68	0.70	0.78	0.87	0.90	0.85	0.94	0.95	0.98	<b>1.00</b>	<b>1.00</b>
	Drugbank	0.12	0.18	0.31	0.64	0.75	0.85	0.76	0.89	0.93	0.95	0.96	<b>0.97</b>
	KEGG	0.68	0.77	0.80	0.94	0.98	0.96	0.98	<b>1.00</b>	0.98	<b>1.00</b>	<b>1.00</b>	<b>1.00</b>
Absolute set coverage $\delta^{ac}$	DBpedia	0.00	0.00	0.00	0.00	0.01	0.01	0.01	0.01	0.02	0.02	0.05	<b>0.07</b>
	GeoNames	0.07	0.07	0.08	0.13	0.14	0.15	0.17	0.18	0.19	0.28	0.30	<b>0.31</b>
	Jamendo	0.18	0.20	0.21	0.27	0.32	0.34	0.32	0.37	0.39	0.46	0.52	<b>0.53</b>
	LinkedMDB	0.00	0.00	0.00	0.01	0.01	0.01	0.01	0.01	0.02	0.03	0.04	<b>0.05</b>
	NY Times	0.04	0.04	0.04	0.10	0.12	0.13	0.15	0.17	0.20	0.28	0.32	<b>0.35</b>
	SW Dog Food	0.00	0.00	0.00	0.01	0.01	0.01	0.01	0.02	0.02	0.06	0.06	<b>0.07</b>
	ChEBI	0.00	0.00	0.00	0.01	0.02	0.02	0.02	0.03	0.03	0.07	0.09	<b>0.10</b>
	Drugbank	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.01	<b>0.02</b>	<b>0.02</b>
	KEGG	0.04	0.04	0.04	0.11	0.13	0.12	0.17	0.19	0.18	0.34	<b>0.35</b>	<b>0.35</b>
Relative set coverage $\delta^{rc}$	DBpedia	0.53	0.58	0.61	0.62	0.68	0.70	0.66	0.72	0.74	0.75	0.80	<b>0.83</b>
	GeoNames	0.96	0.96	0.97	0.99	0.99	0.99	0.99	0.99	0.99	<b>1.00</b>	<b>1.00</b>	<b>1.00</b>
	Jamendo	0.82	0.89	0.91	0.94	0.97	0.98	0.97	0.99	0.99	0.99	<b>1.00</b>	<b>1.00</b>
	LinkedMDB	0.74	0.73	0.73	0.84	0.84	0.84	0.85	0.86	0.86	0.89	<b>0.90</b>	<b>0.90</b>
	NY Times	0.27	0.29	0.25	0.52	0.62	0.66	0.64	0.76	0.80	0.87	0.95	<b>0.96</b>
	SW Dog Food	0.01	0.01	0.01	0.05	0.05	0.05	0.11	0.10	0.10	0.34	0.37	<b>0.38</b>
	ChEBI	0.06	0.10	0.15	0.18	0.38	0.48	0.27	0.55	0.62	0.51	0.82	<b>0.88</b>
	Drugbank	0.07	0.05	0.01	0.14	0.15	0.15	0.17	0.19	0.19	0.26	0.28	<b>0.31</b>
	KEGG	0.20	0.27	0.36	0.50	0.72	0.77	0.59	0.84	0.84	0.89	0.95	<b>0.96</b>

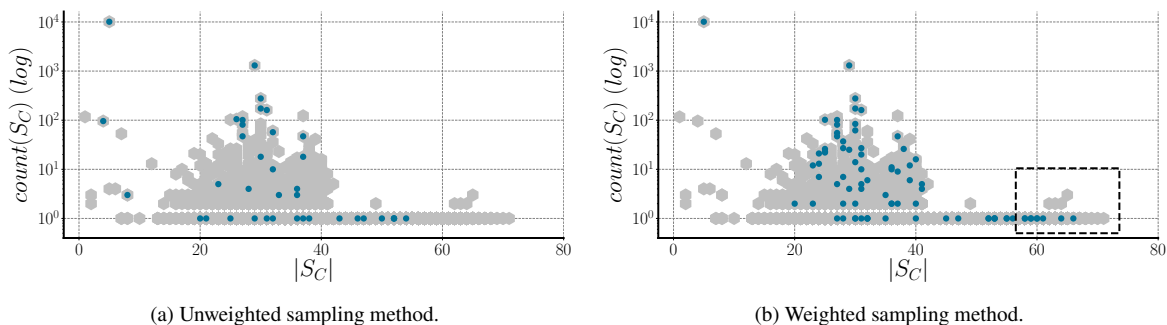


Fig. 3. Sampled characteristic sets for Drugbank for one sample with a  $n^0 = 0.5\%$ . The figures show the sampled characteristic sets in blue with respect to the number of their predicates ( $|S_C|$ ) and their count ( $count(S_C)$ ) on a log-scale. Indicated in gray are all sets of the original dataset.

$$cv_{deg.} = \frac{deg_{std}}{deg.}$$

The two graphs KEGG and ChEBI have a high dispersion of the out-degree distribution with  $cv_{deg.} > 19$  for ChEBI and  $cv_{deg.} > 5$  for KEGG. As a result, the weighted sampling method, which favors high out-degree nodes, is biased more strongly due to this graph property. Regarding the predicate coverage similarity  $\delta^{Pc}$ , we also observe that an increasing sample size improves the similarities in all cases. For five graphs, all predicates are covered by at least one sampling method with the largest sample size. The remaining four graphs (Drugbank, DBpedia, LinkedMDB, and SW Dog Food) have the highest number of distinct predicates (Q4) which reduces the probability of sampling them all.

Considering the absolute ( $\delta^{ac}$ ) and relative set coverage ( $\delta^{rc}$ ), similar results can be observed for the majority of graphs: Even if only a few characteristic sets are sampled (low  $\delta^{ac}$ ), the number of triples covered by those sets in the original graph covered is very high (high  $\delta^{rc}$ ). Take, for example, the unweighted sampling with 5.0‰ sample size for DBpedia. On average, the samples contain only 7% ( $\delta^{ac} = 0.07$ ) of all characteristics sets which cover 83% ( $\delta^{rc} = 0.83$ ) of the triples in the original graph. The cumulative coverage curve of the original graph (Q4) helps to explain these results. For instance, let's compare the curve for SW Dog Food (Fig. 2f) and LinkedMDB (Fig. 2d). The SW Dog Food curve indicates a more uneven distribution of characteristic sets and, as a result, even though more characteristics sets are sampled for SW Dog Food ( $\delta^{ac} = 0.07$ ) than for LinkedMDB ( $\delta^{ac} = 0.05$ ), they cover a smaller portion of the original graph: SW Dog Food  $\delta^{rc} = 0.38$  vs. LinkedMDB  $\delta^{rc} = 0.90$ .

Comparing the different sampling methods (Q2), we observe that the unweighted sampling method outperforms the other methods for the out-degree similarity  $\delta^{od}$ . This is due to the fact, that subjects are sampled with equal probability irrespective of their out-degree which leads to more representative samples. We illustrate this observation in Figure 3 by comparing the characteristic sets that are obtained by the weighted and unweighted sampling methods (in color) in comparison to the characteristic sets of the original graph (in gray). The characteristic sets sampled by the weighted method (Fig. 3b) tend to have a larger set size as indicated by the rectangle, while the unweighted sampling (Fig. 3a) captures more average-sized characteristic sets.

Except for LinkedMDB, the unweighted sampling method exhibits the lowest predicate coverage similarity ( $\delta^{Pc}$ ) in comparison to the other two approaches. The highest difference between the unweighted and weighted sampling method in  $\delta^{Pc}$  is measured for DBpedia, which is the graph with the largest number of distinct predicates. For the remaining graphs, the difference is smaller than 2 percentage points. Combining this observation with Figure 3, we conclude that the unweighted sampling method fails to obtain those predicates used in characteristic sets with high degrees<sup>10</sup>. Comparing the sampling methods (Q2), similar results for absolute  $\delta^{ac}$  and relative set coverage  $\delta^{rc}$  are observed as the unweighted method is outperformed by the other methods. Similar to the predicate coverage, the  $\delta^{rc}$  values are only slightly lower. When taking the number of triples in the samples into consideration ( $|H| \cdot |G|$ ), we find that the weighted sampling method

<sup>10</sup>The degree of a characteristic set is given by the average degree of the entities belonging to it. The degree is therefore a combination of its size (number of predicates) and the predicates' multiplicity.

yields the largest subgraphs. Therefore, if we consider the structural similarity results with respect to the number of triples, we find that even though the unweighted sampling method does not yield the best overall results, it still captures the properties of the original graph at least as accurately requiring a smaller subgraph. We observe that the sample sizes also reflect the structural properties of the original graph (Q4). For instance, the out-degree and its dispersion ( $cv_{deg}$ ) are reflected in the differences in the sample size between the weighted and unweighted method. For example, the samples of the weighted sampling method for the ChEBI graph with  $deg = 94.55$  are more than 100 times large than those of the unweighted method. Finally, the results of the structural similarity measures indicate that the hybrid sampling method obtains samples that balance the properties of the weighted and unweighted samples. Nonetheless, it outperforms neither of the other two methods.

### 5.1.2. Statistical Similarity Measures

We now focus on the statistical similarity measures that assess the accuracy of the *count* and *multiplicity* estimations. In this evaluation, we concentrate on the q-errors due to their more natural interpretation, rather than the proposed similarity measures  $\delta^{count}$  and  $\delta^{multiplicity}$  which are the inverse of the q-error. We start by discussing the results for the counts: q-error<sub>c</sub>. The results for all RDF graphs, sampling methods, sample sizes, and projection functions are visualized in Figure 4. To investigate the results on the level of samples, we aggregated the q-errors for each sample. Therefore, Figure 4 shows the mean q-errors of the 30 samples per group which is computed as

$$\frac{1}{j} \sum_{\hat{j}} \bar{q}\text{-error}_c \hat{S}_C^0$$

Similar to the structural similarity measures, the results indicate an improvement in the estimations with larger sample sizes (Q1) in the majority of cases. Regarding the projection functions, this observation holds for the basic projection function  $\phi_1$  for all datasets, except Drugbank. For the projection function  $\phi_2$ , there are more graphs for which the increased sample size does not yield better count estimations. Finally, especially for the projection function  $\phi_3$ , we observe that increasing sample size does not necessarily reduce the q-errors. Regardless of the projection function, when investigating larger sample sizes, we also need to consider

the percentage of characteristic sets that are sampled as indicated by the absolute set coverage  $\delta^{ac}$  in Table 1. For example, if we consider the SW Dog Food graph, we observe less accurate estimation for the projection function  $\phi_1$  and  $\phi_2$  with increasing sample size. However, when comparing the number of sampled characteristic sets, we find that the largest samples capture 36 times more characteristic sets than the smallest samples (0.18% vs. 6.5%).

Comparing the different sampling methods (Q2), we observe a tendency of the smaller q-errors for the unweighted sampling method, especially for the basic projection function  $\phi_1$ . At the same time, the results also show that the unweighted sampling method is substantially outperformed for DBpedia and Drugbank. This could be due to the large portion of exclusive characteristic sets that are sampled more frequently by the unweighted method. For graphs with fewer exclusive characteristic sets, such as KEGG and Jamendo, the unweighted sampling method yields better results.

Considering the characteristic set diversity of the graphs  $|S_C| \cdot \#Subjects$  (Q4), we find that the projection function  $\phi_3$  yields better results that are comparable or better than the other projection functions for graphs with a larger diversity (e.g., Drugbank, SW Dog Food). For graphs with small diversity, such as GeoNames or Jamendo,  $\phi_3$  yields the highest q-errors w.r.t. the other projection functions. The results of the other statistic-enhanced projection function  $\phi_2$ , are not as strongly affected by the characteristic set diversity and yields the best results on average. Consequently, in case the additional high-level statistics for the statistic-enhanced projection functions are available, the function  $\phi_2$  should be chosen over  $\phi_3$  (Q3). While the basic projection function  $\phi_1$  is slightly outperformed by  $\phi_2$ , it still yields comparable results in the majority of cases without requiring additional statistics.

Summarizing the results across all datasets for the count q-errors (cf. Fig. 4j), we observe that an increase in sample size (Q1) reduces the q-errors while simultaneously generating estimations for more characteristic sets. The unweighted sampling method (Q2) yields the lowest q-errors in the majority of cases and  $\phi_2$  (Q3) shows the best results regarding the average q-errors (lower median) and their dispersion (fewer outliers).

Next, we focus on the estimations of the predicate multiplicities within the characteristic sets. Analogous to the count statistic, we concentrate on the q-errors in the evaluation and aggregate the values per sample by

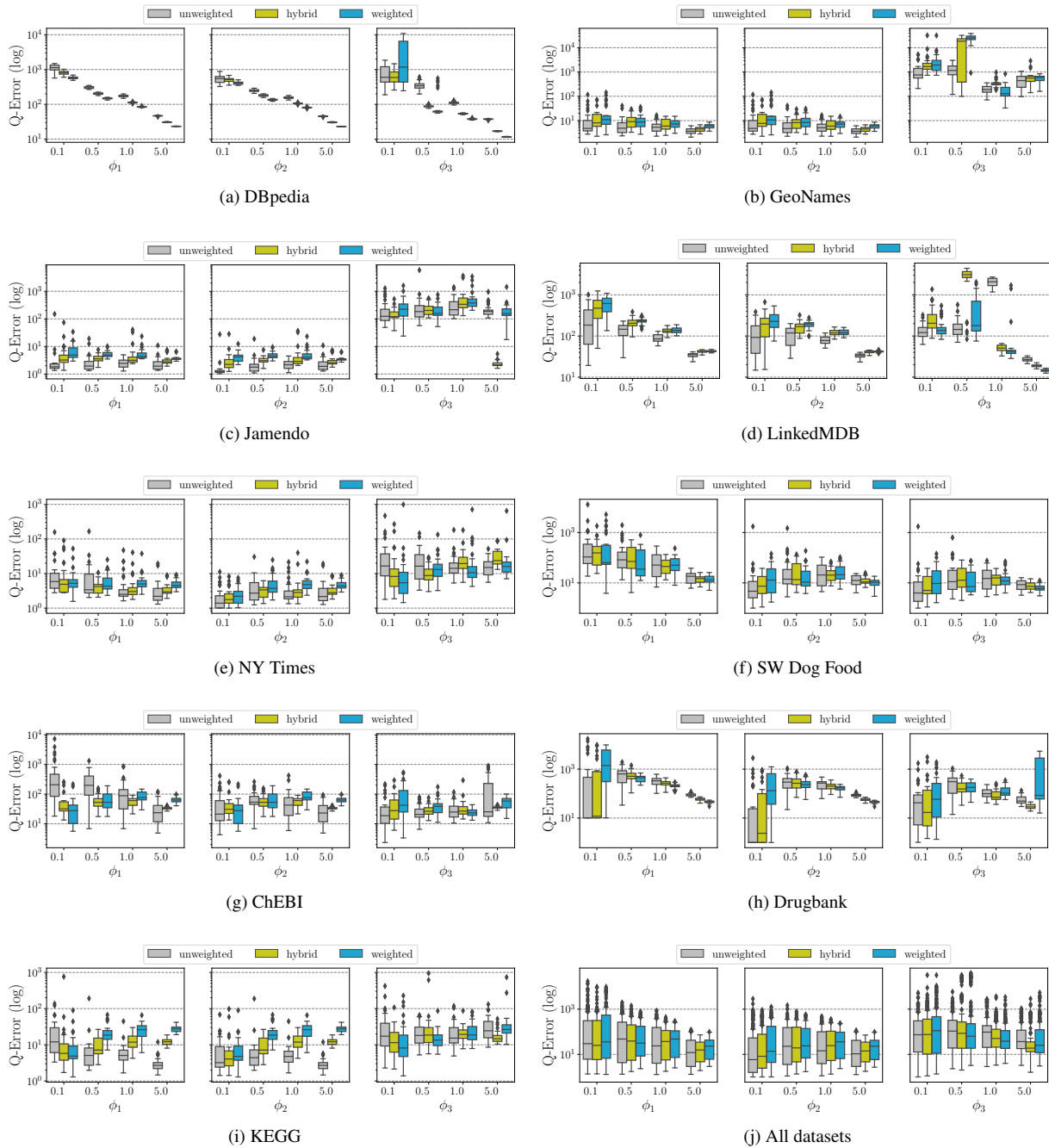


Fig. 4. Box plots of the count estimation q-errors (lower is better) for each dataset, projection function, sample size and sampling method.



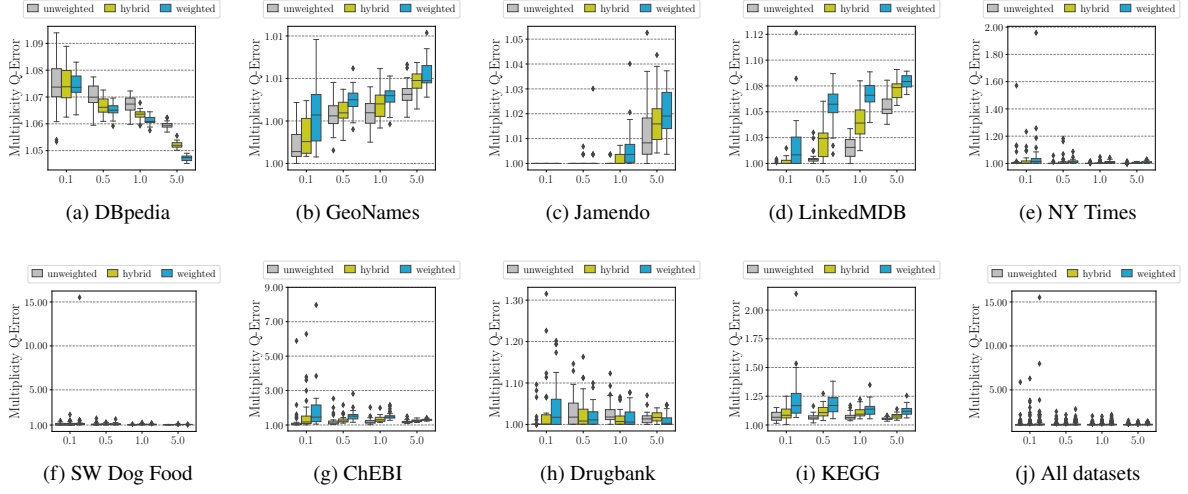


Fig. 5. Box plots of the q-errors (lower is better) of the predicate multiplicity estimations for each dataset, sample size and sampling method.

computing the mean over all its characteristic sets:

$$\frac{1}{j} \sum_j \hat{\sigma}_{C^2}^{\circ} \text{q-error}_m^{\circ} \hat{S}_C^{\circ}$$

The box plots of the samples' mean q-errors are shown in Figure 5 for all graphs, sample sizes, and sampling methods. The first obvious observation is the substantially lower q-errors in comparison to the count statistic. This is because the range of the value to be estimated is lower. Typically, the same predicate is only used a few times for the same subject, with some exceptions that occur multiple times, such as `rdf:type` or `rdfs:label`. This is also reflected in the absolute differences, where the multiplicity q-errors are less affected by the sampling method and sample size. This indicates a uniform predicate usage within the characteristic sets with few outliers. In line with the results for the count statistic, in the majority of cases, an increasing sample size (Q1) improves the results. The results for the different sampling methods show better results for the unweighted sampling method for the majority (seven) of the graphs. The weighted sampling method shows slightly better results for DBpedia and Drugbank. Comparable to the results for the count statistics, the hybrid sampling method finds a balance between the other two sampling methods. Furthermore, the results indicate that multiplicities are more challenging to estimate for graphs with a high out-degree dispersion (Q4), such as ChEBI and KEGG. The best overall estimations are observed for GeoNames, which also has the lowest out-degree dispersion ( $cv_{deg.} = 0.19$ ).

Concluding, as the summarized results in Fig. 5j show, the multiplicity statistics are estimated more accurately than count statistics with improving estimations for larger sample sizes (Q1), marginal differences comparing the sampling methods (Q2), and less diverse results for the different graphs (Q4).

### 5.1.3. Final Discussion

After providing a detailed presentation of our experimental results, we conclude the evaluation by providing answers to our core questions.

**Q1:** The results indicate that larger sample sizes have two major positive effects on the estimations. First, they improve both structural and statistical similarities measures of the estimations. Secondly, they allow for capturing and estimating the statistics for more characteristic sets and, therefore, potentially provide better support to applications that rely on these estimations.

**Q2:** Regarding the different sampling methods, we observe differences in the estimated profile features. On the one hand, the weighted sampling method tends to obtain larger samples for the same number of sampled subjects and, hence, yields slightly better structural similarity values. On the other hand, the unweighted sampling method obtains smaller samples and still obtains competitive structural similarity measures. Irrespective of the projection function, the unweighted sampling method outperforms the weighted sampling method w.r.t. the statistical similarity measures.

**Q3:** The analysis of the results showed that the first statistic-enhance projections function provides the best

estimations. However, it only slightly outperforms the basic projection function with the differences diminishing with increasing sample size. The second statistics-enhanced projection function is outperformed by the others. Especially for graphs with a small diversity in characteristic sets, it yields higher estimation errors.

**Q4:** The investigation of the results showed that the structure of the RDF graph affects the similarity values. Especially the counts of the characteristic sets are misestimated for datasets with a large portion of exclusive characteristic sets and a larger diversity of characteristic sets. In such scenarios, larger sample sizes can help to improve the estimations.

Concluding all findings, a variety of factors, ranging from the sampling method to the structure of the RDF graphs, impact the quality of the characteristic sets profile feature estimation. Therefore, in our second evaluation, we investigate how the CSPF estimation can be leveraged in a specific application and how the estimations impact the performance of the application.

## 5.2. Federated Query Planner Evaluation

After investigating the effectiveness of our approach to estimate profile features according to the similarity measures, we now focus on a particular application of the estimated features. We, therefore, study the performance of the proposed federated query planning approach that relies on CSPF estimations. In particular, we want to study the following core questions.

- Q5** How do the CSPF estimations obtained from the unweighted and weighted sampling methods impact the effectiveness of federated query planning?
- Q6** How do the CSPF estimations from different sample sizes impact the effectiveness of the plans?
- Q7** What is the effect of the different projection functions on the query plan effectiveness?
- Q8** How well do the structural similarity measures reflect the performance of the query plans?

We begin by introducing our experimental setup. We then provide a detailed evaluation and discussion of our experimental results in Section 5.2.1, which we conclude by answering our core questions in Section 5.2.2.

**Benchmark and Sampled Statistics** We study the effectiveness of the proposed query planning approach using the FedBench benchmark [31]. The benchmark consist of the 9 interlinked RDF graphs that are listed in Table 2, and which were also investigated in the

previous evaluation. We selected 25 queries from the Cross Domain (CD1-CD7), Life Science (LS1-LS7), and Linked Data (LD1-LD11) queries. We used a subset of the estimated CSPF generated from the samples of the previous evaluation. In particular, we only considered the *weighted* and *unweighted* sampling methods, leading to a total of 2160 estimated CSPF.<sup>11</sup>

As a baseline, we additionally computed the complete CSPF for all RDF graphs. In accordance with previous approaches [14, 16], we reduce the number of characteristics sets in the profile feature to a maximum of 10 000 characteristic sets per RDF graph.

**Implementation** We implemented the proposed federated query planning approach based on CROP [34] and nLDE [35]. We extended the query engine with an access operator for SPARQL endpoints and implemented our source selection, query decomposition, and left-linear query plan optimizer to create query execution plans. The RDF graphs in the federation are deployed with a single SPARQL endpoint per graph using Virtuoso v7.2.<sup>12</sup> We deployed all endpoints on a single machine (Debian Jessie 64 bit; CPU: 2x Intel(R) Xeon(R) CPU E5-2670 2.60GHz (16 physical cores); 256GB RAM) and execute the query engine on the same machine as well to avoid network latency.

We study the performance on each of the 30 samples for the 2 sampling methods (weighted and unweighted), 4 sample sizes ( $n^0 = f0.1\% \ jEj, 0.5\% \ jEj, 1\% \ jEj, 5\% \ jEj$ ), and 3 projection function ( $\phi_1, \phi_2, \phi_3$ ). This results in a total of  $30 \cdot 2 \cdot 4 \cdot 3 = 720$  configurations for the estimated CSPF. An additional configuration is given by the complete statistics. Moreover, we repeated the execution of each query (25) in each configuration 3 times, resulting in a total of  $3 \cdot 25 \cdot 721 = 54\,075$  query execution measurements. We set a timeout of 120 seconds per query for the engine.

**Evaluation Metrics** We study the following metrics:

1. **Execution Time:** Elapsed time spent by the engine to complete the evaluation of a query in seconds. This includes time spent on query planning.

<sup>11</sup>The previous experimental results suggest that the hybrid sampling method balances the two opposing (weighted and unweighted) sampling methods. We therefore assume that the estimated CSPF based on the hybrid samples will exhibit a similar performance in the following experiments. As a result, we only focus on the edge cases, that is weighted and unweighted samples.

<sup>12</sup><https://virtuoso.openlinksw.com/>

Table 4

Overview for the FedBench Queries. Mean and median execution times, mean number of requests, mean percentage of answers produces, and mean percentage of queries reaching the timeout. Best values per table are indicated in bold.

	unweighted												baseline
	0.1 % <sub>cc</sub>			0.5 % <sub>cc</sub>			1.0 % <sub>cc</sub>			5.0 % <sub>cc</sub>			1000 % <sub>cc</sub>
	$\phi_1$	$\phi_2$	$\phi_3$	$\phi_1$	$\phi_2$	$\phi_3$	$\phi_1$	$\phi_2$	$\phi_3$	$\phi_1$	$\phi_2$	$\phi_3$	-
Mean execution time [s]	7.09	6.69	6.77	4.75	4.69	4.65	3.33	3.34	3.33	<b>1.53</b>	1.54	1.54	8.43
Median execution time [s]	0.66	0.63	0.64	0.42	0.44	0.44	<b>0.42</b>	0.43	0.46	0.62	0.61	0.63	1.01
Mean number of requests	241.95	229.97	229.76	159.46	159.06	155.02	113.94	113.95	111.2	<b>45.69</b>	<b>45.69</b>	45.7	173.92
Answers [%]	72.13	72.09	72.13	90.13	90.13	90.09	96.53	96.53	96.53	<b>100.0</b>	<b>100.0</b>	<b>100.0</b>	<b>100.0</b>
Timeouts [%]	3.6	3.6	3.6	1.47	1.47	1.47	0.93	0.93	0.93	<b>0.0</b>	<b>0.0</b>	<b>0.0</b>	4.0

	weighted												baseline
	0.1 % <sub>cc</sub>			0.5 % <sub>cc</sub>			1.0 % <sub>cc</sub>			5.0 % <sub>cc</sub>			1000 % <sub>cc</sub>
	$\phi_1$	$\phi_2$	$\phi_3$	$\phi_1$	$\phi_2$	$\phi_3$	$\phi_1$	$\phi_2$	$\phi_3$	$\phi_1$	$\phi_2$	$\phi_3$	-
Mean execution time [s]	9.24	7.61	8.46	2.84	<b>1.89</b>	1.98	4.45	4.13	4.49	6.08	6.11	6.45	8.43
Median execution time [s]	0.56	0.55	0.56	<b>0.43</b>	0.45	0.56	0.57	0.58	0.68	0.76	0.77	0.77	1.01
Mean number of requests	170.94	171.75	174.21	<b>64.08</b>	64.14	64.16	178.27	178.18	192.06	256.97	257.02	257.32	173.92
Answers [%]	81.33	81.73	81.69	97.2	98.0	98.0	96.8	97.07	96.8	96.0	96.0	96.0	<b>100.0</b>
Timeouts [%]	5.73	4.36	5.07	1.07	<b>0.27</b>	<b>0.27</b>	2.8	2.53	2.8	4.13	4.13	4.13	4.0

2. *Number of requests*: Number of requests performed by the engine to the SPARQL endpoints during the query execution.
3. *Answer completeness*: Percentage of total answers produced. We computed the complete answers by executing the queries over the union of all graphs.
4. *Number of subexpressions*: Number of subexpressions in the query decomposition to assess how well the predicate co-occurrence of the graphs is captured in the estimated CSPF.

### 5.2.1. Experimental Results

An overview of the experimental results is provided in Table 4. The table shows the mean and median execution times, the mean number of requests, the mean percentage of answers produced, and the mean percentage of query executions reaching the timeout. We start with the results for the unweighted sampling method (Q5). Regarding the mean execution time, we observe the best performance for the unweighted sampling method with the largest sample size 5.0%<sub>cc</sub> and the projection function  $\phi_1$ . This configuration overall outperforms the other sampling methods and also the baseline with complete statistics. The execution times for the other projection functions  $\phi_2$  and  $\phi_3$  are only slightly higher in this case (Q7). Also, these configurations yield the lowest number of requests while producing 100% of all answers with none of the executions reaching the timeout. Moreover, a continuous improvement in all

evaluation metrics with increasing sample size (Q6) is visible for the unweighted sampling method. Both the mean execution times and the mean number of requests are about 5 times lower for the largest sample size. At the same time, the answer completeness increases by almost 30 percentage points, and the number of queries that reach a timeout are reduced from 3.6% to 0%. These results indicate that larger sample sizes in combination with the unweighted sampling method allows for obtaining more efficient query plans since the planner can identify relevant sources more accurately, larger subexpression, and a more efficient join order.

In contrast, the performance results differ for the weighted sampling method (Q5). The weighted sampling method with the second smallest sample size (0.5%<sub>cc</sub>) yields the best performance. When comparing the results for the projection functions (Q7), we find similar results as for the unweighted sampling method, with only slight differences for the different projection functions. The results for the sample size (0.5%<sub>cc</sub>) show the lowest average execution times and number of requests. Moreover, the query execution plans obtained for this sample size yields the highest answer completeness with 98% and the fewest queries that reach the timeout (0.27%<sub>cc</sub>). With an increasing sample size > 0.5%<sub>cc</sub>, the mean execution times increase. However, when combining the observations of the mean with the median execution times, we find that the increase in the median execution times is substantially smaller. This

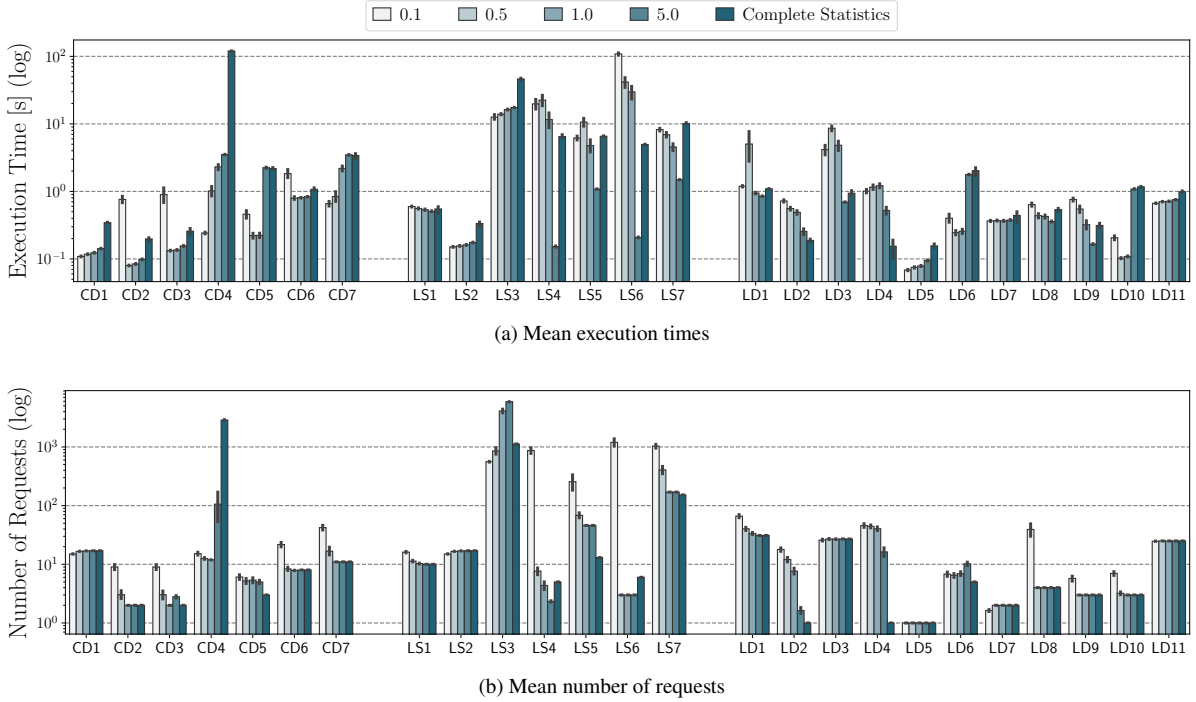


Fig. 6. Mean performances for all FedBench queries for the weighted sampling method and averaged across all projection functions.

indicates, that for a large number of queries the execution time only slightly increases, while for a few queries the execution time is significantly higher.

Hence, we focus on the results for the weighted sampling method in more detail to investigate the following observations: (i) a larger sample size in the weighted sampling does not entail better query execution performance, and (ii) the query planner with complete statistics is outperformed by the majority of configurations with estimated statistics. The mean execution times and mean number of requests (both log scale) per query and sample size for the weighted sampling method (all projection functions) are shown in Figure 6. The trend of an increase in execution time with larger sample sizes is observable especially for the queries CD4 and LS3. For the majority of the remaining queries, we see a similar or even better performance with larger sample sizes according to execution times and number of requests. The observation that few outliers (especially CD4 and LS3) skew the overall average performance results is also true for the performance of the planner with the complete statistics<sup>13</sup>. Taking a closer look at the results for LS3, we find that for the sample size

<sup>13</sup>Note that the query planner is designed to handle estimated statistics. Therefore, a query planner that fully leverages the com-

0.5%, the engine times out 3 times. Intriguingly, for the largest sample size (5.0%) and also for the complete statistics, it reaches the timeout for all 90 executions. Similarly, for query CD4, larger sample sizes also lead to several executions reaching the timeout. The reason for these results lies in the cardinality estimation approach. We observe that CD4 and LS3 have a selective triple pattern in which the object is bound. The STAR-JOINCARDINALITY algorithm tends to overestimate the cardinalities for stars with such triple patterns and the tendency of overestimating is higher with more characteristics sets that are taken into consideration<sup>14</sup>. As a result, this overestimation in the query CD4 and LS3 leads to a sub-optimal join ordering. For example, Figure 7 shows the join ordering for CD4 obtained with one weighted sample with size 0.5% and 5.0%. The estimated cardinalities (in blue) and true cardinalities (in parenthesis) in this example reveal that due to the overestimation of the subexpression with the bound ob-

plete statistics, e.g., with the (federated) characteristic pairs, would potentially obtain a more efficient query plan.

<sup>14</sup>This is due to the fact that the number of relevant characteristic sets for a triple pattern  ${}^1x p o$  increases while the conditional selectivity  $sel^1 o = x j? p = p^o$  only slightly decreases.

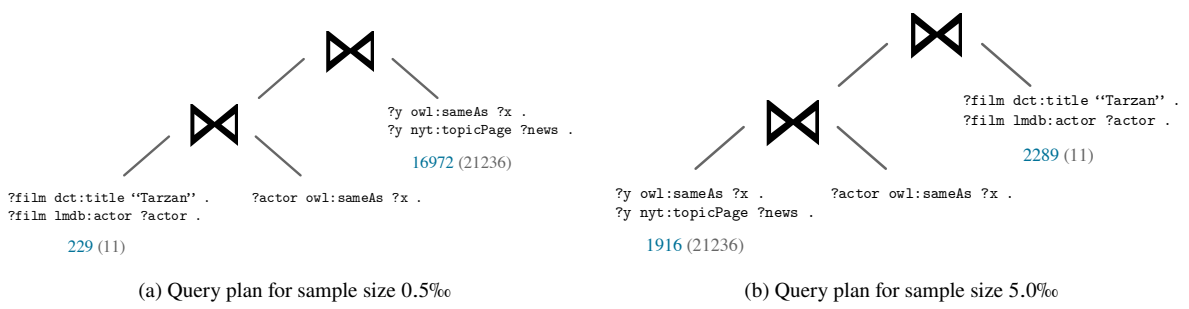


Fig. 7. CD4: Join Ordering obtained by the query planner based on the estimated CSPF from one of the weighted samples. Indicate below is the estimated cardinality (blue) and the actual cardinality (in brackets).

ject (“Tarzan”), the planner chooses a sub-optimal join order in the case of the larger sample.

Finally, we want to examine whether the proposed structural similarity measures (c.f. Section 3.6) reflect the performance of the query plans obtained by our planning approach (Q8). To this end, we compare the impact of the out-degree similarity ( $\delta^{od}$ ), predicate coverage ( $\delta^{Pc}$ ), absolute set coverage ( $\delta^{ac}$ ), and relative set coverage ( $\delta^{rc}$ ) on the query decompositions of our query planner. Specifically, we focus on their impact on the number of subexpressions in the decomposition and the percentage of answers produced by the query plans. For each of the 720 configurations, we compute the mean similarity measure values for all RDF graphs in the federation and the mean answer completeness and number of subqueries for all queries in the benchmark. Figure 8 shows the relation between these measures. As a quantitative measure to assess whether the values are correlated, we compute the Pearson correlation coefficient ( $PCC$ ). We start with the number of subexpressions in the query decompositions as a way to assess how well the predicate co-occurrence is captured in the profile estimations. Fewer subexpressions indicate that the query planner can merge more triple patterns into larger subexpressions to be evaluated jointly according to the characteristic sets in the CSPF (Line 10 in Algorithm 1). This allows for reducing the number of requests as well as the number of intermediate results that need to be transferred. Inspecting the number of subexpressions (Fig. 8a), there is no correlation between the out-degree similarity and the number of subexpression with  $PCC = 0.01$ . For the other measures, however, there is a negative correlation with  $PCC = 0.854$  for  $\delta^{Pc}$ ,  $PCC = 0.746$  for  $\delta^{ac}$ , and  $PCC = 0.814$  for  $\delta^{rc}$ . The correlation indicates higher similarity values yield decomposition with fewer subexpression. Hence, these measures reflect how well the predicate co-occurrence is captured

in the profile feature estimations, allowing the query planner to obtain efficient query plans.

Similar to this observation, the results in Fig. 8b show that, there is no correlation between the answer completeness and the out-degree similarity  $\delta^{od}$  with  $PCC = 0.039$ . For the other similarity measures, we observe a positive correlation of the similarity measure and the answer completeness, which is also reflected in the correlation coefficients with  $PCC = 0.856$  for  $\delta^{Pc}$ ,  $PCC = 0.643$  for  $\delta^{ac}$ , and  $PCC = 0.743$  for  $\delta^{rc}$ . The results indicate that estimated characteristic sets profile features with higher predicate and (absolute and relative) characteristic sets coverage help the query planner to obtain query plans that produce more answers.

### 5.2.2. Final Discussion

Concluding the evaluation of the experimental results for our query planning approach, we summarize the findings by providing answers to our core question.

**Q5:** The results show that the CSPF estimations obtained from the unweighted sampling method allow for determining the most efficient query plans. The difference in performance between the sampling methods is not due to the representativeness of the samples but rather due to sub-optimal join orderings caused by misestimated cardinalities for the weighted samples. This sub-optimal join led to time outs, which negatively affect the aggregated execution time and answer completeness of the weighted sampling. The lowest execution times and number of requests are observed for the unweighted sampling method with the largest sample size. For this configuration, all answers are produced and none of the query executions reach the timeout.

**Q6:** Regarding the results for different sample sizes, CSPF estimations obtained from larger samples using the unweighted sampling method improve the query execution performance. In contrast, the second small-



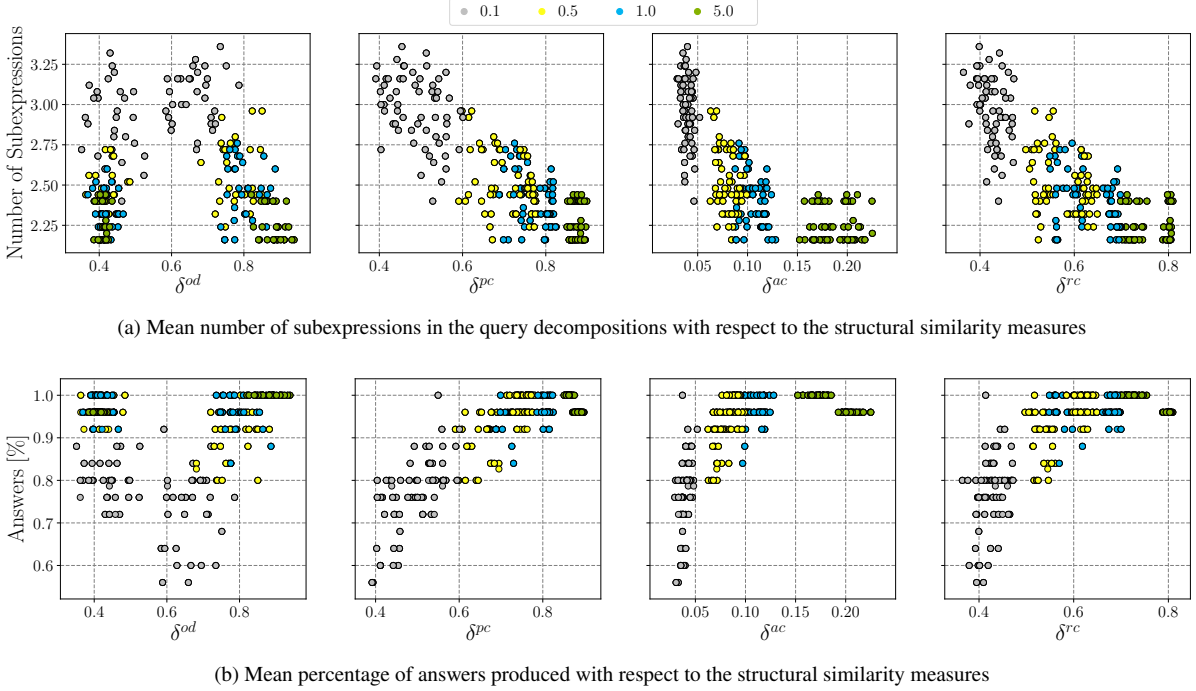


Fig. 8. Relation between the answer completeness and the number of subexpression of the query plans and the structural similarity measures for the estimated CSPF. Sample size are indicated in color.

est sample size yields the best results for the weighted sampling methods. In the latter case, the aggregated results are skewed due to few queries for which the approach overestimates the cardinality of subexpressions with a bound object. Adapting the cardinality estimation approach to better cope with sampled statistics could alleviate these shortcomings.

**Q7:** The count estimations of the projection functions are used to estimate the cardinalities of the subexpressions in the decompositions. The results show that there is only a small difference in query execution performance for the different projection functions (*ceteris paribus*). We observed that, even if the projection functions achieve different accuracy, this barely affects the join ordering. This is because the relative differences between count estimations of the functions are consistent with the relative differences among the true counts.

**Q8:** Considering the structural similarity measures, the results suggest that the similarity measures predicate coverage and (absolute and relative) set coverage correlate with the number of subexpressions in the decomposition and answer completeness of the query plans. The correlation indicates that more efficient query plans can be obtained for CSPF estimation that representative

according to these measures. This also underpins the effectiveness of the measures.

Summarizing the evaluation, we demonstrated the successful application of characteristic sets profile feature estimations to federated query planning. Overall, the unweighted sampling method with the largest sample size showed the best average performance. The experiments show that the proposed query planner provides a foundation to further develop query planning approaches that leverage estimated profile features obtained from RDF graph samples. For instance, future research could investigate source selection techniques as proposed in [12], but relying on samples to get the authority of subjects and objects in the graphs.

**Limitations** Finally, we want to discuss the limitations of the proposed query planner and experiments. The first limitation is the heuristic nature of the planner. As a result, there are no theoretical guarantees for the answer completeness of the query decomposition. For example, in case the planner finds at least one relevant source for a triple pattern in the estimated statistics, it assumes there are no additional relevant sources, which are not captured in the statistics. In other words, the query planner does not know what it does not know about. This limitation could potentially be overcome by



1 obtaining additional information from the RDF graph,  
2 such as its distinct predicates, in the case that the expres-  
3 sivity of the query interface supports this. Secondly, our  
4 evaluation focuses on an implementation that combines  
5 query planning and execution based on CROP. There-  
6 fore, our insights are currently limited to this setup  
7 and future work should investigate our planner with  
8 other query execution engines (e.g. using SPARQL 1.1  
9 queries). This would also allow for a direct comparison  
10 to Odyssey. Finally, additional federated benchmarks  
11 could be investigated to gain further insights.

12 Regardless, our experimental results still provide  
13 valuable insights into the potential of using sam-  
14 ple-based statistics and summaries to support federated  
15 querying approaches without requiring access to the  
16 entire datasets of the federation members.

## 19 6. Related Work

20 We now discuss related work on statistical profiling  
21 (Section 6.1), graph sampling (Section 6.2) and feder-  
22 ated query processing (Section 6.3).

### 25 6.1. Statistical Profiling

26 In the realm of statistical feature profiling for RDF  
27 graphs, a variety features and tools to assess them have  
28 been proposed. Zloch et al. [36] investigate statistical  
29 features of RDF graphs with the focus on the topo-  
30 logical graph structure including degree-based, edge-  
31 based, centrality, and descriptive statistical measures.  
32 Consequently, these measures are not specific to RDF  
33 graphs and, in contrast to our work, do not capture the  
34 semantics on the instance or schema level of the data.  
35 Complementary, Fernández et al. [23] focus on RDF-  
36 specific measures. They propose various schema- and  
37 instance-level metrics to characterize RDF datasets that  
38 incorporate the particularities of RDF graphs. The met-  
39 rics and the resulting statistics are tailored to the devel-  
40 opment of better RDF data storage solutions including  
41 data structures, indexes, and compression techniques  
42 by considering RDF data characteristics.

43 LODStats [37] is a statement-stream-based approach  
44 for gathering comprehensive statistics of RDF datasets.  
45 They present 32 instance- and schema-level statistical  
46 criteria covering both RDF-specific metrics as well as  
47 topological graph metrics. The authors aim to improve  
48 reuse, linking, revising, or querying Linked Open Data  
49 sources but do not discuss specific applications.

1 ProLOD++ [38] is a tool to support various profiling,  
2 mining, and cleansing functionalities for RDF datasets.  
3 It is an extension of ProLOD [39] that computes pro-  
4 files at domain, schema, and data (i.e., instance) level  
5 of datasets. ProLOD++ supports browser-based visual-  
6 izations of these characteristics with the goal to better  
7 support data consumers in understanding, consuming,  
8 and integrating Linked Open Data sources.

9 Finally, ExpLOD [40] is a tool for generating sum-  
10 maries of RDF datasets combining text labels and  
11 bisimulation contractions. These summaries include  
12 schema-level statistical information such as the class,  
13 predicate, and interlinking usage in the dataset. Similar  
14 to ProLOD++ [38], the goal of the created summaries  
15 is to facilitate the understanding the RDF usage, such  
16 as vocabulary usage and links between datasets.

17 In this work, we propose a novel schema-level sta-  
18 tistical profile feature based on characteristic sets that  
19 captures both the topological structure of the graph as  
20 well as its semantics. This statistical feature has not  
21 yet been addressed in related work. We demonstrate an  
22 application of estimations of this feature in federated  
23 query planning.

### 25 6.2. Graph Sampling

26 Next, we focus on sampling approaches for graphs.  
27 We start by introducing related work on sampling large  
28 graphs and networks. Thereafter, we analyze RDF-  
29 specific sampling approaches and their applications.

30 *Graph Sampling* A variety of approaches for sam-  
31 pling large non-RDF graphs have been proposed.  
32 Leskovec et al. [27] provide an overview of approaches  
33 suitable for obtaining representative samples from large  
34 graphs for scale-down sampling for static graphs and  
35 back-in-time sampling for evolving graphs. They con-  
36 sider the distributions of different structural properties  
37 of the graphs, denoted static graph patterns, as the cri-  
38 teria for evaluating the representativeness of the samples  
39 for scale-down sampling. These static graph patterns  
40 include, among others, the in-degree, out-degree, and  
41 cluster coefficient distributions. As a similarity mea-  
42 sure to assess the representativeness of samples, they  
43 use the Kolmogorov-Smirnov D-statistic of the graph  
44 patterns' distribution. They present three major cate-  
45 gories of sampling approaches: (i) random node selec-  
46 tion, (ii) random edge selection, and (iii) graph explo-  
47 ration. The results of their evaluation reveal that there  
48 is no single best solution and the authors conclude that  
49 an appropriate sampling algorithm and sample size de-  
50 pends on the specific application.

1 Ahmed et al. [26] present a detailed framework for  
 2 the problem of graph sampling that focuses on large  
 3 scale graphs. They identify two models of computation  
 4 that are relevant when sampling from large graphs. The  
 5 *static model of computation* allows for randomly  
 6 accessing any location in the graph. The *streaming model*  
 7 *of computation* merely allows for accessing edges in  
 8 a sequential stream of edges. In their evaluation, the  
 9 authors show that the proposed methods preserve key  
 10 graph statistics of the graph (e.g., degree distributions,  
 11 cluster coefficient distribution). Moreover, they demon-  
 12 strate low space- and runtime-complexity that is in the  
 13 order of edges in the sample for these methods.

14 Our sampling methods are a special case scale-down  
 15 sampling of random node selection approaches accord-  
 16 ing to [27], that only sample specific nodes, the entities  
 17 of an RDF graph. However, in contrast to traditional  
 18 graph sampling, we aim to generate representative sam-  
 19 ples for our specific statistical graph feature. Hence, the  
 20 representativeness of a sample is given by the accuracy  
 21 of the profile feature estimation.

22 *RDF Graph Sampling* Sampling approaches for RDF  
 23 graphs have been proposed and applied to different  
 24 problems as well. In the following, we will analyze ex-  
 25 ample applications and show how sampling approaches  
 26 are chosen according to those applications.

27 Debattista et al. [41] propose approximating spe-  
 28 cific quality metrics for large, evolving datasets based  
 29 on samples. They argue that the exact computation of  
 30 some quality metrics is too time-consuming and that an  
 31 approximation is usually sufficient. In particular, they  
 32 apply a sampling-based approach for the quality met-  
 33 rics (i) dereferenceability of URIs, and (ii) links to ex-  
 34 ternal data providers. They use the reservoir sampling  
 35 approach [42] which randomly selects  $n$  items from a  
 36 set of  $N$  elements with the equal probability  $1/N$ . In  
 37 their evaluation, the authors studied how well the ratios  
 38 measured by quality metrics can be estimated based on  
 39 the samples. Therefore, the similarity of estimated and  
 40 true ratio reflects their notion of representativeness. In  
 41 contrast to our work, the authors apply sampling to re-  
 42 duce the computational effort for quality metrics, and  
 43 therefore, the sampling methods do not need to capture  
 44 the semantics or structural features of the dataset.

45 In their work, Rietveld et al. [43] aim to obtain sam-  
 46 ples that entail as many of the original answers to  
 47 *typical* SPARQL queries. They use query logs from  
 48 SPARQL endpoints to determine such typical queries.  
 49 The sampling pipeline proposed by the authors con-  
 50 sists of four steps and aims to obtain the parts of the

1 graph that are relevant for answering the queries. First,  
 2 they rewrite the original RDF graph as a directed un-  
 3 labeled graph. On the resulting graph, they then com-  
 4 pute the structural graph metrics PageRank, in-degree,  
 5 and out-degree for all nodes. Thereafter, they use the  
 6 structural graph metrics to rank the triples in the graph  
 7 and generate the sample by selecting the *top-k* percent  
 8 of all triples. The authors aim to obtain samples with  
 9 a “more manageable size” [43] that produce answers  
 10 to common SPARQL queries and hence, the samples’  
 11 representativeness is measured by the recall for those  
 12 queries. As a result, the resulting samples may be bi-  
 13 ased towards more prominent entities in the graphs and  
 14 less suitable to capture long-tail entities [44]. Different  
 15 from our work, the goal is obtaining *relevant* samples  
 16 which allow answering common queries. Therefore,  
 17 these samples are not *representative* in our sense with  
 18 regards to the semantic and statistical features.

19 Soulet et al. [20] focus on analytical queries, which  
 20 are typically too expensive to be executed directly over  
 21 public SPARQL endpoints. They propose separating  
 22 the computation of such expensive queries by executing  
 23 them over random samples of the RDF graph. Due to  
 24 the properties of the queries, the values for the aggre-  
 25 gations converge as they are executed over increasingly  
 26 larger portions of the graph. As a result, the authors do  
 27 not rely on the necessity of each sample to be represen-  
 28 tative according to the aggregates, but rely on the con-  
 29 vergence of the aggregates towards the true value with  
 30 an increasing number of samples. Similar to Soulet et  
 31 al. [20], our work is motivated by the restrictions that  
 32 occur especially in decentralized scenarios with large,  
 33 evolving datasets where it is not feasible to have local  
 34 access to every dataset. Different from [20], we aim to  
 35 sample the data in such a fashion that a single sample  
 36 can be used to estimate the statistical profile feature and  
 37 do not rely on the convergence properties induced by  
 38 increasing sample sizes.

### 40 6.3. Profiling-based Federated Query Processing

41 We demonstrate how the profile features estimations  
 42 can be applied to federated query processing. There-  
 43 fore, in this section, we analyze other types of pre-  
 44 computed statistics leveraged by federated query en-  
 45 gines to determine efficient plans.

46 ANAPSID [9] is an adaptive federated query engine  
 47 that leverages a *Catalog* with high-level statistics on  
 48 the members in the federation. This catalog contains a  
 49 list of the predicates present in each data member in  
 50  
 51

1 the federation and is used by the engine to perform the  
2 source selection and query decomposition.

3 More fine-grained statistics are leveraged by the fol-  
4 lowing approaches. SPLENDID [10] uses indexes built  
5 from VOID descriptions for query planning. These in-  
6 dexes additionally hold information on the occurrences  
7 of predicates within the classes of the sources. In com-  
8 bination with ASK queries, the indexes are used to de-  
9 compose the query into subqueries to be evaluated  
10 at the sources in the federation. HiBISCuS [12] is a  
11 source selection approach that employs data summaries  
12 to prune the relevant sources. The data summaries are  
13 comprised of the capabilities of a data source which  
14 consists of the subject and object authorities for all  
15 predicates in a data source. These summaries allow  
16 for deciding which source will contribute to the final  
17 query answers according to the authorities of the sub-  
18 jects and objects they contain. CostFed [13] is a cost  
19 model-based planner using an index with statistic pro-  
20 files on the sources, so-called *Data Summaries*. They  
21 extend the data summaries from HiBISCuS with addi-  
22 tional information on both the distribution of objects  
23 and subjects per predicates. In addition to source selec-  
24 tion and query decomposition, the additional statistics  
25 are used in their cost model to estimate the cost of al-  
26 ternative query plans. Similar to CostFed, SemaGrow  
27 [11] leverages a metadata to estimate the cost of alter-  
28 native query plans. The statistics include the number of  
29 distinct subjects, predicates, and objects as well as the  
30 total number of triples that match a given triple pattern.  
31 These statistics are used to estimate the join cardinali-  
32 ties in their cost model. Finally, Odyssey [16] relies on  
33 the characteristic sets of the federation members. Addi-  
34 tionally, the characteristics pairs and federated charac-  
35 teristic pairs are used to capture links between entities  
36 within and across the federation members.

37 Our approach also uses statistics to perform source  
38 selection, query decomposition, and join ordering to  
39 obtain efficient query plans. Specifically, our approach  
40 leverages characteristic sets similar to Odyssey [16].  
41 In contrast to the aforementioned approaches, however,  
42 our approach does not require the complete statistics  
43 but is specifically designed to handle estimated statis-  
44 tics from RDF graph samples. As a result, our approach  
45 aims to leverage the estimated statistics as much as pos-  
46 sible and is also able to cope with potentially misesti-  
47 mated or missing statistics.  
48  
49  
50  
51

## 7. Conclusion

1 In this work, we presented an approach to estimate  
2 RDF dataset profile features and an application of  
3 those estimations in federated SPARQL query plan-  
4 ning. Specifically, we focused on the characteristic  
5 set profile features as it captures not only structural  
6 but also semantic properties of RDF graphs. We pro-  
7 posed a sampling-based approach that utilizes a projec-  
8 tion function to estimate characteristic set profile fea-  
9 ture statistics (*RQ 1*). Furthermore, we introduced four  
10 structural and two statistical similarity measures that  
11 allow for assessing the representativeness of the gener-  
12 ated feature estimations. In our experimental study, we  
13 investigated the effectiveness of the proposed estima-  
14 tion approach using these similarity measures (*RQ 2*).  
15 The evaluation showed that our approach obtains rep-  
16 resentative feature estimations according to the struc-  
17 tural similarity measure. In contrast, the lower statis-  
18 tical similarity values showed that the count distribu-  
19 tions of characteristic sets in the RDF graphs are more  
20 challenging to estimate. These results highlight that the  
21 samples capture structural aspects while it is difficult  
22 to capture the distribution of the characteristic sets.  
23

24 The second main contribution of this work focused  
25 on an application of the proposed feature estimations  
26 to better understand the practical implications of their  
27 representativeness (*RQ 3*). We proposed a federated  
28 query planning approach that leverages the estimated  
29 characteristic set profile features to perform source se-  
30 lection, query decomposition, and join ordering. The  
31 query planner is based on insights from existing charac-  
32 teristic set-based query planning strategies. However,  
33 it is adapted such that it can handle limited and poten-  
34 tially inaccurate information in the profile feature es-  
35 timations. The results of our experiments on the Fed-  
36 Bench benchmark illustrated the feasibility of using  
37 estimated profile features. Overall, we found the best  
38 query plans are obtained for the feature estimations  
39 using the unweighted sampling method with improve-  
40 ments for larger sample sizes. Moreover, the results re-  
41 vealed that the capability of obtaining more efficient  
42 query plans with respect to the number of subexpres-  
43 sions and answer completeness is also reflected in the  
44 structural similarity measures (*RQ 2*).  
45

46 Future work will focus on two directions. The first  
47 direction is extending our profile feature estimation ap-  
48 proach with further sampling approaches that make use  
49 of additional information, such as query logs, to better  
50 capture the relevant parts of the RDF graphs. Besides,  
51 our approach can be extended to estimate other profile

features. The second area of research is the application of the feature estimations to other problems. It would be worth investigating how other applications can benefit from the estimated profile features. Furthermore, we aim to improve the query planning approach by (i) leveraging further information that can be obtained from the samples, and (ii) implementing a hybrid planning strategy that not only relies on the estimated statistics. Finally, an evaluation of our query planner in additional federations will support a better understanding of the advantages and limitations of our approach.

## Acknowledgements

This work was supported by the grant QUOCA (FKZ 01IS17042) from the German Federal Ministry of Education and Research (BMBF).

## Appendix A. Efficiency of CSPF Computation

Part of the motivation of this work is the fact that computing and estimating CSPF over samples of the original graph reduces the computational effort. As discussed in Section 3.5, the computation method needs to be tailored to the access methods of the RDF graph. To showcase the reduction in computational effort when computing CSPF from samples, we focus on a single, general case and we assume the RDF graphs to be given as edge lists (i.e., n-triples files).

*Algorithm* The algorithm first sorts the triples in the graph  $G$  by subject. Thereafter, it iterates the sorted list of triples, determining the characteristic set per subject and storing the corresponding statistics in a hash table or dictionary. This process is detailed in Algorithm 2.

Central to the computation are dictionaries  $Dict^{10}$  (or hash tables) that hold the statistics. The  $keys^{10}$  function of a dictionary returns the set of keys. The  $gets^1k, v^0$  function returns the value for key  $k$ , if  $k$  is a key in the dictionary. Otherwise, it initializes the value for  $k$  with  $v$  and returns  $v$ . The algorithm uses the  $count$  dictionary for the  $count$  statistics and the  $mult$  dictionary for the multiplicities. The algorithm iterates all triples in the graph (Line 7 to Line 18) and keeps track of the current characteristic set in the dictionary  $CS$ , where the keys are the predicates and the values their counts. When a new subject is encountered ( $s^0 < s_i$ ), the algorithm updates the statistics in  $count$  and  $mult$  accordingly. Note that for brevity,

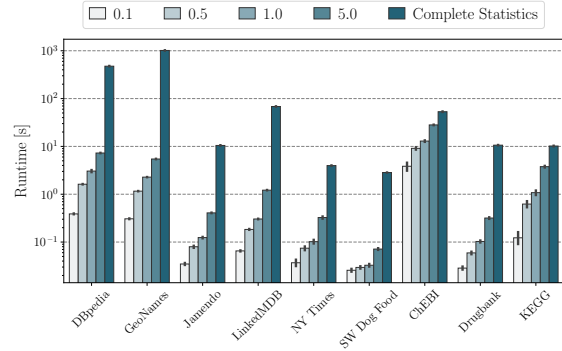


Fig. 9. Runtimes (log-scale) for computing the CSPF for the different sample sizes and the complete RDF graphs.

we omit a final iteration to  $mult$  that divides the absolute multiplicity by the  $count$  to obtain the mean multiplicity as defined in Equation (2).

*Efficiency Results* In Fig. 9, the mean runtimes for computing the CSPF are summarized per sample size and for the entire original graph. For the sake of comparability, the triples were randomly shuffled before computing the CSPF. For each of the 90 samples per dataset and sample size, the CSPF was computed once and the runtimes averaged across the 90 measurements. For the original graphs, this process was repeated 10 times to obtain an average runtime value. The results indicate a substantial reduction in the runtimes for computing the CSPF for the samples than the original graphs (note the log-scale for the runtimes). In addition, the gain in efficiency also depends on the structure of the RDF graph, such as its out-degree dispersion.

## References

- [1] N.F. Noy, Y. Gao, A. Jain, A. Narayanan, A. Patterson and J. Taylor, Industry-scale knowledge graphs: lessons and challenges, *Commun. ACM* **62**(8) (2019), 36–43. doi:10.1145/3331166.
- [2] M.N. Mami, D. Graux, S. Scerri, H. Jabeen, S. Auer and J. Lehmann, Uniform Access to Multiform Data Lakes using Semantic Technologies, in: *Proceedings of the 21st International Conference on Information Integration and Web-based Applications & Services, iiWAS 2019, Munich, Germany, December 2-4, 2019*, ACM, 2019, pp. 313–322. doi:10.1145/3366030.3366054.
- [3] J.Z. Pan, G. Vetere, J.M. Gómez-Pérez and H. Wu (eds), *Exploiting Linked Data and Knowledge Graphs in Large Organisations*, Springer, 2017. ISBN 978-3-319-45652-2. doi:10.1007/978-3-319-45654-6.

**Algorithm 2:** Characteristic Set Profile Feature Creation

---

```

1 Input: RDF Graph  $G = \{s_1, p_1, o_1, \dots, s_n, p_n, o_n\}$ 
2 Output: CSPF: Count statistics  $count$  and multiplicity
3           statistics  $mult$ 
4 // Sort  $G$  by subject in a list
5 1  $G_{sorted} \leftarrow \text{sort}(G)$ 
6 // Append null to determine end of triples
7 2  $G_{sorted} \leftarrow G_{sorted}.append(\text{null}, \text{null}, \text{null})$ 
8 // Dictionaries for count and multiplicity
9 3  $count \leftarrow \text{Dict}$ 
10 4  $mult \leftarrow \text{Dict}$ 
11 // Auxiliary dictionary
12 5  $CS \leftarrow \text{Dict}$ 
13 6  $s^0 \leftarrow \text{null}$ 
14 7 for  $(s_i, p_i, o_i) \in G_{sorted}$  do
15   8 if  $s^0 = \text{null}$  then
16     // Initialize  $s^0$ 
17     9  $s^0 \leftarrow s_i$ 
18   // If a new subject is encountered
19   10 if  $s^0 < s_i$  then
20     // Increase count for characteristic
21     // set of  $s^0$ 
22     11  $count \leftarrow CS.keys \cup \{s^0\}$ 
23      $count \leftarrow count.get(s^0, 0) + 1$ 
24     // Count occurrence of each predicate
25     12 for  $key, value \in CS$  do
26       13  $mult \leftarrow mult.get(key, Dict)$ 
27        $mult \leftarrow mult.get(key, 0) + value$ 
28     14 end
29     // Reset auxiliary dictionary and
30     // update  $s^0$ 
31     15  $CS \leftarrow \text{Dict}$ 
32     16  $s^0 \leftarrow s_i$ 
33   // Increase counter for predicate  $p_i$ 
34   17  $CS \leftarrow CS.get(p_i, 0) + 1$ 
35 18 end
36 19 return  $count, mult$ 

```

---

[4] S. Cebiric, F. Goasdoué, H. Kondylakis, D. Kotzinos, I. Manolescu, G. Troullinou and M. Zneika, Summarizing semantic graphs: a survey, *VLDB J.* **28**(3) (2019), 295–327. doi:10.1007/s00778-018-0528-3.

[5] A. Zaveri, A. Rula, A. Maurino, R. Pietrobon, J. Lehmann and S. Auer, Quality assessment for Linked Data: A Survey, *Semantic Web* **7**(1) (2016), 63–93. doi:10.3233/SW-150175.

[6] M.B. Ellefi, Z. Bellahsene, J.G. Breslin, E. Demidova, S. Dietze, J. Szymanski and K. Todorov, RDF dataset profiling - a survey of features, methods, vocabularies and applications, *Semantic Web* **9**(5) (2018), 677–705. doi:10.3233/SW-180294.

[7] T. Johnson, Data Profiling, in: *Encyclopedia of Database Systems, Second Edition*, L. Liu and M.T. Özsu, eds, Springer, 2018. doi:10.1007/978-1-4614-8265-9\_601.

[8] S. Dietze, E. Demidova and K. Todorov, RDF Dataset Profiling, in: *Encyclopedia of Big Data Technologies*, S. Sakr and A.Y. Zomaya, eds, Springer, 2019. doi:10.1007/978-3-319-63962-8\_288-1.

[9] M. Acosta, M. Vidal, T. Lampo, J. Castillo and E. Ruckhaus, ANAPSID: An Adaptive Query Processing Engine for SPARQL Endpoints, in: *The Semantic Web - ISWC 2011 - 10th International Semantic Web Conference, Bonn, Germany, October 23-27, 2011, Proceedings, Part I*, 2011, pp. 18–34. doi:10.1007/978-3-642-25073-6\_2.

[10] O. Görlitz and S. Staab, SPLENDID: SPARQL Endpoint Federation Exploiting VOID Descriptions, in: *Proceedings of the Second International Workshop on Consuming Linked Data (COLD2011), Bonn, Germany, October 23, 2011*, O. Hartig, A. Harth and J.F. Sequeda, eds, CEUR Workshop Proceedings, Vol. 782, CEUR-WS.org, 2011. http://ceur-ws.org/Vol-782/GoerlitzAndStaab\_COLD2011.pdf.

[11] A. Charalambidis, A. Troumpoukis and S. Konstantopoulos, SemaGrow: optimizing federated SPARQL queries, in: *Proceedings of the 11th International Conference on Semantic Systems, SEMANTICS 2015, Vienna, Austria, September 15-17, 2015*, A. Polleres, T. Pellegrini, S. Hellmann and J.X. Parreira, eds, ACM, 2015, pp. 121–128. doi:10.1145/2814864.2814886.

[12] M. Saleem and A.N. Ngomo, HiBISCuS: Hypergraph-Based Source Selection for SPARQL Endpoint Federation, in: *Proceedings of ESWC*, 2014, pp. 176–191. doi:10.1007/978-3-319-07443-6\_13.

[13] M. Saleem, A. Potocki, T. Soru, O. Hartig and A.N. Ngomo, CostFed: Cost-Based Query Optimization for SPARQL Endpoint Federation, in: *Proceedings of the 14th International Conference on Semantic Systems, SEMANTICS 2018, Vienna, Austria, September 10-13, 2018*, 2018, pp. 163–174. doi:10.1016/j.procs.2018.09.016.

[14] T. Neumann and G. Moerkotte, Characteristic sets: Accurate cardinality estimation for RDF queries with multiple joins, in: *Proceedings of ICDE*, 2011. doi:10.1109/ICDE.2011.5767868.

[15] M. Pham, L. Passing, O. Erling and P.A. Boncz, Deriving an Emergent Relational Schema from RDF Data, in: *Proceedings of the 24th International Conference on World Wide Web, WWW 2015, Florence, Italy, May 18-22, 2015*, A. Gangemi, S. Leonardi and A. Panconesi, eds, ACM, 2015, pp. 864–874. doi:10.1145/2736277.2741121.

[16] G. Montoya, H. Skaf-Molli and K. Hose, The Odyssey Approach for Optimizing Federated SPARQL Queries, in: *Proceedings of ISWC*, 2017. doi:10.1007/978-3-319-68288-4\_28.

[17] G. Moerkotte, T. Neumann and G. Steidl, Preventing Bad Plans by Bounding the Impact of Cardinality Estimation Errors, *PVLDB* **2**(1) (2009), 982–993. doi:10.14778/1687627.1687738. http://www.vldb.org/pvldb/2/vldb09-657.pdf.

[18] L. Heling and M. Acosta, Estimating Characteristic Sets for RDF Dataset Profiles Based on Sampling, in: *The Semantic Web - 17th International Conference, ESWC 2020, Heraklion, Crete, Greece, May 31-June 4, 2020, Proceedings*, A. Harth, S. Kirrane, A.N. Ngomo, H. Paulheim, A. Rula, A.L. Gentile, P. Haase and M. Cochez, eds, Lecture Notes in Computer Science, Vol. 12123, Springer, 2020, pp. 157–175. doi:10.1007/978-3-030-49461-2\_10.

[19] R. Verborgh, M.V. Sande, O. Hartig, J.V. Herwegen, L.D. Vocht, B.D. Meester, G. Haesendonck and P. Colpaert, Triple Pattern Fragments: A low-cost knowledge graph interface for the Web, *J. Web Semant.* **37-38** (2016), 184–206. doi:10.1016/j.websem.2016.03.003.

- [20] A. Soulet and F.M. Suchanek, Anytime Large-Scale Analytics of Linked Open Data, in: *Proceedings of ISWC*, 2019. doi:10.1007/978-3-030-30793-6\_33.
- [21] M.V. Mannino, P. Chu and T. Sager, Statistical Profile Estimation in Database Systems, *ACM Comput. Surv.* **20**(3) (1988), 191–221. doi:10.1145/62061.62063.
- [22] T. Neumann and G. Weikum, The RDF-3X engine for scalable management of RDF data, *VLDB J.* **19**(1) (2010), 91–113. doi:10.1007/s00778-009-0165-y.
- [23] J.D. Fernández, M.A. Martínez-Prieto, P. de la Fuente Redondo and C. Gutiérrez, Characterising RDF data sets, *J. Inf. Sci.* **44**(2) (2018), 203–229. doi:10.1177/0165551516677945.
- [24] A. Gubichev and T. Neumann, Exploiting the query structure for efficient join ordering in SPARQL queries, in: *Proceedings of EDBT*, 2014. doi:10.5441/002/edbt.2014.40.
- [25] M. Meimaris, G. Papastefanatos, N. Mamoulis and I. Anagnostopoulos, Extended Characteristic Sets: Graph Indexing for SPARQL Query Optimization, in: *Proceedings of ICDE*, 2017. doi:10.1109/ICDE.2017.106.
- [26] N.K. Ahmed, J. Neville and R.R. Kompella, Network Sampling: From Static to Streaming Graphs, *TKDD* **8**(2) (2013), 7:1–7:56. doi:10.1145/2601438.
- [27] J. Leskovec and C. Faloutsos, Sampling from large graphs, in: *Proceedings of ACM SIGKDD*, 2006, pp. 631–636. doi:10.1145/1150402.1150479.
- [28] B.F. Ribeiro, P. Wang, F. Murai and D. Towsley, Sampling directed graphs with random walks, in: *Proceedings of the IEEE INFOCOM 2012*, 2012, pp. 1692–1700. doi:10.1109/INFOCOM.2012.6195540.
- [29] M. Newman, *Networks*, Oxford university press, 2018.
- [30] M. Schmidt, M. Meier and G. Lausen, Foundations of SPARQL query optimization, in: *Database Theory - ICDT 2010, 13th International Conference, Lausanne, Switzerland, March 23-25, 2010, Proceedings*, L. Segoufin, ed., ACM International Conference Proceeding Series, ACM, 2010, pp. 4–33. doi:10.1145/1804669.1804675.
- [31] M. Schmidt, O. Görlitz, P. Haase, G. Ladwig, A. Schwarte and T. Tran, FedBench: A Benchmark Suite for Federated Semantic Data Query Processing, in: *The Semantic Web - ISWC 2011 - 10th International Semantic Web Conference, Bonn, Germany, October 23-27, 2011, Proceedings, Part I*, L. Aroyo, C. Welty, H. Alani, J. Taylor, A. Bernstein, L. Kagal, N.F. Noy and E. Blomqvist, eds, Lecture Notes in Computer Science, Vol. 7031, Springer, 2011, pp. 585–600. doi:10.1007/978-3-642-25073-6\_37.
- [32] M. Acosta, O. Hartig and J.F. Sequeda, Federated RDF Query Processing, in: *Encyclopedia of Big Data Technologies*, S. Sakr and A.Y. Zomaya, eds, Springer, 2019. doi:10.1007/978-3-319-63962-8\_228-1.
- [33] M. Vidal, E. Ruckhaus, T. Lampo, A. Martínez, J. Sierra and A. Polleres, Efficiently Joining Group Patterns in SPARQL Queries, in: *The Semantic Web: Research and Applications, 7th Extended Semantic Web Conference, ESWC 2010, Heraklion, Crete, Greece, May 30 - June 3, 2010, Proceedings, Part I*, L. Aroyo, G. Antoniou, E. Hyvönen, A. ten Teije, H. Stuckenschmidt, L. Cabral and T. Tudorache, eds, Lecture Notes in Computer Science, Vol. 6088, Springer, 2010, pp. 228–242. doi:10.1007/978-3-642-13486-9\_16.
- [34] L. Heling and M. Acosta, Cost- and Robustness-Based Query Optimization for Linked Data Fragments, in: *ISWC (1)*, Lecture Notes in Computer Science, Vol. 12506, Springer, 2020, pp. 238–257.
- [35] M. Acosta and M. Vidal, Networks of Linked Data Eddies: An Adaptive Web Query Processing Engine for RDF Data, in: *The Semantic Web - ISWC 2015 - 14th International Semantic Web Conference, Bethlehem, PA, USA, October 11-15, 2015, Proceedings, Part I*, M. Arenas, Ó. Corcho, E. Simperl, M. Strohmaier, M. d'Aquin, K. Srinivas, P. Groth, M. Dumontier, J. Heflin, K. Thirunarayan and S. Staab, eds, Lecture Notes in Computer Science, Vol. 9366, Springer, 2015, pp. 111–127. doi:10.1007/978-3-319-25007-6\_7.
- [36] M. Zloch, M. Acosta, D. Hienert, S. Dietze and S. Conrad, A Software Framework and Datasets for the Analysis of Graph Measures on RDF Graphs, in: *The Semantic Web - 16th International Conference, ESWC 2019, Portorož, Slovenia, June 2-6, 2019, Proceedings*, P. Hitzler, M. Fernández, K. Janowicz, A. Zaveri, A.J.G. Gray, V. López, A. Haller and K. Hammar, eds, Lecture Notes in Computer Science, Vol. 11503, Springer, 2019, pp. 523–539. doi:10.1007/978-3-030-21348-0\_34.
- [37] S. Auer, J. Demter, M. Martin and J. Lehmann, LODStats - An Extensible Framework for High-Performance Dataset Analytics, in: *Proceedings of EKAW*, 2012, pp. 353–362. doi:10.1007/978-3-642-33876-2\_31.
- [38] Z. Abedjan, T. Grütze, A. Jentzsch and F. Naumann, Profiling and mining RDF data with ProLOD++, in: *Proceedings of ICDE*, 2014. doi:10.1109/ICDE.2014.6816740.
- [39] C. Böhm, F. Naumann, Z. Abedjan, D. Fenz, T. Grütze, D. Hefenbrock, M. Pohl and D. Sonnabend, Profiling linked open data with ProLOD, in: *Workshops Proceedings of the 26th International Conference on Data Engineering, ICDE*, 2010, pp. 175–178. doi:10.1109/ICDEW.2010.5452762.
- [40] S. Khatchadourian and M.P. Consens, ExpLOD: Summary-Based Exploration of Interlinking and RDF Usage in the Linked Open Data Cloud, in: *Proceedings of ESWC*, 2010, pp. 272–287. doi:10.1007/978-3-642-13489-0\_19.
- [41] J. Debattista, S. Londoño, C. Lange and S. Auer, Quality Assessment of Linked Datasets Using Probabilistic Approximation, in: *Proceedings of ESWC*, 2015. doi:10.1007/978-3-319-18818-8\_14.
- [42] J.S. Vitter, Random Sampling with a Reservoir, *ACM Trans. Math. Softw.* **11**(1) (1985), 37–57. doi:10.1145/3147.3165.
- [43] L. Rietveld, R. Hoekstra, S. Schlobach and C. Guéret, Structural Properties as Proxy for Semantic Relevance in RDF Graph Sampling, in: *Proceedings of ISWC*, 2014. doi:10.1007/978-3-319-11915-1\_6.
- [44] E. Cao, D. Wang, J. Huang and W. Hu, Open Knowledge Enrichment for Long-tail Entities, in: *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020, ACM / IW3C2*, 2020, pp. 384–394. doi:10.1145/3366423.3380123.
- [45] J. Lehmann, G. Sejdin, L. Böhmann, P. Westphal, C. Stadler, I. Ermilov, S. Bin, N. Chakraborty, M. Saleem, A.N. Ngomo and H. Jabeen, Distributed Semantic Analytics Using the SANS Stack, in: *The Semantic Web - ISWC 2017 - 16th International Semantic Web Conference, Vienna, Austria, October 21-25, 2017, Proceedings, Part II*, C. d'Amato, M. Fernández, V.A.M. Tamma, F. Lécué, P. Cudré-Mauroux, J.F. Sequeda, C. Lange and J. Heflin, eds, Lecture Notes in Computer Science, Vol. 10588, Springer, 2017, pp. 147–155. doi:10.1007/978-3-319-68204-4\_15.



- 1 [46] J. Pérez, M. Arenas and C. Gutiérrez, Semantics and complexity  
2 of SPARQL, *ACM Trans. Database Syst.* **34**(3) (2009), 16:1–  
3 16:45. doi:10.1145/1567274.1567278.
- 4 [47] I. Manolescu and M. Mazuran, Speeding up RDF Aggregate  
5 Discovery through Sampling, in: *Proceedings of the Workshops*  
6 *of the EDBT/ICDT 2019 Joint Conference, EDBT/ICDT 2019,*  
7 *Lisbon, Portugal, March 26, 2019*, 2019. [http://ceur-ws.org/](http://ceur-ws.org/Vol-2322/BigVis_6.pdf)  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51
- Vol-2322/BigVis\_6.pdf.
- [48] K. Alexander, R. Cyganiak, M. Hausenblas and J. Zhao, De-  
scribing Linked Datasets, in: *Proceedings of the WWW2009*  
3 *Workshop on Linked Data on the Web, LDOW 2009, Madrid,*  
4 *Spain, April 20, 2009*, 2009. [http://ceur-ws.org/Vol-538/](http://ceur-ws.org/Vol-538/ldow2009_paper20.pdf)  
5 [ldow2009\\_paper20.pdf](http://ceur-ws.org/Vol-538/ldow2009_paper20.pdf).  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51