# Explaining Federated SPARQL Queries with How-Provenance

Zubaria Asma[1], Mrityunjoy Bhattacharya[2], Daniel Hernández[2], Luis Galárraga[3], Giorgos Flouris[1], Irini Fundulaki[1] and Katja Hose[4,5]

[1]FORTH-ICS, Heraklion, Crete, Greece
[2]University of Stuttgart, Stuttgart, Germany
[3]Inria, Rennes, France
[4]Aalborg University, Aalborg, Denmark
[5]TU Wien, Wien, Austria

**Abstract**

Existing federated query processing systems allow the consumption of heterogeneous and decentralized Web data. However, the problem of computing and interacting with how-provenance for SPARQL federations has received little attention. This paper proposes a method to compute how-provenance for federations, a user interface to interact with this federated provenance data, and a work plan to implement this proposal, including feedback and collaborations obtained in the workshop.

## 1. Introduction

Federated query processing responds to one of the central goals of the Semantic Web, for it makes the heterogeneous and decentralized nature of the Web transparent to information consumers. Federations hold great value for multiple downstream applications such as search engines, smart assistants, or data integration systems. However, some applications require information about the how-provenance of the answers obtained from the federation. The how-provenance of a data unit, e.g., a SPARQL query answer, is a trace of the data sources and operations required to obtain that unit as the result of a data process, e.g., a query [1]. In a federated setting, how-provenance has countless applications in quality control (e.g., accuracy assessment), privacy and access control, engine optimization and debugging, among other use cases.

**Motivation.** Despite the potential of how-provenance in federated RDF/SPARQL systems, existing approaches are not directly applicable to federations. This is so, because neither the formalisms nor the methods to compute how-provenance from query answers can handle data distributed across different endpoints. Based on this observation, we propose to (a) adapt the existing how-provenance formalisms for SPARQL queries to federated engines, and (b) to design novel approaches, based on query rewriting, to annotate query solutions with how-provenance explanations. In a first step these explanations could (c) be subsequently visualized for the sake of several applications including query profiling and optimization, source verification and data discovery, among others. That can empower the users of federated systems in multiple ways.

**Impact.** Federation engines typically focus on optimizing source selection and query execution, while they rarely focus on the interaction with the user. However, for users to develop trust in the answers computed by a federated engine, it is essential to provide provenance, i.e., explanations of how the answer was computed given the information provided by the different sources. By providing such provenance, we strive at significantly enhancing the user experience.

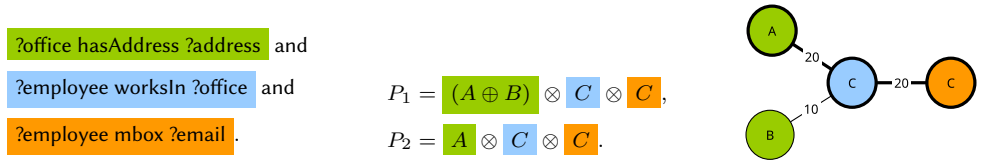
## 2. Proposed solution

### 2.1. Federated provenance computation

Our NPCS implementation [2] employs a middleware that takes a SPARQL query and generates a new query designed to retrieve provenance information. The resulting query seamlessly executes on a singular instance of a SPARQL endpoint, yielding the desired result along with its how-provenance algebraic expression. Expanding upon this foundation, our extended approach, *Fed-NPCS*, integrates Fedex for rewriting. Fedex plays a crucial role in identifying sources involved in answering each triple pattern, utilized to construct a query featuring FILTER clauses.

In the Fed-NPCS framework, federated SPARQL queries undergo a transformation, resulting in rewritten versions enriched with service clauses and NPCS-defined

*COST Action DKG Workshop, 26th - 28th June 2024, Malaga Spain*
asma@ics.forth.gr (Z. Asma);
mrityunjoy.bhattacharya@ms.informatik.uni-stuttgart.de (M. Bhattacharya); daniel.hernandez@ki.uni-stuttgart.de (D. Hernández); luis.galarraga@inria.fr (L. Galárraga); fgeo@ics.forth.gr (G. Flouris); fundul@ics.forth.gr (I. Fundulaki); khose@cs.aau.dk (K. Hose)
0000-0002-9402-7487 (Z. Asma); 0000-0001-6323-7767 (M. Bhattacharya); 0000-0002-7896-0875 (D. Hernández); 0000-0002-0241-5379 (L. Galárraga); 0000-0002-8937-4118 (G. Flouris); 0000-0002-4812-9896 (I. Fundulaki); 0000-0001-7025-8099 (K. Hose)
© 2024 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).




(a) Query (b) Provenance polynomials (c) Visualization

**Figure 1:** The answers to the query are annotated with either polynomial $P_1$ or $P_2$. If each polynomial annotates ten answers, then we know that 20 answers are jointly obtained from services $A$ and $C$. The colors indicate the triple patterns, the label '20' on the edge between the green node $A$ and the blue node $C$ indicates the number of offices described in both endpoints. The line thickness emphasizes the number of entities obtained from an endpoint.

rules, introducing federated effects. These service clauses are then dispatched to multiple instances of SPARQL endpoints, with each sub-query directed to its respective endpoint. The aggregated results are subsequently visualized in a graph.

### 2.2. Supporting the user during federated provenance-based data exploration

The backend we described so far annotates each answer with a polynomial describing how the federated endpoints generate an answer. We propose to combine the provenance of all answers to understand and explore the participation of the endpoints to answer a query. This combination is presented as a graph whose nodes represent endpoints and triple patterns in the query, and the edges indicate that the triple patterns share variables. To understand our proposal, consider a query asking for the office address and the email of each employee, and assume that the data is distributed in three federated endpoints, namely $A$, $B$, and $C$. Figure 1 shows how the combination of the provenance of all answers is used to visualize the contribution of the federated endpoints to the answers of the query. This visualization shows that removing endpoint $B$ does not affect the number of answers, whereas removing endpoint $A$ will reduce the answers to 10, and removing endpoint $C$ will reduce the answers to zero. This visualization can be enriched with multiple interactive functionalities to allow, for example, selecting a subset of the federation nodes, or to relax the query by removing a triple pattern, and explore how the results vary.

## 3. Workplan and Expected Outcomes

We will initially focus on the backend. As mentioned, although NPCS [2] is a workable implementation at the moment, it does not support federated queries. Our plan is to extend the implementation in this direction, aiming to have a working implementation for at least a useful fragment of the SPARQL language by the workshop date. Moreover, we will continue discussing and improving our mockups. We will also develop an initial frontend that will support whatever fragment of SPARQL the backend is supporting at the time.

During the workshop, the then-current version of the system will be presented to participants, along with mockups for the missing components. Our objective during the workshop, and, in fact, our main motivation for participating in the workshop, is to get feedback from potential users and experts with regards to the features offered by the system, especially the frontend. The workshop will serve as an excellent vehicle to elicit requirements and to identify new, potentially useful for users, features, visualisations, and interaction modes. This will help us guide our future work and focus on the features that matter.

Although, our system is not expected to be mature enough for a complete validation during the workshop, we could carry out an initial validation, and also gather volunteers for a future (complete) validation, which is to be performed when the system reaches the desired level of maturity. Last, but certainly not least, we could use the workshop as a means to establish collaborations for future research on related aspects.

## References

[1] D. Hernández, L. Galárraga, K. Hose, Computing how-provenance for SPARQL queries via query rewriting, Proceedings of the VLDB Endowment 14 (2021) 3389–3401. URL: https://doi.org/10.14778/3484224.3484235. doi:10.14778/3484224.3484235.

[2] Z. Asma, D. Hernández, L. Galárraga, G. Flouris, I. Fundulaki, K. Hose, NPCS: Native provenance computation for SPARQL, in: WWW-24 (under review), 2024.