# SemLAV: Querying Deep Web and Linked Open Data with SPARQL

Pauline Folz[1,2], Gabriela Montoya[1,3], Hala Skaf-Molli[1], and Pascal Molli[1]
Maria-Esther Vidal[4]

[1] LINA– Nantes University, France
{pauline.folz,gabriela.montoya,hala.skaf,pascal.molli}@univ-nantes.fr
[2] Nantes Métropole - Direction Recherche, Innovation et Enseignement Supérieur, France
[3] Unit UMR6241 of the Centre National de la Recherche Scientifique (CNRS), France
[4] Universidad Simón Bolívar, Venezuela
mvidal@ldc.usb.ve

**Abstract.** SemLAV allows to execute SPARQL queries against the Deep Web and Linked Open Data data sources. It implements the mediator-wrapper architecture based on view definitions over remote data sources. SPARQL queries are expressed using a mediator schema vocabulary, and SemLAV selects relevant data sources and *rank* them. The ranking strategy is designed to deliver results quickly based only on view definitions, *i.e.*, no statistics, nor probing on sources are required. In this demonstration, we validate the effectiveness of SemLAV approach with real data sources from social networks and Linked Open Data. We show in different setups that materializing only a subset of ranked relevant views is enough to deliver significant part of expected results.

## 1 Introduction

The Deep Web is constituted from data that are not indexed by traditional search engines, and may not have static URL links; it is around 500 times the size of the Surface Web [2]. Performing SPARQL queries without considering the Deep Web can deliver poor results. For example, the execution of the SPARQL query: *Which members of the Semantic Web community are interested in* Dalai Lama*,* Barack Obama*, or* Rihanna*?* (cf figure 1) without the Deep Web support, will deliver no answer. Some semantic data-warehouses such as Virtuoso with SPONGER [1] address this issue by declaring wrappers able to query unsemantified data including web service calls, CSV files, and so on. Such approach is relevant if the number of sources used to answer the query remains low. However, the time for first answers can be very high because the query engine has to contact all the declared wrappers for a query.

Contrary, SemLAV [3] is able to deliver answers quickly for this query. It follows the mediator-wrapper approach where the Deep Web data can be retrieved through view definitions and wrappers. A view constitutes a data source for the mediator. Given a SPARQL query, SemLAV selects relevant views and most importantly, it *ranks* them without requiring costly statistics. SemLAV

```
SELECT DISTINCT *
WHERE {
  ?P foaf:member ?C .
  ?C rdfs:label "Semantic Web" .
  ?P foaf:knows ?WKP .
  ?WKP foaf:name ?N .
   FILTER (?N="Dalai Lama" || ?N="Barack Obama" || ?N="Rihanna")
}
```

Fig. 1: Which members of the Semantic Web community are interested in *Dalai Lama*, *Barack Obama*, or *Rihanna*?

uses wrappers to semantify data of the selected data sources on-demand during query execution. It retrieves data from the ranked sources in a smart order that gives a high probability of delivering results. Consequently, even in the presence of a large number of relevant views, SemLAV is able to deliver results in a reasonable time. In this paper, we demonstrate how SemLAV is able to quickly deliver results for SPARQL queries mixing Deep Web data sources and Linked Open Data defined using around 250 views. A video of the demo is available at `https://www.youtube.com/channel/UCMQO5QVq5UcztE8kkkRRXKQ/videos`.

## 2 SemLAV Overview

Given a query and a set of views, SemLAV computes a ranked set of relevant views for answering the query. Ranking is computed using the number of equivalent covered rewritings detailed in [3]. Views are materialized by calling traditional wrappers such as those defined in SPONGER [1] in sequence or in parallel. Each time a new view is fully materialized, the original query is executed to deliver results as fast as possible. Views used in SemLAV could be also generated by tools like Karma [4]. To illustrate the benefits of SemLAV, consider the query defined in Figure 1, and the following five views:

```
v1(P,A,I,C,L):-made(P,A),affiliation(P,I),member(P,C),label(C,L)
v2(A,T,P,N,C):-title(A,T),made(P,A),name(P,N),member(P,C)
v3(P,N,R,M):-name(P,N),name(R,M),knows(P,R)
v4(P,N,G,R,C):-name(P,N),gender(P,G),knows(P,R),member(P,C)
v5(P,N,R,C,L):-name(P,N),knows(P,R),member(P,C),label(C,L)
```

SemLAV will compute the following sorted bucked for each query subgoal:

| member(P, C) | label(C, L) | knows(P, WKP) | name(WKP, N) |
|---|---|---|---|
| v5(P,N,R,C,L) | v5(P,N,R,C,L) | v5(P,N,R,C,L) | v5(P,N,R,C,L) |
| v4(P,N,G,R,C) | v1(P,A,I,C,L) | v4(P,N,G,R,C) | v4(P,N,G,R,C) |
| v1(P,A,I,C,L) | | v3(P,N,R,M) | v2(A,T,P,N,C) |
| v2(A,T,P,N,C) | | | v3(P,N,R,M) |

The execution of all possible combinations produces the complete answers for the query. To deliver answers quickly, SemLAV ranks the relevant views according to their contribution to cover the query subgoals, *i.e.,* first ranked views are those that cover maximum number of subgoals. Therefore, the number of covered combinations grows as fast as possible.

| # Included views ($k$) | SemLAV ranking | | Random order | |
|---|---|---|---|---|
| | Included views ($V_k$) | # Covered rewritings | Included views ($V_k$) | # Covered rewritings |
| 1 | v5 | $1 \times 1 \times 1 \times 1 = 1$ | v1 | $1 \times 1 \times 0 \times 0 = 0$ |
| 2 | v5, v4 | $2 \times 1 \times 2 \times 2 = 8$ | v1, v2 | $2 \times 1 \times 0 \times 1 = 0$ |
| 3 | v5, v4, v1 | $3 \times 2 \times 2 \times 2 = 24$ | v1, v2, v3 | $2 \times 1 \times 1 \times 2 = 4$ |
| 4 | v5, v4, v1, v3 | $3 \times 2 \times 3 \times 3 = 54$ | v1, v2, v3, v4 | $3 \times 1 \times 2 \times 3 = 18$ |
| 5 | v5, v4, v1, v3, v2 | $4 \times 2 \times 3 \times 4 = 96$ | v1, v2, v3, v4, v5 | $4 \times 2 \times 3 \times 4 = 96$ |

## 3 Demonstration Setup

In this demonstration, we use well known Deep Web sites such as social networks Twitter and Facebook, and Linked Open Data sources such as DBLP, Semantic Web Dog Food, and DBpedia. We define 253 views data sources (views) over Twitter, Facebook, DBLP, Semantic Web Dog Food, and DBpedia. We use several RDF vocabularies to describe the members of a community, and the links between them and the Linked Open Data cloud. The following are our assumptions: *i*) a person is member of a community if there is a link between the person and the community. This link is represented with the `foaf:member` predicate. Links of this type are established, for example, when someone follows a community conference Twitter account, or someone is member of a community group in Facebook, or someone has published a paper in a community conference. *ii*) A person knows another person if there exists a link between them. This link is represented with the `foaf:knows` predicate. Links of this type are established, for example, when: a user is following someone in Twitter, two persons are coauthors of a paper. Sources are described by SPARQL queries, e.g., query in

```
SELECT DISTINCT *
WHERE {
  ?follower <http://xmlns.com/foaf/0.1/name> ?name .
  ?follower <http://xmlns.com/foaf/0.1/knows> ?followed .
  ?follower <http://xmlns.com/foaf/0.1/member> ?community .
  ?community <http://www.w3.org/2000/01/rdf-schema#label> "Semantic Web"
}
```

Fig. 2: Description of the followers of ESWC Conferences Twitter account

Figure 2 describes data extracted from the Twitter account of *ESWC Conferences*.

### 3.1 Queries

We will demonstrate the behavior of SemLAV with the following four queries:

– Query 1: Semantic Web community members who know well-known personalities (e.g., Dalai Lama, Barack Obama, or Rihanna); for these persons show their affiliation, localization, and number of contributions that have done to the community.

– Query 2: Members of different scientific communities that know Tim Berners-Lee; for these persons show their affiliation and localization.
– Query 3: The Semantic Web members that have been more active in Twitter posting tweets about ESWC2014.
– Query 4: Members of the Database community that are known by Semantic Web members. For these persons, show their affiliation and localization.

Figure 3 presents the demonstration interface; all the reported results are computed and plotted dynamically. Queries can be selected at the top. The area enclosed in the blue rectangle (number 1) displays the query, and the `Run` and `Stop` buttons that starts and interrupts the query processing, respectively. Additionally, the number of relevant views and the total number of views are reported, as well as, view definitions and ranking are displayed using the buttons in the upper right side of this area. Area enclosed in the red rectangle (number 2) shows the interface state after 10 minutes running `Query 1`; the SemLAV ranking and a random sorting of the views are also shown. Line charts illustrate relationships between the number of produced answers and the number sources whose data have been retrieved. The number of produced answers has impressively grown before retrieving data from 6% of the sources (15 out of 250 selected sources), such that more than 50% of the answers has been delivered. Additionally, for the same amount of retrieved views, for example for 10 views, SemLAV was able to produce 819 answers, whereas the random sorting could not produce any answer so far. In the area enclosed in the green rectangle (number 3), query results are displayed in a map based on the retrieved locations. Attendees will observe the benefits of the SemLAV strategies in action and how SemLAV is able to deliver a considerable number of answers in a relatively low amount of time.

## 4 Conclusions

In this demonstration, we validate the effectiveness of the SemLAV approach, and illustrate how SPARQL queries can be efficiently executed against real-world sources from social networks and the Linked Open Data cloud. In different setups, sources are selected and ranked by SemLAV in a way that it benefits the incremental delivery of answers, while only a small number of views are retrieved.

## References


1. Virtuoso sponger. White paper, OpenLink Software.
2. B. He, M. Patel, Z. Zhang, and K. C.-C. Chang. Accessing the Deep Web. *Commun. ACM*, 50(5):94–101, 2007.
3. G. Montoya, L. D. Ibáñez, H. Skaf-Molli, P. Molli, and M.-E. Vidal. SemLAV: Local-As-View Mediation for SPARQL. *Transactions on Large-Scale Data- and Knowledge-Centered Systems XIII, Lecture Notes in Computer Science, Vol. 8420*, pages 33–58, 2014.
4. M. Taheriyan, C. A. Knoblock, P. A. Szekely, and J. L. Ambite. Rapidly Integrating Services into the Linked Data Cloud. In *International Semantic Web Conference (1)*, pages 559–574, 2012.

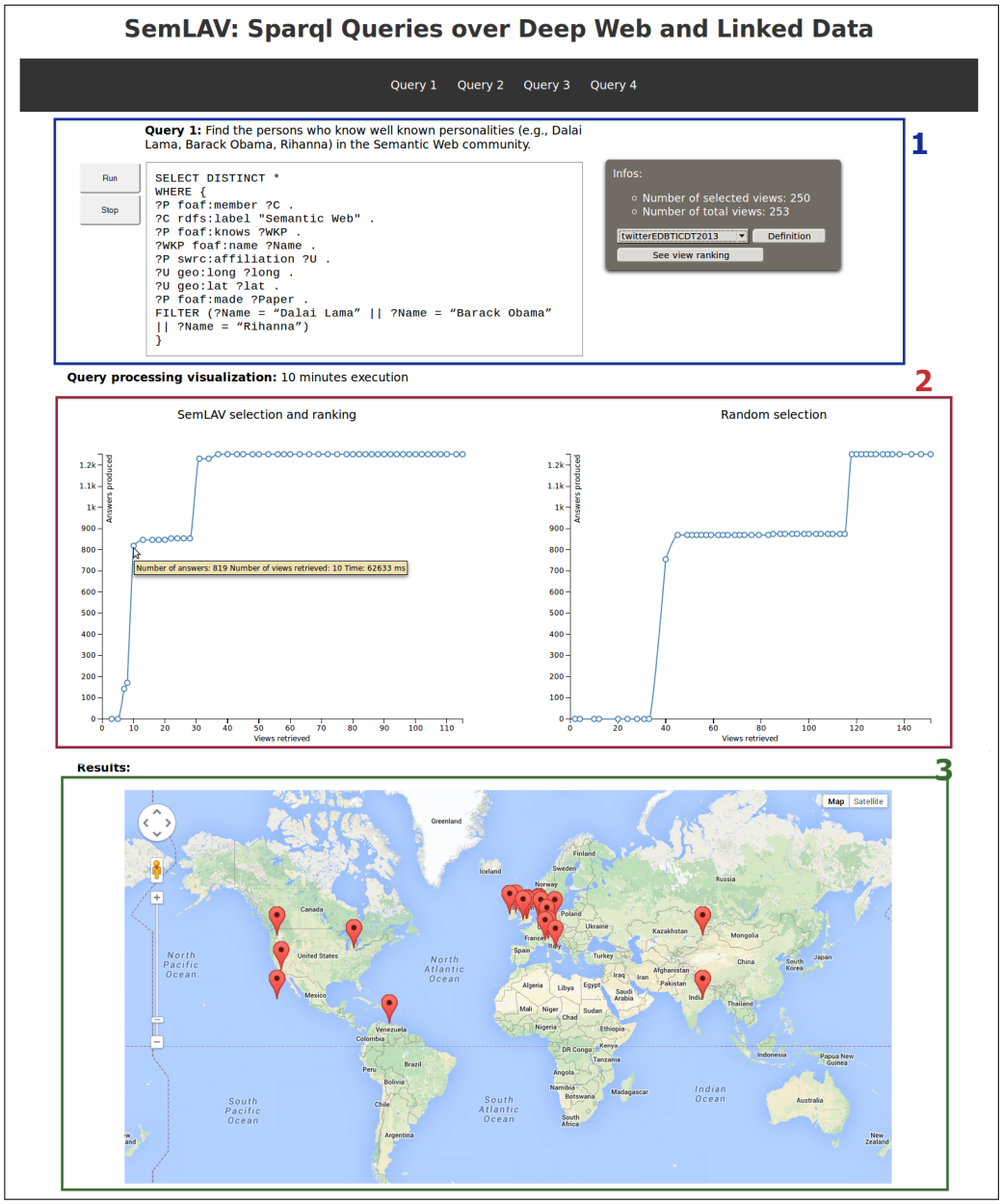


Fig. 3: Snapshots for Query 1 execution. During the demonstration all the reported results will be computed on-the-fly.