RDFDigest+: Exploring RDF/S KBs Using Summaries

Georgia Troullinou
ICS-FORTH, Greece
troulin@ics.forth.gr

Haridimos Kondylakis
ICS-FORTH, Greece
kondylak@ics.forth.gr

Kostas Stefanidis
University of Tampere, Finland
kostas.stefanidis@uta.fi

Dimitris Plexousakis
ICS-FORTH, Greece
dp@ics.forth.gr

ABSTRACT

Given the explosive growth in the size and the complexity of the Data Web, there is now more than ever, an increasing need to develop methods and tools in order to facilitate the understanding and exploration of RDF/S Knowledge Bases (KBs). To this direction, using as a starting point an abridged version of the original data source, highlighting the most representative concepts and providing further exploration operators, would greatly improve the exploitation potential of the information they contain.

In this paper, we present RDFDigest+, a novel tool that enables effective and efficient RDF/S KB exploration using summaries. The tool employees a diverse set of algorithms for identifying the most important nodes, offering a wide range of possibilities to capture importance. Then the selected nodes can be combined using multiple state of the art algorithms in order to generate a complete schema summary graph. In addition, we present a new approach enabling the dynamic exploration of summaries through two novel operations *zoom* and *extend*. Extend focuses on a specific subgraph of the initial summary, whereas zoom on the whole graph, both providing granular information access to the end-user. To make the system more responsive, we enable both offline and online calculations and we also provide approximations and progressive algorithms.

The aim of this demonstration is to dive in the exploration of the sources using summaries and to enhance the understanding of the various algorithms used. For the demonstration, we will use five diverse RDF/S KBs, in terms of size, connectivity and complexity, to show the functionality of the system. Then, we will allow conference participants to directly interact with the system to test its capabilities.

PVLDB Reference Format:
Georgia Troullinou, Haridimos Kondylakis, Kostas Stefanidis, and Dimitris Plexousakis. RDFDigest+: Exploring RDF/S KBs Using Summaries. *PVLDB*, 11 (5): xxxx-yyyy, 2018.
DOI: https://doi.org/TBD

Proceedings of the VLDB Endowment, Vol. 11, No. 5
Copyright 2018 VLDB Endowment 2150-8097/18/1.
DOI: https://doi.org/TBD

1. INTRODUCTION

The recent explosion of the Web of Data and the associated Linked Open Data (LOD) initiative have led to an enormous amount of widely available RDF datasets. These datasets often have extremely complex schemas, which are difficult to comprehend, limiting the exploitation potential of the information they contain. As a result, there is now, more than ever, an increasing need to develop methods and tools that facilitate the quick understanding and exploration of these data sources.

To this direction, many approaches focus on generating ontology summaries [4, 7] Ontology summarization [8] is defined as the process of distilling knowledge from an ontology in order to produce an abridged version. Although generating summaries is an active field of research, most of the works focus only on identifying the most important nodes, exploit limited semantic information or produce static summaries, limiting the exploration and the exploitation potential of the information they contain. In addition, although exploration operators over summaries have already been identified as really useful (e.g. [2]), the available approaches so far are limited, expanding only the hierarchy and the connections of selected nodes.

The lack of an ideal approach, led us to design and develop RDFDigest [5, 6, 1], for efficiently and effectively summarizing RDF/S KBs. The new version of the tool, RDFDigest+, is about to be released, and will soon be available online[1]. The new functionalities include multiple measures for identifying importance, multiple algorithms for linking nodes and a novel set of operators enabling the graphical exploration through summaries.

More specifically, RDFDigest+ focuses on three aspects: a) how to identify the most important nodes of an RDF/S KB, b) how to link those nodes in order to produce a valid sub-schema graph and c) how to enable the active exploration of the KB, presenting various statistical information and enabling zooming operators. Multiple importance measures (i.e. Degree, Betweeness, Bridging Centrality, Harmonic Centrality and Radiality [3]) have been adapted from graph theory and new ones have been constructed (i.e. Relevance and Ego Centrality), offering a wide range of alternatives for identifying the importance of a node. To identify the proper paths connecting those nodes, we model the problem either as a graph Steiner-tree problem or as a maximum cost-spanning tree one, employing approximations to speed-up the whole process. Finally, over these generated

[1]`http://rdfdigest.ics.forth.gr`

summaries, we enable zoom-in and zoom-out operations, in order to get granular information, by adding more important nodes or removing existing ones from the generated summary. In addition, through the extend operator, we allow selecting a subset of the presented nodes in order to visualize other dependent nodes. Both operators can be progressively applied in order to make the whole process more efficient.

In rest of the paper is structured as follows. Section 2 focuses on some preliminaries regarding RDF/S and Section 3 presents system architecture and the various algorithms to be demonstrated. Section 4 describes the demonstration process and finally Section 5 concludes this paper.

2. PRELIMINARIES

In this paper, we focus on RDF/S KBs, as RDF is among the widely-used standards for publishing and representing data on the Web. The representation of knowledge in RDF is based on triples of the form (subject, predicate, object). RDF datasets have attached semantics through RDFS, a vocabulary description language. Representation of RDF data is based on three disjoint and infinite sets of *resources*, namely: URIs ($\mathcal{U}$), literals ($\mathcal{L}$) and blank nodes ($\mathcal{B}$). We impose typing on resources, so we consider 3 disjoint sets of resources: classes ($\mathbf{C} \subseteq \mathcal{U} \cup \mathcal{B}$), properties ($\mathbf{P} \subseteq \mathcal{U}$), and individuals ($\mathbf{I} \subseteq \mathcal{U} \cup \mathcal{B}$). The set $\mathbf{C}$ includes all classes, including RDFS classes and XML datatypes (e.g., xsd:string, xsd:integer). The set $\mathbf{P}$ includes all properties, except rdf:type, which connects individuals with the classes they are instantiated under. The set $\mathbf{I}$ includes all individuals, but not literals. In addition, we should note that our approach adopts the unique name assumption, i.e. that resources that are identified by different URIs are different.

In this work, we separate between the schema and instances of an RDF/S KB, represented in separate graphs (G_S, G_I, respectively). The schema graph contains all classes and the properties they are associated with; note that multiple domains/ranges per property are allowed, by having the property URI be a label on the edge (via a labelling function λ) rather than the edge itself. The instance graph contains all individuals, and the instantiations of schema properties; the labelling function λ applies here as well for the same reasons. Finally, the two graphs are related via the τ_c function, which determines which class(es) each individual is instantiated under. Formally:

Definition 1. **(RDF/S KB)** An *RDF/S KB* is a tuple $V = \langle G_S, G_I, \lambda, \tau_c \rangle$, where:

- G_S is a labelled directed graph $G_S = (V_S, E_S)$ such that V_S, E_S are the nodes and edges of G_S, respectively, and $V_S \subseteq \mathbf{C} \cup \mathcal{L}$.
- G_I is a labelled directed graph $G_I = (V_I, E_I)$ such that V_I, E_I are the nodes and edges of G_I, respectively, and $V_I \subseteq \mathbf{I} \cup \mathcal{L}$.
- A labelling function $\lambda : E_S \cup E_I \mapsto \mathbb{P}(\mathbf{P})$ determines the property URI that each edge corresponds to (properties with multiple domains/ranges may appear in more than one edge) - $\mathbb{P}(\mathbf{P})$ the powerset of $\mathbf{P}$.
- A function $\tau_c : \mathbf{I} \mapsto 2^{\mathbf{C}}$ associating each individual with the classes that it is instantiated under.

For simplicity, we forego extra requirements related to RDFS inference (subsumption, instantiation) and validity (e.g., that the source and target of property instances should be instantiated under the property's domain/range, respectively), because these are not relevant for our results below.

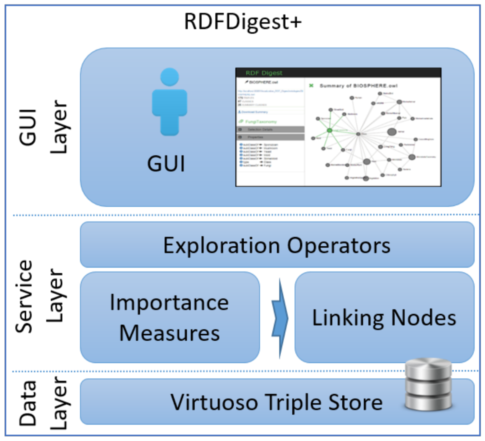


Figure 1: System Architecture.

3. SYSTEM ARCHITECTURE

The main architecture of the system is shown in Figure 1. It consists of three layers, the GUI layer, the service layer and the data layer. The summarization process starts by uploading an RDF/S file, or by providing the URL of a corresponding SPARQL endpoint. In case an external RDF/S file is provided, the file is stored to a Virtuoso triple store, enabling efficient query answering and processing. As soon as the file or the URL endpoint are provided, our engine preprocesses the available information and stores statistical and metadata information in a different Virtuoso graph. As long as existing information is available for a specific file/endpoint, it can be reused and not recomputed. Some of those additional metadata (like number of classes and instance, properties) are also presented through the interface to the end-users. Next, we focus on the service layer describing the various components and the corresponding algorithms there.

3.1 Identification of the most important nodes

The end-user can select the importance measure according to which the most important schema nodes will be selected. The available measures are the following:

- *Degree (DE).* The number of edges incident to a node.
- *Betweenness (BE).* The number of the shortest paths from all nodes to all others that pass through a node.
- *Bridging Centrality (BC).* The product of the betweenness centrality and the bridging coefficient, which measures the global and local features of a node, respectively. A node with high bridging centrality connects densely connected components in a graph.
- *Harmonic Centrality (HC).* The denormalized reciprocal of the harmonic mean of all distances to a node.
- *Radiality (RA).* Radiality centrality will give high centralities to vertices that are a short distance to every

other vertex in its reachable neighborhood compared to its diameter.

- *Ego Centrality (EC).* It shows the importance of a node to its neighborhood. It is the sum of the weighted number of adjacent incoming and outgoing edges of a node's neighbors.

As the aforementioned measures have been developed for generic graphs, we adapt them to be used for RDF/S graphs. To achieve that we first normalize each importance measure IM_i on a scale of 0 to 1:

$$normal(IM_i(v)) = \frac{IM_i(v) - \min(IM_i(g))}{\max(IM_i(g)) - \min(IM_i(g))}, \qquad (1)$$

where i one of the DE, BE, BC, HC, RA, EC. $IM_i(v)$ is the importance value of a node v in the schema graph g, $min(IM_i(g))$ is the minimum and $max(IM_i(g))$ is the maximum importance value in the graph. Similarly, we normalize the number of instances (InstV) that belong to a schema node. As such, the *adapted importance measure* (AIM) of each node is the sum of the normalized values of the importance measures and the instances.

$$AIM_i(v) = normal(IM_i(v)) + normal(InstV(v)) \qquad (2)$$

Besides the aforementioned importance measures, based on measures developed for generic graphs, we also include methods developed for RDF/S specifically, such as *Relevance* [5]. Relevance of a node combines both syntactic and semantic information, judging from the instances it contains, the number and type of incoming and outgoing edges, related also to its neigbors.

Our platform is flexible enough to enable the uninterrupted addition of new centrality measures by just adding new function calls. The diverse set of importance measures offered, enable end-users to explore RDF/S KBs according to way they perceive importance, offering many alternatives and enhancing the exploration abilities of our system.

3.2 Linking most important nodes

Having a way to rank the schema nodes of an RDF/S KB according to the perceived importance, we then select the top-k ones and focus on the paths that link those nodes, aiming to produce a valid sub-schema graph.

RDFDigest+ offers two methodologies for achieving this. The first one focuses on identifying a *maximum cost spanning tree (MST)* in the graph, and on linking the selected nodes using paths from the selected tree. The corresponding algorithm is computationally efficient. However, the main problems with this approach is that although the MST identifies the paths with the maximum weight in the whole graph, the paths selected out of the MST might not maximize the weight of the selected summary (remember that MST connects all nodes in the schema graph whereas summaries only select the most important nodes to be connected using paths from the MST). A second problem there is that many additional nodes are introduced in the result, since there is only one path to be selected between two nodes and in this path many other not important nodes might appear as well.

A different idea that we explore in this paper is to model the problem of linking the most important nodes as a variation of the well-known *Graph Steiner-Tree problem (GSTP)*. The corresponding algorithm tries to minimize the additional nodes introduced for connecting the top-k most important nodes. However, the problem is NP-hard, and as such approximation algorithms should be used for large datasets. As we have already experimented with several approximation algorithms and heuristic optimization procedures [3], RDFDigest+ employees the CHINS algorithm, proved to offer an optimal tradeoff between quality and execution time, starting with a partial solution and then finding one additional node each time.

3.3 Exploration through summaries

Getting the summaries, users can better understand the contents of a KB. However, still the user might find the presented information overwhelming and he/she may like to see less information, focusing for example, only on the top-10 nodes (zoom) or requesting more detailed information for a specific subgraph of the summary (extend).

Extend: The extend operator gets as input a subgraph of the schema graph and identifies other nodes that are depending on the selected nodes. Dependence has not only to do with distance. Like the TF-IDF concept, the basic hypothesis here is that the greater the influence of a property on identifying a corresponding instance is, the less times it is repeated. Specifically, we define the dependence between two classes as a combination of their cardinality closeness, the adapted importance measures (AIM) of the classes and the number of edges appearing in the path connecting these two classes. Given that this path consists of a set of nodes Y, dependence is defined as follows:

$$Dependence(u, v) = \frac{AIM(v) - \sum_{i \in Y} \frac{AIM(i)}{CC((i-1),i)}}{|Y|},$$

where the cardinality closeness CC is defined for a pair of classes as the number of distinct edges over the number of all edges between them.

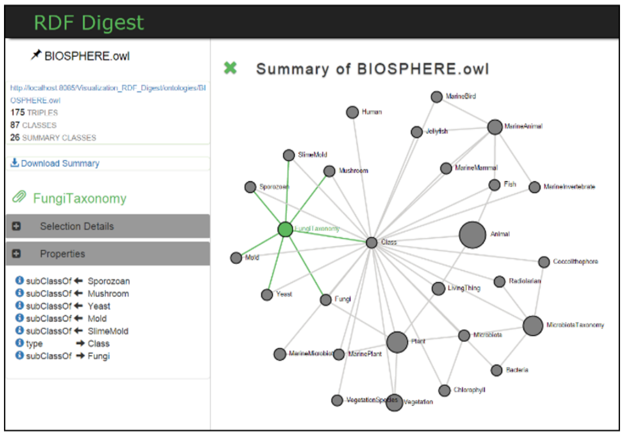
Obviously, as we move away from a node, the dependence becomes smaller by calculating the differences of AIM across a selected path in the graph. We penalize additionally dependence dividing by the distance of the two nodes. The highest the dependence of a path, the more appropriate is the first node to represent the final node of the path.

Zoom: Differently, we focus on zooming, by exploiting the schema graph as a whole. That is, we introduce the *zoom-out* and *zoom-in* operators to produce more detailed or coarse summary schema graphs. To this end, we consider the n' schema nodes with the highest importance in G_S, where n' can be either greater than n, for achieving a *zoom-out*, or smaller than n, for achieving a *zoom-in*, where n represents the size of the given summary.

To incrementally improve visualizations and support rapid decision making, our zoom operators are able to proceed in iterations. By exploiting this iterative mode, at the i^{th} iteration, we form a summary schema graph by performing a zoom-out or zoom-in on the schema graph constructed at the $(i-1)^t h$ iteration.

4. DEMONSTRATION

To demonstrate the functionalities of RDFDigest+ (an instance is shown in Figure 2), we will use five KBs: the BIOSPHERE ontology, the Financial ontology, the Aktors Portal ontology, the CIDOC-CRM ontology and DBPedia.



http://localhost:8085/Visualization_RDF_Digest/ontologies/BIOSPHERE.owl
175 TRIPLES
87 CLASSES
26 SUMMARY CLASSES

subClassOf ← Sporozoan
subClassOf ← Mushroom
subClassOf ← Yeast
subClassOf ← Mold
subClassOf ← SlimeMold
type → Class
subClassOf → Fungi

Figure 2: Extend and zoom operators.

BIOSPHERE (87 classes, 3 properties) models information in the domain of bio-informatics, the Financial ontology (188 classes, 4 properties) incudes classes and properties in the financial domain and the Aktors Portal (247 classes, 167 properties) describes an academic computer science community. Finally, CIDOC-CRM (82 classes, 273 properties) provides definitions and a formal structure for cultural heritage documentation, while DBPedia is a large KB containing 552 classes, 1805 properties and more than 3.3M instances. The variety on the size, the domain and the structure of these ontologies offers an interesting test case for our demonstration.

The demostration will begin by selecting one of the aforementioned ontologies and initially producing a visual summary identifying and linking the most important nodes in the KB (as the selected importance measure perceives importance). In the presented summary graph, the size of a node depends on the its importance. By clicking on a node, additional metadata (e.g. the number of instances, and the connected properties and instances) are provided to enhance the ontology understanding.

Further exploration of the data source is allowed by clicking on the details (on the left) of the selected class and properties. When clicked, its instances and connections appear in a pop-up window.

Moreover, further exploration of the data source is allowed by double-clicking on a node to extend the summary on that specific node. Besides a specific node, a whole area can be selected, requesting more detailed information to be presented regarding the selected nodes. In addition, the summary can be zoomed-in and zoomed-out in order to present more detailed or more generic information regarding the whole summary.

Finally, the user is able to download the summary as a valid RDFS document.

The demonstration will proceed in four phases.

- *Selecting important nodes.* In this phase, the whole idea of exploring RDF/S KBs through summaries will be described and the basics behind each importance measure will be highlighted. According to the user decisions, the results will be discussed.
- *Selecting linking algorithms.* Then, we will focus on the two linking algorithms employed, demonstrating how those two differ between each other, focusing on different perspectives for linking the most important nodes.
- *Exploration operators.* Then we will demonstrate how the two operators employed help to dive in more detailed or coarser information, discussing also the incremental approach on applying those operators.
- *Hands-on phase.* In this phase, conference participants would be invited to directly interact with the system, exploring through summaries one or more of the aforementioned ontologies.

The system will be available also online for the conference participants to experiment with.

5. CONCLUSION

In this demonstration, we present a new platform that enables effective exploration of RDF/S KBs through summaries. This is achieved by selecting first the most important nodes, then linking those nodes and finally enabling exploration operators in the result summary. We will demonstrate our platform using real, diverse datasets, and will allow the direct participation of the conference participants to test its capabilities. To our knowledge, no other system today is available on the web, enabling the summarization of RDF/S KB, providing functionalities for active data exploration through summaries.

6. ACKNOWLEDGMENTS

Acknowledgments: This work was partially supported by the EU project iManageCancer (H2020-643529).

7. REFERENCES

[1] H. Kondylakis, L. Koumakis, M. Psaraki, G. Troullinou, M. Chatzimina, E. Kazantzaki, K. Marias, and M. Tsiknakis. Semantically-enabled personal medical information recommender. In *ISWC*, 2015.

[2] E. Motta, S. Peroni, N. Li, and M. d'Aquin. Kc-viz: A novel approach to visualizing and navigating ontologies. In *EKAW*, 2010.

[3] A. Pappas, G. Troullinou, G. Roussakis, H. Kondylakis, and D. Plexousakis. Exploring importance measures for summarizing RDF/S KBs. In *ESWC*, 2017.

[4] P. Silvio, M. Enrico, and d'Aquin Mathieu. Identifying key concepts in an ontology, through the integration of cognitive principles with statistical and topological measures. In *ASWC*, pages 242–256, 2008.

[5] G. Troullinou, H. Kondylakis, E. Daskalaki, and D. Plexousakis. RDF digest: Efficient summarization of RDF/S KBs. In *ESWC*, 2015.

[6] G. Troullinou, H. Kondylakis, E. Daskalaki, and D. Plexousakis. Ontology understanding without tears: The summarization approach. *Semantic Web*, 8(6):797–815, 2017.

[7] G. Wu, J. Li, L. Feng, and K. Wang. Identifying potentially important concepts and relations in an ontology. In *ISWC*, pages 33–49. Springer, 2008.

[8] X. Zhang, G. Cheng, and Y. Qu. Ontology summarization based on rdf sentence graph. In *WWW*, 2007.