Exploring Importance Measures for Summarizing RDF/S KBs

Alexandros Pappas[1], Georgia Troullinou[2], Giannis Roussakis[2], Haridimos Kondylakis[2(✉)], and Dimitris Plexousakis[2]

[1] Computer Science Department, University of Crete, Heraklion, Greece
apappas@csd.uoc.gr
[2] Institute for Computer Science, FORTH, Heraklion, Greece
{troulin,rousakis,kondylak,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, summarization approaches try to produce an abridged version of the original data source, highlighting the most representative concepts. Central questions to summarization are: how to identify the most important nodes and then how to link them in order to produce a valid sub-schema graph. In this paper, we try to answer the first question by revisiting six well-known measures from graph theory and adapting them for RDF/S KBs. Then, we proceed further to model the problem of linking those nodes as a graph Steiner-Tree problem (GSTP) employing approximations and heuristics to speed up the execution of the respective algorithms. The performed experiments show the added value of our approach since (a) our adaptations outperform current state of the art measures for selecting the most important nodes and (b) the constructed summary has a better quality in terms of the additional nodes introduced to the generated summary.

Keywords: Semantic summaries · Schema summary · RDF/S Knowledge Bases · Graph theory

1 Introduction

The recent explosion of the Data Web 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 exploration and the exploitation potential of the information they contain. In addition, a user, in order to formulate queries, has to examine carefully the entire schema in order to identify the interesting elements of the schema and the data. As a result, there is now, more than ever, an increasing need to develop methods and tools in order to facilitate the quick understanding and exploration of these data sources [11].

E. Blomqvist et al. (Eds.): ESWC 2017, Part I, LNCS 10249, pp. 387–403, 2017.
DOI: 10.1007/978-3-319-58068-5_24

To this direction several works try to provide overviews on the ontologies [15,20,25] maintaining however the most important ontology elements. Such an overview can also be provided by means of an ontology summary. Ontology summarization [29] is defined as the process of distilling knowledge from an ontology in order to produce an abridged version. While summaries are useful, creating a good summary is a non-trivial task. A summary should be concise, yet it needs to convey enough information to enable a decent understanding of the original schema. Moreover, the summarization should be coherent and provide an extensive coverage of the entire ontology.

In this paper, we focus on RDF/S Knowledge Bases (KBs) and explore efficient and effective methods to automatically create high-quality summaries. The goal is to construct better summaries in terms of selecting the most important part of the schema as end-users perceive importance. We view an RDF/S KB as two distinct and interconnected graphs, i.e. the schema and the instance graph. As such, a summary constitutes a valid sub-schema graph containing the most important nodes, summarizing the instances as well. Central questions to the process of summarization is how to identify the most important nodes and then how to link those nodes to produce a valid sub-schema graph. For answering the first question various importance measures have been proposed trying to provide real-valued measures for ranking the nodes of a graph. In this paper, we adapt, for RDF/S KBs, six well-known importance measures from graph theory, covering a wide range of alternatives for identifying importance. Then we try to answer the second question by modelling the problem of selecting a valid sub-schema graph as a Steiner-Tree problem which we resolve using approximations with heuristics. More specifically our contributions are the following:

- We explore the *Degree*, the *Betweeness*, the *Bridging Centrality*, the *Harmonic Centrality*, the *Radiality* and the *Ego Centrality* measures adapting them for RDF/S KBs to consider instance information as well. Our experiments show that the adapted versions of these importance measures greatly outperform other proposed measures in the domain.
- Besides identifying the most important nodes, we try next to identify the proper paths connecting those nodes. We achieve this by modelling the problem as a graph Steiner-Tree Problem trying to minimize the total number of the additional nodes introduced when constructing the summary sub-graph. Since the problem is NP-complete and the exact algorithms proposed require significant execution time, we proceed further to explore three approximations, the SDIST, the CHINS and the HEUM trying to optimize either the insertion of a single component or the connection of the components using their shortest paths. On top of these approximations we implement an improvement procedure using heuristics the I-MST, ensuring that all leaves are terminal nodes.
- Finally, we perform a detailed two-stage experimental evaluation using two versions of the DBpedia. In the first stage we compare the applicability of the adapted measures for identifying the nodes' importance comparing our adaptations with the most frequent nodes queried in the corresponding query logs. We identify that overall our adaptation of Betweeness outperforms all

other important measures. In the second stage, we evaluate the quality the of the selected sub-graphs in terms of additional nodes introduced showing that CHINS performs better without too much overhead in the execution time.

To the best of our knowledge, this is the first time that these six diverse importance measures are adapted and compared for RDF/S summarization purposes. In addition although other recent works focus on using the maximum cost spanning tree [24,25] for linking the selected nodes, this is the first time the problem of summarization is formulated as a Steiner-Tree problem using approximations for the fast identification of the corresponding summaries with many benefits as we shall show in the sequel.

The rest of the paper is organized as follows: Sect. 2 introduces the formal framework of our solution and Sect. 3 describes the various measures for estimating importance. Then in Sect. 4 we present the algorithms for selecting the proper subgraphs, whereas Sect. 5 presents our evaluation. Section 6 presents related work and finally Sect. 7 concludes this paper and presents directions for future work.

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. 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:*

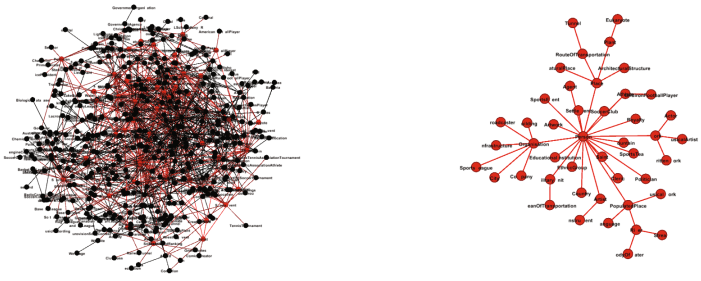
- G_S *is a labelled directed graph* $G_S = (V_S, E_S)$ *such that* V_S, E_S *are the nodes and edges of* G_S, *respectively, and* $V_S \subseteq \mathbf{C} \cup \mathcal{L}$.

- *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 $\lambda : E_S \cup E_I \mapsto 2^{\mathbf{P}}$ determines the property URI that each edge corresponds to (properties with multiple domains/ranges may appear in more than one edge).*
- *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 and would significantly complicate our definitions.

In the following, we will write $p(v_1, v_2)$ to denote an edge e in G_S (where $v_1, v_2 \in V_S$) or G_I (where $v_1, v_2 \in V_I$) from node v_1 to node v_2 such that $\lambda(e) = p$. In addition, a path from a schema node v_s to v_i, denoted by $path(v_s, v_i)$, is the finite sequence of edges, which connect a sequence of nodes, starting from the node v_s and ending in the node v_i. The length of a path, denoted by $d_{path(v_s,v_i)}$, is the number of the edges that exist in that path whereas $d(v_s, v_i)$ is the number of the edges that exist in the shortest path linking v_s and v_i. Finally, having a schema/instance graph G_S/G_I, the closure of G_S/G_I, denoted by $Cl(G_S)/Cl(G_I)$, contains all triples that can be inferred from G_S/G_I using inference. Since in our algorithms we use the closure of the corresponding schema/instance graphs, from now on when we use G_S/G_I we will mean $Cl(G_S)/Cl(G_I)$ for reasons of simplicity unless stated otherwise. This is to ensure that the result will be the same, independent of the number of inferences applied in the input.

Now as an example, consider the DBpedia 3.8 shown in Fig. 1(a). Obviously, it is really difficult to examine all the nodes in order to understand the schema. However, focusing only on the schema summary, shown in Fig. 1(b), allows the user to get a quick overview on the contents of the ontology, identifying and linking the most important nodes. We have to note, that our approach handles OWL ontologies as well, considering however only the RDF/S fragment of these ontologies.



(a) (b)

Fig. 1. The DBpedia 3.8 schema graph (a) and a corresponding schema summary (b).

3 Importance Measures

Schema summarization aims to highlight the most representative concepts of a schema, preserving important information and reducing the size and the complexity of the schema [17]. Despite the significance of the problem, there is still no universally accepted measurement on the importance of nodes in an RDF/S graph. In this section, we describe six alternative measures that have been proposed for capturing importance in directed graphs. Then we will show how we adapt those measures to consider instance information as well for summarizing RDF/S KBs. We selected the *Betweeness*, the *Bridging Centrality*, the *Degree*, the *Harmonic Centrality*, the *Radiality* and the *Ego Centrality* as they constitute the state of the art geometric measures for generic graphs [2]. We do not compare with spectral measures (HITS, PageRank etc.) because they are based on external factors and spectral properties. The complexities of all aforementioned measures is shown in Table 1.

Table 1. The complexities of the examined importance measures.

Measure	Complexity
Degree (DE)	$O(V_S + E_S)$
Betweeness (BE)	$O(V_S \cdot (V_S \cdot E_S))$
Bridging Centrality (BC)	$O(V_S \cdot (V_S \cdot E_S))$
Harmonic Centrality (HC)	$O(V_S \cdot (V_S + E_S))$
Radiality (RA)	$O(V_S \cdot (V_S + E_S))$
Ego Centrality (EC)	$O(V_S + E_S)$

– The simplest importance measure for a graph is the Degree, that is defined as the number of edges incident to a node.

Definition 2 ***(Degree).*** *Let $G_S = (V_S, E_S)$ be an RDF/S schema graph with V_S nodes and E_S edges. The Degree of a node $v \in V_S$ is defined as follows:*

$$DE(v) = deg(v) \tag{1}$$

where $deg(v)$ is the number of edges incident to the node.

– The Betweenness measure is equal to the number of the shortest paths from all nodes to all others that pass through that node. Calculating the betweenness for all nodes in a graph requires the computation of the shortest paths between all nodes.

Definition 3 ***(Betweenness).*** *Let $G_S = (V_S, E_S)$ be an RDF/S schema graph. The Betweenness of a node $v \in V_S$ is defined as follows:*

$$BE(v) = \sum_{s \neq v \neq t} \frac{\sigma_{st}(v)}{\sigma_{st}} \tag{2}$$

where σ_{st} is the total number of shortest paths from node s to node t and $\sigma_{st}(v)$ is the number of those paths that pass through v.

– The Bridging Centrality tries to identify the information flow and the topological locality of a node in a network. It is widely used for clustering or in order to identify the most critical points interrupting the information flow for network protection and robustness improvement purposes. A node with high Bridging Centrality is a node connecting densely connected components in a graph. The bridging centrality of a node is the product of the betweenness centrality and the bridging coefficient, which measures the global and local features of a node respectively.

Definition 4 ***(Bridging Centrality).*** *Let $G_S = (V_S, E_S)$ be an RDF/S schema graph. The bridging centrality of a node $v \in V_S$ is defined as follows:*

$$BC(v) = B_C(v) \cdot BE(v) \tag{3}$$

where $B_C(v)$ is the bridging coefficient of a node which determines how well the node is located between high degree nodes and $BE(v)$ is the betweenness centrality. The bridging coefficient of a node v is defined:

$$B_C(v) = \frac{DE(v)^{-1}}{\sum_{i \in N(v) \cdot \frac{1}{DE(i)}}} \tag{4}$$

where $DE(v)$ is the degree of node v, and $N(v)$ is the set of it's neighbors.

– The Harmonic Centrality was initially defined for undirected graphs by Rochat [19] in 2009 and later for directed graphs by Boldi and Vigna [2]. It is a modification of the *Closeness* [2], replacing the average distance with the harmonic mean of all distances, requiring again the computation of the shortest paths between all nodes.

Definition 5 ***(Harmonic Centrality).*** *Let $G_S = (V_S, E_S)$ be an RDF/S schema graph. The Harmonic Centrality of a node $v \in V_S$ is defined as follows:*

$$HC(v) = \frac{1}{\sum_{u \neq v} d(u, v)} \tag{5}$$

– The Radiality was first proposed by Valente and Foreman [26], to provide information on how close a node is to all other nodes in a graph (i.e. the integration measure of a node to a graph). In order to compute the diameter of a graph we need to compute the shortest paths between all nodes.

Definition 6 ***(Radiality).*** *Let $G_S = (V_S, E_S)$ be an RDF/S schema graph. The Radiality of a node $v \in V_S$ defined as:*

$$RA(v) = \frac{1}{\sum_{u \neq v} (\Delta_G - (1/d(u, v))} \tag{6}$$

where Δ_{G_S} is the Diameter of graph G_s.

– The Ego Centrality (EC) was first introduced in the iManageCancer[1] project. For a node v, EC is the induced subgraph of G, which contains v, its neighbors, and all the edges between them, trying to identify how important a node is to his neighborhood.

Definition 7 ***(Ego Centrality).*** *Let $G_S = (V_S, E_S)$ be an RDF/S schema graph. The Ego Centrality of a node $v \in G_S$ is defined as follows:*

$$EC(v) = \sum_{i=1}^{i=n^{in}} W_i * e.ego_i + \sum_{i=1}^{i=n^{out}} W_i * e.ego_i \tag{7}$$

where:

$$W_i = \sum_{i=1}^{i=n^{in}} 1/v_i^{out} + \sum_{i=1}^{i=n^{out}} 1/v_i^{in} \tag{8}$$

and $e.ego = 1/v_i^{out}$, v_i the adjacent node of a node v using the incoming edge e and $e.ego = 1/v_i^{in}$, v_i the adjacent node of a node v using the outgoing edge e.

3.1 Adapted Importance Measures

In order to take into consideration the instances of each class, we adapt the aforementioned importance measures. To achieve that we first normalize each importance measure IM_i on a scale of 0 to 1:

$$normal(IM_i(v)) = \frac{IM_i(v) - \min(IM_i(g))}{\max(IM_i(g)) - \min(IM_i(g))} \tag{9}$$

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.

$$AIM_i(v) = normal(IM_i(v)) + normal(InstV(v)) \tag{10}$$

4 Construction of the RDF/S Summary Schema Graph

Using the aforementioned AIM we select the top-k important nodes of a directed schema graph (also known as *terminals* in graph theory). Then we to focus on the paths that link those nodes, trying to produce a valid sub-schema graph. We have to note that in the stage of constructing the final RDF/S summary schema graph we are not interested in the direction of the edges since we only want to get a connected schema sub-graph.

[1] http://imanagecancer.eu/.

The latest approaches in the area [25] identify a maximum cost spanning tree (MST) in the graph and then link the most important nodes using paths from the selected maximum-cost spanning tree. The main idea there is to select the paths that maximize the total weight of the selected sub-graph. 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) exploiting the optimal solutions there proposed by Hakimi [6] and Levin [12] independently. The problem is an NP-hard [8] problem and remains NP-complete if all edge weights are equal.

Definition 8 ***(The Graph Steiner-Tree problem (GSTP)).*** *Given an undirected graph $G = (V, E)$, with edge weights $w : E \rightarrow \mathbb{R}^+$ and a node set of terminals $S \subseteq V$, find a minimum-weight tree $T \in G$ such that $S \subseteq V_t$ and $E_t \subseteq E$.*

In our case, we consider as G the G_S ignoring as well the direction in the edges. As such the objective now is not to increase the total weight of the summary graph but to introduce as much as possible less number of additional nodes. This is due to the fact that introducing a lot of additional nodes shifts the focus of the summary and decreases summary's quality.

4.1 Algorithms, Approximation and Heuristics

There had been various exact algorithms for the GSTP. Hakimi [6] proposed the first brute force algorithm that enumerates all minimum spanning trees of sub-networks of G included by super-sets of terminals that runs in $O(2^{V-t} \cdot V^2 + V^3)$. The first dynamic programming algorithms were proposed independently by Dreyfus & Wagner [4] and by Levin [12]. The former runs in $O(3^t \cdot V + 2^t \cdot V^2 + V^3)$ whereas the latter in $O(3^t \cdot V + 2^t \cdot V^2 + t^2 \cdot V)$ and they are based on the optimal decomposition property by creating two node sets, removing one node at each step and solving the GSTP by connecting each set. Levin's method uses a recursive optimization approach that pre-computes the possible sub-trees. Since all aforementioned algorithms have an exponential running time, various approximations such as [16,18,27] have been proposed in order to find good solutions for large networks. A central theme in these approximations, is the use of some principles known from the two classic algorithms for solving the minimum spanning tree problem, Prim's and Kruskal's [27]. We will use the following top-three well-known and good performing methods SDISTG, CHINS and HEUM [27]. These

approximations have a worst case bound of 2, i.e., $Z_T/Z_{opt} \leq 2 \cdot (1 - l/|Q|)$, where Z_T and Z_{opt} denote the objective function values of a feasible solution and an optimal solution respectively, Q the set of terminals and l a constant [1].

SDISTG (Shortest distance graph)

1. Construct a complete graph G' for the node set Q (set of terminal nodes) with each edge having the weight of a shortest path between the corresponding nodes in G.
2. Construct a minimum spanning tree T' of G'.
3. Replace each edge of the tree T' by its corresponding shortest path in G.

CHINS (Cheapest insertion)

1. Start with a partial solution $T = (w, 0)$ consisting of a single terminal node w.
2. While T does not contain all terminal nodes do
   find the nearest nodes $u* \in V_t$ and $p*$ being a terminal node not in V_t.

HEUM (Heuristic measure)

1. Start with a partial solution $T = (Q, 0)$ consisting of Q singleton components (terminal nodes).
2. While T is not connected do
   choose a node u using a heuristic function F and unite the two components of T which are nearest to u by combining them with u via shortest paths (the nodes and edges of these paths are added to T).
   Up to now the most promising way is to choose F according to:
   $\min_{i \leq t \leq \sigma,} \left\{ \frac{1}{t} \cdot \sum_{i=0}^{t} d(u, T_i) \right\}$ where $T_0, \ldots, T_\sigma$ are the components of T such that $d(u, T_i) \leq d(u, T_i) \forall i, j \in \sigma, i < j$.

Besides these approximations, many heuristics can be employed to improve even more the corresponding algorithms. The most promising ones are the I-MST+P and the TRAFO [27]. I-MST+P is a pruning routine that ensures that all leaves are terminal nodes whereas TRAFO transforms a feasible solution to another one trying to overcome the deficiency of bad local optima by allowing the temporary deterioration of the actual solutions. In this paper we use only the I-MST+P since TRAFO requires considerable more time to run and the improvements are insignificant - due to the sparsity of the examined ontologies.

I-MST+P (Improvement procedure with MST+P)

1. Let $T = (V_t, E_t)$ be a feasible solution of the GSTP. The subgraph of G induced by V_t will be defined as G_t.
2. Construct a minimum spanning tree $T = (V'_t, E'_t)$ of G_t.
3. While there exists a leaf of T' being a terminal do
   delete that leaf and its incident edge.

Table 2. Worst-case complexities for linking the most important nodes in a graph.

Algorithm	Weighted graph	Un-weighted graph
MST	$O(E \cdot logV)$	$O(\|V + E\|)$
SDISTG	$O(Q \cdot \|VlogV\|)$	$O(Q \cdot \|V + E\|)$
CHINS	$O(Q \cdot \|VlogV\|)$	$O(Q \cdot \|V + E\|)$
HEUM	$O(V \cdot \|VlogV\|)$	$O(V \cdot \|V + E\|)$

Complexities. Using a Breadth-first search requires $O(V + E)$ time (where E is $O(V)$) to find shortest paths from a source vertex v to all other vertices in the graph. All-pairs shortest paths for unweighted undirected graphs can be computed in $O(V \cdot (V + E))$ time by running the BFS algorithm for each node of the graph. The complexity of SDISTG, CHINS and HEUM differs, due to the usage of different heuristics. Table 2 provides the worst case complexities of those algorithms for weighted/un-weighted graphs.

5 Evaluation

To evaluate our approach, we used two versions of the DBpedia[2]. DBpedia 3.8 is consisted of 359 classes, 1323 properties and more that 2.3M instances, whereas DBpedia 3.9 is consisted of 552 classes, 1805 properties and more than 3.3M instances. Those two versions offer two interesting use-cases for exploration.

To identify the most important nodes of those two versions we do not rely on a limited amount of domain experts with subjective opinions as past approaches do [20,23,25]. Instead, we exploit the query logs from the corresponding DBpedia endpoints trying to identify the schema nodes that are more frequently queried. For DBpedia 3.8 we were able to get access to more than 50K queries whereas for 3.9 we were able to get access to more than 110K queries.

For each examined version, we considered the corresponding query log trying to identify the most important classes. We assess as the most important, the ones that have higher *frequency of appearance* in the queries. A class appears within a query either directly or indirectly. Directly when the said class appears within a triple pattern of the query and undirectly when (a) the said class is the type of an instance or the domain/range of a property that appear in a triple pattern of the query.

In addition, we compare our approach with *relevance*, another measure recently published, combining both syntactic and semantic information, shown to outperform past approaches in the area [23,25].

5.1 Spearman's Rank Correlation Coefficient

Initially we tried to understand the statistical dependence between the ranking of all nodes using the aforementioned measures. To do this we used the Spearman's

[2] http://wiki.dbpedia.org/.

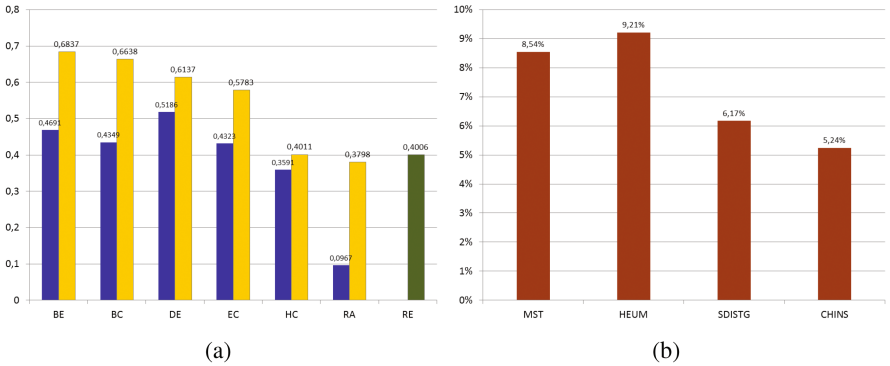


Fig. 2. (a) Spearman's rank correlation for the adapted (yellow) and the non-adapted (blue) importance measures with the frequency ranking and (b) the percentage of additional nodes introduced (Color figure online)

rank correlation coefficient [21], a nonparametric measure of rank correlation. It assesses how well the relationship (measures the strength and direction of association) between two variables can be described using a monotonic function.

Spearman correlation indicates the direction of the association between two variables X, Y. It can vary between -1 and 1, where 1 is total positive correlation, 0 is no correlation, and -1 is total negative correlation. If Y tends to decrease when X increases, the Spearman correlation coefficient is negative. A Spearman correlation of zero indicates that there is no tendency for Y to either increase or decrease when X increases. When X and Y are perfectly monotonically related, the Spearman correlation coefficient becomes 1. The results of our experiments are shown in Fig. 2(a).

As shown, our adapted importance measures show a higher dependence to the frequency ranking than the pure structural ones wit. In addition we can see that measures like the AIM_{BE}, the AIM_{BC} and the AIM_{DE} show a really high correlation with the frequency ranking. Finally, we can see that all adapted measures - except Radiality (AIM_{RA})- show a better correlation than Relevance.

5.2 The Similarity Measure

Next, we would like to evaluate the measures identified in Sect. 3 for their quality with respect to identifying the nodes' importance. Measures like precision, recall and F-measure, used by the previous works [15,17,20] are limited in exhibiting the added value of a summarization system because of the "disagreement due to synonymy" [3] meaning that they fail to identify closeness with the ideal result when the results are not exactly the same with the reference ones. On the other hand, content-based metrics compute the similarity between two summaries in a more reliable way [29]. To this direction, we use the *similarity measure*, first defined in [23], denoted by $Sim(G_S, G_R)$, in order to define the level of

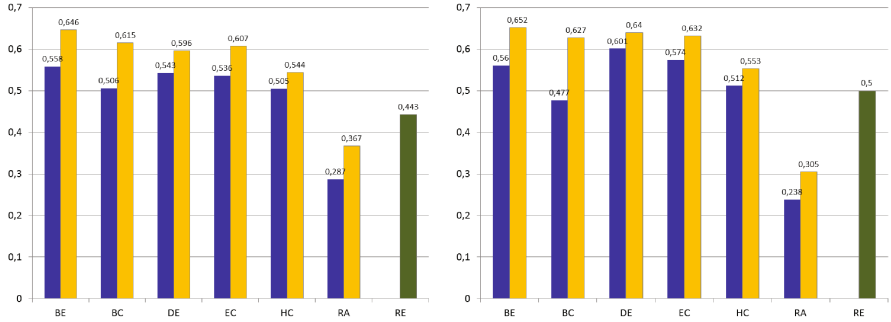
agreement between an automatically produced graph summary $G_S = (V_S, E_S)$ and a reference graph summary $G_R = (V_R, E_R)$:

$$Sim(G_S, G_R) = \frac{|V_S \cap V_R| + a \cdot \sum_{i=k}^{p} \frac{1}{d_{p(c_i, c'_i)}} + b \cdot \sum_{i=m}^{n} \frac{1}{d_{p(c_i, c'_i)}}}{|V_R|}$$

where $c_k, ..., c_p$ are the classes in V_R that are sub-classes of the classes $c'_k, ..., c'_p$ of V_S and that $c_m, ..., c_n$ are the classes in V_R that are superclasses of the classes $c'_m, ..., c'_n$ of V_S. In the above definition a and b are constants assessing the existence of sub-classes and super-classes of G_S in G_R with a different percentage. In [25] the ideal weights for RDF/S KBs have been identified to be $a = 0.6$ and $b = 0.3$ which we use in this paper as well, giving more weight to the super-classes. The idea behind that is that the super-classes, since they generalize their sub-classes, are assessed to have a higher weight than the sub-classes, which limit the information that can be retrieved. Consequently, the effectiveness of a summarization system is calculated by the average number of the similarity values between the summaries produced by the system and the set of the corresponding experts' summaries. The results of our experiments are shown in Fig. 3 and present the average similarity values for generating summaries from 1% to 50% of the corresponding schema graph size. As shown again our adapted measures (in yellow) outperform the pure structural ones (in blue) in all cases. In addition, all measures but AIM_{BA} outperform Relevance showing again the high value of our adaptations. When comparing between the ontology versions we can observe that although AIM_{BE} is the clear winner in all cases, the second best in DBpedia 3.8 is the AIM_{BC} whereas in DBpedia 3.9 is the AIM_{DE}. To interpret these results we shall consider that 193 more classes were added in DBPedia 3.9 introducing only a small number of new edges. This results in a reduction of 37% of the density and an increase of the diameter from 9 to 13. As such, only a few number of nodes have more than one out-going edge and the degree performs better in this case as it captures more effectively the importance of more sparse graphs.

5.3 Additional Nodes Introduced

Next we would like to identify the overhead imposed by the algorithms for linking the most important nodes in terms of the additional nodes that are introduced. The average number of additional nodes introduced per algorithm is shown in Fig. 2(b). We can observe that MST used by our previous work introduces on average 8.5% of additional nodes, whereas CHINS only 4.7% additional nodes. For example, for DBpedia 3.9 this corresponds to 19 additional nodes using MST over CHINS when requesting a summary of 10% of the nodes. This is reasonable, since the Steiner-Tree approximations have the objective of minimizing the additional nodes introduced in the selected subgraph confirmed by our experiments.



(a) DBpedia 3.8 (b) DBPedia 3.9

Fig. 3. Comparing the average similarity of the adapted (in yellow) and the non-adapted (in blue) importance measures in (a) DBpedia 3.8 and (b) DBpedia 3.9 for a summary of 1–50%. (Color figure online)

5.4 Execution Time

Finally, to test the efficiency of our system, we measured the average time of 50 executions in order to produce the corresponding summaries of the two KBs. The experiments run on a Intel(R) Xeon(R) CPU E5-2630 running at 2.30 GHz with 64 GB memory running Ubuntu 12.04 LTS.

The mean execution times for identifying the most important nodes and constructing the corresponding summaries are shown in Fig. 4. As we can observe, the execution times of the various measures can be divided into three categories. The measures that need to compute the shortest paths of all pairs, the measures that need to iterate only the nodes and the edges of the graph and the measures that need to execute queries on external databases or combine complex measures. The Betweenness, the Bridging Centrality, the Harmonic Centrality and the Radiality belong to the first category since they assign weights by calculating the shortest paths between all pairs of nodes. As such they have similar execution times. The Betweenness differentiates from the rest since the set of all shortest paths should be computed for each pair of nodes. The Bridging Centrality uses the Betweenness and as such it takes almost the same time. In the second category we find the Degree and the Ego Centrality. The Degree needs only to iterate over all edges of the graph and "submit" the weight to each node. The Ego Centrality needs one more iteration over all nodes and edges of the graph. Relevance is not included in these graphs since it requires significantly more time (two orders of magnitude larger) using an external triple store to be able to handle mass amounts of data /while the aforementioned algorithms load everything in memory.

For linking the most important nodes, the complexity the SDISTG and CHINS approximation algorithms show that there is a linear function relationship between their execution time and the input data size (the number of the nodes and the edges). HEUM is the only one that has a quadratic time, and this

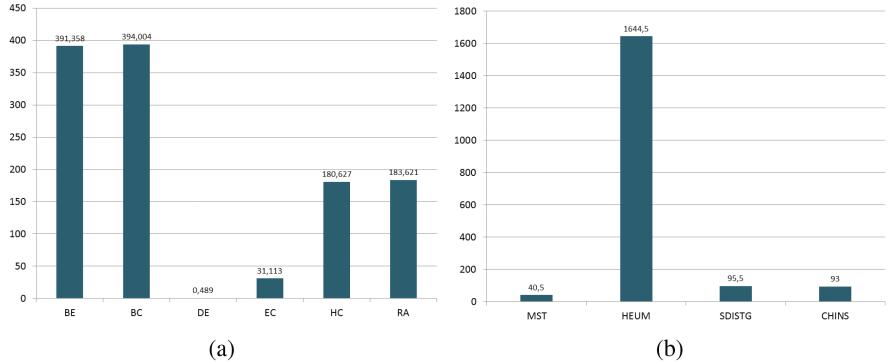


Fig. 4. The average execution times of the importance measures (msec) (a) and the algorithms for linking those nodes (msec) (b)

is due to the fact that it has to construct the shortest paths of all pairs. As such SDISTG and CHINS have a better execution time as the number of terminal nodes is small and HEUM the worst execution time, which grows linearly to the number of nodes. MST is slightly faster than other algorithms because it depends on the size of graph (nodes and edges), in contrast to CHINS and SDISTG that are highly dependent on the number of the terminals and the shortest paths between them.

6 Related Work

The latest years summarization approaches for linked data are constantly gaining ground. For example, a wide variety of research works [5,9,10,14] focus on extracting statistics and producing visual summaries of linked datasets, try to create mainly instance summaries, exploiting the instances' semantic associations [7,13,22] or focus on peer-to-peer systems [15]. However, our system differs from the above in terms of both goals and techniques.

More closely related works are Peroni et al. [20] and Wu et al. [28]. The former [20] try to automatically identify the key concepts in an ontology, combining cognitive principles, lexical and topological measurements such as the density and the coverage, whereas in the latter the authors [28] use similar algorithms to identify the most important concepts and relations in an iterative manner. However, both of these works focus only on returning the most important nodes and not on returning an entire graph summary. Zhang et al. [29] uses measures such as the degree-centrality, the betweenness and the eigenvector centrality to identify not the most important nodes but the most important RDF sentences. In Queiroz-Sousa et al. [17] the authors try to combine user preferences with the degree centrality and the closeness to calculate the importance of a node and then they use an algorithm to find paths that include the most important nodes in the final graph. However the corresponding algorithm prioritizes direct neighbors ignoring that the selection of other paths that could maximize the total

importance of the selected summary. Finally Troulinou et al. [23,25] employee relevance for identifying the most important nodes and then they try to connect those nodes by generating and pruning appropriately the maximum cost spanning tree. However, many additional nodes might be introduced and the selected summary does not guarantee to maximize the total importance of the selected sub-graph.

7 Discussion and Conclusion

In this paper, we try to provide answers to the two main questions in constructing RDF/S summaries: how to identify the most important nodes and how to link the selected nodes to create the final summary. To this direction, we adapt six diverse measures for identifying node's importance and we implement three graph Steiner-Tree approximations in order to link those nodes.

To evaluate our approach we do not rely on subjective reference summaries generated by a limited amount of domain experts but instead we exploit the query logs from the DBpedia endpoints. The performed evaluation shows that the adapted measures perform significant better that the pure structural ones for RDF/S KBs. In addition all but the adapted version of Radiality outperform past approaches in the area. The adaptation of Betweeness is the winner in all cases. In addition, we show that the Steiner-Tree approximation algorithms introduce less additional nodes to the result schema graph. CHINS seems to be the best choice in terms of the quality of the generated summary offering an optimal trade-off between quality and execution time.

As future work, an interesting topic would be to extend our evaluation to spectral properties as well or to focus on how to combine the various measures in order to achieve the best results according to the specific characteristics of the input ontologies. Finally, another interesting topic would be to extend our approach to handle more constructs from OWL ontologies such as class restrictions, disjointness and equivalences.

Acknowledgments. This work was partially supported by the EU projects iManageCancer and CloudSocket under the contracts H2020-643529, H2020-644690.

References

1. Du, D.-Z., Smith, J.M., Rubinstein, J.H. (eds.): Advances in Steiner Trees. Kluwer Academic Publishers, Dordrecht (2000)
2. Boldi, P., Vigna, S.: Axioms for centrality. Internet Math. **10**(3–4), 222–262 (2014)
3. Donaway, R.L., Drummey, K.W., Mather, L.A.: A comparison of rankings produced by summarization evaluation measures. In: NAACL-ANLP Workshop, pp. 69–78 (2000)
4. Dreyfus, S.E., Wagner, R.A.: The steiner problem in graphs. Networks **1**(3), 195–207 (1971)

5. Dudáš, M., Svátek, V., Mynarz, J.: Dataset summary visualization with LODSight. In: Gandon, F., Guéret, C., Villata, S., Breslin, J., Faron-Zucker, C., Zimmermann, A. (eds.) ESWC 2015. LNCS, vol. 9341, pp. 36–40. Springer, Cham (2015). doi:10.1007/978-3-319-25639-9_7
6. Hakimi, S.L.: Steiner's problem in graphs and its implications. Networks **1**(2), 113–133 (1971)
7. Jiang, X., Zhang, X., Gao, F., Pu, C., Wang, P.: Graph compression strategies for instance-focused semantic mining. In: Qi, G., Tang, J., Du, J., Pan, J.Z., Yu, Y. (eds.) CSWS 2013. CCIS, vol. 406, pp. 50–61. Springer, Heidelberg (2013). doi:10.1007/978-3-642-54025-7_5
8. Karp, R.M.: Reducibility among combinatorial problems. In: Jünger, M., Liebling, T.M., Naddef, D., Nemhauser, G.L., Pulleyblank, W.R., Reinelt, G., Rinaldi, G., Wolsey, L.A. (eds.) 50 Years of Integer Programming 1958–2008 - From the Early Years to the State-of-the-Art, pp. 219–241. Springer, Heidelberg (2010)
9. Khatchadourian, S., Consens, M.P.: Explod: summary-based exploration of interlinking and RDF usage in the linked open data cloud. In: ESWC, pp. 272–287 (2010)
10. Khatchadourian, S., Consens, M.P.: Exploring RDF usage and interlinking in the linked open data cloud using explod. In: LDOW (2010)
11. Kondylakis, H., Plexousakis, D.: Ontology evolution: assisting query migration. In: Atzeni, P., Cheung, D., Ram, S. (eds.) ER 2012. LNCS, vol. 7532, pp. 331–344. Springer, Heidelberg (2012). doi:10.1007/978-3-642-34002-4_26
12. Levin, A.Y.: Algorithm for the shortest connection of a group of graph vertices. Sov. Math. Dokl. **12**, 1477–1481 (1971)
13. Navlakha, S., Rastogi, R., Shrivastava, N.: Graph summarization with bounded error. In: ACM SIGMOD, pp. 419–432. ACM (2008)
14. Palmonari, M., Rula, A., Porrini, R., Maurino, A., Spahiu, B., Ferme, V.: ABSTAT: linked data summaries with ABstraction and STATistics. In: Gandon, F., Guéret, C., Villata, S., Breslin, J., Faron-Zucker, C., Zimmermann, A. (eds.) ESWC 2015. LNCS, vol. 9341, pp. 128–132. Springer, Cham (2015). doi:10.1007/978-3-319-25639-9_25
15. Pires, C.E., Sousa, P., Kedad, Z., Salgado, A.C.: Summarizing ontology-based schemas in pdms. In: ICDEW, pp. 239–244 (2010)
16. Plesnik, J.: Worst-case relative performances of heuristics for the steiner problem in graphs (1991)
17. Queiroz-Sousa, P.O., Salgado, A.C., Pires, C.E.: A method for building personalized ontology summaries. J. Inf. Data Manage. **4**(3), 236 (2013)
18. Rayward-Smith, V.J., Clare, A.: On finding steiner vertices. Networks **16**(3), 283–294 (1986)
19. Rochat, Y.: Closeness centrality extended to unconnected graphs: the harmonic centrality index. In: Applications of Social Network Analysis (ASNA) (2009)
20. Peroni, S., Motta, E., d'Aquin, M.: Identifying key concepts in an ontology, through the integration of cognitive principles with statistical and topological measures. In: Domingue, J., Anutariya, C. (eds.) ASWC 2008. LNCS, vol. 5367, pp. 242–256. Springer, Heidelberg (2008). doi:10.1007/978-3-540-89704-0_17
21. Spearman, C.: The proof and measurement of association between two things. Am. J. Psychol. **15**(1), 72–101 (1904)
22. Tian, Y., Hankins, R.A., Patel, J.M.: Efficient aggregation for graph summarization. In: ACM SIGMOD, pp. 567–580. ACM (2008)

23. Troullinou, G., Kondylakis, H., Daskalaki, E., Plexousakis, D.: RDF digest: efficient summarization of RDF/S KBs. In: Gandon, F., Sabou, M., Sack, H., d'Amato, C., Cudré-Mauroux, P., Zimmermann, A. (eds.) ESWC 2015. LNCS, vol. 9088, pp. 119–134. Springer, Cham (2015). doi:10.1007/978-3-319-18818-8_8
24. Troullinou, G., Kondylakis, H., Daskalaki, E., Plexousakis, D.: RDF digest: ontology exploration using summaries. In: ISWC (2015)
25. Troullinou, G., Kondylakis, H., Daskalaki, E., Plexousakis, D.: Ontology understanding without tears: the summarization approach. Semant. Web J. (2017). IOS press
26. Valente, T.W., Foreman, R.K.: Integration and radiality: measuring the extent of an individual's connectedness and reachability in a network. Soc. Netw. **20**(1), 89–105 (1998)
27. Voß, S.: Steiner's problem in graphs: heuristic methods. Discrete Appl. Math. **40**(1), 45–72 (1992)
28. Wu, G., Li, J., Feng, L., Wang, K.: Identifying potentially important concepts and relations in an ontology. In: Sheth, A., Staab, S., Dean, M., Paolucci, M., Maynard, D., Finin, T., Thirunarayan, K. (eds.) ISWC 2008. LNCS, vol. 5318, pp. 33–49. Springer, Heidelberg (2008). doi:10.1007/978-3-540-88564-1_3
29. Zhang, X., Cheng, G., Qu, Y.: Ontology summarization based on RDF sentence graph. In: WWW, pp. 707–716 (2007)