Incremental structural summarization of RDF graphs

François Goasdoué, Pawel Guzewicz, Ioana Manolescu

Incremental structural summarization of RDF graphs

François Goasdoué
Univ Rennes, Inria, CNRS, IRISA
France
fg@irisa.fr

Paweł Guzewicz
École polytechnique and Inria
France
pawel.guzewicz@inria.fr

Ioana Manolescu
Inria and École polytechnique
France
ioana.manolescu@inria.fr

ABSTRACT

Realizing the full potential of Linked Open Data sharing and reuse is currently limited by the difficulty users have when trying to understand the data modeled within an RDF graph, in order to determine whether or not it may be useful for their need.

We demonstrate our **RDFQuotient** tool, which builds compact summaries of heterogeneous RDF graphs for the purpose of first-sight visualizations. An RDFQuotient summary provides an overview of the complete structure of an RDF graph, while being typically many orders of magnitude smaller, thus can be easily grasped by new users. Our summarization algorithms are time linear in the size of the input graph and incremental: they incrementally update a summary upon addition of new data.

For the demo, we plan to show the visualizations of our summaries obtained from well-known synthetic and real data sets. Further, attendees will be able to add data to the summarized RDF graphs and visually witness the incurred changes.

1 INTRODUCTION

Semantic Web graphs are nowadays being published and shared at a massive scale, e.g., Linked Open Data (LOD) Cloud (https://lod-cloud.net) lists 1.200 graphs, while the LOD Atlas portal (http://lodatlas.lri.fr) references more than 20.000 graphs. Some of these graphs are *domain-oriented*, that is, they reflect a certain application domain, e.g., education, medical etc. On the contrary, a few RDF graphs are *encyclopedic*, e.g., DBpedia (https://wiki.dbpedia.org) and YAGO [7], covering many different topics; often, these are unions of many domain-specific ones, e.g., DBpedia is available for download as a set of domain-oriented “datasets”. An overwhelming majority of the RDF graphs found in portals such as LOD Cloud or LODAtlas, https://data.gov.uk, https://data.gov etc. are domain-oriented.

Currently, a large obstacle toward exploiting this wealth of data is the difficulty for human users to make sense of a newly encountered RDF graph. The motivation for our work is to help users learn *at first sight*, without any prior knowledge about the graph and without having to set any parameter, the *(ideally complete) structure of a domain-specific RDF graph.* Given that RDF graphs can be very large, while the human information absorption capacity is relatively limited, *RDF graph summaries* have been used as intermediaries: from a given graph G a summary is extracted, then the summary is shown to the users in order to convey information about the structure and/or content of G.

We demonstrate **RDFQuotient**, a tool for constructing a complete summary of the structure of an RDF graph which does not require any user input. The particular advantage of RDFQuotient is its *tolerance to heterogeneity*, which enables it to build *compact, easy-to-visualize summaries* even from very large graphs, while preserving many of the important structural features of the graph. RDFQuotient summaries can be built efficiently, in linear time in the size of the graph. Further, they can be *incrementally updated*: upon addition of a triple t to a graph G, the summary of $G \cup \{t\}$ can be efficiently computed out of the current summary of G and t, without re-traversing the G triples.

Motivating example Figure 1 illustrates a possible visualization of an RDFQuotient summary of a BSBM [1] benchmark graph of $\mathbf{10^8}$ triples. This visualization reflects the *complete* structure of the graph, using only **5 nodes and 11 edges**, comparable to a simple small Entity-Relationship diagram. This summary reads as follows: (*i*) Non-leaf graph nodes belong to one of five disjoint classes, each represented by a summary node (boxes labeled $N1$ to $N5$ in Figure 1). The number of graph nodes in each class appears in parenthesis after the label Ni of their representative; (*ii*) Graph nodes from a class may have types. Each such type appears under the summary node label, together with its number of occurrences among graph nodes of that class, e.g., 5919 nodes represented by $N3$ are of type Producer, while 3050 are of type Vendor; (*iii*) Graph nodes from a class may have outgoing properties whose values are *leaf* nodes in the graph; the set of all such properties appears in the corresponding summary node, one property per line. For each property, e.g. country for $N3$, the summary node specifies how many graph nodes represented by this summary node have it (8969 in this case), and how many distinct leaf nodes are target of these edges (10 in this case); (*iv*) Graph nodes from a class may have outgoing properties whose values are *non-leaf* nodes in the graph. For each graph edge $n_1 \xrightarrow{a} n_2$, where n_1, n_2 are non-leaf graph nodes and a is the property (edge label), an a-labeled edge in the summary goes from the representative of n_1 to that of n_2. Next to a, this summary edge is also labeled with the number of graph edges to which it corresponds; (v) Properties from a small, fixed vocabulary are considered *metadata* (as opposed to *data*) and therefore they are not used to group nodes in classes, e.g., *rdf-schema#-comment* and *rdf-schema#-label* in Figure 1. More such visualizations can be found online[1]; below, we also work out an example leading from an RDF graph to its summary and then such a visualization.

We propose to *demonstrate the incremental construction of four* related (but different) *summaries*, i.e., show how summaries quickly adjust when triples are added to the summarized RDF graphs. Our summaries can be built from graphs where none, some or all nodes have one or more types; this is important because in many synthetic and real-life RDF graphs we studied, a large share of nodes is untyped [3]. Two of our summaries give preeminence to types (when available) to build the summary; nodes are first grouped by types and then by the relationships to other nodes. By contrast, the two other (including the one in Figure 1) give preeminence to node relationships; nodes are first grouped according to their relationships with others, then, each group is typed with the types of the graph nodes it represents within the summary (this is how each type has been attached to a summary node in our figure). In the total absence of types, each

© 2019 Copyright held by the owner/author(s). Published in Proceedings of the 22nd International Conference on Extending Database Technology (EDBT), March 26-29, 2019, ISBN 978-3-89318-081-3 on OpenProceedings.org.

[1] https://team.inria.fr/cedar/projects/rdfquotient/

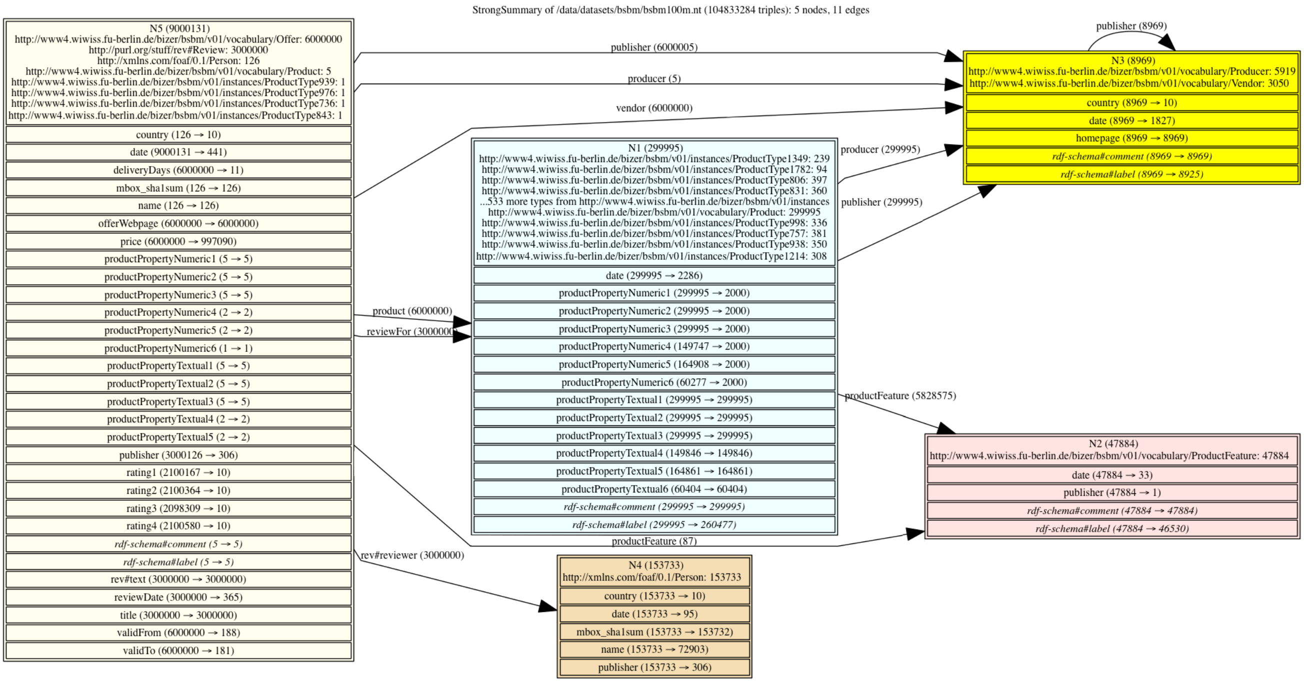

StrongSummary of /data/datasets/bsbm/bsbm100m.nt (104833284 triples): 5 nodes, 11 edges
N5 (9000131)
http://www4.wiwiss.fu-berlin.de/bizer/bsbm/v01/vocabulary/Offer: 6000000
http://purl.org/stuff/rev#Review: 3000000
http://xmlns.com/foaf/0.1/Person: 126
http://www4.wiwiss.fu-berlin.de/bizer/bsbm/v01/vocabulary/Product: 5
http://www4.wiwiss.fu-berlin.de/bizer/bsbm/v01/instances/ProductType939: 1
http://www4.wiwiss.fu-berlin.de/bizer/bsbm/v01/instances/ProductType976: 1
http://www4.wiwiss.fu-berlin.de/bizer/bsbm/v01/instances/ProductType736: 1
http://www4.wiwiss.fu-berlin.de/bizer/bsbm/v01/instances/ProductType843: 1
country (126 → 10)
date (9000131 → 441)
deliveryDays (6000000 → 11)
mbox_sha1sum (126 → 126)
name (126 → 126)
offerWebpage (6000000 → 6000000)
price (6000000 → 997090)
productPropertyNumeric1 (5 → 5)
productPropertyNumeric2 (5 → 5)
productPropertyNumeric3 (5 → 5)
productPropertyNumeric4 (2 → 2)
productPropertyNumeric5 (2 → 2)
productPropertyNumeric6 (1 → 1)
productPropertyTextual1 (5 → 5)
productPropertyTextual2 (5 → 5)
productPropertyTextual3 (5 → 5)
productPropertyTextual4 (2 → 2)
productPropertyTextual5 (2 → 2)
publisher (3000126 → 306)
rating1 (2100167 → 10)
rating2 (2100364 → 10)
rating3 (2098309 → 10)
rating4 (2100580 → 10)
rdf-schema#comment (5 → 5)
rdf-schema#label (5 → 5)
rev#text (3000000 → 3000000)
reviewDate (3000000 → 365)
title (3000000 → 3000000)
validFrom (6000000 → 188)
validTo (6000000 → 181)
publisher (6000005)
producer (5)
vendor (6000000)
product (6000000)
reviewFor (3000000)
rev#reviewer (3000000)
productFeature (87)
N1 (299995)
http://www4.wiwiss.fu-berlin.de/bizer/bsbm/v01/instances/ProductType1349: 239
http://www4.wiwiss.fu-berlin.de/bizer/bsbm/v01/instances/ProductType1782: 94
http://www4.wiwiss.fu-berlin.de/bizer/bsbm/v01/instances/ProductType806: 397
http://www4.wiwiss.fu-berlin.de/bizer/bsbm/v01/instances/ProductType831: 360
...533 more types from http://www4.wiwiss.fu-berlin.de/bizer/bsbm/v01/instances
http://www4.wiwiss.fu-berlin.de/bizer/bsbm/v01/vocabulary/Product: 299995
http://www4.wiwiss.fu-berlin.de/bizer/bsbm/v01/instances/ProductType998: 336
http://www4.wiwiss.fu-berlin.de/bizer/bsbm/v01/instances/ProductType757: 381
http://www4.wiwiss.fu-berlin.de/bizer/bsbm/v01/instances/ProductType938: 350
http://www4.wiwiss.fu-berlin.de/bizer/bsbm/v01/instances/ProductType1214: 308
date (299995 → 2286)
productPropertyNumeric1 (299995 → 2000)
productPropertyNumeric2 (299995 → 2000)
productPropertyNumeric3 (299995 → 2000)
productPropertyNumeric4 (149747 → 2000)
productPropertyNumeric5 (164908 → 2000)
productPropertyNumeric6 (60277 → 2000)
productPropertyTextual1 (299995 → 299995)
productPropertyTextual2 (299995 → 299995)
productPropertyTextual3 (299995 → 299995)
productPropertyTextual4 (149846 → 149846)
productPropertyTextual5 (164861 → 164861)
productPropertyTextual6 (60404 → 60404)
rdf-schema#comment (299995 → 299995)
rdf-schema#label (299995 → 260477)
producer (299995)
publisher (299995)
productFeature (5828575)
publisher (8969)
N3 (8969)
http://www4.wiwiss.fu-berlin.de/bizer/bsbm/v01/vocabulary/Producer: 5919
http://www4.wiwiss.fu-berlin.de/bizer/bsbm/v01/vocabulary/Vendor: 3050
country (8969 → 10)
date (8969 → 1827)
homepage (8969 → 8969)
rdf-schema#comment (8969 → 8969)
rdf-schema#label (8969 → 8925)
N2 (47884)
http://www4.wiwiss.fu-berlin.de/bizer/bsbm/v01/vocabulary/ProductFeature: 47884
date (47884 → 33)
publisher (47884 → 1)
rdf-schema#comment (47884 → 47884)
rdf-schema#label (47884 → 46530)
N4 (153733)
http://xmlns.com/foaf/0.1/Person: 153733
country (153733 → 10)
date (153733 → 95)
mbox_sha1sum (153733 → 153732)
name (153733 → 72903)
publisher (153733 → 306)

Figure 1: Visualization built from an RDFQuotient summary.

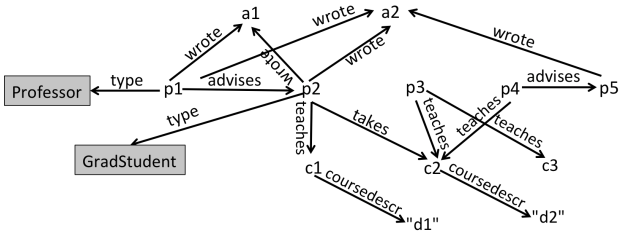


Figure 2: Sample RDF graph.

type-first summary coincides with a type-ignorant one (thus, our four summaries collapse into two).

Below, we define our summaries (Section 2) and summarization algorithms (Section 3). Then, we present the demonstration scenario based on summary visualizations (Section 5). Finally, we compare them with related work and we conclude (Section 6).

2 RDFQUOTIENT SUMMARIES

Let U be a set of URIs, L be a set of literals and B be a set of blank nodes as per the RDF specification. An RDF graph $\mathtt{G}$ is a set of *triples* of the form (s, p, o) where $s \in U \cup B \cup L$, $p \in U$ and $o \in U \cup B \cup L$. The special URI *type*, part of the RDF standard, is used to attach types to nodes. An RDF graph may contain *ontology (schema)* triples; while there are interesting interactions between summarization and ontologies [3], below we only focus on summarizing the non-schema triples, which make up the vast majority of all RDF graphs we encountered. Thus, we consider $\mathtt{G}$ consists exclusively of *type* triples and/or *data* triples (all those whose property is not *type*; we call these *data properties*).

An *RDF equivalence relation* denoted $\equiv$ is a binary relation over the nodes of an RDF graph that is reflexive, symmetric and transitive. Given an equivalence relation $\equiv$, an *RDF graph quotient* is an RDF graph having (i) one node for each equivalence class of nodes; (ii) for each edge $n_1 \xrightarrow{a} n_2$, a summary edge $n_1^{\equiv} \xrightarrow{a} n_2^{\equiv}$, where $n_i^{\equiv}$, $i \in \{1, 2\}$, is the summary node corresponding to the equivalence class of n_i, also called *representative of* n_i. The literature comprises many quotient graph summaries (see Section 6), which differ by their equivalence relations.

The equivalence relations we use are based on the concept of *property cliques*, which encodes a *transitive* relation of *edge co-occurrence on graph nodes.* Given an RDF graph $\mathtt{G}$, two data properties p_1, p_2 are in the same **source clique** iff: (i) there exists a $\mathtt{G}$ node n which is the source of p_1 and p_2 (i.e., $(n, p_1, x) \in \mathtt{G}$ and $(n, p_2, y) \in \mathtt{G}$ for some x and y), or (ii) there exists a data property p_3 such that p_3 is in the same source clique as p_1, and p_3 is in the same source clique as p_2. Symetrically, p_1 and p_2 are in the same **target clique** if there exists a G node which is the target of p_1 and p_2, or a data property p_3 which is in the same target clique as p_1 and p_2. In Figure 2, the properties *advises* and *teaches* are in the same source clique due to p_4. The same holds for *advises* and *wrote* due to p_1; consequently, *advises* and *wrote* are also in the same source clique. Further, the graduate student p_2 teaches a course and takes another, thus *teaches*, *advises*, *wrote* and *takes* are all part of the same source clique. In this example, p_1, p_2, p_3, p_4, p_5 have the source clique $SC_1 = \{advises, takes, teaches, wrote\}$, c_1, c_2, c_3 have the source clique $SC_2 = \{coursedescr\}$ and a_1, a_2 have the empty source clique $SC_3 = \emptyset$. Similarly, the target cliques are, respectively; $TC_1 = \{advises\}$ for p_2, p_5, $TC_2 = \{teaches, takes\}$ for c_1, c_2, c_3, $TC_3 = \{coursedescr\}$ for d_1, d_2, $TC_4 = \{wrote\}$ for a_1, a_2 and $TC_5 = \emptyset$ for p_1, p_3, p_4.

It is easy to see that the set of non-empty source (or target) cliques is a *partition over the data properties of an RDF graph* $\mathtt{G}$. Further, if a $\mathtt{G}$ node n is source of some data properties, they are all in the same source clique; similarly, all the properties of which n is a target are in the same target clique. Based on these cliques, for any nodes n_1, n_2 of $\mathtt{G}$, we define:

- n_1 is **weakly equivalent** to n_2, denoted $n_1 \equiv_{\mathtt{W}} n_2$, iff n_1, n_2 have the same source clique *or* the same target clique;

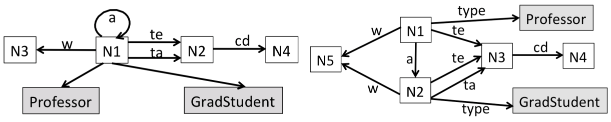


Figure 3: Weak (left) and strong (right) graph summary.

- n_1 is **strongly equivalent** to n_2, denoted $n_1 \equiv_{\texttt{S}} n_2$, iff n_1, n_2 have the same source clique *and* the same target clique.

Further, we decide that in any RDF equivalence relation, *any class node*, i.e., a URI c appearing in a triple of the form $(n, type, c)$, is (*i*) only equivalent to itself and (*ii*) represented by itself in any RDFQuotient summary. This ensures that RDF types (classes), which (when present) denote an important information that data producers added to help understand their RDF graphs, are preserved in the summary.

The equivalence relations $\equiv_{\texttt{W}}$ and $\equiv_{\texttt{S}}$ lead to the **weak**, respectively **strong** summaries, defined as *quotients of* $\texttt{G}$ *through* $\equiv_{\texttt{W}}$, denoted $\texttt{G}_{/\equiv_{\texttt{W}}}$, respectively, *through* $\equiv_{\texttt{S}}$, denoted $\texttt{G}_{/\equiv_{\texttt{S}}}$. Figure 3 illustrates these on the sample graph in Figure 2. For brevity, in Figure 3 and from now on, we use *a*, *w*, *te*, *ta*, *cd* to denote respectively the properties *advises*, *writes*, *teaches*, *takes*, and *coursedescr*.

In $\texttt{G}_{/\equiv_{\texttt{W}}}$, N_1 represents all the people (p_1 to p_5), N_2 represents the courses, N_3 the articles and N_4 the course descriptions. Note the self-loop from N_1 to itself; it denotes that *some* nodes represented by N_1 advise *some* nodes represented by N_1. This summary has only 4 nodes and 5 edges. It conveys the essential information that some nodes advise, write, also they teach and take something that has course descriptions. The Professor and GradStudent types of nodes p_1, respectively p_2, are attached to their common representative N_1.

$\texttt{G}_{/\equiv_{\texttt{S}}}$ differs from $\texttt{G}_{/\equiv_{\texttt{W}}}$ by representing the person nodes in two separate groups: those represented by N_1 advise those represented by N_2. This is because the target clique of p_1, p_3 and p_4 is empty, while the target clique of p_2 and p_5 is $\{advises\}$. This example illustrates the fact (visible from the summary definitions) that $\texttt{G}_{/\equiv_{\texttt{S}}}$ summarizes *at finer granularity* than $\texttt{G}_{/\equiv_{\texttt{W}}}$ (or, equivalently, $\equiv_{\texttt{S}}$ entails $\equiv_{\texttt{W}}$, but the opposite does not hold).

Clique-based structural summarization leads to **compact summaries** even in graphs with **heterogeneous structure**. This is because of the *transitive* aspect of the property cliques. For example, p_1 and p_3 have the same source clique, even though their property sets are disjoint: $\{a, w\}$ for p_1, $\{te\}$ for p_3; they are in the same source clique e.g. due to p_4, which has both a and te. In contrast, previously studied quotient summaries, in particular those aimed for indexing and query optimization, would not accept p_1 and p_3 as equivalent; in general, such summaries lead to more equivalence classes (summary nodes), thus also summary edges, making summaries hard to understand visually.

Type-first summarization The weak and strong summary group nodes according to their incoming/outgoing data triples and then just "carry" their types to the summary. A different choice is to group nodes first by their *set of types* (if any)[2], and use the data triples to group the nodes without types. We define:

- n_1 is **typed-weak equivalent** to n_2, noted $n_1 \equiv_{\texttt{TW}} n_2$, iff (*i*) n_1, n_2 have the same non-empty set of types *or* (*ii*) both n_1, n_2 are untyped, and $n_1 \equiv_{\texttt{W}} n_2$;

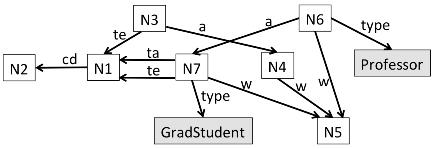


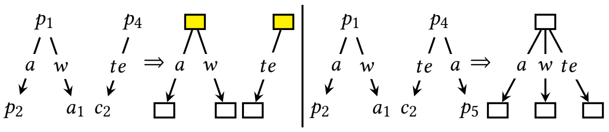
Figure 4: Typed weak graph summary.

- n_1 is **typed-strong equivalent** to n_2, noted $n_1 \equiv_{\texttt{TS}} n_2$, iff (*i*) n_1, n_2 have the same non-empty set of types *or* (*ii*) both n_1, n_2 are untyped, and $n_1 \equiv_{\texttt{S}} n_2$.

These relations lead to the **typed weak** ($\texttt{G}_{/\equiv_{\texttt{TW}}}$), respectively, to the **typed strong** ($\texttt{G}_{/\equiv_{\texttt{TS}}}$) RDFQuotient summaries. Figure 4 illustrates typed weak summarization on our sample graph; on this simple example, $\texttt{G}_{/\equiv_{\texttt{TS}}}$ is identical (in general, it may differ).

3 BUILDING RDFQUOTIENT SUMMARIES

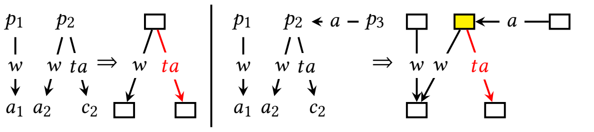
We have devised algorithms which build $\texttt{G}_{/\equiv_{\texttt{W}}}$, $\texttt{G}_{/\equiv_{\texttt{S}}}$, $\texttt{G}_{/\equiv_{\texttt{TW}}}$ and $\texttt{G}_{/\equiv_{\texttt{TS}}}$ through a single traversal of an RDF graph $\texttt{G}$, in two flavors: (*i*) *global*: traverse $\texttt{G}$, compute all the cliques, then traverse it again and represent nodes according to their cliques and/or types; (*ii*) *incremental*: in a single traversal of $\texttt{G}$, *gradually build each source and target clique* based on the triples traversed *up to that point* and *simultaneously* represent $\texttt{G}$ nodes in a summary that is continuously updated; after traversing the last triple of $\texttt{G}$, each incremental summarization algorithm ends up with the respective summary of the full $\texttt{G}$. The algorithms are detailed and their correctness is proved in [3]; below, we illustrate their interesting points on minimal examples.



First, let us see on an example how $\equiv_{\texttt{W}}$ can grow during *incremental weak summarization*. Suppose the graph $\texttt{G}$ in Figure 2 is traversed and summarized starting with: (p_1 *advises* p_2), then (p_1 *wrote* a_1), then (p_4 *teaches* c_2) (see the figure above). When we summarize this third triple, we do not know yet that p_1 is equivalent to p_4, because no common source of *teaches* and *advises* (e.g., p_3 or p_4) has been seen so far. Thus, p_4 is so far not equivalent to any other node, and represented separately from p_1. Now, assume the fourth triple traversed is (p_4 *advises* p_5): at this point, we know that *advises*, *wrote* and *teaches* are in the same source clique, thus $p_1 \equiv_{\texttt{W}} p_4$, and their representatives (highlighted in yellow) must be **fused** in the summary. More generally, it can be shown that $\equiv_{\texttt{W}}$ **only grows** as more triples are visited (i.e., is monotonic), in other words: if in a subset $\texttt{G}'$ of $\texttt{G}$'s triples, two nodes n_1, n_2 are weakly equivalent, then this holds in any $\texttt{G}''$ with $\texttt{G}' \subseteq \texttt{G}'' \subseteq \texttt{G}$.

Incremental strong summarization is even more complex because unlike $\equiv_{\texttt{W}}$, $\equiv_{\texttt{S}}$ **may grow and shrink** during summarization (i.e., is non-monotonic). For instance, assume the summarization of the graph in Figure 2 starts with (p_1 *wrote* a_1), (p_2 *wrote* a_2), (p_2 *takes* c_2) (see the figure below). After these, we know $p_1 \equiv_{\texttt{S}} p_2$; their source clique is $\{wrote, takes\}$ and their target clique is $\emptyset$. Assume the next triple traversed is (p_3 *advises* p_2): at this point, p_1 is *not* $\equiv_{\texttt{S}}$ *to* p_2 *any more*, because p_2's target clique is now $\{advises\}$ instead of $\emptyset$. Thus, p_2 **splits** from p_1, that is, it needs to be represented by a new summary node (shown in yellow below), distinct from the representative of p_1.

[2] We use *set of types* and not just "type" on purpose, because an RDF node may have more than one type. If we classified a node according to *each* of its types, as in e.g. [2], a node with many types would have more than one representative, which is incompatible with quotient summarization.



Further, note that the representative of p_1 and p_2 (at left above) had one *takes* edge (highlighted in red) which was *solely due to p_2's outgoing takes edge.* By definition of a quotient summary, that edge *moves* from the old to the new representative of p_2 (the yellow node). If, above at left, p_1 had also had an outgoing edge labeled *takes*, at right, both nodes in the top row would have had an outgoing *takes* edge. It can be shown that *splits only occur in such cases*, i.e., a node whose target clique becomes non-empty (respectively whose source clique becomes non-empty) and the node was previously represented together with other nodes; if it was represented alone, the respective clique of its representative is just updated.

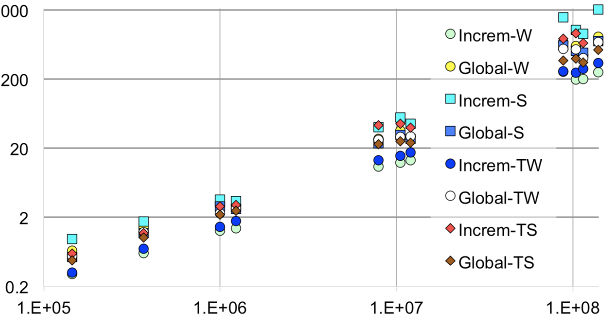


Figure 5: Summarization time (s) vs. graph sizes |G|.

The **amortized complexity** of our summarization algorithms is linear in the number of triples of G. Figure 5 illustrate this empirically on a variety of benchmark (LUBM and BSBM) and real-life (DBLP, Springer conference etc.) datasets ranging from a few hundred thousands to more than 100 million triples; note that both axes are in log-scale. The implementation is made in Java 1.8; RDF graphs are stored in Postgres 9.6 and traversed from there. Increm-W is the fastest overall; it traverses G only once, thus it is faster than global-W which performs one extra pass to compute the cliques. S, TW and TS, in this order, are more expensive, and finally incremental S, which pays an extra performance overhead for growing and shrinking the equivalence relation.

4 FROM SUMMARIES TO VIZUALIZATIONS

The core of our work is on defining and efficiently building summaries; here we present one possible way of rendering them through a vizualization like the one illustrated in Figure 1.

On our four summaries we apply **leaf and type inlining**, as follows. We remove type edges; instead, each type attached to a node in the summary is shown in the box corresponding to the node, after the node ID. Similarly, for each edge $n \xrightarrow{a} m$ where m is a leaf, we include a as an "attribute" of n, and do not render m (we say it has been "inlined" within n). A sizable part of an RDF graph's nodes are leaves; as we will show, inlining them into their parent nodes greatly simplifies the visualization.

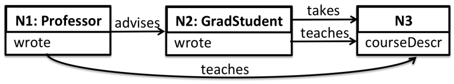


Figure 6: Leaf and type inlining on the sample strong summary from Figure 3 (right).

Figure 6 illustrates inlining for the S summary of our sample graph. This summary is extremely compact, yet rich with information; professors, students, and courses are visible at a glance. Articles have been inlined within their authors as they were leaves in $\mathtt{G}_{/\equiv_{\mathtt{TS}}}$ (Figure 3). This simplification can also be seen as a small loss of information: Figure 6 does not immediately suggest that Professors may have written articles together with GradStudents. However, (*i*) only leaf nodes are folded and (*ii*) after a first glance, users may pursue exploration by other means (e.g., queries to check for such joint articles).

5 DEMONSTRATION SCENARIO

Demonstration attendees will be able to pick an RDF graph from a list of well-known synthetic and real data sets, visualize their summaries, and compare with other close competitor summaries, such as those mentioned in Section 6; some of these vizualizations can be seen online (https://team.inria.fr/cedar/projects/rdfquotient/). Attendees will also be able to add new triples to an RDF graph, to figure out through our summary visualization how changes in the original data are rapidly reflected into the summary thanks to incremental summarization. To make it entertaining, we plan to use RDF data on the conference attendees, from DBLP, other public sources, and made-up triples to get interesting summary changes.

6 RELATED WORK & CONCLUSION

The literature comprises many RDF summarization techniques, more than a hundred of which we covered in a recent co-authored survey [4]. RDFQuotient summarizes *the structure of the data triples*, which form a vast majority of RDF graphs; complementary proposals summarize the ontology, the values, find the most frequent property groups etc. Closest to us are quotient summaries which group nodes by *the set of their outgoing data properties* ("characteristic sets" [6]) and possibly also by *by the set of their incoming data properties* (forward and backward bisimulation [5]). Our clique-based summarization differs from these by the *transitive* aspect of the cliques which leads to *heterogeneity-tolerant summarization.* Indeed, as we plan to show during our demonstrations, more strict summaries such as [5, 6] support query optimization and indexing well, but have too many nodes and edges, even after inlining, for a comfortable vizualization. In exchange, our summaries are not generally appropriate for indexing, as they do not give e.g. access to "all the resources having properties a and b": the graph nodes whose source clique is $\{a, b\}$ may have one or another or both.

REFERENCES

[1] Christian Bizer and Andreas Schultz. 2009. The Berlin SPARQL Benchmark. *Int. J. Semantic Web Inf. Syst.* 5, 2 (2009).

[2] Stéphane Campinas, Renaud Delbru, and Giovanni Tummarello. 2013. Efficiency and precision trade-offs in graph summary algorithms. In *IDEAS*.

[3] Šejla Čebirić, François Goasdoué, Paweł Guzewicz, and Ioana Manolescu. 2017. *Compact Summaries of Rich Heterogeneous Graphs.* Research Report RR-8920. INRIA Saclay ; Université Rennes 1. https://hal.inria.fr/hal-01325900

[4] Sejla Cebiric, François Goasdoué, Haridimos Kondylakis, Dimitris Kotzinos, Ioana Manolescu, Georgia Troullinou, and Mussab Zneika. 2018. Summarizing Semantic Graphs: A Survey. *The VLDB Journal* (2018). https://hal.inria.fr/hal-01925496

[5] Raghav Kaushik, Philip Bohannon, Jeffrey F. Naughton, and Henry F. Korth. 2002. Covering indexes for branching path queries. In *SIGMOD*.

[6] Thomas Neumann and Guido Moerkotte. 2011. Characteristic sets: Accurate cardinality estimation for RDF queries with multiple joins. In *ICDE*.

[7] Thomas Rebele, Fabian M. Suchanek, Johannes Hoffart, Joanna Biega, Erdal Kuzey, and Gerhard Weikum. 2016. YAGO: A Multilingual Knowledge Base from Wikipedia, Wordnet, and Geonames. In *ISWC*.