## **Rewriting Queries over Summaries of Big Data Graphs**

Mariano P. Consens<sup>1</sup>, Valeria Fionda<sup>2</sup>, Shahan Khatchadourian<sup>1</sup>, Giuseppe Pirrò<sup>3</sup>

<sup>1</sup> University of Toronto, Canada
<sup>2</sup> Department of Mathematics, University of Calabria, Italy
<sup>3</sup> KRDB, Free University of Bozen-Bolzano, Italy

This short paper reports on the benefits that traversal queries over existing graph stores (such as RDF databases) can gain from a class of optimizations based on summaries. Summaries, also known as structural indexes, have been extensively covered in the literature (see [2] for a brief overview). Despite this, summary-based optimizations are not widely implemented.

To make both graph traversal queries and summaries readily available in existing RDF stores, we have devised a translation that outputs SPARQL queries that execute over summaries directly represented in RDF. In what follows, we give an overview of our proposal, illustrate it with an example, and mention preliminary evaluation results on real-world data.

**Overview.** Our approach to summary-based optimization of graph traversal queries on top of SPARQL processors consists of three components.

First, we have developed and implemented an algorithm that translates graph traversal queries (see [1, 6] for recent surveys) into SPARQL expressions. The graph traversal language we selected extends the expressiveness of Nested Regular Expressions [5] without compromising the language's low combined data and query complexity.

Second, we construct and store summaries alongside the original graph in the RDF store. The summaries we focus on this work are obtained by dual bisimulation contractions [3, 4].

Third, we build upon our translation algorithm to develop a rewriting optimization that converts graph traversal queries into equivalent SPARQL queries that execute over RDF graphs of both the original data and the summary.

Motivating example. We illustrate our approach to summary-based query rewriting using a fragment of the linkedmdb.org open movie dataset shown in Fig. 1(a). Consider the query that asks for actors that have acted in two movies directed by the same director such that one movie does have information about the country in which it has been shot while the other does not. Evaluating this query in the fragment shown has the actor labelled Joe Turkel as the only answer, since he acted in Paths of Glory (with no country information) as well as in The Shining (shot in GB), and both movies were directed by Stanley Kubrick (Jack Nicholson acts in only one movie in this input graph). This query can be expressed as a graph traversal query starting at the director class node and ending at matching actor nodes.

Fig 1 (b) shows a dual bisimulation summary materialized as an RDF graph that connects to the original data fragment in Fig 1 (a). Each block node in the summary is identified by a hash URI and represents a non-overlapping subset of the fragment's nodes using bc:hasExtent edges to each of the data graph nodes belonging to the block partition (referred to as the block's *extent*). Blocks are also connected by edges that summarize the corresponding semantic relationship between data nodes across partitions (e.g., :actor, :director, :country, etc.). Specifically, a summary edge means that each node in the source block's extent has an edge to a node in the target block's extent, and dually, each node in the target block's extent has an incoming edge from a node in the source block's extent. This specification warranties



Fig. 1. An fragment of the data graph (a) and summary graph (b) for linkedmdb.org.

that graph traversals in the summary graph will match exactly those nodes in the data graph that satisfy that same graph traversal.

A graph traversal of our motivating example can be evaluated on a summary as follows. First, the graph traversal expression is rewritten into a SPARQL query over the summary graph. Edge traversals occur between blocks instead of data graph nodes where summary predicates can be prefixed with a custom namespace. Since a summary query returns blocks, the last step of the rewriting consists of navigating from blocks in the summary to nodes in the data graph by traversing bc:hasExtent edges. Evaluating the rewritten query over the combined summary and data graph can traverse a lower number of nodes and edges than evaluating the original query on the data graph.

**Preliminary experimental evaluation.** Our initial experiments use several open datasets, including linkedmdb.org, which contains information about 17K directors, 97K movies and 60K actors. Computing a dual bisimulation contraction of linkedmdb.org produces a summary with a total of 14K director blocks, 72K movies blocks, and 50K actor blocks (the blocks with the largest extent sizes contain 0.4K directors, 3K movies and 2K actors). This summary has 63% of the nodes and 70% of the edges of the original data graph.

Graph traversal expressions used to generate SPARQL queries for our experiments are similar to our motivating example. We choose graph traversal expressions that induce neighbourhoods with a variety of path lengths, cycles, forward traversals, and backward traversals, and that return a range of zero to millions of results. Our evaluation has been performed using Virtuoso as the underlying triple store for each data graph and its summary. Executing the rewritten SPARQL queries over the combined for <code>linkedmdb.org</code>summary and data graphs can yield speed up factors over 30 times (e.g., elapsed times of tens of seconds for the summary query compared to tens of minutes for the data graph query).

## References

- 1. P. Barcelo. Querying Graph Patterns. In PODS, pages 175-188. ACM, 2013.
- M. P. Consens. Structure Indexing. In Encyclopedia of Database Systems, pages 2857–2862. Springer, 2009.
- A. Dovier, C. Piazza, and A. Policriti. An efficient algorithm for computing bisimulation equivalence. *Theor. Comput. Sci.*, 311(1-3):221–256, 2004.
- 4. R. Paige and R. E. Tarjan. Three partition refinement algorithms. SIAM Journal on Computing, 16(6):973–989, 1987.
- J. Pérez, M. Arenas, and C. Gutierrez. nSPARQL: A Navigational Language for RDF. JWS, 8(4), 2010.
- 6. P. T. Wood. Query Languages for Graph Databases. SIGMOD Rec., 41(1), 2012.