Answering Provenance-Aware Regular Path Queries on RDF Graphs Using an Automata-Based Algorithm

Xin Wang[†‡], Jun Ling[†‡], Junhu Wang[§], Kewen Wang[§], Zhiyong Feng[†‡]
[†]School of Computer Science and Technology, Tianjin University, Tianjin, China
[‡]Tianjin Key Laboratory of Cognitive Computing and Application, Tianjin, China
[§]School of Information and Communication Technology, Griffith University, Australia
{wangx, lingjun}@tju.edu.cn, {j.wang, k.wang}@griffith.edu.au, zyfeng@tju.edu.cn

ABSTRACT

This paper presents an automata-based algorithm for answering the *provenance-aware* regular path queries (RPQs) over RDF graphs on the Semantic Web. The provenance-aware RPQs can explain why pairs of nodes in the classical semantics appear in the result of an RPQ. We implement a parallel version of the automata-based algorithm using the Pregel framework Giraph to efficiently evaluate provenance-aware RPQs on large RDF graphs. The experimental results show that our algorithms are effective and efficient to answer provenance-aware RPQs on large real-world RDF graphs.

Categories and Subject Descriptors

H.2.4 [**Database Management**]: Systems

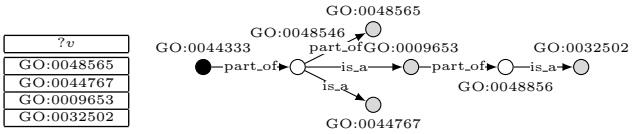
Keywords

Automata; provenance-aware; RDF; regular path queries

1. INTRODUCTION

RDF is a graph-based data model for describing things and their relationships on the Semantic Web. Regular path queries, or RPQs, are widely considered an essential querying mechanism for large RDF graphs. The classical semantics of an RPQ Q over a graph G is a set of pairs (u, v) of end points of paths in G that satisfies Q, however, such a semantics provides no clue as to why a pair of nodes satisfies Q. For example, the RPQ `?u (part_of/part_of)+|(part_of/is_a)+ ?v` asks for pairs of terms (u, v) such that u can reach v through an even number of `part_of` relations or an alternating sequence of `part_of` and `is_a` relations over the biological dataset GO. For u = `GO:0044333`, the answers (i.e., the set of v) to this RPQ are listed in the table shown in Figure 1(a), from which one cannot know how each term is reached from `GO:0044333` at all. In contrast, the graph shown in Figure 1(b) is the provenance-aware answer to the above RPQ, from which one can easily tell how the terms (the gray nodes) can be reached from `GO:0044333` (the black node) through the paths that satisfy the above RPQ.

To the best of our knowledge, Dey et al [1] have been the first to investigate the provenance-aware semantics of RPQs. Their work resorted to translating provenance-aware RPQs into Datalog or SQL implementations. However, from their experimental results on relatively small graphs of the N-queens problem, we could see that their approach can be hardly scalable for large real-world RDF graphs. In this paper, we also study the provenance-aware feature of RPQs. Compared with [1], our work makes two contributions: (1) we present an automata-based algorithm that answers the provenance-aware RPQs natively, and (2) we parallelize our automata-based algorithm using the Pregel framework Giraph [3] to efficiently evaluate provenance-aware RPQs on large-scale real-world RDF graphs.



(a) Classical (b) Provenance-aware

Figure 1: Two semantics of RPQs

2. PROBLEM AND OUR APPROACH

An RDF graph T is a set of triples, each triple (s, p, o) representing a statement of a predicate p between a subject s and an object o. We use $subj(T)$, $pred(T)$, and $obj(T)$ to denote the set of subjects, predicates and objects in T respectively. More formally, an RDF graph T can be modeled as a directed labeled graph $G = (V, E, l)$, where the node set $V = \{v \mid v \in subj(T) \cup obj(T)\}$, the edge set $E \subseteq V \times V$, and $l : E \to pred(T)$ is a function that assigns a label to each edge. For each $(s, p, o) \in T$, there exists an edge $e = (s, o) \in E$ and $l(e) = p$. A path in G from node x to y is a sequence $\rho = v_0 e_0 v_1 \cdots v_{n-1} e_{n-1} v_n$, $v_i \in V$, $v_0 = x$, $v_n = y$, and $e_i = (v_i, v_{i+1}) \in E$. The label of ρ, denoted by $\lambda(\rho)$, is the string $l(e_0)l(e_1)\cdots l(e_{n-1})$ in Σ^*, where $\Sigma = pred(T)$.

An RPQ over G is of the form $?x \xrightarrow{r} ?y$, where $?x$ and $?y$ are variables, and r is a regular expression over the alphabet $\Sigma = pred(T)$, which is defined as $r ::= p \mid r/r \mid r|r \mid r^* \mid r^+$, where $p \in \Sigma$ and $/$, $|$, *, and $^+$ are concatenation, alternation, Kleene closure, and Kleene plus respectively that have the usual meanings. $L(r)$ is the language expressed by r. The *classical* semantics of an RPQ Q over a graph G is defined as a set of pairs of nodes $[\![Q]\!]_G = \{(u, v) \mid$ there exists a path ρ in G from node u to v whose label $\lambda(\rho)$ is in $L(r)\}$. We use $\mathcal{P}_r(x, y)$ to denote the set of all paths from x to y in G whose labels are in $L(r)$. The *provenance-aware* semantics of an RPQ Q over a graph G is a set of graphs (i.e., subgraphs

WWW'14 Companion, April 7–11, 2014, Seoul, Korea.
ACM 978-1-4503-2745-9/14/04.
http://dx.doi.org/10.1145/2567948.2577284.

of G) $[\![Q]\!]_G^{prov} = \{\mathcal{G}_Q(x,y) = (\mathcal{V}_Q(x,y), \mathcal{E}_Q(x,y)) \mid (x,y) \in [\![Q]\!]_G\}$ such that $\mathcal{V}_Q(x,y) = \{v \mid v \in \rho \wedge \rho \in \mathcal{P}_r(x,y)\}$, and $\mathcal{E}_Q(x,y) = \{e \mid e \in \rho \wedge \rho \in \mathcal{P}_r(x,y)\}$.

Let $G = (V, E, l)$ be an RDF graph and $\mathcal{A} = (S, \Sigma, q_0, \delta, F)$ be the DFA converted from the regular expression r of an RPQ Q. We construct $\hat{G} = (\hat{V}, \hat{E}, l)$ from G, where $\hat{V} = V \cup \{\hat{v}\}$ (i.e., $\hat{v}$ is a newly added node to V), $\hat{E} = E \cup \{\hat{e} \mid \hat{e} = (\hat{v}, v) \wedge v \in V\}$, and $l(\hat{e}) = \varepsilon$. Also, we construct $\hat{\mathcal{A}} = (\hat{S}, \hat{\Sigma}, \hat{q_0}, \hat{\delta}, F)$ from $\mathcal{A}$, where (1) $\hat{S} = S \cup \{\hat{q_0}, q_d\}$, (2) $\hat{\Sigma} = \Sigma \cup \{\varepsilon\}$, (3) $\hat{\delta}(q,a) = \delta(q,a), \hat{\delta}(q,\varepsilon) = q_d$ for $q \in S \wedge a \in \Sigma$, and (4) $\hat{\delta}(\hat{q_0}, a) = q_0, \hat{\delta}(q_d, a) = q_d$ for $a \in \hat{\Sigma}$. The *product automaton* of $\hat{G}$ and $\hat{\mathcal{A}}$ is an NFA $\hat{G} \times \hat{\mathcal{A}} = (S', \hat{\Sigma}, q_0', \delta', F')$, where $S' \subseteq \hat{V} \times \hat{S}$, $q_0' = (\hat{v}, \hat{q_0})$, $\delta' : S' \times \hat{\Sigma} \to 2^{S'}$, and $F' = \{(v, q_f) \mid v \in \hat{V} \wedge q_f \in F\}$. For $(u, s_1) \in S'$ and $p \in \hat{\Sigma}$, we have $(v, s_2) \in \delta'((u, s_1), p)$ iff $\hat{\delta}(s_1, p) = s_2$ and $e = (u, v) \in \hat{E} \wedge l(e) = p$. Algorithm 1 shows the procedure for evaluating a provenance-aware RPQ using the above product automaton, in which the function GetResults is used to extract the answers from the product automaton recursively. We have proved that Algorithm 1 can be done in $O(|\hat{G}| \cdot |\hat{\mathcal{A}}|)$ time.

Algorithm 1 Evaluating a provenance-aware RPQ

Input: A graph G and an RPQ Q
Output: $\mathcal{G}_Q = (\mathcal{V}_Q, \mathcal{E}_Q)$ // Provenance-aware semantics

```
1: Construct Â from Q and Ĝ from G
2: Construct the product automaton Ĝ × Â
3: for each (v, q_f) ∈ F' do                    // F' of Ĝ × Â
4:     (V_t, E_t) ← GetResults(Ĝ × Â, (v, q_f))
5:     V_Q ← V_Q ∪ V_t and E_Q ← E_Q ∪ E_t
6: end for
7: function GetResults(Ĝ × Â, (v, q_f))
8:     V_t ← ∅ and E_t ← ∅
9:     if (v, s) ∈ S' is visited then           // S' of Ĝ × Â
10:        return (V_t, E_t)
11:    end if
12:    V_t ← V_t ∪ {v}
13:    for each (u, s') such that (v, s) ∈ δ'((u, s'), p) do
14:        E_t ← E_t ∪ {(u, v)}
15:        (V_t', E_t') ← GetResults(Ĝ × Â, (u, s'))
16:        V_t ← V_t ∪ V_t' and E_t ← E_t ∪ E_t'
17:    end for
18:    Mark (v, s) visited
19:    return (V_t, E_t)
20: end function
```

To efficiently evaluate provenance-aware RPQs on large-scale RDF graphs, we parallelize Algorithm 1 using the Pregel framework [2] that is more suitable for large graph processing than MapReduce according to [2]. In our parallel setting, every node of $\hat{G}$ computes independently and keeps track of its own data, which include incoming and outgoing nodes, states of $\hat{\mathcal{A}}$ in which the node currently is, and other necessary information. We observe that the construction of the product automaton and the recursive answer extraction can be well parallelized. Due to space limitations, we can only present preliminary experimental results in the next section.

3. EXPERIMENTS

Our experiments were conducted on a 4-node cluster connected by a gigabit Ethernet switch. Each node has 4 Intel Xeon CPUs E5-4607 (2.2GHz, 6cores), 64GB memory, and 64-bit Ubuntu 12.04 as the OS. We implemented Algorithm 1 in Java on one single node in the cluster and the parallel algorithm using the Giraph [3] Pregel framework on all 4 cluster nodes. We evaluated two versions of our algorithm over 3 real-world RDF graphs, i.e., GO (86MB, 160775 nodes, 701043 edges), BioGRID (225MB, 817093 nodes, 2162520 edges), and DBpedia (3.5GB, 4079463 nodes, 25910645 edges). Each graph is explored by three queries: (1) $Q1$ `(is_a)`$^+$, $Q2$ `(part_of)`$^+$, and $Q3$ `(regulates)`$^+$ on GO, (2) $Q4$ `(0407)`$^+$, $Q5$ `(0915)`$^+$, and $Q6$ `(0915/0407)`$^+$ on BioGRID, and (3) $Q7$ `(influenced)`*, $Q8$ `(successor|child)`*, and $Q9$ `(spouse/child)`* on DBpedia. Figure 2 shows the results of the above queries. We can observe that the parallel version outperforms the single-node version for all queries in general. Since the GO graph is relatively small, the performance improvement of the parallel version for GO is not so obvious as for BioGRID and DBpedia.

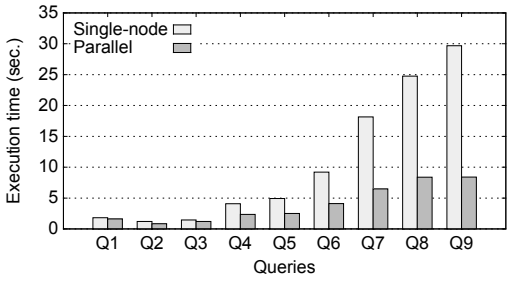


Figure 2: Experimental results

4. CONCLUSION

In this paper, we present an automata-based algorithm for answering the provenance-aware RPQs over RDF graphs, which can provide users the reason why pairs of nodes satisfy a given RPQ. We also implement a parallel version of our automata-based algorithm using the Pregel framework Giraph. The preliminary experimental results show that our algorithms can answer the provenance-aware RPQs on large-scale real-world RDF graphs effectively and efficiently. We will perform an experimental comparison between our approach and [1] in the future full version of this paper.

Acknowledgments. This work is supported by the National Natural Science Foundation of China (61100049), the National High-tech R&D Program of China (863 Program) (2013AA013204), and the Australia Research Council (ARC) Discovery grants DP130103051 and DP1093652.

5. REFERENCES

[1] S. Dey, V. Cuevas-Vicenttín, S. Köhler, E. Gribkoff, M. Wang, and B. Ludäscher. On implementing provenance-aware regular path queries with relational query engines. In *Proceedings of EDBT/ICDT*, pages 214–223. ACM, 2013.

[2] G. Malewicz, M. H. Austern, A. J. Bik, J. C. Dehnert, I. Horn, N. Leiser, and G. Czajkowski. Pregel: a system for large-scale graph processing. In *Proceedings of SIGMOD*, pages 135–146. ACM, 2010.

[3] The Apache Software Foundation. Apache Giraph. http://giraph.apache.org/, 2013.