Web Semantics: Science, Services and Agents on the World Wide Web 36 (2016) 58–76

Contents lists available at ScienceDirect

Web Semantics: Science, Services and Agents on the World Wide Web

journal homepage: www.elsevier.com/locate/websem

A mobile query service for integrated access to large numbers of online semantic web data sources

William Van Woensel [a,*], Sven Casteleyn [b,c]

[a] Dalhousie University, 6050 University Avenue, Halifax NS B3H 4R2, Canada
[b] Universidad Jaime I, E-12006, Castellón de la Plana, Spain
[c] Vrije Universiteit Brussel, B-1050, Brussels, Belgium

ARTICLE INFO

Article history:
Received 3 March 2015
Received in revised form
30 July 2015
Accepted 20 October 2015
Available online 5 November 2015

Keywords:
Mobile computing
Data integration
Data indexing
Data caching
Cache replacement
Open world assumption

ABSTRACT

From the Semantic Web's inception, a number of concurrent initiatives have given rise to multiple segments: large semantic datasets, exposed by query endpoints; online Semantic Web documents, in the form of RDF files; and semantically annotated web content (e.g., using RDFa), semantic sources in their own right. In various mobile application scenarios, online semantic data has proven to be useful. While query endpoints are most commonly exploited, they are mainly useful to expose large semantic datasets. Alternatively, mobile RDF stores are utilized to query local semantic data, but this requires the design-time identification and replication of relevant data. Instead, we present a mobile query service that supports on-the-fly and integrated querying of semantic data, originating from a largely unused portion of the Semantic Web, comprising online RDF files and semantics embedded in annotated webpages. To that end, our solution performs dynamic identification, retrieval and caching of query-relevant semantic data. We explore several data identification and caching alternatives, and investigate the utility of source metadata in optimizing these tasks. Further, we introduce a novel cache replacement strategy, fine-tuned to the described query dataset, and include explicit support for the Open World Assumption. An extensive experimental validation evaluates the query service and its alternative components.

1. Introduction

The Semantic Web has grown with leaps and bounds over the last decade. Large data sources have been put online in semantic format, and made interoperable via initiatives such as Linked Data [1] (e.g., DBPedia, LinkedGeoData). In addition, small online RDF files, for instance capturing item descriptions (e.g., using DCMI) or personal profiles (e.g., using FOAF), also constitute a large part of the Semantic Web. Sindice [2], a Semantic Web search engine, indexes ca. 708 million of these online sources. In a parallel evolution, increased efforts are being made to make regular (HTML) web content machine-readable as well, catalyzed by the commitment of major search engines to leverage such annotations for improving search results [3]. This evolution has given rise to a new Semantic Web segment, comprising web content enhanced with semantic annotations (e.g., RDFa, microdata). Since most of this annotated content can be converted to RDF data (e.g., see [4]), such annotated websites are semantic sources in their own right. The Web Data Commons initiative [5] (2013) found that ca. 26% of crawled webpages already contain semantic annotations.

Via the Semantic Web, mobile clients gain access to a wealth of online, freely available knowledge. Various mobile computing domains currently leverage semantic data, including augmented reality [6,7], recommender systems [8], location-aware [9,10] and context-aware systems [11,12], mobile tourism [13] and m-Health [14]. Typically, these systems access online semantic data via SPARQL query endpoints. Since they relieve mobile clients of computationally intensive query resolution, query endpoints represent an efficient option for mobile clients. On the other hand, client–server roundtrips cause delays, and a poor or unavailable network connection prevents query resolution. Furthermore, setup and maintenance incur costs, especially when scalability is desired, and requires technical expertise and effort. Therefore, they only present an acceptable cost–benefit ratio for large RDF datasets.

Given recent improvements in mobile hardware, coupled with the development of mobile query engines, an alternative is the local querying of semantic web data [12,15,16]. However, local

* Corresponding author.
E-mail addresses: William.Van.Woensel@Dal.Ca (W. Van Woensel), Sven.Casteleyn@uji.es (S. Casteleyn).

http://dx.doi.org/10.1016/j.websem.2015.10.002
1570-8268/

querying requires the manual, a priori replication of relevant data, and gives rise to data freshness issues. Moreover, some domains do not allow establishing data relevance beforehand; e.g., in context-awareness, relevance is determined by the mobile user's current context, which is updated continuously and often in unforeseeable ways. Due to nearly ubiquitous wireless connectivity, opportunities currently exist to bypass these drawbacks and dynamically retrieve relevant semantic data.

We present a client-side, general-purpose mobile query service, to study the performance and feasibility of on-the-fly querying of a mainly untapped portion of the Semantic Web, consisting of large amounts of RDF files and annotated websites. By supplying integrated query access over these sources, the query service can resolve distributed queries, referencing data from multiple sources. In particular, our solution relies on the dynamic identification, retrieval and caching of semantic data relevant to posed queries. For this purpose, the query service includes two key components; (1) a source identification component, to identify query-relevant sources in the online semantic dataset, and (2) a cache component, locally storing data for later re-use. The query service relies on an existing mobile query engine to locally query retrieved RDF data. To reconcile fine-grained data selection with reducing data processing overhead, these components exploit the semantics of RDF(S)/OWL data.

Studying the efficiency and workability of such local, client-side data collection and query support is desirable for a variety of reasons. First, it is an infrastructure-less solution, where no single party needs to invest in highly scalable server infrastructure or cloud subscriptions. Secondly, keeping data and posed queries at client-side ensures privacy, e.g., especially in context-aware scenarios. Third, by collecting data locally, it ensures query capability for applications in conditions of poor or unreliable network connection. Even with sufficient Internet connectivity, local querying avoids client–server roundtrips, which potentially decrease performance at query time, which is most critical. Finally, it very well supports application scenarios where semantic data fragments are retrieved by other means than the Internet (e.g., via Bluetooth connection, from high capacity RFID tags).

This article builds on earlier work [17], where we presented preliminary versions of the main query service components. In this article, we present an elaborated version of the query service, including extensions that tackle previously identified shortcomings. These include a novel cache removal strategy called Least-Popular-Sources (LPS), tailored to our particular situation where cached data originates from online data files. Secondly, in order to fully support integrated Semantic Web querying, we incorporated the Semantic Web Open World Assumption (OWA). Our experimental validation evaluates the query service using a larger, real-world dataset, focusing on the effects of these extensions on performance and completeness of query results; while at the same time studying boundaries of semantic web technology on current mobile devices.

In the remainder of this article, we first discuss challenges and requirements that arise in our particular querying scenario, together with suitable solutions. Next, an overview of the query service is presented, and its general phases are discussed. We continue by detailing the major query service components, as well as the LPS strategy, and further discuss built-in support for the Semantic Web OWA. Subsequently, the query service is evaluated via an experimental validation. We proceed with a review of the state of the art and end with conclusions and future work.

2. Challenges and requirements

The goal of our mobile query service is to provide transparent, integrated access to a currently untapped part of the Semantic Web, comprising online RDF files and annotated websites. In this mobile querying scenario, a number of issues and challenges arise, which we discuss below.

1. *Mobile device restrictions*: although mobile devices are catching up with desktop and laptop computers, they still have limitations regarding processing and memory capacity (e.g., Android applies a maximum heap depending on the device; currently, for devices with 2–3 GB of RAM, this limit is typically 128–192 MB per Android 5.1 app). Furthermore, battery power is limited, and restricts full and continuous utilization of hardware resources.
2. *Large query dataset*: due to its scale, it is impossible to consider the entire Semantic Web as query dataset. Reflecting this, existing approaches to integrated querying only focus on a (configured) Semantic Web subset. However, our experiments (see Section 7) show that querying even moderately sized datasets is currently not feasible on mobile platforms (e.g., the entire dataset needs to be kept in-memory for fast querying).
3. *Dynamic Semantic Web subset and volatile semantic sources*: typically, mobile applications only require access to a specific Semantic Web subset; ruling out the need to consider the entire Semantic Web (see above). For instance, context-provisioning systems [17] require access to semantic context sources (e.g., place descriptions); while recommender systems [8] require semantic descriptions of items to be recommended. Often, these datasets are only known at runtime and subject to change, which necessitates allowing mobile apps to delineate and dynamically extend their relevant Semantic Web selection [18]. Furthermore, semantic sources themselves may change over time. Depending on the usage scenario, changes may be only occasional (e.g., product descriptions in e-commerce) or frequent (e.g., semantic Internet of Things). In any case, our query service needs to be able to cope with a dynamic set of potentially evolving sources.
4. *Data captured in online, third-party files*: in our querying scenario, data items originate from third-party online files. To gain access to their comprised relevant data, such files need to be fully downloaded, thus retrieving both relevant and irrelevant data. As such, data-retrieval overhead is significantly increased. We also note that connectivity interruptions, not uncommon in mobile scenarios, will result in the query dataset becoming inaccessible.

Taking into account these observed challenges, we formulate the following requirements for efficiently querying large sets of online semantic sources:

1. *Minimizing resource usage*: a local query service should not strain mobile memory and processing capacities, or overly drain the device's battery (challenge 1). Since only a relatively limited amount of fast, volatile memory is available (challenge 1), any additional (volatile) memory requirements (e.g., to store supporting index structures) need to be minimal. Ideally, the additional data should fit in volatile memory to avoid frequent swapping with persistent storage, which unavoidably causes performance loss. Secondly, as mentioned, the query service should enable mobile applications to delineate and dynamically extend or update their relevant Semantic Web selection (challenge 3). This means any internal data structures need to be updateable in real-time and with minimal computational effort, while still supporting acceptable query performance. Finally, battery consumption should be kept within acceptable bounds. For instance, this means reducing battery-intensive operations as much as possible, such as source downloads, which require WiFi or 3/4G radios, and large-scale persistent data retrieval.
2. *Minimizing query dataset*: querying large datasets causes performance problems, especially on mobile platforms (challenge 2). Barring extraordinary mobile hardware improvements in the near future, this implies the query dataset should be kept as small as possible, while still allowing complete query results to