Web Semantics: Science, Services and Agents on the World Wide Web 35 (2015) 167–183

Contents lists available at ScienceDirect

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

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

Semantic reasoning on mobile devices: Do Androids dream of efficient reasoners?

Carlos Bobed *, Roberto Yus, Fernando Bobillo, Eduardo Mena

Department of Computer Science & Systems Engineering, University of Zaragoza, Spain

ARTICLE INFO

Article history:
Received 21 January 2015
Received in revised form
19 August 2015
Accepted 1 September 2015
Available online 24 September 2015

Keywords:
Reasoning
Semantic Web
Mobile computing
Android

ABSTRACT

The massive spread of mobile computing in our daily lives has attracted a huge community of mobile application (*apps*) developers. These developers can take advantage of the benefits of semantic technologies (such as knowledge sharing and reusing, and knowledge decoupling) to enhance their applications. Moreover, the use of semantic reasoners would enable them to create more intelligent applications capable of discovering new knowledge, inferred from the available information.

However, using semantic APIs and reasoners on current mobile devices is not a trivial task. In this paper, we show that the most popular current available Description Logics (DL) reasoners can be used on Android-based devices, and detail the efforts needed to port them to the Android platform. We also analyze the performance of these reasoners on current smartphones/tablets against more than 300 ontologies from the ORE 2013 ontology set, showing that, despite a notable difference with respect to desktop computers, their use is feasible.

1. Introduction

In the last few years, we have witnessed a massive spread of mobile computing which is shaping our daily lives. This has been undoubtedly due to the ever increasing computing capabilities of mobile devices, and the almost pervasive connectivity that the current wireless networks provide us with. As a consequence, thousands of mobile applications (*apps*) have been developed. It seems a promising idea to enhance such applications by taking advantage of the well-known benefits of using ontologies in desktop computers [1], such as the improvement of knowledge sharing, reusing and maintenance, the decoupling of the knowledge from the application, or the possibility of discovering implicit knowledge by using semantic reasoners. However, while mobile applications have attracted a whole of attention, the use of semantic techniques in them is quite far from being usual.

The most direct way to add semantic reasoning to mobile applications would be, indeed, to rely on external servers which would perform all the calculations. However, in this ubiquitous and mobile scenario, where context-awareness and privacy preserving play a crucial role, sending sensitive data to a remote server might be an important privacy breach, and even sending non-sensitive data might be dangerous as it could enable the inference of sensitive information. Despite the fact that reasoning on mobile devices has been the subject of interest from the early stages of the Semantic Web [2,3], it has not been until recently that mobile devices have been considered as effective reasoning platforms (in terms of CPU power, memory, and connectivity). Thus, current mobile devices make local reasoning feasible and open new opportunities in this area (e.g., mixing local with remote reasoning by determining what data can be disclosed and sent to the server and joining the results). Moreover, despite the communications advances, there are situations where connectivity with a server can be faulty or nonexistent, and therefore the only way to perform the reasoning is locally.

As an example of mobile applications that are currently taking advantage of the use of local semantic reasoning, we can include Location-Based Services (LBS) providers, mobile health (m-health) systems, and privacy control applications:

- In LBS providers, it is often particularly important to make decisions considering the user context (e.g., deciding the most relevant information about means of transport at the moment). For instance, the *SHERLOCK* system [4] uses a semantic reasoner and background knowledge about means of transport to infer that both a cab and a tram are interesting for a certain mobile user, given the information obtained from sensors on his/her device such as the location and time. Other location-based

* Corresponding author.
E-mail addresses: cbobed@unizar.es (C. Bobed), ryus@unizar.es (R. Yus), fbobillo@unizar.es (F. Bobillo), emena@unizar.es (E. Mena).

http://dx.doi.org/10.1016/j.websem.2015.09.002
1570-8268/

semantic applications that could benefit from semantic reasoners are *DBpedia Mobile* [5] and *mSpace Mobile* [6].
- In the field of m-health, semantic reasoners have been proposed as part of monitoring systems [7,8] and clinical decision support systems [9] due to the sensitivity of the information managed. *Rafiki* [9] is an example of the latter one that uses a semantic reasoner within the mobile device to infer possible diseases for a patient in a rural area, where connectivity is usually non-existent, given his/her symptoms and context.
- Regarding privacy control applications, local semantic reasoning could also help systems to preserve the privacy of users by inferring whether information about him/her should be shared with other people or applications according to the user context. This has been investigated for smartphones [10] and Google Glass [11].

We expect more applications such as those presented to appear as we have detected the interest in the use of semantic reasoners on mobile devices (for example, the webpage dedicated to our research on Semantic Web on mobile devices [12] had 1500 visits during 2014, and around 2800 during the first semester of 2015).

However, the development of semantic mobile applications has not (yet) spread due, in part, to the fact that there are currently no remarkable efforts to enable mobile devices with semantic reasoning capabilities. A first possibility is developing new reasoners specifically designed for mobile devices. Examples of this alternative include *mTableau* [13], *Pocket KRHyper* [2], *Delta* [14], and *MiniME* [15] reasoners [16]. In order to reuse as much as possible the existing work to optimize current Description Logics (DL) reasoners, we are more interested in another choice: reusing existing semantic reasoners on mobile devices. In particular, we have focused on porting and using existing DL reasoners on Android as they implement the latest reasoning algorithms and optimizations, and their codes have already been tested thoroughly. Moreover, as we will see later in this paper, the efforts needed to port some of them are indeed below the costs of developing new reasoners from scratch, as only a tiny fraction of the source code must be changed.

We have focused our research on devices using Android operating system [17] due to several reasons: (1) its diffusion (51% of devices use this operative system according to a recent estimation,[1] but with a prevision of a heavy steady increase according to its shipment share of 78% during the first quarter of 2015[2]), (2) its openness and thorough documentation, and (3) the existence of a Java-like native virtual machine (Dalvik) that makes it easier to reuse existing Java applications, something very important since most of the semantic APIs and reasoners have been developed in this language.

In our previous works [18,19], we began to study the reuse of existing Semantic Web APIs and DL reasoners on Android. The objective of this paper is to continue this line of research by studying more DL reasoners, performing a more complete evaluation of their performance, and providing a starting point for those who want to add semantics to their mobile systems.

In particular, the contributions of this paper are the following:

- We detail the level of support of some of the existing Semantic Web APIs and DL reasoners on Android devices. Moreover, we share our experience porting and using an important number of available reasoners (CB, ELK, HermiT, jcel, JFact, Pellet, and TrOWL), which makes it easier to port future versions, and might help porting other reasoners. Finally, we also highlight the difficulties that we have found in different versions of the same reasoners.
- We perform a complete empirical evaluation of the performance of the reasoners on Android devices and desktop computers. We consider two reasoning tasks (classification and consistency checking), analyzing the execution time and the impact of memory, virtual machine, and the OWL 2 profile (comparing the differences between the fragments OWL 2 DL and OWL 2 EL). We use more than 300 ontologies from the ORE 2013 ontology set.

The rest of this manuscript is organized as follows. Section 2 starts by overviewing some Semantic Web APIs and reasoners that will be considered throughout this paper and Section 3 summarizes our experiences porting these technologies to Android. Then, after describing our experimental setup in Section 4, Sections 5 and 6 evaluate the performance of the reasoners for the OWL 2 DL and OWL 2 EL profiles, respectively, while Section 7 studies the impact of the memory and the virtual machine. Next, Section 8 includes a global discussion. Finally, Section 9 overviews some related work and Section 10 sets out some conclusions and ideas for future work.

2. Overview of Semantic Web technologies

In this section we overview the Semantic Web technologies that will be considered throughout this paper. In particular, we present the semantic APIs (Jena and OWL API) and DL reasoners that we tried to reuse on Android devices. For completeness sake, we will also reference other relevant DL reasoners in the literature. Finally, we review DL reasoners specifically designed for mobile devices.

2.1. Semantic web APIs

Jena[3] [20] is an ontology API to manage OWL ontologies and RDF data in Java applications. Jena is appropriate to manage OWL 1 Full ontologies, but support for OWL 2 is not available yet. However, it is much more used for the serialization of RDF triples and the manipulation of RDF graphs. Jena can interact with semantic reasoners to discover implicit knowledge. The latest versions of Jena are split into two packages, namely `jena-fuseki` (with the Jena SPARQL server), and `apache-jena` (with APIs, SPARQL engine, RDF database, and other tools).

OWL API[4] [21] is an ontology API to manage OWL 2 ontologies in Java applications and provides a common interface to interact with DL reasoners. It can be considered as a de facto standard, as the most recent versions of most of the semantics tools and reasoners use the OWL API to load and process OWL 2 ontologies. The OWL API is able to process each of the OWL 2 syntaxes defined in the W3C specification (functional, RDF/XML, OWL/XML, Manchester, and Turtle) and to identify the OWL 2 profiles (OWL 2 DL, OWL 2 EL, OWL 2 QL, and OWL 2 RL). The OWL API is less appropriate for the management of OWL 2 Full or RDF ontologies.

2.2. DL reasoners

CB[5] [22] (Consequence-Based) reasoner supports a fragment of OWL 2 (Horn-$\mathcal{SHIF}$). As its name suggests, the reasoner algorithm does not build models but infers new consequent axioms. CB is implemented in OCaml and, as far as we know, the only supported reasoning task is classification. It can be used from command line, as a Protégé plug-in, and through the OWL API.

1 http://www.netmarketshare.com, last accessed 2015-07-04.
2 http://www.idc.com/prodserv/smartphone-os-market-share.jsp, last accessed 2015-07-04.
3 http://jena.apache.org.
4 http://owlapi.sourceforge.net.
5 http://www.cs.ox.ac.uk/isg/tools/CB.