# A Mobile Reasoner for Semantic-based Matchmaking

M. Ruta, F. Scioscia, G. Loseto, F. Gramegna, and E. Di Sciascio

Politecnico di Bari, via E. Orabona 4, I-70125 Bari, Italy
E-mail: {m.ruta,f.scioscia,disciascio}@poliba.it, {gramegna,loseto}@deemail.poliba.it

**Abstract.** Reasoning in pervasive computing has to face computational issues inherited by mobile platforms. This paper presents a prototypical reasoner for mobile devices, which leverages Semantic Web technologies to implement both standard (subsumption, satisfiability, classification) and non-standard (abduction, contraction) inferences for moderately expressive knowledge bases. System features are surveyed, followed by early performance analysis.

## 1 Introduction

Semantic Web technologies have been proposed as a candidate to promote interoperability and intelligent decision support in ubiquitous computing contexts –*e.g.*, supply chain management and u-commerce [6], peer-to-peer resource discovery [7] and so on– keeping rich and structured the exchanged information. Reasoning and query answering for resource discovery in ubiquitous contexts is a critical issue. Mobile computing platforms –albeit increasingly effective and powerful– are still featured by hardware/software limitations. Most mobile inference engines currently provide only rule processing for materialization of entailments in a Knowledge Base (KB) [4, 5], not supporting advanced inferences and extensive reasoning over ontologies [5]. Standard satisfiability and subsumption provide only boolean "yes/no" answers to queries. Non-standard inferences, like Concept Abduction and Concept Contraction, are needed to enable a more fine-grained semantic ranking as well as explanations of outcomes [2]. Implementation of tableaux algorithms on mobile devices exhibited serious memory impact [8]. Moreover, current Semantic Web reasoners cannot be ported without a significant re-write effort. In [7] a different design approach was followed to adapt non-standard inferences to ubiquitous computing. Expressiveness of logic language and axioms was limited in a way that structural algorithms could be adopted, but maintaining it enough for broad application areas. Similar principles motivated independent research work on $\mathcal{EL}^{++}$ structural reasoners [1] and the definition of OWL 2 *profiles*. This paper introduces a prototypical mobile reasoner[1] compliant with Semantic Web technologies through the OWL API [3] and implementing both standard reasoning tasks (subsumption, classification, satisfiability) and non-standard inferences for semantic-based matchmaking

[1] Mini-ME, the Mini Matchmaking Engine – http://sisinflab.poliba.it/swottools/minime/

Table 1: Syntax and semantics of $\mathcal{ALN}$ constructs and simple-TBoxes

| Name | Syntax | Semantics |
|---|---|---|
| Top | $\top$ | $\Delta^{\mathcal{I}}$ |
| Bottom | $\bot$ | $\emptyset$ |
| Intersection | $C \sqcap D$ | $C^{\mathcal{I}} \cap D^{\mathcal{I}}$ |
| Atomic negation | $\neg A$ | $\Delta^{\mathcal{I}} \setminus A^{\mathcal{I}}$ |
| Universal quantification | $\forall R.C$ | $\{d_1 \mid \forall d_2 : (d_1, d_2) \in R^{\mathcal{I}} \rightarrow d_2 \in C^{\mathcal{I}}\}$ |
| Number restriction | $\geq nR$ | $\{d_1 \mid \sharp\{d_2 \mid (d_1, d_2) \in R^{\mathcal{I}}\} \geq n\}$ |
| | $\leq nR$ | $\{d_1 \mid \sharp\{d_2 \mid (d_1, d_2) \in R^{\mathcal{I}}\} \leq n\}$ |
| Inclusion | $A \sqsubseteq D$ | $A^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ |
| Equivalence | $A \equiv D$ | $A^{\mathcal{I}} = D^{\mathcal{I}}$ |

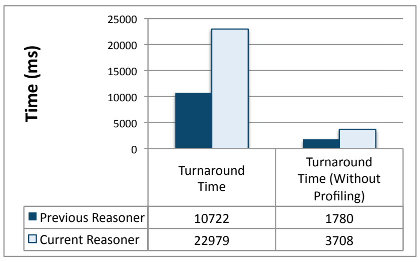
(abduction and contraction [2]). It is developed in Java, with Android as target platform. The paper is so structured: Section 2 describes the system, while Section 3 reports on early experiments before concluding in Section 4.

## 2 System Outline

The system prototype is compatible with the Android Platform version 2.1 (API level 7) or later. It runs either as a *service* (*i.e.*, a background daemon) invoked by Android applications, or as a library. In the latter form, it runs unmodified on Java Standard Edition, version 6 or later. The system supports OWL 2 ontology language, in all syntaxes allowed by the adopted OWL API library. The adopted logic language is $\mathcal{ALN}$ (Attributive Language with unqualified Number restrictions) Description Logics in *simple-TBox* hypothesis [7], which keeps polynomial the computational complexity of standard and non-standard inferences. Main constructs are summarized in Table 1. The reasoning framework is based on structural algorithms. When a knowledge base is loaded, it is preprocessed with *unfolding* and *Conjunctive Normal Form* (CNF) *normalization* [7]. Beyond standard ***Concept Satisfiability*** (a.k.a. consistency) and ***Subsumption test***, the proposed system supports ***Concept Contraction*** and ***Concept Abduction*** non-standard inferences, enabling a semantic matchmaking and relevance ranking of available resources w.r.t. a request [2, 7]. ***Ontology Satisfiability*** and ***Classification*** reasoning services over ontologies are also supported.

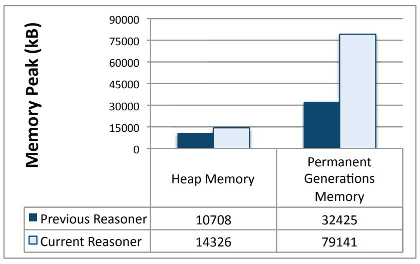
## 3 Performance Evaluation

Performance evaluation has been carried out as a comparison with a previous version [7] of the matchmaker, which was developed for the Java ME (Micro Edition) platform. This choice is motivated due to the lack of a mobile device supporting both Java ME and Android, hence it was unfeasible to test the old and new versions on a unique target handheld. The comparison has been done by running the same test suite with both matchmakers on a PC equipped with an Intel Core2 Duo T7300 CPU, 2 GB of RAM, Fedora 16 operating system and Oracle Java SE 7u3. The test performs unfolding and normalization on an ontology in [7] –having a size of 528 kB– and 100 request/supply pairs –with



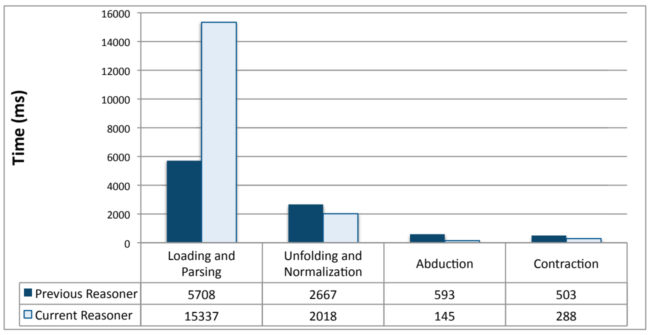
| | Turnaround Time | Turnaround Time (Without Profiling) |
|---|---|---|
| ■ Previous Reasoner | 10722 | 1780 |
| □ Current Reasoner | 22979 | 3708 |

Fig. 1: Turnaround Time



| | Heap Memory | Permanent Generations Memory |
|---|---|---|
| ■ Previous Reasoner | 10708 | 32425 |
| □ Current Reasoner | 14326 | 79141 |

Fig. 2: Memory Usage



| | Loading and Parsing | Unfolding and Normalization | Abduction | Contraction |
|---|---|---|---|---|
| ■ Previous Reasoner | 5708 | 2667 | 593 | 503 |
| □ Current Reasoner | 15337 | 2018 | 145 | 288 |

Fig. 3: Processing Time

an average size of 4.2 kB– and finally executes Concept Contraction and Concept Abduction inferences between each pair. The VisualVM Java profiler was used to measure execution times and memory consumption. The computational impact of the profiler is evident (see Figure 1): the actual execution time may be up to 6 times smaller. We further analyzed the time required by the main processing steps: loading and parsing input documents; unfolding and normalization; abduction; contraction. As shown in Figure 3, most of the time (66.7%) was spent by the OWL API to load and parse input files. It took about 15 seconds to process the ontology and 100 request/supply pairs whereas the previous code took 5.7 seconds (53.2% of total time). As far as normalization and unfolding are concerned, the old tool completed in 2667 ms, while the new one did the same in 2018 ms (1.32 times faster, approximately). Figure 3 also shows execution times for the abduction and contraction tasks. The old reasoner ran 100 abduction tests in 593 ms, while the current one did the same in 145 ms, *i.e.*, 4 times faster. For the contraction service, the previous software called the method 100 times without checking for concept compatibility first, resulting in a total time of 503 ms. The new reasoner took only 26.4 ms to perform 100 compatibility checks; 64 of them returned false and triggered contraction for a total of 262 ms, which is approximately 1.74 times faster than the old one. Figure 2 shows the *heap* –used for dynamic allocation of new class instances and arrays– and *permanent generation* –where long-lived objects are moved– memory peak that was reached during the test. Memory consumption appears to have increased for both areas w.r.t. the previous system. Experiments evidence design and adopted optimization resulted in faster execution of non-standard inference services for semantic matchmaking w.r.t. previous tools. The integration of the OWL API improved flexibility in using KBs, and compatibility with Semantic Web languages. Nevertheless, it is a performance bottleneck for processing time and presumably for memory consumption.

## 4 Conclusion

A prototypical reasoner for mobile computing was presented. Early experiments evidenced correctness and efficiency of non-standard reasoning services, whereas the integration of the OWL API introduced a performance penalty. Future work will be devoted to OWL interface optimization and to the enhancement of both allowed reasoning features and supported logic languages.

## Acknowledgments

The authors acknowledge partial support of national project ERMES (Enhance Risk Management through Extended Sensors) - PON (2011-2014).

## References

1. Baader, F., Brandt, S., Lutz, C.: Pushing the EL envelope. In: International Joint Conference on Artificial Intelligence. vol. 19, p. 364. Lawrence Erlbaum Associates LTD (2005)
2. Colucci, S., Di Noia, T., Pinto, A., Ragone, A., Ruta, M., Tinelli, E.: A Non-Monotonic Approach to Semantic Matchmaking and Request Refinement in E-Marketplaces. International Journal of Electronic Commerce 12(2), 127–154 (2007)
3. Horridge, M., Bechhofer, S.: The OWL API: a Java API for working with OWL 2 ontologies. Proc. of OWL Experiences and Directions 2009 (2009)
4. Kim, T., Park, I., Hyun, S., Lee, D.: MiRE4OWL: Mobile Rule Engine for OWL. In: Computer Software and Applications Conference Workshops (COMPSACW), 2010 IEEE 34th Annual. pp. 317–322. IEEE (2010)
5. Motik, B., Horrocks, I., Kim, S.: Delta-Reasoner: a Semantic Web Reasoner for an Intelligent Mobile Platform. In: Twentyfirst International World Wide Web Conference (WWW 2012). ACM (2012), to appear
6. Ruta, M., Di Noia, T., Di Sciascio, E., Piscitelli, G., Scioscia, F.: RFID meets bluetooth in a semantic based u-commerce environment. In: Proc. of the ninth international conference on Electronic commerce. pp. 107–116. ACM (2007)
7. Ruta, M., Di Sciascio, E., Scioscia, F.: Concept abduction and contraction in semantic-based p2p environments. Web Intelligence and Agent Systems 9(3), 179–207 (2011)
8. Sinner, A., Kleemann, T.: KRHyper - In Your Pocket. In: Proc. of 20th International Conference on Automated Deduction (CADE-20). pp. 452–457. Tallinn, Estonia (July 2005)