Towards Timely and Efficient Semantic Reasoning for the Networked Society

Bo Xing[1], Johan Hjelm[2], Takeshi Matsumura[2], Shingo Murakami[2], Toshikane Oda[2], and Andrew Ton[1]

[1] Ericsson Research, USA, 200 Holger Way, San Jose, CA 95134, USA
[2] Ericsson Research, Japan, Momento Shiodome 2-3-17, Higashi-Shinbashi, Minato-ku, Tokyo 105-0021, Japan

Abstract. This paper presents our work in progress on enabling computerized reasoning capability in machine-to-machine communication scenarios for the Networked Society (or Internet of Things). Such reasoning capability is about drawing high-level conclusions on the situation in real time based on raw data streams generated by various sources. There are challenges posed by the dynamic and heterogenous availability of raw data coming from different sources, as well as the stringent time constraints for conclusions to be made. Our goal hence is to make machine-based reasoning processes time-efficient, resource-efficient, and scalable. We present an approach that addresses the challenges by decomposing a reasoning process into two stages: "shallow reasoning" and "deep reasoning". The former deals with the dynamic and heterogenous availability of raw data from different sources, while the latter executes semantic reasoning with a lightweight workload that has been reduced by the former. We present our prototype implementation of a reasoning system that adopts the proposed approach in a proactive healthcare use case. Performance evaluation is currently ongoing to verify the effectiveness of our approach.

Keywords: Reasoning, Semantic, Ontology, Timeliness, Efficiency.

1 Introduction

At Ericsson, we envision that, by the year of 2020, there will be 50 billion connected devices on this planet. By then, any device that would benefit from being connected will be connected (through any means). It is Ericsson's strong belief that when one person connects, their life changes; with everything connected, our world changes. This vision of "Networked Society" [1] (or "Internet of Things") is not about simply providing bit-pipes for people and their devices. It is rather to inter-connect the device network, the people network and the information network, and to have them autonomously and intelligently work for the good, facilitating us to collaborate, innovate, sustain, learn, care and participate.

To enable the needed autonomy and intelligence in the Networked Society, machine-based reasoning capability is an indispensable building piece. It is the

D. Uhler, K. Mehta, and J.L. Wong (Eds.): MobiCase 2012, LNICST 110, pp. 264–273, 2013.

capability of machines deriving high-level conclusions from low-level raw data sets. The conclusions facilitate decisions to be made for further actions to be taken without human intervention. In many cases, input data constantly comes from various data sources (e.g., machines or sensors) as streams and is brought together for machines to make sense of it, so that appropriate actions can be promptly and automatically taken by the same or other machines. For example, a public safety surveillance system comprising sensors and actuators reasons over contextual readings collected from the sensors to infer the situation, and based on that triggers the right actuators to respond to the situation accordingly. In another example, a health monitoring application detects anomalies in a person's health condition based on biometric measurements from personal health devices plus other contextual info, and reports it to the person's primary doctor.

Such Machine-to-Machine (M2M) communication, processing and actuation scenarios pose a set of challenges on a reasoning system. First, the reasoning mechanisms available today (such as fuzzy logic, Bayesian logic, case-based reasoning, and semantic reasoning) were designed for different purposes and hence come with varying resource consumption profiles (in terms of processing power usage, processing time and memory footprint). How to combine and make the best use of these techniques while optimizing overall resource consumption needs to be explored. Second, on the input end of a reasoning system, availability of data at various sources is not always deterministic. Inconsistent update frequency at data sources, broken connectivity to data sources and others can all cause unavailability, intermittent availability or delayed availability. Third, on the output end, the receivers of the conclusions from the reasoning system oftentimes have stringent requirements on timeliness - they issue real-time queries/requests and need to get back the answer before a deadline. Hence, they typically are okay with a coarse-grained answer as long as it meets the deadline. In case of less stringent time constraints, they certainly would like the answer to be as fine-grained as possible. Fourth, with millions and billions connected things in the "Networked Society", a reasoning system will need to deal with a large number of reasoning tasks concurrently. Hence, making it scalable is of great importance. Last but not the least, the reasoning system desirably comes with semantic support for increased interoperability, meaning that the conclusions it draws are made available in machine-understandable forms, enabling more sophisticated events to be auto-triggered and the corresponding actions to be auto-taken.

In this paper, we develop a reasoning framework that aims to address to the maximum extent the aforementioned challenges, namely resource efficiency, data availability, timeliness, scalability and semantic support. In brief, our approach is to constitute the core of a reasoning system with "*Shallow Reasoners*" and "*Deep Reasoners*". Deep reasoners employ semantic web technologies to enable interoperability and semantic support, and hence take the relatively heavy-weight reasoning workload. On the other hand, shallow reasoners are intended to offload part of the reasoning work (as a first attempt to enhance resource efficiency) as well as to deal with data availability at various sources. Thus, a reasoning process becomes a pipelined 2-stage procedure. In the first stage - shallow reasoning,

shallow reasoners each handling data from a particular source pre-processes the data and generates intermediate results that will facilitate semantic reasoning. In the second stage - deep reasoning, deep reasoners takes over from shallow reasoners the intermediate results as they become available, and incrementally performs ontology-based reasoning over them; they make available just-in-time conclusions with increasing granularity. Deep reasoners are optimized to minimize resource consumption and concurrently accommodate as many reasoning tasks as possible. As a proof of concept, we have implemented a prototype reasoning system that is targeted for a specific use case scenario - proactive healthcare.

The remainder of the paper is organized as follows. We first review related prior work in Section 2. We then present our approach to reasoning and the rationale behind it in Section 3. Following that, Section 4 describes our implementation of a prototype system that adopts the proposed approach to address a real-world use case. We finally conclude the paper in Section 5 by briefing our plan on performance evaluation.

2 Related Work

The vision that we are moving towards the Internet of Things era has been shared within the research and industrial communities [2] [3]. A large body of studies have been done and continue to be conducted on how to make the evolution happen [4] [5]. The first thing clearly is to make physical things connected, communicate and be recognizable, and a significant amount of efforts have made that possible today through various types of technologies [6] [7]. That however does not suffice to make a Networked Society, as intelligence needs to be brought to the connected, recognizable and communicating things for them to be able to work with people, for people and on behalf of people. A popular approach towards enabling this intelligence is to fuse semantic web technologies into connected sensors and objects, so that the information generated by any of them is understood in a common and consistent way [8] [9]. With that, sharable knowledge is created through the use of semantic reasoning techniques [10] [11].

3 Our Approach to Reasoning

One simple way of reasoning is to use pre-programmed reasoning rules, as is typically implemented in database applications. Such rules allow for deriving conclusions based on specific data - but only on those data. Including a new data type requires re-programming. In the "Networked Society" scenario, data sources are not static, and new data sources can be added and existing data sources removed in an ad-hoc manner. Considering that, a more favorable approach to reasoning is to have heterogeneous data structured in a machine-understandable way, which allows conclusions to be drawn automatically by a piece of software, often assisted by a set of generic rules. Thus, there are no longer limitations on which data sources can be used and which problem spaces can be addressed.

In light of the above, we choose to represent data using ontology. An ontology formally represents knowledge as a set of concepts within a domain, and the relations between these concepts [12]. By using ontology, the explicit specification of a problem domain can be shared, thus enabling interoperability between the reasoning system and other systems, services and applications. The use of ontology also makes it easy to introduce new problem spaces to the reasoning system (by simply importing ontologies that capture different domains). Moreover, with the expressiveness of ontology, supporting sophisticated reasoning, processing and querying with semantics is possible.

3.1 Overview

Ontology-based reasoning, however, is a non-trivial and costly job at run-time – it is highly time- and resource-consuming [13]. Even though the ontology used is kept as small as possible, long reasoning time and high resource consumption is still an unavoidable price to pay. To minimize the negative impact that ontology-based reasoning has on performance while still harnessing its power in sophisticated reasoning and semantic support, we need to (1) optimize the size (and possibly the shape) of the *base ontology* that describes the reasoning problem domain, (2) use ontology-based reasoning only for the part of the reasoning work that needs its power in semantic reasoning, and (3) decouple ontology reasoning from data sources, which may come and go dynamically and have availability constraints.

To that end, we approach reasoning with two key mechanisms. First, we define the base ontology to include only high-level concepts and to entail a conclusion tree. Without low-level concepts (such as measurements at sensory machines), the base ontology significantly shrinks in size. The conclusion tree contains possible conclusions at different levels of granularity, thus allowing for timely exposure of conclusions in an incremental manner. Second, we conduct reasoning as a two-step process – "shallow reasoning" and "deep reasoning". Shallow reasoning is the process of reasoning over raw data from data sources and deriving intermediate results that are representable in the base ontology. Deep reasoning is the process of reasoning over the base ontology containing the intermediate results and drawing conclusions. Working together, shallow reasoning offloads the part of work that is simple but deals with large amount of data (probably with availability constraints), while deep reasoning makes the most efficient use of the expensive but powerful semantic reasoning techniques.

3.2 Base Ontology – Minimizing Reasoning Complexity and Enabling Timely Availability of Conclusions

For minimizing its size, we define the base ontology such that it does not include low-level concepts such as measurements at sensory machines, but only high-level concepts like what a measurement indicates. To enable timely availability of conclusions, it conceptually entails a conclusion tree.

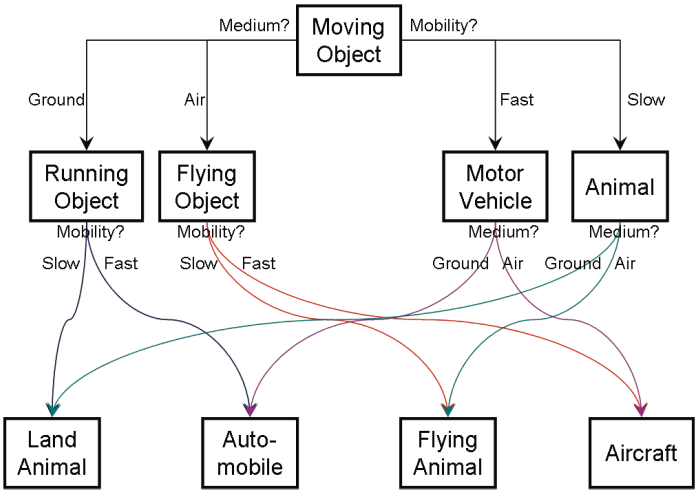

Fig. 1. An Illustrative Example of a Conclusion Tree

The nodes in a conclusion tree represent possible conclusions the reasoning system may arrive at. Fig. 1 illustrates an example conclusion tree in the case of a surveillance reasoning system, where the goal is to detect the type of a moving object based on sensors that identify the speed of the object and the medium it moves in. The root node of conclusion tree is the known concept that all possible conclusions belong to (e.g., an moving object in the case of a surveillance system). Every other node represents a conclusion that is refined over the conclusions represented by its parent nodes (note that here each node can have multiple parent nodes, and hence the conclusion tree is not a tree in the graph theory sense).

The refinement of a conclusion to reach a child node in the conclusion tree is enabled by the availability of additional data from data sources, and is formally stated as a rule (or Equivalent Class property) in the base ontology. In our example, when Medium data is available, a conclusion of Animal can be refined to become either Land Animal or Flying Animal. Further, the refinement can be accompanied by a confidence value or a formula to calculate it (e.g., defined as a SWRL (Semantic Web Rule Language) rule), indicating the accuracy or certainty of reaching the refined conclusion. The depth of a node in the tree implies the granularity of the corresponding conclusion - the deeper a node is at, the finer-grained the conclusion is; a leaf node represents a conclusion that has the finest granularity possible. In our example, the four conclusions at the bottom of the tree are the best conclusions we may end up drawing.

3.3 Deep Reasoning – Further Reducing Resource Consumption and Enhancing Scalability

In addition to exploiting shallow reasoning for lower resource consumption, optimizations are employed inside the deep reasoning process to further reduce the footprint. Although the conclusions from deep reasoning are persisted in an ontology database, the process of deep reasoning is executed in memory. This is in light of the poor performance of executing ontology reasoning over ontology databases.

The deep reasoning process manages each reasoning task as an ontology graph comprising instances of the concepts defined in the base ontology. Taking the surveillance system depicted in Fig. 1 as an example, a reasoning task handles detecting the type of one particular moving object. Hence, its ontology graph would contain instances of concepts including the mobility of the particular moving object and the medium it moves in, which are instantiated from intermediate results generated by shallow reasoners once they become available. Whenever such intermediate results are received from shallow reasoners, ontology-based reasoning is executed over ontology graphs to derive possible conclusions.

For an ontology graph, as soon as a conclusion that is not the root node of the conclusion tree is arrived at, the inferred ontology is persisted in the ontology database. Thus, the conclusion is immediately available for the receivers of outcomes from the reasoning system. For example, the conclusion that the moving object is a motor vehicle is good enough for some actions to be taken by the surveillance system; later when the conclusion is refined to be an aircraft, the surveillance system could finetune its reaction. Finally, once a conclusion that is the leaf node of the conclusion tree is arrived at, its ontology graph is removed after being persisted. The reasoning task finishes, as the best possible conclusion has been drawn.

In order to enhance scalability, each run of ontology-based reasoning works over a set of ontology graphs with an upper-bounded number of axioms (an axiom is a statement in an ontology). This is to account for the fact that the performance of ontology reasoning degrades exponentially with the increase in ontological axioms. When there are a large number of ontology graphs to be reasoned over, they are split up and assigned to different runs. Those runs can take place either sequentially or concurrently (if distributed to multiple deep reasoners physically residing on different machines).

4 Prototype Implementation

In this proof-of-concept prototype, we target a simple use case, where a user's situation (such as relaxing, dancing, drinking, mountain hiking, accident on freeway, etc.) is continuously derived in real time based on data collected from various sources. Such reasoning capability would facilitate recommendation, advertisement or event triggering services to become more personalized and context-aware. The data sources for the prototype include (but are not restricted to): (1) a fitness wrist band the user wears, (2) a smart phone the user carries, and

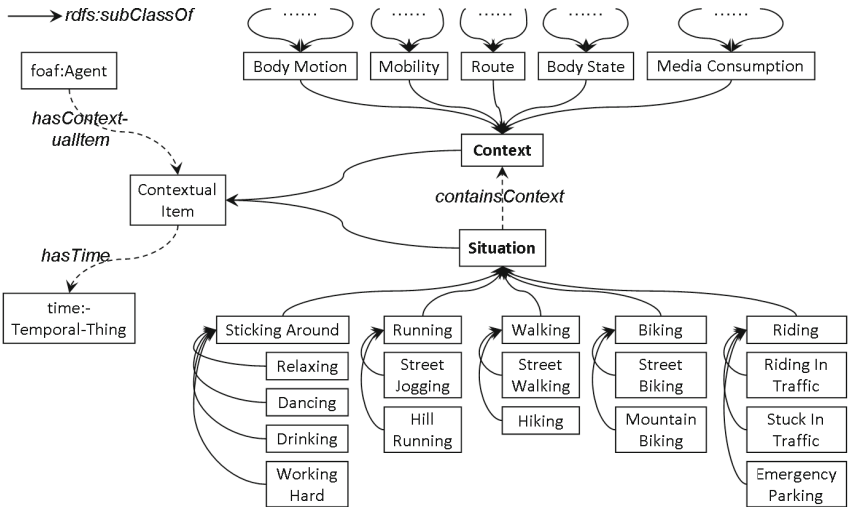

Fig. 2. Visualization of the Base Ontology

(3) an online music service the user subscribes to. Types of data available from these sources are (1) Galvanic Skin Response (GSR), (2) skin temperature, (3) wrist moving intensity, (4) GPS coordinates, and (5) music listened to.

4.1 Base Ontology

Fig. 2 is a visualization of the base ontology that defines our targeted problem. As we are concerned with deriving a user's situation during a particular period of time, the most important class in the ontology is the "Situation" class. That is the class which all conclusions to be drawn are instantiated as individuals as. Another important class is the "Context" class, which defines possible intermediate results that may come from Shallow Reasoners inferred from raw data. The direct subclasses of "Context" are the types of the intermediate results: "BodyMotion", "Mobility", etc. Further down this hierarchy are the possible values of each type of intermediate results, defined as their subclasses. For example, under "BodyMotion", there are "SignificantBodyMotion", "MediumBodyMotion", "InsignificantBodyMotion". Between "Situation" and "Context", there is an object property indicating that a "Situation" contains a "Context". This captures the fact that a situation comprises multiple contextual elements - they determine what situation it is. It is also the basis for ontology reasoning in our prototype. Under the "Situation" class, all possible conclusions are defined as the subclasses of "Situation" in a hierarchical way, forming a conclusion tree. Each subclass is defined with an equivalent class property, describing what "Contexts" the "Situation" should contain. For example, the "Relaxing" class is an equivalent class of "Situation" and "containsContext StaticMobility" and "containsContext InsignificantBodyMotion" and "containsContext CalmBodyState".

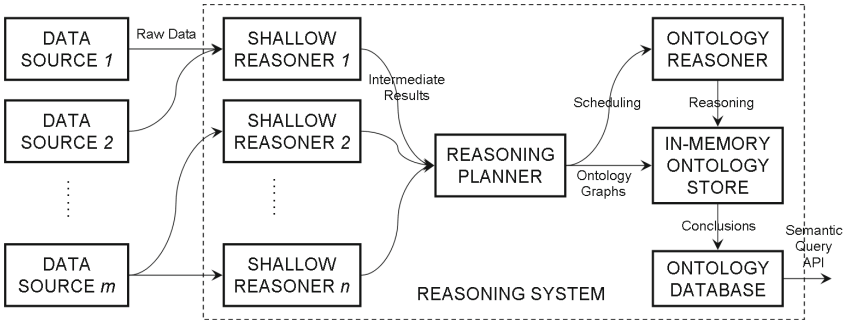

Fig. 3. Prototype Implementation Architecture

4.2 Architecture

The high-level architecture of the prototype is depicted in Fig. 3 (note that the components are logical, and physically can be distributed when needed). A set of Shallow Reasoners are connected to the corresponding data sources, preprocessing incoming raw data and deriving intermediate results. Each Shallow Reasoner connects to one or multiple data sources, and generates one or more intermediate results. They may employ various reasoning techniques such as fuzzy logic, Bayesian networks, etc. For our targeted use case in particular, we have a Body Movement reasoner, a Body State reasoner, a Mobility reasoner, and a Music Emotion reasoner. These Shallow Reasoners either use pre-programmed rules or exploit fuzzy logic reasoning through the use of the FuzzyDL library [14]. The output ends of the Shallow Reasoners are connected to the Reasoning Planner, which is responsible for scheduling and keeping track of reasoning tasks. The In-Memory Ontology Store is where the ontology graphs are dynamically loaded and ontology reasoning takes place; it is implemented using the Manchester OWL-API [15]. We choose to use Pellet [16] as the Ontology Reasoner, considering its incremental classification capability, compatibility with OWL-API, and support for SWRL rules. The Ontology Database is where the final conclusions (in the form of ontologies) are persisted and exposed to the outside world through semantic queries (e.g. SPARQL) or non-semantic queries (e.g., REST API parameters). We choose to use graph database Neo4j [17] as the base to develop our Ontology Database, and optimize it for potentially popular queries in our target scenarios.

Upon receiving an intermediate result from a Shallow Reasoner, the Reasoning Planner instantiates an individual of the "Context" class capturing it. If not done yet, it creates an ontology graph for the concerned situation by instantiating a "Situation" individual. The "Context" individual is filled in the ontology graph based on its relations with the "Situation" individual and other properties. The Reasoning Planner keeps track of the unique ID of each ontology graph , and associates a timer with it, controlling its lifetime inside the In-Memory Ontology Store. The Ontology Reasoner is scheduled to execute periodically when

there are ontology graphs in the In-Memory Ontology Store. When executing, it reasons over the base ontology plus all ontology graphs using incremental consistency checking and classification. It finds out the most specific subclass each "Situation" individual belongs to. If for a particular ontology graph, the "Situation" individual is derived to be also an individual of a subclass of "Situation", the inferred ontology graph is immediately persisted in the Ontology Database. Thus, the currently best available conclusion (the finest-grained type of situation) is made available for access right away. Meanwhile, if a finest-grained conclusion (a subclass of "Situation" that does not have any subclass) has been reached, the ontology graph is removed from the store (otherwise it will be removed when its predefined lifetime expires). As additional raw data comes from data sources over time, the conclusions exposed at the Ontology Database keep being updated with increasingly finer granularity.

5 Concluding Remarks

In this paper, we have presented our approach to timely and efficient reasoning for the Networked Society, where connected things intelligently work with, for and on behalf of people. We also presented our prototype implementation that proves the concepts in a specific use case setting. We are currently working on evaluating the performance of our approach by experimenting with the prototype system. The key performance metrics we look at are CPU usage, memory consumption and reasoning time. We design and run experiments to compare our approach against a mechanism that places all the workload on semantic reasoners, and to compare it against a system without using the conclusion tree method. Further, driven by the specific use case that we target, we vary the number of users to assess the scalability of the approach. We look forward to reporting the results of the evaluations in the near future.

References

1. Ericsson, Networked society, `http://www.ericsson.com/networkedsociety/`
2. Gershenfeld, N., Krikorian, R., Cohen, D.: The internet of things. Scientific American 291(4) (2004)
3. SAP, Internet of things: An integral part of the future internet (2011), `http://services.future-internet.eu/images/1/16/A4_Things_Haller.pdf`
4. C. Associati, The evolution of internet of things (2011)
5. Uckelmann, D., Harrisson, M., Michahelles, F.: An architectural approach towards the future internet of things. Architecting the Internet of Things (2011)
6. Welbourne, E., Battle, L., Cole, G., Gould, K., Rector, K., Raymer, S., Balazinska, M., Borriello, G.: Building the internet of things using rfid: The rfid ecosystem experience. IEEE Internet Computing 13(3) (2009)
7. Kortuem, G., Kawsar, F., Sundramoorthy, V., Fitton, D.: Smart objects as building blocks for the internet of things. IEEE Internet Computing 14(1) (2010)
8. Sheth, A., Henson, C., Sahoo, S.: Semantic sensor web. IEEE Internet Computing 12(4) (2008)

9. Pfisterer, D., Römer, K., Bimschas, D., Kleine, O., Mietz, R., Truong, C., Hasemann, H., Kröller, A., Pagel, M., Hauswirth, M., Karnstedt, M., Leggieri, M., Passant, A., Richardson, R.: Spitfire: toward a semantic web of things. IEEE Communications Magazine 49(11) (2011)
10. Wei, W., Barnaghi, P.: Semantic Annotation and Reasoning for Sensor Data. In: Barnaghi, P., Moessner, K., Presser, M., Meissner, S. (eds.) EuroSSC 2009. LNCS, vol. 5741, pp. 66–76. Springer, Heidelberg (2009)
11. Steller, L.A., Krishnaswamy, S., Gaber, M.M.: Enabling scalable semantic reasoning for mobile services. International Journal on Semantic Web and Information Systems 5(2) (2009)
12. Gruber, T.: A translation approach to portable ontology specifications. Knowledge Acquisition 5(2)
13. Reichle, R., Wagner, M., Khan, M.U., Geihs, K., Lorenzo, J., Valla, M., Fra, C., Paspallis, N., Papadopoulos, G.A.: A Comprehensive Context Modeling Framework for Pervasive Computing Systems. In: Meier, R., Terzis, S. (eds.) DAIS 2008. LNCS, vol. 5053, pp. 281–295. Springer, Heidelberg (2008)
14. Fuzzydl, `http://gaia.isti.cnr.it/straccia/software/fuzzyDL/fuzzyDL.html`
15. Owl-api, `http://owlapi.sourceforge.net/`
16. Pellet, `http://clarkparsia.com/pellet/`
17. Neo4j, `http://neo4j.org/`