AKTive Workgroup Builder (AWB): Constraint Satisfaction Problem Solving over the Semantic Web

Craig McKenzie cmckenzie@csd.abdn.ac.uk
Alun Preece apreece@csd.abdn.ac.uk
Peter Gray pgray@csd.abdn.ac.uk
Department of Computer Science, University of Aberdeen, Aberdeen AB24 3UE, UK
http://www.csd.abdn.ac.uk/research/akt/cif

ABSTRACT

In this paper we introduce the AKTive Workgroup Builder (AWB) web application and describe how it uses distributed RDF data, defined against an OWL Lite ontology, to build and solve a user defined Constraint Satisfaction Problem (CSP). We describe our approach to mixed mode reasoning using both ontological and rule based methods and discuss how some of the factors relating to this affect the design of the system. We explain how we utilise derivation rules, expressed in the Semantic Web Rule Language (SWRL) to further enrich our knowledge. Fully quantified constraints are then expressed against this semantic data using CIF/SWRL – an extension of SWRL using our Constraint Interchange Format (CIF). To the best of our knowledge, the AWB is unique in that no other semantic web application combines these various mechanisms to perform hybrid reasoning.

Keywords

Semantic Web, Quantified Constraints, Derivation Rules, Reasoning, Inference, Application

1. INTRODUCTION

As more and more semantic data becomes available we aim to explore hybrid reasoning on the Semantic Web (SW), specifically the interplay between ontological inference, rules and constraints. An interesting starting domain was within the context of the CS AKTive Space [3], a repository of information about the Computing Science (CS) community in the UK. The AKTive Workgroup Builder (AWB) is a SW application that attempts to solve the practical problem of assembling a workgroup from a pool of known individuals. Using the demo version of the AWB, a user constructs a workgroup by following these steps:

1. Information about the pool of people to be considered is gathered from relevant datasource(s);
2. Quantified constraints are specified by the user about the composition of the workgroup(s), e.g. minimum and maximum size, the focus, etc.;
3. Reasoning is performed against the data to determine eligibility, e.g. does a person have the relevant research interests and can those interests be determined if not explicitly stated?
4. Finally, the constraint satisfaction problem is built and then passed on to a solver that attempts to compose workgroup(s) that satisfy the stipulated constraints.

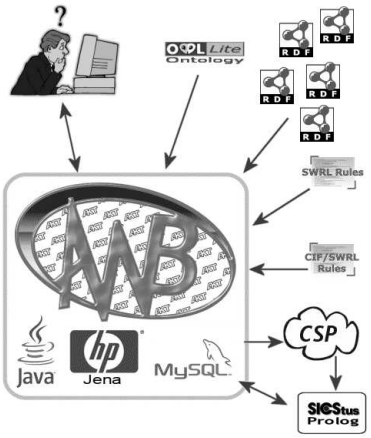

Figure 1: The architecture of the AWB.

2. ARCHITECTURE & DESIGN FACTORS

The AWB application is implemented as a Java Server Pages application with Jena[1] managing the RDF processing and some of the reasoning, and Prolog being used for the Constraint Satisfaction Problem (CSP) solving (see Figure 1). Utilising SW data has the fundamental problem of locating a datasource and ensuring that the provided information is usable in practical reasoning. Initially we had planned to access the CS AKTive Space repository directly, however the information contained within it is against an OWL Full ontology and since this has no guarantee of decidability, we were forced to 'slim down' the ontology to ensure tractable reasoning. In our early experiments the sheer volume of data (10M+ triples) also impeded our progress – due mainly to the reasoning lead-time. Therefore, as a starting point we derived an OWL Lite ontology, with a subset of cleaned-up

[1]Developed by HP Labs. http://jena.sourceforge.net

Demos and Posters of the *2nd European Semantic Web Conference (ESWC 2005)*, Heraklion, Greece, 29. May - 1. June 2005

data that was manageable, yet still realistic[2]. Even with these concessions, the time taken to gather information and perform reasoning falls outwith the time deemed as acceptable to serve a web page in real-time. To overcome this hurdle, the application was designed to create a pre-reasoned triple-cache stored in a backend database, populated prior to workgroup building. When the user runs the *build* function, it is the pre-reasoned cache information that they access. We can regard this as a 'materialised view' (including much derived data) on a distributed source.

3. DERIVATION RULES

A reasoner can derive additional ontological entailments, based upon property and class hierarchies. However, sometimes a derivation rule is required in order to infer pertinent information that cannot be determined otherwise. SWRL is used to state these rules, since it is based upon OWL Lite and DL and therefore has close ties to the underlying ontology. It allows Horn clauses to be asserted about the semantic data to create implications which we use to encode our derivation rules (making them available along with the data). We utilise these to only create new facts based on the instance data. For example, consider the following rule, used to determine the `hasBaseLocn` property: "*If a person has an affiliation with a university and that university has a postal address of a city, then this implies that the person has a base location of the same city where the university is located.*" In informal SWRL syntax, where $?x$ denotes a variable, this can be written as:

`Person(`$?p$`)` $\wedge$ `University(`$?u$`)` $\wedge$ `hasAffiliation(`$?p$`,`$?u$`)` $\wedge$
`hasAddress(`$?u$`,`$?a$`)` $\wedge$ `City(`$?c$`)` $\Rightarrow$ `hasBaseLocn(`$?p$`,`$?c$`)`

In the AWB the derivation rules are passed onto the CSP solver (translated into Prolog) along with the data, so that the rules are used only if needed, effectively in backward-chaining.

4. SEMANTIC WEB CONSTRAINTS

A gap exists in the Logic layer of the SW because there is currently no standard method for expressing fully quantified constraints against semantic data in a natural manner, explained in [2]. In the context of the AWB this means being able to pass user specified constraint information relating to the construction of a *particular* workgroup or *all* workgroups. Such a representation is important because it allows constraint information to be made available along with the data itself – potentially allowing a partially solved problem to be passed onto a solver or providing provenance information about how a solution is constrained. To express the necessary constraints, we chose to extend SWRL with our Constraint Interchange Format [1] (CIF)[3]. The AWB uses constraints defined by the user to control the construction of the workgroup. The following example shows how a constraint can be expressed using CIF/SWRL [2] and draws upon the previously specified `hasBaseLocn` property: "*Any workgroup containing at least five members must contain at least two individuals from differing sites (base locations).*"

[2]This also removes the need to address side issues relating to the management of the cache.

[3]The SWRL-FOL proposal now introduces explicit quantification although this is still under discussion and no RDF syntax is yet available. `http://www.daml.org/2004/11/fol/proposal`

This can be written as:

`(`$\forall$`g`$\in$`Workgroup) hasSize(g,s)` $\wedge$ `greaterThanOrEqual(s,5)`
$\Rightarrow$ `(`$\exists$`p1,p2`$\in$`Person) hasMember(g,p1)` $\wedge$ `hasMember(g,p2)`
$\wedge$ `hasBaseLocn(p1,b1)` $\wedge$ `hasBaseLocn(p2,b2)`
$\wedge$ `notEqual(p1,p2)` $\wedge$ `notEqual(b1,b2)`

Further examples, in RDF syntax, are given in [2].

With regard to SW reasoning, our representation is based on range restricted First Order Logic (FOL). While this takes a closed-world assumption it does not, in practice, contradict the vision of an open-world web because the context of the problem essentially closes the world at run-time.

5. DISCUSSION & CONCLUSION

For a SW application to be usable the time taken to perform reasoning must be addressed. Since the workgroup building process exceeds the acceptable threshold for rendering a web page we resort to a 'call-back message' informing the user once the solution has been found. While this implementation allows us to side-step a potential usability problem, the underlying issue still stands. The AWB has been designed to allow the exploration of the trade-offs, in practical terms, of the effectiveness of different reasoning approaches (e.g. greedy vs. lazy, forward vs. backward chaining) and this work is ongoing. The current AWB implementation attempts to reduce the problem by combining the approaches of greedy ontological reasoning (i.e. deriving all possible entailments from the ontological data and storing them in the pre-reasoned cache) and lazy rule based reasoning (i.e. mapping SWRL derivation rules to Prolog predicates and allowing the CSP to only utilise them as needed). We also have the scope for further investigation of complexity and scalability trade-offs (e.g. using an OWL DL ontology and increasing the size of the dataset). To the best of our knowledge, the AWB is unique in that no other SW application combines these various mechanisms to perform hybrid reasoning – ontological reasoning (OWL Lite), rule based reasoning (SWRL) and constraint based reasoning using our novel constraint representation CIF/SWRL.

6. ACKNOWLEDGMENTS

This work is supported under the Advanced Knowledge Technologies (AKT) IRC (EPSRC grant no. GR/N15764/01) comprising Universities of Aberdeen, Edinburgh, Sheffield, Southampton and the Open University. See `http://www.aktors.org`

7. REFERENCES

[1] K. Hui, P. Gray, G. Kemp, and A. Preece. Constraints as Mobile Specifications in e-Commerce Applications. In *Proceedings of the 9th IFIP 1.6 Working Conference on Database Semantics (DS-9): Semantic Issues in e-Commerce Systems*, pages 357–379, 2001.

[2] C. McKenzie, A. Preece, and P. Gray. Extending SWRL to Express Fully-Quantified Constraints. In G. Antoniou and H. Boley, editors, *Rules and Languages for the Semantic Web (RuleML 2004)*, pages 139–154. Springer, 2004.

[3] N. Shadbolt, N. Gibbins, H. Glaser, S. Harris, and m. schraefel. CS AKTive Space, or How We Learned to Stop Worrying and Love the Semantic Web. *IEEE Intelligent Systems*, 19(3):41–47, 2004.