Exploiting Tractable Fuzzy and Crisp Reasoning in Ontology Applications

Jeff Z. Pan, Edward Thomas, Yuan Ren and Stuart Taylor
University of Aberdeen, UK

Abstract

The Semantic Web movement has led to the publication of thousands of ontologies online. These ontologies present and mediate information and knowledge on the Semantic Web. How to provide tractable reasoning services for fuzzy and crisp ontologies has been a pressing research problem over the last five years. In this paper, we present a reusable semantic infrastructure that comprises a tractable reasoning system TrOWL, an ontological search engine ONTOSEARCH2 and a folksonomy extension component Taggr. We show that such an infrastructure can be used to support different ontology applications with tractable reasoning services by making use of tractable profiles in OWL 2 and some of their fuzzy extension.

I. Introduction

Ontologies play a key role in the Semantic Web, where the W3C recommendation OWL[1] and its successor OWL 2[2] have become the de facto standards for publishing and sharing ontologies online. Increasingly these ontologies are being used by a variety of organisations, covering the definitions of a very wide range of subjects. Fuzzy ontologies are envisioned to be useful in the Semantic Web. On the one hand, ontologies serve as a semantic infrastructure, providing shared understanding of certain domain across different applications, so as to facilitate machine understanding of Web resources. On the other hand, being able to handle fuzzy and imprecise information is crucial to the Web.

[1]http://www.w3.org/2004/OWL/

[2]http://www.w3.org/TR/owl2-overview/

Providing tractable reasoning services for OWL ontologies is not a trivial problem — OWL 1 DL has a worst-case computational complexity of NExpTime, while 2NExpTime for OWL 2 DL. This means that increasingly large ontologies require exponentially more computing resources to reason over them. Because of this, OWL 2 includes a number of tractable profiles which have combined complexity of PTIME-complete or better; including OWL 2 EL, OWL 2 QL and OWL 2 RL. However, these profiles all greatly restrict the expressive power of the language. As tool support for these profiles is still limited, it is also very easy for an ontology developer to accidentally exceed the complexity of their target profile by using a construct that is beyond the capability of that language fragment. Approximation ([1], [2], [3], [4], [5], [6], [7]) has been identified as a potential way to reduce the complexity of reasoning over ontologies in very expressive languages such as OWL 1 DL and OWL 2 DL.

On the front of fuzzy ontology reasoners, there are similar observations. There exists no fuzzy OWL 1 DL ontology reasoners that could be efficient and/or scalable enough to handle the scale of data that the Web provides. Interestingly, there currently exist two fuzzy ontology reasoners, namely the tableaux based fuzzy reasoner FiRE[3] ([8], [9]), which supports a nominal and datatype-free subset of fuzzy-OWL 1 DL, i.e. fuzzy-$\mathcal{SHIN}$, and the mixed integer programming fuzzy reasoner *fuzzyDL*[4], which supports fuzzy-OWL Lite, namely fuzzy-$\mathcal{SHI}f\ (\mathcal{D})$ [10]. Like their crisp counterparts, fuzzy-$\mathcal{SHIN}$ and fuzzy-$\mathcal{SHI}f\ (\mathcal{D})$ come with (at least) EXPTIME computational complexity but the reasoners are not as highly optimised as crisp DL reasoners, thus the scalability of the above systems is doubtful. Another approach to reason with fuzzy ontologies is to reduce to crisp ontologies, as implemented in DeLorean [11][5]. However this approach does not reduce the complexity but increases the number of axioms. Following current research developments in crisp DLs, there is an effort on lightweight fuzzy ontology languages. In particular, Straccia [12] extended the DL-Lite ontology language [13] to fuzzy DL-Lite.

In this paper, we present a reusable semantic infrastructure based on tractable OWL 2 profiles. On the one hand, the infrastructure supports a general framework [14] of query languages for fuzzy OWL 2 QL, including the support for threshold queries and general fuzzy queries. On the

[3]http://www.image.ece.ntua.gr/~nsimou

[4]http://gaia.isti.cnr.it/~straccia

[5]http://webdiis.unizar.es/~fbobillo/delorean

other hand, the infrastructure supports faithful approximate reasoning ([6], [7]) for OWL 2 DL, based on OWL 2 EL and OWL 2 QL. The rest of the article is organised as follows: In Sec. II we present some use cases of reasoning in fuzzy and crisp ontologies. In Sec. III we present the reusable reasoning infrastructure. In Sec. IV we present a scenario of using the infrastructure in one of the presented use cases in details. Finally in Sec. V we present some evaluation results on the usefulness of the infrastructure on the presented use cases.

II. Use cases

Our reusable semantic infrastructure supports a variety of reasoning and search services for different real-world application scenarios. Before we go into the details of the technologies developed for our infrastructure, we first illustrate their necessity and versatility by briefly introducing some motivating applications.

A. *Mashup*

A mashup is a kind of data-integration application, where information regarding input queries is collected from different sources, in different medias, and presented to users in a coherent manner. MusicMash2[6] is an ontology-based semantic mashup application, that is intended to integrate music related content from various folksonomy based tagging systems, linked open data[7], and music metadata Web services. MusicMash2 provides the functionality for users to search for tagged images and videos that are related to artists, albums, and songs.

An application of this nature presents two main problems. The first problem lies with the availability of populated domain ontologies on the Web. The Music Ontology (http://www.musicontology.com/) provides classes and properties for describing music on the Web; however, to instantiate the ontology, MusicMash2 must integrate music meta-data from various sources. The resulting populated ontology may be very large, so a suitable mechanism for reasoning over this ontology must be used – this mechanism must be scalable and efficient enough to handle queries on this ontology with an acceptable response time for users.

The second problem is that searching both within and across folksonomy based systems is an open problem. A naive approach to folksonomy search, such as those provided in most tagging

[6]http://www.musicmash.org/

[7]http://linkeddata.org/

systems[8], results in unacceptable precision for domain specific searches. The lack of search precision is due to the limitations of tagging systems themselves [16]. MusicMash2 addresses this problem by making use of the fuzzy reasoning services, keyword-plus-entailment search (cf. Sec. III-A) and the folksonomy search expansion service (cf. Sec. III-B) of our infrastructure.

B. Process Refinements Validation

During software development, processes are frequently modelled and refined with more and more detailed knowledge of the workflow and business logic of the system. During refinements, the activities in the abstract process are decomposed, and re-organised in the specific process. For example, the following Figure 1 (taken from [17]) shows two models of a hiring process in the standard language BPMN (Business Process Modelling Notation), where the *Specific* model refines the *Abstract* model. The original two activities $SelectApplicant$ and $HireApplicant$ are decomposed during the procedure of refinement into more fine-grained activities. And the flow structure becomes more complex with the parallel and/or exclusive gateways.

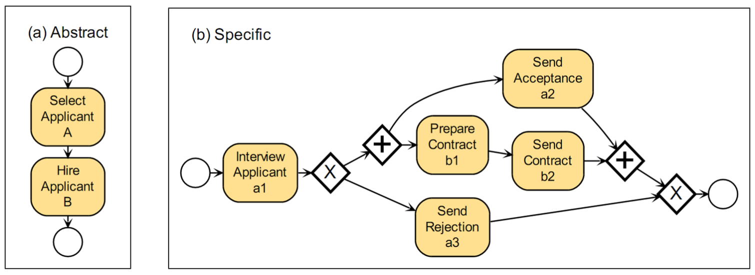


Fig. 1. A refinement step

Due to the potentially complex decomposition and re-structuring of activities, it is usually difficult to tell if the refined processes properly capture the intended behaviour of the original processes. In [17], an ontology based approach has been developed to reduce a process refinement problem to an ontology concept satisfiability/subsumption checking problem by encoding the relations between activities with ontology axioms.

This application raises an important challenge. The process refinement ontologies use the ontology language OWL 1 DL, and contain general concept inclusions (GCIs) of particular

[8]E.g. YouTube API: http://www.youtube.com/dev/

patterns. Due to the EXPTIME complexity, at the time of developing this ontological solution, mainstream reasoners such as Pellet and FaCT++ failed to provide complete classification results in a short period of time. It is crucial to develop new reasoning techniques to handle these GCIs in an efficient and quality-guaranteed manner. This requirement is fulfilled by the faithful approximate reasoning services (cf. Sec. III-C) of our tractable semantic infrastructure.

C. *Software Engineering Guidance*

The case study of physical device configuration of the MOST project [9] uses ontologies to validate the consistency of configurations of network devices. A typical configuration can be seen in Figure 2 (taken from [18]). As we can see, a configuration usually involves multiple

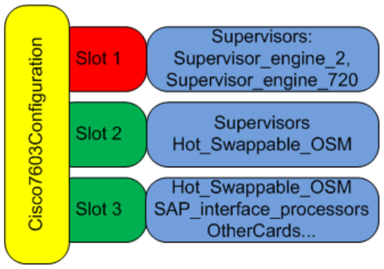


Fig. 2. A Cisco7603 Configuration

slots, and every slot can host certain types of cards. The types of cards in a same configuration also have certain dependencies and restrictions. All these constraints are encoded as axioms in ontologies as expressive as OWL 2 DL [19].

These ontologies can sometimes be inconsistent, reflecting an invalid configuration of a physical device. To understand how this is manifested in the physical device and provide guidance on how it may be resolved, it is necessary to perform reasoning efficiently in modelling time, and to find justifications for the inconsistency. Furthermore, in order to provide suggestions to engineers, some of the concepts or properties in the ontology need to be closed. Traditional ontology reasoning imposes the Open World Assumption and does not support such services. Closed domain reasoning services are required to address this issue. This requirement is eventually

[9]www.most-project.eu/

fulfilled by the support of NBox (Negation as failure Box [20]) and approximate reasoning services (cf. Sec. III-C) in our tractable semantic infrastructure.

To conclude this section, we summarise the challenges to existing reasoning technologies that are raised by the above use cases:

1) The scalability and efficiency of querying on large scale populated domain ontologies.
2) The precision of folksonomy search, especially for vague and/or ambiguous terms.
3) The efficiency of reasoning in expressive and very expressive ontology languages.
4) The availability of local closed world reasoning services.

In the rest of the article, we will present our proposed solutions to these challenges.

III. A Tractable Semantic Infrastructure

In this section, we present a tractable semantic infrastructure. Our architecture consists of three layers of semantic tools and they can all be accessed by external applications via APIs. The system and its interaction with the use cases are illustrated in the Figure 3.

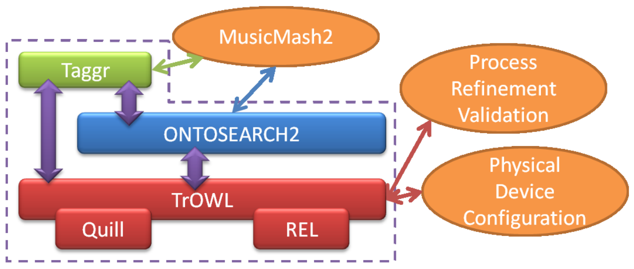


Fig. 3. Semantic infrastructure and its interaction with use cases

TrOWL[10] [21] is a tractable reasoning infrastructure of OWL 2. It includes a built-in OWL 2 QL reasoner Quill and a built-in OWL 2 EL reasoner REL. It has been continuously extended with new reasoning services, such as approximation ([6], [7]), forgetting [22], justification, local closed world reasoning (NBox) [20] and stream reasoning ([23], [24]). It can directly support the process refinement validation and physical device configuration use cases.

[10]http://trowl.eu/

ONTOSEARCH2[11] ([25], [26]) is an ontological search engine that is based on the TrOWL ontology infrastructure. It supports keyword-plus-entailment search by making use of fuzzy and crisp reasoning services from TrOWL.

Taggr [27] is a folksonomy extension component which has been built on ONTOSEARCH2. MusicMash2 uses both the ONTOSEARCH2 and Taggr.

The architecture includes an ontology repository (as part of TrOWL), where users can submit and share ontologies and a query rewriter that rewrites user queries submitted in SPARQL so that they can be executed on the repository. A useful feature of the repository is that it automatically associates keywords (in values of annotation properties as well as implicit metadata in target ontologies) with concepts, properties and individuals in the ontologies. Default annotation properties include the `rdfs:label`, `rdfs:comment`, `rdfs:seeAlso`, `rdfs:isDefinedBy`, and `owl:versionInfo` properties; we also define the `dc:title`, `dc:description`, and `foaf:name` properties (from Dublin Core[12] and FOAF[13]) as annotation properties. Implicit metadata is drawn from the namespace and ID of each artefact in the ontology.

These keywords are weighted based on ranking factors similar to those used by major search engines[14]. The system uses these scores to calculate the $tf \cdot idf$ [28] for each keyword found within the ontology, and normalises them using a sigmoid function such as the one shown in (1) to a degree between 0 and 1.

$$w(n) = \frac{2}{1.2^{-n} + 1} - 1 \tag{1}$$

In Sec. III-A, we first introduce how the knowledge and their degree values can be used to provide fuzzy reasoning services with ONTOSEARCH2. Then in Sec. III-B we show how to use Taggr to extend the search services to folksonomies. Finally we present the details of TrOWL in Sec. III-C to show how the underlying crisp reasoning services are accomplished.

A. *Fuzzy Reasoning Services*

Being able to handle fuzzy and imprecise information is crucial to the Web. TrOWL also consists of a query engine for fuzzy OWL 2 QL [14]. The query engine supports threshold

[11]http://www.ontosearch.org/

[12]http://dublincore.org/

[13]http://www.foaf-project.org/

[14]http://www.seomoz.org/article/search-ranking-factors

queries and general fuzzy queries over fuzzy OWL 2 QL ontologies. Users of the query engine can submit fuzzy OWL 2 QL ontologies via the Web interface of ONTOSEARCH2[6], and then submit f-SPARQL [14] queries, such as the following one, against their target ontologies.

```
#TQ#
PREFIX music: <http://musicmash.org/NS/>
SELECT ?x WHERE {
  ?x a music:MusicArtist .
  ?x a music:Popular . #TH# 0.7
  ?x a music:Active .  #TH# 0.8
}
```

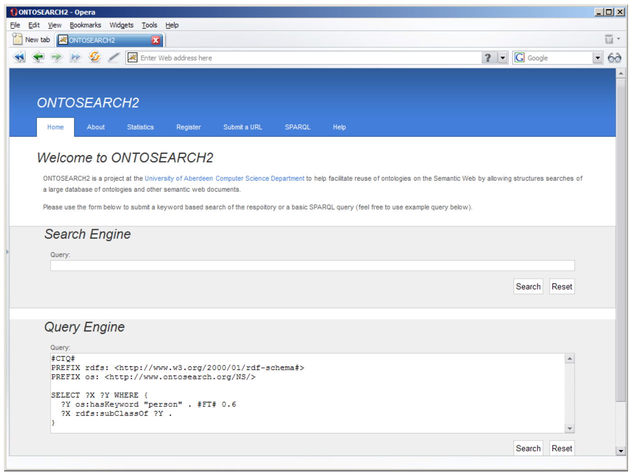
where #TQ# declares a threshold query, while #TH# is used to specify thresholds for atoms in the query. Therefore this query searches for an instance of `MusicArtist` which is a member of the class `Popular` with degree `0.7`, and a member of the class `Active` with degree `0.8`.

Preliminary evaluations show that performance of the fuzzy OWL 2 QL query engine is in most cases close to the performance of the crisp OWL 2 QL query engine [14].

ONTOSEARCH2 is an ontological search engine that allows users to search its repository with keyword-plus-entailment searches, such as *searching for ontologies in which class X is a sub-class of class Y, and class X is associated with the keywords "Jazz" and "Rock", while class Y is associated with the keyword "Album"*. The search could be represented as the following threshold query:

```
#TQ#
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX os: <http://www.ontosearch.org/NS/>
PREFIX kw: <http://www.ontosearch.org/KW/>

SELECT ?x WHERE {
 ?x os:hasKeyword kw:jazz . #TH# 0.5
 ?x os:hasKeyword kw:rock . #TH# 0.7
 ?x rdfs:subClassOf ?y .
 ?y os:hasKeyword kw:album . #TH# 0.8}
```



Welcome to ONTOSEARCH2

ONTOSEARCH2 is a project at the University of Aberdeen Computer Science Department to help facilitate reuse of ontologies on the Semantic Web by allowing structures searches of a large database of ontologies and other semantic web documents.

Please use the form below to submit a keyword based search of the respoitory or a basic SPARQL query (feel free to use example query below).

Search Engine

Query:

Query Engine

Query:

```
#CTQ#
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX os: <http://www.ontosearch.org/NS/>

SELECT ?X ?Y WHERE {
  ?Y os:hasKeyword "person" . #FT# 0.6
  ?X rdfs:subClassOf ?Y .
}
```

Fig. 4. ONTOSEARCH2 Screenshot

where kw:jazz, kw:rock, and kw:album are representative individuals for keywords “jazz”, “rock”, and “album”, respectively. The thresholds 0.5, 0.7, and 0.8 can be specified by users. The keyword-plus-entailment searches are enabled by the fuzzy DL-Lite query engine as well as the semantic approximation components. Such services allow both TBox and ABox queries.

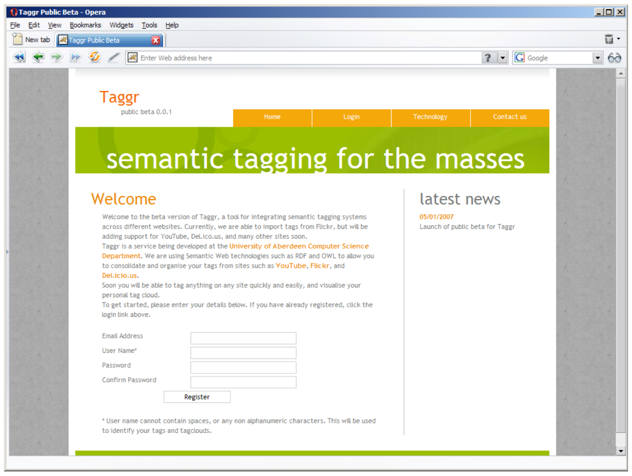
B. *Relating Folksonomies to Ontologies*

The *Taggr* system provides a simplified interface to ONTOSEARCH2 to perform useful operations that are related to folksonomy based systems. It stores a basic ontology (which we refer to as the “tagging database”) in the TrOWL repository, capturing the relationships between users, tags and resources in the folksonomy based systems it supports[15].

Taggr allows users to provide a set of arbitrary resources and related tags to be added to the tagging database in the TrOWL repository. A Web service and a traditional Web interface are provided so that users can interact with the tagging database without having to understand the internal representation used by the system.

Taggr also provides an ontology-enabled common interface for folksonomy based systems. It provides the functionality to gather resources and their related tags from the systems that it supports, and populate them to its tagging database from time to time. In case an application

[15]Taggr currently supports the YouTube and Flickr tagging systems.



Taggr
public beta 0.0.1

semantic tagging for the masses

Welcome

Welcome to the beta version of Taggr, a tool for integrating semantic tagging systems across different websites. Currently, we are able to import tags from Flickr, but will be adding support for YouTube, Del.ico.us, and many other sites soon.
Taggr is a service being developed at the University of Aberdeen Computer Science Department. We are using Semantic Web technologies such as RDF and OWL to allow you to consolidate and organise your tags from sites such as YouTube, Flickr, and Del.icio.us.
Soon you will be able to tag anything on any site quickly and easily, and visualise your personal tag cloud.
To get started, please enter your details below. If you have already registered, click the login link above.

Email Address
User Name*
Password
Confirm Password

* User name cannot contain spaces, or any non alphanumeric characters. This will be used to identify your tags and tagclouds.

latest news

05/01/2007
Launch of public beta for Taggr

Fig. 5. Taggr Beta Screenshot

requests some resources that Taggr does not know about, it can simply make a call to Taggr to request that it updates its tagging database with resources related to the search.

Furthermore, Taggr allows users to specify which search expansion method(s) [27] and which reference ontology(-ies) to use for the expansion. The extended search will first be evaluated against its tagging database. As the ontological constraints needed for the search expansions require only the expressive power of OWL 2 QL, Taggr can make use of the semantic approximation(s) of the reference OWL DL ontology(-ies) for all entailment checking. Due to the logical properties of semantic approximation, TrOWL can provide sound and complete results for all the needed entailment checking. Details of the search expansion methods go beyond the scope of this paper.

C. *Crisp Reasoning Services*

TrOWL is a tractable reasoning infrastructure for OWL 2. A syntax checker front-end will briefly examine the syntax of the loaded ontology and decide which built-in reasoners should be used, based on the configuration of the system. There are two major built-in reasoners. Quill provides reasoning services over RDF-DL and OWL 2 QL and REL provides reasoning over OWL 2 EL. Besides, TrOWL also supports full OWL 2 DL reasoning using a plug-in reasoner such as Pellet or FaCT++. The OWL 2 DL reasoning will have the same quality as the plug-in reasoner. Alternatively, TrOWL can also use the approximation facilities provided by REL to deliver sound and practically complete reasoning for OWL 2 DL. Such separation of reasoners make it possible to provide tailored and optimised services, instead of a "one-size-fit-all" reasoner.

1) Language Transformations: The transformation from OWL 2 DL to OWL 2 QL is based around the Semantic Approximation technology from OWL DL to DL-Lite [6]. This technology can be easily applied on OWL 2 DL as well. Semantic Approximation adopts the idea of knowledge compilation to pre-compute the materialisation of an OWL 2 DL ontology into OWL 2 QL axioms, by using a heavyweight reasoner. Because the semantics of OWL 2 QL are a subset of, and are hence compatible with, the direct semantics OWL 2 DL, any reasoning results against the approximated OWL 2 QL ontology are always sound with regards to the original OWL 2 DL ontology. Furthermore, it has been shown that for conjunctive query answering, which is the strength of the QL language, results against the semantic approximation are also complete for queries with no non-distinguished variables, or with non-distinguished variables only in leaf nodes of the query. This already covers a majority of conjunctive queries, including the queries supported by the new SPARQL specification.

The transformation from OWL 2 DL to OWL 2 EL is based on the soundness preserving approximate reasoning approach [7], [29]. This approach represents non-OWL 2 EL concept expressions with fresh named concepts, non-OWL 2 EL role expressions with fresh named roles, and then maintaining their semantics, such as complementary relations, inverse relations, cardinality restrictions, etc. in separate data structures. In the reasoning stage, additional completion rules have been devised to restore such semantics. With this approach, the overall complexity for OWL 2 DL ontologies can be reduced to PTime. Although known to be incomplete, evaluation shows that, such transformation can classify existing benchmarks very efficiently with high recall (over 95%) [7], [29].

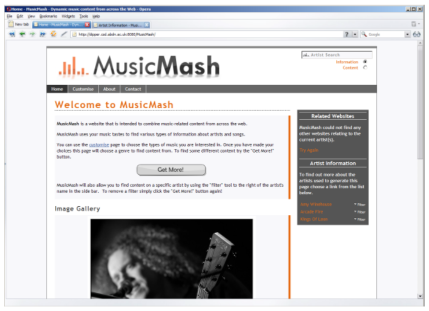
Except for the improved performance of standard reasoning services, the language transformations also enable many other non-standard reasoning services. For example, the support of local closed world reasoning can be reduced to incremental reasoning by restricting the extensions of concepts, and the domains and ranges of properties, which can be further realised more efficiently by transformation to OWL 2 EL as OWL 2 EL has PTIME incremental reasoning capacity. In TrOWL, such closed world reasoning is realised by first specifying an NBox (Negation As Failure Box) in the ontology [20]. An NBox is a set of concept and property expressions. Any predicate (concept or property) in the NBox will be closed.

2) Lightweight Reasoners: The Quill reasoner implements a novel and unique database schema for storing normalised representations of OWL 2 QL ontologies. Any conjunctive query can be rewritten into a single, simple SQL query over the underlying database, using the database itself to perform the transitive completion of class and property subsumption with an innovative exploitation of the highly optimised database mechanisms. To support this, new query rewriting and ontology normalisation algorithms have been developed. In its initial testing across large knowledge bases with deep concept hierarchies, such as the DBPedia dataset and the Yago ontology, a significant performance improvement over other DL-Lite query engines is observed. In an extreme case, using the standard query rewriting algorithm PerfectRef over a deep class or property hierarchy can result in a set of hundreds or thousands of conjunctive queries, where the new method only ever results in a single query. Quill supports all reasoning tasks for OWL 2 QL, including consistency and satisfiability checking, and query answering, and by using an OWL 2 DL reasoner it can perform semantic approximation of more expressive ontologies.

The REL reasoner [7] is a Java implementation of an OWL 2 EL reasoner. It extends the EL+ algorithm [30] with the completion rules for OWL 2 EL [31] and further optimises them. On top of that, the syntactic approximation and reasoning rules are also implemented. This allows REL to provide tractable reasoning for OWL 2 EL ontologies and make up the core component of the soundness-preserving syntactic approximation for OWL 2 DL ontologies. REL is an approximate OWL 2 DL reasoner with high general recall; it is complete for all the HermiT benchmark ontologies. In additional, REL also consists of an OWL 2 EL conjunctive query engine [32], which allows queries over OWL 2 EL ontologies to be answered more efficiently without semantic approximation. Non-standard reasoning services such as local closed world reasoning is also supported by REL. By integrating the idea of truth maintenance systems, REL also supports justification and ontology stream reasoning.

IV. Scenario: Mashing Linked Music Data

In this section, we describe a concrete scenario illustrating how the semantic infrastructure (presented in the previous section) could enhance a typical Web 2.0 application. Let us consider the story of “Sarah”, a keen Web developer with an interest in Web 2.0 and new Web technologies in general.



MusicMash is a website that is intended to combine music-related content from across the web.

MusicMash uses your music tastes to find various types of information about artists and songs.

You can use the customise page to choose the types of music you are interested in. Once you have made your choices this page will choose a genre to find content from. To find some different content try the "Get More!" button.

MusicMash will also allow you to find content on a specific artist by using the "filter" tool to the right of the artist's name in the side bar. To remove a filter simply click the "Get More!" button again!

MusicMash could not find any other websites relating to the current artist(s).

To find out more about the artists used to generate this page choose a link from the list below.

Fig. 6. Fig. 6. MusicMash 1.0 Screenshot: The original version of MusicMash which does not use any Semantic Web components.

A. A Web 2.0 Application

Sarah's interest in Web 2.0 had grown from her interest in music. She enjoys using Web 2.0 websites to find multimedia content about her favourite artists. However, she found that it was inconvenient to have to browse many different websites to find content related to a single artist. Meanwhile, she had been reading about Web Services and Mashup as part of her interest in Web development. After reading a few articles on the Web, Sarah decided that she could address this inconvenience by building her own mashup application. The objective of this application is to combine music-related resources – videos, images, biographies and discographies – into a single website. Sarah also decided that if her application were to be of use, it would have to provide accurate search results in a timely fashion. Sarah named her new web application "MusicMash" and began work on the project.

B. Searching Folksonomies

To retrieve video and image content for her new site, Sarah made use of the public Web Service APIs provided by YouTube and Flickr. She quickly developed the first prototype. This version allowed users to perform searches based on an artist's name. MusicMash used the artist's name when making calls to the YouTube and Flickr APIs to retrieve videos tagged with the each word from the artist's name.

Sarah soon found that this early version of MusicMash suffered from a major drawback – that artists' names are often ambiguous search terms in YouTube and Flickr. For example, when she searched for Focus (the Dutch Progressive Rock band), only 5 out of the top 20 results returned by YouTube were relevant to that artist. Sarah noticed that users on YouTube often tagged music videos with both the artist's name and song title. She soon realised that including a song title with the artist's name generated more relevant search results.

Sarah then extended MusicMash to populate a database of music metadata retrieved from the MusicBrainz[16] web service. Using this metadata, Sarah could extend MusicMash to automatically expand a simple artist's name search, to an *artist's name plus song title* search, for each song by that artist. This search expansion technique resulted in an impressive increase in the precision of the search results. However, the volume of API calls needed for the search expansion resulted in an unacceptable amount of time required to return search results..

C. The Switch to Taggr

The performance issues in MusicMash were due to the large number of HTTP requests to Web Service APIs generated by Sarah's search expansion technique. The solution to these issues is to ensure that the APIs perform the search expansion internally hence needing only one HTTP request. Sarah learned of a system called Taggr that provides the same search expansion functionality as MusicMash via a Web Service API. Sarah decided to redevelop MusicMash using the Taggr API, rather than accessing YouTube and Flickr directly.

The Taggr API allows Sarah to input the original search term(s) from the user and some extra parameters to specify how the search expansion should be performed. More specifically, the parameters indicate whether video or image resources results should be returned; what the input search term(s) should identify, S (a music artist in this case); where to find the extra keywords for the search expansion, T (a song title in this case); and how S and T are related, P (a music artist is the creator of a song). Taggr uses OWL DL ontologies to represent its metadata internally. The parameters S and T should be specified as OWL classes and P as an OWL property. However, Sarah did not know which OWL class was used to identify music artists and song titles on the Web. She follow a link from the Taggr website to ONTOSEARCH2. She then made use of ONTOSEARCH2's ontology search engine to find out which ontologies contained resources relating to music. Sarah typed "music" into the ONTOSEARCH2 search engine and one of the first results returned was from the Music Ontology[17]. After further investigation Sarah found

[16]http://www.musicbrainz.org/

[17]http://purl.org/ontology/mo/

that the Music Ontology was exactly what she needed to describe the classes and properties for music artists, albums and songs.

Sarah then set about trying some searches on Taggr using the concepts `mo:MusicArtist` and `mo:Track`, related via the property `mo:foaf:made`; allowing her to replicate the search expansion from MusicMash. She first used Taggr to check for new keywords that were generated by its search expansion. Sarah tried the keyword "Coldplay" and was surprised to see that Taggr did not provide any new keywords. She then searched ONTOSEARCH2 directly for "Coldplay" and again, no results were returned. Sarah realised that she would have to provide the Music Ontology individuals herself in order for the search expansion to work correctly.

D. *From Relational Databases to Ontologies*

Since ontology individuals are required by Taggr to replicate the MusicMash search expansion. Sarah decided to drop her database of music metadata in favour of Music Ontology instances stored in the ONTOSEARCH2 repository. ONTOSEARCH2's submission and query engine provided the tools that she needed to insert new individuals and query against them. Sarah decided to populate her ontology using Web Services that can be easily linked. More specifically, using MusicBrainz API for basic artist, album and song information, she could extend the metadata with other sources that referred to MusicBrainz identifiers, such as Last.fm and DBpedia. This new version of MusicMash was named MusicMash2.

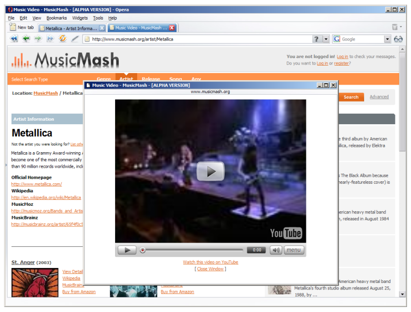


Fig. 7. MusicMash2 Screenshot: The second version of MusicMash built using our infrastructure.

The final two problems left for Sarah to address are when there is no Music Ontology instances relating to a user's search or where there were insufficient resources returned by Taggr. She decided that for any search for which MusicMash2 did not immediately return more than five results to the user, a request would be made to Taggr to populate its tagging database with more resources from tagging systems in its library. Taggr would then send requests to its supported tagging systems to retrieve the first 500 results based on the original search term(s). Similarly when MusicMash2 returns no individuals in the Music Ontology relating to the search, it initiates a background task to retrieve the required information from the Web services in its library. The advantage of this method is that information relating to previously unknown artists can now be added automatically to the Music Ontology in ONTOSEARCH2 and Tagging Database in Taggr. Sarah decided that it was an acceptable trade-off that the first user wait for the information to be retrieved, in order for future searches to be returned in a more acceptable amount of time.

V. Usefulness Evaluation

1) Mashup: A typical scenario for MusicMash2 can be illustrated by a user searching for information related to an artist. The user first enters the name of the artist into the search box. On completion of a successful search, MusicMash2 displays the related artist information to the user. This includes a short abstract from DBpedia, the artists discography and links to the artist's homepage and Wikipedia articles. The user can also select the Video Gallery tab to display videos relating to the current artist. The Video Gallery makes use of Taggr to return high precision search results for related videos. An example of an artist's page can be viewed at http://www.musicmash.org/artist/Metallica. In [27], a usefulness evaluation is conducted to compare the YouTube search engine and MusicMash2. 15 artists' names are used as search terms (see Table I), and the presison of the top 20 results for each search term returned by the two systems is illustrated in Figure 8.

Figure 8 shows that in all but one search, Taggr outperformed YouTube in terms of precision. YouTube returned good results for less ambiguous artist names (searches #1 - 4) but poor results for ambiguous names (searches #5 - 15); i.e., where the artist's name can correspond to many non-music related videos. In searches #13 and #14, Taggr loses some precision for the more ambiguous artist's names "Yes" and "Kiss". In the final search, #15, Taggr performs slightly worse than YouTube, by 15%. The reason for the loss of precision could be because there were instances where the artist's name and song title combinations are themselves ambiguous, and in some cases less effective than YouTube's built-in search facilities. A simple approach to addressing this issue would be to attempt to rank the results by examining the complete set of

TABLE I

ARTIST NAMES USED IN THE EVALUATION

Search #	Search Term	Search #	Search Term
1	The Beatles	9	Thursday
2	Porcupine Tree	10	Pelican
3	Radiohead	11	Focus
4	Metallica	12	Isis
5	Eels	13	Yes
6	Doves	14	Kiss
7	Finch	15	Jet
8	Strung Out		

tags for a resource and checking how many are actually matched by the keywords generated in the search expansion.

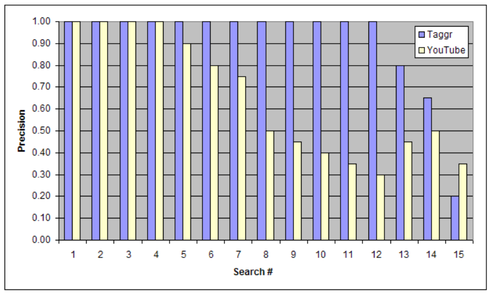


Fig. 8. Result set precision for *top k* search.

2) Process Refinement Validation: The usefulness on process refinement validation can be examined by comparing the utility of the ontological solution with other solutions, and also comparing the performance of the optimisation of approximation with off-the-shelf reasoners. The completeness and soundness of the solution by using approximation has been theoretically proved in [17]. In [33] a comprehensive empirical study is conducted by SAP to test the usefulness and performance of the validation methodology and reasoning facilities. The usefulness evaluation shows that, with the overall solution, a process developer can become about 3 times faster than before (productivity) and achieve 1.6 times their original correctness rate (quality). A further performance comparison between REL and Pellet suggested that, when both using ontologies to validate process refinement, the syntactic approximation-based REL solution is significantly faster than the off-the-shelf reasoner Pellet, and more scalable when increasing the length, branching and parallelisms of process models. When compared to non-ontological solutions such as the Petri Net, REL becomes less scalable and efficient when the parallelism of the processes are increased, due to the factorial increase of possible executions of processes caused by parallelism. However, Petri Net is only capable of identifying the 1st invalid activity while an ontology-based solution can find all of them in one go.

3) Software Engineering Guidance: The technologies developed in the TrOWL have also been used to support the ontological solution of the physical device configuration case study. Its usefulness can be evaluated by comparing it to the pure model-based state-of-the-art approach used in companies. In [34] COMARCH conducted an empirical evaluation and the results suggest that by using the ontology-integrated solution, the development time is reduced by 76% and bug-fixing time is reduced by 100%, meaning that no error was found in the modelling with the ontological solution. This correctness result is also formally proved in [19], which shows that despite the high expressive power and computational complexity of the physical device ontology, the completeness and soundness of using syntactic approximation implemented in TrOWL can be guaranteed for the considered types of tasks.

VI. CONCLUSION

In this paper, we have presented a novel tractable semantic infrastructure based on the OWL 2 profiles. The infrastructure provides tractable fuzzy and crisp ontology reasoning services, as well as keyword-plus-entailment search services and tailored support for folksonomy systems, validation services for business process refinement and guidance services for ontology driven software development.

By combining open ontologies with information retrieved from proprietary knowledge bases, we increased the access to this information through open interfaces. Since ONTOSEARCH2, a front end of our infrastructure, is publicly accessible through a standardised interface (SPARQL), it is possible for other applications to be built on top of the ontologies generated by existing applications, such as MusicMash2. Hence, we provide an infrastructure that may stimulate further development and/or population of domain ontologies.

ACKNOWLEDGMENT

The case studies, implementations and evaluations presented in this work have been partially funded by the EU project MOST. The authors would like to thank our MOST colleagues Yuting Zhao, Andreas Friesen, Jens Lemcke, Tirdad Rahmani and Krzysztof Miksa.

REFERENCES

[1] H. Stuckenschmidt and F. van Harmelen, "Approximating Terminological Queries," in *Proc. of FQAS2002)*, 2002, pp. 329–343.

[2] H. Wache, P. Groot, and H. Stuckenschmidt, "Scalable Instance Retrieval for the Semantic Web by Approximation," in *Proc. of WISE-2005 Workshop on Scalable Semantic Web Knowledge Base Systems*, 2005.

[3] P. Hitzler and D. Vrandecic, "Resolution-Based Approximate Reasoning for OWL DL," in *Proc. of the 4th International Semantic Web Conference (ISWC2005)*, 2005.

[4] P. Groot, H. Stuckenschmidt, and H. Wache, "Approximating Description Logic Classification for Semantic Web Reasoning," in *Proc. of ESWC2005*, 2005.

[5] C. Hurtado, A. Poulovassilis, and P. Wood, "A Relaxed Approach to RDF Querying," in *Proc. of the 5th International Semantic Web Conference (ISWC-2006)*, 2006.

[6] J. Z. Pan and E. Thomas, "Approximating OWL-DL Ontologies," in *Proceedings of the 22nd Conference on Artificial Intelligence (AAAI-07)*, 2007, pp. 1434–1439.

[7] Y. Ren, J. Z. Pan, and Y. Zhao, "Soundness Preserving Approximation for TBox Reasoning," in *the Proc. of the 25th AAAI Conference Conference (AAAI2010)*, 2010.

[8] G. Stoilos, G. Stamou, J. Pan, V. Tzouvaras, and I. Horrocks, "Reasoning with very expressive fuzzy description logics," *Journal of Artificial Intelligence Research*, vol. 30, no. 5, pp. 273–320, 2007.

[9] G. Stoilos, N. Simou, G. Stamou, and S. Kollias, "Uncertainty and the semantic web," *Intelligent Systems, IEEE*, vol. 21, no. 5, pp. 84–87, 2006.

[10] G. Stoilos, G. Stamou, V. Tzouvaras, J. Pan, and I. Horrocks, "Fuzzy owl: Uncertainty and the semantic web," in *Proc. of the International Workshop on OWL: Experiences and Directions*, vol. 280. Citeseer, 2005.

[11] F. Bobillo, M. Delgado, and J. Gómez-Romero, "Delorean: A reasoner for fuzzy owl 2," *Expert Systems with Applications*, 2011.

[12] U. Straccia, "Answering vague queries in fuzzy dl-lite," in *Proceedings of the 11th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems,(IPMU-06)*. Citeseer, 2006, pp. 2238–2245.

[13] D. Calvanese, G. D. Giacomo, D. Lembo, M. Lenzerini, , and R. Rosati, "DL-Lite: Tractable description logics for ontologies," in *Proc. of AAAI 2005*, 2005.

[14] J. Z. Pan, G. Stamou, G. Stoilos, S. Taylor, and E. Thomas, "Scalable Querying Services over Fuzzy Ontologies," in *the Proc. of the 17th International World Wide Web Conference (WWW2008)*, 2008.

[15] G. Acampora, V. Loia, and M. Gaeta, "Exploring e-Learning Knowledge Through Ontological Memetic Agents." *IEEE Comp. Int. Mag.*, vol. 5, no. 2, pp. 66–77, 2010.

[16] A. Passant, "Using Ontologies to Strengthen Folksonomies and Enrich Information Retrieval in Weblogs," in *Proc. of 2007 International Conference on Weblogs and Social Media (ICWSM2007)*, 2007.

[17] Y. Ren, G. Gröner, J. Lemcke, T. Rahmani, A. Friesen, Y. Zhao, J. Z. Pan, and S. Staab, "Process refinement validation and explanation with ontology reasoning," University of Aberdeen, University of Koblenz-Landau and AP AG, Tech. Rep., 2011. [Online]. Available: http://www.abdn.ac.uk/~csc280/pub/ProcessRefinement.pdf

[18] Y. Zhao, C. Wende, J. Z. Pan, E. Thomas, G. Groener, N. Jekjantuk, Y. Ren, and T. Walter, "D3.4 -guidance tools for language transformations," MOST Project, Project Deliverable ICT216691/UNIABDN/WP3-D4/D/PU/b1, 2010.

[19] Y. Ren, "Syntactic approximation in pddsl: A completeness guarantee," University of Aberdeen, Tech. Rep., 2010, http://www.abdn.ac.uk/ csc280/TR/pddsl.pdf.

[20] Y. Ren, J. Z. Pan, and Y. Zhao, "Closed World Reasoning for OWL2 with NBox," *Journal of Tsinghua Science and Technology*, no. 6, 2010.

[21] E. Thomas, J. Z. Pan, and Y. Ren, "TrOWL: Tractable OWL 2 Reasoning Infrastructure," in *the Proc. of the Extended Semantic Web Conference (ESWC2010)*, 2010.

[22] Z. Wang, K. Wang, R. Topor, and J. Z. Pan, "Forgetting Concepts in DL-Lite," in *the Proc. of the 5th European Semantic Web Conference 2008 (ESWC2008)*, 2008.

[23] Y. Ren, J. Z. Pan, and Y. Zhao, "Towards Scalable Reasoning on Ontology Streams via Syntactic Approximation," in *the Proc. of the ISWC2010 Workshop on Ontology Dynamics (IWOD2010)*, 2010.

[24] Y. Ren and J. Z. Pan, "Optimising Ontology Stream Reasoning with Truth Maintenance System," in *the Proc. of the ACM Conference on Information and Knowledge Management (CIKM 2011)*, 2011.

[25] J. Z. Pan, E. Thomas, and D. Sleeman, "ONTOSEARCH2: Searching and Querying Web Ontologies," in *Proc. of WWW/Internet 2006*, 2006, pp. 211–218.

[26] E. Thomas, J. Z. Pan, and D. Sleeman, "ONTOSEARCH2: Searching Ontologies Semantically," in *Proc. of OWL: Experiences and Directions (OWL-ED2007)*, 2007.

[27] J. Z. Pan, S. Taylor, and E. Thomas, "Reducing Ambiguity in Tagging Systems with Folksonomy Search Expansion," in *Proc. of the 6th European Semantic Web Conference 2008 (ESWC2009)*, 2009.

[28] G. Salton and M. J. McGill, *Introduction to modern information retrieval.* McGraw-Hill, 1983.

[29] Y. Ren, J. Z. Pan, and Y. Zhao, "Towards soundness preserving approximation for abox reasoning of owl2," in *Description Logics Workshop 2010 (DL2010)*, 2010.

[30] F. Baader, C. Lutz, and B. Suntisrivaraporn, "Is Tractable Reasoning in Extensions of the Description Logic EL Useful in Practice?" in *Proceedings of the 2005 International Workshop on Methods for Modalities (M4M-05)*, 2005.

[31] F. Baader, S. Brandt, and C. Lutz, "Pushing the $\mathcal{EL}$ Envelope," in *Proc. of the 19th Int. Joint Conf. on Articial Intelligence (IJCAI 2005)*, 2005.

[32] Y. Zhao, J. Z. Pan, and Y. Ren, "Implementing and evaluating a rule-based approach to querying regular el+ ontologies," in *In Proc. of the International Conference on Hybrid Intelligent Systems (HIS2009)*, 2009.

[33] A. Friesen, J. Lemcke, D. Oberle, and T. Rahmani, "D6.4 - evaluation of case-study," MOST Project, Project Deliverable ICT216691/SAP/WP6-D4/D/RE/b1, 2010.

[34] M. Kasztelnik, K. M. Miksa, and P. Sabina, "D5.4 - evaluation of case-study," MOST Project, Project Deliverable ICT216691/CMR/WP5-D4/D/RE/b1, February 2010.