Good report on ontologies paper

Technology, Internet

Abstract

An ontology refers to a formal, common, shared description of significance concepts within a specific domain. Ontologies can be used to support various tasks in diverse areas of research such as information retrieval, knowledge management, databases, natural language processing, “ geographic information systems”, digital libraries, “ multi agent systems” or “ visual information retrieval systems”. An ontology makes it possible to have shared knowledge as well as reuse where there can be communicating of information resources between software and human agents. However, ontologies have conceptually been in development since the 1970’s and in the late 1990s’-early 2000’s, we began seeing languages developed to automate these ontologies process (RDF, RDFS, OWL). Despite all these, we haven’t yet seen widespread ontology use for semantic web applications.

In this paper, there is considering of RDF, RDFS and OWL DL, in which it is found that, there are some main differences in regard to expressive power, among them. The RDF(S) are found to be less expressive as compared with OWL DL. The paper presents an extensive review of other ideas including FOAF/SPARQL/TURTLE and the shortcomings or challenges. The paper presents approaches from research articles and talks about SADI. Moreover, there is a discussion about the causes of ontomological application “ slowdown” and there is a discussion about findings of automatic semantic extraction and the success of SADI, in regard to implementation.

Introduction

What are Ontologies?

There is possibility of finding in the literature a number of ontologies definitions. Among the most cited definitions is the one that was proposed by Gruber, where an ontology is defined as “ a formal, explicit, specification of a shared conceptualization” (Gruber 1993: 199). This definition is analyzed by Fensel by making identification of the four key concepts involved: an accurate mathematical account hints the term “ formal”, the accuracy of concepts as well as their relationships defined clearly are expressed by the word “ explicit”, the presence of an agreement among ontology users is given a hint by the word “ shared”, and “ an abstract model of a phenomenon termed “ conceptualization” (Fensel 2000: 70).

Gruber’s proposed definition is general; however, the ontologies can be given a definition within specific contexts. For instance, considering the agents paradigm, it is established that “ an ontology is a formal description of the concepts and relations which can exist in a community of agents” (Russell and Norving 2010: 41). The significance of an ontology’s terms can be seen in the following definition: “ an ontology is a hierarchical structured set of terms to describe a domain that can be used as a skeletal foundation for a knowledge base” (Noeto 2003: 4).

An ontology essentially encompasses a specification of terms used and agreements to undertake determination of the meaning of such terms, together with relationships that exist between them (Starlab 2011: 14). From the definition, some important features of ontologies can be identified. For instance, it can be seen that ontologies are utilized in describing a specific domain. Moreover, the relations and terms are defined clearly in such a domain. There is also a mechanism used for organizing terms, “ commonly a hierarchical structure is used as well as IS-A or HAS-A relationships” (Noeto 2010: 4). In addition, an agreement exists between ontology users in such a manner that there is consistent use of the terms’ meanings.

Uses of Ontologies

There can be utilization of ontologies to support various tasks in diverse areas of research such as information retrieval, knowledge management, databases, natural language processing, “ geographic information systems”, digital libraries, “ multi agent systems” or “ visual information retrieval systems” (Noeto 2010: 6).

An ontology offers to meta information that describes the data semantics (Fensel 2000: 70). Ontologies make it possible to have shared knowledge as well as reuse where there can be communicating of information resources between software and human agents. The semantical links in ontologies are found to be machine readable, in manner that they make it possible to make statements, “ and asking queries about a subject domain due to the use of a conceptualization, which describes entities and their relationships” (Noeto 2010: 6). This conceptualization, allows that vocabulary’s software agents to represent as well as to communicate knowledge. There can be brief summarizing of the ontologies’ usefulness in the agent-based systems as they allow knowledge-level interoperability. In some other areas of research, ontologies support mutual understanding, systems engineering, interoperability between particular tools, declarative specification and reusability (Farquhar et al. 1997: 707).

Conversely, there is utilizing of ontologies in building knowledge bases. An ontology as well as a set of its classes' individual instances form a knowledge base. Agents query knowledge bases in order to develop, use again, as well as maintain them. Ontologies serve to " concentrate state-independent information while the core of knowledge bases is formed by state-dependent information" (FIPA 2001: 1).

Ontologies are capable of operating as repositories in organizing information for some specific communities. There can be using of ontologies for the acquisition of knowledge. For instance, team works can use them as a common support to undertake classification of an organization's knowledge. They enable users to reutilize knowledge in fresh systems. Ontologies can play a role of forming a base to build " knowledge representation languages" (Tramullas 2009: 1).

Moreover, " semantic integration of heterogeneous information sources such as digital libraries can benefit with the incorporation of ontologies" (Noeto 2010:

6). Among applications, there are those that utilize domain ontology in integrating information resources while other applications enable each resource to utilize its own ontology. Every single user can as well have his or her own ontology basing on his or her interests, role or language in a " determine domain" (Noeto 2010: 18). Ontologies offer a source of accurately defined terms.

In the applications of information retrieval, ontologies play a role of disambiguating user queries, to undertake elaboration of thesaurus or taxonomies of terms to enhance retrieved results quality. Machine-learning methods are utilized in extending ontologies basing on the users interactions.

RDF/RDFS/OWL

RDF and RDFS

RDF or “ Resource Description Framework” is built on former developments such as the “ platform for Internet Content Selectivity” and Dublin Core content rating initiative. RDF statement is found to be of the form “[Subject property object]” (Pan and Horrocks 2003: 5). The RDF-annotated resources are normally given names by URIrefs. RDF serves to annotate the Web resources concerning the named properties. The values of the named properties can be URIrefs of literals or Web resources, that is, representations of the data values. RDF statements set is referred to as RDF graph.

RDF Schema or RDFS can be viewed as an initial attempt to support the expression of simple ontologies having RDF syntax. In the RDFS, “ predefined Web resources rdfs: Class, rdfs: Resources and rdfs: Property can be used to declare classes, resources, and properties, respectively” (Pan and Horrocks 2003: 5). RDFS serves to predefine the meta-properties, which can be utilized in representing the background assumption in the ontologies: “ rdf: type, rdfs: subClassOf, rdfs: subPropertyOf, rdfs: domain and rdfs: range” (Pan and Horrocks 2003: 5). In clear terms, RDFS is ontology language that is simple, which supports only property and class hierarchies and range and domain constraints for the properties. However, basing on the “ RDF Model theory”, it is far much more complex than that.

RDF MT offers semantics not just for the RDFS ontologies, but for RDF triples as well. The RDF MT is set up on some simple interpretations. Given URI references V, simple interpretation I of V is given definition by (a) “ an non-empty set of properties of resources” referred to as the domain of I, (b) a set IP, (c) a mapping IEXT, and (d) a mapping IS, : from URIrefs in V to IR U IP” (Pan and Horrocks 2003: 6). Figure below gives a simple interpretation of I of V = {a, b, c}, in which URIref b is interpreted as being a property since IS(b) = 1 € IP and IEXT(IS(b)), is a set of the pairs of resources, which are in IR, that is, {(1, 2), (2, 1)}. And since {IS(a), IS(c)} € IEXT(IS(b)), I([a b c .] = true, therefore, a conclusion can be that I satisfies [a b c].

Source: Pan and Horrocks 2003

RDF MT offer semantics not just for the RDFS ontologies, but for the RDF triples as well. RDF MT is set up on some simple interpretations. Basing on these interpretations, RDF MT offers semantics for the RDF triples as well as RDFS statements by “ RDF-interpretations and RDFS-interpretations, respectively” (Pan and Horrocks 2003: 7). Such interpretations are the simple interpretations, which satisfy additional semantic conditions as well as axiomatic statements. In an intuitive manner, RDF-interpretations call for the IP to be an IR’s sub-set. In a similar manner, RDFS-interpretations present a requirement that the entire classes are actually, resources. In addition, RDFS-interpretations serve to bring in class extension function or ICEXT that operates in the following manner: “ A class URIref is firstly mapped (by IS) to a resource in IR and then is further mapped (by ICEXT) to a set of resources which are instances of this class” (Pan and Horrocks 2003: 7).

OWL

OWL is “ standard (W3C recommendation) for expressing ontologies in the Semantic Web” (Pan and Horrocks 2003: 9). The OWL language serves to facilitate higher Web resources’ machine understandability as compared with those supported by the RDFS by offering extra constructors for setting up property and class descriptions as well as new axioms together with formal semantics. OWL recommendation certainly comprise of 3 languages of raising the level of expressive power: OWL full, OWL lite and OWL DL. OWL DL and OWL lite are, just like DANL+OIL, essentially quite sensitive Description Logistics, there are nearly equal to the SHOIN(D+) and SHIF(D+) Descriptive Logics. On the other hand, OWL Full offers similar constructor sets as OWL DL; however, it enables them to be utilized in a manner that is unconstrained. It is not hard to prove that OWL Full is actually undecidable and this is for the reason that it does not put restrictions on utilization of the transitive properties. Thus, OWL DL is indeed “ the most expressive decidable sub-language of OWL” (Pan and Horrocks 2003: 10).

In summary, there are some main differences in regard to expressive power, between RDF, RDFS and OWL DL. The RDF(S) is found to be less expressive as compared with OWL DL in a way that: it does not offer any kind of constructor to construct property or class descriptions and also that, RDF(S) are found not to offer a large number of axioms regarding properties, classes and individuals the same way OWL DL does. Conversely, RDF(S) supports the axioms regarding meta-properties and meta-classes, but OWL DL does not support these. OWL Full offers all the mentioned axioms and constructors, encompassing RDF(S)'s remodeling. On the other hand, OWL Full is actually not decidable (Pan and Horrocks 2003: 11).

Implementation

Findings of Automatic Semantic Extraction Paper and the Success of SADI

SADI is found to be quite lightweight in comparison with a large number of other approaches to the Semantic service provision. SADI comprises of two main best-practices which include: the output instance IRI, which is similar to the input instance URI, and all service output and input data are RDF instances of OWL classes. The former best practice standardizes effectively the behavior of services by making them to be all " annotators", in which inputs turn out to be decorated by the additional information before they are returned to the client. On the other hand, the latter best practice allows flexible and sophisticated matchmaking between the in-hand data and tools, which can work on data, and do this using a progressively widely utilized RDF.

The SADI services use and produce the RDF instances of the OWL DL classes. Encompassed in services interface document are the reference to OWL-DL classes, which give definition to the " input and output data-types that the service will consume and produce" (Wilkinson et al. 2011: 8). Ontologies that define such classes may exist at any place on the web, and they may be " owned" or may not by the service provider. On the other hand, the input and output URI class has to resolve, via HTTP GET, to OWL document. It is clear that SADI enables any provider to utilize classes from whatever OWL ontology in " the definition of their own service interface" (Wilkinson et al. 2011: 8).

The data service that is used by SADI service is the OWL-DL class’s instance, which gives description of the service’s input. In the same manner, the output is actually an instance of output class. RDF-N3 as well as RDF-XML serializations are, in the current day, supported and is shown in the “: Content-type of the HTTP header” (Wilkinson et al. 2011: 8).

Because both the service and the client are operating on a prospectively very huge RDF graphs, it is imperative to show what IRI in that graph serves to represent the data instances “ root”. Again, here there is complete reliance on the “ semantic Web standards”, having a requirement that the input instance has to be categorized basing on the input class of the service provider, and clearly types by utilizing the rdf: type predicate. This plays a role of reducing the service provision complexity by not having the requirement of providers to engage in reasoning over the incoming data – a significant consideration with regard to encouraging extensive SADI adoption. In addition, it enables the writing of services in languages, which do not have very strong support for the logical reasoners, like Perl. During the time of accepting the incoming data, a provider just undertakes extraction of the URI from input document, which has rdf: type property with a particular value equal to the input class of that service. Client software, in a similar manner, can expect that “ service provider has added the rdf: type property to its root output data node in accordance with its output class, and thus it is similarly straightforward for the client to identify output data elements within the returned graph” (Wilkinson et al. 2011: 8).

The invocation of SADI services is carried out by passing RDF graph to service end-point through HTTP POST and any other tool, which can help in executing an HTTP POST, can be utilized to undertake invocation of a SADI service. Of great importance, SADI utilizes a “ non-parameterized POST”, that is, does not utilize HTTP FORM encoding. In this manner, all information that is required for the invocation of service has to be present in data itself, as the invocation occurs through one anonymous data “ package”. SADI helps in accomplishing this by differentiating a variety of service of data control elements, basing in their ontological type.

The input data is “ decorated” up to the time it turns out to be the output class instance. This is the crucial aspect of SADI specification, which contributes towards having striking interoperable behaviors of SADI. In addition, in this way of modeling services as well offers simple solutions to the problems, which would or else call for having project-specific standards such as input-output mapping in multiplexed invocation. In simple terms, after a service analyzing the values/predicate attached to a particular input node, “ it then adds the analytical output to that same node through one or more new predicate/values” (Wilkinson et al. 2011: 8). The output is linked to the input as a fresh property of that particular input URI. All values and predicates added by a service “ are defined in the Output OWL Class, and as such, output data is then rdf: type’d according to output class definition” (Wilkinson et al. 2011: 8).

Considering the increasing size of the bioinformatics datasets, as well as the move to cloud computing, SADI natively offers support to the capability of passing data by reference. It is also found that in the case of input as well as output data, “ the URI of the owl: individual may be annotated with an rdfs: isDefinedBY predicate” (Wilkinson et al. 2011: 8). That predicate’s Object

URI, when resolved, has to offer triples that have any missing data for that particular individual. In this manner, there is possibility of passing large data objects from “ service-to-service” without essentially passing the data, “ but still provide the ability to retrieve that data in a standards-compliant way” (Wilkinson et al. 2011: 8).

Conclusion

Considering RDF, RDFS and OWL DL, there are some main differences in regard to expressive power, among them. The RDF(S) is found to be less expressive as compared with OWL DL in a way that: it does not offer any kind of constructor to construct property or class descriptions and also that, RDF(S) are found not to offer a large number of axioms regarding properties, classes and individuals the same way OWL DL does. On the contrary, RDF(S) supports the axioms regarding meta-properties and meta-classes, but OWL DL does not support these. OWL Full offers all the mentioned axioms and constructors, encompassing RDF(S)’s remodeling. It has also be seen that, putting the increasing size of the bioinformatics datasets, as well as the move to cloud computing into consideration, SADI natively offers support to the capability of passing data by reference.

Works Cited

Farquhar, Ali et al. “ The Ontolingua Server: A tool for collaborative ontology Construction.”. International Journal of Human-Computer Studies, 46. 6 (1997): 707-727.

Fensel, David. “ The semantic web and its languages.” IEEE Computer Society 15. 6 (2000): 67−73.

FIPA. “ Foundation for Intelligent Physical Agents.” FIPA Ontology Service Specification, March 2001.

Gruber, Robert. “ A Translation Approach to Portable Ontology Specifications.” Knowledge Acquisition, 5. 1, (1993): 199−220.

Noeto, Maria A. “ An overview of ontologies.” Center for Research in Information and Automation Technologies, 2003.

Pan, Jeff and Horrocks Ian. “ RDFS(FA): Connecting RDF(S) and OWL DL.” 2nd International Semantic Web Conference, 2003

Russell, Stuart and Norvig Peter. Artificial Intelligence: A Modern Approach. 2nd Edition. New Jersey: Prentice Hall, Englewood Cliffs, 2010. Print.

Starlab, Steve. “ Systems Technology and Applications Research Laboratory home page.” Faculty of Sciences, Department of Computer Science, Thesis Paper Submitted to Vrije University, Brussels, 2011.

Tramullas, Jairus. Material from Profesor Jesús Tramullas Saz, Department of Documentation,

Universidad de Zaragoza, Spain, 2009.

Wilkinson, Mark D. et al. “ The Semantic Automated Discovery and Integration (SADI) Web service Design-Pattern, API and Reference Implementation.” Journal of Biomedical Semantics, 2. 1, (2011): 8.