Performance Evaluation of the Semantic Web Reasoners

Deepika Chaudhary, Jaiteg Singh, D. P. Kothari

Abstract: ***The major benefit of working on Ontology Web Language (OWL) is its ability to define semantics such that the information becomes more valuable. To realize the full power of semantics, it is essential to integrate a reasoning engine to it. The software codes that perform inferences are often referred to as reasoning engines or reasoners. The reasoners can be classified into categories: tableau based and rule based reasoners. The rule based reasoners combines the assertions with a set of logical rules to infer new knowledge chunks. The Jena framework offers several ways to integrate rule based reasoners programmatically. The operation is similar to creating a more advanced model from a simpler one. The objective of this paper is to list and classify the reasoners according to OWL 2 profiles thereafter the focus of this study is to develop a model which evaluate the performance of Semantic Web Reasoner based on few parameters.***

Keywords: Semantic Web Reasoners, Jena Framework, OWL, Inference.

I. INTRODUCTION

Now days when data is coming from different corners of the world and in various forms i.e. unstructured, semi-structured and structured it is the semantic technology which is acting as an umbrella for the enterprises to integrate the heterogeneous data sets and enrich them with proper data analytics and a way to optimize and manage these mission-critical data sets. The ability to process the exponentially increasing data sets which are coming from both internal and external data sources is the key for efficient data analysis and is very much required for in the field of business analytics or any other data science strategy. Semantic web technologies are nothing else but a collection of all the tools and techniques required to process such mission-critical data sets and therefore implement Semantic web applications (Kumar, B., 2015). The semantic web offers a robust and sensible approach to realize mastery over the multitude of data. The Semantic web technologies provide us the platform to design web pages in a manner that can be dealt by computer easily.

The Semantic Web permits information to work things out through illation. Imagine data which can add data to itself, by itself. This can be the ability that solely a couple of data technologies provide. These systems are challenged with the job to process a pool of information which is scattered throughout the web and requires extra care to validate this information. Once the information is described, it is the power of the reasoning system that can entail newer relationships out of it. For example, say your database consists of the two facts; Irina is a female, a female is a type of person. Then, the query asking all persons would include Irina too. Though this can be easy and simple, it can act as a true example for justifying the value reasoner in Semantic Web. Abstract thought makes every knowledge item additional valuable as a result of this; it will have control over the creation of the latest information. This research highlights the role of reasoning in any Semantic Web application and how an efficient reasoner could help us in integrating the interoperable data coming from distributed and dynamic resources.

Below we present the core components of any Semantic Web System

1.2 Core component of Semantic web applications

The Semantic Web allows the definition and structuring of information such that it makes the process of integration more easier, modify logical thinking and permit extraction of purposeful data whereas the data is in any type distributed, dynamic or numerous (Hebler J., Fisher M, 2008). The core component of any Semantic Web consists of the following

a) Statement: A statement, in alternative words, consists of multiple components that usually form a triple. A triple incorporates subject, object, and predicate. These statements outline the structure of the knowledge, specific instances, and limits there structure. These statements are associated with each other to make the information internet that's conjointly a region of the Semantic Web.

b) Uniform Resource Identifier (URI): URI is a unique name for data items mentioned in statements represented throughout the internet. Therefore, each component of this statement comprises the subject, object, and a predicate. This URI affirms the uniqueness and identity of the objects throughout the entire WWW.

c) Ontology: The collection of statements that describe concepts, constraints and their relationships is known as Ontology. It is similar to a schema defined for a database or a diagram describing a class. Much rich ontology exists for their direct inclusion into applications.

d) Language: Statements are defined concerning the Semantic Web Language. The language is a collection of various keywords and offers a degree of complexity and expressiveness.

e) Instance Data: These are the statements about instances rather than some generic concepts.

Revised Manuscript Received on November 15, 2019.

Deepika Chaudhary, Chitkara University Institute of Engineering & Technology, Chitkara University, Punjab, India. Email: Deepika.chaudhary@chitkara.edu.in.

Jaiteg.Singh, Chitkara University Institute of Engineering & Technology, Chitkara University, Punjab, India. Email: jaiteg.singh@chitkara.edu.in.

D.P.Kothari, Director Research Wainganga College of Engineering & Management, Nagpur, India,. Email: dpkvits@gmail.com.

f) Reasoners: Reasoners add on the inference to the knowledge base. Through inferences, one can create additional statements that offer classification and realization. There are several types of reasoners which can be plugged in other tools and frameworks.

g) Rule Engines: Rule engines support inference typically beyond logic. They add on a strong dimension to the Semantic Web.

i) Semantic Frameworks: Is a combination of all the above tools working as an integrated unit.

The next section focuses upon how the knowledge is represented in any Semantic Web systems and also describes the requirements of the reasoning process.)

1.3 Knowledge representation & Reasoning

Knowledge Representation is a process in which along with using the vocabulary we also: Add new knowledge chunks and set the relationships with the previous knowledge, alter the existing behavior by adding new beliefs, debug erroneous behavior by locating the faulty beliefs and thus can justify the working of the system. Overall, then, this system can be defined as the representation of known facts about its world and adjustments to its behavior accordingly (Minsky, M., 1974.). Knowledge representation and reasoning are closely related or in other words are incomplete without each other which mean we would fire actions based on what the system believes as opposed to just what is being explicitly stated. For an example if we explicitly state the following two facts,

a) p is a patient and is allergic to some medicine m.

b) Anyone who is allergic to medicine m is also allergic to medicine m'.

Know if we closely observe the scenario a picture is painted in which all the patients allergic to m are also allergic to m'. This process of entailing new knowledge is called Logical Entailments.

Definition 1: If some prepositions are represented by a certain set of statements S entails some other prepositions represented by a set of sentence p when the truth of preposition p is implicit in preposition s.

A knowledge base needs to apply an inference component to interpret semantics and realize the enriched information. Applications that perform inference are often referred to as reasoners or reasoning engine. The role of reasoning in any Semantic Web system is to exhibit the entailments/inference on explicitly stated clauses within a given knowledge base. This can be achieved through the use of rules, a rule engine triggers on RDF store, algorithms like decision trees, tableau algorithms, and also programmatically. Many Semantic Web frameworks accomplish the inference through rule reasoning engines. The engine thus integrates all the assertions which are a part of knowledgebase with a collection of logical rules to obtain assertions and initiate necessary actions. Every entailment is based on some logic which can be defined as the study of entailment relations, languages, true conditions, and the rules of inference. The first language which defined the logics was first order predicate logic (FOL) which was invented by Professor Gottlob Frege for the formalization of mathematical inference but was adopted by the research community. It must be highlighted here that FOL was just the starting point (Chimakonam, J.O., 2012). In the Semantic Web, we can also control the reasoning process in a domain specific way.

Two major problems associated with reasoning is soundness and completeness of data which means that if we try to entail the KB in a reasonable amount of time we tend to miss a few entailments which make the process logically incomplete and on the other hand if we try to infer in reasonable amount of time then it gives us some incomplete answers making reasoning logically unsound (Horrocks, I., Sattler, U. and Tobies, S., 2000.).

The next section describes the background knowledge which is required for a complete understanding of how logical entailments are obtained using the reasoning process.

1.4 Reasoning on RDF graphs

The major benefit of working on OWL is its ability to define semantics such that the information becomes more valuable. In the Semantic Web domain, an inference is a process required to optimize data integration by finding new relationships and by analyzing its content (Brickley, D., 2000.). For performing inference, the applications must have a subsystem which can infer and interpret the results. This subsystem is called as a reasoning engine which usually triggers the rules on the RDF Store. A reasoning engine discovers new relationships build on the given data and based on rules which are expressed in vocabulary. In general, vocabularies are a classification technique which defines the classes and sub-classes and also defines the relationships among them and their instances. In technical terms, we called this document as ontology. Few Semantic Web applications perform entailments using rule-based reasoning engines. These engines embed assertions along with the set of logical rules and concentrate on inferring new relationships based on the existing one. A rule can be defined as a means of knowledge representation that often goes beyond expressivity. Reasons, why rules are required, are because OWL lacks a certain amount of expressiveness which can be integrated using rules. Many Semantic web applications execute the process of inference using rule-based reasoners. These subsystems combine various assertions which are defined in the knowledge base with the help of a set of logical rules to derive new patterns. The example of a, few sets of rules is expressed in the figure given below: These rules are specified by W3C consortium (W3C Consortium, 2005.) and are mentioned in table1 and the syntax for rule instantiation is given in Fig 1.1.

Table 1.1 W3C Reasoning rules

Rules	If Pattern contains	Then Inference drawn
rdfs2	aaa rdfs:domain xxx . yyy aaa zzz .	yyy rdf:type xxx .
rdfs3	aaa rdfs:range xxx . yyy aaa zzz .	zzz rdf:type xxx .
rdfs4a	xxx aaa yyy .	xxx rdf:type rdfs:Resource .
rdfs4b	xxx aaa yyy.	yyy rdf:type rdfs:Resource .
rdfs5	xxx rdfs:subPropertyOf yyy . yyy rdfs:subPropertyOf zzz .	xxx rdfs:subPropertyOf zzz .
rdfs6	xxx rdf:type rdf:Property .	xxx rdfs:subPropertyOf xxx .
rdfs7	aaa rdfs:subPropertyOf bbb . xxx aaa yyy .	xxx bbb yyy .
rdfs8	xxx rdf:type rdfs:Class .	xxx rdfs:subClassOf rdfs:Resource .
rdfs9	xxx rdfs:subClassOf yyy . zzz rdf:type xxx .	zzz rdf:type yyy .
rdfs10	xxx rdf:type rdfs:Class .	xxx rdfs:subClassOf xxx .
rdfs11	xxx rdfs:subClassOf yyy . yyy rdfs:subClassOf zzz .	xxx rdfs:subClassOf zzz .

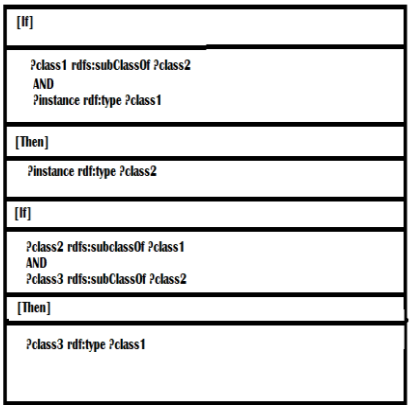


Figure 1.1 the syntax for rule instantiation

There can be many different kinds of rules based on the kind of application. Some languages allow only conjunctive rules (A and B imply C) while others allow disjunctive rules (A or B imply C). Chaining is a continuous process where the rules are applied to the explicitly stated facts first, and slowly, they infer new facts. There are mainly two methods of chaining for a rule-based reasoner: forward chaining & backward chaining. In the proposed methodology, we have adopted the process of forwarding chaining. In forwarding chaining the inference process starts by firing of rules on any node and checks for the new axioms drawn; the reasoner thus keeps on firing the rules to each of the individual nodes until it stops finding new inferences.

1.5 Types of reasoning engine

1) Fact++ - An open source, C++ based reasoner supporting a large subset of OWL DL (Levin, M.K. and Cowell, L.G., 2015.).

2) Hermit – This Java based OWL reasoner is rooted upon a new tableau reasoning algorithm. It can be integrated into Protégé and Java applications using the OWL API [30].

3) KAON2 – This reasoner uses Java based framework for reasoning with OWL DL ontologies. It supports reasoning over a large subset of OWL DL.

4) Pellet – It is an OWL Description Logic (DL) reasoner. It is java based and is also an open source that supports a most of the constructs of OWL, which includes constructs available in OWL2. Pellet is developed and commercially supported by Clark and Parsia.

5) Racerpro – This commercially available reasoner supports a large subset of OWL DL (Waterfeld, W., Weiten, M. and Haase, P., 2008.).

6) Vampire – This award-winning, commercially licensable, first-order logic theorem prover has been the subject of the investigation looking into its applicability as an OWL DL reasoner.

7) CEL - is designed for large ontologies which work on the principle of polynomial time subsumption algorithm [6].

II. LITERATURE REVIEW

Here we present highlights of the major paper reviewed. [1] In their paper did an extensive analysis of large scale reasoning algorithms which works on the increasing volume of semantically interlinked data. The paper also presented a structured review of the literature where scalable reasoning approaches were used over different OWL profile languages. They also emphasized upon large scale ontologies reasoning which can be more successful if parallelization is done and the computational power can also be distributed among nodes.[44] In this paper, the authors described a new inference engine, the Mini-Me (the mini matchmaking engine) which supports smart semantic applications on PC and Smartphones. In this paper, they have also worked on portability factor. They have suggested a framework through which the inference engines can be easily ported on different hardware's. [31] In this paper, the authors have compared classical reasoners like Hermit, Pellet, Racer, and Fact++. They have also made a comparison of a few Semantic web frameworks like Jena, Protégé, and Swoop, etc. They have classified the reasoner on the various traits like response time and have also commented that Pellet has the lowest response time and Racer has the highest. [36] In their study emphasized that although there is a boom in data represented in the form of RDFS and ontologies, still there are very few approaches which have the capabilities of supporting any given rule set. They proposed a framework which supports any given rule in the distributed reasoning framework. They also focussed on the challenges that they have faced in implementing the proposed framework. [37] In this study, the author's proposed a hybrid system which combines a Description Logic Reasoner with a forward chaining rule engine. The proposed system was named as HeaRT-Pellet.

In the proposed system, integration of mature rule set was done with the large fact-based ontologies. [55] in their study applied the classical reasoning tools on web-scale data and gave directions as to how to optimize performance in parallelization. This paper focused on the need for reasoning on web scale data. The study introduced three major applications of the semantic web: Hadoop, LarKC and MaRVIN, and Had. [11] in the paper, suggested the scalability of Ontology reasoning tasks as important. They in their study did an in-depth analysis of Ontology framework and also identified a few clusters. They also benchmarked a few of the ontology reasoners based on clusters and the nature of queries. In [13] the study done by Celig D. proposed a novel technique which self discovers the web service required to satisfy a particular user requirement. He also suggested that the lack of the knowledge base in information retrieval process of UDDI make the process more difficult. He proposed a agent semantic search agent (SSA) which will find the required Web services satisfying the need of the individual user. [3,4] The author Grigoris and Van in their book introduced the Vision of The Semantic Web. They then introduced the structured web documents using XML. The book also describes a complete knowledge of all web resources, including RDF and OWL. The inference mechanism and Logic rules are also described with examples. This book also gave an introduction to ontology engineering and areas where the Semantic web can be applied. [38] The authors in their research stated the importance of Reasoning for the success of any semantic web application. He emphasized the use of Description Logics to provide sound and complete reasoning on Web Ontology Language (OWL). Below we present the summarization of reviewed literature in a tabular form.

Table 2.1 Literature survey on Semantic Web Technologies

Author's/Year	Title of the Paper	Description
Antoniou G., Batsakis, S., utharaju, R., Pan, J.Z., Qi, G.,Tachmazidis , I., Urbani, J. and Zhou, Z/2018	A survey of large-scale reasoning on the Web of data. *The Knowledge Engineering Review*	A structured review of the literature where scalable reasoning approaches were used over different OWL profile languages. They also emphasized that the reasoning on the OWL EL profile ontologies can be successful if parallelization is done where the computational power can be distributed among nodes.
Scioscia, F., Ruta, M., Loseto, G., Gramegna, F., leva, S., Pinto, A. and Di Sciascio, E., /2018	Mini-ME matchmaker and reasoner for the Semantic Web of Things	In this paper, the authors described a new inference engine, the Mini-Me (the mini matchmaking engine) which supports smart semantic applications on PC and Smartphone's. The authors have also addressed the issue of portability.
Khamparia , A. and Pandey, B. 2017	Comprehensive analysis of semantic web reasoners and tools: a survey	In this paper, the authors have compared various classical reasoners. They have also made a comparison of a few Semantic web frameworks like Protégé, Jena, Swoop etc. The author has classified the reasoners on the parameter response time.
Mutharaju, R., Mateti, P., and Hitzler, P., /2015	Pellet-HeRT–proposal of an architecture for ontology systems with rules	In this study, the author's proposed a hybrid system which combines a Description Logic Reasoner with a forward chaining rule engine. The proposed system was named as HeaRT-Pellet. In the proposed system integration of mature rule set was done with the large fact based ontologies.
Nigel Shadbolt ,Tim Berners-Lee	The Semantic Web Revisited	Nigel Shadbolt, in his study, highlighted that how the digital world has evolved at a prodigious rate, after the term Artificial Intelligence has been Coined at Darth Mouth Conference. The study highlighted the need for data integration and how Artificial intelligence can help in the science of the web where the focus is on developing and deploying a system of human and machines.

III. CLASSIFICATION OF REASONER

The process of Inference is the backbone of every semantic web application. The complete summary of OWL reasoners helps Semantic Web designers to select a suitable reasoner for their applications. A reasoning engine can be defined as a software code which has the ability to entail new axioms from a group of already defined facts or axioms. Efficiency, Correctness, Inference Capabilities are the few key parameters of a Sound Semantic Web Reasoner. Few examples of Semantic web reasoners are FaCT++, Pellet, HermiT, Kaon2, Hoolet, etc. These reasoners are classified based on the subset of ontologies.

The Ontologies can be classified into three profiles, and so do the reasoners. We in our study [14] have clearly described the need for OWL Profiles and different categories of reasoners for OWL 2 DL RL and QL reasoners. [35] did an extensive survey for proper classification of Semantic Web and have classified 19 different reasoners which were released between 1975 and 2009. The classification was done based on inference capability, soundness, and completeness. Few standards have also been established to evaluate the performance of Semantic Web Reasoners. The two already established standards are University Ontology Benchmark (UOBM) and Lehigh University Benchmark (LUBM). Below, we present the classification of various Semantic Web Reasoners.

Table 3.1 Classification of reasoners

Reasoner	User Interface	OWL Profile	Characteristics	Rule Support	Complexity
Pellet	OWL API JENA	DL EL	Soundness Completeness Expressivity: Partial	Yes(SWRL)	PTime Complete
FACT++	OWL API	DL	Soundness Completeness Expressivity: Partial	No	PTime Complete
RacerPro	OWLAPI	DL	Soundness Completeness Expressivity: Partial	Yes(SWRL)	PTime Complete
HermiT	OWLAPI	DL	Soundness Completeness Expressivity: Partial	Yes(SWRL)	PTime Complete
CEL	OWLAPI	EL	EL+	NO	PTime Complete
ELK	OWLAPI JENA	EL	EL		PTime Complete
Sonorocket	OWLAPI	EL	EL+	NO	PTime Complete
OWLgres	SPARQL Parser	QL	Soundness Completeness Expressivity: Partial Conjunctive Query Answering		NP Complete

3.2 Design of a model for performance evaluation of Semantic Web reasoned

This section presents the algorithm for materializing the ontologies based on the hybrid approach, which calls two separate reasoners to entail ontologies as discussed in the previous section.

Input: Ontology (Tbox, Abox), Entailment Rules.

Output: Materialized Ontology in the forms of inferred Triples used for Query Answering.

Step 1: Input Ontology to the Reasoner.

Step2: Classifying Axioms into Schema Based (TBOX) and Assertional Based (ABOX).

Step 3: Make a call to a Generic Description Logic Reasoner for entailment of Terminological based Triples.

Step 4: Uploading the rule set for in the Agenda (The knowledge base containing the Domain specific Rules)

Step 5: Calling a Rule-based reasoner for Domain Specific (ABOX) entailments.

Figure 3.1 Algorithm for materializing ontology

3.3 Architecture of Semantic Web reasoner and its description

The reasoners in Semantic Web applications follow a generic architecture [Figure 3.1].

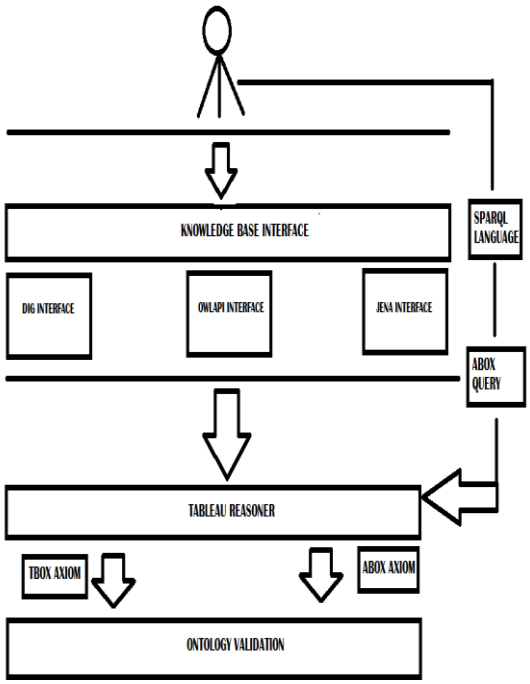


Figure 3.2 Generic Architecture

We start our exploration with an examination of Jena's programming framework. Jena is implemented in the Java programming language. The Jena Semantic Web framework maintains a consistent treatment of the Semantic Web through its use of Java classes and variables. The Jena framework employs the following major Java Classes.

a) Resource – A class representing an element contained within a statement such as subject, predicate or objects. There also exists a Jena resource referred to as reified statement that considers a triple a single resource.

b) Statement – A Semantic Web triple defined as a combination of subject, object and a predicate. The statement class allows simple interrogation of its containing components.

c) Graph – Basic method for maintaining Semantic Web data. A graph allows basic add, delete, find and contain operations.

d) Model – A model builds on the basic graph to offer rich interactions with Semantic Web data. The applications read, write, reason and query Semantic Web data through access to Jena model.

e) Reasoner – Contains the reasoner processing through either internal or external reasoner. External reasoner enables third party reasoner access to the knowledge base. The reasoner designed here is a third party reasoner which is invoked using Java calls.

3.4 Setting up the Model

For Jena, it all begins with the creation of the Model object. The working of the proposed model is based on the segmentation of the Tbox and Abox reasoning. In the proposed methodology the reasoner first load the ontology; next the segregation of both Terminological axioms (Tbox) and Assertion axiom (Abox) take place [8,16]. Then the programmed reasoner makes a call to pellet reasoner for Terminological entailments which successfully classifies the elements of a given ontology. Once this process is over it uploads the rules set for rule-based entailments. The proposed model integrates the effectiveness and efficiency of tableau algorithm along with the scalability of rule-based reasoners. The experimental research shows that way the efficiency and scalability of the reasoner can be enhanced.

3.5 Data set preparation and OWL inference

In this section, we present hands on example, of enabling various levels of inference in knowledgebase containing a simple OWL ontology.

3.5.1 Ontology corpus

Here a few of standard ontologies have been selected. The corpus includes the standard University of Bench Mark. The Lehigh University Benchmark (LUBM) was developed to deal successfully with the evaluation process of Semantic Web depositories in standardized and systematic ways. It consists of a domain ontology for a university. University of Bench Mark (UOBM) provides three separate data sets. In Annexure A, we present the Ontology and then the three levels of inference [29].

These ontologies contain elements of RDFS and OWL. In these ontologies, several classes and subclasses are defined. These Ontologies were given as input to the hybrid reasoner programmed using the Jena Framework. In table 3.2 , we present the table depicting the dataset used for classification & realization.

Table 3.2 Training Dataset (OWL2 Profile ontologies)

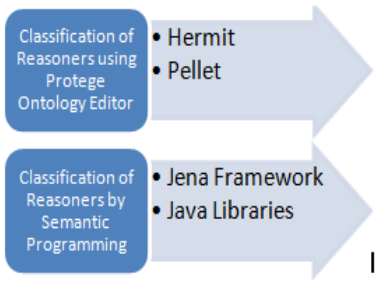
Ontology Expressiveness	Ontology URL	Evaluated Ontology Storage System
OWL DL	http://www.co-ode.org/ontologies/pizza/pizza.owl	RDF Store
OWL - DL	http://protege.cim3.net/file/pub/ontologies/travel/travel.owl	RDF Store
OWL – DL	http://owl.man.ac.uk/2006/07/sssw/people.owl	RDF Store
OWL – DL	University Bench Ontology	RDF Store
OWL - DL	http://www.w3.org/TR/owl-guide/wine.rdf	RDF Store
OWL - DL	https://bioportal.bioontology.org/ontologies/CMO	RDF Store
OWL - DL	https://bioportal.bioontology.org/ontologies/pato	RDF Store

3.6 Machine configuration

All the experiments in this research work were conducted on Intel ® Core™ i3 – 6100u CPU @ 2.50 GHz with 4 GB of RAM and 64-bit operating system.

IV. RESULT ANALYSIS

Several metrics are useful for measuring the performance of a reasoner. In our study, the results are analyzed using two methods. In the first method, we classified and realized the ontologies using protégé framework where the ontologies were given as an input to the Pellet and Hermit reasoner, and the results were noted. It was observed that in most cases the Pellet reasoner was a bit faster than Hermit. Although it has been stated by many researchers that the reasoner can behave differently even if the input is same, therefore one reasoner can take seconds to entail while others may require minutes to perform the entailments which can further be extended till hours.. Figure 4.1 presents the two modes based on which the experimentation was done to analyze the results when the same input is given to different reasoners.



.Figure 4.1 Methods for classification of reasoners

In this section, we compare the performance of the reasoners on three different OWL profile languages OWL DL, OWL RL, and OWL EL.In table 4.1 we have presented the result obtained when the mentioned ontologies were given as input to both Pellet and Hermit reasoner using Protégé Framework.

Table: 4.1 Comparison of Semantic Web Reasoners

Code	Ontology URL	Statements	Time taken to Classify & Realize the Elements Pellet Reasoner (ms)	Time taken to Classify & Realize the Elements Hermit Reasoner (ms)
UB	University Bench Ontology	45	123	242
TR	http://protege.cim3.net/file/pub/ontologies/travel/travel.owl	61	220	567
PE	http://owl.man.ac.uk/2006/07/sssw/people.owl	61	225	227
PI	http://www.co-ode.org/ontologies/pizza/pizza.owl	101	723	1041
PATO	https://bioportal.bioontology.org/ontologies/pato	2731	708	848
CMO	https://bioportal.bioontology.org/ontologies/CMO	2991	845	900
GALEN	https://bioportal.bioontology.org/ontologies/GALEN	23143	Java heap space failed	Out of Memory : Java Heap

In Graph 4.1 we compare the time taken to classify and realize the elements of three different ontologies of OWL2 profile. For the purpose of experimentation, the proposed algorithm was fired on a few other ontologies and the result are presented in the figures given below.

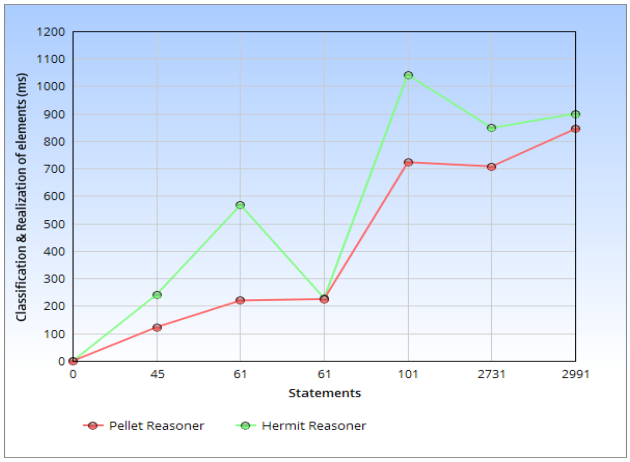


Figure 4.1 Result Analysis

Below we present the result analysis of the hybrid reasoner, which was semantically programmed using the Jena framework, various Java libraries. The methodology of the proposed reasoner is already discussed above in the chapter. This reasoner works by partitioning the Axioms into Tbox and Abox categories. It then calls pellet reasoner to classify the Tbox axioms and calls the Jena rule engine to classify the Abox axioms. So the metrics on which the results are analyzed is the time taken to separate the two axioms and after separation, how much time it takes to perform Tbox and Abox reasoning. Table 4.2 presents the statistics for the Hybrid Reasoner.

Table 4.2: Classification of Reasoner (Hybrid)

Code	Statements	Time Taken to separate Tbox and Abox axioms(ms)	Time taken to classify & Realize using Hybrid Reasoner
UB	45	1199	00:00
TR	61	1518	00:00
PE	61	1593	00:01
PI	101	1782	00:03
PATO	2731	3727	00:04
CMO	2991	3848	00:04
GALEN	23143	10503	It took an approx hour to properly classify the Elements

In Figure 4.2, we present the result for the above-stated table. The analysis depicts that there is a relation between the no of elements classified by an ontology concerning the time taken to separate both the axioms. The benefit of using this methodology is that when we segregate the axioms, then we can optimize the reasoning process and the scalability factor can be improved.

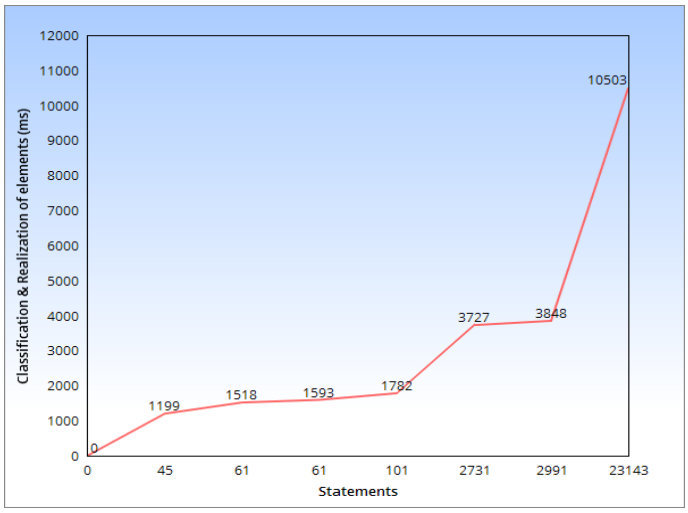


Figure 4.2: Result Analysis (Hybrid)

V. CONCLUSION

At the beginning of the paper, we have classified the reasoner based on their characteristics, complexity, rule support, and profiles. Next, we presented the working model of a hybrid reasoner which segregates the axioms into Tbox and Abox and thus performed the reasoning. The reasoner calls a Pellet reasoner for performing Tbox reasoning and then based on the rule set performs the Abox reasoning. This hybrid reasoner was designed and deployed using the Jena Framework. In the end, the performance evaluation of the results obtained was discussed and beautifully presented.

To conclude, it can be stated that

a) There is a probability that even if the input remains the same then also there is a chance that there is a deviation between the results as some reasoners may perform better on a particular input while others may fail.

b) By breaking the axioms into Tbox and Abox, we can improve the efficiency of a reasoner in terms of time taken to infer.

c) On the same ontology one reasoner can entail the axioms in seconds and the other can require minute or even hours to entail.

d) The hybrid reasoner works best for OWL EL Ontologies as Hermit and Pellet failed to perform an Inference.

REFERENCES

1. Andročec, D., Novak, M., & Oreški, D. (2018). Using Semantic Web for Internet of Things Interoperability: A Systematic Review. International Journal on Semantic Web and Information Systems (IJSWIS), 14(4), 147-171.
2. Allemang, D. and Hendler, J., (2011). 'Semantic web for the working ontologist: effective modeling in RDFS and OWL'. Elsevier.
3. Antoniou, G. et al. (2018) 'A survey of large-scale reasoning on the Web of data', The Knowledge Engineering Review. Cambridge University Press, 33.
4. Antoniou, G. and Van Harmelen, F., (2004). A semantic web primer. MIT press.
5. Baader, F. et al. (2007) 'Completing Description Logic Knowledge Bases Using Formal Concept Analysis.', in IJCAI, pp. 230–235.
6. Baader, F., Lutz, C. and Suntisrivaraporn, B. (2006) 'CEL�a polynomial-time reasoner for life science ontologies', in International Joint Conference on Automated Reasoning, pp. 287–291.
7. Beni, G. (2009) 'Swarm intelligence', Encyclopedia of Complexity and Systems Science. Springer, pp. 1–32.
8. Bergman, M. K. (2018) 'Platforms and Knowledge Management', in A Knowledge Representation Practionary. Springer, pp. 251–272.
9. Bernstein, A., Hendler, J., & Noy, N. (2016). A new look of the Semantic Web.
10. Bizer, C., Primpeli, A., & Peeters, R. (2019). Using the Semantic Web as a source of training data. Datenbank-Spektrum, 1-9.
11. Bock, J. et al. (2008) 'Benchmarking OWL reasoners', in ARea2008-Workshop on Advancing Reasoning on the Web: Scalability and Commonsense.
12. Bonabeau, E. et al. (1999) Swarm intelligence: from natural to artificial systems. Oxford university press.
13. Celik, D. and Elgi, A. (2005) 'A semantic search agent approach: finding appropriate semantic Web services based on user request term (s)', in 2005 International Conference on Information and Communication Technology, pp. 675–687.
14. Chaudhary, Deepika; Mantri, Archana; Kothari, D P.' International Journal of Advanced Research in Computer Science'Udaipur Vol. 5, Iss. 3, (Mar 2014).
15. Fensel, D. and Van Harmelen, F. (2007) 'Unifying reasoning and search to web scale', IEEE Internet Computing. IEEE, 11(2), pp. 95–96.
16. De Giacomo, G. and Lenzerini, M. (1996) 'TBox and ABox reasoning in expressive description logics.', KR, 96(316–327), p. 10.
17. Datta, R., et al., (2008). 'Image retrieval: Ideas, influences, and trends of the new age. ACM Computing Surveys (Csur)', 40(2), p.5.
18. Ding, L. Y., Zhong, B. T., Wu, S., & Luo, H. B. (2016). Construction risk knowledge management in BIM using ontology and semantic web technology. Safety science, 87, 202-213.
19. English, M., Auer, S., & Domingue, J. (2016, May). Block chain technologies & the semantic web: A framework for symbiotic development. In Computer Science Conference for University of Bonn Students, J. Lehmann, H. Thakkar, L. Halilaj, and R. Asmat, Eds (pp. 47-61).
20. Fillies, C., Wood-Albrecht, G. and Weichhardt, F. (2003) 'Pragmatic applications of the Semantic Web using SemTalk', Computer Networks. Elsevier, 42(5), pp. 599–615.
21. Gayo, J.E.L., Prud'Hommeaux, E., Boneva, I. and Kontokostas, D., (2017). 'Validating RDF Data. Synthesis Lectures on Semantic Web: Theory and Technology', 7(1), pp.1-328.
22. Glimm, B., Horrocks, I., Motik, B., Stoilos, G. and Wang, Z.,(2014). 'HermiT: an OWL 2 reasoner'. Journal of Automated Reasoning, 53(3), pp.245-269.
23. Guéret, C., Schlobach, S., Dentler, K., Schut, M., & Eiben, G.(2012). 'Evolutionary and swarm computing for the semantic web'. IEEE Computational Intelligence Magazine, 7(2), pp.16-31.
24. Guo, Y., Pan, Z. and Heflin, J., 2005. LUBM: A benchmark for OWL knowledge base systems. Web Semantics: Science, Services and Agents on the World Wide Web, 3(2-3), pp.158-182.
25. Haarslev, V. and Möller, R., 2001, June. RACER system description. In International Joint Conference on Automated Reasoning (pp. 701-705). Springer, Berlin, Heidelberg.
26. Hebeler, J., Fisher, M., Blace, R. and Perez-Lopez, A., (2011). Semantic web programming. John Wiley & Sons.
27. Hinchey, M.G., Sterritt, R. and Rouff, C., (2007). 'Swarms and swarm intelligence'. Computer, 40(4), pp.111-113.
28. Hitzler, P., Krotzsch, M. and Rudolph, S., (2009). Foundations of semantic web technologies. Chapman and Hall/CRC.
29. Hogan, A., (2014). Reasoning Techniques for the Web of Data(Vol. 19). Ios Press.
30. Horridge, M. and Bechhofer, S.,(2011). 'The owl api: A java api for owl ontologies'. Semantic Web, 2(1), pp.11-21.
31. Horrocks, I., Sattler, U. and Tobies, S.,(2000). 'Practical reasoning for very expressive description logics'. Logic Journal of IGPL, 8(3), pp.239-263.
32. Khamparia, A. and Pandey, B.,(2017). 'Comprehensive analysis of semantic web reasoners and tools: a survey'. Education and Information Technologies, 22(6), pp.3121-3145.
33. Levin, M.K. and Cowell, L.G., (2015). 'owlcpp: a C++ library for working with OWL ontologies'. Journal of biomedical semantics, 6(1), pp.35.
34. Li, P., Zeng, Y., Kotoulas, S., Urbani, J. and Zhong, N.,(2009), October. 'The quest for parallel reasoning on the semantic web'. In International Conference on Active Media Technology(pp. 430-441). Springer, Berlin, Heidelberg.
35. Mishra, R.B. and Kumar, S., (2011). 'Semantic web reasoners and languages'. Artificial Intelligence Review, 35(4), pp.339-368.
36. Moussallem, D., Wauer, M., & Ngomo, A. C. N. (2018). Machine translation using semantic web technologies: A survey. Journal of Web Semantics, 51, 1-19.
37. Maier, F., Mutharaju, R. & Hitzler, P. (2010). 'Distributed reasoning with EL++ using MapReduce'. Technical report, Department of Computer Science, Wright State University, USA. http://knoesis.wright.edu/pascal/resources/publications/elpp-mapreduce2010.pdf.
38. Nalepa, G.J. and Furmańska, W.T., (2010), September. 'Pellet-HeaRT–proposal of an architecture for ontology systems with rules'. In Annual Conference on Artificial Intelligence (pp. 143-150). Springer, Berlin, Heidelberg.
39. Parsia, B. and Sirin, E., (2004), November. 'Pellet: An owl dl reasoner'. In Third international semantic web conference-poster (Vol. 18, p. 2). Publishing.
40. Ratnayake, S., Rupasinghe, R., Ranatunga, A., Adikari, S., de Zoysa, S., Tennakoon, K., & Karunananda, A. (2008). 'Using swarm Intelligence to perceive the Semantic Web'. In Information and Automation for Sustainability, 2008. ICIAFS 2008. 4th International Conference on (pp. 91-96). IEEE.

41. Ren, Y., Pan, J. Z. & Lee, K. (2012). 'Optimising parallel ABox reasoning of EL ontologies'. In Proceedings of the 2012 International Workshop on Description Logics, DL-2012, Rome, Italy, June 7–10, 2012, CEUR Workshop Proceedings 846. CEUR-WS.org.
42. Salvadores, M., Correndo, G., Harris, S., Gibbins, N. & Shadbolt, N. (2011).'The design and implementation of RDFS backward reasoning in 4Store'. In *Proceedings of the 8th Extended Semantic Web Conference on The Semanic Web: Research and Applications—Part II*, ESWC'11, 139–153. Springer-Verlag.
43. Sarker, M. K., Xie, N., Doran, D., Raymer, M., & Hitzler, P. (2017). Explaining trained neural networks with semantic web technologies: First steps. arXiv preprint arXiv:1710.04324
44. Shadbolt, N., Berners-Lee, T. and Hall, W., (2006). 'The semantic web revisited'. IEEE intelligent systems, 21(3), pp.96-101.
45. Schlicht, A. & Stuckenschmidt, H. (2008). 'Distributed resolution for ALC'. In Proceedings of the 21st International Workshop on Description Logics (DL2008), Dresden, Germany, May 13–16, CEUR Workshop Proceedings 353. CEUR-WS.org.
46. Scioscia, F., Ruta, M., Loseto, G., Gramegna, F., Ieva, S., Pinto, A. and Di Sciascio, E., (2018). 'Mini-ME matchmaker and reasoner for the Semantic Web of Things'. In Innovations, Developments, and Applications of Semantic Web and Information Systems (pp. 262-294).
47. Sirin, E., Parsia, B., Grau, B.C., Kalyanpur, A. & Katz, Y. (2007). 'P ellet: a practical OWL-DL reasoner'. Journal of Web Semantics 5, 51–53.
48. Tachmazidis, I., Antoniou, G. & Faber, W. (2014).'Efficient computation of the well-founded semantics over big data'. TPLP 14, 445–459.
49. Tamburrini, G., (1998). 'Rule-Based Systems. Fuzzy Logic and Expert Systems Applications', 6, pp.123.
50. Tsarkov, and I. Horrocks, (2006) "Framework for an automated comparison of description logic reasoners.", Proceedings The Semantic Web – ISWC 2006, volume 4273, Springer.
51. Urbani, J., van Harmelen, F., Schlobach, S. & Bal, H. E. (2011). 'QueryPIE: backward reasoning for OWL Horst over very large knowledge bases'. In *10th International Semantic Web Conference, Bonn, Germany, October 23–27, 2011, Lecture Notes in Computer Science* 7031, 730–745. Springer.
52. Urbani, J., Margara, A., Jacobs, C., Voulgaris, S. & Bal, H. (2014). 'AJIRA: a lightweight distributed middleware for MapReduce and stream processing'. In Distributed Computing Systems (ICDCS), 2014 IEEE 34th International Conference on, 545–554. IEEE.
53. Vogt, L., Baum, R., Köhler, C., Meid, S., Quast, B., & Grobe, P. (2018, November). Using Semantic Programming for Developing a Web Content Management System for Semantic Phenotype Data. In International Conference on Data Integration in the Life Sciences (pp. 200-206). Springer, Cham.
54. Volz, R., Staab, S. & Motik, B. (2005). 'Incrementally maintaining materializations of ontologies stored in logic databases'. Journal of Data Semantics 2, 1–34.
55. Wu, K. & Haarslev, V. (2012). 'A parallel reasoner for the description logic ALC'. In Proceedings of the 2012 International Workshop on Description Logics, DL-2012, Rome, Italy, June 7–10, 2012, CEUR Workshop Proceedings 846. CEUR-WS.org.Google Scholar
56. Zhang, Y., Li, C., Chen, N., Liu, S., Du, L., Wang, Z., & Ma, M. (2019). Semantic web and geospatial unique features based geospatial data integration. In Geospatial Intelligence: Concepts, Methodologies, Tools, and Applications (pp. 230-253). IGI Global.

AUTHORS PROFILE

Dr. Deepika Chaudhary is working as an associate professor at Department of Computer Applications, CUIET, Chitkara University Punjab. She is a Phd in Computer Science has more than 20 years of teaching experience. Her area of Interest include Data Mining, Business Intelligence, Software Engineering and Quality Assurance.

Dr. Jaiteg Singh is a Phd in Computer Science with 14 years of experience in Research, development and training, academics at Institute of Higher Technical Education. Areas of expertise are Software Engineering, Business Intelligence, data and opinion mining, cartography, curriculum design, pedagogical Innovation and management. Areas of Interest includes sustainable software enginnering, Education Technology, offline education system and cloud computing.

Dr. D. P. Kothari is PhD in Electrical Engineering.. Earlier Dr. Kothari served as Vice Chancellor, VIT, Vellore, Director in-charge and Deputy Director (Administration) as well as Head in the Centre of Energy Studies at Indian Institute of Technology, Delhi and as Principal, VRCE, Nagpur. Dr. Kothari, who is a recipient of the most Active Researcher Award, has published and presented 812 research papers in various national as well as international journals, conferences, guided 50 Ph.D scholars and 68 M. Tech students, and authored 50 books in various allied areas. Dr. Kothari is a Fellow of the National Academy of Engineering (FNAE), Fellow of Indian National Academy of Science (FNASc), Fellow of Institution of Engineers (FIE), Fellow IEEE ,Hon. Fellow ISTE and Fellow IETE. On 20th April 2016 he received 'Living Legend' Award in Chennai Conference. Currently Dr. Kothari is with Wainganga College of Engineering & Management, Nagpur serving as Director Research .