OWL/XML PARSER

Aditya Vijay Jogalekar[1], Sharath Maligera Eswarappa[2] and Tejas Morbagal Harish[3]

[1] Universität Bonn, Germany
s6adjoga@uni-bonn.de
[2] Universität Bonn, Germany
s6shmali@uni-bonn.de
[3] Universität Bonn, Germany
s6temorb@uni-bonn.de

Abstract

The acronym OWL stands for Web Ontology Language. Since 2004 OWL is a W3C recommended standard for the modeling of ontologies, and since then has seen a steeply rising increase in popularity in many application domains. OWL documents are used for modeling OWL ontologies. Different syntaxes have been standardized in order to express these. One of them is based on OWL XML syntax and is somewhat more readable for humans. This document demonstrates the use of OWL/XML Parser to read OWL files in OWL XML syntax and to build OWLAxioms out of it. These OWLAxioms are represented as Resilient Distributed Datasets(RDD)[2]. The RDD contain representation of OWL dataset as string based representation of the parsed OWLAxioms from the OWL File using the OWL API.

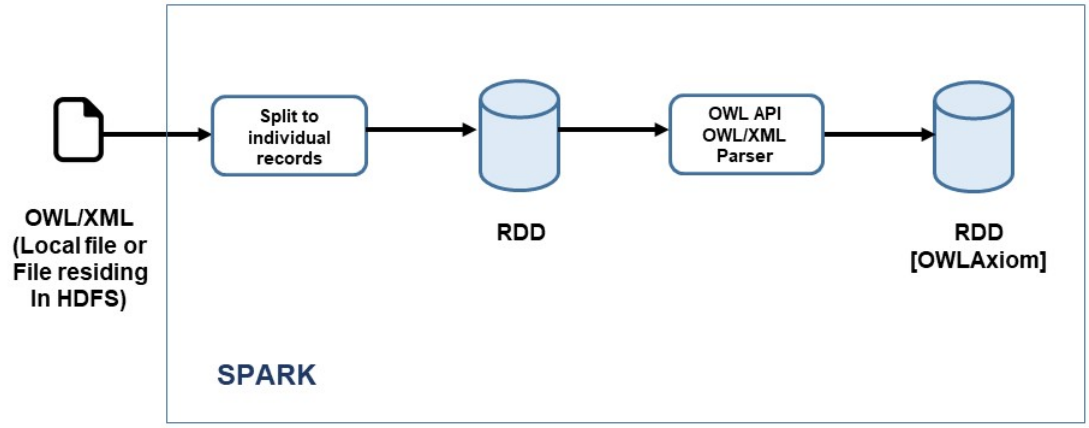
1 Problem Definition

Due to the huge amount of ontology graph present in world wide web, it is a challenging task to process and store those ontology graph. Having systems able to handle such massive data is needed. Existing parsers unable to deal with large ontologies. The aim of the project is to implement OWL/XML parser which is able to parse large ontologies written in OWL/XML format and convert it to RDD of OWLAxioms.

2 Approach

The approach for building the OWL/XML Parser is as follows:

The OWL file is processed using HadoopXMLStreams to process begin and end tags of the OWL Ontologies to identify classes, properties and individuals. The Spark uses HadoopXML-Streams, processes and reads the XML data as String type RDD's and using OWL API it is parsed to OWL Axioms. The OWL Axioms returns each record as RDD containing the class, datatype property and objectproperty.



The above image demonstrates the flow in which OWL/XML Parser is approached and implemented as explained in the below section.

2.1 Data Structures

StreamXmlRecordReader: to mark the begin and end of OWL Record.
sparkContext.hadoopRDD: An RDD that provides core functionality for reading data stored in Hadoop

2.2 Datasets

LUBM Benchmark[3]

3 Implementation

For Implementation of the above workflow Scala Programming language and Spark Framework are used. Below are the major classes/functions that were used for parsing OWL/XML.

3.1 Major Functions

This section includes the major functions used in the implementation of OWL/XML Parser.

3.1.1 OWLXMLSyntaxOWLExpressionsRDDBuilder:

1. We use hadoop streaming StreamXmlRecordReader to process OWL/XML file.
2. Create job configuration for hadoopXML streaming and set the begin and end tags for each of owlClass, owlDatatypeProperty, owlObjectProperty and prefixes.

```
val owlClassConfig = new JobConf()

owlClassConfig.set("stream.recordreader.class", "org.apache.hadoop.streaming.StreamXmlRecordReader")

owlClassConfig.set("stream.recordreader.begin", "<owl:") // start Tag
```

```
owlClassConfig.set("stream.recordreader.end", "</owl:Class>") // End Tag
```

3. we set up the data path for Hadoop XML streaming processing.

```
org.apache.hadoop.mapred.FileInputFormat.addInputPaths(owlClassConfig, filePath)
```

4. Read XML record and save in RDDs as block.

```
val owlClassRecord: RDD[(Text, Text)] =

spark.sparkContext.hadoopRDD(owlClassConfig,classOf[org.apache.hadoop.streaming.StreamInputFormat],

classOf[org.apache.hadoop.io.Text], //class for the key

classOf[org.apache.hadoop.io.Text]) //class for the value
```

5. Convert each of the OWl/XML Record block to string datatype

```
val rawOwlClassRDD: RDD[String] =

owlClassRecord.map { case (x, y) => (x.toString()) }
```

6. Each of the converted string OWL/XML record block is then parsed using OWLAPI[1]

3.1.2 RecordParse:

Each OWL/XML record block is taken as input,a ontology is built and OWLAxioms are generated. The function returns all OWLaxioms for each of OWLclass, OWLDatatypeproperty and objectypeproperty.

```
val manager = OWLManager.createOWLOntologyManager

Val axiomResult= ontology.axioms().toScala[Stream].toSet.toList.asJavaCollection


OWLAxioms.addAll(axiomResult)
```

4 Results and Discussion

To evaluate our implementation we have used the dataset univ-bench.owl provided by The Lehigh University Benchmark (LUBM)which is a benchmark developed for evaluating the performance of Semantic Web repositories with respect to extensional queries over a large dataset that commits to a single ontology

(a) Input: OWL Class

```
<owl:Class rdf:ID="AdministrativeStaff">
 <rdfs:label>administrative staff worker</rdfs:label>
 <rdfs:subClassOf rdf:resource="#Employee" />
</owl:Class>
```

Output: OWLClass Axioms

```
[Declaration(Class(<http://swat.cse.lehigh.edu/onto/univ-bench.owl#AdministrativeStaff>)),
SubClassOf(<http://swat.cse.lehigh.edu/onto/univ-bench.owl#AdministrativeStaff>
<http://swat.cse.lehigh.edu/onto/univ-bench.owl#Employee>), AnnotationAssertion(rdfs:label
<http://swat.cse.lehigh.edu/onto/univ-bench.owl#AdministrativeStaff> "administrative
worker"^^xsd:string)]
```

(b) Input: OWl DataProperty

```
<owl:DatatypeProperty rdf:ID="age">
  <rdfs:label>is age</rdfs:label>
  <rdfs:domain rdf:resource="#Person" />
</owl:DatatypeProperty>
```

Output: OWLDatatypeProperty Axioms

```
Declaration(DataProperty(<http://swat.cse.lehigh.edu/onto/univ-bench.owl#age>)),
DataPropertyDomain(<http://swat.cse.lehigh.edu/onto/univ-bench.owl#age>
<http://swat.cse.lehigh.edu/onto/univ-bench.owl#Person>), AnnotationAssertion(rdfs:label
<http://swat.cse.lehigh.edu/onto/univ-bench.owl#age> "is age"^^xsd:string)
```

(c) Input: OWL ObjectProperty

```
<owl:ObjectProperty rdf:ID="advisor">
  <rdfs:label>is being advised by</rdfs:label>
  <rdfs:domain rdf:resource="#Person" />
  <rdfs:range rdf:resource="#Professor" />
</owl:ObjectProperty>
```

Output: OWLObjectProperty Axioms

```
AnnotationAssertion(rdfs:label <http://swat.cse.lehigh.edu/onto/univ-bench.owl#advisor>
"is being advised by"^^xsd:string),
ObjectPropertyRange(<http://swat.cse.lehigh.edu/onto/univ-bench.owl#advisor
<http://swat.cse.lehigh.edu/onto/univ-bench.owl#Professor>
```

5 Conclusion and Future Work

Currently,Sansa OWL Spark supports Functional Syntax and Manchester Syntax. The proposed project will support parsing of OWL/XML syntax.

The project can be extended to build OWL axioms from other RDF formats like Turtle or N-Triples.

6 Project Timelines

Week	Activity	Person Responsible
Week1-4	Introduction Classes to Scala,Spark and Sansa Framework	Sharath,Aditya,Tejas
Week 5	Project Assingment	Sharath,Aditya,Tejas
Week 6	First Presentation	Sharath,Aditya,Tejas
Week 7	Discussion with Mentor and Understanding OWL	Sharath,Aditya,Tejas
Week 8	Understanding Sansa Implementations of other OWL formats	Sharath,Aditya,Tejas
Week 9-10	Streaming to split xml file intoindividual Records	Tejas
Week 10-11	Converting individual Records into OWL Expressions	Aditya
Week 11-12	Axiom generation from OWl expressions	Sharath
Week 13-	Project Report	Aditya,Sharath,Tejas

References

[1] University of Manchester. Owl api. Available at `http://owlapi.sourceforge.net/`.

[2] Apache Spark. [online]. Available at `https://spark.apache.org/docs/latest/rdd-programming-guide.html`.

[3] Lehigh University. Lubm benchmark. [online]. Available at `https://github.com/rvesse/lubm-uba`.