Semantic Search based on Ontology Alignment for Information Retrieval

Manju Mony
M.E. Student (Comp Engg.)
K. J. Somaiya CoE
Vidyavihar, Mumbai

Jyothi M. Rao
Dept. of Computer Engineering
K. J. Somaiya CoE
Vidyavihar, Mumbai

Manish M. Potey
Dept. of Computer Engineering
K. J. Somaiya CoE
Vidyavihar, Mumbai

ABSTRACT
Traditional search is keyword based search and does not focus on relationship between the words. In semantic search, the search is performed on basis of the meaning of the terms and concepts. The semantics i.e. meaning is expressed through structured knowledge representation or Ontologies. Resource Description Framework (RDF) is used as the data model and SPARQL query language is used to query the RDF data. Currently, there is lot of RDF based data set, e.g. Dbpedia, Freebase, Geonames etc. Semantic search is gaining popularity in recent times. As systems cannot exist in isolation but need to interact with each other, complex systems may require integration of multiple systems. Semantic search systems may take input from heterogeneous systems, in which case, the common and shared entities have to be identified and mapped properly. Also, in semantic search, user has to enter query in formal query language SPARQL, which is quite difficult to learn and use for laymen. We are presenting herewith, an application in which we have constructed multiple ontologies that are inherently unique but are related to each other. The system performs ontology alignment to allow for inter-operation between them. Also, it provides a natural language query (NLQ) interface. It converts the Natural Language (NL) input to SPARQL query. It answers queries across these multiple ontologies and abstracts them as a single linked unit. Currently, it is working for simple as well as some type of complex queries.

General Terms
Semantic search, Ontology, Natural Language Processing (NLP)

Keywords
Domain based ontology, Ontology alignment, OWL, RDF, SPARQL, Triple store

1. INTRODUCTION
As the use of IT grows in leaps and bounds, systems are becoming more and more complex, composed of multiple sub-systems and system integration is of the utmost importance.

Healthcare and medical insurance schemes require linkage between diverse systems such as e-governance, healthcare and insurance. There are shared entities in them. For example, e-governance systems have various healthcare schemes, which are implemented and availed of in healthcare. Also medical insurance field requires accurate and validated data from healthcare industry. Thus, these systems exchange data with each other and are inter-dependent.

The advantage of linking these together to form a single system, is that we can integrate multiple knowledge bases and co-relate the data with each other. We can check the flow of data between these heterogeneous systems using a shared vocabulary.

A semantic search system can be used to describe, link and query the systems. Such a system should be able to answer queries from all the linked systems such as

- Show me the count of citizens by village and by scheme.
- Show me all the citizens registered for various schemes.
- List all the statistics for all states.

The purpose of this work is to use various concepts in semantic technologies to construct such a system. The combination of following semantic technologies like

- Ontologies i.e. shared vocabulary,
- RDF data model,
- SPARQL query language,
- Ontology Alignment to find matching concepts,
- Guided Semantic Search to locate various concepts
- Mapping of NL to SPARQL

are used in developing this linked system.

1.1 Ontology
Ontology contains the concepts of a domain i.e. the domain vocabulary. In this system, three ontologies are identified and constructed. They are

- E-governance citizens and schemes ontology,
- Hospitalization details ontology and
- Insurance claims ontology.

The components of an Ontology are:

- Classes – which represent the concepts
- Relations – which represent the relation between classes
- Attributes – which represent the properties of the classes
- Individuals – Instances of the classes

The steps in Ontology construction are:

- Determine Domain and Focus
- Consider Reuse
- Development of a Terminology
- Development of Classes and Class Hierarchies
- Definition of Properties
- Definition of Property Constraints
- Definition of Individuals

1.2 Semantic Search

In the words of Tim Berners-Lee, the founder of the Semantic Web - "The Semantic Web is an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in co-operation". [1] Semantic Search uses Semantic Web technologies to perform search and information retrieval.

Firstly, semantics pertains to meaning, rather than syntax. Traditional search is keyword based, whereas in semantic search, the search is performed on basis of the meaning of the terms and concepts. For this, the semantics i.e. meaning is expressed through structured knowledge representation or Ontologies, which is the data model.

Secondly, in semantic search, the content can be read and interpreted by machines. For this, the data is expressed in Resource Description Framework (RDF) language, in which statements are stored in triple i.e. Subject-Predicate-Object format. RDF is the language of the Semantic Web just as HTML is the language of the current web. [2]

1.3 Ontology Alignment

Ontologies can be developed for different systems. However, these resources may be having or sharing common entities. Ontology Alignment helps to identify the common resources between one or more source ontologies. The similar entities are labeled using owl:equivalentClass and owl:equivalentProperty Web Ontology Language (OWL) constructs.

1.4 Natural Language Interface

Ontology helps to identify the concepts (classes) and relationships (properties) of a domain. The actual data is stored in the form of RDF triples i.e. Subject-Object-Predicate. The RDF data is queried by using SPARQL Query language, which can search and locate the required RDF data. To access a RDF database, an individual has to master the SPARQL language, which is a new technology. If a natural language query interface is provided, then it will help the user to query the system with ease. First, the application has to translate the Natural Language Query into SPARQL query, which will be fired on the RDF data, to retrieve the required triples. A domain independent Natural Language Interface helps in efficient maintenance and portability.

1.5 Guided Semantic Search

By displaying the concepts of the ontology to the user such as the classes, data properties (attributes having value as a literal) and object properties (attributes having value as other attributes i.e. relationships between classes), a user can understand the scope of the data in the RDF database, and thus formulate a query, which can retrieve appropriate records. [3]

1.6 Triple Store

Just as a relational database stores the tables and relational data, a triple store is used to store RDF triples. A triple store thus enables to store the data in a persistent state as well as safeguards the data.

The rest of the paper is organized as follows. Section 2 contains a review of work done in previous related systems. Section 3 explains the flow of the proposed system. Section 4 details the Experimental Setup. Section 5 displays the Results. Section 6 concludes the paper.

2. RELATED WORK

FREyA, NLP-Reduce and AutoSPARQL are some of the Natural Language Interfaces developed for Semantic Search systems. NLP Reduce is a Natural Language Interface for semantic search. It is naive and domain-independent. It uses Ginseng tags to annotate the RDF data. It uses only a subset of NLP techniques such as stemming and synonym expansion. [4] FREyA identifies Potential Ontology Constructs (POC) from the syntactic parse tree. It then tries to map the POC to matching Ontology Concepts (OC). In case of overlaps, it issues user clarification dialogues. [5] AutoSPARQL allows the users to query the system without any knowledge of the underlying ontology. It uses supervised machine learning to learn the concepts and user queries and improve the performance of the system. [6]

Fernandez et al have suggested an information retrieval model based on ontology. Instead of keywords which appear in the documents, the inverted index stores semantic meanings associated with the documents which are called as annotations. Users can express their query in natural language, using a question-answering system called as Power-Aqua. The system translates NLQ to ontology terms forming SPARQL query. The SPARQL query returns the list of matching semantic entities. The inverted index is referred to obtain list of relevant documents. These are ranked by computing semantic similarity alogorithm which is a modification of vector space model. [7]

Rashmi Chauhan, Rayan Goudar et al have developed a domain ontology based semantic search for efficient information retrieval through automatic query expansion. This work performs extraction of concepts from the natural language query and then later performs query expansion by using Wordnet API to construct an enhanced query. The Web pages of sports domain are converted to RDF and SPARQL queries are fired to retrieve the data. Precision and recall of the information retrieval is calculated with and without query expansion. [8]

Chunfei Zhang and Zhiyi Fang have developed a model for electronic medical records (EMR) based on ontology. Construction of domain ontology for EMR is detailed. A mapping algorithm and similarity algorithm to achieve expansion of the semantic concept are presented to improve the recall rate and expansion. The architecture of EMR retrieval system is described as a three layer model, which is presentation layer, service layer and conceptual layer. [9]

Swaran Lata et al detail a model for semantic search for e-governance and agriculture. It first processes raw agricultural data obtained from government sources on different web sites using Matlab functions to remove errors. Protégé framework is used to design and develop the agriculture ontology in OWL. [10]

R. Suganyakala et al and Sandhya Revuri et al have constructed a natural language interface for movie ontology for semantic search. [11]

The aim of this work is to build on the current work done in this area and construct a semantic search system for health insurance domain along with a natural language query interface. It involves construction of multiple ontologies, identifying the shared entities in them and performing ontology alignment. It also provides guided semantic search to assist the user in formulating meaningful queries. It focuses on domain independent automatic mapping of NL to SPARQL for ease of use and portability.

3. PROPOSED SYSTEM

The domain ontology based semantic search system will have a structure as detailed below.

3.1 Flow Of the System

The ontologies are constructed and aligned. The input data in various formats have to be converted to RDF data model, based on the vocabulary of the ontologies.

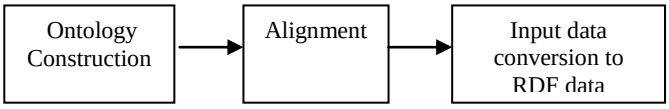


Figure 1: Ontology and data setup

The flow of the system from entering input in NL upto display of results to the user is presented herewith.

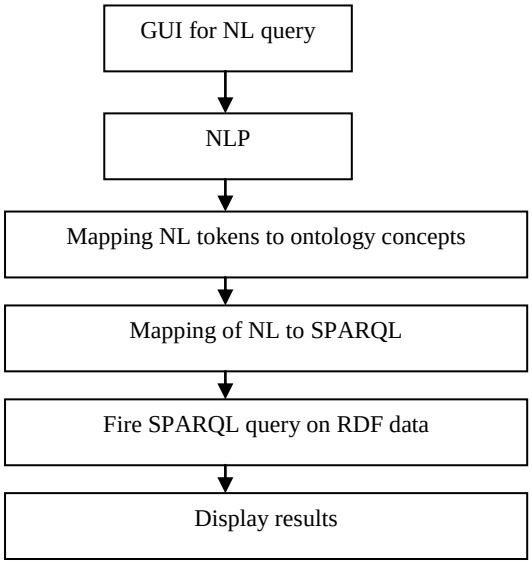
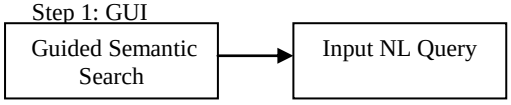


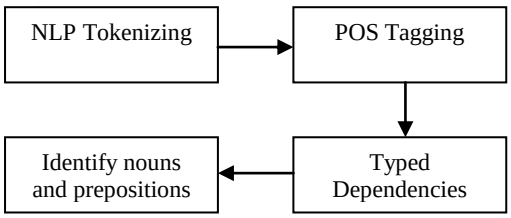
Figure 2: Processing Flow

3.2 Steps In Processing

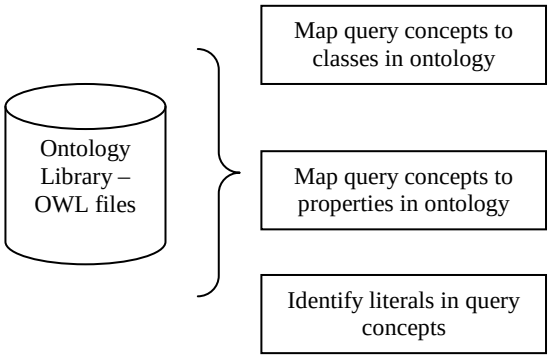
The different processing steps are explained below:-

Step 1: GUI

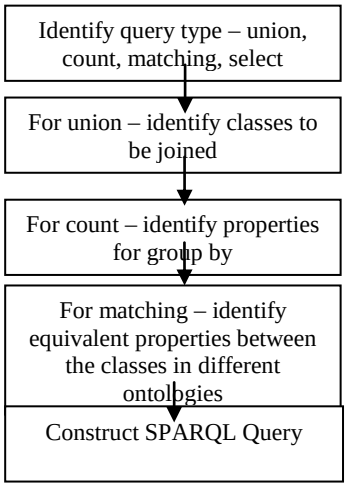




Step 3: Identification of matching concepts between NLQ and ontologies



Step 4: Perform mapping to SPARQL query



Step 5: Fire SPARQL query

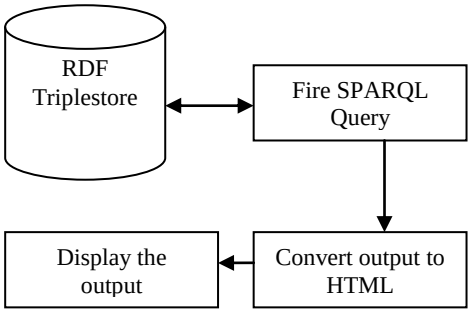


Figure 3: Detailed Steps

4. IMPLEMENTATION DETAILS AND EXPERIMENTS

We are constructing a semantic search for health insurance domain that involves construction of ontologies for various sources such as e-governance, healthcare and insurance, with scope restricted to health insurance related data. The ontologies are aligned with each other by using OWL equivalent classes and properties. A domain-independent natural language interface is constructed to map the Natural Language Query to SPARQL. Guided semantic search explains the concepts of the ontology to the user so that user can understand scope of the system and formulate input queries accordingly. It is working for simple queries containing classes, properties and literals, complex queries containing FILTER, UNION, COUNT, GROUP BY clauses and for alignment queries identifying matching entities across the ontologies. This system is constructed and tested using Health Insurance statistical data provided by IIB.

The web based system is developed using Jena API Semantic web framework, Spring framework and Java.

4.1 Approach

The ontologies are constructed and aligned. The instances in the ontologies are populated by converting excel data to RDF. The rest of data is entered manually through Protege editor. The RDF data is loaded into the Triple store.

The user enters the NL Query on the screen. The Ajax request is fired to the Spring controller. The NL query is mapped to SPARQL query by querying the concepts in the ontology and identifying the mappings between NLQ and the ontology. The NL Query is converted to SPARQL based on query type. The SPARQL query is fired using Jena API and Marklogic API on the triple store. The results are retrieved and converted to appropriate format and displayed on screen.

4.2 Tools and Technologies

Protege ontology editor is used to construct the ontology [12].

Alignment API is used for aligning the ontologies and the function used is String Distance function to identify common sounding names in the ontologies. [13] Google Refine is used for data conversion to RDF format. [14] Marklogic is used as Triple Store. [15]

The above modules 5 to 10 are developed using JEE, Core Java, Jena API. The GUI is developed using Ajax and JQuery. The Natural Language Processing is done using Stanford Core NLP API. Stanford Parser is used for parsing the sentence. [16][17]. Pellet is used for Reasoning.

Marklogic Java API is used to connect from Jena framework to the triple store.

4.3 Domain Ontology construction

This module involves the construction of

- E-governance ontology for healthcare schemes,
- Healthcare hospital details ontology,
- Health Insurance ontology

Oegov US Defence ontologies are reused for e-governance [18].

Classes, data properties (attributes having value as a literal) and object properties (attributes having value as other attributes i.e. relationships between classes) are identified.

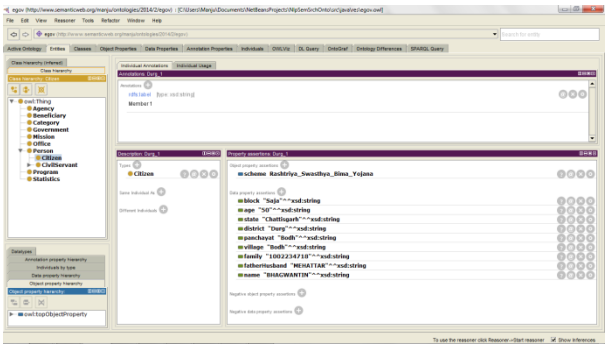


Figure 4: Ontology of e-governance schemes and citizens

Table 1. E-governance ontology for citizens and schemes

Classes	Label	Example of Instance
Government* (Reused from oegov)	States	Maharashtra Govt
Agency*	Ministries	Ministry of Health
Office*	Departments under ministries	Department of Health
Mission*	Missions managed by Departments	National Health Mission
Program	Schemes run by missions	Central Govt. Health Scheme
Beneficiary	Scheme Eligibility	Orange Ration Card
Category	Treatments under scheme	Burns
Person*	Citizens, Civil Servant	Citizen details
Statistics	Health Statistics of various states	Crude Birth Rate, Crude Death Rate

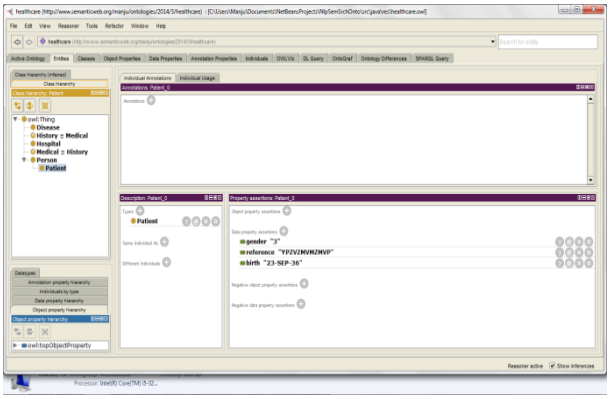


Figure 5: Ontology of healthcare hospital records

Table 2. Hospital ontology

Classes	Label	Example of Instance
Hospital	Hospital details	Hospital details
Person	Patients	Patient details
History	Electronic Health Records	EHR details
Medical	Electronic Health Records in another format	EHR details
Disease	Diseases	Disease codes

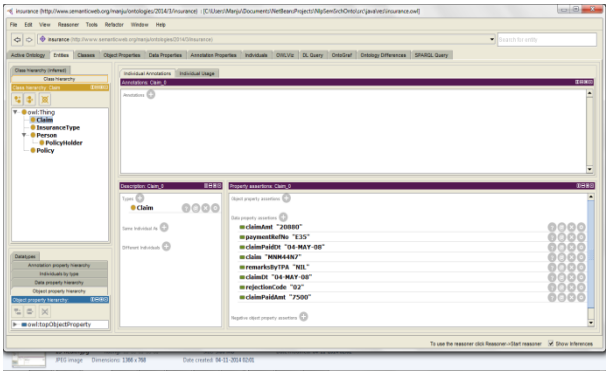


Figure 6: Ontology of health insurance

Table 3. Insurance ontology

Classes	Label	Example of Instance
Types	Insurance Types	Health Insurance
Holder	Policy holders	Policy holder details
Policy	Policies of holders	Policy details
Claim	Claims of holders	Claim details

4.4 Ontology Alignment

Alignment API is used to identify matches between the ontologies and the function used is String Distance function to identify common sounding names in the ontologies.

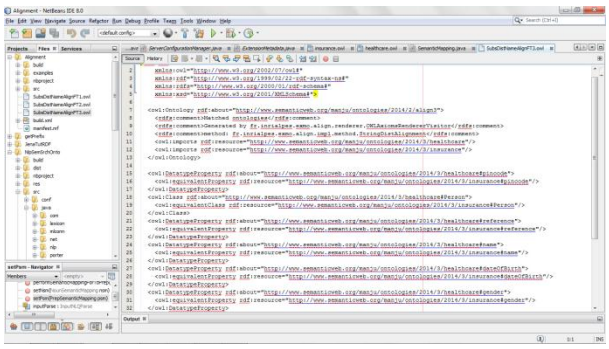


Figure 7: Automatic alignment of Equivalent classes and properties of Person class of healthcare and corresponding class in insurance ontology

4.5 RDF Data Conversion

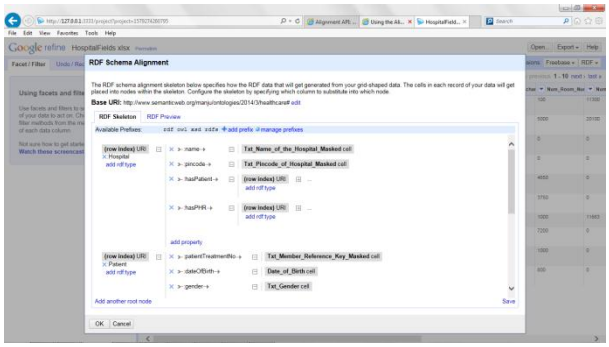


Figure 8: Conversion of health insurance data from excel into corresponding RDF in Google Refine converter tool

4.6 Triple Store

After data is converted into triples in RDF format, the triples will be stored in the Triple store.

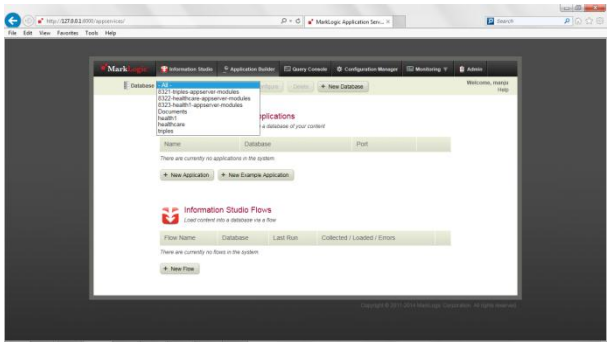


Figure 9: Marklogic Triple store containing database

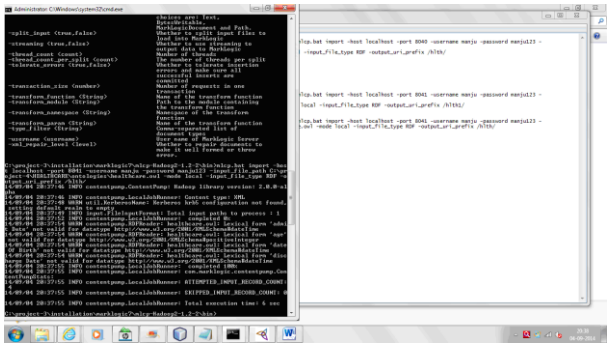


Figure 10: Bulk Loading of RDF triples into Marklogic using Marklogic Content Pump

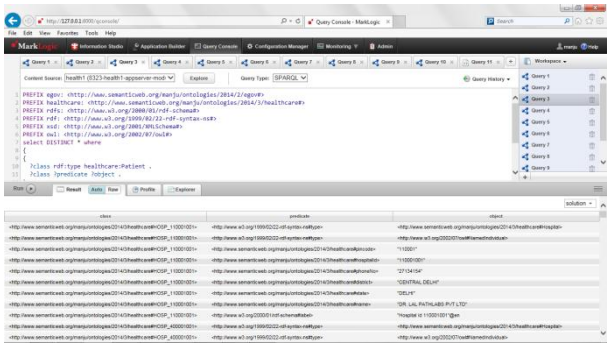


Figure 11: Marklogic Console screen for SPARQL queries

4.7 User Interface

This module accepts the Natural Language query and returns the results to the user on screen. The concepts in the ontology are displayed to the users to enable Guided Semantic Search.

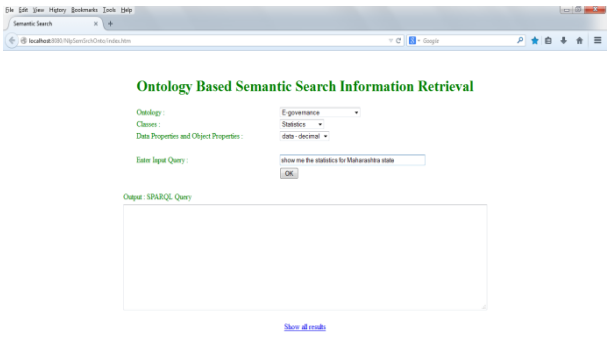


Figure 12: Guided Semantic Search and NL Input Query

4.8 Query Handling

This module will perform Natural Language Processing on the query. It will perform tokenizing, identification of Parts of Speech (POS) and dependencies between the words.

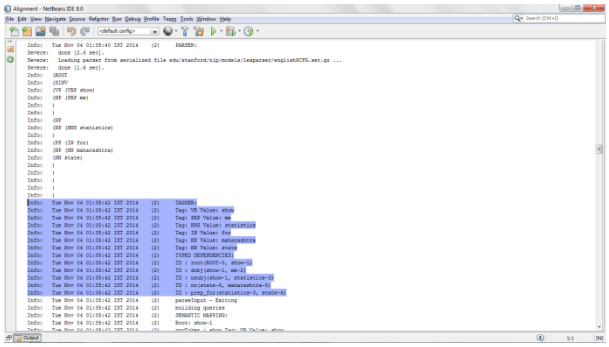


Figure 13: Identifying POS, Typed Dependencies, nouns and prepositions

4.9 Reasoning

This module performs reasoning on the data to identify the hidden relationships and draw inferences.

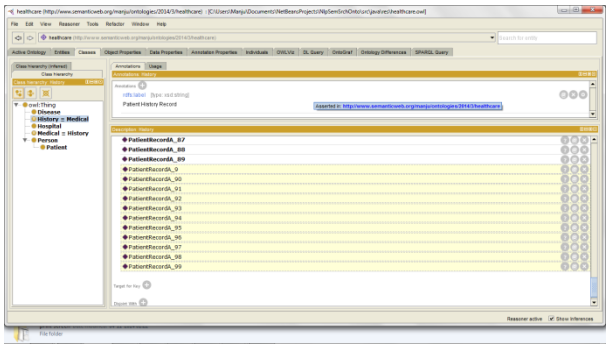


Figure 14: Pellet Reasoner inferring records of equivalent classes History and Medical in healthcare ontology

4.10 Mapping NL to SPARQL

This module performs a domain independent conversion from Natural Language Query into SPARQL. The concepts in the NL Query are mapped to the concepts in the ontologies.

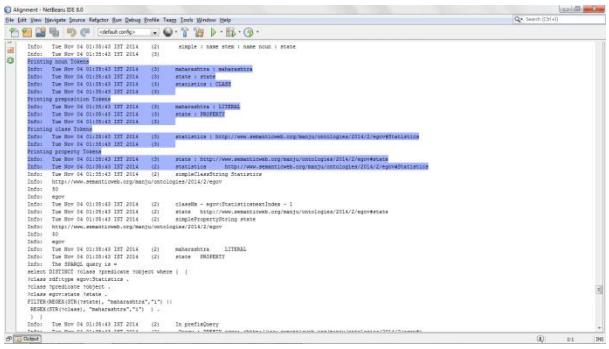


Figure 15: Mapping of query concepts to ontology concepts

4.11 Semantic Search Engine

The Semantic search engine is used for searching the RDF data. The SPARQL query, after mapping, from NL, is fired on the Triple Store.

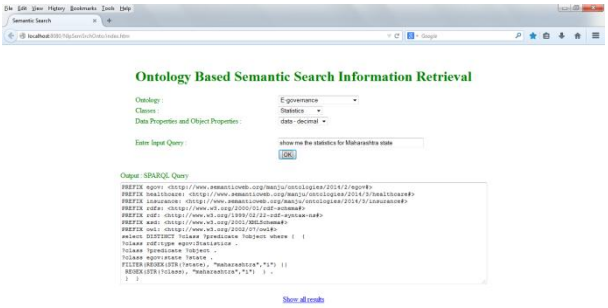


Figure 16: Construction of SPARQL Query

4.12 Results And Processing

This module transforms the results from triple store output format and returns them to the user.

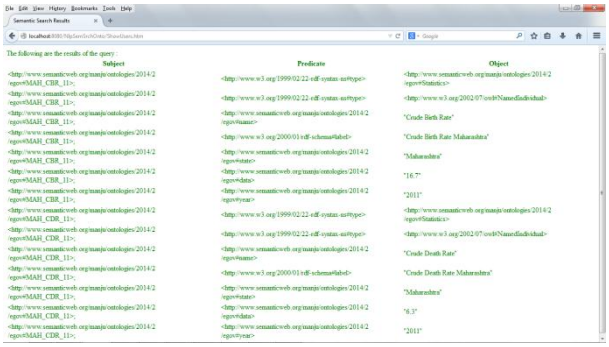


Figure 17: Display of results as Uniform Resource Identifiers (URIs) in Subject-Predicate-Object format

5. RESULTS

The system is tested by formulating different types of Natural Language queries. The SPARQL query is evaluated for correctness. The results are compared with the actual triples in the database.

5.1 Queries Handled By the System

Table 4. Types of Queries

Query Type	Natural Language Query	SPARQL Query (Mapped from NL)

Union	show me all hospitals and diseases details	select DISTINCT ?class ?predicate ?object where { {
		?class rdf:type healthcare:Hospital .
		?class ?predicate ?object .
		}
		UNION {
		?class rdf:type healthcare:Disease .
		?class ?predicate ?object .
		} }

Check class instance	show me details of claim	select DISTINCT ?class ?predicate ?object where { { ?class rdf:type insurance:Claim .

	MNM44N7	?class ?predicate ?object . FILTER(REGEX(STR(?object), "mnm44n7","i") || REGEX(STR(?class), "mnm44n7","i")) . } }
check class, property, literal	show me hospitals with pincode NMMCMC	select DISTINCT ?class ?predicate ?object where { { ?class rdf:type healthcare:Hospital . ?class ?predicate ?object . ?class healthcare:pincode ?pincode . FILTER(REGEX(STR(?pincode), "nmmcmc","i") || REGEX(STR(?class), "nmmcmc","i")) . } }
Multiple properties of same class	show me policies with insurer ZMV and tpa ZV	select DISTINCT ?class ?predicate ?object where { { ?class rdf:type insurance:Policy . ?class ?predicate ?object . ?class insurance:tpa ?tpa . ?class insurance:insurer ?insurer . FILTER(REGEX(STR(?insurer), "zmv","i") || REGEX(STR(?class), "zmv","i")) . FILTER(REGEX(STR(?tpa), "zv","i") || REGEX(STR(?class), "zv","i")) . } }

COUNT for class	show me the count of citizens	select (COUNT(?class) AS ?c)
		where
		{
		?class rdf:type egov:Citizen .
		}

COUNT for more than one property	show me the count of citizens by village and by scheme	select ?village ?scheme (COUNT(?class) AS ?c)
		where
		{
		?class rdf:type egov:Citizen .
		?class egov:village ?village .
		?class egov:scheme ?scheme .
		}
		GROUP BY ?village ?scheme
		ORDER BY DESC (?c)

Equivalent classes (Displays records of both medical and history class)	show me medical records with diagnosis H26	select DISTINCT ?class ?predicate ?object where { { ?class rdf:type healthcare:Medical . ?class ?predicate ?object . ?class healthcare:diagnosis ?diagnosis . FILTER(REGEX(STR(?diagnosis), "h26","i") || REGEX(STR(?class), "h26","i")) . } }
Ontology alignment by identifying Equivalent Properties between two classes	Show me matching patients and holders	SELECT ?x ?y ?reference1 where { { ?x rdf:type healthcare:Patient . ?y rdf:type insurance:Holder . ?x healthcare:gender ?gender1 . ?y insurance:gender ?gender2 . FILTER(?gender1 = ?gender2) . ?x healthcare:birth ?birth1 . ?y insurance:birth ?birth2 . FILTER(?birth1 = ?birth2) . ?x healthcare:reference ?reference1 . ?y insurance:reference ?reference2 . FILTER(?reference1 = ?reference2) . } }

Union (different ontologies)	show me all claims and hospitals details	select DISTINCT ?class ?predicate ?object where { {
		?class rdf:type healthcare:Hospital .
		?class ?predicate ?object . }
		UNION {
		?class rdf:type insurance:Claim .
		?class ?predicate ?object .
		} }

The Java program converts these natural language queries to SPARQL, referring the ontologies. Using Marklogic Java API, the converted SPARQL queries are executed against the triple store. All the queries ran successfully except for COUNT which is a SPARQL 1.1 feature and is currently not available in latest Marklogic 7 version. The COUNT query ran successfully in Protégé.

5.2 Parameters For Checking

$$\text{Precision} = \frac{\text{retrieved relevant}}{\text{Total retrieved}}$$

$$\text{Recall} = \frac{\text{retrieved relevant}}{\text{Total relevant}}$$

For queries handled correctly by the system, the Precision and Recall is 100%.

5.3 Knowledgebase Size

Table 5. Knowledgebase Size

Number of ontologies	3
Number of classes	20
Number of data properties	65
Number of object properties	14
Number of instances	1187
Number of triples	40,137

6. CONCLUSION

Domain based ontology is used to define the terminology of this system. This provides a shared vocabulary to its users. Integration of ontologies helps to align the different ontologies and share the data. Alignment of ontologies automatically reduces effort and saves time. By identifying the shared concepts and entities, two disparate data sets can be linked and queried. Publishing of standard vocabularies and its reuse must be encouraged to develop a common framework for users for querying over any domain. Guided Semantic search helps the user to understand the concepts in the domain and vocabulary and formulate a more accurate query. Domain independent mapping of Natural Language Query to SPARQL query assists the user to use the system easily and also provides portability. We are getting good search results for this system, for the type of queries explained above. Currently the system can map simple queries and some complex constructs like COUNT, UNION and querying across aligned ontologies. It can be further enhanced to resolving more complex SPARQL keywords and aggregates like MIN, MAX, date matching, greater than, less than etc. The RDF Datasets on the web are increasing daily as more and more RDF data is added to the web. By uploading our RDF data to the Semantic Web, other systems and users can take advantage of this data by referring the URIs in the system. This can become a part of the Linked RDF data in the Semantic Web.

7. ACKNOWLEDGEMENTS

We wish to thank Insurance Information Bureau of India (IIB), for allowing us to use the health insurance statistics data at
https://iib.gov.in/IRDA/homePageAction.do?method=loadHomePage1. [19]

8. REFERENCES

[1] Tim Berners-Lee, James, The Semantic Web, Scientific American, May 2001, vol. 284, no. 5, pp 34-43.

[2] D. Brickley and R. V. Guha (Eds), RDF Vocabulary Description Language 1.0: RDF Schema, W3C Recommendation, 10 February 2004. Available at http://www.w3.org/TR/rdf-schema/

[3] Khadija Elbedweihy, Stuart N. Wrigley, and Fabio Ciravegna. 2012. Evaluating semantic search query approaches with expert and casual users. In Proceedings of the 11th international conference on The Semantic Web - Volume Part II (ISWC'12), Springer-Verlag, Berlin, Heidelberg, 274-286

[4] Kaufmann, E., Bernstein, A., Fischer, L.: NLP-Reduce: A naive but domain independent natural language interface for querying ontologies. In: Franconi, E., Kifer, M., May, W. (eds.) ESWC 2007. LNCS, vol. 4519, Springer, Heidelberg (2007)

[5] D. Damljanovic. M. Agatonovic. and H. Cunningham. "FREyA : An Interactive Way of Querying Linked Data Using Natural Language, " in Proceedings of the 8th international conference on The Semantic Web. 2011. pp. 125-138.

[6] Lehmann, J., Buhmann, L.: Autosparql: Let users query your knowledge base. In:Proceedings of the 8th Extended Semantic Web Conference on The Semantic Web:Research and Applications - Volume Part I. pp. 63{79. ESWC'11, Springer-Verlag,Berlin, Heidelberg (2011)

[7] Fernandez, M., Cantador, I., Lopez, V., Vallet, D., Castells, P., Motta, E.. Semantically enhanced Information Retrieval: an ontology-based approach. Web Semantics: Science, Services and Agents on the World Wide Web, North America, 9, jan. 2012.

[8] Chauhan, R.; Goudar, R.; Sharma, R.; Chauhan, A., "Domain ontology based semantic search for efficient information retrieval through automatic query expansion," Intelligent Systems and Signal Processing (ISSP), 2013 International Conference on , vol., no., pp.397,402, 1-2 March 2013

[9] Chunfei Zhang, Zhiyi Fang, A New Electronic Medical Record Retrieval System Based on Ontology and Mapping Algorithm, Journal of Information & Computational Science 10:14, 4603-4610, 2013

[10] Swaran Lata, Bhaskar Sinha, Ela Kumar, Somnath Chandra, Raghu Arora, Semantic Web Query on e-Governance Data and Designing Ontology for Agriculture Domain, International Journal of Web & Semantic Technology (IJWesT) , 04(03), 65 – 72, 2013

[11] R. Suganyakala & Dr. R. R. Rajalaxmi , "Movie Related Information Retrieval Using Ontology Based Semantic Search", International Conference on Information Communication and Embedded Systems (ICICES), 2013

[12] Protege Overview, http://protege.stanford.edu/overview/

[13] The Alignment API, http://alignapi.gforge.inria.fr

[14] Open Refine, http://openrefine.org

[15] Marklogic, http://www.marklogic.com

[16] The Stanford Parser: A statistical parser, http://nlp.stanford.edu/software/lex-parser.shtml

[17] Marie-Catherine de Marneffe, Christopher D. Manning, "The Stanford typed dependencies manual" in Revised for Stanford Parser v1.6.2, February, 2010.

[18] Oegov ontologies for e-government, http://oegov.org

[19] IIB, https://iib.gov.in

[20] Maria Keet C., Aspects of Ontology Integration, 2004

[21] Studer R, Benjamins VR, Fensel D, Knowledge Engineering: Principles and Methods, IEEE Transactions on Data and Knowledge Engineering, 25(1-2), pp. 161-197, 1998.

[22] P Hitzler, M Krotzsch, S Rudolph, Foundations of semantic web technologies, Chapman and Hall/CRC, 2010

[23] John Hebeler, Matthew Fisher, Ryan Blace, Andrew Perez-Lopez, and Mike Dean, Semantic Web Programming, John Wiley \& Sons Inc., Chichester, West Sussex, Hoboken, NJ, (2009).

[24] Dr. Harald Sack, Feb 2013, Semantic Web Technologies Course, Courses – Open HPI, Retrieved from https://openhpi.de/course/semanticweb