Ontology Based Information Retrieval System for Academic Library

Amol N. Jamgade and Shivkumar J. Karale
Department of Computer Technology
Yeshwantrao Chavan College of Engineering
Nagpur, India

***Abstract*— Information retrieval system is taking an important role in current search engine which performs searching operation based on keywords which results in enormous amount of data available to the user, from which user cannot figure out the essential and most important information. This limitation may be overcome by a new web architecture known as semantic web which overcome the limitation of keyword based search technique called conceptual or semantic search technique. Natural language processing technique is mostly implemented in QA system for asking user's question and several steps are also followed for conversion of questions to query form for getting an exact answer. In conceptual search, search engine interprets the meaning of user's query and the relation among the concepts that documents contains with respect to a particular domain that produces specific answers instead of giving list of answers. In this paper, we proposed ontology based semantic information retrieval system and Jena semantic web framework in which, user enters an input query which is parsed by Standford Parser then triplet extraction algorithm is used. To all input query, SPARQL query is formed and then it is fired on the knowledge base (Ontology) that finds appropriate RDF triples in knowledge base and retrieve the relevant information using Jena framework.**

Index Terms*— Semantic web, Ontology, Query processing, Information retrieval, RDF, SPARQL, WordNet, Jena API

I. INTRODUCTION

Information retrieval system is taking an important role in current search engine optimization concept. Natural language processing technique is mostly implemented in Question Answering (QA) system for asking user's question and several steps are also followed for conversion of questions to query form for getting an exact answer and hence processing of information on web is mostly restricted to manual keyword searches which results in irrelevant information retrieval [3]. This limitation may be overcome by a new web architecture known as semantic web. In order to overcome the limitation of keyword based search technique, conceptual search technique is implemented [1]. In conceptual search, search engine interprets the meaning of user's query and the relation among the concepts that documents contains with respect to a particular domain. Ontology provides a shared and reusable piece of knowledge about a specific domain, and has been applied in many fields, such as Semantic Web, e-commerce and information retrieval, etc. More and more researchers begin to pay attention to ontology research. Until now, many ontology editors have been developed to help domain experts to develop and manage ontology for example Protégé, OntoEdit, TopBraid. One important benefit is that we can significantly save time and effort by reusing existing ontologies instead of building new ones every time. Another advantage is that heterogeneous system and resources can interoperate by sharing a common knowledge [5]. In the proposed system, the meaningful concept is extracted from user's input query. Using this concept, query expansion is performed [1] that is the query is converted into meaningful format. In the proposed system, input query is converted into a SPARQL query. SPARQL is RDF database language. SPARQL query is then fired on to the RDF database and accesses the relevant information. Search engine performs searching operation based on keywords which results in enormous amount of data available to the user from which he or she cannot figure out the essential and most important information so basic objective of this project is to provide accurate information to a specific question instead of giving list of expected information. QA system using Ontology technique, have higher results precision than the system using the keyword based information retrieval techniques because of the semantics of the keywords.

The objectives of the proposed system are as following:

- To developed ontological database for storage of domain specific knowledge.
- Generating SPARQL query from users input query.
- Semantic search of NL query.
- To retrieve related answer from the domain specific ontology.

II. RELATED WORK

AquaLog (Lopez etal, 2007) is a portable question-answering system which takes queries expressed in natural language and an ontology as input and returns answers drawn from one or more knowledge bases (KBs), which instantiate the input ontology with domain-specific information. AquaLog used shallow parsing and WordNet for converting

978-1-4799-6818-3/15/$31.00 © 2015 IEEE

natural language queries to SPARQL AquaLog adopts a triple-based data model.

Linked Open Data Question Answering (LODQA) System is developed to generate SPARQL queries from natural language, with the goal of providing an easy-to-use interface to search linked open RDF data. LODQA performs linguistic analysis and ontology lookup. For linguistic analysis, LODQA adopts Enju that is trained on English-language questions (Hara et al., 2011). For ontology lookup, LODQA uses OntoFinder3, which searches ontologies in BioPortal for ontology terms.

Semantic Information Retrieval Using Ontology in University Domain in which the developed system retrieves the web results more relevant to the user query through keyword expansion. The results obtained here will be accurate enough to satisfy the request made by the user. The system will be of great use to the developers and researches who work on web. For ranking an algorithm has been applied which fetches more apt results for the user query [2].

III. System Description

In our proposed system, user interface is created in which user enters input query in natural language. Further it processes by Standford parser, for all input queries parse tree that is tree bank structure is constructed by Standford parser. Ontotriples are constructed using Ontology. Then SPARQL query is formed and it is fired on the knowledge bases that finds appropriate RDF triples in knowledge base and retrieve the relevant information using Jena semantic web Framework.

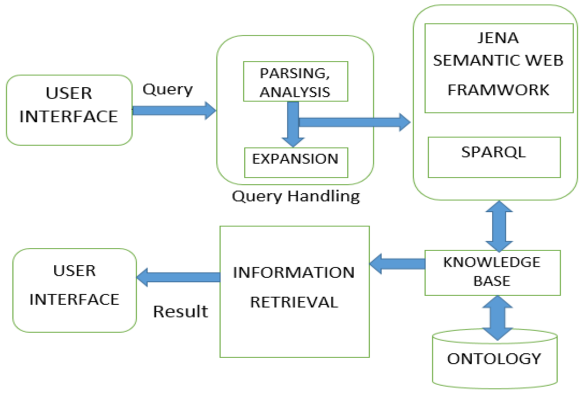


Fig.1. System Architecture

A. Query Parsing and Analysis:

In this phase, the analytical operation of the question is found out. This Analysis is responsible for processing Natural Language Processing (NLP). Query is processed by Standford Parser, for all input queries parse tree that is tree bank structure is constructed. It is a technique to identify the type of a question, type of an answer, subject, verb, noun, phrases and adjectives from the question. Tokens are separated from the question and the meaning is analyzed and the reformulation of question/query is sent to the next stage [7].

B. Reformulation and Classification of Query:

In this phase, the reformulation of query that is further expansion is generated with the help of WordNet or domain specific local dictionary.

C. Knowledge Base:

The Knowledge Base of this proposed system is domain specific. The storage of ontology is the necessary one to retrieve the relevant and correct answer from the knowledge base. In our system RDF database is used which can be easily linked in protégé or TopBraid.

D. Information Retrieval Search Engine:

The user can search answers from ontology. If the concept exists in the knowledge base, the system can answer the question quickly.

·

IV. Methodology

A. Semantic Web:

The next generation intelligent web called the semantic web offers users the ability to work on shared meaningful knowledge representations on the web. Semantic Web creates an artificial intelligence application which will make web content meaningful to computers, thereby unleashing a revolution of new abilities and it intends to support machine-processing capabilities that will automate web applications and services. Agents (software programs) will perform various tasks by communication with other agents and seeking information from web resources [2]. Semantic Web is a vision the idea of having data on the web defined and linked in a way that it can be used by machines not just for display purposes, but for automation, integration and reuse of data across various applications [3].

B. Ontology:

Ontology formally describes a list of terms which represent important concepts, such as classes of objects and the relationships between them. To compare conceptual information across two knowledge bases on web, a program must have a way to discover common meanings and the solution to this is to collect information at a common place called Ontologies. Building Ontologies is divided into three steps: ontology capture, ontology coding and possible integration with existing Ontologies [2]. Ontology life cycle involves steps like specification, conceptualization, formalization, integration, implementation and maintenance [8].

C. WordNet:

WordNet [9] is a lexical database for the English language. WordNet can help a question answering system to identify synonyms. For example, verbs “start” and “begin” will be recognized as synonyms by WordNet. The synonym information can be used to help match a question with an appropriate rule.

D. Resource Description Framework (RDF):

The Resource Description Framework (RDF) [13] is a family of World Wide Web Consortium (W3C) specifications originally designed as a metadata data model. It is a directed, labeled graph data format and its general-purpose is to represent the information in the web. In many applications which are relevant with Natural Language Processing (NLP) or Semantic Web, RDF is broadly used to organize knowledge. It plays an important role in knowledge representation and ontology. An RDF Model consists of a set of triples. A triple include three parts: subject, predicate and object. Its formulation is <subject, predicate, object>. For instance, we can represent "Albert Einstein's given name is Einstein" as a triple <Albert_Einstein, hasGivenName, Einstein>. We can read, write and operate RDF easily by the open source Java Project "Jena".

E. SPARQL:

SPARQL [14] is the query language for RDF data, it issimilar to SQL and widely used in the query processing and inference engine like "ARQ", "Pellet", "Jena" and etc. We can query a triple by any component of the triple. SPARQL supports some constraining queries, optional pattern matching, optional graph pattern along with the operation of conjunctions and disjunctions. We can also make regular expression restriction by the keyword "FILTER". Either the results of SPARQL queries are results set or RDF graphs.

F. Jena API:

Jena API is used for mapping SPARQL query on RDF. Jena is as a number of major subsystems with clearly defined interfaces between them. RDF triples and graphs, and their various components, are accessed through Jena's RDF. Jena stores information as RDF triples in directed graphs, and allows your code to add, remove, manipulate, store and publish that information. RDF API has basic facilities for adding and removing triples to graph and finding triples that match particular patterns. Here you can also read in RDF from external sources whether files or URS and serialize a graph in correctly-Formatted text form. Both input and output support most of the commonly-used RDF syntaxes. The collection of the standards that defines semantic web technologies includes SPARQL-the query languages for RDF, Jena conforms to all of the published standards and, tracks the revision and updates in the under-development areas of the standard. Handling SPARQL, both for query and updates, is responsibility of the SPARQL, API [4].

V. Overview of Implementation

A. Database Used

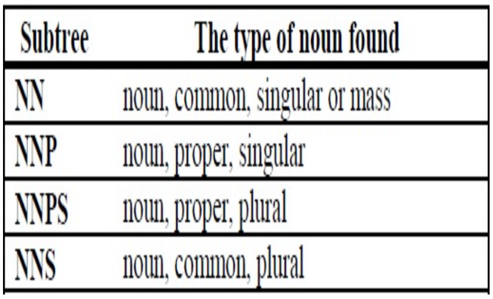
Protege is editor which creates data into RDF format. In our proposed system, Protege is used to create ontology for Academic Library in RDF data format. Academic Library Ontology contains several classes and subclasses. The root node, Academic Library contains classes Subject, Department, Administrator, Book_Title_Name etc similarly the class department has subclasses ComputerScience, Mechanical, Civil etc. Those subclasses are also further divided into classes. Individual instances for those classes and subclasses are created for instance, the class subject has instances operating system, compiler, java etc. Properties for those classes and subclasses are created which used as predicate in RDF for example, book_title_name, bookcodes, rack_number etc are properties of operating system subject book. In this way hierarchical view of Academic Library Ontology is developed then exported into RDF format which are written in XML which shows RDF schema and Knowledge base KB for that schema. KB takes values of properties identified for classes and subclasses [11].

B. Conversion of Input Query to SPARQL Query

User interface in which user enters input query in natural language. Further it processes by Standford parser, for all input queries parse tree that is tree bank structure is constructed. Then triplet that is Subject, Predicate, Object extraction algorithm is used to extract the triplets from tree bank structure. Then SPARQL query is formed [4].

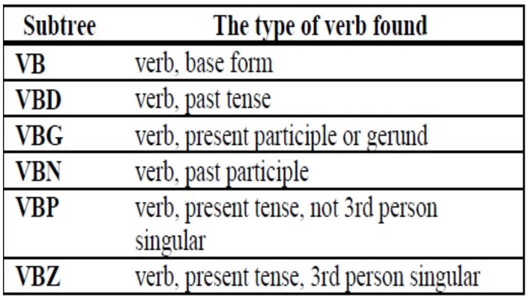
1) Triplets Extraction from Sentences

A sentence (S) is represented by the parser as a tree having three children: a noun phrase (NP), a verbal phrase (VP) and the full stop (.). The root of the tree will be S. Firstly we intend to find the subject of the sentence. In order to find it, we are going to search in the NP subtree. The subject will be found by performing breadth first search and selecting the first descendent of NP that is a noun. Nouns are found in the following subtrees:



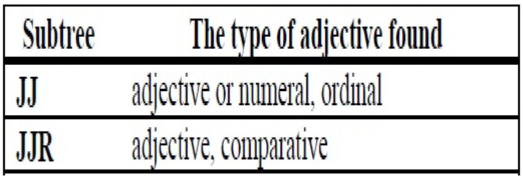
Subtree	The type of noun found
NN	noun, common, singular or mass
NNP	noun, proper, singular
NNPS	noun, proper, plural
NNS	noun, common, plural

Secondly, for determining the predicate of the sentence, a search will be performed in the VP subtree. The deepest verb descendent of the verb phrase will give the second element of the triplet. Verbs are found in the following subtrees:



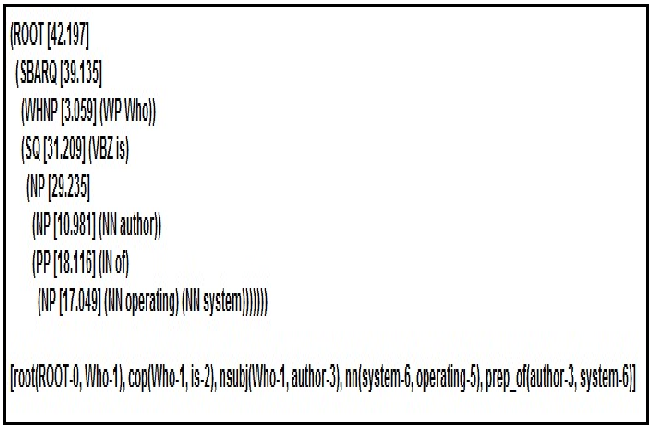
Subtree	The type of verb found
VB	verb, base form
VBD	verb, past tense
VBG	verb, present participle or gerund
VBN	verb, past participle
VBP	verb, present tense, not 3rd person singular
VBZ	verb, present tense, 3rd person singular

Thirdly, we look for objects. These can be found in three different subtrees, all siblings of the VP subtree containing the predicate. The subtrees are: PP (prepositional phrase), NP and ADJP (adjective phrase). In NP and PP we search for the first noun, while in ADJP we find the first adjective. Adjectives are found in the following subtrees:



Subtree	The type of adjective found
JJ	adjective or numeral, ordinal
JJR	adjective, comparative

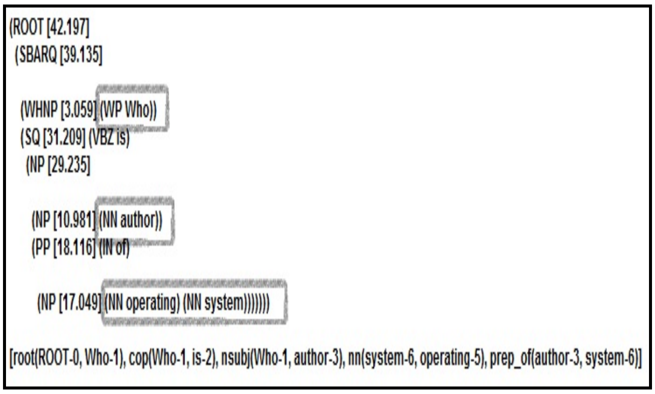
Stanford Parser generates a Treebank parse tree for the input sentence. Figure depicts the parse tree for the input sentence "*Who is author of operating system"*.



```
(ROOT [42.197]
 (SBARQ [39.135]
  (WHNP [3.059] (WP Who))
  (SQ [31.209] (VBZ is)
   (NP [29.235]
    (NP [10.981] (NN author))
    (PP [18.116] (IN of)
     (NP [17.049] (NN operating) (NN system))))))

[root(ROOT-0, Who-1), cop(Who-1, is-2), nsubj(Who-1, author-3), nn(system-6, operating-5), prep_of(author-3, system-6)]
```

Fig.3 The parse tree generated by Stanford Parser

After applying the triplet extraction algorithm presented in follow, we obtain the result presented in following figure

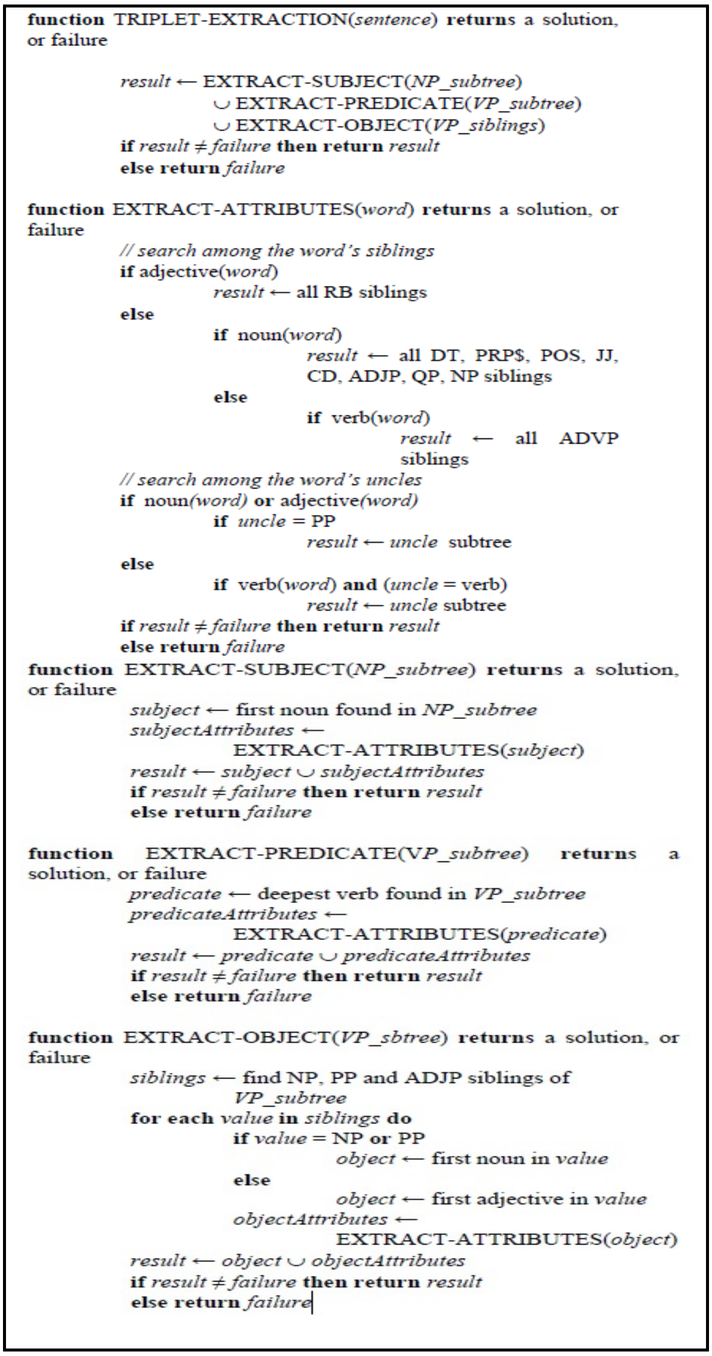


```
(ROOT [42.197]
 (SBARQ [39.135]

  (WHNP [3.059] (WP Who))
  (SQ [31.209] (VBZ is)
   (NP [29.235]

    (NP [10.981] (NN author))
    (PP [18.116] (IN of)

     (NP [17.049] (NN operating) (NN system))))))

[root(ROOT-0, Who-1), cop(Who-1, is-2), nsubj(Who-1, author-3), nn(system-6, operating-5), prep_of(author-3, system-6)]
```

Fig.4 The triplet structure containing the triplet elements with their attributes.

Once we get triplets, SPARQL query can be generated, refer to results. Before generating SPARQL query WordNet or local dictionary can be used to identify synonyms for predicates if matching predicate is not found in RDF database [15].

2) Triplets Extraction Algorithm



```
function TRIPLET-EXTRACTION(sentence) returns a solution, or failure
        result ← EXTRACT-SUBJECT(NP_subtree)
                ∪ EXTRACT-PREDICATE(VP_subtree)
                ∪ EXTRACT-OBJECT(VP_siblings)
        if result ≠ failure then return result
        else return failure

function EXTRACT-ATTRIBUTES(word) returns a solution, or failure
        // search among the word's siblings
        if adjective(word)
                result ← all RB siblings
        else
                if noun(word)
                        result ← all DT, PRP$, POS, JJ, CD, ADJP, QP, NP siblings
                else
                        if verb(word)
                                result ← all ADVP siblings
        // search among the word's uncles
        if noun(word) or adjective(word)
                if uncle = PP
                        result ← uncle subtree
        else
                if verb(word) and (uncle = verb)
                        result ← uncle subtree
        if result ≠ failure then return result
        else return failure
function EXTRACT-SUBJECT(NP_subtree) returns a solution, or failure
        subject ← first noun found in NP_subtree
        subjectAttributes ←
                EXTRACT-ATTRIBUTES(subject)
        result ← subject ∪ subjectAttributes
        if result ≠ failure then return result
        else return failure

function EXTRACT-PREDICATE(VP_subtree) returns a solution, or failure
        predicate ← deepest verb found in VP_subtree
        predicateAttributes ←
                EXTRACT-ATTRIBUTES(predicate)
        result ← predicate ∪ predicateAttributes
        if result ≠ failure then return result
        else return failure

function EXTRACT-OBJECT(VP_sbtree) returns a solution, or failure
        siblings ← find NP, PP and ADJP siblings of
                VP_subtree
        for each value in siblings do
                if value = NP or PP
                        object ← first noun in value
                else
                        object ← first adjective in value
                objectAttributes ←
                        EXTRACT-ATTRIBUTES(object)
        result ← object ∪ objectAttributes
        if result ≠ failure then return result
        else return failure
```

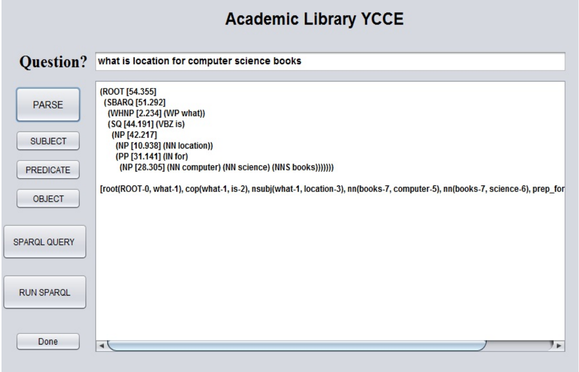
C. Interfacing Using Jena API

Jena API is used for mapping generated SPARQL query as above mentioned on RDF. Jena provides SPARQL API to handle both SPARQL query and their update. Jena stores information as RDF triples in directed graphs, and allows your code to add, remove, manipulate, store and publish that information. A key feature of semantic web applications is that the semantic rules of RDF, RDFS and OWL can be used to infer information that is not explicitly stated in the graph.

For instance, if class C is a sub-class B, and B is a sub-class of A, then by implication C is a sub-class Of A. Jena's interference API provides the means to make these entailed triples appear in the store just as if they had been added explicitly. The Jena API provides a number of rule engines to perform this job, either using the built-in-rules sets for OWL and RDFS, or using application customs rules. In this way Jena API enables the SPARQL query for mapping with RDF database then it is fired on RDF database and retrieves the relevant information performing semantic search into database

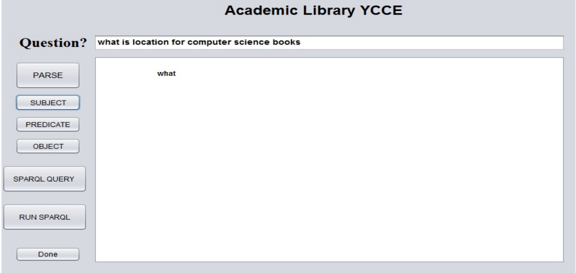
VI. RESULTS

Sample output for query after parsing by Standford parser and then get subject, predicate, object by using triplet extraction algorithm



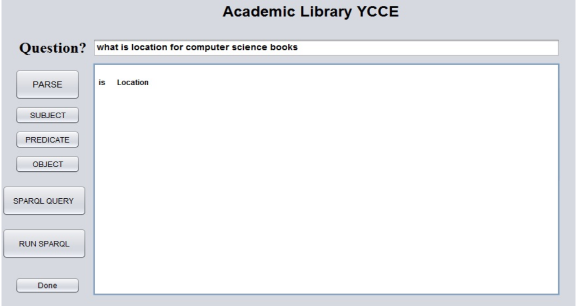
what is location for computer science books

```
(ROOT [54.355]
  (SBARQ [51.292]
    (WHNP [2.234] (WP what))
    (SQ [44.191] (VBZ is)
      (NP [42.217]
        (NP [10.938] (NN location))
        (PP [31.141] (IN for)
          (NP [28.305] (NN computer) (NN science) (NNS books)))))))

[root(ROOT-0, what-1), cop(what-1, is-2), nsubj(what-1, location-3), nn(books-7, computer-5), nn(books-7, science-6), prep_for
```



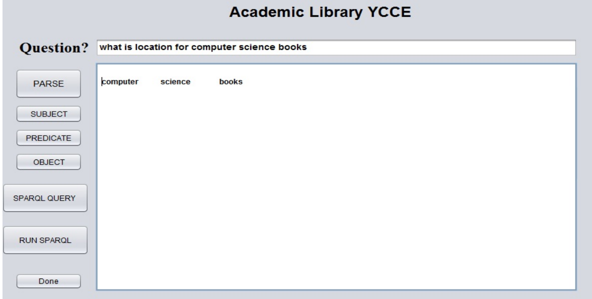
what is location for computer science books

```
what
```



what is location for computer science books

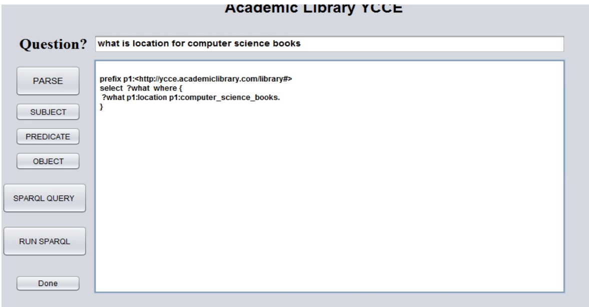
```
is    Location
```



what is location for computer science books

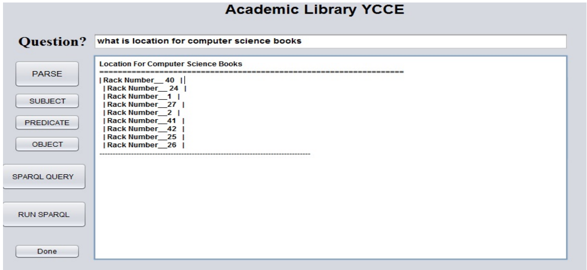
```
computer      science      books
```

Further SPARQL query is generated for above triplets and final relevant answer is retrieved using Jena API



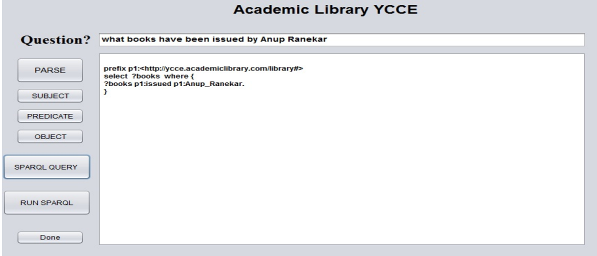
what is location for computer science books

```
prefix p1:<http://ycce.academiclibrary.com/library#>
select  ?what  where {
 ?what p1:location p1:computer_science_books.
}
```



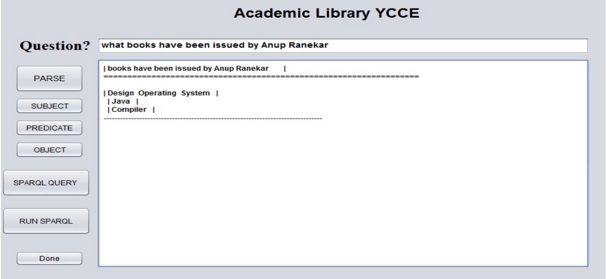
what is location for computer science books

```
Location For Computer Science Books
=====================================================================
| Rack Number__ 40  ||
  | Rack Number__ 24  |
  | Rack Number__1  |
  | Rack Number__27  |
  | Rack Number__2  |
  | Rack Number__41  |
  | Rack Number__42  |
  | Rack Number__25  |
  | Rack Number__26  |
-------------------------------------------------------------------------------
```



what books have been issued by Anup Ranekar

```
prefix p1:<http://ycce.academiclibrary.com/library#>
select  ?books  where {
?books p1:issued p1:Anup_Ranekar.
}
```



what books have been issued by Anup Ranekar

```
| books have been issued by Anup Ranekar      |
=====================================================================

| Design  Operating  System  |
  | Java  |
  | Compiler  |
-------------------------------------------------------------------------------
```

VII. Conclusion.

The proposed system overcomes the limitation of keyword based query handling systems and capable extracting relevant information instead of giving list of answers. Triplet extraction algorithm is worked efficiently to extract triples from sentences. Jena API is used for mapping of SPARQL with RDF database and retrieving the relevant information. Protege editor is used for creating ontology in RDF data format for Academic Library. This system is domain specific but in future the method can be applied for different domains also. This system is currently capable of handling simple queries. Because of partial implemented, further development would be required to answer to all possible types of queries.

References

[1] Rashmi Chauhan, Ryan Goudar, Robin Sharma, Atul Chauhan, "Domain ontology based semantic search for information retrieval through automatic query expansion", Dept. of Computer Science and Engineering GEU Deharadun, India, 2013 International Conference

[2] Sandhya Revuri, Sujata R Upadhyaya, P Sreeniva Kumar, "Using domain ontology for information retrieval", Dept. of Computer Science and Engineering Indian Institute of Technology Madras Chennai- India.

[3] S. Kalaivani and K. Duraiswamy, "Comparison of question answering systems based on ontology and semantic web in different environment", Journal of Computer Science 8 (9): 1407-1413, 2012 ISSN 1549-3636 © 2012 Science Publications

[4] Sven Groppe, Jinghua Groppe, Dirk Kukulenz, Volker Linnemann, "A SPARQL engine for streaming RDF data", Institute of Information Systems(IFIS), University of Lubeck, International IEEE Conference on Signal-Image Technologies and Internet-Based System, 2008

[5] Jinxing Cheng, Bimal Kumar, Kincho H. Law, "A Question answering system for project management application", Journal of Natural Language Engineering, Vol. 1, No. 1, pp. 29-81, 2013.

[6] Rhee S.K., J.Lee, and M.-W. Park, "Ontology-based semantic relevance measure", in Proc.s of the 1st SWW Workshop (ISWC), Korea, 2007

[7] Lakshmi Tulsi R., Goudar R. H., Sreenivasa Rao M., Desai P.D. ,"Domain ontology based knowledge representation for efficient information retrieval", Journal of Information System and Communication Issn:0976-8742&E-Issn:0976-8750, 2012.

[8] Ayesha Ameen, khaleel Ur Rahman Khan, B. Padmaja Rani, "Construction of university ontology", IEEE conference, 2012.

[9] Rachid Ahmed-Ouamer, Arezki Hammache, "Ontology based information retrieval for e-learning of computer science", IEEE conference, 2010.

[10] B. Chandrashekran, John R. Josephson, "What are ontologies and why do we need them? ", IEEE Intelligent System, [J], 1999.PP20-25

[11] Feng Luo and L. Khan, "Ontology construction for information selection", Technical Report, Computer Science Department, University of Texas at Dallas. 2002

[12] Astrova, N. Korda, and A. Kalja, "Storing OWL ontologies in SQL relational databases", INTERNATIONAL JOURNAL OFELECTRICAL, COMPUTER, AND SYSTEMS ENGINEERINGVOLUME 1 NUMBER 4 2007 ISSN 1307-5179

[13] G. Klyne, J. Carroll. Resource Description Framework (RDF): Concepts and Abstract Syntax. W3C Recommendation. (2004). [10 Feb 2004]. http://www.w3.org/TR/rdf-concepts/.

[14] E. Prud'hommeaux, A. Seaborne, Bristoleds, et al. SPARQL Query Language for RDF. W3C Recommendation. (2008). [15 January [2008]. http://www.w3.org/TR/rdf-sparql-query/.

[15] Delia Rusu, Lorand Dali, Blaž Fortuna, Marko Grobelnik, Dunja Mladeniæ, "Triplet extraction from sentence".