Transforming Natural Language Query to SPARQL for Semantic Information Retrieval

Sharmela Shaik [#1], Prathyusha Kanakam[*2], S Mahaboob Hussain[#3], D. Suryanarayana[*4]

[#1]PG Scholar, CSE, Vishnu Institute of Technology
[*2]Assistant Professor, CSE, Vishnu Institute of Technology
[*4]Professor, CSE, Vishnu Institute of Technology Bhimavaram, A.P, India

Abstract— *To retrieve the information in a semantic manner requires a special query language to apply on the huge web and database. In general, the entire user queries will be in the form of natural language to search using the traditional search engine applications and no guarantee that user will satisfy with the outcome results. According to the users, querying the databases in natural language is a very easy method for the desired data but it might be difficult to understand the NL query by a machine. Therefore, this paper clearly explains the procedure and importance of reforming the natural language (NL) query into SPARQL query to apply to the database to retrieve the accurate semantic results. SPARQL is an RDF query language which is a semantic query language used to retrieve data and give precise results. Natural language query is an English sentence interpreted by the computer and appropriate action taken. Thus, in this paper architecture introduced to translate an NL query into SPARQL to retrieve semantic results and compared them with the traditional search engines.*

Keywords — *Information retrieval, NLP, natural language, RDF, SPARQL, semantic Web, semantic search, URIs, tagging, POS.*

I. INTRODUCTION

The Web offers huge amounts of unstructured textual documents that do not have a predefined data model do not fit well into relational tables, product descriptions, and reviews are representative of this type of information. Recently researchers discovered that, unstructured information about 70-80 % of all data in organizations growing fifty times of structured data [1]. The huge amounts of textual data available on the Internet and company intranets are an opportunity and a challenge to a number of applications. It performs on the results of information retrieval (IR) engines and it requires a certain level of understanding of the natural language in which such textual data is given. For this and many other reasons, IR increased use of natural language processing (NLP) methods [2].

NLP used to analyse text, allowing machines to understand how humans speak. This human-computer interaction used in real-world applications like automatic text, summarization, semantic analysis, topic extraction, named entity recognition, parts-of-speech tagging. NLP used for text mining, machine translation, and automated question answering. Searching for information mainly relies on handling a large amount of unstructured data. Archived and streaming data have to be included into the diagnosis to achieve proper results. Varying in data representations and the difficulties of processing unstructured data require additional support for engineers through predefined search queries or diagnostic tools. Search queries have to update to different logical representations. To formulate and perform target search tasks, engineers need additional support from IT experts more frequent [3].

II. WORKING METHODOLOGY WITH LODQA SYSTEM

LODQA (Linked Open Data Question Answering) is a system that takes natural language query as input, and finally produces SPARQL queries as output, together with the answers to them certain SPARQL endpoint [4]. Figure 1 represents the three modules of the system are Graphicator, TermFinder, and GraphFinder.

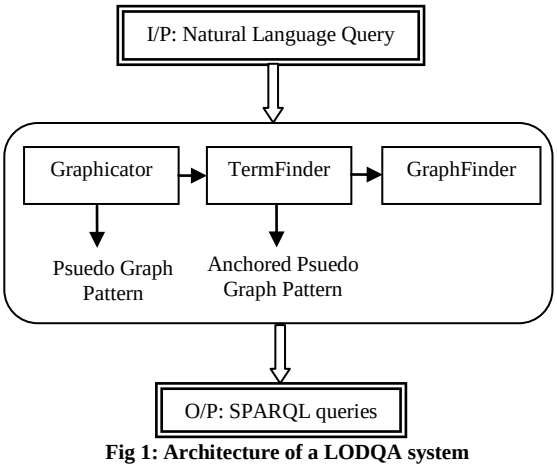

Fig 1: Architecture of a LODQA system

A. Graphicator

It is the first module of the LODQA system. The first module, which handles the NL query, is also responsible for parsing the NL query, and producing a graph representation of the query, and called as pseudo graph pattern (PGP). A PGP contains nodes and relations. Typically, the nodes correspond to the basic noun phrases (BNPs) and the relations to the dependency paths between the BNPs as expressed in

the NL query. Additionally, a PGP specifies which node is the focus of the query, i.e. what the user wants to get as the answer to the query, e.g., "*Post Graduation*" in the following example query.

NLQ: *What is the Post Graduation course after engineering?* For this query, the pseudo graph pattern represented as in Figure 2.

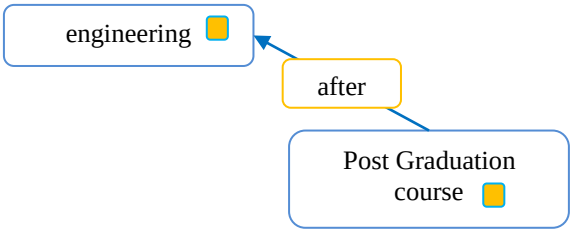

Fig 2: Pseudo Graph Pattern

A PGP is a graph pattern as to search a target RDF graph for sub-graphs that correspond to the PGP. However, it called as a "pseudo" graph pattern, since it is not yet grounded at the target dataset.

B. TermFinder:

It is the second module of the LODQA system. Once Graphicator module has produced a PGP from a given NL query, the TermFinder module is responsible for finding the URIs and values of the nodes in the PGP the URIs and the values have to be actually present in the target dataset. Other than that, there is no chance for the PGP to match with any part of the dataset. For the normalization, each node of the PGP is connected to a URI in the dataset. It is described as the PGP and is represented in the dataset. It is also called as the PGP as an anchored PGP (APGP). A natural language term may be normalized to more than one RDF terms due to ambiguity. Therefore, more than one APGPs may be produced from one PGP through normalization. For instance, consider the NL query.

NLQ:*"What is the Post Graduation course after engineering?"*

Now, the Termfinder is applied to the above NL query, URIs will be formed and represented as three terms *{Post Graduation; engineering; after}.*

C. GraphFinder

GraphFinder is the third module of the LODQA system. For an APGP produced through the Graphicator and the TermFinder modules, the GraphFinder module is responsible for searching the target dataset for corresponding parts, considering possible variations, which may occur in the dataset. To absorb structural discrepancy between the APGP and actual structure in the target dataset, GraphFinder attempts to generate SPARQL queries for all the possible structural variations. Then the SPARQL queries submitted to the target endpoint and collected answers provided to the user.

The graph finder converted into SPARQL query with arguments and relations. These arguments are can be either a primitive type, such as S, N, or NP, or complex, such as S\NP, or NP/N. A forward slash denotes that the argument should appear to the right, while a backslash denotes that the argument should appear on the left. The graph finder has Parts of Speech (POS) grammar notations and some of them represented in Table I.

TABLE I
PARTS OF SPEECH (POS) REPRESENTATION

Tag	Description
DT	Determiner
IN	Preposition subordinating conjunction
NN	Noun, singular or mass
NNS	Noun, plural
NNP	Proper noun, singular
NNPS	Proper noun, plural
WP	Wh –pronoun
VBZ	Verb, 3rd person singular present
VB	Verb, base form

To represent the third module GraphFinder, consider the query.

NLQ:*"What is the Post Graduation course after engineering?"*

The resultant GraphFinder represented in Figure 3 with all the terms separated and denoted with POS tags. Here, every term in the sentence represented with any one of the POS tags. Thus, words in sentence is lemmatized and assigned with proper categories of grammar for easy identification of triplets.

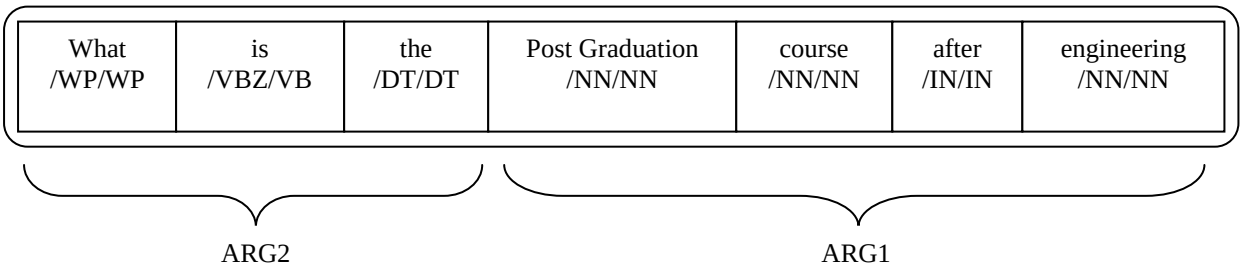

Fig 3: GraphFinder for the query with POS tags representation.

III. EXPERIMENTAL ANALYSIS

The importance of the machine is to provide the relevant information to the users. Here, the machine is the search engine that take inputs in the form of a natural language query. Now, the duty of this search engine is to retrieval according to the query given.

Here, the novelty of this paper is to introduce this SPARQL and RDF database for semantic

information retrieval [5]. The natural language query translates into SPARQL query for retrieving of accurate results from RDF knowledge base [6]. SPARQL also having a select statement to extract the data and results will be displayed in tabular form. The SPARQL is a query language, which traverse through RDF graph.

Therefore, the natural language should convert into a specific syntactical format, i.e., SPARQL. In between the conversion of NL query to SPARQL the sentence undergoes a process for lemmatization and parts of speech (POS) tagging. This procedure will make ready to convert the sentence into SPARQL with important terms by removing all necessary words by this lemmatization process [7]. The main purpose of Natural Language Query is for an English sentence to be interpreted the computer and appropriate action taken and the query is translated into SPARQL query.

This approach is based on natural language processing technology. Natural language is one of the most natural ways of communication and it is difficult to represent complex queries human users will be able to search linked RDF data without having to learn the complex SPARQL language.

TABLE II
PARTS OF SPEECH (POS) REPRESENTATION

Natural Language Query: "What is the Post Graduation course after engineering?"	**Tagging**
	What/WP is/VBZ the/DT post-graduation/NN course/NN after/IN engineering/NN
	Parsing
	(ROOT (SBARQ (WHNP (WP What)) (SQ (VBZ is) (NP (NP (DT the) (NN post) (NN graduation) (NN course)) (PP (IN after) (NP (NN engineering)))))))
	Universal dependencies
	root(ROOT-0, What-1) cop(What-1, is-2) det(course-6, the-3) compound(course-6, post-4) compound(course-6, graduation-5) nsubj(What-1, course-6) case(engineering-8, after-7) nmod(course-6, engineering-8)

Now, consider the natural language query, *"What is the Post Graduation course after engineering?"*, this query will undergo the POS tagging to prepare the SPARQL query ready for retrieval process as shown in Table II.

Then, SPARQL query will be formed by the matching process of the terms from POS and reserved terms in the database as shown below. Namespaces will be added automatically when the procedure starts to construct the query.

prefix dbo: <http://dbpedia.org/ontology/>
prefix dbp: <http://dbpedia.org/property/>
prefix dbpedia: <http://dbpedia.org/resource/>
prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
prefix tto: <http://example.org/tuto/ontology#>
prefix ttr: <http://example.org/tuto/resource#>
prefix xsd: <http://www.w3.org/2001/XMLSchema#>
prefix ont:<http://www.co-ode.org/ontologies/ont.owl#>
SELECT ?thing ?postgraduationcourse ?duration ?entranceexam ?job
WHERE { ?thing rdf:type dbo:engineering . }

To retrieve the semantic results, this SPARQL query is applied on the RDF database. This process is applied to several natural language queries and obtained the relevant SPARQL queries. Thus, by this SPARQL queries users retrieve the efficient and accurate semantic results. Consider the below query.

NLQ: *What is the duration of courses and jobs after B.A?*

Consider the given natural language query is searched on the Google search engine , the results will appear in the tags on the page. Again, user needs to go through the links and search for the desired result [8]. Therefore, direct results may not be retrieved as shown in figure 4.

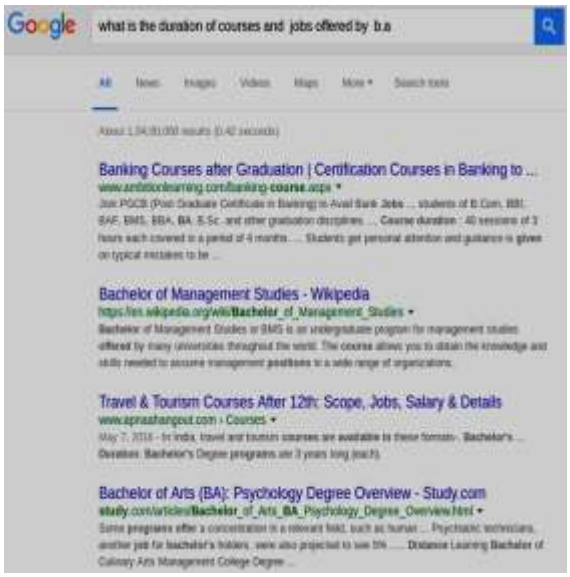

Fig 4: Result screenshot of Google search engine

Since, direct semantics results are not retrieved, the idea is to apply this POS and SPARQL method [9]. Applying the POS and SPARQL method to retrieve the semantics results is as follows.

POS Tagging:

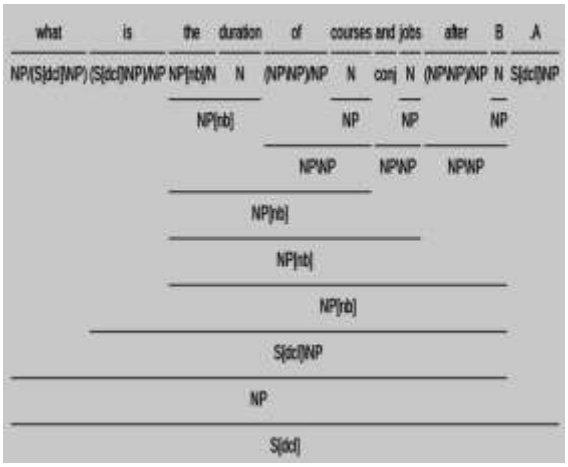

Fig 5: POS Tagging of the given NL Query 1.

From the Figure 5, some words will retrieved and form SPARQL query as below.

SPARQL Query:

SELECT ?Courses ?Duration ?Jobs WHERE {
?thing a ont:Bachelor_of_Arts.
?thing ont:Courses ?Courses .
?thing tto:Duration ?Duration .
?thing tto:Jobs ?Jobs .}

Semantic Results in Triplet form:

The desired results retrieved as in the Table III.

TABLE III
RESULTANT OUTPUT FOR SPARQL QUERY

Courses	Duration in Years	Jobs
Master of Arts	2.0	Teacher, Bank Jobs, Defence Jobs, Railway Jobs
Advertising and Commercial Diploma	2.0	Digital Marketing, Creativity
Event Management Diploma	2.0	Event Manager, Marketing
Bachelor of Journalism	1.0	Press sub-editor, Broadcast Journalist
M.B.A	2.0	Business Analyst, Financial Manager, Entrepreneurship.
Bachelor of Law	5.0	Lawyer, Judiciary Editing Law books

The conferred approach is based on natural language processing technology.

From Table III results displayed in a semantic way and every possible specification declared in a triplet tabular form. Natural language is one of the most natural ways of communication and it is difficult to represent complex queries human users will be able to search linked RDF data without having to learn the complex SPARQL language.

IV. CONCLUSION

The purpose of this work is to provide end-users with a means to query ontology-based knowledge bases using natural language queries and thus the complexity of formulating a query expressed in a graph query such as SPARQL. This work takes place in that field of research that how to interpret a natural language (NL) query and translate it in SPARQL. This paper presented the approach and designed to allow end-users to query graph-based KBs. This approach mainly characterized by the use of query patterns leading the interpretation of the user NL query and its translation into a formal graph query.

ACKNOWLEDGMENT

This work has been funded and supported by the Department of Science and Technology (DST), Govt. of India, under the Grants No. SRC/CSI/153/2011.

REFERENCES

[1] Büttcher, Stefan, Charles LA Clarke, and Gordon V. Cormack. *Information retrieval: Implementing and evaluating search engines*. Mit Press, 2016.
[2] Hirschberg, Julia, and Christopher D. Manning. "Advances in natural language processing." *Science* 349.6245 (2015): 261-266.
[3] Hess, Stephen, et al. "Systems and methods for parsing search queries." U.S. Patent No. 9,317,608. 19 Apr. 2016.
[4] "LODQA : Question-Answering Over Linked Open Data". *Lodqa.org*. N.p., 2016. Web. 24 June 2016.
[5] Prud'Hommeaux, Eric, and Andy Seaborne. "SPARQL query language for RDF." *W3C recommendation* 15 (2008).
[6] Lassila, Ora, and Ralph R. Swick. "Resource description framework (RDF) model and syntax specification." (1999).
[7] Suryanarayana, D., et al. "Stepping towards a semantic web search engine for accurate outcomes in favor of user queries: Using RDF and ontology technologies." *2015 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)*. IEEE, 2015.
[8] Kanakam, Prathyusha, et al. "An Analysis of Exploring Information from Search Engines in Semantic Manner." *International Journal* 4.5 (2014).
[9] Suryanarayana, D., et al. "Cognitive Analytic Task Based on Based on Search Query Logs for Semantic of Semantic Identification." IJCTA, 9(21), 2016, pp. 273-280