ISSN 2250-3153

Implementation of Query Processor Using Automata and Natural Language Processing

Jasmeen Kaur *, Bhawna chauhan *, Jatinder Kaur Korepal **

* Computer Science & Engineering, B.S.Anangpuria Institute of Technology & Management
** Computer Science, Guru Nanak Khasla College

Abstract: The field of query processing has recently been coupled with natural language processing (NLP) that has shown dramatic shift in both research direction and methodology in the past few years. In past, most of the work was done on computational linguistics which drew focus on purely symbolic methods. Recently, more prominence is given to hybrid methods that combine new empirical corpus-based methods, including the use of probabilistic and information theoretic techniques, with traditional symbolic methods. The main purpose of Natural Language Query Processing is to interpret an English sentence and hence a complementary action is taken. Querying to databases in natural language is a convenient method for data access, especially for newbie's who have less knowledge about complicated database query languages such as SQL. This paper emphasise on the structural designing methods for translating English Query into SQL using automata**.**

***Index Terms*:** Natural Language Processing (NLP), automata, SQL, ORACLE

I. INTRODUCTION

Natural Language Processing (NLP) is an approach to analyze text based data or verbal stimulations through both set of theories and set of technologies. It's a very active area of research and development, there is no universal definition that would satisfy everyone, but there ar**e** various aspects, which could be a part of anyone's knowledge. It is a division of AI which includes Information Retrieval, Machine Translation and Language Analysis. The major area of concern for NLP is to enable communication between user and system without storage or memorization of complex commands and procedures. In other words, NLP can help the system understand the languages that humans normally use for conversations among themselves. Natural language may be the easiest to learn and use, it has proved to be the hardest for a real time implementation. Despite of various challenges, natural language processing is widely regarded as a promising and important attempt in the field of computer researches. The main area of concern for most computational linguists is to enhance the capability of the computer to understand and generate natural language so that in due course of time people can deal with their computers through text or speech as if they were addressing another person. The Applications that would be generated with NLP capabilities would be able to fully realize and process natural language along with translating languages correctly and in real time and also extracting information from a variety of data sources on demand of users.

II. RELATED WORK

The very first attempts at NLP database interfaces are just as old as any other NLP research. In fact database NLP may be one of the most important successes in NLP since it began. Asking questions to databases in natural language is a very convenient and easy method of data access, especially for casual users who do not understand complicated database query languages such as SQL. The success in this area is partly because of the real-world benefits that can come from database NLP systems, and partly because NLP works very well in a single-database domain. Databases usually provide small enough domains that ambiguity problems in natural language can be resolved successfully. Here are some examples of database NLP systems:

LUNAR (Woods, 1973) involved a system that answered questions about rock samples brought back from the moon. Two databases were used, the chemical analyses and the literature references. The program used an Augmented Transition Network (ATN) parser and Woods' Procedural Semantics. The system was informally demonstrated at the Second Annual Lunar Science Conference in 1971. [1]

LIFER/LADDER was one of the first good database NLP systems. It was designed as a natural language interface to a database of information about US Navy ships. This system, as described in a paper by Hendrix (1978), used a semantic grammar to parse questions and query a distributed database. The LIFER/LADDER system could only support simple one-table queries or multiple table queries with easy join conditions. [4]

III. SYSTEM DESCRIPTION

To describe a system that supports this kind of query processing, first we need to understand **ORACLE.** Let us consider a database say **ORACLE.** Within this database I have placed certain tables which are properly normalised. If a user wishes to access the database he/she should be aware of SQL statements to read the data from the database. Our view is to create such processor that can eliminate this part and enables the user to access the database in his/her language.

For Example:

Suppose if we want to access a record of a student from

STUDBSAITM table whose name is **"Jasmeen"** then to view the record we need to fire the following query i.e.

SELECT * FROM STUDBSAITM WHERE s_firstname='Jasmeen'?

But a person who is not aware of such queries, will not be able to access the data from the database until he/she knows about the semantics and syntax of firing a query to the database. But by using NLP this can be done very easily.

Thus, the above query can be written as:

What are the details of student whose name is 'Jasmeen'

Both the statements i.e. SQL and NLP will result in same output but being a normal person who have less knowledge about SQL can easily access the database.

IV. NLP SCOPE

- The natural language processing is done on Interrogative Statements; the input statement will be in English.
- Input is taken in the form of questions (wh- like where, who, whom, what etc.)
- A limited Data Dictionary is used where all possible words related to a particular system is stored.
- All the names in the input queries should be in single quotes.
- Data Dictionary used can be- STUD, G_M_INFO and MARKS.

STUD	G_M_INFO	MARKS
roll_no	roll_no	roll_no
Name	name	name
email_id	f_name	semester_no
phone_no	m_name	total_avg
gender	f_no	
	blood_grp	
	m_history	

Fig 1 Tables used in the system

V. NLP ARCHITECTURE

Generally NLP has following steps [6] :-

- Morphological analysis: Individual words are analyzed into their components and non word tokens are separated from the words.
- Syntactic analysis: Linear sequences of words are transformed into structures that show how the words relate to each other.
- Semantic Analysis: The structures created by the syntactic analyzer-are assigned meanings.
- Discourse integration: The meaning of an individual sentence may depend on the sentences that precede it and may influence the meanings of the sentences that follow it..
- Pragmatic Analysis: The structure representing what was said is reinterpreted to determine what was actually meant

The system includes the following modules:

- GUI: Designing the front end or the user interface where the user will enter the query in Natural Language.
- Parsing: Derives the Semantics of the Natural Query given by the user and parses it in its technical form
- Query Generation: After the successful parsing of the statement given by the user, the system generates a query against the user statement in SQL and further gives it to the back end database.
- Data Collection: This module collects the output of the SQL statement and places it in the User Interface Screen as a result form.

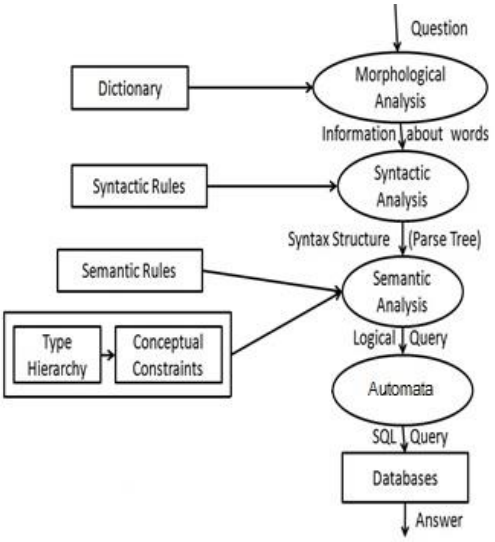


Fig 2 Stages in Natural Language Interpretation Process [6]

VI. AUTOMATA

An automaton is the study of mathematical model of computing called abstract machines or the computational problems that can be solved using them. An Automaton means "self-acting".

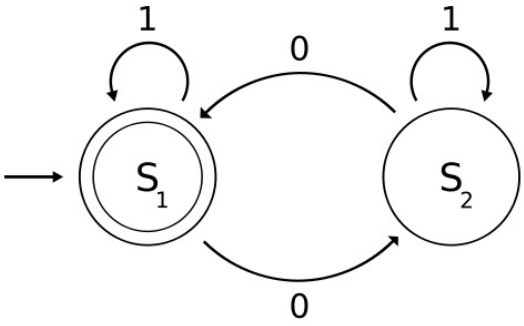


Fig 3 A simple Automata

The figure above illustrates a finite state machine, which belongs to one well-known variety of automaton. This automaton consists of

- States(represented in the figure by circles)
- Transitions (represented by arrows)

As the automata encounter a symbol of input, it jumps to a particular state that is fixed. Every time the same input

encounters on the particular state it will reflect the same transition (or jump), according to its transition function (which takes the current state and the recent symbol as its inputs).

It does play a major role in theory of computation, compiler design, artificial intelligence, parsing and formal verification.

Hence, A deterministic finite automaton M is a 5-tuple, (Q, Σ, δ, q_0, F), consisting of

- a finite set of states (Q)
- a finite set of input symbols called the alphabet (Σ)
- a transition function
  ($\delta : Q \times \Sigma \rightarrow Q$)
- an initial state ($q_0 \in Q$)
- a set of final states ($F \subseteq Q$)

For e.g.-
Let us consider a automata ending with ab below is the figure illustrating the Finite state machine

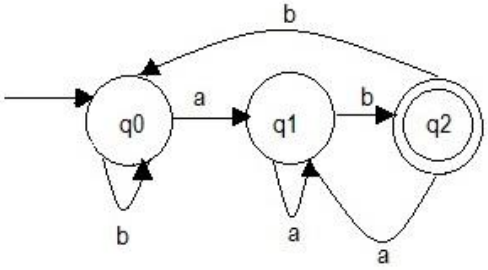


Fig 7 An automata ending with ab

for above example, we have M(Q,Σ,δ,q_0,F)
= M({q0,q1,q2},{a,b},δ,{q0},{q2})
Thus, all the strings except strings ending with ab will be rejected.
And hence we will use the same process to determine the keyword user is entering if the keyword matches the input symbol of a state then it moves to the next state otherwise it remains on that particular stage itself.
If at the end we reach the final state that means correct question has been asked.

VII. PROCESS INTRODUCTION

Firstly, the English input is checked by the NLP, then an automata matches table and attribute names and joins up tables if the query involves more than one table. After that the post-processor can construct the resulting SQL query and output it.

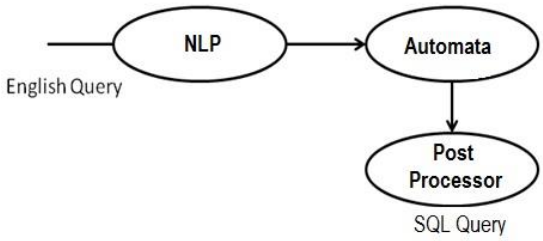


Fig 6 An architecture of System

The question

"What is student name with marks greater than '800'? " is processed by the system as given below.

- The statement is checked through NLP for its technical correction
- The automata then attempts to find whether the question asked is appropriate according to the data dictionary we have.
- This sentence will only have one possible SQL query, so only one result is returned:

SELECT NAME FROM STUD WHERE TOTAL_AVG > 800

- The query is an SQL query.

Translating from one language to another always requires some kind of parsing. Parsing is the process of identifying structure in data. It determines whether the input data has some pre-determined structure and respond accordingly.

VIII. PROPOSED ALGORITHM

1. NLP Processing
2. Tokenization (scanning)
   a. Split the Query in tokens
   b. Pass the tokens to automata
3. Extract patterns using automata
4. Check for the articles and connectors and ignore them
5. Replace matching keyword with proper attribute names
6. Map value for identified attribute and corresponding table
7. Transform it into SQL

Let us consider the above example then the algorithm will work as follows:

- Input :- **"What is student name with marks greater than 800"**
- NLP Processing will check for any mistake in question forming
- Input is passed through following Automata

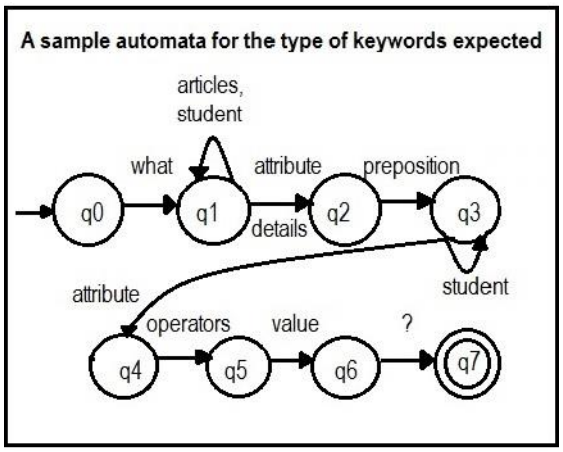


Fig 8 Query Processor Automata Module

And the generation of query by dividing it into tokens and then passing it through the automata in Fig 8 will result in

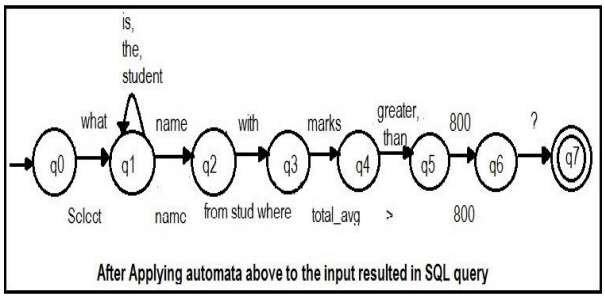


Fig 9 Query Processor Automata Module Implementation

Similarly, automata for related places of articles and connectors are established and we can obtain SQL query easily.

IX. RESULTS

Hence using the above mentioned algorithm a normal query can result in corresponding SQL query.
The interface developed is shown below:-
We have 2 textboxes

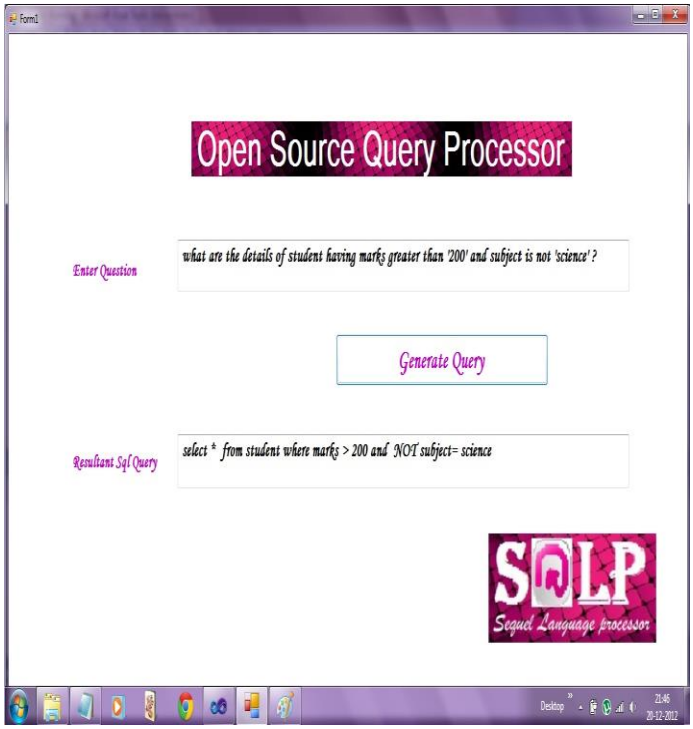
1- To Enter English Query
2- To display corresponding SQL query

Steps to be followed:-

- Write the English query in front of Enter Question Label
- Click on Generate Query Button
- The resultant SQL query will be displayed in the next textbox
- If any non matching keywords or erroneous query is questioned then a Message box for "Not able to read it" is displayed

Screenshot-1

Screenshot – 2



X. FUTURE ENHANCEMENTS

- More new automata implementations can be added to it to increase its efficiency.
- So far, this system considers selection and a few simple aggregations. The next step of the research is, to accommodate more complex queries
- Somewhere to control a big database is difficult

XI. CONCLUSION

- Natural Language Processing can bring powerful enhancements to virtually any computer program interface.
- A system that is capable of handling simple queries with standard join conditions is introduced here but because not all forms of SQL queries are supported, further development would be required.

REFERENCES

[1] Huangi,.Guiang Zangi, phillip C-y Sheu A Natural Language database interface based on probabilistic context free grammar". IEEE international workshop on Semantic Computing and systems 2008
[2] Akama, S. (Ed) *Logic.1onguage and computation.* Kulwer Academic: publishers. pp. 7-11, 1997.
[3] ELF Software CO. *Natural-Language Database interfaces* from *ELF Software Co.* cited november 1999. Available from internet
[4] Hendrix. G.G, Sacerdoti, E.D,sagalowicz. D. Slocum. J. "'Developing a natural Language interface to complex data in *ACM Transaction on database* system. 3(2). pp. 105- 147,1978.
[5] http://www.cnlp.org/publications/03nlp.lis.encyclopedia.pdf
[6] http://www.enggjournals.com/ijcse/doc/IJCSE10-02-02-20.pdf
[7] Joseph, S.W., Aleliunas, R. "A knowledge-based subsystem for a natural language interface to a database that predicts and explains query failures", in IEEE CH, pp. 80-87, 1991.
[8] Mitrovic, A. A knowledge-based teaching system for SQL,University of Canterbury, 1998. Moore, J.D. "Discourse generation for instructional applications: making computer tutors more like humans", in Proceedings AI-ED, pp.36-42,1995.
[9] Suh, K.S., Perkins, W.C., "The effects of a system echo in a restricted natural language database interface for novice users", in IEEE System sciences, 4, pp. 594-599, 1994.
[10] Whenhua, W., Dilts, D.M. "Integrating diverse CIM data bases: the role of natural language interface", in IEEE Transactions on systems, man, and cybernetics, 22(6), pp. 1331-1347, 1992.
[11] [9] Dan Klein, Christopher D. Manning: Corpus-Based Induction of Syntactic Structure: Models of Dependency and Constituency. ACL 2004: 478-485

AUTHORS

First Author – Jasmeen Kaur, Student(M.tech), B.S.Anangpuria Institute of Technology & Management , jasmeen.kaur13@gmail.com

Second Author – Bhawna Chauhan, Assistant Professor, B.S.Anangpuria Insitute of Technology & Management

Third Author – Jatinder Kaur Korepal, Lecturar, Guru Nanak Khalsa College