Structured Query Generation by Using PTQL As a Solution for Incremental Information Extraction

Gayatri Naik,
Research Scholar, Singhania University,
Rajasthan, India

Dr. Praveen Kumar,
Singhania University, Rajasthan, India

Abstract: *Improving the ability of computer systems to process text is a significant research Challenge. Many applications are based on partially structured databases, where structured data conforming to a schema is combined with free text. Information extraction systems traditionally implemented as a pipeline of special-purpose processing modules targeting the extraction of a particular kind of information. Most recent IE approaches are suitable for only static corpora. A major drawback of such an approach is that whenever a new extraction goal emerges or a module is improved, extraction has to be reapplied from scratch to the entire text corpus even though only a small part of the corpus might be affected. In this project, we describe a novel approach for information extraction in which extraction needs are expressed in the form of database queries, which are evaluated and optimized by database systems. Using database queries for information extraction enables generic extraction and minimizes reprocessing of data by performing incremental extraction to identify which part of the data is affected by the change of components or goals. Furthermore, our approach provides automated query generation components so that casual users do not have to learn the query language in order to perform extraction. To demonstrate the feasibility of our incremental extraction approach, we performed experiments to highlight two important aspects of an information extraction system: efficiency and quality of extraction results.*

Keywords: ***Text mining, query languages, information storage and retrieval.*****1.**

Introduction:

Information extraction (IE) is the task of automatically extracting structured information from unstructured and/or semi-structured machine-readable documents. In most of the cases this activity concerns processing human language texts by means of natural language processing (NLP).Information extraction systems are traditionally implemented as a pipeline of special-purpose processing modules targeting the extraction of a particular kind of information. A major drawback of such an approach is that whenever a new extraction goal emerges or a module is improved extraction has to be reapplied from scratch to the entire text corpus even though only a small part of the corpus might be affected. In this paper, we describe a novel approach for information extraction in which extraction needs are expressed in the form of database queries, which are evaluated and optimized by database systems. Using database queries for information extraction enables generic extraction and minimizes reprocessing of data by performing incremental extraction to identify which part of the data is affected by the change of components or goals.

2. Literature Survey

2.1 Existing System

Information extraction has been an active research area over the years. The main focus has been on improving the accuracy of the extraction systems, and IE has been seen as a one-time execution process. Such paradigm is inadequate for real-world applications when IE is seen as long as running processes. An example of a real-world application of IE is the extraction from evolving text, such as the frequent update of the content of web documents. [4] Existing Approaches are

1. Rule-Based IE Approaches.
2. Machine Learning IE Approaches.

2.2. Related Work

1. UIMA: an architectural approach to unstructured information processing in the corporate research environment.

IBM Research has over 200 people working on Unstructured Information Management (UIM) technologies with a strong focus on Natural Language Processing (NLP). [2] These researchers are engaged in activities ranging from natural language dialog, information retrieval, topic-tracking, named-entity detection, document classification and machine translation to bioinformatics and open-domain question answering. An analysis of these activities strongly suggested that improving the organization's ability to quickly discover each other's results and rapidly combines different technologies and approaches would accelerate scientific advance. In this paper we give a general introduction to UIMA focusing on the design points of its analysis engine architecture and we discuss how UIMA is helping to accelerate research and technology transfer. Approach provides mechanisms to generate extraction queries from both labeled and unlabeled data. Query generation is critical so that casual users can specify their information needs without learning the query language. Learning extraction patterns from training data. However, training data are not always readily available and annotations of training data are both labor-intensive and time consuming.

2. A Relational Approach to Incrementally Extracting and Querying Structure in Unstructured Data: Computer Sciences Department University of Wisconsin-Madison 2008.

It propose the use of a relational database as a workbench not only for storing and querying structured data, but also for incrementally evolving structure from unstructured data. [3] They conducted a case study of applying approach to evolve and query the structure in the contents of Wikipedia. Their experience in this study demonstrated that approach exploits existing technology effectively and allows one to quickly and incrementally discover and query the structure lurking in unstructured documents. Much scope for future work remains – virtually every aspect of the system can be "drilled down" upon to discover and evaluate alternative approaches.

3. System Approach:

Novel Database-Centric Framework for Information Extraction. We first give an overview of our approach, and discuss each of the major components of our system

Our approach composed basic two phases:

- Initial phase: for processing of text and
- Extraction phase: for using database queries to perform extraction.

As shown in fig 1 the Text Processor in the initial phase is responsible for corpus processing and storage of the processed information in the Parse Tree Database (PTDB). The extraction patterns over parse

trees can be expressed in our proposed parse tree query language. In the extraction phase, the PTQL query evaluator takes a PTQL query and transforms it into keyword-based queries and SQL queries, which are evaluated by the underlying RDBMS and information retrieval (IR) engine. To speed up query evaluation, the index builder creates an inverted index for sentences according to words and the corresponding entity types. Fig 1 illustrates the system architecture of our approach. [1]

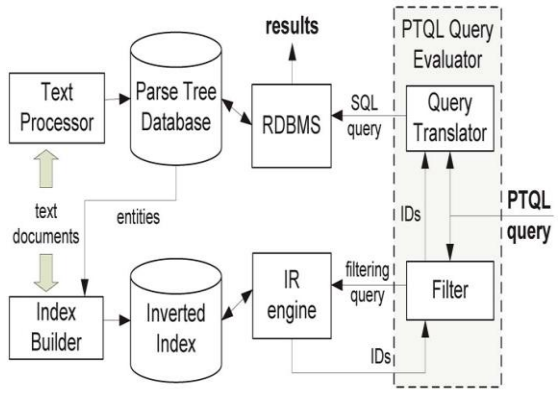

Fig 1: System architecture of the PTQL framework.

4. Link Grammar (LG)

It is a theory of syntax by Davy Temperley and Daniel Sleator which builds relations between pairs of words, rather than constructing constituents in a tree-like hierarchy. There are two basic parameters: directionality and distance. Link grammar is similar to dependency grammar, but dependency grammar includes a head-dependent relationship, as well as lacking directionality in the relations between words. The Link Grammar parser is a dependency parser based on the Link Grammar theory. Link Grammar consists of a set of words and linking requirements between words. [10]

5. Parse Tree Database

We propose a new paradigm for information extraction that utilizes database management systems as an essential component of our extraction framework. Database management systems become a logical framework of choice that can serve such dynamic extraction needs over file-based storage systems. Text processing components such as named entity recognizers and syntactic parsers are deployed for the entire collection. The intermediate output of the processing modules is stored in a relational database known as the parse tree database. Extraction then becomes a matter of issuing database queries in the form of parse tree query language (PTQL). In the event of a change of extraction goals or a module update, the responsible module is deployed for the entire text corpus and the processed data is populated into the parse tree database with the previously processed data. Incremental extraction is performed so that database queries are issued to identify sentences with newly recognized mentions. Once the affected sentences are identified, extraction can then be performed only on such sentences rather than the entire corpus. By storing the processed data, our approach avoids the need to reprocess the entire collection of text unlike the file-based pipeline approaches. Avoiding reprocessing of data is particularly important for extraction. We highlight the technical contributions of the architecture proposed Novel Database-Centric Framework for Information Extraction. Unlike traditional approaches for IE, our approach is to store intermediate text processing output in a specialized database called the parse tree database. Extraction is formulated as queries so that it is no longer necessary to write and run special-purpose programs for each specific extraction goal. Our approach minimizes the need of reprocessing the entire collection of text in the presence of new extraction goals and deployment of improved processing components. Fig 5.1

shows PTD Model.

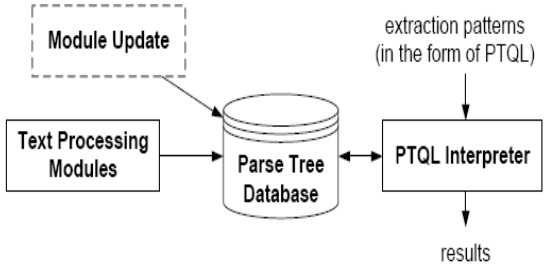

Fig2: Parse tree database use for extraction & update of modules

6. Inverted Index

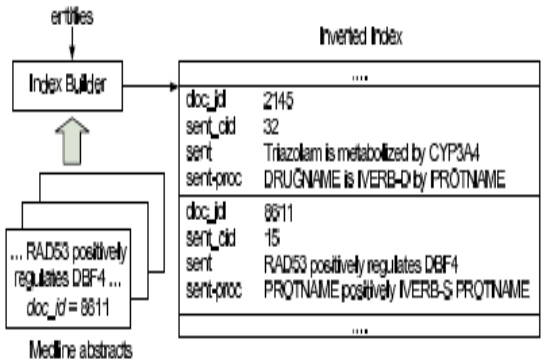

Fig3: An extended inverted index to handle queries that involve concepts rather than just instances.

Fig. 3, the index builder relies on the text preprocessor to recognize entities and replace the entities with identifiers in the sentences. Each sentence in the documents are indexed on its own so that each keyword-based filtering query retrieves a sentence rather than the entire document. The index builder includes the original sentences in the inverted index, as well as sentences with entities replaced with identifiers. For instance, the sentence "RAD53 positively regulates DBF4" is indexed as PROTNAME positively IVERB-S PROTNAME under the field name sent-proc in the inverted index.

A fundamental design criterion for the query language is the ability of expressing linguistic patterns based on constituent trees. One of the novel features of our proposed query language PTQL is the ability to express links and link types between pairs of nodes, so that PTQL can be used to express linguistic patterns based on constituent trees and links, as well as link types. We propose a high level extraction query language called PTQL. PTQL is an extension of the linguistic query language LPath that allows queries to be performed not only on the constituent trees but also the syntactic links between words on linkages.

7. PTQL Query

It is made up of four components:

1. Tree pattern: describes the hierarchical structure and the horizontal order between the nodes of the parse tree.

2. Link condition: describes the linking requirements between nodes.

3. Proximity condition: is to find words that are within a specified number of words.

4. Return expression: defines what to return. We start with the basic element of PTQL queries called node expressions.

7.1 Query Evaluation

Our approach for the evaluation of PTQL queries involves the use of IR engine as well as RDBMS. The role of the IR engine in query is to select sentences based on the lexical features defined in PTQL queries, and only the subset of sentences retrieved by the IR engine are considered for the evaluation of the conditions specified in the PTQL queries by RDBMS. We summarize the process of the evaluation of PTQL queries as follows.

1. Translate the PTQL query into a filtering query.

2. Use the filtering query to retrieve relevant documents D and corresponding sentences S from the inverted index.

3. Translate the PTQL query into an SQL query and instantiate query with document

id 2 D and sentence id s 2 S.

4. Query PTDB using the SQL query generated in Step 3.

5. Return the results of the SQL query as the results of the PTQL query.

7.2 Query Generation

An important aspect of an IE system is its ability to extract high-quality results. In this section, we demonstrate the feasibility of our approach by first describing two approaches in generating PTQL queries automatically:

1. Training set driven query generation

2. Pseudo-relevance feedback driven query generation.

The first query generation approach takes advantage of manually annotated data to generate PTQL queries for the extraction. As training data are not always available, we introduce the latter approach that identifies frequent linguistic structures to generate PTQL queries without the use of training data.

8. Flow chart of System

It shows the flow of actual system activities perform during operation.

9. Results

We showed that we can retrieve the document file names based on the user query and the query searches based on the phrase based search which retrieves semantic data. We can get progressive corpus without extraction from scratch by storing the intermediate results in the table and retrieve the document by using PTQL query.

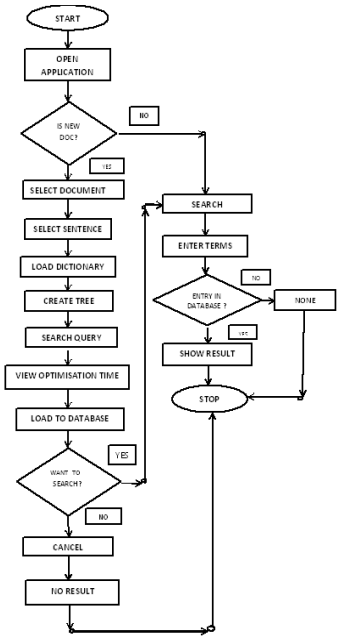

Fig.4 Flow of System operation.

We first illustrate the performance of our approach in terms of query evaluation and the time savings achieved through incremental extraction. We evaluate the extraction performance for our two approaches

1. Automatic training set driven query generation & data processing i.e. Without PTQL.

2. Automatic Query generation by using user keyword based query generation & data processing i.e. With PTQL.

Input Data Sample: Biomedical Research Data.

Rad53 is a protein kinase required for cell-cycle arrest in response to DNA damage. Rad53 controls the S-phase checkpoint as well as G1 and G2 DNA damage checkpoints. It prevents entry into anaphase and mitotic exit. After DNA damage via regulation of the Polo kinase Cdc5. Rad53 also seems to be involved in the phosphorylation of Rph1.A Zink finger protein that acts as a damage-responsive repressor of the photolyase gene Phr1 in saccharomyces cerevisiae. The 2C type phosphatase, Ptc2, which is required for DNA check point inactivation after double-strand breaks, appears to dephosphorylate Rad53.

1.Select Document : Data Processing

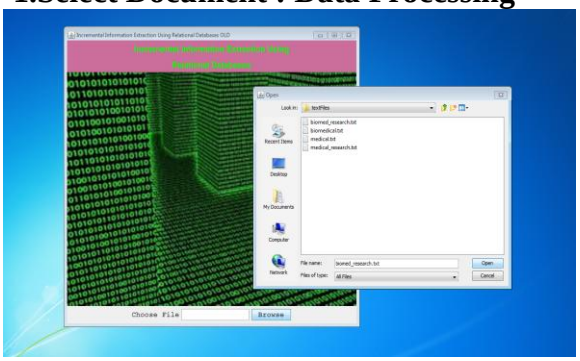

Fig 5: select document snapshot.

2. Parse Tree : Data Processing without PTQL

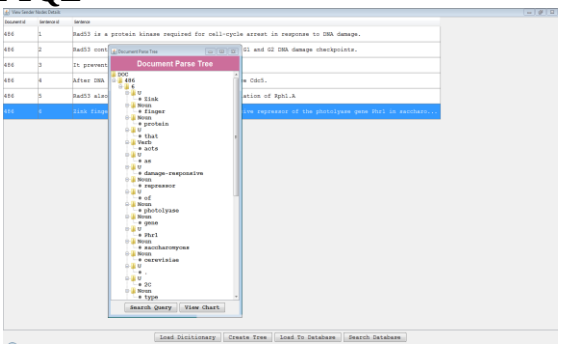

Fig6 : Parse Tree snapshot.

3 Search Query : Data Processing without PTQL

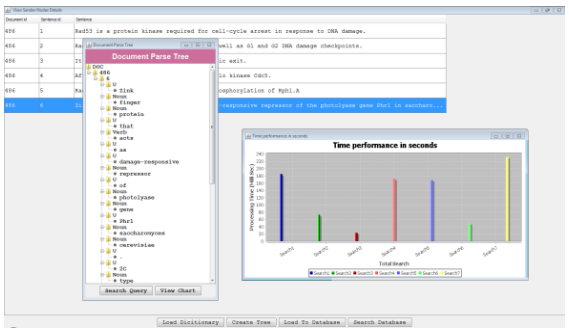

Fig7 : Search Query snapshot

4 Time Performance : Data Processing without PTQL

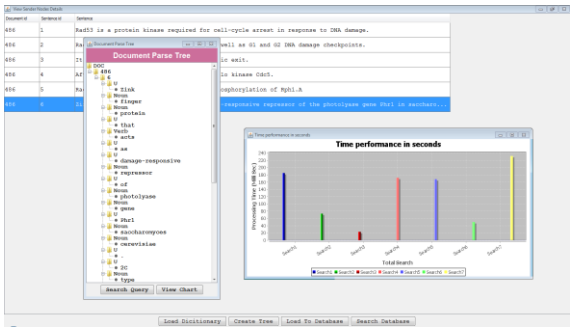

Fig 8: Time Performance snapshot.

5. Parse Tree: Data Processing with PTQL

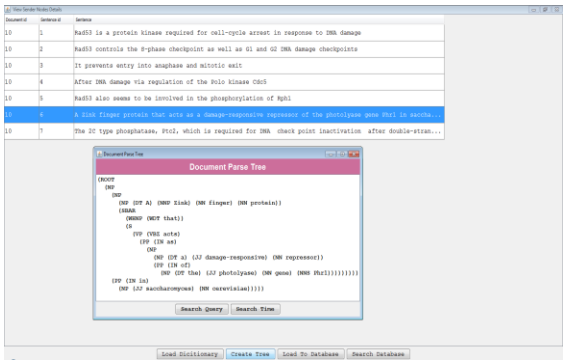

Fig 9: Parse tree snapshot

6. Search Query: Data Processing with PTQL

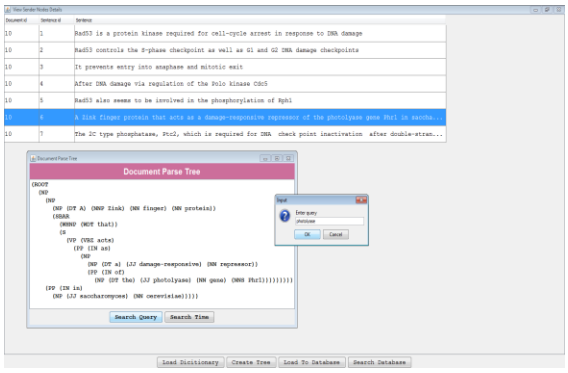

Fig10: Search Query snapshot

7. Time Performance: Data Processing with PTQL

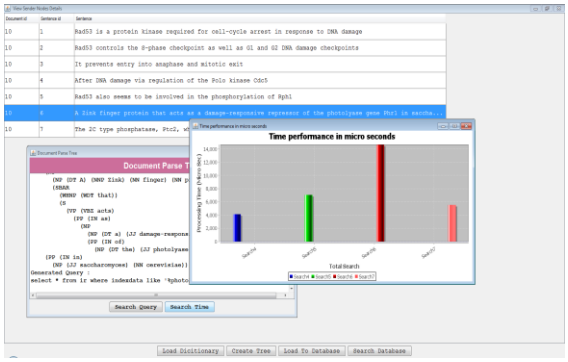

Fig11: Time Performance snapshot

Conclusion

Information extraction has been an active research area over the years. The main focus has been on improving the accuracy of the extraction systems, and IE has been seen as an one-time execution process. Such paradigm is inadequate for real-world applications when IE is seen as long running processes. We describe how our proposed extraction framework differs from traditional IE systems, rule-based IE systems and IE systems based on machine learning. While new documents can be added to our text collection, the content of the existing documents are assumed not to be changed, which is the case for Medline abstracts. Our focus is on managing the processed data so that in the event of the deployment of an improved component or a new extraction goal, the affected subset of the text corpus can be easily identified. The filtering process utilizes the efficiency of IR engines so that a complete scan of the parse tree database is not needed without sacrificing any sentences that should have been used for extraction. Furthermore, our approach provides automated query generation components so that casual users do not have to learn the query language in order to perform extraction. To demonstrate the feasibility of our incremental extraction approach, we performed experiments to highlight two important aspects of an information extraction system: efficiency and quality of extraction results. Hence we conclude that

1. Using database queries instead of writing individual special purpose programs, information extraction becomes generic for diverse applications.
2. Two phase method avoids performing initial phase again.
3. Automate Query generation minimize users efforts & improve Time performance.

Future Scope:

We will implement this application for web documents that will access through World Wide Web. We will create Autonomous protocols for the same application to run properly on Distributed network.

References

[1] Luis Tari, Student Member, IEEE Computer Society, Phan Huy Tu, Hakenberg, Yi Chen, ember, " Incremental Information Extraction Using Relational Databases" IEEE Computer Society, Tran Cao Son,

Graciela Gonzalez, and Chitta Baral Vol.24.No.1 Jan 2012.

[2] D. Ferrucci and A. Lally, "UIMA: An Architectural Approach to Unstructured Information Processing in the Corporate Research Environment," Natural Language Eng., vol. 10, nos. 3/4, pp. 327- 348, 2004.

[3] Eric Chu Akanksha Baid Ting Chen AnHai Doan Jeffrey Naughton "A Relational Approach to Incrementally Extracting and Querying Structure in Unstructured Data". Computer Sciences Department University of Wisconsin-Madison 2008

[4] S. Sarawagi, "Information Extraction," Foundations and Trends in Databases, vol. 1, no. 3, pp. 261-377, 2008.

[5] D. Grinberg, J. Lafferty, and D. Sleator, "A Robust Parsing Algorithm for Link Grammars," Technical Report CMU-CS-TR- 95-125, Carnegie Mellon Univ. 1995.

[6] F. Chen, A. Doan, J. Yang, and R. Ramakrishna, "Efficient Information Extraction over Evolving Text Data," Proc IEEE 24th Int'l Conf. Data Eng. (ICDE '08), pp. 943-952, 2008.

[7] F. Chen, B. Gao, A. Doan, J. Yang, and R. Ramakrishna, "Optimizing Complex Extraction Programs over Evolving Text Data," Proc 35th ACM SIGMOD Int'l Conf. Management of Data (SIGMOD '09), pp. 321-334, 2009.

[8] S. Bird et al., "Designing and Evaluating an XPath Dialect for Linguistic Queries," Proc 22nd Int'l Conf. Data Eng. (ICDE '06), 2006.

[9] H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan, "GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications," Proc. 40th Ann. Meeting of the ACL, 2002.

[10] D. Sleator and D. Temperley, "Parsing English with a Link Grammar," Proc Third Int'l Workshop Parsing Technologies, 1993.