Efficient Natural Language Query interface to databases

[1]B.SUJATHA, [2]Dr.S.Vishwanatha Raju, [3]S.Nagaprasad
[1, 2, 3] Computer Science Engineering Department of CSE, Hyderabad

Abstract: NDLIB overcomes the need to write SQL queries as naïve users may not be aware of the structure of the database. NDLIB allows access to database through natural language queries making them logically independent from data. This led to the development of new type of processing called Natural language Interface to Database. Research indicates existing NDLIB systems lack flexibility of forming queries in user's own format. They also provide an incomplete specification to the requested data. Other areas of challenge are ambiguities in various forms: structural, word sense, referential and literal. We are proposing EFLEX system (A Flexible and Efficient Natural Language Query interface to databases) to tackle above challenges. The proposed system consists of three major components. They are a) an analyzer b) a mapper and c) a translator. The function of analyzer is to interpret the queries entered by the next user. Mapper is used to correspond natural language query to SQL query. Finally the Translator component performs actual translation of a query. The system is designed to be efficient in the way a user's query is translated into SQL query. The efficiency is achieved by using KMP Algorithm to translate a Natural language query into SQL query. The evaluation was performed on the first version of the software. The empirical validation of the prototype shows improved performance.

Keywords: NDLIB, EFLEX, SQL, KMP, Analyzer, Mapper, Translator

1. Introduction:

Natural language processing is the technology for dealing with our most ubiquitous product: human language, as it appears in emails, web pages, tweets, product descriptions, newspaper stories, social media, and scientific articles, in thousands of languages and varieties. In the past decade, successful natural language processing applications have become part of our everyday experience, from spelling and grammar correction in word processors to machine translation on the web, from email spam detection to automatic question answering, from detecting people's opinions about products or services to extracting appointments from your email. There are fundamental algorithms and mathematical models for human language processing and how you can use them to solve practical problems in dealing with language data wherever you encounter it are to be explored. Research in natural language processing has been going on for several decades dating back to the late 1940s. Machine translation (MT) was the first computer-based application related to natural language. While Weaver and Booth started one of the

earliest MT projects in 1946 on computer translation based on expertise in breaking enemy codes during World War II, it was generally agreed that it was Weaver's memorandum of 1949 that brought the idea of MT to general notice and inspired many projects. He suggested using ideas from cryptography and information theory for language translation. Research began at various research institutions in the United States within a few years. Early work in MT took the simplistic view that the only differences between languages resided in their vocabularies and the permitted word orders. Systems developed from this perspective simply used dictionary-lookup for appropriate words for translation and reordered the words after translation to fit the word-order rules of the target language, without taking into account the lexical ambiguity inherent in natural language. This produced poor results. The apparent failure made researchers realize that the task was a lot harder than anticipated, and they needed a more adequate theory of language. However, it was not until 1957 when Chomsky published Syntactic Structures introducing the idea of generative grammar, did the field gain better insight into whether or how mainstream linguistics could help MT. During this period, other NLP application areas began to emerge, such as speech recognition. The language processing community and the speech community then was split into two camps with the language processing community dominated by the theoretical perspective of generative grammar and hostile to statistical methods, and the speech community dominated by statistical information theory and hostile to theoretical linguistics. Due to the developments of the syntactic theory of language and parsing algorithms, there was over-enthusiasm in the 1950s that people believed that fully automatic high quality translation systems would be able to produce results indistinguishable from those of human translators, and such systems should be in operation within a few years. It was not only unrealistic given the then-available linguistic knowledge and computer systems, but also impossible in principle. The inadequacies of then-existing systems, and perhaps accompanied by the over enthusiasm, led to the ALPAC (Automatic Language Processing Advisory Committee of the National Academy of Science - National Research Council) report of 1966. The report concluded that MT was not immediately achievable and recommended it not be

funded. This had the effect of halting MT and most work in other applications of NLP at least within the United States.

Although there was a substantial decrease in NLP work during the years after the ALPAC report, there were some significant developments, both in theoretical issues and in construction of prototype systems. Theoretical work in the late 1960‟s and early 1970‟s focused on the issue of how to represent meaning and developing computationally tractable solutions that the then-existing theories of grammar were not able to produce. In 1965, Chomsky introduced the transformational model of linguistic competence. However, the transformational generative grammars were too syntactically oriented to allow for semantic concerns. They also did not lend themselves easily to computational implementation. As a reaction to Chomsky's theories and the work of other transformational generativists, case grammar of Fillmore, semantic networks of Quillian, and conceptual dependency theory of Schank were developed to explain syntactic anomalies, and provide semantic representations. Augmented transition networks of Woods extended the power of phrase-structure grammar by incorporating mechanisms from programming languages such as LISP. Other representation formalisms included Wilks‟ preference semantics and Kays functional grammar.

A major problems faced by most of the unsophisticated users with computer systems is that they generally make use of special purpose or dedicated language which are familiar to those trained with the those computer machines. It would be highly desirable for machines to converse in English or other languages to their general users. Several systems are developed that guaranty at least elementary levels of Natural Language Interactions.

Natural Language process (NLP) is concerned with the event of computational models of aspects of human language process. There square measure main reasons of such development:

1. To develop automatic tools for language process.
2. To realize a far better understanding of human communication.

Building machine models with human language process talents needs knowledge of however humans acquire store and method language. It additionally needs knowledge of the planet and of language.

2. NLP Applications: The applications utilizing natural language processing embody the following:

Machine Translation: Machine Translation is a process of Automatic translation of text from one human language to a different. In order to hold out this translation, it's necessary to own an understanding of words and phrases, grammars of multiple languages concerned, linguistics of the language, and world information.

Speech Recognition: Recognition is a method of mapping acoustic speech signals to a collection of words. The difficulties arise within the pronunciations of various words.

Speech Synthesis: Speech Synthesis refers to automatic production of speech (observations of linguistic communication sentences). Such systems will browse out your mails on phone or perhaps browse out a story book for you. So as to get expressions, text must be processed.

Language Interfaces: It permits querying a info victimization linguistic communication sentences. it's a system that allows the user to access data keep during a info by writing requests expressed in some linguistic expressions.

Information Retrieval (IR): Retrieval is involved with distinguishing documents relevant to user's question. Indexing word sense illumination, question modification and cognitive content have additionally been employed in IR system to reinforce performance.

Information Extraction: This captures and outputs factual data contained at intervals a document. Similar to IR system, it responds to user's requirement. The knowledge need isn't expressed as a keyword question instead it's such as pre-defined database schemas or templates.

3. Recent Developments in EFLEX system

This section provides a fast outline of three specific EFLEX systems developed recently in various universities.

1. Nalix: NALIX (Natural Language Interface for Associate in Nursing XML Database) is Associate in Nursing EFLEX SYSTEM developed at the University of Michigan, metropolis by Yunyao Li, Huahai principle, and H. V. Jagadish 2006. The data used for this technique is protractible language (XML) info with Schema- Free XQuery as a result of the data source language. Schema-Free Query is also a source language designed primarily for retrieving information in XML. The thought is to use keyword search for databases. As per Li et al: "Database query languages can be intimidating to the non-expert, leading to the immense recent popularity for keyword based search in spite of its significant limitations. The Holy Grail has been the development of a natural language query interface. We present NaLIX, a generic interactive natural language query interface to an XML database. Our system can accept an arbitrary English language sentence as query input, which can include aggregation, nesting, and value joins, among other things. This query is translated, potentially after reformulation, into an XQuery expression that can be evaluated against an XML database. The translation is done through mapping grammatical proximity of natural language parsed tokens to proximity of corresponding elements in the result XML. In this demonstration, we show that NaLIX, while far from being able to pass the Turing test, is perfectly usable in practice, and able to handle even quite complex queries in a variety of application domains. In addition, we also demonstrate how carefully designed features in NaLIX facilitate the interactive query process and improve the usability of

the interface". However, pure keyword search positively cannot be applied. Therefore, some richer question mechanisms units of measurement are to be identified. Given a bunch of keywords, each keyword has several candidate XML components to relate. These entire candidates unit of measurement else to MQF (Meaningful question Focus), which might automatically notice all the relations between these components. The foremost advantage of Schema-Free Query is that it is not necessary to map an issue into the precise data schema, since it's going to automatically notice all the relations given positive keywords. NALIX is classed as syntax primarily based system, since the transformation processes unit of measurement drained three steps: generating a analyse tree, collateral the analyse tree, and translating the Associate in Nursingalyse tree to a Query expression. NALIX is totally different from the ultimate syntax primarily based approaches; at intervals the means that the system was built: NALIX implements a reversed-engineering technique by building the system from an issue language toward the sentences.

2. Precise: Precise is a system developed at the University of Washington by Ana-Maria Popescu, Alex Armanasu, Oren Etzioni, David Ko, and Alexander Yates (2004). The target information is within the sort of an electronic database treating SQL because the search language. It introduces the concept of linguistically tractable sentences that are unit sentences which will be translated to a novel linguistics interpretation by analyzing some lexicons and semantic constraints. PRECISE was evaluated on 2 information domains. The primary one is that the ATIS domain, that consists of spoken questions about air, their written forms, and their correct translations in SQL search language. In ATIS domain, 95.8% of the queries were semantically tractable. Treating these queries offer PRECISE ninety four exactness. The second domain is that the GEOQUERY domain. This domain contains data regarding U.S. Geography. 77.5% of the queries in GEOQUERY area unit semantically tractable. Treating these queries offer PRECISE 100 percent accuracy. The strength of PRECISE relies on the power to match keywords in a very sentence to the corresponding information structures. This method is finished in 2 stages, 1st by narrowing the probabilities treating Maxflow formula and second by analyzing the grammar structure of a sentence. Thus PRECISE is in a position to perform imposingly in semantically tractable queries. As different EFLEX SYSTEM systems, PRECISE has its own weaknesses. Whereas it's ready to come through high accuracy in semantically tractable queries, the system compensates for the gain in accuracy at the price of recall. Another drawback is as PRECISE adopts a heuristic based mostly approach; the system suffers from the matter of handling nested structures.

3.Wasp:Word Alignment-based linguistics Parsing (WASP) is also a system developed at the University of American state, city by Yuk Wah Wong. Whereas the system is meant to handle the broader goal of constructing "a complete, formal, symbolic, meaty illustration of a language sentence", it can also be applied to the EFLEX SYSTEM domain. A predicate logic (Prologue) was used as a result of the formal query language. WASP learns to create a linguistics programme given a corpus a bunch of language sentences annotated with their correct formal question languages. It wants no prior knowledge of the syntax; as a result of the whole learning technique are finished exploitation math AI techniques. WASP was evaluated on the GEOQUERY domain, identical domain as PRECISE. GEO- question corpus consists of 880 queries at intervals the employment set and 250 queries at intervals the take a glance at set, that are united on into one larger data set. Each data set was divided to 10 equal-sized subsets, and traditional 10-fold cross validation was accustomed estimate the system performance. WASP achieved eighty six.14% exactitude and seventy 5.00% recall at intervals the GEOQUERY do- main. The system was in addition evaluated on a ramification of various natural languages: English, Spanish, Japanese and Turkish. There are no important variations determined between English and Spanish, but the Japanese corpus has the lowest exactitude and so the Turkish corpus has the lowest recall. The strength of WASP comes from the facility to create a linguistics programme from annotated corpora. This approach is beneficial as a result of it uses math AI with lowest oversight. Therefore, the system ought not to manually develop a synchronic linguistics in many domains. Moreover, whereas most of EFLEX SYSTEM systems use English as their language, WASP has been tested on several languages. In spite of the strength, WASP put together has a pair of weaknesses. The first is: the system relies alone on the analysis of a sentence and its potential question translation, therefore and additionally the data [*fr1] is thus left untouched. There is uncountable data which is able to be extracted from a data, just like the lexical notation, the structure, and so the relations at intervals. Not exploitation this data prevents WASP to realize higher performances. The second drawback is that the system wants Associate in nursing oversized amount of annotated corpora before it's going to be used; Associate in nursing building such corpora wants an oversized amount of labour.

4. Lessons learnt: Learning derived from NALIX related studies:

A thesaurus component will help in adding non-answered user queries

Keyword based query answering is not sufficient as same query can be derived from different keywords.

Toy set is a potential dataset for test bed

Database can be XML

Architecture components can be as depicted in Figure 1

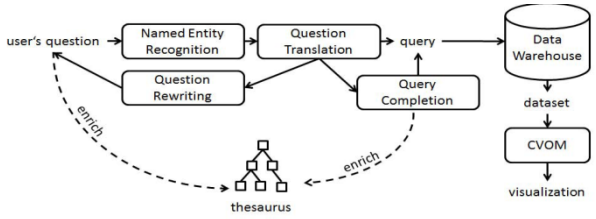

Figure 1: Analysis of Question

"Entity Recognizer (NER) aims at identifying known objects of the data model that comprise dimensions, measures and values of dimensions. The Question Translation component builds a technical query from a user's question. The Question Rewriting component rewrites the query if no answer can be retrieved. The Query Completion component is used when queries are not expressive enough (e.g. when no dimension is comprised in the query). The CVOM1 component provides visualization to a dataset. In this paper, the focus is on three components: the Question Translation component, the Question Rewriting component.

Evaluation results in are as depicted in Figure 2

Entity type	found entities	distinct found entities
PLACE_OTHER	3	3
TIME_PERIOD	3	2
ORGANIZATION	19	9
PLACE_REGION	117	17
COMPANY	7	6
COUNTRY	106	38
PERSON	13	12
DATE	9	9
CITY	4	4

Figure 2: Analysis of Question [Extracted from 34]

- Reasoning feature can be a future work:

Learning derived from PRECISE [32] related studies:

- Statistical and machine learning techniques can resolve parsing problems

-ATIS dataset can be used for test bed

- Automation of syntactic parsing and semantic grammar generation

- Framework for tractable and non-tractable queries discrimination

-NLI needs to be reliable and predictable

-"By giving a set of lexical constraints, semantic constraints, and syntactic constraints, we define what it means for an SQL statement to be a valid interpretation of a question. Our theoretical goal is to find classes of questions for which we can provably identify all the valid interpretations".

- For test bed considerations we can use natural language questions compiled by researchers at U.T. Austin (Tang & Mooney 2001) - which obtained 100% precision and approximately 80% recall in then evaluated their system on the well-known ATIS dataset

System Setup	PRECISE	PRECISEL
$Parser_{ORIG}$, *No Over-rides*	61.7%	60.1%
$Parser_{ORIG}$, *Over-rides*	89.5%	85.3%
$Parser_{ATIS}$, *No Over-rides*	92.2%	88.1%
$Parser_{ATIS}$, *Over-rides*	93.8%	89.1%
$Parser_{CORRECT}$, *Over-rides*	95.1%	91.3%

Table 1: **Impact of Parser Enhancements. The PRECISE column records the percentage of questions where the small set of SQL queries returned by PRECISE contains the correct query; PRECISEL refers to the questions correctly interpreted if PRECISE is forced to return exactly one SQL query. $Parser_{ORIG}$ is the original version of the parser, $Parser_{ATIS}$ is the version re-trained for the ATIS domain, and $Parser_{CORRECT}$ is the version whose output is corrected manually. *Over-rides* refer to the automatic use of semantic over-rides to correct parser errors.**

Figure 3: Impact of Parser Enhancements

Learning derived from WASP [33] related studies: As per [33]: "This paper considers a more ambitious task of semantic parsing, which is the construction of a complete, formal, symbolic, meaning representation (MR) of a sentence. Semantic parsing has found its way in practical applications such as natural-language (NL) interfaces to databases (Androutsopoulos et al., 1995) and advice taking (Kuhlmann et al., 2004). The algorithm learns a semantic parser given a set of NL sentences annotated with their correct MRs. It requires no prior knowledge of the NL syntax, although it assumes that an unambiguous,

Context-free grammar (CFG) of the target MRL is available. The main innovation of this algorithm is its integration with state-of-the-art statistical machine translation techniques. More specifically,

a statistical word alignment model (Brown et al., 1993) is used to acquire a bilingual lexicon consisting of NL substrings coupled with their translations in the target MRL. Complete MRs are then formed by combining these NL substrings and their translations under a parsing framework called the synchronous CFG (Aho and Ullman, 1972), which forms the basis of most existing statistical syntax-based translation models (Yamada and Knight, 2001; Chiang, 2005). Our algorithm is called WASP, short for Word Alignment-based Semantic Parsing. In initial evaluation on several real-world data sets, we show that WASP performs favorably in terms of both accuracy and coverage compared to existing learning methods requiring the same amount of supervision, and shows better robustness to variations in task complexity and word order." WASP was experimented in ROBOCUP and GEOQUERY

WASP performance results as depicted in Figure 2.4

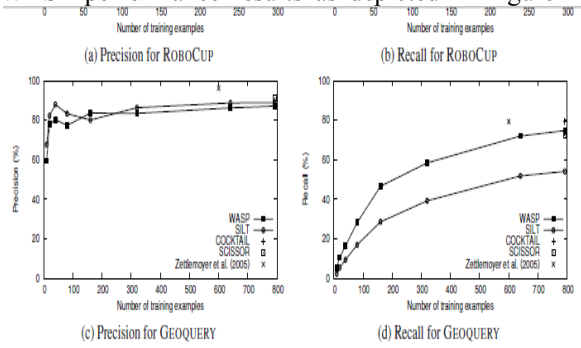

Figure 6: Precision and recall learning curves comparing various semantic parsers

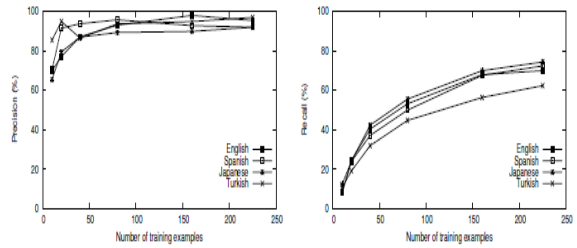

Figure 7: Precision and recall learning curves comparing various natural languages

Figure 4: Precision and Recall learning curves

Most of the previous implementations, which were made to implement these kinds of systems, are listed below. They follow the approaches like Dialogue, Corpus, Menu Based, Lexical Analysis and some sort of parsers. These techniques are broadly classified into four categories as follows

1. Pattern-Matching
2. Syntax-Based
3. Semantic Grammars
4. Intermediate Representation Language systems

Pattern Matching systems are easy to implement but have critical limitations. Each linguistic keyword parsed by the user input will have to verify and cross refer with the language rules to obtain the required information, and then only system can assist the meaning and converts the input to the target language. Whenever the length of the query increases, mapping mechanism will leads to time consuming process.

Syntax Based systems uses Grammars. These transforms parse trees into SQL queries. These implementations will follow some constraints to construct the user input.

Semantic Grammars adopts syntactic & Semantic Processing. Domains understand ability is a critical issue when these are applied to a new domain.

Intermediate Representation Language systems uses AVL tree Construction and it is very complicated to implement

5. PURPOSE AND NEED OF NLI –TO-DB

Most of the previous implementations, which were made to implement these kinds of systems, are listed below. They follow the approaches like Dialogue, Corpus, Menu Based, Lexical Analysis and some sort of parsers. These techniques are broadly classified into four categories as follows

1. Pattern-Matching
2. Syntax-Based
3. Semantic Grammars
4. Intermediate Representation Language systems

Pattern Matching systems are easy to implement but have critical limitations. Each linguistic keyword parsed by the user input will have to verify and cross refer with the language rules to obtain the required information, and then only system can assist the meaning and converts the input to the target language. Whenever the length of the query increases, mapping mechanism will leads to time consuming process.

Syntax Based systems uses Grammars. These transforms parse trees into SQL queries. These implementations will follow some constraints to construct the user input.

Semantic Grammars adopts syntactic & Semantic Processing. Domains understand ability is a critical issue when these are applied to a new domain.

Intermediate Representation Language systems uses AVL tree Construction and it is very complicated to implement

Natural Language Interface to information People via laptop all round the world, access, accumulate and manipulate large amount of knowledge each second of the day. These large amounts of knowledge ar situated in private personal computers or remote locations. Mostly, knowledge is kept in some quite repository system like information. Knowledge in information is sometimes managed by database management system and access to information is expedited through a special interaction language referred to as SQL or some version of it. To override the quality of SQL for non-professionals, many researchers have clothed to use language (NL). The concept of mistreatment NL has prompted the event of latest kind of process methodology referred to as language Interface to information. Here we tend to ar that specializes in language interface to information, an application of language process. A language Interface to a Database (EFLEX SYSTEM) may be a system that permits the user to access data keeps during a database by writing requests expressed in some language.

Advantages of EFLEX SYSTEM: Advantages of language Interface to information are as follows:

No linguistic communication: One advantage of EFLEX SYSTEMs is meant to be that the user isn't needed to be told an artificial communication language. Formal question languages like SQL are difficult to be told and master, a minimum of by non-computer-specialists.

No want for coaching: Graphical interfaces and form-based interfaces ar easier to use by occasional users; still, invoking forms, linking frames, choosing restrictions from menus, etc. constitute artificial communication languages that ought to be learned and mastered by the end-user. In distinction, a perfect EFLEX SYSTEM would enable queries to be formulated within the user's linguistic communication.

Better for a few queries: It has been argued that there are some quite queries (e.g. queries involving negation, or quantification) which will be simply expressed in language, but that seem troublesome (or a minimum of tedious) to specific mistreatment graphical or form-based interfaces. As an example, "Which department has no programmers?" (Negation), or "Which company provides each department?" (Universal quantification), can be simply expressed in language; however they'd be troublesome to specific in most graphical or form-based interfaces. Queries just like the higher than will, of course, be expressed in information question languages like SQL, however complicated information question language expressions ought to be written.

Easy to Use for Multiple information Tables: Queries that involve multiple information tables like "list the address of the farmers who got bonus bigger than ten thousand rupees for the crop of wheat", are troublesome to form in graphical programme as compared to language interface.

Disadvantages of EFLEX SYSTEM: Disadvantages of language interface to information system ar as follows:

Linguistic Coverage Not Obvious: A frequent criticism against EFLEX SYSTEMs is that the system's linguistic capabilities are not obvious to the user. Current EFLEX SYSTEMs will solely address restricted subsets of language. Users notice it troublesome to know (and remember) what kinds of queries the EFLEX SYSTEM will or cannot address. Formal question languages, form-based interfaces, and graphical interfaces generally do not suffer from these issues. Within the case of formal question languages, the syntax of the source language is sometimes well-documented, and any syntactically correct question is sure to tend a solution. Within the case of form-based and graphical interfaces, the user will typically perceive what kinds of queries will be input, by browsing the choices offered on the screen; and any question which will be input is sure to tend a solution. For example, information doesn't contain data concerning profit of farmers and user inputs the question "which farmer gets most profit". This question is out of the linguistic coverage of the system.

Linguistic vs. abstract Failures: When the EFLEX SYSTEM cannot perceive a matter, it's usually not clear to the user whether the rejected question is outside the system's linguistic coverage, or whether it's outside the system's abstract coverage. Thus, users usually try and rephrase queries touching on ideas the system doesn't grasp (e.g. rephrasing {questions concerning questions on questions about} salaries towards a system that is aware of nothing about salaries), as a result of they suppose that the matter is caused by the system's restricted linguistic coverage. In alternative cases, users don't try and reword queries the system might conceptually handle; as a result of they are doing not notice that the actual phrasing of the question is outside the linguistic coverage, which an alternate phrasing of constant question may be answered. Some EFLEX SYSTEMs arrange to solve this drawback by providing diagnostic messages, showing the rationale a matter cannot be handled. For example, user asks a question "list the names of farmers WHO ar thirty five years old" and the information has data concerning the age of the farmer however „age" word isn't there in question. Thus this question isn't out of linguistic coverage however conceptually it's not right, as a result of system doesn't perceive what thirty five is representing i.e. whether it is for age or for address.

Users assume intelligence: EFLEX SYSTEM users are usually misled by the system's ability to method language, and they assume that the system is intelligent, that it's wisdom, or that it can deduce facts, whereas actually most EFLEX SYSTEMs haven't any reasoning skills. This problem doesn't arise in formal question languages, form-based interfaces, and graphical interfaces, wherever the capabilities of the system ar a lot of obvious to the user. For example, once user asks a question "list the names of farmers WHO ar thirty five years old", he/she isn't specifying the word age; presumptuous that system can understand it mechanically. However system isn't thus intelligent.

4. Non-appropriate Medium: It has been argued that language isn't AN applicable medium for communicating with a system. Language is claimed to be too verbose or too ambiguous for human-computer interaction. EFLEX SYSTEM users ought to type long queries, whereas in form-based interfaces solely fields ought to be crammed in, and in graphical interfaces most of the work is often done by mouse-clicking. In language interface user must kind full sentence with all the connecters (articles, prepositions, etc) however in graphical or kind based mostly interfaces it's not required.

Possible Implementations: Pattern matching approach to natural language analysis is to interpret input utterances as a whole, rather than building up their interpretation by combining the structure and meaning of words or other lower-level constituents. With this approach, the interpretations are obtained by matching patterns of words against the input utterance. Associated with each pattern is an interpretation, so that the derived interpretation is the one attached to the pattern that matched. In the simplest case, this arrangement is simply a list of correspondences between equivalence classes of utterances (the ones that match a given pattern) and interpretations (the ones associated with each pattern). In more sophisticated variations of the approach, patterns may involve higher-level constituents or semantic elements, so that some aspects of the interpretation may become constructive, but the basic flavor of the approach still remains to go as directly as possible from the input utterance to the Pattern Matching systems are easy to implement but have critical limitations. Each linguistic keyword parsed by the user input will have to verify and cross refer with the language rules to obtain the required information, and then only system can assist the meaning and converts the input to the target language. Whenever the

length of the query increases, mapping mechanism will leads to time consuming process.

To make more complete analyses of the input using the same techniques would require far too many patterns - in t he extreme, one pattern for every possible utterance. Moreover, many of these patterns would contain common sub-elements because they mentioned the same objects or had t he same concepts arranged with slightly different syntax. In order to resolve these problems within the pattern matching approach, hierarchical pattern matching methods have been developed in which some patterns match only part of the input and replace that part by some canonical result. Other higher-level patterns can then match on these canonical elements in a similar way, until a top-level pattern is able to match the canonical input as a whole according to the standard pattern matching paradigm. In this way, similar parts of different utterances can be matched by the same patterns and the total number of patterns is much reduced and made more manageable.

Syntax-Based:

Syntax deals with the ways that words can fit together to form higher-level units such as phrases, clauses, and sentences. Syntactically-driven parsing is, therefore, naturally constructive, i.e. the interpretations of larger groups of words are built up out of the interpretations of their syntactic constituent words or phrases. In this sense, it is just the opposite of pattern matching in which the emphasis is on interpretation of the input as a whole. The most natural way for syntactically-driven parsing to operate is to construct a complete syntactic analysis of the input natural language string first and only then to construct the internal representation or interpretation. As we will see, this leads to considerable inefficiency, and more recent syntactically-driven approaches have tried to intermix parsing and interpretation.

Syntax Based systems uses Grammars. These transforms parse trees into SQL queries. These implementations will follow some constraints to construct the user input.

The problem here is that the context-free nature of the grammar does not allow agreements such as the one required in English between subject and object. To enforce such an agreement, we would have to have two completely parallel grammars, one for singular sentences and the other for plural. Moreover, a grammar which also allowed passive sentences and would have to have another completely different set of rules, even though the passive and the active forms of the same sentence have a clear syntactic relation, not to mention semantic equivalence.

These duplications are multiplicative rather than additive, leading to exponential growth in the number of the grammar rules. Thus in terms of the number of rules involved, and in terms of being unable to capture related phenomena by related rules, context-free grammars turn out to be quite unsuitable for natural language analysis.

Semantic Grammars

Semantic Grammars adopts syntactic & Semantic Processing. Domains understand ability is a critical issue when these are applied to a new domain.

Now we turn to the uses of case frames in parsing natural language, and in particular to certain parsing techniques available to parsers whose target representation is based on case frames. In essence, parsers built around case grammars help to combine bottom up recognition of structuring constituents with more focused top-down instantiation of less structured, more complex constituents. We mentioned that case frames consist of a header and a collection of semantically defined cases.

There is a bit more to it than that. Each case consists of filler and a positional or a lexical marker. We saw examples of case fillers in the previous sections. A positional case marker says that the filler of the case occurs in a predefined location in the surface string. A lexical case marker says that the case filler is preceded by one of a small set of marker words (usually prepositions) in the surface string.

A typical case-frame parsing algorithm that operates on this case frame data structure could be summarized as follows:

For each case frame in the grammar, attempt an unanchored match of the header pattern against the input string. If none succeed, the input is un-parsable by the grammar.

If one or more matches are found, perform the following steps for each case header, and the one(s) that account for the entire input are the possible parses of the input string. Retrieve the case frame indexed by the recognized case header. Attempt to recognize each required case, as follows:

If the case is marked lexically, do an unanchored match for the case marker (a very simple one-or-two word pattern), and if that succeeds, perform the more complex recognition of the case filler by anchored match to the right of the case marker, or by a more complex parsing strategy (such as recognizing an embedded case frame starting at that location in the input).

If the case is marked positional, do an anchored match of the case filler (or again a more complex recognition strategy) starting at the designated point in the input string.

If the case maker can be marked either way, search first for t he lexical marker, and failing that attempt to recognize it positional.

If one or more required cases are not recognized, return an error condition. This signifies a possible ellipsis, incorrect selection of the case frame, ill-formed user input, or insufficient grammatical coverage. The following sections address issues of robust recovery from ill-formed user input.

Attempt to recognize all the optional cases by applying the same method used to parse the required ones. If some are not recognized, however, do not generate error conditions.

If after all the required and optional cases have been processed, and there is remaining input, generate a potential error condition denoting spurious input, insufficient coverage, or garbled or ill-formed input that may be recognized by more flexible parsing strategies.

The advantages of case frame instantiation over other parsing techniques can be summarized as follows: Case frames combine bottom-up recognition of simple structuring constituents, such as case headers and case markers, with top-down recognition of semantically more complex, but syntactically less significant case fillers. The differential treatment of different constituents provides more efficient parsing in general, allows for ellipsis resolution, and makes possible some forms of error recovery, as discussed below.

Case frames combine syntax and semantics. Positional and case-marker information is used in concert with semantic recognition of case fillers, thus reducing (though certainly not eliminating) structural and lexical ambiguity. Case frames are a fairly convenient representation for back-end systems to use. In contrast, parse trees must first be interpreted semantically and subsequently transformed into a representation more convenient for other modules in the system.

Some types of extra grammatical sentences for more complete accounts are listed below with examples, that might be encountered by an interface to a data base of college courses in which the courses are identified by a the name of a department followed by a number.

Spelling errors: Transfer Jim Smith from Economics 237 too Mathematics 156

Unless a natural language interface can deal with problems in these classes easily, it will appear very uncooperative and stupid to its users, who will tend either not to use it if they have that choice, or to use it with a high-level of frustration. We will examine techniques available to deal with some of the above deviations from grammaticality in more detail. Spelling errors are the most common and normally the most easily corrected of all grammatical deviations.

The usual basic approach when a word is found to be outside the vocabulary of a natural language interface is to compare the word against a set of known words and substitute the word (or words) from that list found to be closest to the unknown word according to some metric and subject to some threshold of closeness. We do not have time here to go into the methods of comparison, but clearly, the process will be made more efficient and less prone to error by shortening the list of words against which to compare the unknown word.

For this reason, methods of language analysis, such as semantic grammars and case-frame instantiation, which are able to apply strong top-down constraints to their recognition, are at a significant advantage when it comes to spelling correction.

Intermediate Representation Language systems:

Intermediate Representation Language systems uses AVL tree Construction and it is very complicated to implement. Largely in response to the problems of transformational grammar, developed a method of expressing a syntactic grammar that was computationally tractable and yet still could capture linguistic generalizations in a short way, in many cases more concisely than transformational grammar itself. The formalism developed for this system was known as an augmented transition network or ATN. It consisted of a recursive transition network (formally equivalent in expressive power to a context-free grammar), augmented by a set of tests t o be satisfied before an arc was traversed and a set of registers that could be used to save intermediate results or global state. The network recognizes simple sentences with just a subject, verb and direct object in all combinations of active, passive, declarative, and interrogative.

Very large ATN grammars of several hundred nodes have been developed that capture large subjects of English. However, ATNs also have several disadvantages:

Complexity and Non-Modularity: As the coverage of an ATN increases, so does its structural complexity. It becomes extremely difficult to modify or augment an existing ATN without causing large numbers of unforeseen side-effects. For instance, if another outgoing arc is added to a node with a large number6 incoming arcs in order to handle an additional type of phrase that is a valid continuation of the parse represented by one of the incoming arcs, it could to lead spurious and incorrect parses when the node is reached via a different incoming arc.

Fragility: The current position in the network is a very important piece of state information for the operation of an ATN. If an input should be slightly ungrammatical, even by a single word, it is very hard to find the appropriate state to jump to that would enable the parse to continue.

Inefficiency through Back Tracking Search: The natural way to search an ATN is through backtracking. Because intermediate failures are not remembered in such a search, major inefficiencies can result through repetition of the same sub parses arrived at through different paths through the network. Chart parsing techniques were designed as alternatives to ATNs precisely to avoid these inefficiencies.

Inefficiency through Meaningless Parsers: Normally the grammar of an ATN is purely syntactic and a complete syntactic parse is produced before any semantic interpretation is performed. In that situation, many spurious meaningless parses can be produced, especially if the grammar is large and comprehensive. To combat this, recent parsers in the ATN tradition have tried to interpret each constituent as it was produced, thus preventing complete parses based on constituents that could be predicted to be nonsensical.

Conclusion: We are proposing EFLEX system (A Flexible and Efficient Natural Language Query interface to databases) to tackle challenges in NLIDBs. The empirical validation of the prototype shows improved performance. We have designed a new product by name EFLEX and hopefully will be accepted and used by the society for their ease in data

retrieval through queries given in natural language. We are also planning to file for patent of EFLEX (around 18 claims) with Indian Patent office, Mumbai. We validated EFLEX model using the following:
Statistical analysis technique called ANOVA showed linearity and strong correlation with state of the art. The results validated the hypothesis. ANOVA results confirm difference between proposed EFLEX System and state-of-the-art cannot be considered. We evaluated product EFLEX using SUMI questionnaire. SUMI showed summarized quantification of user experience. The results showed that the product PS had the following features: high learn ability, need of less keystrokes and an easier interface. The scope for improvement in the product was in the following areas: proper popup of error messages, consistency, documentation and better retrieval of data. The evaluation was performed on the first version of the software. The benchmark for evaluation being PRECISE our proposed EFLEX System reaches at par or beyond performance. Our goal was to evaluate correct sql translations for natural language queries. The accuracy on text input is 93.9. The results also showed improvement as parser showed improvement.

References:

[1]. Androutsopoulos, G.D. Ritchie, P. Thanisch, "Natural Language Interfaces to Databases – An Introduction",arXiv:cmp-lg/9503016v2 16 Mar 1995
[2].W.A. Woods, R.M. Kaplan, and B.N. Webber", The Lunar Sciences Natural Language Information System: Final Report. BBN Report 2378", Bolt Beranek and Newman Inc.,Cambridge, Massachusetts, 1972.
[3].G. Hendrix, E. Sacerdoti, D. Sagalowicz, and J. Slocum, " Developing a Natural Language Interface to Complex Data." ACM Transactions on Database Systems, 3(2):105–147,1978.
[4].E.F. Codd, "Seven Steps to RENDEZVOUS with the Casual User.", In J. Kimbie and K. Koffeman, editors, Data Base Management. North-Holland Publishers, 1974
[5]. D.L. Waltz, "An English Language Question Answering System for a Large Relational Database.", Communications of the ACM, 21(7):526–539, July 1978.
[6].R.J.H. Scha, "Philips Question Answering System PHILIQA1.", In SIGART Newsletter, no.61. ACM, New York, February 1977.
[7].D. Warren and F. Pereira, "An Efficient Easily Adaptable System for Interpreting Natural Language Queries", Computational Linguistics, 8(3-4):110–122, July-December 1982 .
[8].B.J. Grosz, "TEAM: A Transportable Natural-Language Interface System.", In Proceedingsof the 1st Conference on Applied Natural Language Processing, Santa Monica, California, pages 39–45, 1983.
[9]. B.J. Grosz, D.E. Appelt, P.A. Martin, and F.C.N. Pereira. "TEAM: An Experiment in the Design of Transportable Natural-Language Interfaces", Artificial Intelligence, 32:173–243, 1987.
[10]. P. Martin, D. Appelt, and F. Pereira, "Transportability and Generality in a Natural-Language Interface System.", In B.J. Grosz, K. Sparck Jones, and B.L. Webber, editors, Readings in Natural Language Processing, pages 585–593. Morgan Kaufmann Publishers, California, 1986.
[11].D. W. Williams, J. Huan, and W. Wang, "Graph database indexing using structured graph decomposition," in ICDE, 2007.
[12].B. Gao, M. Ester, J. Cai, O. Schulte, and H. Xiong, "The min. consistent subset cover problem and its applications in data mining," in KDD, 2007.
[13].M. Minock, "A phrasal approach to natural language interfaces over databases," in NLDB, 2005.
[14]. E. Sneiders, "Automated question answering using question templates that cover the conceptual model of the database," in NLDB, 2002.
[15]. O. Hartig and R. Heese, "The SPARQL query graph model for query optimization," in ESWC, 2007.
[16]. C. Chen, X. Yan, P. S. Yu, J. Han, D.-Q. Zhang, and X. Gu, "Towards graph containment search and indexing," in VLDB, 2007.

