Problems With Domain-Independent Natural Language Database Access Systems

Steven P. Shwartz
Cognitive Systems Inc.
234 Church Street
New Haven, Ct. 06510

In the past decade, a number of natural language database access systems have been constructed (e.g. Hendrix 1976; Waltz et al. 1976; Sacerdoti 1978; Harris 1979; Lehnert and Shwartz 1982; Shwartz 1982). The level of performance achieved by natural language database access systems varies considerably, with the more robust systems operating within a narrow domain (i.e., content area) and relying heavily on domain-specific knowledge to guide the language understanding process. Transporting a system constructed for one domain into a new domain is extremely resource-intensive because a new set of domain-specific knowledge must be encoded.

In order to reduce the cost of transportation, a great deal of current research has focussed on building natural language access systems that are domain-independent. More specifically, these systems attempt to use syntactic knowledge in conjunction with knowledge about the structure of the database as a substitute for conceptual knowledge regarding the database content area. In this paper I examine the issue of whether or not it is possible to build a natural language database access system that achieves an acceptable level of performance without including domain-specific conceptual knowledge.

A performance criterion for natural language access systems.

The principle motivation for building natural language systems for database access is to free the user from the need for data processing instruction. A natural language front end is a step above the "English-like" query systems that presently dominate the commercial database retrieval field. English-like query systems allow the user to phrase requests as English sentences, but permit only a restricted subset of English and impose a rigid syntax on user requests. These English-like query systems are easy to learn, but a training period is still required for the user to learn to phrase requests that conform to these restrictions. However, the training period is often very brief, and natural language systems can be considered superior only if no computer-related training or knowledge is required of the user.

This criterion can only be met if no restrictions are placed on user queries. A user who has previously relied on a programmer-technician to code formal queries for information retrieval should be permitted to phrase information retrieval requests to the program in exactly the same way as to the technician. That is, whatever the technician would understand, the program should understand. For example, a natural language front end to a stock market database should understand that

(1) Did IBM go up yesterday?

refers to PRICE and not VOLUME. However, the system need not understand requests that a programmer-technician would be unable to process, e.g.

(2) Is GEMCO a likely takeover target?

That is, the programmer-technician working for an investment firm would not be expected to know how to process requests that require "expert" knowledge and neither should a natural language front end. If, however, a natural language system cannot achieve the level of performance of a programmer-technician it will seem stupid because it does not meet a user's expectations for an English understanding system.

The "programmer-technician criterion" cannot possibly be met by a domain-independent natural language access system because language understanding requires domain-specific world knowledge. On a theoretical level, the need for a knowledge base in a natural language processing system has been well-documented (e.g. Schank & Abelson 1977; Lehnert 1978; Dyer 1982). It will be argued below that in an applied context, a system that does not have a conceptual knowledge base can produce at best only a shallow level of understanding and one that does not meet the criterion specified above. Further, the domain-independent approach creates a host of problems that are simply non-existent in knowledge-based systems.

Problems for domain-independent systems: inference, ambiguity, and anaphora.

Inferential processing is an integral part of natural language understanding. Consider the following requests from PEARL (Lehnert and Shwartz 1982; Shwartz 1982) when it operates in the domain of geological map generation:

(3) Show me all oil wells from 1970 to 1980.
(4) Show me all oil wells from 6000 to 7000.
(5) Show me all oil wells 1 to 2000.
(6) Show me all oil wells 40 to 41, 80 to 81.

A programmer-technician in the petrochemical industry would infer that (3) refers to drilling dates, (4) refers to well depth, (5) refers to the map scale, and (6) refers to latitude/longitude specifications.
Correct processing of these requests requires inferential processing that is based on knowledge of the petrochemical industry. That is, these conventions are not in everyone's general working knowledge of the English language. Yet they are standard usage for people who communicate with each other about drilling data, and any system that claims to provide a natural language interface to a data base of drilling data must have the knowledge to correctly process requests such as these. Without such inferential processing, the user is required to spell out everything in detail, something that is simply not necessary in normal English discourse.

Another problem for any natural language understanding system is the processing of ambiguous words. In some cases disambiguation can be performed syntactically. In other cases, the structure of the database can provide the information necessary for word sense disambiguation (more on this below). However, in many cases disambiguation can only be performed if domain-specific, world knowledge is available. For example, consider the processing of the word "sales" in (7), (8) and (9).
(7) What is the average mark up for sales of stereo equipment?
(8) What is the average mark down for sales of stereo equipment?
(9) What is the average mark up during sales of stereo equipment?
(10) What is the average mark down during sales of stereo equipment?

These four requests, which are so nearly identical both lexically and syntactically, have very distinct meanings that derive from the fact that the correct sense of "sales" in (7) is quite different from the sense of "sales" intended in (8), (9), and (10). Most people have little difficulty determining which sense of "sales" is intended in these sentences, and neither would a knowledge-based understander. The key to the disambiguation process involves world knowledge regarding retail sales.

Problems of anaphora pose similar problems. For example, suppose the following requests were submitted to a personnel data base:

(11) List all salesmen with retirement plans along with their salaries.

(12) List all offices with women managers along with their salaries.

While these requests are syntactically identical, the referents for "their" in (11) and (12) occupy different syntactic positions. As human information processors, we have no trouble understanding that salaries are associated with people, so retirement plans and offices are never considered as possible referents. Again, domain-specific world knowledge is helpful in understanding these requests.

Structural knowledge as a substitute for conceptual knowledge.

One of innovations to emerge from the construction of domain-independent systems is a clever mechanism that extracts domain-specific knowledge from the structure of the data base. For example, the resolution of the pronoun "their" in both (11) and (12) above could be accomplished by using only structural (rather than conceptual) knowledge of the domain. For example, suppose the payroll database for (11) were structured such that SALARY and RETIREMENT-PLANS were fields within a SALESMAN file. It would then be possible to infer that "their" refers to "salesmen" in (11) by noting that SALARY is a field in the SALESMEN file, but that SALARY is not an entry in a RETIREMENT-PLANS file.

Unfortunately, this approach has limited utility because it relies on a fortuitous database structure. Consider what would happen if the data base had a top-level EMPLOYEES file (rather than individual files for each type of employee) with fields for JOB-TYPE, SALARY, COMMISSIONS, and RETIREMENT-PLANS. With this database organization, it would not be possible to determine that

(13) List all salesmen who have secretaries along with their commissions.

"their" refers to "salesman" and not "secretaries" in (13) on the basis of the structure of the database. To the naive user, however, the meaning of this sentence is perfectly clear. A person who couldn't determine the referent of "their" in (13) would not be perceived as having an adequate command of the English language and the same would be true for a computer system that did not understand the request.

Pitfalls associated with the domain-independent approach.

In a knowledge-based system such as PEARL, a natural language request is parsed into a conceptual representation of the meaning of the request. The retrieval routine is then generated from this conceptual representation. As a result, the parser is independent of the logical structure of the database. That is, the same parser can be used for databases with different logical structures, but the same information content. Further, the same parser can be used whether the required information is located in a single file or in multiple files.

In a domain-independent system, the parser is entirely dependent on the structure of the database for domain-specific knowledge. As a result, one must restructure the parser for databases with identical content but different logical structure. Similarly, the output of the parser must be very

different when the required information is contained in multiple files rather than a single file.

Because of their lack of conceptual knowledge regarding the database, domain-independent systems rely heavily on key words or phrases to indicate which database field is being referred to. For example,

(14) What is Bill Smith's job title?

might be easily processed by simply retrieving the contents of a JOB-TITLE field. Different ways of referring to job title can also be handled as synonyms. However, domain-independent systems get into deep trouble when the database field that needs to be accessed is not directly indicated by key words or phrases in the input request. For example,

(15) Is John Jones the child of an alumnus?

is easily processed if there exists a CHILD-OF-AN-ALUMNUS field, but the query

(16) Is one of John Jones' parents an alumnus?

contains no key word or phrase to indicate that the CHILD-OF-AN-ALUMNUS field should be accessed. In a knowledge-based system, the retrieval routine is generated from a conceptual representation of the meaning of the user query and therefore key words or phrases are not required. A related problem occurs with queries involving aggregation or quantity. For example,

(17) How many employees are in the sales department?

might require retrieving the value of a particular field (e.g. NUMBER-OF-EMPLOYEES), or it might require totalling the number of records in the EMPLOYEE file that have the correct DEPARTMENT field value, or, if the departments are broken down into offices, it might require totalling the NUMBER-OF-EMPLOYEES field for each office. In a domain-independent system, the correct parse depends upon the structure of the database and is therefore difficult to handle in a general way. In a knowledge-based system such as PEARL, the different database structures would simply require altering the mapping between the conceptual representation of the parse and the retrieval query.

Finally, this reliance on database structure can lead to wrong answers. A classic example is Harris' (1979) "snowmobile problem". When Harris' ROBOT system interfaces with a file containing information about homeowner's insurance, the word "snowmobile" is defined as any number > 0 in the "snowmobile field" of an insurance policy record. This means that as far as ROBOT is concerned, the question "How many snowmobiles are there?" is no different from "How many policies have snowmobile coverage?" However, the correct answers to the two questions will often be very different. If the first question is asked and the second question is answered, the result is an incorrect answer. If the first question cannot be answered due to the structure of the database, the system should inform the user that this is the case.

Conclusions.

I have argued above that conceptually-based domain-specific knowledge is absolutely essential for natural language database access systems. Systems that rely on database structure for this domain-specific knowledge will not achieve an acceptable level of performance -- i.e. operate at the level of understanding of a programmer-technician.

Because of the requirement for domain-specific knowledge, conceptually-based systems are restricted to limited domains and are not readily portable to new content areas. However, eliminating the domain-specific conceptual knowledge is throwing the baby out with the bath water. The conceptually-based domain-specific knowledge is the key to robust understanding.

The approach of the PEARL project with regard to the transportability problem is to try and identify areas of discourse that are common to most domains and to build robust modules for natural language analysis within these domains. Examples of such domains are temporal reference, location reference, and report generation. These modules are knowledge-based and can be used by a wide variety of domains to help extract the conceptual content of a request.

REFERENCES

Dyer, M. (1982). In-Depth Understanding: A Computer Model of Integrated Processing for Narrative Comprehension. Yale University, Computer Science Dept., Research Report #219.

Harris, L. R. (1979). Experience with ROBOT in 12 commercial natural language data base query applications. Proceedings of the 6th International Joint Conference on Artificial Intelligence.

Hendrix, G. G. (1976). LIFER: A natural language interface facility. SRI Tech. Note 135. Dec. 1976.

Lehnert, W. (1978). The Process of Question Answering. Lawrence Erlbaum Associates, Hillsdale, New Jersey.

Lehnert, W. and Shwartz, S. (1982). Natural Language Data Base Access with Pearl. Proceedings of the Ninth International Conference on Computational Linguistics. Prague, Czechoslovakia.

Sacerdoti, E. D. (1978). A LADDER user's guide. Technical Note 163. SRI Project 6891.

Schank, R. C. and Abelson, R. (1977). Scripts, Plans, Goals and Understanding. Lawrence Erlbaum Associates, Hillsdale Ner Jersey, 1977.

Shwartz, S. (1982). PEARL: A Natural Language Analysis System for Information Retrieval (submitted to AAAI-82/applications division).

Waltz, D. L., Finin, T., Green, F., Conrad, F., Goodman, B., Hadden, G. (1976). The planes system: natural language access to a large data base. Coordinated Science Lab., Univ. of Illinois, Urbana, Tech. Report T-34, (July 1976).