

EXPERIENCE WITH ROBOT
IN 12 COMMERCIAL NATURAL LANGUAGE DATA BASE QUERY APPLICATIONS

Larry R. Harris
Department of Mathematics
Dartmouth College
Hanover, New Hampshire 03755

The ability to understand Natural Language has long been a goal of Artificial Intelligence research, and it is still far from being solved. However, in the early 1970's the AI research techniques reached a point whereby certain applications became feasible for the first time. Since that time, several systems such as PLANES[1], LIFER[2], and ROBOT[3,4,5] have been built that have demonstrated that the current state of the art is sufficient for quite good natural language data base query.

The implementation of the ROBOT system has been geared for high performance and installability in actual real world environments. As such, it offers the AI research community some insight into the difficulties encountered when putting the current AI technology in the hands of people in the real world. This paper discusses the unexpected linguistic and semantic difficulties encountered in the 12 commercial applications to which ROBOT has been applied during the last year and a half.

1. INTRODUCTION

There are many differences between natural language systems developed for research and natural language systems developed for production. Not the least of these differences is the fact that the production systems are eventually used in the absence of the developers of the system. As such, valuable insights into the problem of natural language processing can be gained. In spite of the fact that all of us speak English, none of us know exactly what the user is going to type. Beyond this, we don't know how the user's questions will change after several months of use. This is valuable information that only systems that have actually been installed for long periods of time in the user's environment can provide.

The ROBOT system has been applied to 12 different applications during the past 18 months. In each case, the system has been "turned over" to users for evaluation. The phrase "turned over" is jargon of the software industry meaning that the program is now completely in the hands of the users. This is particularly difficult to do for natural language systems, since so much application-specific tuning is required.

But the system must be officially turned over if we wish to observe the unencumbered user. Naive users, in particular, are greatly influenced when technicians or researchers are observing their use of a computer system. It is only after the system is completely in their hands that we can begin to gain accurate data on how well the system satisfies their needs.

2. THE 12 APPLICATIONS

ROBOT has now been completely turned over to users at the following companies: Bigelow-Sanford Carpet, Commercial Union Assurance, Du Pont Chemical, Planning Research Corporation, Dunn and Bradstreet and Software Ag of North America.

The following is a list of the different data bases to which ROBOT has been applied in these installations -- By ROBOT development staff: Customer file, Homeowner's Insurance file, Employee relations file, Car file, Employee file, Data dictionary, Financial file and budget control; by local staff: Sales file, Membership file, Accounting file and Planning file.

The variation in the different discourse domains of these applications illustrates the importance of the domain-independent parsing techniques employed by ROBOT (see [5]). The sizes of the applications range from 1,000 records in the test files to over 2.4 million records.

3. ACTUAL USER REQUESTS

The following questions were selected from log files in response to a survey requesting interesting queries that have been asked by local users that ROBOT was able to answer. Although this sample is biased towards requests that ROBOT can handle, it does demonstrate the level of sophistication that users expect from the system.

SHOW SUBTOTALS OF DIRECT COMMISSION BY MONTH FOR 1ST QTR '76 IN REGION M WHERE NET ARISING AMOUNT IS AT LEAST \$150,000.

FOR DOMESTIC MACHINERY, PRINT THE JULY 1976 SALES DOLLARS AND SALES QUANTITIES, SORTED BY SALES CLASS.

FOR COMMERCIAL DIVISION AND THOSE PRODUCTS IN EXCESS OF \$500,000 YEAR-TO-DATE, SHOW ME ALL TIME PERIODS WHOSE PRE-TAX EARNINGS ARE MORE THAN \$200,000 AND PRINT THEIR NAME.

FOR SUMMARIES WITH LOSS RATIO GREATER THAN 200, REPORT REGION, BRANCH AND LOSS RESERVE BY MONTH.

PRINT ALL PRODUCTS WHOSE NOVEMBER 1977 SALES QUANTITIES ARE BETWEEN 100,000 AND 300,000 AND WHOSE PRE-TAX EARNINGS EXCEED \$1,000,000.

4. PROBLEMS ENCOUNTERED

Although the overall experience with ROBOT at all of these sites has been quite good, there have nonetheless been several very difficult problems of a linguistic or semantic nature that have surfaced. The remainder of this paper discusses these problems and potential solutions for them. A problem is defined as any undesired response to a request that cannot be produced by appropriate dictionary definitions. Note that this is a rather strict definition since it presumes that the parser and the set of primitives that can be used to create dictionary definitions are fixed. Even the simplest change to the APL grammar is considered a major problem because this cannot be done by local staff at the site. A solution is defined to be a

domain-independent extension to ROBOT and/or the dictionary utility that allows the desired response to be produced by appropriate dictionary definition. The importance of the domain-independence of the solution should be clear from the variety of applications that ROBOT currently deals with. It is also clear that researchers who do not share our concern for domain-independent solutions may find the following problems very easy to solve in any given application. With this disclaimer we can now begin discussing the problems known as:

The "ME OK IN" Problem,
The Snowmobile Problem,
The Matrix Problem, and
The Summary Problem.

4.1 The "ME OK IN" Problem

This problem arises as a result of coded data in the data base. The dictionary must contain enough information about the coding scheme so that it can be made transparent to the user. Typically, users want to have the flexibility of referring to the data in either the long form or the coded form. For example, in a coded state field, the user may want to refer to New Hampshire either as NH or New Hampshire. The specific problem arises when a coded item intersects with other words. For example, the abbreviation for Maine is ME, and the abbreviation of Oregon is OK and the abbreviation for Indiana is IN. It is particularly hard for the parser to tell in some uses of the word IN whether its the preposition "in" or Indiana. ROBOT normally deals with lexical ambiguity by considering all interpretations and then heuristically selecting the best one. This approach is fine in most circumstances, but does not work very well when one interpretation is much more common than the other. For example, the request "PRINT FOR ME THE NAMES OF ALL SECRETARIES." would be detected as ambiguous. Does it refer to all secretaries or just those in Maine? When you are aware of the information the program has to work with, it's hard to fault it for entering into a clarification dialog with the user in cases like this. But when you are constantly badgered by the system to help weed out such obscure interpretations, it can get very tiresome.

The proper solution for this problem seems to be an improved heuristic that is perhaps a little more ruthless in weeding out interpretations using coded data items. Unfortunately the environment does not provide the normal clues, such as capitalization, that help people make this distinction. Also the typewritten form of expressions tends to exacerbate the problem since people tend to use coded references rather frequently as a way of reducing typing. The current, admittedly short term, solution of ROBOT is to restrict selected coded data items so that they cannot be used as words in a request. This allows the person that builds the dictionary to decide whether the ambiguous meaning is desirable or not. This has the effect of forcing users to refer to Indiana, Oregon and Maine in their full form, where other states can be referenced either way.

4.2 The Snowmobile Problem

This problem arose in the context of the question "how many snowmobiles are there?". This question was directed towards a homeowner insurance file that included information on snowmobile coverage. The interpretation ROBOT gave to this request was "how many policies have snowmobile coverage?". Needless to say, the answers to these questions are very different. The subtlety of this problem isn't so much linguistic as it is in the data representation. If the file being queried would have contained one record for each snowmobile, then ROBOT would have answered it properly. But the file contained one record per policy and a field containing the number of snowmobiles covered by that policy. ROBOT's dictionary defined the meaning of snowmobile to be any number greater than 0 in this field. From this it should be clear why ROBOT first found the policies with at least one snowmobile and then counted the policies.

The point is that the computation of the answer depends on how the data is stored. The syntactic structure of the request is of little help in figuring what computation is required to answer the request. Consider the following requests:

"how many snowmobiles are there?"
 "how many secretaries are there?"
 "how many salaries are there?"

These are all syntactically identical. One might even argue that they are semantically homomorphic. But each request requires that the answer be computed in a totally different way. We have already seen that the snowmobile request requires summing the contents of the snowmobile field. Assuming that secretary is an instance of a job title, then the number of secretaries is computed by counting the number of records with secretary occurring in the job field. The proper response to the request about salaries may well be to give the number of different salaries that occur in the file. Needless to say, the algorithm required to compute this is very different from the other "How many" questions. ROBOT is currently being fitted with enough domain-independent facilities to permit the inclusion of enough domain-specific knowledge to know which algorithm to employ in each case.

4.3 The Matrix Problem

This problem arises because there are many ways of referring to data aggregates. For example, one of the applications had historical data in several fields such as profit and total sales. Thus, users could refer to the profit in 1978 or the 1972 total sales. In many requests the year specifier would refer to several fields. For example, "In 1972, what were the profits and total sales." In other cases several years might apply to one field. For example, "Give me the total sales for 1972 and 1973."

These questions could not be processed because they require combining the syntactic information gained from the parse with domain-specific information about the data base. The answers to these questions are generated in different ways depending on whether the years themselves are stored as data items. In the particular instance in which this problem was first encountered, ROBOT required very direct scoping of the time referent. For example: "List the 1972 profits and the 1972 total sales?", or "List the 1972 total sales and the 1973 total sales."

Because this was perceived as a rather severe linguistic limitation, ROBOT's dictionary facility now allows the specification of how the time referents and the periodic fields are related. This information can then be

integrated with the syntactic information generated by the parser to determine the set of fields being referenced. In fact, one of the actual requests that appeared in the user survey required this facility. The phrase "the July 1976 sales dollars and sales quantities" could not have been understood without it.

4.4 The Summary Problem

The summary problem arises because there may be two different ways to answer the same question within the same data base. This often occurs because a certain amount of summarization may occur as a data base is being prepared for query by extracting it from a larger production file. For example, if a file containing information on all sales that took place in the previous month is to be queried, it is often extracted from a larger file containing all sales. During this extraction process, it may be desirable to calculate certain summary statistics and add them to the file. Thus all of the total sales may be computed for each salesman.

The problem occurs when a person asks for a piece of summarized information. For example, "what are the total sales for salesman Smith?" The desired result is quite obvious. The program should merely print the already computed result rather than computing it again from the raw data. The reason that this is difficult is because the user's request is independent of the fact that the summarized information already exists, and thus the parser generates an interpretation for computing the total from the raw data. In the example, this would require retrieving all of the detail records for the salesman Smith and adding up the sales amounts. Since this would be prohibitive in large data bases, some way of avoiding this recalculation must be found.

If the dictionary contained a semantic description of what summarized information was precalculated, then the semantic structure for the request would be compared against these templates. When a match occurred, the retrieval of a single field could replace the computational portion of the semantic structure for the request. This approach would require the definition of some relatively complex semantic structures within the dictionary, since the summarized information could be of

arbitrary complexity. ROBOT's dictionary building facility will soon be enhanced to allow the semantic description of summarized information.

5. SUMMARY

The ROBOT system is the first natural language system to be put into production use in a variety of real world environments. The problems encountered in these applications are of interest to the research community because in many cases they challenge the fundamental principles on which virtually all current natural language query systems are based.

We have presented four such problems in this paper that were motivated by actual field experience with ROBOT. As such, they begin to form a minimal specification for the actual required functionality of a natural language data base query system. These problems are among the first difficult issues that future systems must be able to solve if they are to be of significant use in the real world.

REFERENCES

- [1] Waltz, D.L., "An English Language Question Answering System for a Large Relational Data Base", CACM, July 1976, vol 21, no 7, p 520
- [2] Hendrix, G. et al, "Developing a natural language interface to complex data", VLDB Proceedings, Oct 1977
- [3] Harris, L.R., "User Oriented data base query with the ROBOT natural language query system", IJMMIS (1977) 9, pp 697-713
- [4] Harris, L.R., "ROBOT: A High Performance Natural Language Interface for Data Base Query", TR 77-1, Math Dept, Dartmouth College
- [5] Harris, L.R., "Using the Data Base Itself as a Semantic Component to Aid in the Parsing of Natural Language Data Base Queries", TR 77-2, Math Dept, Dartmouth College
- [6] Winograd P., Understanding Natural Language, Academic Press
- [7] Woods, W.A. et al, "The Lunar Sciences Natural Language Information System", B&N Report 2376, June 1972