

PROCESSING NATURAL LANGUAGE FOR AN EXPERT SYSTEM USING A SUBLANGUAGE APPROACH

Elizabeth Liddy and Corinne Lyon Jörgenson
School of Information Studies
and Ernest Sibert and Edmund S. Yu
School of Computer and Information Science, Syracuse University
Syracuse, NY

Abstract

We report an expert system for analyzing free-text responses concerning medical histories in life insurance applications. The system uses a sublanguage grammar based substantially on semantic word classes, and is implemented using a logic-programming formalism. The grammar is intentionally ambiguous, but the system incorporates additional ambiguity rules which reject most inappropriate readings. Setting aside utterances which have grossly misspelled or unrecognizable words, the system produces an appropriate reading for more than 95% of the examples examined.

Introduction

It is increasingly clear from research published in information retrieval (IR) journals and presented at IR conferences, that linguistic analysis needs to play a central role if we are to succeed in solving the problems of information retrieval. Since our tasks revolve around the manipulation of two types of texts, namely queries and documents, the contribution that both theoretical linguistics and the more computational field of natural language processing should make is substantial. In the past, it has been mainly the morphological and syntactic levels of linguistic analysis that have been applied in processes such as stemming and automatic noun phrase identification. The more difficult levels of analysis, such as semantics and discourse, have yet to be fully exploited in our systems and therefore these systems do not adequately represent the *semantic* content of either texts or queries. There are, however, some notable exceptions, including the German system TOPIC [2].

Information retrieval is not alone in its need to represent and manipulate meaning, as any system which functions using natural language text must also be able to process language at these more complex levels. This is particularly true of those expert systems whose input is in the form of natural language. However, expert systems generally function in a single knowledge domain rather than across all areas of knowledge as is the case with traditional IR systems. It could therefore be assumed that solutions to language processing problems found in expert system research might not be useful to the tasks of information retrieval. However, recent IR research using discourse linguistics on the predictable structure of abstracts [3] or on using text structure to automatically construct literature abstracts [4] has shown that use of text-level regularities may alleviate the necessity of dealing with the particularities of a variety of domains. Taking a different approach, some IR researchers have focused their efforts on developing systems which work in one knowledge area alone. PLEXUS, a reference and referral expert system in the domain of gardening by Vickery, Brooks and Robinson [5], and CANSEARCH an expert system for retrieving cancer therapy documents by Pollitt [6] are good examples of this approach.

Our approach is akin to the specialized expert systems of Vickery *et.al.* and Politt in that it too works within

one domain, namely medical histories on life insurance applications, but is also similar in approach to the work of Paice and Liddy in that it relies on the regularities of a particular text-type to reveal something about the meaning of the natural language text. What may be of most potential interest to the field of IR is that the sublanguage approach which has shown itself useful in this research, appears to be generalizable to the analysis of users' queries across databases. Linguistic theory suggests that, given a common discourse situation and a common task, such as reporting one's medical history or phrasing a query to a database, the natural language utterances so produced would have predictable structure and key phrasings across individuals in spite of the unique details of the particular situation [7].

The work we report in this paper derives from a very practical need to process real data from naturally occurring texts for use in an operational system currently under development. Our work is part of the development of an expert system for life insurance underwriting which processes data from application forms which are completed by insurance sales people in the presence of the applicant. The form has twenty-seven fields, most requiring either yes/no, multiple choice, or responses restricted in possible range, such as dates or names. The responses to these structured questions can be interpreted in relatively straight-forward ways and the rules in the knowledge base for deciding whether or not to write the life insurance policy using these regularized responses were already well developed before we joined the project. The problem we dealt with was the data-field in which the agent can enter free-text comments on the particulars gathered from the applicant about his/her personal medical history.

The sublanguage approach immediately suggested itself as most appropriate. In that the applicants were instructed to comment only on their personal medical histories, the range of semantic data to be understood was quite restricted. In addition, the fact that the physical size of the field on the application was roughly 7"x2", strongly suggested that the responses would be brief utterances, possibly lacking the syntactic regularities of well-formed sentences.

Sublanguage Grammar - The notion of sublanguage suggests that there are restricted subsets of linguistic phenomena which can be observed to occur when language is used within a certain semantic field for a specific pragmatic purpose. A consequence of this situation is that natural language processing systems dealing with restricted language samples need not be capable of handling the full range of a language's resources in terms of either lexical items or syntactic structures. Z. Harris [8] was the first to suggest that sublanguages exist within a language, the same as sub-systems exist in mathematics. Although the set of sentences in a sublanguage is a subset of the sentences in the whole language, it is not necessarily the case that the grammar of the sublanguage is included in the grammar of the whole language.

Sager's Linguistic String Project is the largest, cohesive effort in the sublanguage research effort [9]. In addition, the sublanguage approach has been usefully applied in automatic translation of weather reports [10]; stock market reports [11]; legal language [12]; and Navy messages [13].

Sublanguage Components

The free-text responses which deal with the applicant's recent medical problems and treatments have word usage patterns which are extremely varied and require a multi-level approach. The sublanguage components necessary to handle the variety of input and the constituent rules were derived by analysing thousands of individual records for content, structure, and meaning. While many records consist of two or three words and can be approached by simply using lexical analysis and a semantic-based syntactic structure such as "Exam - Outcome," we found no overall consistency in pattern for longer utterances. Thus larger patterns such as "problem- treatment-outcome" were not ultimately useful as the lengthier utterances were more a "stream of consciousness" response such as

High blood pressure took Normodyne but now Vasotec due to side effects.

The sublanguage analysis consists of four levels or stages of analysis

- Preprocessing
- Lexical Analysis
- Adjacency Analysis
- Filtering for Ambiguity

The joint function of these analyses is to divide and organize the text into discrete lexical units which can then be passed to the parser and plotted into the Information Format. The Information Format itself constitutes a part of the sublanguage grammar, as its structure was determined by analysis of the utterances in the domain of the sublanguage. It contains three broad categories: 1) Client, 2) Client History which includes problem words, incident words, and medical action words, and 3) Miscellaneous Words. Each category also may have associated Modifiers and Negatives. More detailed levels of classification such as disease, injury, or psychological aspect are contained within these broad categories.

At the preprocessing level, abbreviations and contractions are checked and converted to a full, standardized form. Specialized scanning techniques were developed to handle frequently occurring but non-standard abbreviations such as those for "check-up," which may or may not include a hyphen and may be abbreviated as "ckp", "cup", "chp", "chup", or any of other numerous ways. Certain carefully chosen medical terms are also handled in this way. Commonly misspelled words ("urinary track infection," "tubaligation") are also normalized. One important aspect of the preprocessing analysis is the expansion of contractions which contain a negative, allowing the program to recognize and correctly place the negative and thus determine the presence or absence of disease. The preprocessing level has the capability of detecting certain run-on forms of words and accidental inclusion of keyboard symbols or punctuation as well.

At the lexical analysis stage, the semantic word classes of words and certain short phrases are assigned. It is possible for a word to be ambiguous (have more than one sense in the lexicon entry) and be tagged with as many as three different word classes. Word classes are determined on the basis of the word's semantic function within the utterance and are often constrained by the domain of the sublanguage. In addition, some

words may appear to be assigned nonstandard word classes because of their peculiar use within the sublanguage. For example, "pregnant" is tagged as DISEASE (DS), while the word "military" is an EXAM in this sublanguage. Some terms may have multiple non-standard classes assigned, for example, the word "child" can function semantically as either a RELATIVE-CLIENT (RL) or EXAM (EX) or a time-modifier (TMOD). In terms of grammatical class analysis, some words which are not actually prepositions may be assigned to the Word Class PRP based on their role of introducing words which function as modifier phrases.

The lexicon, whose construction began with the more obvious word classes for this sublanguage such as body-part (BP), symptom (SYM), and exam (EX), was later expanded to include a more detailed classification of terms such as PHYSICAL EVENT (PE), (e.g., "car accident"), FOREIGN OBJECT (FO), (e.g., "sliver"), and PSYCHOLOGICAL ASPECT (PSY) (e.g., "divorce") which originally were simply classed as "incidents". This detailed level of analysis in the lexicon enables the processor to handle a great variety of sentence constructions without a fully-developed parser. The lexicon at present contains approximately 3500 words and can handle about 90% of the vocabulary encountered.

Future refinements of the system would include a secondary lexicon for less common medical conditions and treatments. A sample of the basic word classes is contained in Figure 1 and includes words relating to the broader categories contained in the Information Format.

At the Adjacency Analysis level, phrases composed of members of two or more word classes are tested against rules for combining adjacent words or word class members into larger lexical units. This allows the processor to recognize a large number of phrases without their numerous variations being entered into the lexicon. While there is little reliable consistency in overall utterance structure there is enough consistency within smaller sections of an utterance to make rules of this type useful. Rules are of the general form:

"Word" or "Word Class" ADJACENT "Word" or "Word Class" = "Word Class"

A rule such as:

DISEASE ADJACENT TEST = TEST

allows for a variety of words classed as DISEASE to combine with terms classed as TEST and reduces the size of the lexicon. For example, the words "Strep"

CLIENT HISTORY

Problem

Body-part = BP
Symptom = SYM
Disease = DS
Injury = INJ
Problem Verb = DSVB

Medical Action

Type of Exam = EX
Medical Personnel = DR
Medical Action Verb = DRVB
Medical Institution = MI
Treatment = TR
Medication = RX

Result

Outcome = OC
Test = TS
Result = RS

CLIENT

Client Noun or Pronoun = CN

MODIFIERS

Locator Modifier = LMOD
Time Factor Modifier = TMOD
Amount Modifier = AMT
Negative = NEG
Preposition = PRP
Locator PRP = LPRP
Time PRP = TPRP
Adjective = ADJ
Time Adjective = TADJ

Figure 1. Reduced Data Structure for Client Record

ADJECTIVE. Culture would be assigned to the format as the Word Class TEST. Numerical phrases are also handled by adjacency rules, an example being "150 over 90," a blood pressure reading which is interpreted as a test RESULT. Adjacency rules such as this and ones dealing with dates function within certain numerical constraints. Adjacency rules also appear in the higher-level grammar.

An Ambiguity Filter is needed to handle ambiguous words in the lexicon and ambiguous phrases produced by the Adjacency Analyzer. Disambiguation is achieved by rules which perform both syntactic and semantic analysis. A simple example of syntactic analysis is the determination of Word Class and therefore of placement of words which may be used either as adjectives or adverbs (and which are often grammatically incorrect in the utterance). Semantic analysis is used for another common occurrence: a word which may be either a Verb or another Word Class. An example of this is the word "left" (Verb or Modifier). The program can correctly construe the Modifier form in the utterance "Cancer left breast in hospital" and the Verb form in the utterance "Premature birth left child in hospital" because lexical analysis into Word Classes has provided the necessary semantics for the two nouns ("child" and "breast").

The lexical units produced by the program components discussed above are next passed to the Parser, which plots these discrete units of information into the Information Format. The sentence structures often contain no punctuation or other breaks, but with the prior fine level of analysis the parser is able to provide a correct interpretation of information given out of context. The parser recognizes various Word Class structures which accommodate a variety of forms, including Noun Phrases, Verb Phrases and modifiers. Some of the simpler structures involve modifiers such as Time Modifiers, Adjectives, and Adverbs. As an example, the structure for Time Modifier simply instructs the parser to look for the form TMOD (a discrete Word Class either from the lexicon or created by adjacency rules) followed by an optional second TMOD. In the phrase "at age five years," the numeral is combined with "age" and "years" to become a TMOD in the Information Format. More complex constructions are needed for Conjunctions, Negatives, and Prepositional Phrases, which involve scanning the utterance to determine the end of the Prepositional Phrase. Certain Word Classes, such as a second Negative or another Preposition, will signal the end of a prepositional phrase. Other Word Classes combined with specific prepositions (with possible intervening modifiers) will also consistently signal the end of a phrase. An example of this type of prepositional phrase is the form "in" or "on" plus the Word Class BODY-PART.

"on lower left leg"

The Prepositional Phrase itself can serve two functions: it can be either a modifier phrase or serve lexically as a noun phrase containing a discrete unit of information. Determination of the function of a prepositional phrase is based on the major Word Class of the object of the preposition. A prepositional phrase with a compound object may become two phrases, each containing the preposition and one of the objects. This allows the correct placement of the prepositional phrase in the Information Format when the phrase contains two different Word Classes or when the phrase is incorrectly placed in the utterance.

The Information Format is the final output of the sublanguage analysis and becomes the input for the expert system which determines the insurability of the applicant. As such, it is critical that the system have a

high degree of reliability and accuracy. Further refinements prior to instantiation of the Information Format include the differentiation of verb word classes and further incorporation of syntax to help determine certain ambiguous Word Classes. For instance, a record will often have the type of exam as the first unit of information, and the outcome will be the last information unit. The ability of the program to recognize negation wherever it occurs in the utterance is of major importance, since one of the most important parts of processing the applications is the determination of the presence or absence of a disease or pathological condition.

The Implementation

The programs are written using LOGLISP [14, 15], running with LISP-VM [16] on IBM 370 VM systems. LOGLISP is a logic programming system which integrates Horn clause logic with LISP. The logic system is somewhat akin to PROLOG [17, 18], except that it uses a LISP-like syntax and permits the invocation of LISP functions at any point in the logic. LISP functions may themselves invoke the logic system at any point, perhaps recursively.

The programs accept an utterance, which is taken as a list of words, numerals, and a few punctuation marks, and produce one or more *information formats* representing readings of the utterance. Although extremely rare, it is possible that the programs produce no readings at all for certain utterances. Even utterances with unrecognized words will usually be handled, with such "unknowns" identified in the format.

We use the non-deterministic aspect of logic programming to express the idea that there are many ways an utterance might be understood, and to search for those which are appropriate. As mentioned, words may have several meanings (word classes), and phrases may be parsed in several ways (the grammar is ambiguous). On the other hand, we try to avoid ambiguity, so that most utterances will have exactly one reading. We use both "bottom-up" and "top-down" techniques to analyze an utterance. The bottom-up portions translate the original utterance into a sequence of elementary units with identified word classes. The top-down portions parse this into a sequence of structured phrases (noun phrases, verb phrases, modifier phrases) which are then placed in the information format.

Figure 2 shows the overall organization, which corresponds exactly to the grammatical components described earlier.

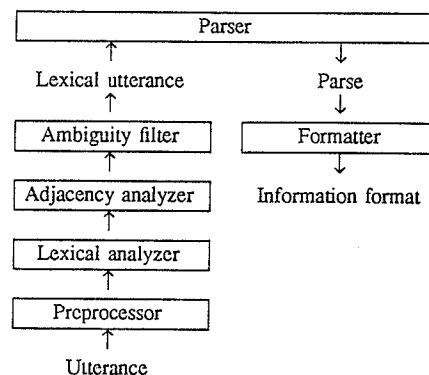


Figure 2.
Overview of analysis

Bottom-up Processing

The initial utterance, however it is obtained, is represented as a list of atoms, either symbols or numbers. The bottom-up portion of the system produces a *lexical utterance*, which is a list of words or phrases tagged with word class identifiers. Because the entries of this list may consist of phrases of a few words, viewed as an indivisible lexical unit, such an entry normally has the form

(<word-class> (<word> . . .))

even when only one word is involved. All of the bottom-up processing is local in nature, that is, it depends only on the immediate context of the word or words in question, though in several instances it takes into account whether these words occur at the beginning or end of the utterance. In some cases we detect phrases which we wish to treat as indivisible units during parsing, but which should be analyzed into finer units in the information format. Such phrases appear in the lexical utterance as entries of the form

(<word-class>
(Phrase (<word-class> (<word> . . .)) . . .))

We could imagine putting phrases within phrases, but have not thus far had occasion to do so.

The program components correspond very closely to the organization of the grammar, described earlier. Here we discuss a few points about the implementation.

Preprocessor - The preprocessor produces a transformed utterance in which recognized variants are represented in standard form. The preprocessor may sometimes enlarge the lexicon temporarily, as when a plural form of an entry in the lexicon is detected.

Lexical and Adjacency Analyzers - Although lexical and adjacency analysis are conceptually quite distinct, the processing of these is closely integrated, the lexicon being used (quite heavily) by the adjacency rules, rather than being applied as a separate phase. The lexicon consists mostly of facts associating word classes with specific words or phrases, with a few rules for numbers. The adjacency rules are described by facts, but include a number of (logic programming) rules which specify the analysis.

Ambiguity Filter - The adjacency analyzer rather often produces two or more analyses. The ambiguity filter implements the rules which attempt to detect inappropriate analyses on the basis of local relationships. The filter never transforms utterances, it simply accepts or rejects them. Certain ambiguities which depend on a larger context are accepted in this phase, but detected during parsing.

Top-down Processing

Parser - The lexical utterance which results from bottom-up processing is presented to the parser, which proceeds left-to-right, more or less in a recursive-descent style. Though the parser is usually guided by the next unit in the utterance, it frequently explores several alternatives in hope of finding one or more correct readings. The parser construes the utterance as a sequence of phrases, usually noun or verb phrases, though unattached modifier phrases may occur as well.

A noun or verb phrase consists of a primary component, the noun or verb, which corresponds to a category in the information format, and various modifying components. An unattached modifier phrase is just that. It will be attached later.

The parser will construe almost anything in some fashion, but in certain cases is designed to fail when

HIGH BLOOD PRESSURE TREATED , NO MEDICATION SINCE JUNE 1985
*** CLIENT HISTORY ***

DISEASE: HIGH BLOOD PRESSURE

*** MEDICAL ACTION ***

MEDICAL VERB: TREATED

MEDICATION: MEDICATION

TIME: SINCE JUNE 1985

NEGATION: NO

Figure 3. Sample Information Format

an unsuitable reading of an ambiguous utterance is detected. A few such cases arise when the proper reading can only be detected by global analysis, usually involving occurrences within prepositional phrases.

Formatter - Most of the programs in the formatter are concerned simply with arranging the utterance in the information format, but some very long-range analysis is included as well. Unattached modifier phrases are attached to noun or verb phrases as determined from the overall parse, and a few very special situations not detected earlier are handled. The example in figure 3 is representative, though the utterance is more grammatical than many we encounter.

Experimental Results

Our specification of the sublanguage grammar and the programs (as delineated above) which apply it to naturally occurring utterances were tested on a sample of 1367 utterances. The sample utterances were comprised of two sets. Set 1, containing 685 utterances, was drawn from the thousands of cases we had used for specifying the sublanguage and developing the programs. Set 2, containing 682 utterances, was drawn from cases which we had not dealt with prior to this testing. The utterances were downloaded from the tapes received from the insurance company and passed directly to the sublanguage programs with no intervention or adjustment by the researchers.

Results (see Table I) will be reported separately for each set, although the figures were remarkably similar, another indication that we are indeed dealing with a linguistically homogeneous sublanguage. Of the 685

Table I: Results from Sets 1 & 2

	Set 1	Set 2
Total Number of Utterances	685	682
Utterances Containing Unknowns due to:		
Misspellings	102	96
Unusual Abbreviations	28	23
Not in Lexicon	25	65
Total Utterances with Unknowns	155	184
Unparsable Utterances	6	8
Total excluded from analysis	- 161	- 192
Testable Sample Utterances	524	490
Parses Produced for Testable Sample	600	532
Average Parses per Utterance	1.145	1.086
Utterances with no Appropriate Parse	19	20
Error Rate	3.6%	4.1%

utterances that comprised Set 1, 155 utterances contained Unknowns. Unknowns are words which are not in the system's lexicon for one of the following reasons. 102 (66%) of these were due to spelling errors and 28 (18%) were due to use of irregular abbreviations. The remaining 25 (16%) of the unknowns were words not yet in the lexicon. In addition, 6 utterances were unparseable, due most likely to an unusual combination of terms on which the adjacency and ambiguity rules as implemented at that time failed.

The unparseable utterances and the utterances with Unknowns were excluded from the test set in order to better measure how well the rules of the grammar and the programs were working. For the remaining 524 utterances which form the test sample for Set 1, a total of 600 parses were produced; an average of 1.145 parses per utterance, better than the objective of no more than 1.5 parses per utterance that we had established with our funder. In addition, we had said that of these multiple parses at least one must be an appropriate reading. In analyzing the test output we were overly severe on judging the readings, taking as inappropriate any reading that is not as precise as possible. Therefore, the so-called error rate of 3.6% is to be interpreted as a very stringent measure.

As can be seen from the results on Set 2, the sublanguage grammar and programs produced very similar results when applied to the hitherto unseen sample of utterances. The procedures were exactly the same as for Set 1, that is, the utterances were downloaded from the tapes and passed directly to the sublanguage programs with no intervention or adjustment by the researchers. The results show that the only major difference was the expected one, a larger number of utterances with Unknowns, 65 or 35%, due to terms which were not in the lexicon. An additional 96 or 52% of the Unknowns were caused by spelling errors. The average number of parses per utterance for Set 2 (1.086) was again better than the proposed 1.5 parses per utterance and even better than Set 1. The error rate (4.1%) is only slightly higher than for Set 1, a result that shows quite convincingly how well the Sublanguage approach allows a system to correctly process new text based on rules developed on samples of the same text-type.

Conclusions

Perhaps we should point out that the particular sublanguage grammar developed for this implementation is distinguished from others reported in the literature by its primary reliance on semantics. In this sublanguage, even traditional syntactic categories such as noun phrases and verb phrases are defined by acceptable combinations of semantically determined lexical units. A few strictly grammatical classes such as 'preposition' are used, but wherever possible the specific semantic role performed by a word, such as 'locator preposition' or 'time preposition', is reflected in the word class assigned.

This sublanguage approach to the task of representing natural language text for use in an expert system for insurance underwriting has produced good results both in terms of correct parses (96.2% across Sets 1 and 2) and number of parses per utterance (1.116 across both Sets). Both the grammar and its implementation in LOGLISP have shown themselves to be reliable and workable. The system's high level of performance strongly suggests that the utterances being treated do constitute a sublanguage, whose specialized vocabulary and syntax we have been able to capture in our programs. We are currently involved in further refinements to the grammar and in integrating the NLP

component with the knowledge base used by the other components of the expert system.

We believe that the sublanguage technique described herein offers a useful approach for analysis of other types of natural language utterances, such as queries to a retrieval system. Although the particulars of individual information needs may vary, as do the illnesses and medical treatments in this sublanguage, the basic nature of the act of information seeking may be sufficiently homogeneous to make the sublanguage approach worth further investigation in information retrieval.

Notes

- [1] The investigation reported herein was supported by the Mutual Life Insurance Company of New York through the Center for Advanced Technology in Computer Applications and Software Engineering at Syracuse University.
- [2] V. Hahn and V. Reimer, *The TOPIC Project: Text-oriented procedures for information management and condensation of expository texts*, Constance, W. Germany, University of Constance (1985).
- [3] E.D. Liddy, *Discourse-level structure of natural language texts: An exploratory study of empirical abstracts*, Ph.D. dissertation, Syracuse University (1988).
- [4] C. Paice, "Constructing literature abstracts by computer: Techniques and prospects", *Information Processing & Management* (in press).
- [5] A. Vickery, H. Brooks, and B. Robinson, "A reference and referral system using expert system techniques", *Journal of Documentation* (1987) 1-23.
- [6] S. Pollitt, "CANSEARCH: An expert systems approach to document retrieval", *Information Processing & Management* (1987) 119-138.
- [7] N. Sager, C. Friedman and M. Lyman, *Medical Language Processing*, (Reading, MA: Addison-Wesley, 1987).
- [8] Z. Harris, *Mathematical Structures of Language*, (New York: Wiley, 1968).
- [9] N. Sager, C. Friedman and M. Lyman, *op. cit.*
- [10] J. Lehrberger, "Automatic translation and the concept of sublanguage", in R. Kittredge and J. Lehrberger (Eds.), *Sublanguage: Studies of Language in Restricted Semantic Domains* (Berlin: de Gruyter, 1982), 81-106.
- [11] R. Kittredge, "Variation and homogeneity of sublanguages", *Ibid.*, 108-137.
- [12] V.R. Charrow, J. Crandall, and R.P. Charrow, "Characteristics and functions of legal language", *Ibid.*, 175-190.
- [13] E. Fitzpatrick, J. Bachenko, and D. Hindle, "The status of telegraphic sublanguages" in R. Grishman and R. Kittredge (Eds.), *Analyzing language in restricted domains* (Hillsdale, NJ: Lawrence Erlbaum, 1986).
- [14] J.A. Robinson and E.E. Sibert, "LOGLISP: Motivation, design and implementation", in K.L. Clark and S.-A. Tärnlund (eds.), *Logic Programming* (London: Academic Press, Inc., 1982), 299-313.
- [15] J.A. Robinson, E.E. Sibert, and D.W. Aha, *The LOGLISP Programming System LISP/VM Version (VM-V3M1)*, (Syracuse University, Logic Programming Research Center, 1986).

- [16] IBM Corporation, *LISP/VM User's Guide* (Houston, 1984).
- [17] A. Colmerauer, H. Kanoui, R. Pasero, and P. Roussel, *Un système de communication homme-machine en française*, (Research Report, Aix-

Marseille: Groupe Intelligence Artificielle, Université Aix-Marseille II, 1973).

- [18] R.A. Kowalski, "Predicate logic as programming language", in *Proceedings IFIP 74* (Amsterdam: North Holland Publishing Co., 1974), 569-574.