INTRODUCTION

For several years we have been involved at MITRE in the development of an on-line text-processing system intended for use by information analysts in the establishment and manipulation of their own personal files. Our primary motivation in this program as in its predecessor has been to incorporate linguistically based techniques for analyzing the information content of language in what we hope will be practical, useful systems.

The previous program, the "English Preprocessor Project," was directed toward developing a capability that would allow formatted data files to be accessed by natural language queries (Walker, 1965). At the time we began it in 1961, complex data retrieval requests could be formulated only by programmers who were intimately familiar with the structure of the data base in question. Bolstered by the conviction that computer-based systems should be responsive directly to the people who were using their products, we designed such a system, concentrating our efforts in particular on its "front end."

Our initial goal was the development of a syntactic analysis procedure that would process both the query sentences and declarative sentences which could be used to update the data base itself. (We hoped also to be able to generate well-formed sentences as responses to a query.) Since our linguistic inspiration was (and still is) Chomsky's transformational approach (Chomsky, 1957, 1965; etc.), our product was a syntactic analysis procedure for transformational grammars (Zwicky, et al., 1965; Walker, et al., 1966).

During the time it took to develop the MITRE Analysis Procedure, other groups had established English-like query languages for data retrieval systems that did provide reasonable access for unsophisticated users. However, during this period there was not a corresponding increase in complexity of the techniques for file manipulation which would have justified further work on natural language analyzers for input processing. Consequently, we shifted the emphasis of our program to work with text files.

(It is appropriate to remark here, parenthetically, that some new approaches to file access and data base manipulation hold considerable promise for sustaining the use of natural language input in data retrieval. I find particularly interesting in relation to our original approach the recent work of Woods (1968), Green and Raphael (1968), and Becker (1969), but for more general reference, see the survey by Simmons (1969) on "Natural Language Question Answering Systems: 1969.")

Our work in text-processing began as an attempt to see how the MITRE Analysis Procedure could be used practically for analyzing textual data. We recognized then (and still do) that no grammar has been written within any linguistic theory, transformational or otherwise, which is adequate for processing random text passages automatically. Of course, only recently have these theories begun to inspire semantic research that is at all encouraging (e.g., for transformational theory: Fillmore, 1968; McCawley, 1968; Bierwisch, 1967). Consequently, our interest has been in designing an on-line system where the user could control the material to be analyzed, simplifying it where necessary, could supplement the

tools where that was possible, and, perhaps most important, could interpret the results in the context of his own problems even though the characterizations of these results might not be wholly justified theoretically.

Our focus on personal files is a result of these considerations. We want to provide for a person working with textual data (who will be referred to hereafter as an "analyst") a set of techniques that will allow him to construct and manipulate his own files. Our intent is to design these techniques so that they have as much theoretical integrity as possible and so that eventually they will satisfy many analysts working on different kinds of problems. We hope that our text-processing system will provide a test-bed within which theoretical hypotheses can be tested and that through studies of the experience of a number of analysts we can arrive at procedures which have general value and validity.

THE TEXT-PROCESSING SYSTEM

OVERVIEW

The text-processing system is being implemented on an IBM 360/50 computer with IBM 2260 display consoles. It is programmed in TREET (Haines, 1969), a list processing language and system which operates under OS. In its present form there are three major components. The first, chronologically as well as in sophistication, is a linguistically based procedure called SAFARI. It allows summary statements about textual information content to be stored and retrieved in sentence form so that the syntactic relations among the lexical items are represented directly in the data base.

The second component of the text-processing system is a set of procedures for editing textual materials. In addition to inserting and deleting data, the analyst can select or annotate lines, paragraphs, or whole selections to create files and subfiles for temporary or long-term storage or for report generation.

The third component of the text-processing system is a set of procedures for text-searching. Two methods have been provided to find simple patterns of characters, strings, words, or phrases. In the first, intended for smaller corpuses, the text is scanned directly. The other makes use of an index of the words in a text and is much more efficient for larger files. In both methods, synonym sets, which can be set up or modified on-line, allow for more complex search specifications.

These three kinds of text-processing capabilities will be available to the analyst as he sits in front of his display console. They

constitute alternatives responsive to certain of his needs. We are interested in introducing additional options to provide a broader range of capabilities, but the present components are still being modified and refined to increase their efficiency and to provide for more effective interfaces between them. The brief descriptions that follow are intended only to indicate in somewhat more detail the operation of each component.

SAFARI--A LINGUISTICALLY BASED PROCEDURE

The first component of the text-processing system, SAFARI, was a direct outgrowth of our work with the MITRE Analysis Procedure. We adapted the Procedure so an analyst could use it to code statements about information from his text files or to process queries for searching in files consisting of analyzed statements. For input an analyst can scan through a text on a display scope and can select or prepare sentences which summarize items of interest. These sentences are analyzed syntactically and stored in the data base as tree structures. Questions about the data base, analyzed similarly, are matched against the stored structures. Relevant statements are retrieved and presented to the analyst, who also may recover the original text passages from which they were derived.

The SAFARI procedure was programmed initially for the IBM 7030 (Stretch) computer; that version has been described elsewhere (Walker, 1967). SAFARI is now implemented on the IBM 360/50 computer with IBM 2260 displays. A detailed discussion of the programs in this new version (except for the on-line interface) can be found in Norton (1968); portions of the present and subsequent descriptions are adapted from that report and from other project reports.

The procedure itself consists of four stages: (1) lexical categorization together with morphological analysis of each word of the input sentence; (2) context-free parsing of the resulting string of lexical categorizations; (3) application of transformational (reversal) rules to the set of surface structures produced, which, in addition to rejecting inappropriate parsings and deriving the proper base structures, also standardizes the resulting tree in a canonical format; (4) (a) for declarative sentences, storage of the canonical representations in the data base; (b) for interrogative sentences, searching in the data base for structures matching their own canonical representations.

The lexicon provides for a word or stem a list of its categorizations, each of which contains a category label and a feature-value pair. Words not found in the lexicon are processed by a morphological analysis procedure into stems and affixes. Analysis rules determine possible stem-affix structures. A structure is acceptable if the stem is in the lexicon and its combination with that affix is in accord with a set of morpheme-combinatorial rules. Redundancy rules are applied to the structures to assign by default feature-value pairs for certain otherwise unmarked structures (e.g., nouns not marked plural are singular, nouns marked "human plus" are also "animate plus"). (Additional information about the morphological analysis procedure and about its linguistic basis can be found in Chapin and Norton (1968) and in Chapin (1967).)

The parser is a bottom-to-top algorithm which produces for the string of (lists of) lexical categories (excluding the feature information) all of the possible surface structures according to a

particular set of context-free phrase structure rules. It is essentially the one in the MITRE Analysis Procedure, although the implementation is more efficient.

Before the transformational rules are applied to the set of surface trees for the sentence, the feature-value pairs provided by the lexical analysis are attached to the appropriate terminal nodes. The transformations operate substantially as in the MITRE Analysis Procedure to produce base trees, except that here feature anomalies also provide grounds for rejection, thus reducing the number of spurious ambiguities. In addition, special transformations have been added to convert the base structure into a canonical form in order to facilitate searching.

Declarative sentences, processed in the manner described, are stored in the data base as trees. Each tree is cataloged under those words (stems) that appear as terminal nodes under certain grammatical category labels, the choice of which can be specified (and easily changed) by the analyst. Subordinate clauses are stored so that they can be searched separately, but with a pointer to the full tree in which they appear. Questions, after processing, are matched against trees in the data base. The search is restricted to those trees which contain words appearing as terminal nodes under particular grammatical categories in the query sentence. Each node of the query is compared with the corresponding node of each tree in the recovered set. A comparison is successful if the two nodes are identical, if the node in the query is a question word, or if both nodes belong to the same equivalence class (class membership can be defined by the analyst). Feature agreement also can be required for nodes whose values are on a special list (also under the analyst's control).

Conjoined phrases are handled so that the order of occurrence within a sentence is immaterial. The question matches a stored statement if all of the nodes in the query compare appropriately with those in the tree for that statement. Consequently, it is possible for the trees in the data base to have additional structure not in the question. Since the particular matching algorithm applies recursively from left to right, nodes in the stored trees might have additional daughters or right sisters. Some of the formatting transformations mentioned above introduce optional nodes with null values to guarantee that all possible left sisters will be present in a tree.

The grammatical rules included in the current version of SAFARI are not of particular linguistic interest. Furthermore, although they were derived on the basis of a particular text corpus, they have not been used enough to establish their utility even for that data base. The grammar was written primarily for checking out the system, although it is worth investigating further to determine its practical value. It was written as a recognition grammar directly, rather than first defining a subset of sentences explicitly by generative rules and then establishing the corresponding set for the syntactic analysis procedure. The grammar allows statements and questions to be written, using simple relative clauses, adverbial and adjectival prepositional phrases, and a variety of conjoined constructions. The inclusion of a small number of syntactic features (inherent features on nouns, strict subcategorization and selectional features on verbs) has enabled us to assess their usefulness in the analysis procedure. By postponing the testing of context restrictions until the transformational rules have applied, a large amount of structural ambiguity is eliminated from the surface parsing.

The programs within the linguistically motivated parts of SAFARI (that is, the morphological analysis, context-free parsing, and transformational application) are designed so that the rules can be changed easily, allowing the procedure to be used for testing grammars (cf. Gross, 1967, 1968, and Gross and Walker, 1969 for related work on grammar testers using similar programming strategies). The transformational rules for establishing canonical trees for storage and retrieval can be changed as easily, allowing different formats to be tried.

THE EDITING OPERATIONS

In contrast to SAFARI, the editing procedures are more traditional. The major limitation on their flexibility results from the display hardware we are using. The techniques available with the SAFARI implementation on the IBM 7030 computer, which used DD-13 graphic displays under lightgun control, were much more elegant (cf. Gross, 1967). However, the actual editing operations were not substantially different.

The analyst can modify text by typing over material displayed, by specifying a segment on a line and its replacement, by inserting or deleting lines, and by moving lines from one file to another. In these ways he can make additions to a file, correct it, format the data, make annotations, insert index terms. Or he can create a new file directly, or out of pieces from other files. A file, modified or new, can be rearranged or reformatted so that it is suitable for report generation. It also is possible to create a file that records actions taken by the analyst and to which he can refer on-line.

TEXT-SEARCHING PROCEDURES

SAFARI allows an analyst to recover complex relations among information elements relating to texts he has previously processed. However, it is desirable to be able to identify new texts that might be worth analyzing in depth. In addition, often it is not necessary to provide such a deep analysis to discriminate relevant information for a given task; the occurrence of certain words or phrases may be a sufficient clue. Two procedures for searching text have been developed for these situations: one performing a direct character-by-character scan, the other using an index to locate potentially relevant sentences which then are scanned directly. Both procedures allow relatively complex patterns to be identified.

For the direct scan, the text is considered to be separated, by the occurrence of blanks, into pseudowords, which may include punctuation or such coding as capitalization indicators. The search request can consist of sequences of constituents occurring within specifiable distances of each other with or without the use of synonyms. In the standard search (without synonyms) a constituent can be a word--which will match the letter portions of pseuodwords, a string--which will match any specified sequence of characters, or a disjunction of constituents--one or none of which may be required to match. Exclusion lists can be established in this standard search, but, because of their inefficiency, their use is not encouraged. Note that the string constituent will allow capitalization codes, punctuation, and other non-alphanumeric elements to be included in a search request.

The synonym search adds to the standard direct scan two extra features. First, phrases can be used as single constituents. Second, and more important, synonym sets can be established so that for each constituent in a search request, all relevant synonyms are automatically included as disjunctions. These sets typically would be set up in advance, but they can be modified on-line.

In both types of direct scan, pointers to the locations of matching portions of text are accumulated. Upon completion the number of matches is presented to the analyst, and he can view them successively in context on the page (i.e., ten-line segment of text) in which they appear. If the data base has been set up so that different categories of text are identified explicitly (e.g., title, author, source, body), searches can be restricted to specified categories.

The index search retains most of the features of the direct scan searches but makes use of an index of the text to restrict the scope of the search to just those sentences which have the words contained in the search request. These sentences are then scanned directly for the requested pattern using the procedures described above. Since an index must be prepared, this kind of search is most appropriate for stable files. It is particularly appropriate for larger amounts of text or where a large number of synonyms are used. Words, phrases, and arbitrary strings can figure in the search request, but the index will be of value only for those strings whose left-most character corresponds to the beginning of a word (e.g., a stem without a prefix). Both standard and synonym searches can be made, substantially as in the direct scan approach, except that phrases can be used in the standard search as well.

In the index search, the actual sentences satisfying the search request are accumulated in a separate file together with information identifying the text in which they occur and the line number on which they begin. This answer file may be viewed, printed, or edited, and the text pages in which the answers appear may be retrieved.

DISCUSSION

The three components of the text-processing system described in the preceding section are intended to provide a range of capabilities for an analyst working on-line with textual data. However, it should be noted that none of the components is in production status at this time, although the editor is being used for routine correction of input materials and exploratory work using other editing features and the text-searching programs is about to begin. Therefore, we cannot say anything informative about user satisfaction, nor is it meaningful to give program parameters or timing statistics, since they are certain to change (hopefully for the better, particularly as regards timing for SAFARI). Considering the focus of this Conference, what does seem appropriate is some discussion of the relevance of this system to computational linguistics.

It is obvious that the SAFARI procedure falls under the scope of computational linguistics, no matter how narrowly defined. And it is possible to list the morphological and syntactical analyses of SAFARI separately so that the plural reference in the title of this paper ("Computational Linguistic Techniques...") is satisfied. However, although the necessity to be practical and sensitive to what analysts actually can use and need to use in processing text prompted our introduction and elaboration of text-editing and text-searching techniques, computational linguistic relevance is not wholly lacking. While some might argue that editing and searching are computational linguistic operations, we do take the word "linguistic" in that phrase quite seriously. Accordingly, we are trying to incorporate within these operations

strategies motivated by linguistic considerations. So far, this influence can be seen only in our work in text searching.

The text-searching procedures we have developed emphasize the identification of patterns. While we certainly want to be able to identify words and word co-occurrence groupings, we also want to recognize stems and affixes, on the one hand, and clause and phrase structures on the other. Thus, by specifying in the search request "computer" as a word with its synonyms, "center" as a stem to allow for plurals, a distance delimiter to allow three or four intervening words, a disjunction of "in" and "at", and a capitalization symbol we hoped to identify some (but certainly not all) of the computer centers mentioned in our data base that were identified with certain institutions or locations. The creation of synonym sets can be the product of a study of syntactic and semantic relations; it need not be ad hoc and arbitrary. In trying to understand the limitations on synonym substitution in search requests we found we were aided significantly by an evaluation of the linguistics of certain construction types. Similarly, we believe that linguistic insights may help to organize indexes so that they are most useful. We hope that it may prove possible to introduce increasing amounts of linguistic sophistication into search procedures in these ways.

The point being made here is that it is not necessary to resolve the question of whether text-searching procedures are computational linguistic techniques. Rather, we are interested in determining whether linguistic considerations can heighten the effectiveness of these less sophisticated but currently much more practical ways of handling

textual data. We hope that further work with our text-processing system will enable us to make this evaluation.

Acknowledgments

The work described in this paper is a product of the following people whose ideas, implementations, and descriptions have significantly influenced its form and content: Carter Browne, Stanley Cohen, Jeanne Fleming, Richard Glantz, Louis Gross, Ted Haines, and Lewis Norton.

REFERENCES

Becker, J. D. The modeling of simple analogic and inductive processes in a semantic memory system. In D. E. Walker and L. M. Norton (Eds.), Proceedings of the International Joint Conference on Artificial Intelligence, 1969. In press.

Bierwisch, M. Some semantic universals of German adjectivals. Foundations of Language, 1967, 3, 1-36.

Chapin, P. G. On the syntax of word-derivation in English. MTP-68, The MITRE Corporation, September 1967.

Chapin, P. G., and Norton, L. M. A procedure for morphological analysis. MTP-101, The MITRE Corporation, July 1968.

Chomsky, N. Syntactic structures. Mouton, The Hague, 1957.

Chomsky, N. Aspects of the theory of syntax. M.I.T. Press, Cambridge, Mass., 1965.

Fillmore, C. J. The case for case. In E. Bach and R. T. Harms (Eds.), Universals in linguistic theory. Holt, Rinehart and Winston, N. Y., 1968.

Green, C. C., and Raphael, B. The use of theorem-proving techniques in question-answering systems. Proceedings of 23rd ACM National Conference, 1968, 169-181.

Gross, L. N. On-line programming system: user's manual. MTP-59, The MITRE Corporation, 1967.

Gross, L. N. A computer program for testing grammars on-line. MTP-102, The MITRE Corporation, 1968.

Gross, L. N., and Walker, D. E. On-line computer aids for research in linguistics. In A. J. H. Morrell (Ed.), Information Processing 68. North-Holland, Amsterdam, 1969.

Haines, E. C. TREET, a list processing language and system. MTP-104, The MITRE Corporation, March 1969.

McCawley, J. D. The role of semantics in a grammar. In E. Bach and R. T. Harms, (Eds.), Universals in linguistic theory. Holt, Rinehart and Winston, N. Y., 1968.

Norton, L. M. The SAFARI text-processing system: IBM 360 programs. MTP-103, The MITRE Corporation, September 1968.

Simmons, R. F. Natural language question answering systems: 1969. TNN-87, University of Texas Computation Center, January 1969.

Walker, D. E. (Ed.) English preprocessor manual. SR-132, The MITRE Corporation, May 1965.

Walker, D. E. SAFARI, an on-line text-processing system. Proceedings of the American Documentation Institute, 1967, 4, 144-147.

Walker, D. E., Chapin, P. G., Geis, M. L., and Gross, L. N. Recent developments in the MITRE syntactic analysis procedure. MTP-11, The MITRE Corporation, June 1966.

Woods, W. A. Procedural semantics for a question-answering machine. AFIPS Conference Proceedings: Fall Joint Computer Conference, 1968, 33, 457-471.

Zwicky, A. M., Friedman, J., Hall, B. C., and Walker, D. E. The MITRE syntactic analysis procedure for transformational grammars. AFIPS Conference Proceedings: Fall Joint Computer Conference, 1965, 27, 317-326.