AUTOMATED LANGUAGE ANALYSIS

1971-1972

Report on research for the period

September 1, 1971 - August 31, 1972

Sally Yeates Sedelow, Principal Investigator

This research was supported by
the Office of Naval Research

through

Contract N00014-70-A-0357-0001
under Project No. NR348-005

Approved for public release; distribution unlimited

The University of Kansas
Departments of Computer Science and Linguistics
Lawrence, Kansas 66044

AUTOMATED LANGUAGE ANALYSIS

1971-1972

Report on research for the period

September 1, 1971 - August 31, 1972

Sally Yeates Sedelow, Principal Investigator

This research was supported by
the Office of Naval Research

through

Contract N00014-70-A-0357-0001
under Project No. NR348-005

Approved for public release; distribution unlimited

The University of Kansas
Departments of Computer Science and Linguistics
Lawrence, Kansas 66044

AUTOMATED LANGUAGE ANALYSIS

1971-1972

Report on research for the period

September 1, 1971 - August 31, 1972

Sally Y. Sedelow, Principal Investigator
Martin Dillon, Consultant
Walter Sedelow, Consultant
Juliet Shaffer, Consultant
Herbert Harris
Frank Joyce
Thomas Kosakowski
Sam Warfel

1 September 1972

Table of Contents

AUTOMATED LANGUAGE ANALYSIS

Sally Yeates Sedelow, Principal Investigator

ABSTRACT

A report on the emphases in the Project for Automated Language Analysis for the period September 1, 1971 - August 31, 1972. Efforts related to the VIA content-analysis package included (1) "final" proofing and correcting of the Roget's International Thesaurus tapes, both (2a) pragmatic and (2b) theoretical approaches to problems associated with the recognition of prefixes, (3) development of programs for an interactive version of the ring-structure VIA, and (4) implementation of a FORTRAN list-structure VIA for the Honeywell 635. Also, work continued on a statistical support package.

PREFACE

I would like to express my appreciation to the Information Systems Staff, Office of Naval Research, for their ready responses to requests for assistance. The Project is also indebted to the Resident ONR representative, Mr. Reuel Lipman, for his advice on contractual matters, and to the University of Kansas Office of Research Administration and Office of Research Accounting for financial administration and other assistance with the contract. Mr. Mark Katz, University of Kansas Computation Center, has been extremely cooperative and helpful to all of us associated with this research effort. The Departments of Computer Science and Linguistics have also provided support; we are especially indebted to the Department of Computer Science for computing assistance and for secretarial support -- with special thanks, for the latter, to Mrs. Adran Wagner.

I. Survey of Project on Automated Language Analysis

September 1, 1971 - August 31, 1972

A. Introduction

The Automated Language Analysis Project is based upon the assumption that the encoding of information in natural language is one of the most critically important human actions. Hence, we hold that research directed toward identifying encoding patterns in natural language is of major singificance for that rather extensive segment of the human environment which is verbal. But, ironically given its importance, it is a crucial aspect of human behavior that is still not very well understood.

In recent years, projects directed toward major language research applications areas--machine translation, information tetrieval, automatic abstracting, general-purpose question-answering--have to some extent given way either to very restricted subsets of some one of the above, or to other analogous "big" problems, or to more basic research on those linguistics problems which are proving to be insurmountable obstacles to solutions in the major applications areas. A rather extensive survey this past year of scholars and scientists engaged in language research in the United States (Sally Yeates Sedelow and Walter A. Sedelow, Jr., Language Research and The Computer, University of Kansas, 1972. (480 pp.)), indicated rather clearly that many individuals who had earlier been concerned with a research effort in one of the areas listed above now feel that research on the nature of language, itself--most especially upon semantics, the aspect of language concerned with meaning--is necessary before any major applications breakthroughs can be realized. The development of more powerful computer-based parsers, in itself a form of basic research, has also lead to a call for more work on semantics, as the limitations of even the best syntactically-focused parsers have become apparent.

Among the encoding patterns with which this Project has been concerned, semantic patterns have been of salient importance. The VIA (Verbally-Indexed Associations) programs are directed toward computer-based recognition of semantically-related words in any written or transcribed document or text. These programs have been used with considerable success for a range of texts and documents: U.S. Presidential State of the Union Messages, English literary texts, population research information retrieval, and, with adaptation, for French literary texts.

In their earliest form, the programs required a non-computer-aided search of thesauri, synonym dictionaries, etc. by the research scientist or scholar in order to draw up lists of words semantically-related to those in the text meeting occurrence frequency thresholds. The important aspect of that search for this present commentary is the type of thesaurus, etc., being consulted; a salient difference between the approach taken here and that in many other research projects, both past and present, has been the use of general-purpose thesauri and dictionaries. Many computer-based information retrieval projects depend upon thesauri which are specially-prepared lists of words for a given application area, *e.g.*, chemistry or physics. Other computer-based applications areas, such as question-answering or programs designed to provide communication between humans and robots also tend to rely upon listings of words restricted to a very limited discourse area. Very often, programs dealing with a limited vocabulary are said to be generalizable to much larger universes of discourse, but the labor implied by such generalization thus far has seemed to prevent actual achievement of generalization. By contrast our approach has been to see whether already available compendia of semantically-related words are (1) at all usable as they stand: (2) characterizable so that the results obtained from using them may be adequately interpreted: (3) modifiable, so that inherent biases can be eliminated and additions and deletions intelligently effected.

Another aspect of semantics with which this project has been concerned is prefixation, which will be discussed in the next section, as well as, in greater detail, later in this report.

Statistical approaches to the identification of encoding patterns in natural language have received considerable attention over a period of years, and the advent of the computer as a data gathering tool has provided additional impetus for such studies. The bibliography entitled "Selected Items from an Inventory of Measures of Language" in Language Research and the Computer (Sedelow and Sedelow, pp. 319-352) provides many references to both computer-aided and non-computer-aided statistical studies of natural-language documents and texts. Since the Automated Language Analysis Project is directed toward the development of methodology which will provide for as comprehensive a characterization as possible of patterns in natural language, statistical approaches are, of course, a valuable tool for the Project, and our efforts in that area also will be discussed in this report.

In summary, the Project is concerned not with *ad hoc* solutions to natural-language problems of a highly restricted nature but with the development of widely-applicable methodology which can be rigorously and empirically tested through the use of the computer.

B. Summary of Past Year's Work

Major sections of the research conducted under the auspices of this project for the past year form parts of M.A. and Ph.D. theses which are midway; thus, extended descriptions of this work will appear in future reports. Likewise, computer program listings and user documentation associated with this research will not be included in this report but will be provided with those extended descriptions in subsequent reports.

The VIA-related work this past year has included completion of proof-reading of the key-punched version of Roget's International Thesaurus, making the necessary corrections in the thesaurus, research on automatic recognition of strings functioning as prefixes, and research directed toward producing an interactive version of the ring-structure VIA.

The other major research thrust has included developing and testing statistical programs. This work is, in fact, also VIA-related because the programs use the results of VIA program runs.

Reference was made earlier to the use--for the VIA programs--of already available compendia of semantically-related words. Earlier project reports as well as other publications* have described comparative studies of Webster's Dictionary of Synonyms and two versions of Roget's Thesaurus. These studies entailed key-punching all words in analogous entries (e.g., 'dead') and then doing separate VIA runs on the same section of text for each input data set in order to determine the relative efficacy of the three reference works. The result of these studies was a decision to seek permission to put Roget's International Thesaurus into computer-accessible form. This permission was granted by the publisher, Thomas Y. Crowell Co., the thesaurus has been key-punched and it is now possible experimentally to use the thesaurus for VIA runs--thus eliminating the necessity of the non-computer-aided search of thesauri, etc., alluded to in the Introduction to this report.

At this stage, (for several reasons) the Thesaurus is used only experimentally. The most important reason is that it is desirable to

---

* Sally Yeates Sedelow, Stylistic Analysis, Third Annual Report, SDC Document TM 1908/300/00, March 1, 1967. DDC #AD 651-591.

____________, et al., Automated Language Analysis, Report on research for the period March 1, 1967, to February 29, 1968, University of North Carolina. DDC #AD 666-587.

____________, et al., Automated Language Analysis, Report on research for the period March 1, 1968, to February 28, 1969, University of North Carolina. DDC #AD 691-451.

____________, and Walter A. Sedelow, Jr., "Categories and Procedures for Content Analysis in the Humanities," The Analysis of Communication Content, ed. by Gerbner, et al., John Wiley & Sons, Inc., 1969, pp. 487-499.

____________, "Communicating with a Computer About Humanistic Research," Proceedings: Computer Applications to Problems in the Humanities, A Conversation in the Disciplines, ed. by Burelbach, State University College, Brockport, New York, 1969, pp. 34-56.

have extensive information about the content of the thesaurus, itself, before making statements about results produced through the use of the thesaurus. The collecting of information about the thesaurus has awaited the availability of a corrected version of the thesaurus. Proof-reading for errors was only recently completed and the corrections for the errors turned up in the proof-reading have just been made. But, as Herbert Harris' article later in this report makes clear, even though errors which either existed in the original printed version of the thesaurus or were introduced through key-punching have been corrected, there are inconsistencies in the thesaurus, itself, which would interfere with exhaustive processing by a computer program. Further, in order to gather initial statistics on the thesaurus, it is desirable to be able to deal, for example, with words and phrases which are linked in a variety of ways and formed in a wariety of ways through the single printed word, or. Hence, now that we have a corrected version of the Thesaurus, which of course we will save in that form, we need to do some editing in order to make the Thesaurus amenable to our processing. The programming for the editing is now in progress.

The second reason for using the Thesaurus only experimentally is that it can produce very rich (extensive) results and we need to decide what types of categorization and subcategorization are possible and desirable when running VIA aginst the Thesaurus. (It should be noted, in passing, that although the VIA programs once required the scholar or scientist to hand-produce the input thesaurus and that is no longer necessary because Roget's is now in computer-accessible form, it is still possible for anyone who wants to hand-produce the input thesaurus to do so.) One possibility is to let the VIA program use the Thesaurus to produce all the first-level links, from among which the scholar or scientist can then choose those which he wants to see developed down through as many as five levels. Should he want to choose on the basis of a frequency threshold (e.g., more than 50 occurrences of the given word or root group) or specification (e.g., only words or root groups with one occurrence) it would not be necessary for him to intervene "manually". In order to gain some sense of the relative increase in output provided by incremental use of Roget's for input, we ran Milton's Paradise Lost against all of Roget's using randomly selected content words in Paradise Lost as search keys, pairing each key word with every word in Roget's co-occurring in categories containing the search key. Then, if any word in Roget's was identical to or had the same root form as a word occurring in Paradise Lost, the word from Roget's was placed in a print file. Figure 1 shows a sample of the results when 5000 pairs (search key - category word), picked at random, are used as input. Figure 2 shows a sample (for the same search key, 'angel') when 5000 more pairs, picked at random, are added to the original 5000 as input. (In these examples, Frequency of Occurrence is provided only for words which appear in Paradise Lost, as indicated by ONLY EXACT MATCH both as a comment standing alone and as part of ONLY EXACT MATCH AND ROOT GROUP EQUIVALENT. ONLY ROOT GROUP EQUIVALENT means that the word printed out in Level 2 occurred only in Roget's; it did not occur in that form in Paradise Lost (hence the 0 in FREQUENCY OF OCCUR) but in some other form having the same root as the form printed out, e.g., the word Lovable does not occur in Paradise Lost but the word, Love, does, or as Figure 2 makes clear, the word, Seconder, does not occur, but Second does. MATCH COUNT specifies the number

Figure 1

FREQUENCY OF OCCUR	MATCH COUNT	PRIME WORD --LEVEL 1-	--LEVEL 2-	
46	277	ANGEL		BOTH EXACT MATCH AND ROOT GROUP EQUIVALENT
2	4449	--------------	PATRON	BOTH EXACT MATCH AND ROOT GROUP EQUIVALENT
2	4561	--------------	PILOT	ONLY EXACT MATCH
0	5422	--------------	SECONDER	ONLY ROOT GROUP EQUIVALENT
3	6111	--------------	SWEETS	BOTH EXACT MATCH AND ROOT GROUP EQUIVALENT

Figure 2

FREQUENCY OF OCCUR	MATCH COUNT	PRIME WORD --LEVEL L-	--LEVEL 2-	
46	277	ANGEL		BOTH EXACT MATCH AND ROOT GROUP EQUIVALENT
1	358	--------------	ARCHANGELIC	BOTH EXACT MATCH AND ROOT GROUP EQUIVALENT
0	539	--------------	BACKER	ONLY ROOT GROUP EQUIVALENT
2	1496	--------------	DARLING	ONLY EXACT MATCH
0	1517	--------------	DEARY	ONLY ROOT GROUP EQUIVALENT
19	3046	--------------	HOLY	BOTH EXACT MATCH AND ROOT GROUP EQUIVALENT
0	3559	--------------	KISSABLE	ONLY ROOT GROUP EQUIVALENT
2	3583	--------------	LAMB	BOTH EXACT MATCH AND ROOT GROUP EQUIVALENT
0	3750	--------------	LOVABLE	ONLY ROOT GROUP EQUIVALENT
2	4449	--------------	PATRON	BOTH EXACT MATCH AND ROOT GROUP EQUIVALENT
2	4561	--------------	PILOT	ONLY EXACT MATCH
6	5219	--------------	RIGHTEOUS	ONLY EXACT MATCH
0	5318	--------------	SAINTLIKE	ONLY ROOT GROUP EQUIVALENT
26	5422	--------------	SECOND	BOTH EXACT MATCH AND ROOT GROUP EQUIVALENT
0	5422	--------------	SECONDER	ONLY ROOT GROUP EQUIVALENT
3	111	--------------	SWEETS	BOTH EXACT MATCH AND ROOT GROUP EQUIVALENT

assigned to a given group of words having the same root; for example, ANGEL and any words sharing that root have been assigned MATCH COUNT number 277--these numbers being assigned for programming convenience. As is obvious, doubling the number of input pairs considerably increases the output; and, of course, it should be remembered that these pairs were picked at random--so the quantity of output could have been much greater or, to be sure, much less. Extensive experimentation has been difficult because the programs which produced these runs are at the University of North Carolina and the Principal Investigator is at the University of Kansas; but a FORTRAN, Honeywell 635 version of the list-structure VIA programs is now available at Kansas so that further work on the use of Roget's can proceed.

Research on prefixation is relevant to the VIA programs because the "first pass" at grouping together words which are semantically related consists of pulling together words having the same root. Computer programs which deal with the morphology of suffixes can be used with some confidence both because strings functioning as suffixes can be relatively reliably identified and because suffixes, although carrying semantic information, do not ordinarily alter the meaning of the root to the extent prefixes do (e.g. please-d or pleas-ing in contrast to dis-please). Nonetheless, we have added a Prefix module to the VIA programs because it is often desirable to group a prefixed word with other words having the same root. Earlier reports* have described the

---

* Sally Y. Sedelow, et al., Automated Language Analysis, Report on research for the period March 1, 1968, to February 28, 1969, University of North Carolina. DDC #AD 691-451.

__________, Automated Analysis of Language Style and Structure in Technical and Other Documents, Technical Report No. 1, University of Kansas, September, 1971. DDC #735-134.

results of using the Prefix program and the problems raised by the effort to decide when a given string is functioning as a prefix. Our efforts to deal with these problems have taken two tacks: pragmatic and theoretical.

The pragmatic approach to prefixing is described in two articles in this report by Sam Warfel. The core of this approach was to see whether, for those cases where a string of characters might or might not be functioning as a prefix, the semantic groupings provided by Roget's as it stands and as it might be modified would provide a resolution for the problem. In other words, if a given word without a prefix and a form suspected of being a prefixed version of the same root occur in the same category or in intersecting categories, then would it be safe to assume that the two words do, indeed, share a root? The results of this investigation are described in Warfel's first article. The second paper suggests an approach, as yet untested, which might be used as a follow-on for those cases which are not resloved by the search in Roget's.

The theoretical approach to prefixing is also being pursued by Warfel**, but no write-up of that research is included in this report because it is part of Warfel's doctoral dissertation, now in progress. Publication of those results will thus come later.

With reference to the Prefix module, it also should be noted that the computer program, itself, has been modified. Frank Joyce has provided a description of the program additions to PREFIX and we do in-

---

** Sally Y. Sedelow, et al., "Toward a Theory of Prefixing," Automated Analysis of Language Style and Structure in Technical and Other Documents, Technical Report No. 1, University of Kansas, September, 1971. DDC #735-134, pp. 76-96.

clude a listing of the complete PREFIX module in its FORTRAN version for the Honeywell 635 (PREFIX was not included in the listings of the FORTRAN version of VIA programs, given in last year's report).

A ring-structure VIA, in contrast to the list-structure version discussed earlier and examplified in Figures 1 and 2, was prompted by our desire to provide an alternative to the hierarchiacal format implied by linked lists or a uni-directional tree, as well as to enable searches to be keyed by conceptually-related groups of words in addition to the single words and root groups used for the list-structure version. A batch version of a ring-structure VIA* was designed and implemented but it quickly became apparent that both the versatility and quantity of output offered by the ring-structure approach was better suited to an interactive mode than to batch. That is, the interactive mode would enable the user to experiment with the variety of options made available, so as to choose the option holding most promise for a given text or document and it would also enable him to "prune" the output as he watched it being displayed. Tom Kosakowski is working on programs for the implementation of the interactive version; again, because this work forms a master's degree project which is in medias res, it will not be described in this report but, instead, will be reported on later.

Another master's degree thesis which will be emerging from this

---

* William Buttelmann, "Ring-Structure Version of VIA" in Sally Yeates Sedelow, et al., Automated Language Analysis, Report on Research for the period March 1, 1967, to February 29, 1968, University of North Carolina, DDC #AD 666-587, pp. 19-27; 85-106.

__________, "Ring-Structure Version of BIA," in Sally Yeates Sedelow, et al., Automated Language Analysis, Report on research for the period March 1, 1968, to February 28, 1969, University of North Carolina, DDC #AD 691-451, pp. 40-48; 54-81, 125-198.

project will be an account of Frank Joyce's work on a statistical support for automated language analysis. As Joyce points out in this current report and in his article in last year's report*, the "normality" of much language data--except for very large samples, e.g., a million words--is, at best, open to question. "Independence" is another assumption in many statistical models which bedevils efforts to use statistics for pattern recognition in natural language. This report contains a short outline of some of Joyce's current work which will be described in detail in his forthcoming thesis.

Finally, reference should be made to a monograph footnoted earlier which is related to this project in the sense that some of the research mentioned in the monograph as well as some of the general ideas expressed there are the direct result of work done as part of this project. The monograph, Language Research and the Computer, was prepared to describe the results of a study--funded by the National Science Foundation--which looked into the possible desirability of a national center or network for computer-based language research. Some of the views of the principal investigators for that project which are stated in the monograph derive from work on this Automated Language Analysis project, and have been adumbrated in reports and papers describing research undertaken as part of this project.

C. Plans for the Future

The first order of business for the work on Roget's International

---

* Frank Joyce, "Statistical Analysis of Linguistic Frequency Data," in Sally Yeates Sedelow, et al., Automated Analysis of Language Style and Sturcture in Technical and Other Documents, University of Kansas, September, 1971, DDC #735-134, pp. 66-75.

Thesaurus will be the collection of statistics concerning the content of the thesaurus. In order to collect those statistics, the thesaurus must be edited so as to provide the internal consistency which is currently lacking (cf. earlier comments, as well as the article later in this report by Herbert Harris). In order to handle the thesaurus more readily from a programming point of view, as well as to study some of its characteristics, it must be indexed. We elected not to keypunch the index in the printed version on the assumption that we could produce an index according to a computer-implemented algorithm which would be more rigorous and hence more consistent and manageable by computer. Therefore, once the editing has been completed, the thesaurus will be indexed. Then, the task of collecting statistics will begin. We would also like to make simple comparisons of the words appearing in Roget's with those occurring in other well-known and frequently used word lists e.g. the Thorndike-Lorge List, the Brown Million Word Corpus of 1961 American English, the Webster's Collegiate Dictionary, and any others to which we can gain access; special-purpose thesauri for information retrieval applications in given areas would be of interest to us in order to see how much overlap there is between such special-purpose lists and a general-purpose thesaurus. We are also concerned with the internal structure of the thesaurus, e.g. types of connectivity and their extent.

The pragmatic approach to prefixing can be tested more thoroughly through computer programs run against the Thesaurus (cf. articles in this report by Harris and Warfel), and we would hope to make a start on that at some point during this next year. As noted earlier, aspects of the theory of prefixing will form Sam Warfel's forthcoming doctoral dissertation.

Work on the statistical support system will continue, with Frank Joyce's contribution to be summed up in his master's thesis.

It also should be possible this year to experiment with the use of Roget's for the list-structure version of VIA as implemented in FORTRAN on the Honeywell 635; we hope that experimentation with the interactive ring-structure version also can be initiated.

II. Further Editing of Roget's Thesaurus Tape and Some Observations on Further Studies of the Thesaurus

by Herbert Harris

The tape of Roget's Thesaurus has been proofread and corrected. It now consists of about 77,500 card image records. The attempt has been to make the tape correspond as closely as possible to the printed text of Roget's International Thesaurus (Thomas Y. Crowell, Third Edition, 1962). Each card image represents one line of one column of the text. The text consists of 657 double column pages. The index is not included on the tape. Although the intent was to make the text of Roget's Thesaurus available in machine readable form, certain compromises have been made between representing the external form of the Thesaurus and representing the concepts that certain changes of type face indicate.

The cross-reference numbers appear in two different type faces. In the paragraphs they are a small boldface and in lists of words they are in the type face of the list. This change appears to be an aesthetic device; therefore all cross-reference numbers are treated alike, that is they are preceded by a single forward slash (\). This consistency gained in representing the concept of cross-reference would seem to make the tape easier to use. The difference in type face is recoverable since lists are indicated and cross-references can be identified as occurring within the context of a list. The paragraph numbers have not been marked as boldface although in the text they do appear in boldface. Boldface is also used in the text to indicate the most commonly used words, and its use for the paragraph numbers appears to be an aesthetic device again. Boldface is also used for the headings of lists; these headings have been indicated on the tape as being in boldface. They can be distinguished from the conceptual uses of boldface by the fact that they are always followed in the next record by the change-of-format operator for lists, ****LIST.

Lists also present several other problems. The list operator serves to identify the members of a complete listing. But when a list cuts across two pages the page number appears as a part of the list. The page number can be recognized in a list by its leading zero and the fact that there will be a blank in column forty of the card image. One column of the list begins in column one of the card and the second begins in column forty. Four spaces have been used to indicate page numbers with the letter a or b indicating the column on the page, e.g., 0042a for page 42, column a. Because there are 657 pages in the text section of the Thesaurus, there will always be a leading zero.

The format of the lists in the text has been maintained on the tape. A list in the text appears as two columns alphabetically sequenced from top to bottom of column one and then from top to bottom of column two. In case such a listing is broken by a page number the above format is kept on one page and the sequence is restarted on the second page.

All changes in type face are indicated by special characters. The pound sign (#) is the downshift from all previous operators. For example there are cases of double operators, e.g., "@$double-entendre#" (an entry which is in boldface italics); the pound sign indicates a downshift from both the @ and the $ signs. The following are the character equivalents for typeface changes:

@ ----------------- boldface
$ ----------------- italics

\ --------------- cross-reference numbers
\\ --------------- part of speech headings
* --------------- bounce shift to capital

The following parts of speech headings are used in the Thesaurus:

1. nouns
2. verbs, verb
3. adjs.
4. advs.
5. prons.
6. preps.
7. phrs., phr.
8. interjs.
9. interrogs.
10. conjs.
11. proverbs

In addition to those listed above, these parts of speech may also be followed by *etc.* in italics. They may also occur in combinations separated by a comma.

Diacritics are indicated by special characters that precede the letter marked by the diacritic. The key is:

← --------------- én --------------- ←en

+ --------------- èn --------------- +en

= --------------- ên --------------- =en

↑ --------------- ën --------------- ↑en

< --------------- façade ----------- fa<cade

/ --------------- mañana ----------- ma/nana

In addition the following special symbols have been used:

&& --------------- £ (pound sign)
$$ --------------- $
?? --------------- ¿ ¿Que?
!! --------------- ¡ ¡Que!

The following were judged to be errors in the text and were changed on the tape:

1. 277.56 @flying,# flitting (The comma was added after *flying*.)
2. 279.13 [slang, Eng.] (The space was added after the comma.)
3. 334.13 @phosphorescence,# (This word is in a different type of boldface print in the book. It is regular boldface on the tape.)
4. 356.9 @ductile,# facile (The comma was added after ductile.)

5. 401.12 Verbs.# (The period was deleted on the tape. This is not an abbreviation.)
6. 535.27 (F..) (The second period was deleted on the tape.)
7. 538.7 Come upon (un)expectedly or without warning (The un was added on the tape.)
8. 205.9\273.10# (This should be 276.31. The cross-reference number is incorrect.)
9. 103.18 Ints.# (This should be Interjs. This appears to be the only occurrence of this part of speech designation.)

The spacing for U.S. was kept as in the book and no space was left between characters forming acronyms.

The use of or in the text entries presents considerable problems for anyone using a computer to process the Thesaurus. Or is used for several purposes and the usage is not algorithmic. A few examples will illustrate the problems: Or is used to indicate a difference in spelling of some word, "beadsman $or# bedesman" (1036.16). Sometimes it is used to indicate a morphological variant of some word with the same meaning, "sound $or# resound the praises of" (966.12). Most of the time it is used to indicate a possible substitution of some word in a phrase maintaining the same meaning, "clap $or# pat on the back" (966.11). Often in a phrase the rule of thumb is that the word prior to the or is deleted and the word immediately following the or is inserted in its place. In "put in a word $or# good word for" (966.11) this substitution yields "put in a good word for". But what will be the other entry, for which the or does not operate and there is no substitution? Since only "word $or# good" affects the substitution routine, what follows good will appear to be a part of the matrix phrase. Word will be substituted for good yielding "put in a word word for", an incorrect result. Without recognizing the recurrence of word after the or or what is worse the fact that the head noun is both modified and unmodified, this exception will not be caught.

When there are three or more items to be substituted in a phrase the or is not repeated, but a tilde is used in its place. On the tape a right pointed bracket (>) is used in place of the tilde. The substituted item may either come at the beginning or the end of the entry, "wound>, sting $or# cut to the quick" (864.14) and "stick in one's crop,> craw $or# gizzard" (864.13). The substitution of one word for another as in the case of the or by itself will have to be expanded in these cases to "substitute everything in the vacant slot that occurs between the tildes or between a tilde and an or"; there are cases where more than one word must be substituted for one word, e.g., "muster>, summon up $or# gather courage" (891.13). Also a word cannot be defined just by a space since some of the substitutions are made after a hyphen, "nerve-racking $or#> wracking," (857.15) and "self-complacence $or#> complacency" (866.2). This latter use of the tilde also differs from other uses since it does not involve a disjunction of three or more items.

But there are cases where the expanded substitution procedure will not work, e.g., "buy>, sell $or# deal in futures" (831.23). In this case substituting for only one word will yield "buy in futures", "sell in futures", and "deal in futures", where it should be "buy futures", "sell futures", and "deal in futures".

There are also uses of the tilde when the or does not explicitly appear, e.g., "to his executors,> administrators and assigns" (816.27).

The following is a case which uses two or's with a tilde: "lower 〉, haul down $or# strike one's flag $or# colors: (763.8). Most of the or's are in italics, but there are a few cases where the surrounding words are in italics and the or is in plain type face. Because the Roget's tape has been made to follow the text as closely as possible these variant uses of the or face anyone who wishes to use the tape without pre-editing. I have discussed them to alert potential users.

The next step toward the use and analysis of Roget's Thesaurus will be to make a comprehensive index of it. This project presents several problems. One problem is of course to redo the entries with or in order to separate out the multiple entries for which the use of the or is shorthand. Another problem is to decide what to consider an entry in a larger sense. Some entries are separated by semicolons and others by commas. But there are examples where a phrase has an internal comma and the surrounding entries are separated by commas. Obviously in such cases if the comma is used as an entry indicator, the phrase will be broken into two entries. Probably the index will key on both commas and semicolons. There are cases where an explanatory note will appear at the beginning of a paragraph in parentheses. Without this explanatory note, in many cases some of the entries in that paragraph will not be taken in the correct sense, e.g. "§50.15 (be angry)...storm." Probably these explanatory notes can be dispensed with in an index since the locations of the entries will be kept. The attempt will be to keep all information associated with each entry in the index. All entries with the same graphemic sequence should be under the same head in the index, but the locations should show whether a given use is deemed a common one, as indicated by boldface, whether it is colloquial, etc. This procedure will mean associating the square-bracketed information following an entry with the location number and not with the entry itself. There are many square brackets, and they are used for a range of types of information. For example, the translation of some foreign phrases appears in brackets: "$ex cathedra# [L., from the chair, with authority]"(737.17). Sometimes brackets contain an indication of the language or country where the word is used; sometimes they name the author of a quotation. Sometimes there is an assumption that the place of usage is in the U.S. and the level of usage, e.g., slang, colloquial, etc., is indicated. There needs to be a key to the information in brackets. In effect I am proposing that the bracketed information about an entry is a function of the semantic field in which it occurs. Associating bracketed and typeface information with the location of a graphemic sequence in the Thesaurus will assume this hypothesis, i.e., that this type of information is a function of the semantic field in which the word is used.

Some bracketed information is entered only once but must be distributed to a number of items. A group of entries all separated by commas may be followed by a square bracket, the first word in the bracket being <u>all</u>, e.g., ". . .; decagram, decigram, decaliter, deciliter, decare, decameter, decimeter, decastere[all metric meas.];" (99.6). In all cases observed so far the <u>all</u> information can be distributed to the previous semicolon. There are also occurrences of <u>both</u> in brackets. The following adds zest to my life but I am not sure how to handle it: "put pep,> zip, etc, into [all slang]" (160.9).

Given such an index, there are some interesting things to explore.

Suppose that the arrangement of concepts in the Thesaurus is viewed as linear and that the concepts in categories one and two are more closely related than those in one and one thousand. (This is not presently the case, but it can be made more so by rearranging categories to make adjacent those which have the closest semantic relationship; the present numbers could also be retained, so as not to lose the original structure.) We can now compare the locations of an entry directly with this linear listing of the categories. Does the usage of the entry cluster in some area? Warfel (this volume) has pointed out that there are words that do seem to cluster in two areas. Is this perhaps a criterion for having two entries in a dictionary? Perhaps the entries will cluster except for one or two occurrences but bracketed information will then give some clue as to the reason for these anomalous occurrences. Perhaps the clustering of locations can be used to build a scale of relatedness of the concepts in the Thesaurus. With some data in hand, other possibilities will no doubt suggest themselves.

Second, suppose we take the thousand entries from the Thesaurus with the most locations. These might be compared to the Thorndike-Lorge list to see if the most commonly used words in writing are the ones with the widest semantic distribution.

Aside from constructing an index for the Thesaurus, some investigations of the Thesaurus as a whole will be undertaken in the coming year. The Thesaurus is provided with a hierarchical structure in the Synopsis of Categories section. One question that can be investigated is the degree of overlap between categories on each level of the hierarchy. Another exploration will concern the degree of connectedness in the Thesaurus: Will one word lead to the whole Thesaurus if the paths of its associated words are all followed? If not the whole Thesaurus, how many categories can be accessed through a given word? Are all words the same with respect to this property or do some have greater path potential than others?

Roget's Thesaurus will also provide a way to investigate the semantics of language as general phenomena insofar as the Thesaurus reflects the judgements of some native speakers about the relatedness of words in English. The classification of words in the Thesaurus seems to be based upon a range of criteria. To get some feel for these differences take the following four paragraphs from Roget's: First, 999.5 is a list of United States Federal Courts; second, 864.1 are words which in the abstract express displeasure. Succeeding paragraphs in 864 list displeasures grouped by intensities, whether the irritant is referred to as such, e.g., aggravation, pest, etc. (864.2), whether the irritant has a cause, e.g., irritation, provacation, chafe, etc. (864.3); third, 28.8 is a list of quantities, e.g., armful, bag(ful), etc; fourth, 48.8 is a list of knots, e.g., anchor knot, becket knot, etc. In each of the cases the semantic concept has a different relationship to the words in the group, but the word groups are for the most part appropriate.

The list of United States Courts results from the intersection of two criteria, one conceptual and one geographical, i.e., the concept of a court of law (999 is labeled Tribunal) and the geographical concept of location in the United States. It is not a list of close synonyms. It is a list of names. An analogous case is a list of geometric figures grouped together because they are geometric figures. The words expressing displeasure are grouped together as synonyms or near synonyms of each other. The list of quantities does not consist of synonyms, but the words do have in common the fact that they are quantities. The list of knots is merely a list of names. This list might be further subclassified by occupation, e.g., sailor, cowboy, hangman, etc.

Now a range of information type is useful. For example button and zipper both occur in 48.7 fasteners. With this information and knowledge of the fact that lips don't have them, we can interpret zip and button to mean fasten in "zip up your lip" or "button up your lip". The fact that lips don't have buttons or zippers prevents a literal interpretation of "zip up your lip" but this information is real world information. The list of federal courts of the United States is also information (in this case, encyclopedic) about the real world. But to construct encyclopedic lists based upon the zip and button examples would require negatively based clusters, e.g., all things that don't . . .; such clusters would entail impossibly long lists.

But the fact is that zip and button used as verbs are near synonyms of fasten at a higher level of abstraction. To zip is to fasten by means of a zipper. This example hints that there may be ways to account for Thesaurus classification through a complex definition of the word.

The classification of quantities may be related to the zip and button case since it seems possible to add -ful to any concrete noun and have a quantity. The adding of the suffix in this case is an overt concept addition, whereas the presence of fasten was covert in the zip and button example.

If I speak of "my displeasure", I am speaking of the feeling I have. And if I then in the course of the conversation refer to "the annoyance", I refer to the external cause of "my displeasure". If I speak of "my annoyance", I may be referring to my feeling, but I do not often speak of "the displeasure" when referring to the external irritant. The Thesaurus puts these two words in different paragraphs but groups them under 864 Displeasure. These judgements seem appropriate, even though the reason for their appropriateness is only adumbrated.

The task of seeing the relationship of the Thesaurus to language as a whole is to try to make explicit the criteria for grouping words in the Thesaurus, to say what constitutes evidence for the satisfaction of the criteria, and to give a principled system that leads to the judgements in the Thesaurus (that is once a word is defined we could predict where a native speaker would classify it). These are large questions that need study, which, in turn, will provide information about the semantics of language.

Also it is possible that there is an interaction between what a thesaurus is used for and what should be included in it. Such applications as information retrieval, machine translation, and text analysis may emphasize different criteria of classification. For example, information retrieval seems to utilize the hierarchical arrangement of abstract categories more than some forms of text analysis, where more abstract levels may be used little if at all.

Thesauri in special fields are and must be more specific than a general purpose thesaurus. The construction of special area thesauri also involves the interaction between model assumptions and classification schemes. Hopefully the investigation of the general purpose thesaurus as represented by Roget's will suggest useful ways to classify special purpose thesauri.

Roget's Thesaurus can provide statistical and distributional information about word meanings. As an embodiment of native speaker's judgements about their language, it can be an entré to the task of making explicit the basis for judgements of semantic relatedness. And with the relationships of semantic classification in hand we may be able to see the relationship of special purpose thesauri to each other.

III. A. The Value of a Thesaurus for Prefix Identification

by Sam Warfel

Part of the package of programs which is being assembled for automated language processing by this project's research group is a program called PREFIX whose basic function is to identify word tokens which are related semantically and morphologically by discovering common roots with different prefixes. At present this is accomplished by a process which matches each token with a list of prefix forms which have been identified by linguists and with a list of words which may be either inclusions or exclusions from a determination of prefixation. The inclusions and exclusions (CLUD) list was compiled by hand, relying on native speaker intuition concerning whether a word is prefixed or not (see previous research reports for a description of the program). It has been one of the aims of this research project to make more explicit the criteria used in this kind of decision.

In last year's report I argued that four types of information are necessary to determine whether or not a word is prefixed. "1) the meaning of the prefix, that is, the possible meanings of a particular prefix form, 2) the meaning of the word which remains when the prefix form is removed, 3) the rules for determining the semantic relationship of this particular prefix to the roots to which it is attachable, and 4) the meaning of the entire word" (Warfel, 1971). The types of information in 1) and 3) and their interaction appear to be necessary to any analysis which deals with the nuances of semantic interpretation which speakers of English can distinguish in prefixed words. However, it is possible that a simple determination of whether or not a word is prefixed could be made using the types of information in 2) and 4) given some means of measuring semantic relatedness. In this report I will present the results of an investigation into the use of Roget's International Thesaurus as the basis for such a semantic relatedness metric.

The first section of this paper indicates how a thesaurus functions as a semantic relatedness tester and discusses a number of weaknesses in the approach. The second section presents the results of a hand simulation of a proposed program to use the Thesaurus for prefix analysis. The third section presents the algorithm which can be implemented on the computer to further test the basic assumptions of this investigation.

ROGET'S THESAURUS AS A SEMANTIC RELATEDNESS TESTER

Before discussing the use of thesauri for prefix analysis, it is necessary to say something about the organization of Roget's International Thesaurus (hereafter referred to as 'the Thesaurus') and its use as a tool for measuring semantic relatedness. The basic text of the Thesaurus (which this research project now has in machine readable form) consists of 1040 semantic categories each with a number and a label, e.g., 854. Lack of Feelings. Each of these numbered categories is divided syntactically and semantically. The semantic sub-categories are numbered consecutively following the decimal point through the category but are not labeled. The syntactic labels occur with the numbered sub-categories and indicate that all following sub-categories belong to that syntactic category until another syntactic label is given in the sequence. The sub-categories consist of the words which are considered semantically related along several dimensions which I am not prepared to discuss here. The words in these sub-categories are further divided into semicolon delimited sub-sub-categories which I will discuss further below.

The 1040 categories are related at a higher level in the Thesaurus by the "Synopsis of Categories" which is not part of the basic text, but is presented as an outline following the Preface. In this synopsis the Thesaurus is divided into eight classes. Each class is divided into several labeled sub-classes indicated by Roman numerals and each sub-class is divided into labeled sub-sub-classes designated by capital letters. Each sub-sub-class is divided into several of the 1040 categories which are numbered consecutively throughout the text.

In discussing possible models of thesauri Martin Dillon and David J. Wagner (1970) indicate that the Thesaurus can be considered to have a hierarchical structure with six levels based on the formal structure presented in the Thesaurus itself. The following is the example which they give for the word perfect:

Class Six: Intellect
- I. Intellectual Facilities and Properties
  - L. Conformity to Fact
    - 515. Truth
      - Adjectives
        - 515.14
          - (perfect)

This says, in effect, that one occurrence of the word perfect in the Thesaurus is to be found in sub-category 515.14 which is one of several sub-categories of adjectives under category 515. Truth. Category 515. Truth is found in the "Synopsis of Categories" under the letter L. Conformity to Fact which is in turn a part of Roman numeral I. Intellectual Facilities and Properties which is a division of Class Six: Intellect.

I would like to suggest that for the purposes of a semantic relatedness metric two changes be made in the hierarchy Dillon and Wagner give. In the first place, a close examination of their sixth level in the text indicates that, in most categories, words which are most closely related are grouped and delimited by semicolons. For example the list of words in sub-category 515.3 is punctuated as follows:

> accuracy, correctness, rightness;
> exactness, exactitude;
> preciseness, precision;
> mathematical precision, pinpoint precision, scientific exactness;
> faultlessness, perfection;
> definiteness, positiveness, absoluteness;
> faithfulness, fidelity;
> strictness, severity, rigidity, rigorousness, rigor;
> niceness, nicety, delicacy, subtlety, fineness, refinement;
> meticulousness 531.3.

The words grouped between semicolons (each line as presented here) appear intuitively to be more closely related to each other than to words in the other groups. Therefore, a refinement of the hierarchy to include this grouping as the lowest level should improve the use of the hierarchy as a semantic relatedness metric.

The second change I propose is that the syntactic categories be ignored for the purpose of measuring semantic relatedness. Whether this decision is correct or not is an empirical question which can be tested given an operating system to do so. However, it appears from the hand simulation done of the program described in the following sections of this paper, that the syntactic categories add nothing to the determination of prefixed words. Since syntactic category membership is most often indicated by suffixes, the relationship between <u>adjust</u> and <u>readjust</u> is as easily found as that between <u>adjustment</u> and <u>readjustment</u>. The relationship between syntactic and semantic categories is more complex than what I have indicated, but a better understanding of the interaction will have to wait on further study of the Thesaurus.

In keeping with the two changes proposed above I suggest that for the purpose of determining semantic relatedness, a hierarchy with the following six levels be recognized: (I will refer to these six levels in the discussion below.)

Class Six: Intellect
  I. Intellectual Facilities and Properties
    L. Conformity to Fact
      515. Truth
        515.3
          Accuracy, correctness, rightness;
            (correctness)

The basic assumption concerning the usefulness of such a hierarchy for a semantic relatedness metric is that words which occur in the same category at any level are more closely related to each other than to words outside that category, e.g., a word which occurs in 515.3 will be more closely related to a word in 515.4 than to a word in 517.2. As will be demonstrated below this assumption holds in a large number of cases with intuitively satisfying results.

It appears, however, that there are categories which are closely related on the fourth level which are not indicated as related in the formalism of the Thesaurus. Most of these are categories which are related to each other by opposition or negation. For example look at the fourth level categories in the following hierarchy:

Class One: Abstract Relations
  VI. Time
    E. Recurrent Time
      135. Frequency
      136. Infrequency
      137. Regularity of Recurrence
      138. Irregularity of Recurrence

In this example it is obvious that categories 135 and 136 are more closely related to each other than to either category 137 or 138, and that the latter pair are similarily related. The actual relationships might better be represented with an additional level as follows:

Class One: Abstract Relations
VI. Time
E. Recurrent Time
1)
135. Frequency
136. Infrequency
2)
137. Regularity of Recurrence
138. Irregularity of Recurrence

Another example with the extra level included is the following:

Class Six: Intellect
I. Intellectual Faculties and Processes
K. Qualifications
1)
506. Qualification
507. No Qualifications
2)
508. Possibility
509. Impossibility
3)
510. Probability
511. Improbability
4)
512. Certainty
513. Uncertainty
5)
514. Gamble

For use in prefix determination and especially in negative prefix determination it would be desirable to be able to relate the four pairs given above.

In the examples given thus far relatedness can be correlated to some degree with adjacency of categories, i.e., two words have a greater probability of being related if they occur in adjacent fourth level categories. However, the following example shows that non-adjacent categories can also be profitably considered closely related:

Class Six: Intellect
I. Intellectual Faculties and Processes
L. Conformity to Fact
515. Truth
516. Maxim
517. Error
518. Illusion
519. Disillusionment

For the purpose of prefix analysis it would be helpful to relate non-adjacent categories 515 and 517. In fact a comparison of 515.3 and 517.2 shows the relatedness of the words in the two categories, the latter set being the negation of the first.

515.3	517.2
accuracy, correctness, rightness; exactness, exactitude; preciseness, precision;	inaccuracy, inaccurateness, uncorrectness, inexactness, inexactitude, unpreciseness;
. . .	. . .

Notice in 517.2 that the negation is carried by a prefix in each case.

There are also triplets of categories which could be profitably considered closely related for the purpose of prefix analysis. Look at the following fourth level categories:

Class Six: Intellect
- I. Intellectual Faculties and Processes
  - G. Conclusion
    - 493. Judgment
    - 494. Prejudgment
    - 495. Misjudgment
    - 496. Overestimation
    - 497. Underestimation

The first three categories should be considered closely related for any words related to 'judgment' which have pre- or negative prefixes.

In this study an equivalence table of related categories (rather than an added level in the hierarchy) will be used to supplement the formal hierarchy in order to capture the relationships mentioned above. The algorithm given in the last section of this paper will assume the extablishment of such a table.

Given the Thesaurus with its hierarchy of categories as modified by the suggested table of related fourth level categories, it is possible to define degrees of semantic relatedness in terms of the number of nodes separating words in the modified hierarchy. At this point the metric is effective only in a gross way and will remain so until a better understanding of the relatedness of the categories is attained, i.e., until the higher categories are refined by a greater elaboration of the hierarchy. Now that the project group has the Thesaurus in machine readable form, a number of studies are planned which should improve upon the semantic relatedness metric suggested here (see Harris in this report).

PREFIX ANALYSIS USING THE THESAURUS

Based on the definition of prefixes implicit in the four types of information for prefix determination given in the introduction, ("1) the meaning of the prefix, ... 2) the meaning of the word which remains when the prefix form is removed, 3) the rules for determining the semantic relationship of this particular prefix to the roots to which it is attachable, and 4) the meaning of the entire word.") the Thesaurus can provide the means for doing prefix analysis algorithmically. For example prevent can be properly analyzed as non-prefixed by determining that the word prevent does not occur in any of the categories related to categories associated with the unprefixed root vent. A prefixed word is thus operationally defined as any word with an initial prefix form which has a proper intersection of related categories with the word which remains when the prefix form is removed. A decision as to what constitutes a "proper intersection of related categories" will have to wait on 1) the implementation of the program suggested here and 2) the refinement of the hierarchy mentioned above.

However, a preliminary sampling of words analyzed by using the Thesaurus index in hand simulation with a table to equate a number of fourth level categories indicates that prefixes can be correctly identified in a large number of cases at the fourth level. Consider the following lists:

LIST 1: Words with the same 5th level categories. (Starred pairs occur in the same 6th level categories.)

centralize	225.10	decentralize	225.10	centrality
*sequence	153.1	consequence	153.1	effect
*valuate	493.9	evaluate	493.9	judgment
*announce	542.11	preannounce	542.11	forboding
*occupy	477.19	preoccupy	477.19	thought
	528.13		528.13	attention
buy	826.7	rebuy	826.7	purchase
*pole	238.2	antipole	238.2	contraposition
*junction	47.1	conjunction	47.1	junction
molar	257.7	premolar	257.7	sharpness
acid	378.1	antacid	378.1	chemicals

LIST 2: Words with the same 4th level categories.

fend	797.8	defend	797.10	defense
joy	863.2	enjoy	863.10	pleasure
rich	835.5	enrich	835.8	wealth
rupture	690.7	disrupt	690.11	impairment
guise	230.1	disguise	230.9	clothing
courage	891.1	encourage	891.16	courage
danger	695.13	endanger	695.6	danger
labor	166.9	elaboration	166.3	production
caution	893.1	precaution	893.3	caution
meditate	651.7	premeditate	651.8	intention
notice	701.2	prenotice	701.1	warning
adjust	60.12	readjust	60.9	arrangement
	692.11		692.13	restoration
adapt	26.12	readapt	26.13	agreement
arrange	60.7	rearrange	60.12	arrangement
admit	305.10	readmit	305.15	reception
compass	235.4	encompass	235.5	enclosure
ply	663.10	apply	663.11	use
ordain	638.7	preordain	638.6	predetermination

LIST 3: Words with adjacent, related fourth level categories.

conceivable			inconceivable		
	508.6	possibility		509.6	impossibility
	546.9	intelligibility		547.5	unintelligibility
refutable			irrefutable		
	513.15	uncertainty		512.15	certainty
thinkable			unthinkable		
	508.6	possibility		509.6	impossibility
capacity			incapacitate		
	156.2	power, potency		157.9	impotence

workable			unworkable		
	508.7	possibility		509.2	impossibility
stop			unstop		
	265.7	closure		264.12	opening
steady			unsteady		
	513.18	uncertainty		512.17	certainty
used			unused		
	123.18	oldness		122.7	newness
	640.16	custom, habit		641.4	unaccustomedness
welcome			unwelcome		
	923.9	hospitality, welcome		924.7	inhospitality
ideal			unideal		
	533.7	imagination		534.5	unimaginativeness
mature			premature		
	126.9	age		125.7	youngling
pertinent			impertinent		
	9.8	relation		10.6	irrelation

LIST 4: Words with non-adjacent, related fourth level categories.

tie			untie		
	47.9	junction		49.9	disjunction
	48.1	bound, fastening			
accuracy			inaccuracy		
	515.3	truth		517.2	error
build			rebuild		
	166.12	production		168.8	reproduction
separable			inseparable		
	49.19	disjunction		47.15	junction
doubted			undoubted		
	502.12	unbelief		500.20	belief

LIST 5: Words with no fourth level related categories.

toward	untoward	precede	cede
bend	prebend	face	preface
late	prelate	fix	prefix
coil	recoil	born	reborn
birth	rebirth	act	react
begin	rebegin	appear	reappear
strict	restrict	tinction	distinction
gaged	engaged	ordinate	coordinate
graph	telegraph	imitable	inimitable
moralize	demoralize	tribute	distribute
cent	recent	pare	prepare

Notice that the pairs given in the first four lists are words which it would be profitable to relate in any program such as the VIA program where the basic task is to discover semantic themes in a text. Almost all of the pairs agree with a native speaker's intuition of prefixation.

Some are felt to be related but not with the usual meaning predictable from the prefix form, e.g., preoccupy would mean 'to occupy beforehand' given the usual meaning of pre- and occupy. In other cases the pairs are related only in a sense which is relatively infrequently used, e.g., impertinent usually means 'impudent' rather than 'not pertinent' which is the basis for the analysis given here. On the other hand the relationship between some pairs is captured in several meanings as can be noted where two categories intersect for a single pair. A particularly clear example of this is with the pair used-unused where there are intersections of both uses of the words, 'old' and 'habit'.

In addition to the positive evidence for this program for prefix analysis, List 5 properly disallows a large number of pairs of words which are intuitively felt to be insufficiently related to be considered prefixed. However, there are several pairs in List 5 which could profitably be considered prefixed which are not so analyzed using the program suggested. This problem will be discussed below.

As a measure of the effectiveness of the suggested algorithm, 38 word pairs which Dr. Sedelow discusses in the research report of 1969 were tested to see if the results agreed with her assessment of the worth of the analysis. The pairs are given in the following lists under the classifications she gives. (A starred entry indicates that the pair was identified by the hand simulation of the algorithm discussed in the next section.)

LIST 6: Root groups helped by the PREFIX run.

integration-disintegration
*able-enable
*courage-encourage
*danger-endanger
*doubted-undoubted
moralize-demoralize
*capacity-incapacitate
*labor-elaboration
*sequence-consequence
*valuate-evaluate

LIST 7: Root groups harmed by the PREFIX run.

cent-recent
pare-prepares
tribute-distribution

LIST 8: Root groups which might be helpful in some contexts.

*compass-encompass
*conceivable-inconceivable
cover-discover
evitable-inevitable
*fend-defend
fluence-influence
*joy-enjoy
*junction-conjunction
ligation-obligation
mode-outmode
*ply-apply
promise-compromise
*refutable-irrefutable
*rich-enrich
*rupture-disrupt
see-foresee
*separable-inseparable
stead-instead
strict-restrict
*thinkable-unthinkable
tinction-distinction
gaged-engaged
ordinate-coordinate
vestigation-investigation
jugate-subjugate

In summary, of the ten pairs considered usefully related in List 6, eight are identified by the program and one exception (moralize-demoralize) is questioned by Dr. Sedelow in the original analysis; of the three pairs considered misleading, none are identified by the program as related; and of the twenty-five questionable pairs, eleven are identified as related and fourteen as not related. (The problem of different forms of the root has been ignored here, but should not cause a great problem in implementation.)

The program is also able to deal with cases where the identity of the prefix is in question. For example, the word unideal could be interpreted by the machine as either (un)ideal or (uni)deal. The program could disambiguate the analysis by conducting a category search of the units resulting from each segmentation. In this case unideal and ideal occur in adjacent, related categories while unideal and deal have no related categories.

In spite of considerable success in using the Thesaurus to analyze prefixes, there is a residue of problems. The primary problem is with word pairs which are intuitively felt to be prefixed, but which are not analyzed as such by the program. The most obvious examples are given in List 9.

LIST 9: Prefixed words not considered related by the program.

birth-rebirth	imitable-inimitable
begin-rebegin	weave-unweave
born-reborn	arrange-prearrange
appear-reappear	see-foresee

None of these pairs have intersecting adjacent or non-adjacent, related categories. In fact, most do not even share higher level categories, e.g., weave-unweave are unrelated except in the trivial sense that they both occur in the Thesaurus, as shown in the hierarchy below.

Thesaurus

Class 2: Space	Class 1: Abstract Relations
II. Dimensions	III. Quantity
B. Linear Dimensions	C. Conjunctive Quantity
221. Weaving	45. Simplicity
221.7	45.4
weave, loom, tissue;	... unweave ...;
(weave)	(unweave)

Although some of the other pairs are related at a lower point in the hierarchy they all appear intuitively to be as related as other pairs which have an intersection on the fourth level. The problem is thus the result of inconsistency in the organization of the Thesaurus. This is apparent with the adjacent, related categories. Most negation and reversal relationships are expressed in adjacent categories under the same third level heading. Examples are given above in the discussion of the equivalence table. However, the pair integration-disintegration are not related under the same third level heading even though they are adjacent categories.

III. Quantity

D. Wholeness	C. Conjunction Quantity
54. Whole	53. Disintegration
54.1	53.1
integration, embodiment;	disintegration, decomposition ...;
(integration)	(disintegration)

While there is a fourth level category labeled "disintegration" there is no comparable category labeled "integration" despite the fact that there are categories labeled "order" and "disorder" as well as "continuity" and "discontinuity."

The same inconsistency is apparent with talkativeness-untalkativeness where the third level categories containing the two words are quite far removed from each other unlike the other positive-negative category pairs.

III. Communication of Ideas

B. Modes of Communication	M. Uncommunicativeness; Secrecy
552. Communication	611. Uncommunicativeness
552.4	611.2
communicativeness, talkativeness ...;	taciturnity, untalkativeness ...;
(talkativeness)	(untalkativeness)

A different problem arises with the pairs birth-rebirth and born-reborn where the first word in each pair occurs in fourth level categories related to either "beginning" or "physical birth" while the second word in each pair occurs only in fourth level categories related to the religious experience of conversion. Therefore, the words in the pairs are not judged by the program to be prefixed since the metaphoric relationship between the two uses of birth are not related in the Thesaurus.

THE ALGORITHM

Figure 1 presents a high level flow chart of the system which has been tested through hand simulation using the present index of the Thesaurus. The completely automated version will use an index of the Thesaurus compiled by the computer in the form of an alphabetical listing of all the words and phrases in the Thesaurus with all the fifth level category numbers associated with each word or phrase.

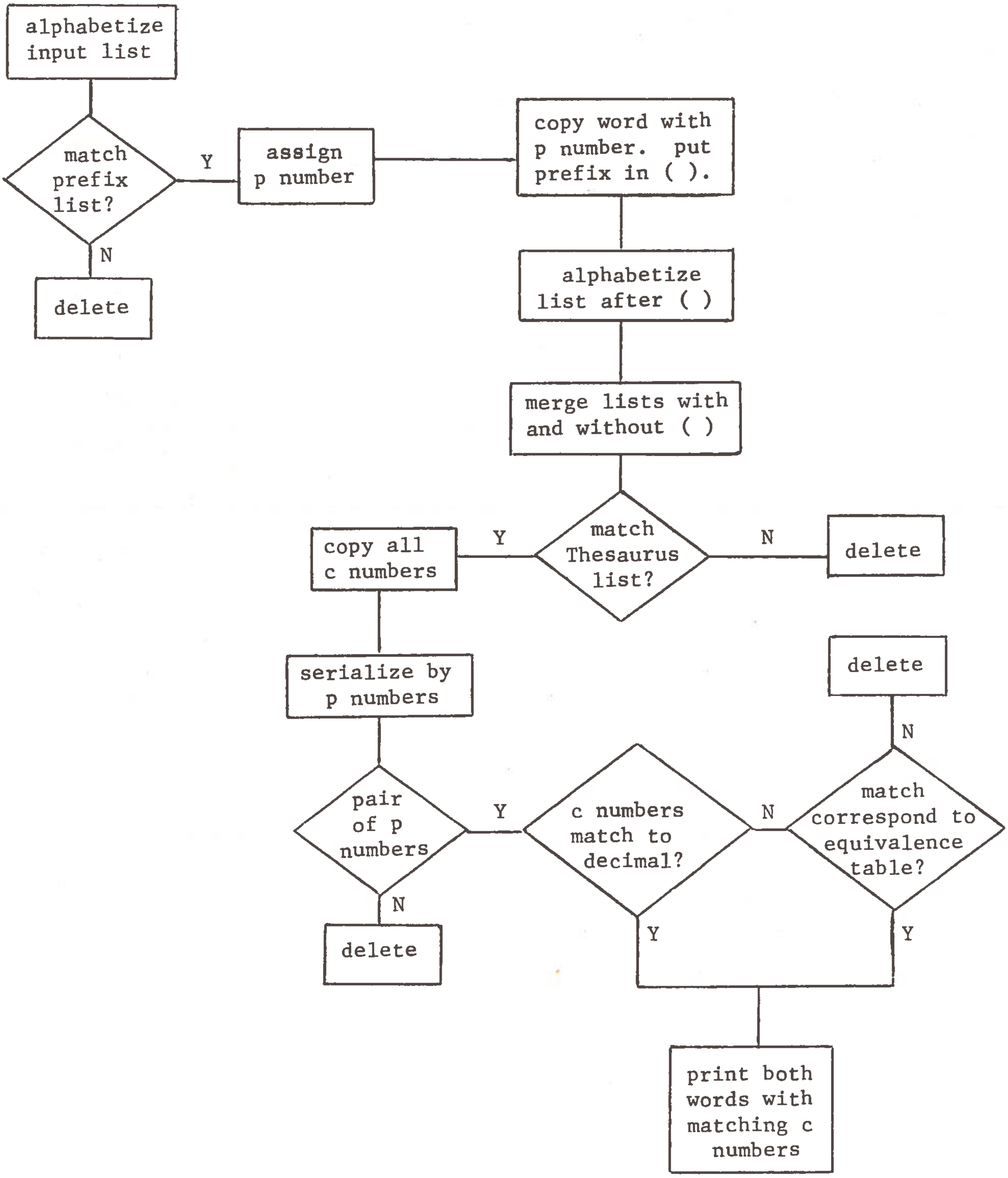
The initial phase of the program alphabetizes the input list of words to be tested for prefixation. This list is matched with a list of prefix forms. When a match is made the word receives a P (prefix) number and a copy of the word and P number is made with the prefix form enclosed in parentheses. A sample list might look like the following:

preach	P1	(pre)ach	P1
premature	P2	(pre)mature	P2
premeditate	P3	(pre)meditate	P3
prevent	P4	(pre)vent	P4

The list of copies is then alphabetized according to the letters following the close of the parentheses and the resulting list is merged with the list of original words.

(pre)ach	P1
(pre)mature	P2
(pre)meditate	P3
preach	P1
premature	P2
premeditate	P3
prevent	P4
(pre)vent	P4

Figure 1


alphabetize input list
match prefix list?
Y
N
delete
assign p number
copy word with p number. put prefix in ( ).
alphabetize list after ( )
merge lists with and without ( )
match Thesaurus list?
Y
N
copy all c numbers
delete
serialize by p numbers
delete
N
pair of p numbers
Y
N
delete
c numbers match to decimal?
N
Y
match correspond to equivalence table?
Y
print both words with matching c numbers

Through this process all words without prefix forms are dropped from consideration.

The file is now run against the Thesuarus index. Any character strings which do not match a word in the Thesaurus are dropped. At this point (pre)ach P1 would be dropped since there is not match for ach in the Thesaurus. When a match is made all C (category) numbers associated with the word in the Thesaurus are copied.

(pre)mature	P2	C126.9, C145.12, C321.6, C689.10 ...
(pre)meditate	P3	C477.11, C651.7, C477.12
preach	P1	C597.3, C560.16, C597.10 ...
premeditate	P3	C651.8
premature	P2	C125.7, C130.7, C131.8
prevent	P4	C728.14
(pre)vent	P4	C264.4, C302.2, C302.9, C395.17 ...

The file is then serialized by P numbers and any words without matching P numbers are dropped. In our example preach would be dropped at this point since (pre)ach has already been dropped and there is no longer a pair of words with the number P1.

The final part of the program matches the C numbers of words having the same P number. Since preliminary investigations indicate that for the purpose of prefix analysis a match at the fourth level of the semantic hierarchy is most productive, a match will be attempted only to the decimal point. If a match of C numbers is made, the word without parentheses is considered to be prefixed and is printed with the matching C number(s). If there is no direct match a further test is made to see if the C numbers match C numbers on the equivalence table which relates categories in the manner discussed in the first section. The table would look something like the following:

1=2, 3=4, 5=6, 9=10, 60=63, ..., 125=126, ...

If a match is made using this table, the word without parentheses is considered to be prefixed and is printed with the C number(s) which are related through the equivalence table. Words which fail this test are not considered to be prefixed and are dropped.

premature	P2	C125.7, C130.7, C131.8
(pre)mature	P2	C126.9, C145.12, C321.6, C689.10 ...
premeditate	P3	C651.8
(pre)meditate	P3	C477.11, C651.7, C477.12

In the example meditate and premeditate share C numbers (651) and mature and premature share C numbers related through the equivalence table (125=126). Vent and prevent fail both tests and prevent is thus not considered to be prefixed.

CONCLUSIONS

From this brief investigation it appears that further research into the use of the Thesaurus as a semantic relatedness metric promises to be fruitful at least as a source for automated prefix analysis. If the problems of inconsistency in the Thesaurus mentioned above can be solved and the equivalence table established on some principled basis, the algorithm given here (or some modification of it) could provide a list of prefixed words in English using the word list of a dictionary as input. The short-term, practical result would be to substantiate and/or correct the intuitively compiled inclusion-exclusion list presently used with the PREFIX program in the VIA package. The more theoretical results will be in three areas: 1) The functioning system will provide a principled means for determining which initial letter sequences should be considered prefix forms in English. Some prefixes on present lists may have to be dropped if analyzing them as prefixes does not produce a sufficient number of prefixed-non-prefixed pairs. 2) The system will provide a metric for measuring productiveness of prefixes. Those which produce the largest number of prefixed-non-prefixed pairs can be considered the most productive in the language with some allowance made for mitigating factors such as technical vocabulary.

3) The development of the system will provide considerable information about the organization of the Thesaurus and the semantic fields it is designed to explicate. This information may suggest improvements in the design of thesauri in general.

III. B. Disambiguation of Prefixes Through Context

by Sam Warfel

In another paper in this report I discuss an algorithm for prefix analysis which uses Roget's International Thesaurus. One problem which I did not discuss there concerns a number of words which are ambiguous relative to prefixation, i.e., these words are properly considered to be prefixed in one semantic context but not prefixed in another context. For example, aside from the context, there is no way to determine whether the following words should be analyzed as prefixed: recoil, recollect, recover, reflex, rebuff, relay, return, resent, resign, present, prevent. Notice that in each case there is an interpretation which is consistent with an analysis of the word as prefixed and an interpretation which seems to have almost no semantic relationship to the corresponding prefix analysis. Notice also that some words seem to have a greater chance than others of occurring either with a prefix interpretation or not. For example, recoil, recollect, and rebuff appear to have a better chance of occurring with a prefix interpretation than words such as return, present, or prevent. The fact that none of the six words would be considered prefixed as analyzed with the Thesaurus algorithm mentioned above probably reflects the low probability that any of the words will occur as prefixed words. Nonetheless, all of the six can be understood as prefixed in the proper context, and I feel that a distributional analysis would show a difference in probability between the two groups.

To deal with the problem of context sensitivity in prefix analysis I suggest a modification of the PREFIX program to allow the context to be considered before the final determination of prefixation is made in the program. As it presently operates the PREFIX program matches the initial letters of a word with a list of prefixes. When a match is found a search is made of a list of words beginning with the prefix sequence. If the text word matches a word on this list (called a CLUD list) then the key associated with this list is consulted to determine whether the list contains words which are to be included as prefixed words or excluded as exceptions having the correct initial letter sequence, but not considered as prefixed. If there is no match with a word on the CLUD list and the key indicates that the list contains exclusions, then the word is considered to be prefixed.

I propose to change the program so that each text word is matched not only with the CLUD list, but also with a list of prefix-ambiguous words. An additional search of the text is made to determine if words semantically related to the root occur in the immediate context of the word in question. If a sufficient number of related words are found to satisfy the weighting requirement (to be discussed below) then the word is considered to be prefixed.

The proposed program defines the immediate context as the paragraph preceding the paragraph in which the word in question occurs and the portion of the paragraph in which it occurs which precedes the occurrence of the word. This context is searched for words which are closely semantically related to the stem which remains if the prefix is removed. These words (SYN words) can be supplied from synonym dictionaries, thesauri, or the researcher's own experience with the language. For example, the decision to consider recoil to be a prefixed word will depend on the occurrence of the following words (gleaned from the coil entry in Roget's International Thesaurus) in the immediately preceding text: coil, convolve, wind, twine, twirl, twist, turn. One might also want to include words referring to coilable objects such as rope, cable, lasso, and wire and to snakes such as snake, rattler, viper, etc. The program allows the decision to be weighted by varying the threshold for the minimum number of occurrences of words on the SYN list which must be present before a decision in favor of prefix analysis is made.

The proposed program necessitates some minor changes in the PREFIX program. These changes are given in a flowchart beginning on page 59 where the single lined boxes refer to the present program and the double lined boxes represent changes. (See Sedelow 1969:246 for the complete flow chart of the present program.) In order for the revised program to work as expected, each of the ambiguous words will have to be included on the CLUD lists and marked with a Q number consisting of a two digit number unique to each word and a one digit number indicating the weighting factor. (This code allows 99 Q words, a number which appears to be sufficient.) Thus the Q number for recoil might be Q12-2, where the last digit is the weighting factor. The revised program will then recognize the ambiguous words and remove them from consideration before the inclusion-exclusion decision made through use of the key. The ambiguous words (Q words) are segmented, that is parentheses are placed around the prefix sequence of letters, and a copy of the Q word and the location(s) number(s) of the text word are stored in FOLLOWUP. The only other change is to record the results of PREFIX on tape instead of printing it directly, since the results are subject to revision depending on the outcome of the program proposed here.

Before a discussion of the program is appropriate, however, some attention must be given to the organization of the list of SYN words. Since there is considerable saving of time in processing the text alphabetically, it is deemed necessary to establish the SYN list in alphabetic order. This decision is not without complications since the list must include words related to all of the Q words even though only a subset of Q words may occur in a given text. Several solutions are possible. All of the SYN words might be searched for in the text with a reference back to the text-specific Q word list to see if the particular Q word in question is in the text under consideration. An alternative is accepted here. The SYN list is compared with the list of Q word numbers in FOLLOWUP and only those with a correspondence number are retained for the search of the text. Whether this is the most economical approach is left up to the programmer and/or a test of the program.

Conceptually then the SYN list is a set of lists interleafed to maintain alphabetic order. It might be easier to have separate lists for each Q word and alphabetize the necessary SYN words resulting from the occurrence of the text-required set of Q words. Again, I am not able to make a decision at this point. Therefore, I conceive the SYN list for the Q words recoil, recollect and present to look something like the following: (the particular words chosen are intended to be only illustrative)

Q Words: (re)coil Q12-2
(re)collect Q32-3
(pre)sent Q07-6

SYN Words:

Q32-3	accumulate
Q32-3	amass
Q32-3	assemble
Q32-3	cluster
Q12-2	coil
Q32-3	collect
Q12-2	convole
Q07-6	dispatch
Q32-3	gather
Q32-3	group
Q12-2	lasso
Q07-6	mail
Q07-6	post
Q12-2	rope
Q07-6	send
Q07-6	sent
Q12-2	twine
Q12-2	wind

The proposed program (LOOKSEE) is given in Appendix B in flow chart form. The first step in the program calls a subroutine GETREADY which builds a text specific SYN list. The first portion of GETREADY reads in the list compiled by PREFIX and stored in FOLLOWUP. The program then searches the SYN list and compiles a list of SYN words in alphabetic order which are related to the Q words on the FOLLOWUP list as indicated by the referencing Q numbers. Control then passes back to the main program.

The program next establishes counters for all of the location numbers given in the FOLLOWUP list, labels each counter with a Q number and location number, and sets the counters to "0." At this point the text is processed again. Each word is compared with the list of SYN words (a process simplified by the fact that both lists are alphabetized). When a match is found the program calls a subroutine LOCATE.

LOCATE reads in the location number(s) listed with the Q word associated with the matched SYN word. This location number is compared with the location number(s) of the matched text word. If a comparison of the chapter, paragraph, and sentence numbers indicates that the text word occurs preceding the Q word either in the immediately preceding paragraph or in the sentences of the same paragraph, then the appropriate counter is incremented by 1. When this subroutine has examined all of the possible location number matches, control passes back to the main program which continues to match SYN and text words.

When all of the text has been processed, the reading of each counter is compared with the last digit of the corresponding Q number. If the counter reading exceeds the weighting number related to the Q word, the word is considered to be a prefixed form and is entered accordingly in its proper order on the tape which was the output of PREFIX. If the counter reads a smaller number than that of the Q word, the word is considered not to be a prefixed form and is not entered on the prefix list. After all counters are read the program ends.

The final digit of the Q word can be set by the researcher as high as is necessary to provide the weighting which I suggested was necessary for a proper analysis. In the example of Q words given above the entry for present was weighted with a 6. This indicates that the word present usually is not to be analyzed as prefixed. However, in the context of 7 or more occurrences of words relating to "sending" the likelihood that the word does mean "to send beforehand" is greatly increased. On the other hand, this number allows less than 7 occurrences of send-related words without forcing a prefix analysis. The numbers selected in the examples are of course quite arbitrary at this point.

Notice that the program allows different decisions for different tokens of the same Q word type. Since the counters are incremented by location matches, it may be the case that the word recoil exceeds the Q word threshold in contextual SYN words at one point in the text but does not exceed that threshold at another point. This feature would seem to be necessary in dealing with a lengthy text.

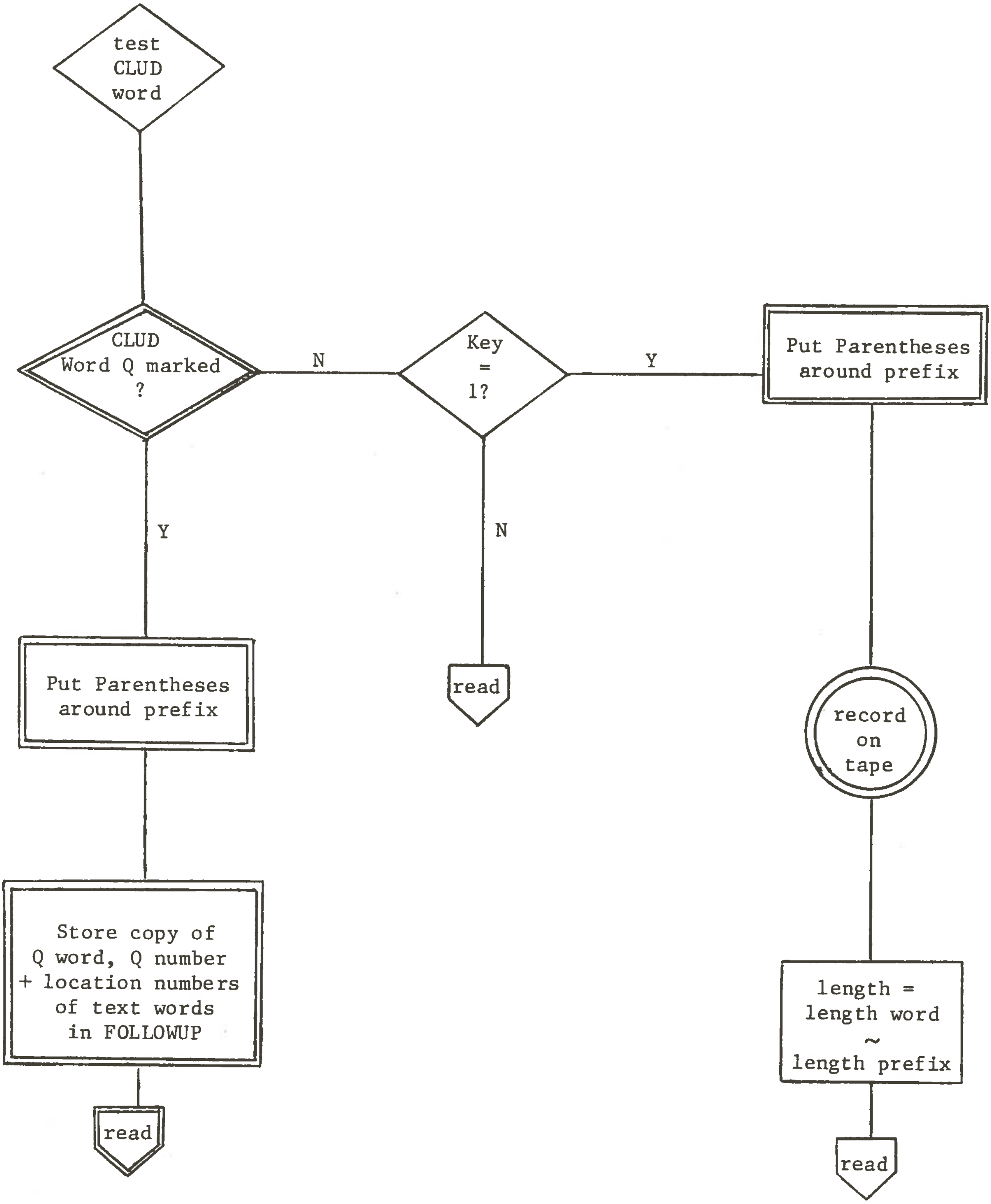
The entire program rests on several assumptions about the nature of texts. The strongest claim is that the semantic context is relevant to the disambiguation of the words in question and is discoverable by looking at the lexical items which occur in the immediately preceding sentences. I have no evidence that this is so. It does not, however, seem unreasonable that the reader must be forewarned in some way so that when one of these ambiguous words occurs he knows immediately how to analyze it and that this forewarning must include at least the subject matter under discussion which in turn must involve lexical choices. A second assumption which is even more tenuous is that there is a constant ratio between the analyzability of these words as prefixes and the number of occurrences of words semantically related to the stem of such a word. Again I do not know that this is true. What does appear obvious is that some of these words are more often ambiguous than others. It should, therefore, be possible to establish ratios for the number of times the word occurs as a prefixed form as opposed to the number of times it occurs as a total unit. However, this information will not help the computer to know in any given instance whether or not the word should be considered as prefixed. Therefore, I have built in the weighting factors just in case it proves to be true that a greater number of occurrences of words semantically related to the stem does indicate when the word should be analyzed with a prefix.

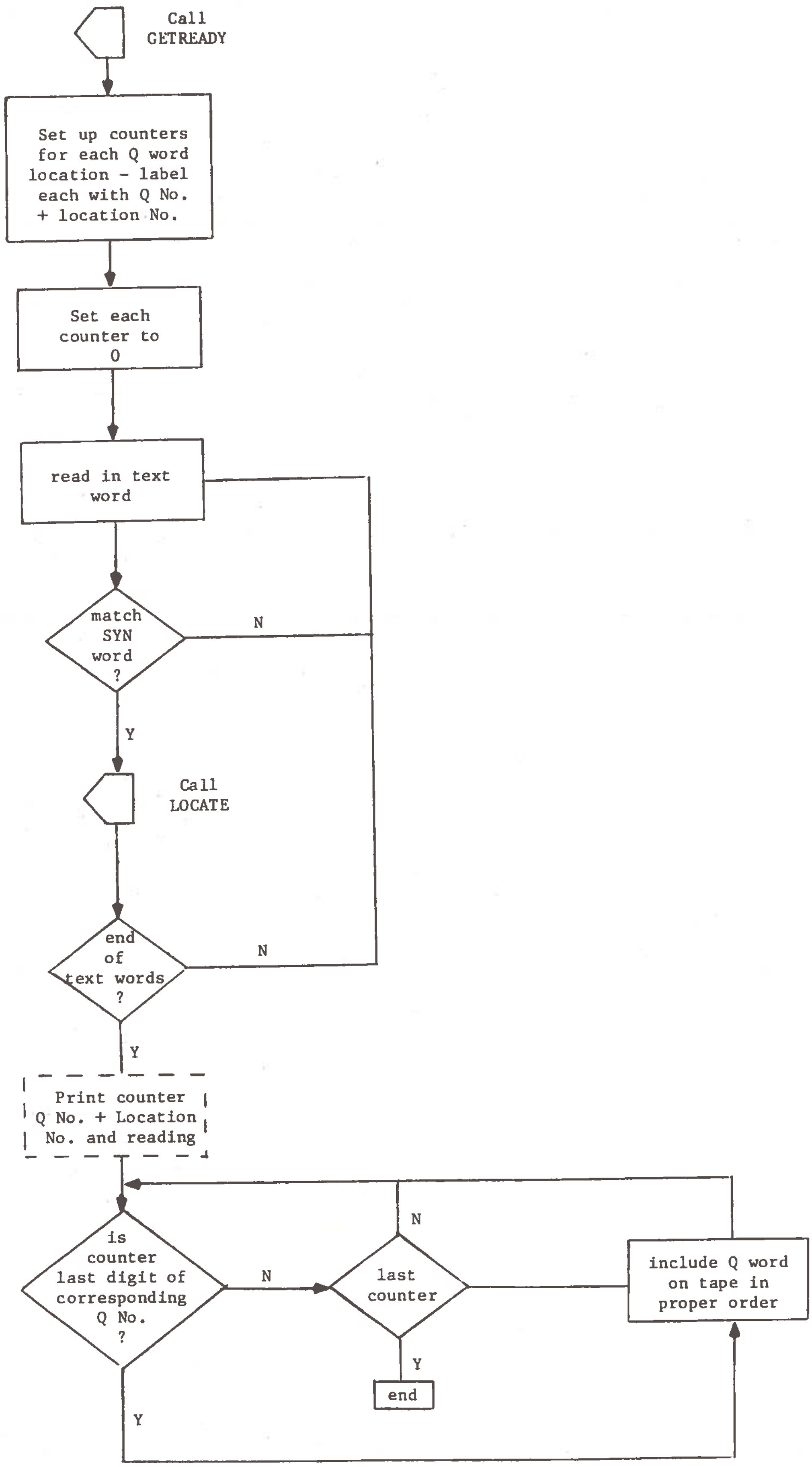
Both of the assumptions are empirically testable. The program itself should provide the means of testing the assumptions. Although it is written to be used as part of the PREFIX package, the program could be used for testing these assumptions by using the printing instructions given in the dashed lined boxes on the flow chart. The printout should provide the necessary information for establishing the optimum SYN words and the proper weighting factors.

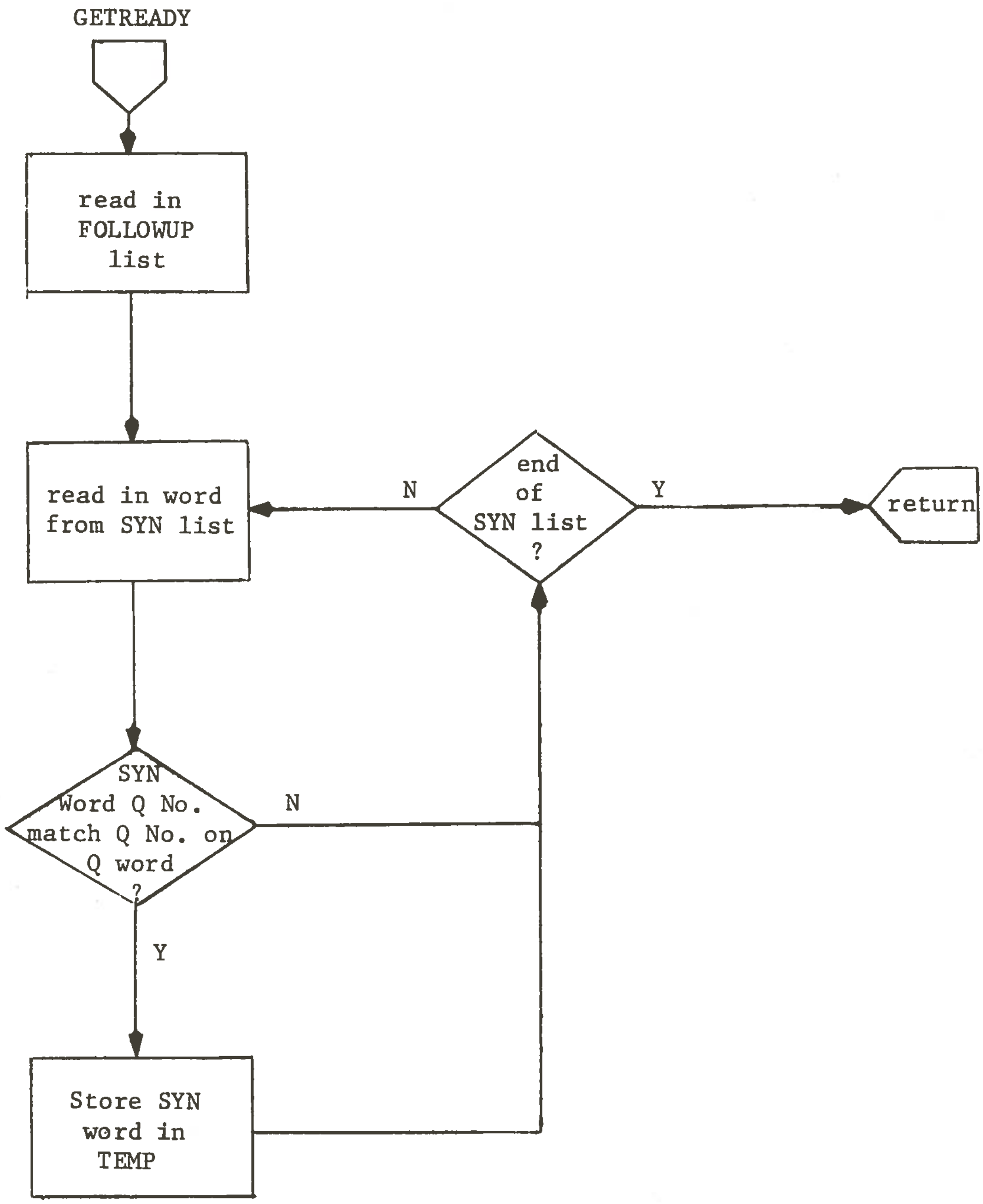
BIBLIOGRAPHY

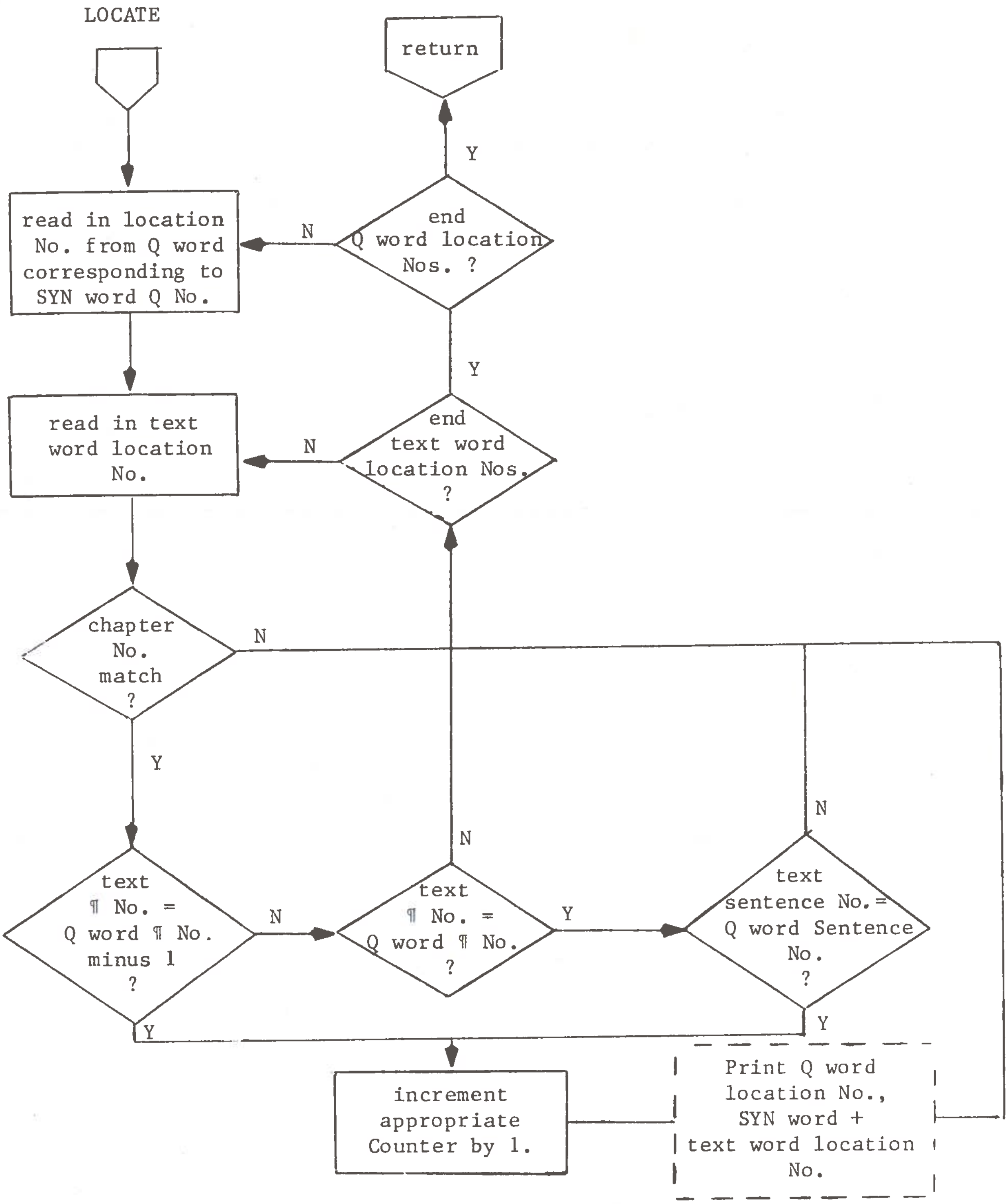
Roget's International Thesaurus. 1962. 3rd. ed. New York: Thomas Y. Crowell Company.

Sedelow, Sally Y. 1969. Automated Language Analysis. Departments of English and Computer & Information Science, University of North Carolina, Chapel Hill, N. C.


test
CLUD
word
CLUD
Word Q marked
?
N
Key
=
1?
Y
Put Parentheses
around prefix
Y
N
Put Parentheses
around prefix
read
record
on
tape
Store copy of
Q word, Q number
+ location numbers
of text words
in FOLLOWUP
length =
length word
~
length prefix
read
read


Call
GETREADY
Set up counters for each Q word location - label each with Q No. + location No.
Set each counter to 0
read in text word
match SYN word ?
N
Y
Call
LOCATE
end of text words ?
N
Y
Print counter Q No. + Location No. and reading
is counter last digit of corresponding Q No. ?
N
last counter
N
Y
end
include Q word on tape in proper order
Y


GETREADY
read in FOLLOWUP list
read in word from SYN list
N
end of SYN list ?
Y
return
SYN Word Q No. match Q No. on Q word ?
N
Y
Store SYN word in TEMP


LOCATE
return
Y
read in location No. from Q word corresponding to SYN word Q No.
N
end Q word location Nos. ?
Y
read in text word location No.
N
end text word location Nos. ?
chapter No. match ?
N
Y
N
N
text ¶ No. = Q word ¶ No. minus 1 ?
N
text ¶ No. = Q word ¶ No. ?
Y
text sentence No. = Q word Sentence No. ?
Y
Y
increment appropriate Counter by 1.
Print Q word location No., SYN word + text word location No.

IV. Statistical Support System Overview

by Frank Joyce

This section contains a discussion of the statistical support system as it now exists and of some of our plans for future expansion. This section will be followed by a description of some of the results of applying the system to "live data." Since the current system is still in the experimental stage, we would expect to make a number of changes in the near future, both with respect to programs and to data files being passed between modules. However, the broad outlines of the system should remain fairly constant.

The system is composed of a number of program modules (see figure 1).* Each program is assigned a task that is rather limited as to scope in order to allow a flexible system so that new functions can be more easily integrated and so that new methods of interaction between functions can be implemented as painlessly as possible. As experimentation continues, we hope to be able to identify more clearly just which data processing and data analysis functions are most central to the language analysis task and then create a system of modules implementing these functions which would allow an investigator in effect to create his own system by controlling the flow of data from module to module.

The present system allows the investigation of a variety of language features. One segment of the system can be used to examine the nature of the type-token function (the relation between the number of word types, N, and the number of tokens, n). A number of models have been proposed to describe this function (Carroll, 1968). The subsystem extracts data on the growth of vocabulary with respect to text length

* pp. 87 ff.

and this data will be fed into programs now under development which will summarize it and try to fit the empirical data to the theoretical models. At the same time data is extracted which is used by later programs to build distributions of frequency of vocabulary usage.

Another subsystem builds and analyzes frequency distributions for a number of characteristics. As was mentioned last year (Joyce, 1971), frequency distributions are commonly used in the area of computational stylistics to summarize such characteristics as letters per word, words per sentence, or frequency of word usage. These distributions can be further summarized by calculating measures of central tendency, such as the mean or various percentiles, and measures of dispersion, such as the standard deviation or the inter-quartile deviation. The distributions may also, under certain circumstances, be used as input for programs which handle contingency table analysis, perhaps in order to examine differences in the distributions obtained from different samples taken from the same or different works. A more detailed description of this subsystem and other subsystems will be found below.

Still other subsystems examine the placement of function words within the sentence or carry out information theoretic and contingency table analyses of data supplied by other programs in the system. These latter three subsystems are still in a very experimental stage of development, and we hope greatly to expand their power in the near future. We also plan to add subsystems to carry out functions such as cluster analysis, analysis of variance, and factor analysis as the need arises.

SVIA01

The heart of the current system is program SVIA01, which uses the text file created by the existing VIA system and creates data files to be used by the statistical system. The output file from program SUFFIX is sorted back into the linear order of the original text file (SUFFIX operates on the text file sorted into alphabetical order). Three output files are created. The first file contains frequency data for the following characteristics: letters per content word, letters per function word, letters per word (combines the two previous mutually exclusive categories), letters per sentence, content words per sentence, function words per sentence, total words per sentence, words per paragraph, sentences per paragraph, sentences per chapter, and paragraphs per chapter.

SVIA01 is based on a program described in last year's report (Joyce, 1971) which has since been modified. For example, the data which was gathered for the unit we called a clause will not be extracted until we develop a more theoretically satisfying way of indentifying clauses. Also the previous program extracted data for characteristics which were somewhat redundant. Thus, the distribution for letters per paragraph could probably be predicted with a fair degree of accuracy by looking at the distributions for letters per word and words per paragraph. In addition, the wide variation in values which can be found for such a characteristic greatly complicates problems of storage allocation for the resulting frequency distribution. While it might be useful in some future application to again extract data for such characteristics, it was decided to simplify the present system by deleting these characteristics. The advantage of writing a modular system is that such additions and deletions can be made with a minimum of pain.

SVIA01 is itself actually composed of six subroutines. One subroutine handles the necessary details for processing the incoming word, another subroutine is called when the end of a sentence is reached and handles the necessary bookkeeping entailed by the new sentence. Thus, if it were deemed useful to once again extract information about letters per paragraph, it would be a simple task to create a new counter which would be incremented at the end of each sentence (by adding the counter which keeps track of letters per sentence to it) and passing the counter to the routine which handles the end of paragraph condition. Five of the six routines are: SVIA01, which is the main control module; NEWCHP, which is called when the end of a chapter is encountered; NEWPAR, which is called at the end of a paragraph; NEWSEN, which is called at the end of a sentence; and NEWORD, which is called after each word is read in.

The sixth subroutine is NSEG. This routine recongnizes a new textual unit, that of a fixed length segment, the size of which is determined by the investigator. As we observed last year, it is not entirely clear what constitutes a valid sampling technique when one is dealing with natural language. Two primary questions may be asked. Firstly, how large a sample is required to adequately represent a larger body of text with respect to the characteristics which interest the investigator? Secondly, how should the sample be extracted? Factors relevant to the second question include the proper unit, that is, whether words, sentences, or whole paragraphs are to be extracted, and whether the sample should be spread as evenly as possible throughout the text or extracted in such a way as to obtain a large body of continuous text. In order to obtain satisfactory answers to these questions, it will be necessary to make a detailed investigation of the structural characteristics of various stylistic features throughout a body of text. To facilitate such an examination, the text is broken up into segments and the frequency records contain a field which identifies the segment from which it came. We currently use a basic segment size of 1,000 words. Of course, sentence and paragraph boundaries rarely fall exactly on a segment boundary. Rather than trying to apportion such units between the two adjoining segments, thereby destroying their identity as sentences or paragraphs, we arbitrarily assign the over-lapping unit to the segment in which it began. While this method is the simplest from the standpoint of programming logic, it does open the possibility that a very long paragraph could be assigned to a segment in which only the first word of that paragraph occurred. An alternative method would be to assign the unit to the segment which contains the major portion of the unit. If one is dealing with texts which contain many long paragraphs, it might be best to ignore the values obtained for small segment sizes. Fortunately, this problem does not seem overly crucial in the texts we are currently studying. For example, in the first chapter of the RAND translation of Soviet Military Strategy, the longest paragraph contained 207 words and 95 percent of the paragraphs contained 125 words or less. Proper care should be taken in making comparisons between distributions for different segments; however, in general, paragraph lengths of this magnitude should not be expected to lead to serious biases, particularly after the distributions are combined into distributions for segments of up to 16,000 words. Other programs in the system build frequency distributions, combine distributions for each segment into distributions for larger segments, and provide information about the stability of these distributions from segment to segment. These programs include SVIA21, SVIA31, and SVIA41 which are described below.

Records in the frequency file have three fields. The first field is a numeric "type" code which identifies the characteristic for which the record was obtained. For example, the type code for "letters per content word" is 1 while the type code for "letters per function word" is 2. Since the records are obtained by processing the text file in text order, values for various characteristics will be intermixed. The type code allows the file to be sorted, thereby bringing all records of the same type together for further processing, such as building a frequency distribution. The second field identifies the segment from which the datum was drawn. A "1" in this field means the datum was obtained from the first segment, a "2" means the second segment, and so forth. The third field contains the actual datum itself. As an example, consider a content word with five letters drawn from the first segment. Then the corresponding record would contain "1" in the type field, "1" in the segment field, and "5" in the datum field.

The second output file contains information about the placement of function words within the sentence. In general, function words, such as articles, conjunctions, and prepositions, are considered to have little or no independent meaning within the sentence. Rather, they serve as syntactic markers. In light of this assumption, we are interested in studying the positional features of function words within the sentence. In addition, the programs developed for this purpose may prove useful for studying positional characteristics of other features within larger units of discourse. A record is created for each function word showing the segment in which it occurred, its location within the sentence, and the total length of the sentence. These records are used by later analysis programs.

The third output file is a modification of the input file. The input file contains indexing information provided by program INDEX. This includes fields giving the linear order within the text for each word or punctuation record, starting with the first record, and the position of each record within each sentence. However, INDEX does not make a distinction between actual text words and punctuation marks. This distinction is made later by program SUFFIX. SVIA01 recalculates these two index items in terms of words only, i.e., it ignores the recently identified punctuation marks. One advantage of this recalculation is that after the file is sorted back into alphabetical order, the segment number of any word can be calculated by dividing the new linear order number by the segment size. At this point we were faced with making a decision about where to put these new fields. We could have substituted them for the original fields; however, since there is some remote possibility that a later application might be found for the information that includes the punctuation marks, we decided against this course. Rather than expand the record size at this time, we decided to store the information in two existing fields that are apparently of somewhat minor importance. The new word in sentence counter is stored in the field normally reserved for the page number of the original text on which the word was found and the new linear order counter is stored in the idiom flag field, which is currently unused. Since these fields could conceivably prove useful in the future, the permanent version of this system will probably expand the output file by two fields. This file is used by programs which extract information about vocabulary growth, frequency of vocabulary usage, and information theoretic properites of letters.

SVIA21

The raw frequency data file created by program SVIA01 is sorted into ascending order by type (i.e., words per sentence, sentences per paragraph, etc.), segment, and value (number of words, sentences, etc.). This file is used as the input file for program SVIA21. The program produces a listing which contains the frequency distributions and various summarizing statistics for each segment within each data type (see figure 2).* In addition, it combines the distributions for the original 1,000 word segments to produce distributions for segments of 2,000, 4,000, 8,000 and 16,000 words. Included in the summarizing statistics for each segment and segment size are the high and low values, arithmetic mean, variance, standard deviation, the coefficient of variation, the first, fifth, 25th, 50th (median), 75th, 95th, and 99th percentiles, the inter-quartile deviation, and the moment coefficients of skewness and kurtosis.

The frequency distribution records are written out to a file for use by later programs. The records in this file contain fields showing the type code, segment size code, datum value, and the frequency with which that value occurred. This file was created by accumulating records in the original frequency file which had identical values. It should be noted that this file contains records only for those values which actually occur in the text. Thus, if the text does not contain any words which have eight letters, SVIA21 does not create a frequency

* p. 95

record for the value eight showing a frequency of zero. This is in contrast to program SVIA31, which will be described next, which writes out the entire frequency distribution, including those values which have a frequency of zero. At this point, it is not clear which method is more efficient. The first method does not require passing information about zero frequencies; and this has certain advantages since many of the distributions we deal with are highly skewed--that is, the bulk of the values fall in a cluster toward the lower end of the distribution with a few values strung out to the right. However, it is necessary to carry along four fields of information for each record using this method. When writing out the entire distribution, the first two fields can contain the low and high values respectively and remaining fields can contain the frequencies for each successive value. Thus, the necessity of passing zero values may be offset by a saving of "overhead" information fields. Further experience with the system should provide information about the relative merits of the two methods.

SVIA31

Program SVIA31 continues the analysis of the frequency data begun by program SVIA21. The latter program produces a listing showing the frequency distribution and its summarizing statistics for each segment within each type of distribution. Comparison of results between segments is somewhat complicated by the necessity of turning from page to page in the output. Program SVIA31 alleviates this difficulty by listing the summarizing statistics for each type and segment size in tabular form (see figure 3).* In addition, the summarizing statistics are passed to program SVIA41 which provides a summary of the individual statistics themselves with respect to the segments. This facili-

* pp. 96-97.

tates the determination of stability characteristics between segments -- for example, by providing information about the variation found in the values of each test statistic within segments of the same size. The frequency distributions are output in a form which can be used by contingency table analysis programs to further examine stability characteristics.

SVIA41

As mentioned above, program SVIA41 facilitates the investigation of the stability of various test statistics throughout the body of the text, particularly with respect to the effects of segment size. For each segment size within each type of frequency distribution, program SVIA31 produced a variety of summarizing statistics for each segment. SVIA41 accepts as input the values of these statistics themselves for the segments and finds the arithmetic mean, variance, standard deviation, coefficient of variation, and coefficients of skewness and kurtosis for each statistic (see figure 4).*

VOCAB1 and VOCAB2

Programs VOCAB1 and VOCAB2 are used to derive information about vocabulary, i.e., whether the text contains a wide variety of words or a relatively small number of words used with great frequency. The prediction of the type/token function would allow the investigator to estimate the total number of different types that would be obtained if word tokens were sampled indefinitely. This would provide an estimate of an author's total working vocabulary.

VOCAB1 accepts as input the text file created by SVIA01 after it has been sorted back into alphabetical order. In addition, the records for the same word type should occur in the same order as in the text.

* p. 98

That is, the linear order number should be used as a secondary sort field to ensure that tokens which occur earlier in the text are processed before tokens of the same type which occur later. This ordering facilitates the calculation of frequency data for each text segment.

As has been observed, the segment in which the token occurred can be calculated by dividing the linear order number calculated by SVIA01 by the segment size. VOCAB1 calculates the frequency with which each type occurred in each segment for segment sizes of 1-, 2-, 4-, 8-, and 16,000 words. Each record in the word frequency file has three fields: a type code which indicates the segment size, the segment from which it was drawn, and the frequency. The format of these records allows them to be input to program SVIA21 and the programs which follow it. A parameter card is input to SVIA21 to cause the program to bypass the section of code in which the distributions for the basic segment size is combined into larger sizes. This calculation has already been done by VOCAB1. A file is also created containing a record for each word type which shows in which 1,000 word segment that word type first occurred. These records will be used to calculate information about vocabulary growth with text growth.

The Carroll article reports the results of applying the lognormal model to the Brown University 1,000,000 Word Corpus (Kucera and Francis, 1967) and the Lorge Magazine Count (Thorndike and Lorge, 1944, pp. 252-253, cited in Carroll, 1968). In both cases the count was made "by counting separately every combination of letters found." Therefore "_arm_, _arms_, _arm's_, _arms'_, _arming_, and _armed_ were counted separately." The results from VOCAB1 will likewise count these forms as different words. However, the text file has been run through the VIA system including SUFFIX and, optionally, PREFIX; and, therefore, we can also calculate vocabulary usage information that is based on root forms, rather than graphic forms. Program VOCAB2 is similar to VOCAB1, except that it accepts as input the text file from SVIA01 sorted on the matchcount filed instead of alphabetically sorted. The output files reflect the values obtained for the word forms grouped together under the same matchcount. We plan to develop some routines to allow us to make comparisons between distributions obtained using the two differing methods.

SVIAF1 and SVIAF2

Program SVIAF1 builds a triangular contingency table for function word location data for each segment. The rows of the table correspond to the function word's position within the sentence and the columns correspond to the length of the sentence in which the word occurred. Thus, cell (i,j) of the table contains the number of times that a function word occurred in the ith position of a sentence that had j words. These tables are written out to a file for further analysis and are also written out on a listing for the investigator. However, we have not yet found a truly satisfactory method of producing this listing. There is a wide variation in the lengths of sentences; and, therefore, the resulting table is too large to be output on a single sheet of paper. At present, each row takes five pages and, even then, only shows the results for sentences with 150 or fewer words. In order for this information to be valuable to the investigator, we must find ways of summarizing the data.

Program SVIAF2 is almost identical to SVIAF1, except that the table is built for the entire text rather than for each segment. The input file for SVIAF1 has been sorted in ascending order on the segment, location, and sentence length fields, in that order. The same file is sorted only on the location and sentence length fields before being passed to SVIAF2.

INFORMATION THEORETIC SUBSYSTEM

For this project, style has been defined as "the patterns formed in the linguistic encoding of information." (Sedelow and Sedelow, 1966). A branch of statistical analysis which provides a methodology for quantifying organization, or patterning, is information theory. Thus, we have begun to explore the possibility of applying information theoretic techniques in order to study linguistic patterns. The following brief discussion is based heavily on (Attneave, 1959), and a more detailed, very readable discussion of basic concepts may be found therein.

The formal concept of information is based on the idea that the less expected the outcome of a given event is, the more informative that outcome is. Thus, the informational value of a particular event is sometimes called the surprisal of the event. Formally, if p_i is the probablity that the ith alternative among all possible outcomes for an event, will occur, then h_i, the informational value of the ith alternative, is defined by the equation:

$$h_i = \log(1/p_i).$$

(Throughout this discussion, logarithms are taken to base 2.) The measure of the average information associated with an event will be the sum of the information values for the individual alternatives with each multiplied by its probability as a weighting factor. Thus, letting

H represent this measure (the Shannon-Wiener measure), we have:

$$H = \sum_i p_i \log(1/p_i) = -\sum_i p_i \log p_i.$$

H takes on its largest value when all alternatives are equiprobable, that is, given m possible alternatives:

$$H_{max} = \Sigma\, p \log(1/p) = (\Sigma p)\log(1/p) = \log(1/p) = \log m.$$

The relative information, or relative entropy, associated with a particular value of H is:

$$R = H/H_{max} = H/\log m.$$

The redundancy associated with an event is defined as the complementary quantity: C = 1 − R. Thus, redundancy is a measure of the predictability of an event.

The concept of redundancy can be applied to the results of a stochastic process, where a stochastic process is defined to be "any system which gives rise to a sequence of symbols to which probability laws apply" (Attneave, p. 13). The sequences of letters and words making up a text are examples of stochastic processes. In order to apply most information theoretic measures, the stochastic process should be at least approximately ergodic; that is, the probability laws characterizing the process should remain constant for all parts of the process.

The redundancy of a stochastic process ranges from 0, when all symbols are equally likely and thus the predictability of the next symbol is not increased by knowledge of the history of the sequence, to a maximum value of 1 when symbols are generated in a completely lawful manner and thus the next symbol can be predicted with complete accuracy. The order of redundancy is an indication of the way in which the pattern is redundant. First order redundancy results when the individual symbols have unequal probabilities. An example of complete first order redundancy arises when a "two-headed" coin is repeatedly tossed, yielding the sequence HHHH.... Second order redundancy arises when knowledge of the immediately preceding symbol increases the predictability of the following symbol. An extreme case is the alternating sequence HTHTHTH... in which knowledge of the preceding symbol determines the second symbol. In general, "a sequence has Nth-order redundancy whenever some of the possible patterns of N successive symbols are more probable than others" (Attneave, p.14). Thus, in the HTHTHTH... sequence, the pairs HT and TH occur with equal probability ($p = 1/2$) while the pairs HH and TT never occur. Thus, the formula for H yields: $H = 1/2 \log 2 + 1/2 \log 2 = 1$. In contrast, H_{max} for the four possible pairs of two different symbols is: $H_{max} = \log 4 = 2$. Then, the relative entropy is .50 and the second order redundancy is given by: $C = 1 - .50 = .50$.

As Attneave points out, the existence of redundancy does not necessarily imply a simple alternation pattern. Thus, in the sequence HHHHHTTTTTTTTHHH..., the pairs HH and TT are more frequent than the pairs TH and HT; therefore, non-zero second-order redundancy exists in this sequence.

If any order of redundancy greater than the first is present, the sequence is more or less patterned; that is, sequential dependencies exist. However, knowing the order and degree of redundancy does not mean that the kind of patterning has been discovered. In order to completely specify a stochastic process, the probability of every possible sequence of N events must be specified, where N is the greatest range of sequential dependencies which exists in the process. There are a number of ways of specifying these transitional probabilities between symbols, and studying the nature of those transitions. For example, Markovian matrices can be used to specify the transitional probabilities and an N dimensional contingency table could be used to study the nature of any order of interaction between expected or empirically obtained frequencies for the various possible sequences. The problem with these methods is that they require a matrix or tabular representation which very quickly becomes unmanageably large as either the number of alternatives or the order of interaction becomes larger than a very small number. For example, the Nth-order transitional probabilities for the 27 alternatives presented by a sequence of the 26 letters of the English alphabet plus the space would require 27^N cells. Obviously, an unmanageable number. In practice, most cells would have zero probabilities. This is true because of such characteristics of the English written language as the fact that there are no sequences of more than three consonants (barring borrowed words from other languages, such as Welsh town names in a feature article in a Sunday supplement), and the small number of characters that can follow a q. However, even using sparse matrix techniques, the problem would probably be too unwieldy to invite direct application of matrix techniques to study higher order Markov properties of the English written language. Application of contingency table techniques is further complicated by the fact that common measures of statistical significance are invalidated by the large number of cells with zero expected frequencies.

However, Attneave gives a method which can be run on modern computers for computing higher order redundancies, although at the time he was writing he considered the computational task impractical for actual use with respect to the English alphabet. A file can be created at the time the redundancies are being calculated which can be used to list the sequences which were actually found in the text and their frequencies. This list should provide information about the nature of the patterning present.

The method given in Attneave is used to calculate $\hat{H}_N$, an Nth-order estimate of H which assumes no redundancy exists at a higher lever. $\hat{H}_N$ is calculated using the formula

$$\hat{H}_N = \hat{H}(N\text{-gram}) - \hat{H}(\overline{N - 1}\text{-gram})$$

where $\hat{H}(N\text{-gram}) = \Sigma\hat{p}(N\text{-gram})\log(1/\hat{p}(N\text{-gram})$.

The quantity $\hat{p}(N\text{-gram})$ is the proportion with which each sequence of N symbols occurred in the text. This inductive formula allows one to start with the first-order estimate $\hat{H}_1$ and work as high as ones inclinations and computer resources will allow. The program we are currently developing will calculate the redundancy estimates up to the 20th-order.

The actual algorithm works backward in a sense. An introductory program uses the text order file that is used by the VOCAB programs to derive the information used by this method for calculating the redundancies. This program creates a file of records, each containing 20 fields. The first record contains the first 20 characters of the text, including the space but excluding punctuation. The second record contains characters 2-21 of the text. Similarly, other records are built with the program acting, in effect, as a sliding template to find all the sequences of 20 characters which exist in the text. This file is then sorted alphabetically over the 20 fields. A second program calculates the actual redundancy estimates. The frequency of occurrence of each 20-character sequence is calculated and divided by the total number of 20-character sequences present in the text to get $\hat{p}$(20-gram). As processing continues, the frequencies of shorter length sequences are also accumulated. To see how this is done, consider processing of a file of three character sequences where the characters are either H or T. The frequency of occurrence of HHH will be calculated first, followed by the frequency for HHT. By combining these two frequencies, we get the frequency for the two character sequence HH. Similarly, the sum of the frequencies for HH and HT yields the frequency for H. Thus, by the time the total file has been processed, we have calculated values for $\hat{H}$(N-gram), N=1,20. We then use the inductive formula given above to derive the redundancy estimates. The value of this method lies in the fact that the character sequence records reside on mass storage files, while very little core storage is requred to do the actual calculations.

Shannon, using a method described in the Attneave article, estimated that English was 75 percent redundant. We hope to calculate the exact estimates directly from the text and compare these results with those obtained by Shannon, who relied on guesses by human subjects to calculate the sequential dependencies existing in English. This method can also be modified and applied to the word length data obtained from SVIA01 in order to derive estimates of patterning found among word lengths.

PRACTICAL APPLICATION OF THE SYSTEM

Four samples of text have been chosen to provide a data base which will be used by current and future programs in the system. These samples are; two different translations from the Russian of Soviet Military Strategy, James Joyce's Portrait of the Artist as a Young Man, and a sample from Joyce's collection of short stories, The Dubliners. The latter sample consists of the entire story "The Sisters" and a large continuous selection from "The Dead".

This section will discuss a few preliminary results obtained by processing the selection from The Dubliners and the first chapter of the RAND Corportation translation of Soviet Military Strategy. We have not been able to run the entire system for the whole data base due to the fact that we do not currently have enough peripheral storage to save all the files for such a large data base. This shortage should be remedied in the next month and we shall then process all of the data base.

A number of interesting comparisons can be made between the sample for Dubliners, (N = 8,285) and the first chapter of Soviet Military Strategy (N = 17,982). First, the content words found in Soviet Military Strategy (heareafter referred to as SMS) are generally longer than those in Dubliners. Thus for 18 segments of SMS (17 segments of 1,000 words each and one segment 982 words long) the mean content word length ranges from 7.20 letters to 7.91 letters. The medians range from 6.82 to 7.71. For the nine segments of the Dubliners (eight segments of 1,000 words each and one segment 285 words long) the means range from 5.22 to 5.92 letters and the medians range from 5.16 to 5.89 letters. Thus the average length of a content word in SMS tends to be about two letters greater than the average length found in Dubliners. In contrast, there is very little difference between function word length for the two samples; in fact, the function words in the Joyce sample have a slightly larger average value. The means for SMS range from 2.76 to 3.06 while the means for the Joyce sample range from 2.81 to

3.14. The respective medians range from 2.61 to 2.87 in contrast to a range from 2.68 to 2.92. It should also be noted that there is less variation in values between segments for the function words than for the content words. Another contrast involving the function-content word distinction is that SMS contains more content words per 1,000 words of text than the Joyce sample. The rate in SMS ranges from 512 to 582 content words per 1,000 text words, while the Joyce segments contain from 368 to 521 content words per 1,000. This may be due to the fact that SMS contains a factual discussion of a rather technical area, while the Joyce sample is a fictional narative which includes passages of dialogue. We intend to explore this distinction further, as well as compare these results with the values obtained from Portrait.

The SMS sample tends to have longer sentences than the Joyce sample. The means range from 22.28 to 32.59 words per sentence in SMS and from 11.71 to 23.37 words in the Joyce sample. The respective medians range from 18.83 to 29.00 in contrast to a range from 7.50 to 22.25. The contrast is even greater if the seventh segment of the Joyce sample is excluded. The largest mean is then 18.27 and the largest median is 14.70.

SMS also tends to have more words per paragraph than Dubliners; however, the number of sentences per paragraph is very similar. The mean words per paragraph for SMS varied from 50.00 to 99.56 and the medians varied from 40.50 to 79.00. The means for Dubliners varied from 25.97 to 162.50 and the medians ranged from 14.50 to 112.50. Again, section seven contained abnormally high values. Excluding section seven's mean of 162.50, the highest mean then becomes 53.42 and the high-

-est median after a similar exclusion is 25.50. Section seven also contains the longest paragraph of either sample, 364 words. We have not yet identified this area of text in the original listing, but it is quite possible that a keypunching error has caused a failure to distinguish the end of a paragraph and that this paragraph is acutally composed of two or more paragraphs. Even so, this section of text would contain much longer paragraphs than found in the rest of the Joyce sample. (Identifying this segment of text is difficult since, for this experimental run, we only have a listing of the original punched cards. Note that when we ran the text through INDEX, we did not use the option which prints out a listing of the text words and their associated indexing information. If we had exercised the print option, identifying the relevant area in the text would be quite simple. One problem with the listing from INDEX is that only one text word is printed per line; and, thus, a 9,000 word sample would lead to printing 9,000 lines in addition to lines for each occurrence of a punctuation mark. We are now writing a utility routine which uses the output file from INDEX to reconstruct the original text and also prints out the indexing information associated with the first word of each line).

In contrast to the situation which exists for words per paragraph, the number of sentences per paragraph is very similar for the two samples. The means for SMS range from 1.71 to 3.56 and the medians from 1.33 to 3.00. For Dubliners the means range from 1.96 to 3.32 (the mean for section seven is 7.00) and the medians range from 1.34 to 2.29 (the median for section seven is 6.50). The maximum number of sentences found in any paragraph of SMS is eight, while the maximum number found in the Joyce sample is 13.

Thus, a rough discrimination between the Joyce and SMS samples can be made on the basis of such indicators as word and sentence length. A more interesting analysis should result when the programs are run against the entire data base, since we shall then be able to compare the two translations of SMS and also make comparisons between the samples from Dubliners versus Portrait of the Artist. It is to be expected that these indicators will prove to be much less sensitive to differences between works by the same author and in the same genre. However, they are the beginning of a large inventory of stylistic indicators which we hope this system will eventually be able to derive. Among other uses, such an inventory could serve as input to a system of cluster analysis programs in order to obtain a taxonomy of style which is both objectively obtained and readily replicated.

We plan to use the results obtained by the present system to provide the basis for an examination of a variety of clustering techniques. That is, the cluster analysis programs will use the values obtained for a variety of variables from a number of segments taken from different works. On the basis of the values obtained, the programs would group together those segments which were most similar. In the present example, one might expect that such a system of programs could successfully distinguish between the SMS and Joyce segments. However, as more subtle indicators are developed and applied against a more diverse collection of works, it is conceivable that a particular passage written by one author might be more similar to passages written by another author than to other passages written by the same author.

The results described above will also serve as input to programs which handle linear discriminant functions. The common t-test described in most statistics books aids one to decide whether two samples come from the same population or from populations with different means. This test deals with observations on a single normally distributed variable. Linear discriminant functions provide a method of making an analogous test between two or more samples, each individual in which has been measured in respect of several variables. Given primary data consisting of k groups, containing $n_1, n_2, \ldots n_k$ individuals respectively where each individual has been defined in terms of p variables, one could derive a set of discriminant functions of the form $d = a_1x_1 + a_2x_2 + \ldots a_px_p$ where $a_1, a_2, \ldots a_p$ are discriminant coefficients computed so as to minimize the overlap between one group and another (Davies, 1971). Thus, it might prove possible that variables such as median content word length measured for a number of segments from SMS and Joyce, respectively, could be used to build a linear discriminant function which would be able to assign further segments to the proper author.

REFERENCES

Attneave, Fred. Applications of Information Theory to Psychology. Henry Holt and Company, New York, 1959.

Carroll, John B., "Word-frequency Studies and the Lognormal Distribution" in Proceedings of the Conference on Language and Language Behavior. Eric M. Zale, ed. Appleton-Century-Crofts, New York, 1968.

Davies, R.G. Computer Programming in Quantitative Biology. Academic Press, New York, 1971.

Joyce, Frank D. "Toward a Statistical Support Package" in Automated Analysis of Language Style and Structure in Technical and Other Documents. Sally Y. Sedelow, Principal Investigator. University of Kansas, Lawrence, Kansas, 1971.

Joyce, James. DUBLINERS. Compass Books Edition. The Biking Press, Inc., New York, 1965.

Kucera, Henry and W. Nelson Francis. Computational Analysis of Present-day American English. Brown University Press, Providence, Rhode Island, 1967.

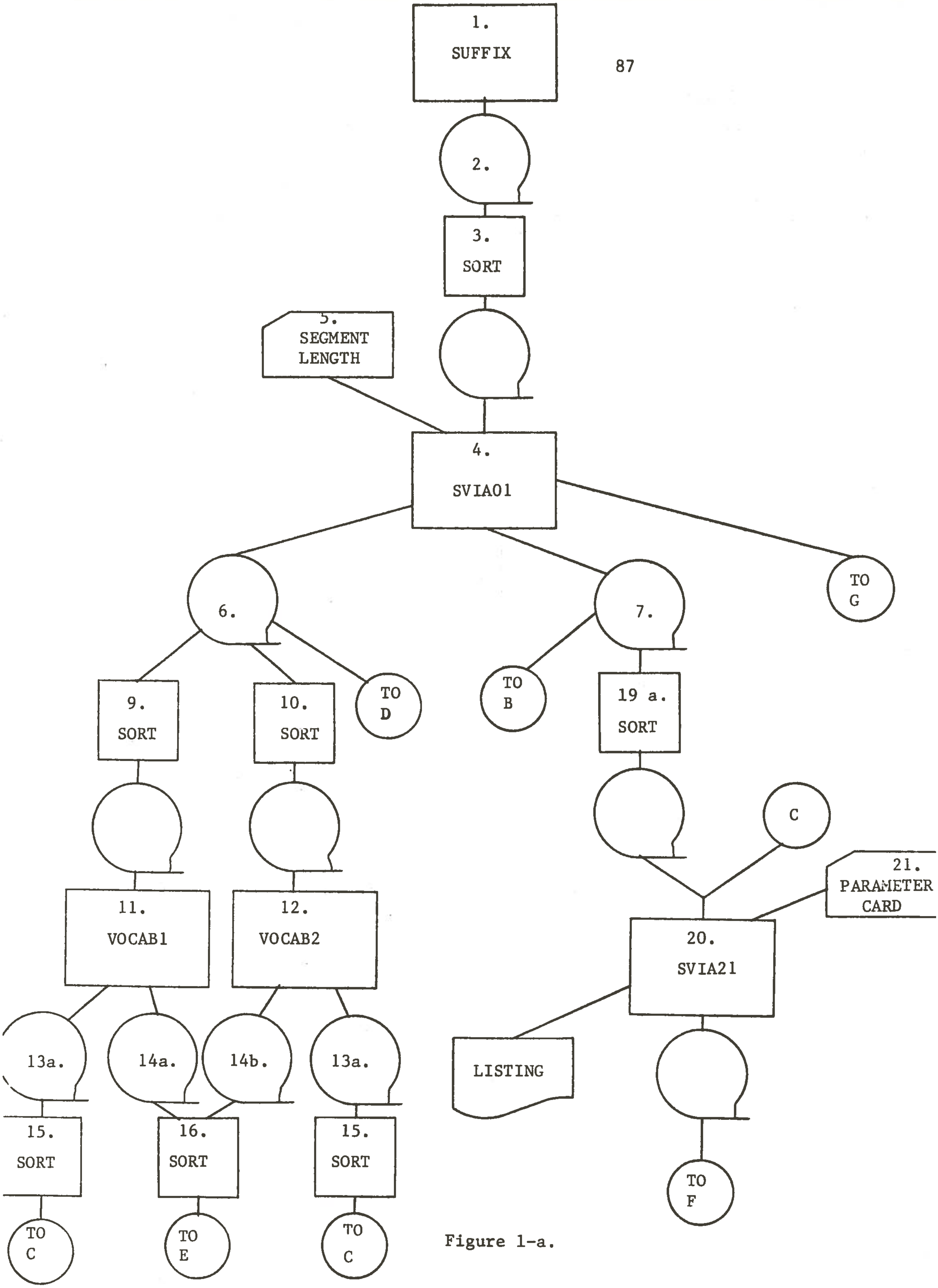
Sedelow, Sally Yeates and Walter A. Sedelow, Jr. "A Preface to Computational Stylistics," in The Computer and Literary Style. Jacob Leed, ed. Kent State University Press, 1966, p.l.

Sedelow, Sally Yeates. Automated Analysis of Language Style and Structure in Technical and Other Documents. Technical Report #1. Contract N00014-70-0357-0001 under Project No. NR348-005. University of Kansas, Lawrence, Kansas, 1971.

Sokolovskii, V. D. Soviet Military Strategy, trans. Dinerstein, Goure, Wolfe. RAND Research Study. Prentice-Hall, Inc., 1963.

Sokolovskii, V. D. Soviet Military Strategy. Frederick A. Praeger, Inc., New York, 1963.

Thorndike, E. L., and I. Lorge. The Teacher's Word Book of 30,000 Words. Bureau of Publications, Teachers College, Columbia University, New York, 1944.



1.
SUFFIX
2.
3.
SORT
5.
SEGMENT
LENGTH
4.
SVIA01
TO
G
6.
7.
9.
SORT
10.
SORT
TO
D
TO
B
19 a.
SORT
C
21.
PARAMETER
CARD
11.
VOCAB1
12.
VOCAB2
20.
SVIA21
13a.
14a.
14b.
13a.
LISTING
15.
SORT
16.
SORT
15.
SORT
TO
F
TO
C
TO
E
TO
C

Figure 1-a.

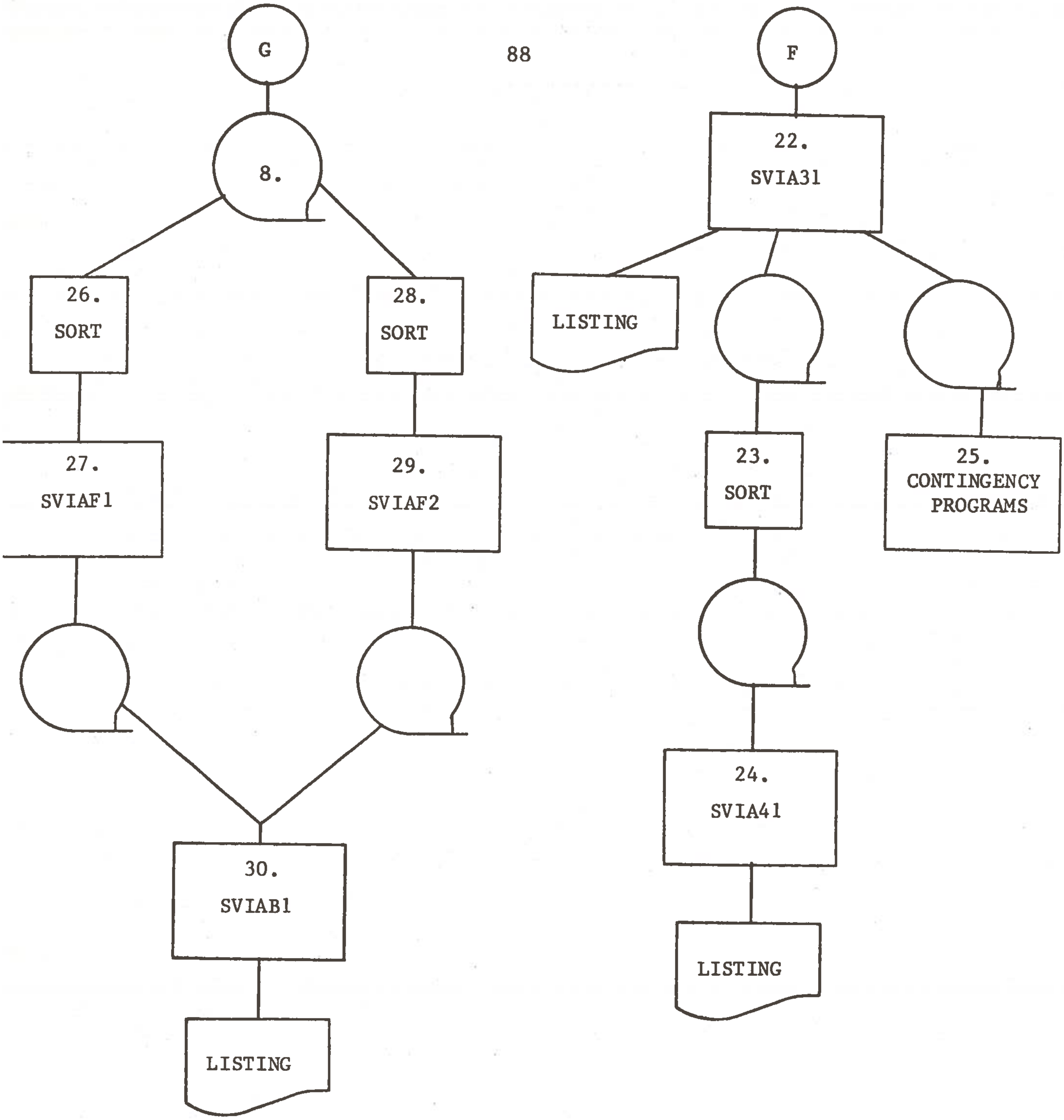

G
8.
26.
SORT
27.
SVIAF1
28.
SORT
29.
SVIAF2
30.
SVIAB1
LISTING
F
22.
SVIA31
LISTING
23.
SORT
24.
SVIA41
LISTING
25.
CONTINGENCY
PROGRAMS

Figure 1-b.

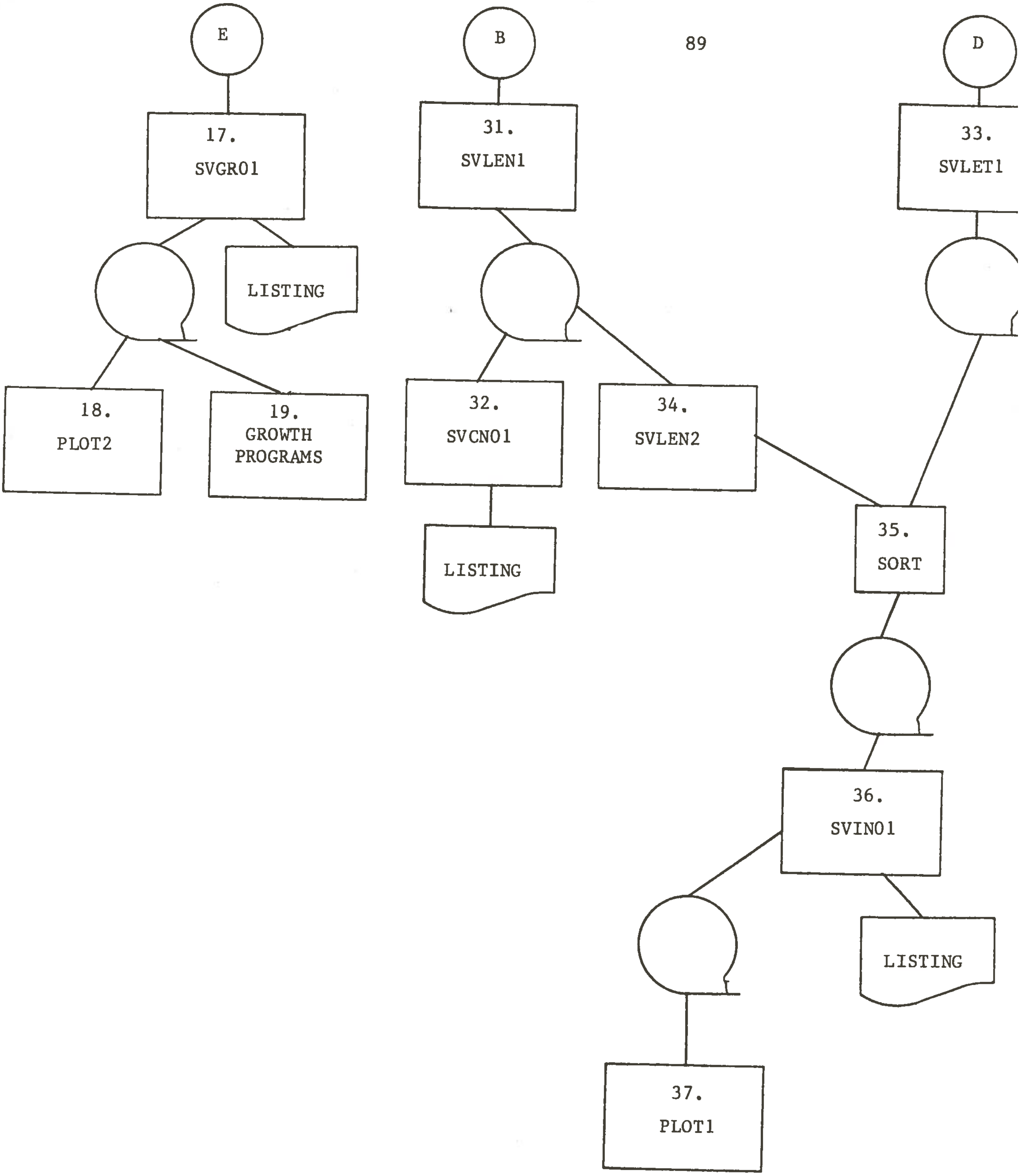

E
17.
SVGR01
LISTING
18.
PLOT2
19.
GROWTH
PROGRAMS
B
31.
SVLEN1
32.
SVCN01
LISTING
34.
SVLEN2
D
33.
SVLET1
35.
SORT
36.
SVIN01
LISTING
37.
PLOT1

Figure 1-c.

KEY TO FIGURES 1-a, 1-b, 1-c

1. SUFFIX - Identifies legal suffixes.

2. TEXT FILE - In alphabetical order.

3. SORT - Sort into linear order (ascending on linear word number field).

4. SVIA01 - Derives basic data.

5. PARAMETER CARD - Contains number of words per segment. Integer in columns 1-5.

6. TEXT FILE - Text file with newly calculated linear word and word in sentence information.

7. FREQUENCY FILE - Basic frequency data.

8. FUNCTION WORD LOCATION - Contains location in sentence and sentence length information for each function word.

9. SORT - Sort the text file from SVIA01 into matchcount order (primary - ascending on matchcount field, secondary - word field).

10. SORT - Sort SVIA01 text file into alphabetical order (primary-ascending on word field, secondary - ascending on prefix field).

11. VOCAB1 - Derives vocabulary growth data and word frequency data and word frequency data based on matchcount (root group).

12. VOCAB2 - Derives vocabulary growth data and word frequency data based on spelling of the word.

13a & 13b. WORD FREQUENCY FILES - These files contain the frequency of occurence data derived by VOCAB1 and VOCAB2, respectively.

14a & 14b. VOCABULARY GROWTH FILE - These files contain the data about increase of vocabulary with text growth.

15. SORT - Sort file 13a or 13b. (primary - ascending on type field, secondary - ascending on segment field, tertiary - ascending

on frequency field). The file created by this sort will be used as input to program SVIA21 (see 20, below).

16. SORT - Sort file 14a or 14b. (primary - ascending on type field, secondary - ascending on segment field).

17. SVGR01 - Calculates increase in vocabulary types with increase in text length. Prints out a listing and puts out cumulative distribution records.

18. PLOT2 - This program, which does not yet exist, will produce a tape to be used on a plotter to display growth curves.

19. GROWTH PROGRAMS - This box represents a group of programs currently being developed. These programs will be used to compare the obtained growth distributions to distributions obtained using a variety of theoretical models for vocabulary growth (lognormal, Zipf, logarithmic type-token ratio, etc.).

19a. SORT - Sorts the frequency file created by program SVIA01 (primary field - type, secondary field - segment, tertiary field-value).

20. SVIA21 - Builds and lists frequency distributions and calculates basic statistics describing those distributions (see Fig. 2).

21. PARAMETER CARD - Contains a "0" in column 1 if frequency distributions are to be combined into frequency distributions for large segments (input file is from SVIA01), "1" in column 1 if distributions are not to be combined (input file is from VOCAB1 or VOCAB2).

22. SVIA31 - Uses summarized frequency records to build frequency distributions. Lists results of summarizing statistics for routine SVIA41. Outputs the frequency distribution in a form

suitable for use by programs to contingency table analysis.

23. SORT - Sorts the file containing the various measures obtained by SVIA31 (primary - type field, secondary - segment size).

24. SVIA41 - Reads in the values for the statistical measures calculated by SVIA31. Treats these measures themselves as variables and calculates summarizing statistics (See Figure 4).

25. CONTINGENCY TABLE PROGRAMS - This box represents a subsystem which will carry out a variety of analyses on contingency tables. These programs are currently under development. (See also 32, below).

26. SORT - Sorts the function word location file, which was produced by program SVIA01, into the order required by program SVIAF1 (primary - segment field, secondary - location field, tertiary - sentence length).

27. SVIAF1 - Builds triangular contingency tables for each segment showing function word location (rows) against sentence length (columns). These tables are listed and written out to a file for use by program SVIAB1 (See 30, below).

28. SORT - Sorts the function word location file, which was produced by program SVIA01, into the order required by program SVIAF2 (primary field - location, secondary field - sentence length).

29. SVIAF2 - Builds triangular contingency table for the entire text showing function word location (rows) against sentence length (columns). This table is listed and written out to a file for use by program SVIAB1.

30. SVIAB1 - Tests the function word location tables for a significant "birth order" affect, that is, are function words more likely

to occur in some positions in a sentence than in other positions. This program is currently under development.

31. SVLEN1 - Uses the letters per word records created by program SVIA01 to create a file of word length pairs. The first member of the pair is the length of a text word and the second member is the length of the immediately following text word. The second member of the pair then becomes the first member of the following pair.

32. SVCNO1 - Uses the word length pairs to study the sequential dependencies among word lengths. Lists the contingency table obtained with cell (i,j) representing the frequency with which a word of length i was immediately followed by a word of length j, lists the corresponding table of expected frequencies under the null hypothesis of no sequential dependencies, and lists the corresponding table showing the contribution of each cell to the total chi-square score measuring the degree of deviation between the observed values and the expected values. Lists the row and column contributions in descending order based on their contribution.

33. SVLET1 - Acts as a sliding template to create records containing all of the different sequences of letters (and the blank) which occurred in the text. Current sequence length is 20.

34. SVLEN2 - Uses the word length pairs created by program SVLEN1 to create records for longer sequences. Current sequence length is 20.

35. SORT - Sorts the sequences produced by SVLET1 or SVLEN1 into ascending order.

36. SVINO1 - Uses the sequence file to calculate the information contained in sequences of order 0 to a current maximum of order 20.

37. PLOT1 - This program, which does not yet exist, will produce a tape to be used by a plotter to display information contained in sequences of successively higher order.

SOVIET MILITARY STRATEGY--RAND

LETTERS PER CONTENT WORD
SEGMENT 1 SIZE 1000 WORDS

NUMBER LETTERS	FREQ.	CUM. FREQ.
2	6	6
3	30	36
4	29	65
5	50	115
6	63	178
7	86	264
8	102	366
9	62	428
10	55	483
11	29	512
12	11	523
13	6	529
14	4	533
15	5	538
16	1	539
18	3	542

LOW	2
HIGH	18
N	542
MEAN	7.568
VARIANCE	6.898
STD.DEV.	2.626
COEF.VAR.	34.7
QUARTILE DEV.	1.664

PERCENTILES	
1st	2.403
5th	3.203
25th	5.825
50th	7.569
75th	9.153
95th	11.764
99th	15.216

SOVIET MILITARY STRATEGY--RAND

LETTERS PER CONTENT WORD
SEGMENT 2 SIZE 1000 WORDS

NUMBER LETTERS	FREQ.	CUM. FREQ.
2	3	3
3	28	31
4	34	65
5	72	137
6	69	206
7	67	273
8	103	376
9	54	430
10	44	474
11	28	502
12	12	514
13	6	520
14	3	523
15	2	525
18	2	527

LOW	2
HIGH	18
N	527
MEAN	7.319
VARIANCE	6.316
STD.DEV.	2.513
COEF.VAR.	34.3
QUARTILE DEV.	1.715

PERCENTILES	
1st	2.581
5th	3.334
25th	5.427
50th	7.358
75th	8.856
95th	11.452
99th	14.077

FIGURE 2. EXAMPLE OF INFORMATION OBTAINED BY PROGRAM SVIA21

Soviet Military Strategy - RAND

LETTERS PER CONTENT WORD

SEGMENT SIZE 1000 WORDS

								PERCENTILES							
SEGMENT	LOW	HIGH	N	MEAN	VAR.	STDDEV	COEEVAR.	5th	25th	50th	75th	95th	Q.DEV.	SKEW	KURT
1	2	18	542	7.568	6.90	2.63	34.704	3.20	5.83	7.57	9.15	11.76	1.66	-1.81	1.086
2	2	18	527	7.319	6.32	2.51	34.338	3.33	5.43	7.36	8.86	11.45	1.71	-1.84	0.715
3	3	15	539	7.425	5.82	2.41	32.485	3.21	5.60	7.62	9.02	11.62	1.71	-1.98	-0.104
4	2	16	549	7.204	5.69	2.38	33.098	3.44	5.43	7.07	8.81	11.59	1.69	-2.02	-0.039
5	2	18	523	7.459	7.66	2.77	37.095	3.14	5.39	7.36	9.22	12.20	1.91	-1.91	0.322
6	2	18	529	7.276	6.47	2.54	34.965	3.47	5.57	7.11	8.87	11.80	1.65	-1.88	0.660
7	2	18	532	7.297	7.67	2.77	37.947	3.23	5.30	7.11	8.88	12.21	1.79	-1.72	1.292
8	3	18	526	7.291	6.70	2.59	35.504	3.61	5.64	7.09	8.64	11.26	1.50	-1.57	3.126
9	2	18	512	7.537	6.98	2.64	35.060	3.29	5.68	7.57	9.12	12.24	1.72	-1.80	0.753
10	2	18	551	7.688	7.66	2.77	35.999	3.41	5.59	7.71	9.34	12.30	1.87	-1.79	0.776
11	2	18	549	7.907	7.87	2.81	35.486	3.72	5.96	7.62	9.48	13.00	1.76	-1.68	0.775
12	2	18	553	7.503	9.07	3.01	40.143	3.40	5.33	7.30	9.02	12.46	1.85	-1.52	2.313
13	2	18	523	7.520	8.44	2.91	38.634	3.46	5.53	7.18	9.21	12.69	1.84	-1.75	1.330
14	2	18	525	7.263	7.18	2.68	36.885	3.62	5.46	6.82	8.86	12.37	1.70	-1.79	1.328
15	3	15	538	7.333	5.77	2.40	32.765	3.50	5.62	7.29	8.98	11.40	1.68	-2.05	-0.306
16	2	18	556	7.540	6.89	2.62	34.812	3.81	5.70	7.40	9.05	11.86	1.67	-1.72	2.040
17	3	18	582	7.541	8.02	2.83	37.556	3.44	5.54	7.42	9.22	12.23	1.84	-1.77	1.509
18	2	18	546	7.333	6.65	2.58	35.177	3.59	5.39	7.02	9.09	11.58	1.85	-2.05	0.527

FIGURE 3. EXAMPLE OF INFORMATION OBTAINED BY SVIA31. Part 1

LETTERS PER FUNCTION WORD

SEGMENT SIZE 1000 WORDS

SEGMENT	LOW	HIGH	N	MEAN	VAR.	STD.DEV.	COEF.VAR.	5th	25th	50th	75th	95th	Q.DEV.	SKEW	KURT
1	1	10	458.	2.836	1.15	1.07	37.811	1.54	2.07	2.71	3.33	4.77	0.63	-1.38	6.405
2	1	9	473.	2.763	1.07	1.03	37.412	1.54	2.01	2.61	3.25	4.79	0.62	-1.42	6.327
3	1	10	461.	2.850	1.18	1.08	38.065	1.51	2.14	2.78	3.30	4.86	0.58	-1.23	11.534
4	1	10	451.	2.867	1.43	1.20	41.746	1.51	2.02	2.68	3.39	5.07	0.68	-1.31	7.038
5	1	7	477.	2.807	1.09	1.05	37.275	1.52	2.02	2.66	3.34	4.91	0.66	-1.54	1.846
6	1	12	471.	2.960	1.78	1.33	45.021	1.53	2.05	2.72	3.42	5.54	0.68	-1.03	9.488
7	1	9	468.	2.966	1.44	1.20	40.507	1.58	2.10	2.76	3.43	5.24	0.67	-1.16	5.069
8	1	12	474.	3.061	1.94	1.39	45.492	1.45	2.14	2.87	3.55	5.47	0.70	-1.05	8.943
9	1	9	488.	2.949	1.37	1.17	39.669	1.53	2.11	2.79	3.42	5.24	0.66	-1.21	3.127
10	1	8	449.	2.906	1.36	1.17	40.105	1.54	2.05	2.70	3.38	5.25	0.66	-1.21	2.384
11	1	9	451.	2.816	1.30	1.14	40.539	1.53	2.02	2.65	3.28	5.12	0.63	-1.23	5.503
12	1	10	447.	2.834	1.34	1.16	40.798	1.47	2.03	2.69	3.32	5.12	0.64	-1.25	6.871
13	1	10	477.	2.941	1.48	1.22	41.373	1.55	2.08	2.75	3.42	5.23	0.67	-1.19	5.942
14	1	9	475.	2.954	1.41	1.19	40.179	1.53	2.11	2.80	3.45	5.14	0.67	-1.25	4.772
15	1	10	462.	2.987	1.47	1.21	40.614	1.53	2.11	2.81	3.46	5.28	0.67	-1.18	4.040
16	1	7	444.	2.910	1.19	1.09	37.489	1.55	2.11	2.77	3.40	5.03	0.65	-1.34	2.694
17	1	7	418.	2.883	1.24	1.11	38.588	1.55	2.06	2.70	3.34	5.21	0.64	-1.23	2.033
18	1	10	436.	2.885	1.17	1.08	37.420	1.56	2.13	2.76	3.31	5.13	0.59	-1.17	6.027

FIGURE 3. EXAMPLE OF INFORMATION OBTAINED BY SVIA31. Part 2

LETTERS PER CONTENT WORD

SIZE = 1000

STATISTIC	MEAN	VAR.	STD. DEV.	COEF. VAR.	SKEW	KURTOSIS
LOW	2.222	0.183	0.428	19.251	1.461	3.537
HIGH	17.556	1.085	1.042	46.873	-2.098	6.321
N	539.000	272.706	16.514	743.121	0.814	4.687
MEAN	7.445	0.031	0.177	7.949	0.985	4.662
VAR	7.097	0.880	0.938	42.205	0.303	3.055
STDDEV	2.659	0.031	0.175	7.894	0.165	2.922
COEFVR	35.703	4.174	2.043	91.933	0.414	3.343
.05	3.437	0.034	0.185	8.327	0.257	2.915
.25	5.555	0.029	0.171	7.717	0.662	3.840
.50	7.312	0.061	0.248	11.138	-0.117	2.647
.75	9.046	0.043	0.208	9.343	0.168	3.363
.95	12.001	0.231	0.481	21.639	0.317	2.803
QDEV	1.745	0.011	0.103	4.614	-0.374	3.669
SKEW	-1.814	0.023	0.151	6.779	0.096	3.175
KURTOS	1.006	0.761	0.872	39.257	0.777	4.177

SIZE = 2000

STATISTIC	MEAN	VAR.	STD. DEV.	COEF. VAR.	SKEW	KURTOSIS
LOW	2.111	0.111	0.333	15.789	3.000	13.000
HIGH	17.778	0.444	0.667	31.579	-3.000	13.000
N	1077.556	736.028	27.130	1285.097	0.718	3.614
MEAN	7.448	0.017	0.132	6.258	1.121	4.544
VAR	7.107	0.654	0.809	38.304	0.066	4.344
STDDEV	2.662	0.023	0.152	7.204	-0.088	4.324
COEFVR	35.734	2.939	1.714	81.209	-0.397	3.357
.05	3.444	0.021	0.146	6.914	-0.046	2.213
.25	5.554	0.005	0.073	3.438	0.375	2.615
.50	7.311	0.039	0.198	9.369	0.213	3.646
.75	9.049	0.021	0.144	6.844	-0.346	4.182
.95	12.027	0.204	0.451	21.387	0.599	3.442
QDEV	1.747	0.005	0.073	3.438	-0.140	2.859
SKEW	-1.810	0.018	0.134	6.328	0.350	4.115
KURTOS	1.047	0.408	0.639	30.269	-0.184	4.631

FIGURE 4. EXAMPLE OF INFORMATION OBTAINED BY PROGRAM SVIA41.

V. A. New Utility Routines for Program PREFIX

by Frank Joyce

As described in (Sedelow, 1971) program PREFIX identifies legitimate English prefixes, using a table look-up procedure. In employing the look-up procedure, PREFIX uses a file of potential prefixes and a file of clud words. The clud words indicate when a word with initial characters matching a potential prefix is a legitimate prefixed word, rather than just beginning with the same letters.

The routine which was formerly used to build these files required that the cards which contained the prefixes and clud words be so arranged that the prefixes were encountered in alphabetical order. The logic of PREFIX is such that the order of the prefix file must not be changed after it has been built (since the clud word records contain pointers to their corresponding prefixes). In order to ease the burden of building these files, two new routines have been written to build these files. The first routine, PREF1, identifies the prefixes and clud words and creates a file which can be sorted to ensure that the prefixes are in proper sequence. The second routine, PREF2, uses the sorted file to create the actual prefix and clud word files. Following PREF2, the clud word file is then sorted alphabetically.

PREF1

PREF1 is the first of two programs used to build the prefix and clud word files used by program PREFIX. The input file is a deck of cards, or card images, containing the prefixes and their associated clud words. The cards are read in under the 80A1 format specification. The prefix must begin in column one, and be followed by one of the words

"INCLUD" or "EXCLUD", depending on which type of clud word list that prefix requires (the program PREF1 actually only checks for "IN" or "EX"). The prefix must be separated from the clud word type identifier by at least one space. The card containing the prefix should be followed by cards containing its associated clud words, one word per card. The clud words must be enclosed by apostrophes (i.e., 'CLUDWORD') and the opening apostrophe must be in column one. In program PREFIX, the text words will be compared only to the letters in the clud word field and, therefore, the clud word 'ATYPI' will match with both "ATYPICAL" and "ATYPICALLY". If it is desired that the text word match the clud word exactly, then the clud word must be followed by the blank character, indicated as part of the clud word. Thus, 'REDO ' will match with the test word "REDO" but will not match with the text word "REDOUBT". It is not necessary for the prefixes or clud words to be placed in the deck in alphabetical order.

The end of the input file should be marked by placing a card with three exclamation points (!!!) after the last record.

EXAMPLE

```
A INCLUD
'ATYPICAL'
'ASYMMETRIC'
RE EXCLUD
'RED '
'REBEL'
.
.
.
!!!
```

The output file records are 28 computer words long. The first eight computer words contain the prefix, with one letter of the prefix per computer word. The next 18 computer words contain the clud word.

If the record is for the prefix itself, the clud word field will contain blanks. The next word contains a zero if the prefix has an exclude list and a one of the prefix has an include list. The final computer word is a length field. If the record is for the prefix itself, then this field contains the number of letters in the prefix. If the record is for an associated clud word, then the field contains the number of letters in the clud word. The format is represented diagrammatically as follows, with length in computer words (for the Honeywell 635, one computer word contains 36 bits):

8	18	1	1
PREFIX	CLUD WORD	KEY	LENGTH

NOTE: Alphameric data are represented in the BCD code which uses six bits per character. Thus, in the 635's 36 bit word, there is room for up to six alphameric characters in each computer word. In the FORTRAN IV VIA package, the characters are stored one per computer word in order to facilitate character manipulation. The character is stored in the leftmost character position followed by five blanks, just as when a letter is read in under the FORTRAN 1A1 format specification.

This file should be sorted before it is used by program PREF2, which finishes building the prefix and clud word files. The primary field is the prefix field and the secondary field is the clud word field. Thus the records may be sorted alphabetically, treating the first 26 computer words as one field for purposes of the sort. Since the prefix record contains blanks in the clud word field, it will precede its associated clud words in the sorted file. Also, since the clud word records have the prefix in their prefix field, they will immediately follow the prefix record.

FILES

Input records -- FORTRAN logical unit 5 (generally the card reader).

Output file -- Written unformatted, or binary, (i.e., the format of the records is not altered) to FORTRAN logical unit 20.

The input file will probably be cards. The output file should be written to magnetic tape or disc, and is only an intermediate file.

ERROR MESSAGE

The following error messages may be written out to FORTRAN logical unit 6 (the line printer):

1. CLUD WORD, XXXXXXXXXXXXXX, IS THE RESULT OF TRUNCATION.
   INPUT RECORD WAS 'XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX'

   Probable cause: The indicated card contained a clud word which contained too many characters (currently 18). The first 18 characters were used and processing continued.

2. FIRST COLUMN WAS BLANK. RECORD REJECTED.
   INPUT RECORD WAS ' XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX'

   Probable cause: The indicated card did not contain a character in column one. The card was rejected and processing continued.

3. ILLEGAL CLUD WORD CARD. RECORD REJECTED.
   INPUT RECORD WAS 'XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX'

   Probable cause: The indicated card contained an apostrophe in column one and either an apostrophe or blank in column two. The card was rejected and processing continued.

4. ILLEGAL LIST TYPE ON PREFIX CARD. RECORD REJECTED.
   INPUT RECORD WAS 'XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX'

Probable cause: The indicated card should contain a prefix; but was not followed by either the word "INCLUD" or the word "EXCLUD". This message could also result if a clud word is not enclosed by apostrophes. The card was rejected and processing continued.

5. PREFIX, XXXXXXXX, IS THE RESULT OF TRUNCATION.
   INPUT RECORD WAS 'XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX'

   Probable cause: The indicated card contained a clud word which contained too many characters (currently 8). The first eight characters were used and processing continued. This message could also arise if a clud word was not enclosed by apostrophes.

PREF2

Program PREF2 accepts the sorted file created by PREF1 and builds the prefix and clud word files used by program PREFIX. The input file is read in binary (i.e., no change is made to the format of the input records) from logical unit 20. This file should have been written to magnetic tape or disc by program PREF1 and sorted into alphabetical order as described in the section dealing with PREF1.

PREF2 writes the file of prefixes in binary to logical unit 12 and the file of clud words is written to logical unit 10. Both files should be written to magnetic tape or disc for use by program PREFIX. The prefix file records each contain 10 computer words. The first word contains a zero if the prefix has an associated exclusion list and a one if it has an inclusion list. The second computer word contains the number of letters in the prefix. The last eight words contain the prefix, one letter per computer word. The clud word file records each contain 20 computer words. The first word is a pointer to the associated prefix. The second computer word contains the number of letters in the clud word, one letter per computer word.

The clud word file should be sorted into alphabetical order on the word field itself (the last 18 computer words) before it is used by program PREFIX. This is necessary since the clud words were read in with the prefixes. Thus, "atypical" with a prefix of "a", will come before any clud word associated with the prefix "ab". The sort will put the clud word file into alphabetical order.

NEW I/O MODULES

The programs that use files of text words have been modified so that the text word file is read in or written out by a separate subroutine (I/O modules). These subroutines are called in place of directly reading or writing the record in a main routine. These subroutines are INDWRT, which writes out the 36-word records that INDEX and PREFIX produce; INDRD, which reads in 36-computer word records for INDEX and SUFFIX; SUFRD, which reads in 39-computer word records; and SUFWRT, which writes out 39-computer word records (such as those produced by SUFFIX).

The subroutines are called with two parameters. The first parameter is the name of the array that contains the record to be read or written. The second parameter is the FORTRAN logical unit number for the file. Thus, the following example illustrates the calling sequence for INDRD if the record is to be read from FORTRAN logical unit 20:

```
CALLING PROGRAM
DIMENSION INREC(36)
.
.
.
```

```
CALL INREC INDRD(INREC,20)
.
.
.
END

SUBROUTINE INDRD(INREC,INFILE)
DIMENSION INREC(36)
READ (INFILE) INREC
RETURN
END
```

One of the important reasons for making the change was so that the actual text record files could be compacted into less than the 36- or 39-computer word records for transmission between programs in the VIA system. The I/O modules can then be used to compact or expand the records before they are transmitted to the VIA routines. The need for this change became clear when we began to process very large text files. For example, the text file produced by program INDEX in processing the entire text for either translation of Soviet Military Strategy would require three 2400 foot computer tapes recorded at 800 BPI to contain the 36-word records. By compacting these records into six computer words, less than half a tape is required to contain the file. The resulting increase in efficiency of such data processing tasks as sorting more than pays for the time required to compact and expand the records.

We were able to accomplish this compaction since none of the fields in the records used by VIA actually require the entire 36 bits of a Honeywell 635 word. By compacting several fields into the same computer word, we were able to obtain the smaller records. The compacting and expanding routines were written in assembly language to allow efficient processing at the bit level.

V. B. Additions and Errata for Technical Report #1,

Automated Analysis of Language Style and Structure

in Technical and Other Documents,

Sally Y. Sedelow, *et al.*, September, 1971.

by Frank Joyce

Page 122. "29-WORD" (line 14) should be "39-WORD"

Page 123. matchcounts of 99999 (line 5) and 99998 (line 6) should be reversed.

Page 53. formula in footnote 1 should read

$$m_r = \frac{\Sigma f(X - \overline{X})^r}{N}$$

Page 101. Add (as insert between first and second lines on Page 101): Calling sequence: CALL BRKUP (NDEST, NSQURC, NOCHAR) where NDEST is the first word of the destination (unpacked) string, NSQURC is the first word of the source (packed) string, and NOCHAR contains the number of characters to be transferred. EXAMPLE: (in this example assume the following dimension statement: DIMENSION NDEST (18), NSOURC (18)).

VI. Professional Activities of Project Personnel

Sally Y. Sedelow

Publications:

The Computer and Language Research: A Study of the Concept of a National Center/Network for Computational Research on Language, University of Kansas. (October, 1972) Co-Author with Walter A. Sedelow, Jr.

"Language Analysis in the Humanities," Communications of the ACM, Vol. 15, No. 7, July, 1972, pp. 644-647.

"The Dartmouth Conference: Humanities," Interface, SIGCUE, ACM, Vol. 5, No. 5, October, 1971, pp. 245-246.

Automated Analysis of Language Style and Structure in Technical and Other Documents, Technical Report No. 1, Contract N00014-70-A-0357-0001, Office of Naval Research, University of Kansas, September, 1971. DDC #735-134.

Papers/Seminars/Addresses/etc.:

"Report on the Study of a Center/Network for Computational Research on Language," Annual Meeting, Association for Computational Linguistics, Chapel Hill, N.C., July, 1972.

"Software, National Center for Computer Based Language Research," Conference on Validation and Distribution of Computer Software, University of Colorado, March, 1972.

Activities:

Co-Editor, Computer Studies in the Humanities and Verbal Behavior, 1966--.

Co-Principal Investigator, NSF Study re Possible National Center/ Network for Computational Research on Language, 1971-72.

Advisory Committee for Computing Activities, National Science Foundation, 1972--.

Committee on Information Technology, American Council of Learned Societies, 1970--.

Field Reader of Proposals, U. S. Department of Health, Education, and Welfare, 1966--.

Proposal Evaluation, Canada Council, 1968--.

Proposal Evaluation, National Endowment for the Humanities, 1969--.

Proposal Evaluation, Special Projects Program, NSF, 1970--.

Faculty Senate Committee on Scholarly Publication, Kansas University, 1971--.

Reviewer of papers for FJCC, 1968-- and SJCC, 1969--.

Invited Participant, Conference on Validation and Distribution of Computer Software, University of Colorado, March 30-31, 1972.

Consultant, Institute for Educational Computing, Claremont Colleges, January, 1972.

Secretary, Computer Applications Section, Midwest Modern Language Association, 1971-72.

Session Chairman, Conference on Research Trends in Computational Linguistics, Washington D.C., March, 1972.

Martin Dillon

Publications:

"Heuristic Selection of Advanced Bases for a Class of Large Linear Programming Models," accepted for publication by the Journal of Operations Research Society of America.

"The Quantitative Analysis of Language: Preliminary Considerations," accepted for publication by Computer Studies in the Humanities and Verbal Behavior.

"Models of Thesauri and Their Applications," with David J. Wagner in Automated Analysis of Language Style and Structure, 1970-1971, Lawrence, Kansas: University of Kansas, 1971.

"Preliminary Study of the United Nations Fund for Population Activities Decision Information System." Contract 2-20-565, February 1972. Chapel Hill, N.C.: Carolina Population Center, University of North Carolina at Chapel Hill.

Principles of Operations Research by Harvey M. Wagner, (Englewood Cliffs, N.F.: Prentice-Hall, 1969) in Information, Storage and Retrieval, Vol. 7, pp. 147-148.

Style and Vocabulary: Numerical Studies, by C. B. Williams (New York: Hafner Publishing Co., 1970) in Technometircs, Vol. 13, no. 3, pp. 708-709.

Activities:

"Automated Analysis of Interdisciplinary Discourse Barriers," funded by National Aeronautics and Space Administration, Principal Investigator.

"Analysis of Automated Information Systems for Population Planning," funded by Agency for International Development, Principal Investigator.

"Information, Values and Urban Policy Formation," funded by National Institute of Mental Health, Co-Investigator.

Information and Policy Sciences, Panel Chairman.

Southeastern Regional Division Society for General Systems Research, 1971 conference.

United Nations Fund for Population Activities.

Walter A. Sedelow, Jr.

Publications:

The Computer and Language Research: A Study of the Concept of a National Center/Network for Computational Research on Language, University of Kansas. (1972). Co-author with Sally Y. Sedelow.

"Models, Computing, and Stylistics," in Current Trends in Stylistics, B. B. Kachru and H. F. W. Stahlke, ed., Linguistic Research, Inc. (Forthcoming, 1972). Co-author with Sally Y. Sedelow.

"Remarks re Computer Art," in Proceedings 7th National Sculpture Conference (1972). (Forthcoming).

"A Perspective for World Shakespeare Congress Efforts as to Computers and New Methodologies," Computer Studies in the Humanities and Verbal Behavior. (Forthcoming).

Papers/Seminars/Addresses/etc.:

"Computer Science and the General University Curriculum," lecture at Washington State University, Pullman, Washington, January, 1972.

"Computers and the Nature of Knowledge," lecture at Illinois Institute of Technology, Chicago, Illinois, February 10, 1972.

Panel Chairman, "Computers and Historians," History Graduate Students' Club, University of Kansas, February, 1972.

"The Problem of Sociology," Sociology Graduate Students' Forum, University of Kansas, March 28, 1972.

"The Computer and the Integrity of the Individual," Sigma Xi Lecture, University of Kansas, March 29, 1972.

Participant, Invitational Conference (NSF) on Validation and Distribution of Computer Software, University of Colorado, March 31, 1972.

Panelist, "Computer Art: Hardware and Software vs. Aesthetics," 7th National Sculpture Conference, University of Kansas, April 28, 1972.

Activities:

Principal Investigator, National Science Foundation Study Grant re Possible Center/Network for Computational Research on Language (Ce/NCoReL), Study Grant GJ-28599, 1971-72.

Member, Advisory Panel, "Alternatives in the Management and Financing of Computing in Universities" (NSF), Denver Research Institute, University of Denver, 1972-73.

Board of Editors, Computer Studies in the Humanities and Verbal Behavior, 1966--.

Referee, Social Forces, 1971-72.

Consultant, Institute for Educational Computing, Claremont Colleges, Claremont, California, January, 1972.

Referee, National Endowment for the Humanities, 1971.

Referee, Technical Papers, IFIP Congress 71, 1971.

Consultant, Department of Computer Science, Illinois Institute of Technology, Chicago, Illinois, February, 1972.

Member, Advisory Council, Computer Studies Institute, International Academy at Santa Barbara, California, 1972--.

Member, University Senate Committee on Lectures and Convocations, University of Kansas, 1971-72.

Member, Subcommittee on Student Information of Affirmative Action Board, University of Kansas, 1971-72.

Juliet Shaffer

Publications:

"An Exact Multiple Comparisons Test for a Multinomial Distribution," British Journal of Mathematical and Statistical Psychology, 1971, Vol. 24, pp. 267-272. (November, 1971).

"Directional Statistical Hypotheses and Comparisons Among Means," Psychological Bulletin, 1972, Vol. 77, pp. 195-197. (March, 1972).

Papers/Seminars/Addresses/etc.:

Talk on analysis of multidimensional contingency tables and possible applications in linguistics to a Linguistics Seminar, University of Kansas, 1972.

Activities:

Statistical Editor, Computer Studies in the Humanities and Verbal Behavior.

Referee, Behavior Research Methods and Instrumentation.

Herbert Harris

Activities:

M.A. Thesis: Morphological Parsing Using a Complete Root Dictionary

Frank Joyce

Activities:

Vice-President, Student ACM Chapter

Thomas Kosakowski

Activities:

President, Student ACM Chapter

Sam Warfel

Activities:

Passed comprehensive examination for Ph.D. in Linguistics

VII. Appendix: PREFIX, PREF1 and PREF2 Listings

```
CPREFIX      PROGRAM TO FIND LEGAL PREFIXES
      DIMENSION ICLUD(18),INWORD(18),ISAVWD(18),INPFIX(8),IPRFIX(8),
     1  MPREFX(40,8),KEYTAB(40),IPRLEN(40),INDEXN(8)
      DIMENSION INREC(36)
      INTEGER HIVALU
      EQUIVALENCE (INREC(1),INDEXN(1)),(INREC(9),IPRFXL)
      EQUIVALENCE (INREC(10),INWRDL),(INREC(11),INPFIX(1))
      EQUIVALENCE (INREC(19),INWORD(1))
      DATA IFOUR,IBLANK,(ISAVWD(I),I=1,18),NOCHAR,NCHARP/4,19*6H
     1  18,8/,JEOFSW/0/
      DATA IZERO1/1H0/
      DATA HIVALU/07777777777777/
C
C
      REWIND 10
      REWIND 12
      CALL FLGEOF(10,ICEOF)
      CALL FLGEOF(12,IPREOF)
      CALL FLGEOF(15,ITXEOF)
C
C        READ FIRST CLUD WORD
C
      READ(10)        IPTR,ICLDLN,ICLUD
      IF(ICEOF.EQ.1) GO TO 49000
C
C        READ FIRST PREFIX
C
      READ (12)       KEY,INPRFL,IPRFIX
      IF(IPREOF.EQ.1) GO TO 55000
      LETSAV = IZERO
C
C        READ TEXT RECORD
C
      CALL INDRD(INREC,15)
      IF(INDEXN(1).EQ.HIVALU) GO TO 60000
C        IF TEXT WORD HAS LESS THAN FOUR LETTERS, IT DOES NOT HAVE A
C            PREFIX
13000 IF(INWRDL.LT.IFOUR) GO TO 35000
C
C        DOES TEXT WORD BEGIN WITH SAME LETTER AS PREVIOUS TEXT WORD
C
      LTEST = LCOMP(INWORD(1),LETSAV)
      IF(LTEST) 71000,1900n,14500
C
C        SECTION TO READ IN PREFIXES WHICH BEGIN WITH SAME LETTER AS
C           TEXT WORD
C
14500 IF(IPREOF.EQ.1) GO TO 54500
      IPRTAB = 0
15000 LTEST = LCOMP(INWORD(1),IPRFIX(1))
      IF(LTEST) 18000,1550n,17000
15500 IPRTAB = IPRTAB + 1
      DO 16000 J = 1,NCHARP
      MPREFX(IPRTAB,J) = IPRFIX(J)
16000 CONTINUE
      KEYTAB(IPRTAB) = KEY
      IPRLEN(IPRTAB) = INPRFL
17000 READ (12)       KEY,INPRFL,IPRFIX
      IF(IPREOF.EQ.1) GO TO 18000
      GO TO 15000
```

```
C
C         PREFIXES HAVE BEEN READ IN
C
18000 LETSAV = INWORD(1)
C
C         COMPARE TEXT WORD WITH CLUD WORD
C
19000 IF(JEOFSW.EQ.1) GO TO 27000
      DO 21000 J=1,ICLDLN
      LTEST = LCOMP(INWORD(J),ICLUD(J))
      IF(LTEST) 27000,20000,22000
20000 IF(ICLUD(J).EQ.IBLANK) GO TO 23000
21000 CONTINUE
      GO TO 23000
C
C         READ NEW CLUD WORD
C
22000 READ (10)       IPTR,ICLDLN,ICLUD
      IF(ICEOF.EQ.1) GO TO 50000
      GO TO 19000
C
C         CLUD WORD MATCHED THE TEXT WORD
C         IF KEYTAB(IPTR) = 0, TEXT WORD IS EXCLUDED
C
23000 IF(KEYTAB(IPTR).EQ.0) GO TO 35000
C
C         LIST IS INCLUSION LIST, TEXT WORD HAS PREFIX
C
      ILEN = IPRLEN(IPTR)
      NPTR = IPTR
      GO TO 33000
C
C         TEXT WORD WAS NOT AMONG THE CLUD WORDS
C
27000 NPTR = 0
28000 NPTR = NPTR + 1
C
C         SEARCH PREFIX TABLE FOR PREFIX THAT MATCHES INITIAL LETTERS
C            OF TEXT WORD
C
      IF(NPTR.GT.IPRTAB) GO TO 35000
C         PREFIX SHOULD HAVE EXCLUSION LIST SINCE THE WORD WAS NOT ON
C            THE CLUD LIST
      IF(KEYTAB(NPTR).EQ.1) GO TO 28000
C         COMPARE PREFIX TO TEXT WORD
      ILEN = IPRLEN(NPTR)
      DO 29000 J=1,ILEN
      LTEST = LCOMP(INWORD(J),MPREFX(NPTR,J))
      IF(LTEST) 35000,29000,28000
29000 CONTINUE
      IF(ILEN.EQ.INWRDL) GO TO 28000
C
C         TEXT WORD HAS A LEGAL PREFIX
C
33000 N1 = IPRLEN(NPTR)
      IOWRDL = INWRDL - N1
      N2 = N1 + 1
33100 INREC(9) = IPRLEN(NPTR)
      INREC(10) = IOWRDL
      DO 33020 J=1,NCHARP
```

```
      INPFIX(J) = MPREFX(NPTR,J)
33020 CONTINUE
      M = 0
      DO 33040 J=N2,NOCHAR
      M = M + 1
      INWORD(M) = INWORD(J)
33040 CONTINUE
      M = M + 1
      DO 33060 J=M,18
      INWORD(J) = IBLANK
33060 CONTINUE
      CALL INDWRT(INREC,20)
      DO 33200 J=1,NOCHAR
      ISAVWD(J) = INWORD(J)
      IF(INWORD(J)-IBLANK) 33200,33500,33200
33200 CONTINUE
33500 CALL INDRD(INREC,15)
      IF(INDEXN(1).EQ.HIVALU) GO TO 60000
      DO 34000 J=1,NOCHAR
      LTEST = LCOMP(INWORD(J),ISAVWD(J))
      IF(LTEST) 13000,33700,13000
33700 IF(INWORD(J)-IBLANK) 34000,33100,34000
34000 CONTINUE
      GO TO 33100
C
C       TEXT WORD DOES NOT HAVE A LEGAL PREFIX
C
35000 CALL INDWRT(INREC,20)
      DO 35200 J=1,NOCHAR
      ISAVWD(J) = INWORD(J)
      IF(INWORD(J)-IBLANK) 35200,35500,35200
35200 CONTINUE
35500 CALL INDRD(INREC,15)
      IF(INDEXN(1).EQ.HIVALU) GO TO 60000
      DO 36000 J=1,NOCHAR
      LTEST = LCOMP(INWORD(J),ISAVWD(J))
      IF(LTEST) 13000,35700,13000
35700 IF(INWORD(J)-IBLANK) 36000,35000,36000
36000 CONTINUE
      GO TO 35000
C
C
C         ERROR MESSAGES
C
49000 WRITE (6,49100)
49100 FORMAT (1X,53H* * *  NO INCLUSION-EXCLUSION FILE PRESENT  * * *
     1*)
50000 JEOFSW = 1
      GO TO 27000
54000 CALL INDRD(INREC,15)
      IF(ITXEOF.EQ.1) GO TO 60000
54500 CALL INDWRT(INREC,20)
      GO TO 54000
55000 WRITE (6,55100)
55100 FORMAT (1X,54H* * *  NO PREFIX FILE PRESENT.  END OF JOB.  * * *
     1 *)
      STOP
60000 WRITE (6,61000)
61000 FORMAT (20X,19HNORMAL TERMINATION.)
      DO 62000 I=1,36
```

```
      INREC(I) = HIVALU
62000 CONTINUE
      CALL INDWRT(INREC,20)
      STOP
71000 WRITE (6,71100)
71100 FORMAT (1X,57H* * * *  TEXT WORD OUT OF SEQUENCE.  END OF JOB.  *
     1* * *)
        WRITE (6,50) LETSAV,ICLUD,IPRFIX,INWORD
50       FORMAT (1X,45A1)
      STOP
      END
```

```
CPREF2          ROUTINE TO BUILD PREFIX FILES
C
      INTEGER PREFIX(8),CLUDWD(18),SAVPRE(8)
      INTEGER HIVALU,BLANK
      DATA HIVALU,LETSAV,BLANK/07777777777777,2*1H /
      DATA NCHARP,IPRCTR,ICLCTR/8,2*0/
C
C               READ NEW PREFIX
C
      READ (20) PREFIX,CLUDWD,KEY,LENGTH
      IF(PREFIX(1).EQ.HIVALU) GO TO 3500
      DO 2000 I=1,NCHARP
      SAVPRE(I) = PREFIX(I)
 2000 CONTINUE
 2100 IF(CLUDWD(1).NE.BLANK) GO TO 4000
      IF(PREFIX(1).EQ.LETSAV) GO TO 2400
      LETSAV = PREFIX(1)
      IPTR = 1
      GO TO 2500
 2400 IPTR = IPTR + 1
 2500 IPRCTR = IPRCTR + 1
      WRITE (12) KEY,LENGTH,PREFIX
      DO 2600 I=1,NCHARP
      SAVPRE(I) = PREFIX(I)
 2600 CONTINUE
C
C               READ NEW RECORD
C
 2650 READ (20) PREFIX,CLUDWD,KEY,LENGTH
      IF(PREFIX(1).EQ.HIVALU) GO TO 3000
      DO 2700 I=1,NCHARP
      IF(PREFIX(I).NE.SAVPRE(I)) GO TO 2100
      IF(PREFIX(I).EQ.BLANK) GO TO 2800
 2700 CONTINUE
 2800 WRITE (10) IPTR,LENGTH,CLUDWD
      ICLCTR = ICLCTR + 1
      GO TO 2650
 3000 WRITE (6,3100) IPRCTR,ICLCTR
 3100 FORMAT (30X,18HNORMAL TERMINATION//30X,16HPREFIX RECORDS--,
     1  I4/30X,19HCLUD WORD RECORDS--,I4)
      WRITE (10) (HIVALU,I=1,20)
      WRITE (12) (HIVALU,I=1,10)
      STOP
 3500 WRITE (6,3600)
 3600 FORMAT (1X,31HNO INPUT FILE.  JOB TERMINATED.)
      STOP
 4000 WRITE (6,4100) PREFIX,CLUDWD,KEY,LENGTH
 4100 FORMAT (1X,39HMISSING PREFIX RECORD.  JOB TERMINATED/5X,
     1  8A1,1X,18A1,2I3)
      STOP
      END
```

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author) University of Kansas Lawrence, Kansas 66044	2a. REPORT SECURITY CLASSIFICATION Unclassified 2b. GROUP

3. REPORT TITLE

Automated Language Analysis

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)

5. AUTHOR(S) (First name, middle initial, last name)

Sedelow, Sally Yeates

6. REPORT DATE 1 September 1972	7a. TOTAL NO. OF PAGES 124	7b. NO. OF REFS

8a. CONTRACT OR GRANT NO. N00014-70-A-0357-0001 b. PROJECT NO. c. d.	9a. ORIGINATOR'S REPORT NUMBER(S) 9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)

10. DISTRIBUTION STATEMENT

Distribution of this report is unlimited.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY

13. ABSTRACT

A report on the emphases in the Project for Automated Language Analysis for the period September 1, 1971 - August 31, 1972. Efforts related to the VIA content-analysis package included (1) "final" proofing and correcting of the Roget's International Thesaurus tapes, both (2a) pragmatic and (2b) theoretical approaches to problems associated with the recognition of prefixes, (3) development of programs for an interactive version of the ring-structure VIA, and (4) implementation of a FORTRAN list-structure VIA for the Honeywell 635. Also, work continued on a statistical support package.

DD FORM 1473 1 NOV 65 (PAGE 1)

S/N 0101-807-6801

Security Classification

Security Classification

14 KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Automated Language Analysis						
Stylistic Analysis						
Thesauri						
Prefixing						
Content Analysis						
Statistical Package						

DD FORM 1 NOV 65 1473 (BACK)
(PAGE 2)

Security Classification