acm

SIGLASH NEWSLETTER

SPECIAL INTEREST GROUP ON
LANGUAGE ANALYSIS AND STUDIES IN THE HUMANITIES

December 1974 Volume 7, Number 5

Random-Accessible Text System for Associative Text Analysis

John B. Smith and Paul W. Schuepp

Pennsylvania State University
University Park, Penn. 16802

Introduction

*R*andom-*A*ccessible-*T*ext *S*ystem for *A*ssociated *T*ext *A*nalysis (RATSATAN) is a set of macro routines designed to be used in PL/I main procedures on IBM equipment; as other manufacturers begin to market PL/I compilers, these routines may possibly be used on other hardware. The routines are intended for natural language analyses but may be used for any data consisting of a sequence of specified symbols or symbol combinations. They relieve the researcher from all data management responsibilities, regardless of the size of the text, and provide a number of germinal language analysis functions that can be combined for a sensitive, highly flexible analysis of a literary work or other natural language text.

RATSATAN is an extension of work reported in earlier papers by Smith. One, entitled "Some Lucubrations and Specifications for a Natural Language Analyzer," characterized the computational problems inherent in natural language analysis, describes a set of fourth level functions that meet those demands, and matches them with a specific hardware architecture, still in the developmental stage.[1] At the heart of that system are the notions of recursive definition of categories (categories of categories of categories...) and associative access to the text (the ability to locate each occurrence of a character string of set of such strings within their full text contexts without expensive, time consuming searching). Smith concluded that the computational requirements of natural language analysts, when codified into a "rough draft" of a programming language, exactly match the characteristics of a hypothetical high order programming language that would grow naturally out of an associative memory and processing machine. When that paper was written it seemed possible that such a machine could be developed by mid-decade; today it is evident that this will not be the case. Rumors coming out of IBM's advanced systems division suggest that the next generation of machines is likely to offer associative processing capabilities (perhaps by cycling the contents of memory rather than by driving memory) but it is unlikely this group of machines will be marketed before the late '70s or '80s.

In a second paper, entitled "RATS: A Middle-Level Text Utility System," Smith described a list processing software system that realized many of the characteristics decribed in the first paper.[2] Intended for the researcher who can program or who has access to programming assistance, RATS provides random access to a text such that each occurrence of a given word can be located directly along with full text context. That is, the program can directly address each occurrence of a given word and then move sequentially from that word to the beginning of the text and to the end of the text. RATS, however, was limited in its original form by the amount of main storage available. Texts too large to be held in main storage could be processed only by relatively elaborate programming techniques involving intermediate storage, multiple passes through the data, and multiple job steps.

RATSATAN resolves most of the technical difficulties of managing large textual data sets and goes one step further than RATS in actually providing germinal language analytic functions that compute many of the measures, collections of words, or contextual strings that constitute much computational language analysis. Below we shall describe the data management routines, first, and then the associative processing functions. This description is intended only as an outline of the basic design of RATSATAN, not as a user's manual. The latter is in preparation and will be available later this fall.

Data Management Facilities

Preliminary *Processing*. Texts to be used with RATSATAN must first be processed by RATS. Textual data is prepared virtually as it would appear on the page except that syntactic marks of punctuation are separated from the preceding word by a blank. Textual data in this format is scanned by RATS and some six index numbers are generated that specify where in the text a specific word occurs. Five of the numbers are hierarchical and, for prose, represent volume, chapter, paragraph, sentence, and word in sentence, respectively. The sixth index number is a linear sequence value that runs cumulatively through the entire text (the first word is numbered one; the second, two; etc. until the last word). RATS then represents the text as a list structure by dividing the records described above into three separate files of data structures connected by pointers. The first file is a dictionary of word types, the frequency of occurrence of each type, and a pointer to the index information stored in file two that describes each occurrence or token of the word type. The second file contains the index information for each word, stored in alphabetical order for the corresponding word. It also contains a pointer to the third file. The third file duplicates in structure the second file; however, it is ordered linearly. (The index information for the first word in the text appears first, the information for the second, second, etc. through the text). For each location, the word that would appear there is indicated by a pointer back to the dictionary. The figure that follows illustrates the RATS structure for a single sentence by tracing the word, *the* through the pointer structure. Input for RATSATAN is a text that has been built into this three-file structure by RATS.

CREATE. The first step in RATSATAN, proper, is the CREATE step. It takes the data structure generated by RATS and loads it into a random access device; currently, it is implemented for an IBM 3330 disk. While files 2 and 3, the alphabetic and linear files of index information, can be accessed through modular arithmetic, a separate search dictionary to be used with the RATS dictionary by the data management routines is created in which the first word on each track is listed. The search dicitonary is passed along with the three data files to the user's analytic program or they may be stored permanently for subsequent analyses.

INPUT MODULE. All data management is handled by an input module, called by the processing routines described below. If a dictionary word is desired, the input module performs a binary search on the search dictionary to determine the appropriate track; if that track is not in core, it reads the entire track into a buffer and then performs a binary search, using an over-laid structure, to locate the appropriate record. Since the alphabetical and linear files are accessed through numerical pointers, the appropriate track and position within the track can be derived directly through modular arithmetic.

BASIC RETRIEVAL ROUTINES. While all RATSATAN routines may call the INPUT MODULE, it is used most closely with three retrieval routines. SEARCH locates the dictionary position of a given word. RETRIEVE WORD, its opposite, identifies the character string form of a word stored at a particular dictionary location. RETRIEVE NUMERICAL returns any one of the specified index values from file 2 or 3 or the frequency or pointer from file 1. As with all RATSATAN routines, one routine may serve as an argument for another allowing then to be used in combination.

Associative Processing Features

The associative processing routines provide through software the illusion that the user has associative memory and associative processing capabilities, that is the user's program can access each occurrence of a particular word in its full text context. Most of the associative processing capabilities are based on a set of categorization features.

Categorization Features. A single function allows the user to identify a collection of word types as a category and to associate it with a name. Thus one may identify, say, all of the words that are proper nouns, or images, or words that constitute a theme. This function is recursive so that previously defined categories may be combined to form still larger categories or previously defined categories can be mixed with additional, individual words. Once defined, categories may be saved, along with their names, on auxilary storage and read in by another function for future analyses. Since a category is just a set, we have provided a UNION function that combines a group of categories and an INTERSECTION function that determines the individual words shared by each of a group of specified categories. Any processing routine that takes as an argument a text word can also be applied to a category or one of the category functions (UNION and INTERSECTION). It is this feature that gives RATSATAN much of its power and generality.

Frequency Count Function. This feature allows the user to compute the frequency of occurrence of a word or category within a specified interval of text. The textual interval may be defined at any hierarchical level (sentence, chapter, etc.) by specifying the lower and upper bounds. When used with the linear number, these bounds may be the actual linear numbers or percentages. With the latter, RATSATAN determines the linear number boundaries for, say, 1% of the text; placed inside a loop, the function can, thus, be used to compute a distribution with 100 cells over the text. The distribution values, in turn, can be displayed graphically or passed to an analytic procedure, such as a factor analysis program or a nonmetric multidimensional scaling program.

Concordance. The concordance routine produces a concordance for the word or category of words specified. Context is defined by the user at any hierarchical or linear level with the number of units preceding and succeeding each occurrence.

Use

All features are meant to be embedded in a PL/I program; thus, the researcher has at his disposal the full flexibility of the programming language as well as the RATSATAN functions. PL/I and RATSATAN features may be combined for flexibility while relieving the user from many of the laborious aspects of programming. Once the user is aware of the protocols and calling sequences of the data management routines, he may expand the basic set of functions to include special purpose functions of his own design. Our own current plans for expansion call for one additional capability that will allow us to define categories of contextual patterns. This function will define a category of all occurrences of, for example, a word from category 1 preceded within five words by a word from category 2 but not a word from category 3 and succeeded by a word from category 4 before the end of the sentence. While this function will be useful in a variety of contexts, one of its applications will be equivalent to the General Inquirer concept of content analysis.

We are currently preparing a detailed user's manual for RATSATAN. All programs and documentation will be available through the Penn State Computation Center after October 1, 1974.

Footnotes

[1] John B. Smith. "Some Lucubrations and Specifications for a Natural Language Analyser," *Computer Studies in the Humanities and Verbal Behavior,* Vol. 4, No. 2 (August, 1973), 91-96.

[2] John B. Smith, "RATS: A Middle-Level Text Utility System," *Computers and the Humanities,* Vol. 6, No. 5 (May, 1972), 277-84.