Natural Language Information Formatting: The Automatic Conversion of Texts to a Structured Data Base

NAOMI SAGER

Linguistic String Project
New York University
New York, New York

1. Introduction

1.1 Language Processing

The field of computerized language processing encompasses a wide range of goals and methodologies, ranging from such theoretical objec-

ISBN 0-12-012 117-4

tives as the modeling of human linguistic behavior (e.g., Schank, 1975; Lindsay and Norman, 1977) and human language acquisition (Reeker, 1976), to such applicational goals as machine translation (Josselson, 1971; Lehmann, 1978), natural language systems for man–machine communication (Waltz, 1977; Winograd, 1972; Thompson and Thompson, 1975) and speech recognition (Hill, 1971; Otten, 1971; Walker, 1975). Recent reviews of the entire field are to be found in Walker (1973) and Damerau (1976).

What unites these varied endeavors is the need to come to grips with the special features of natural language as a communication system. Language is the major medium of both storing and transmitting information. Thus, whether the goal is to transfer that information from one language to another, or to access the stored information in response to user queries, or to model linguistic processes for their own sake, some theory as to how information is carried by language and how common meanings are extracted from different linguistic forms is required.

In addition, it is by now clear that the procedural formulation of linguistic processes, as opposed to the purely descriptive, sets up its own requirements. "Decoding" the linguistic communication (recognition rather than generation) involves questions of strategy: How shall the requisite facts about language be organized for computation? How much of this information should be stored as properties of individual words and how much relegated to grammatical and semantic procedures? How should the burden of processing be divided between syntactic and semantic components? What should the processing algorithm be? And assuming that grammatical facts are well established (but computational linguists have had to reformulate them for use in procedures), the remaining question, how shall the relevant semantic categories be established? These questions and others have helped to define the specialized field of natural language processing and have also led to divergent methods within the field.

The present article approaches these problems from the viewpoint of information science, with applications envisioned primarily in science information retrieval and data base management. It attempts to provide a general solution to the following problem: Given a collection of documents on a particular subject written in English or another natural language, how can computer programs arrange the information contained in the documents so that it can be accessed from different points of view for a variety of informational tasks. This assumes that there are underlying common features in the texts that can be made explicit by formal procedures. For the field of information management the solution to this problem would extend the data bases on which data processing functions

which have already been automated can operate. Presently, these operations require that the data be supplied in structured form, that is, in the form of a table or the equivalent. While natural language "front ends" for such systems have been developed, mainly in the form of question–answering programs (Simmons, 1970; Woods *et al.*, 1972; Petrick, 1975; Plath, 1975; Waltz, 1977), this has left open the question as to whether the data base itself, if recorded in natural language, can also be processed by computer programs.

1.2 Text Analysis

The problem, how to structure free running text so that its content is made accessible for processing, has been under study since the early days of computer science. Some of the first large undertakings in automatic language processing (Harris, 1959; Kuno and Oettinger, 1963; Zwicky *et al.*, 1965; Keyser and Petrick, 1967) had immediate or future goals of this type. However, numerous obstacles prevented the early achievement of such programs. Among these were the lack of an appropriate formalism for natural language computations, the lack of suitable grammars and dictionaries, the lack of mechanisms to treat syntactic ambiguity and to recover implicit elements from the discourse, and the unresolved problem of semantic representation. Fortunately, the decade or more of research on these problems has yielded solutions which, if not complete, nevertheless demonstrate that a text-structuring capability is being created and has already reached the stage of application to certain advanced information processing tasks.

1.3 Natural Language Data Bases

A stimulus to research on text-structuring programs has been the advent of machine readable natural language data bases. With changes in the technology of publishing, large quantities of machine readable text are being created daily. In current practice the machine readable form of the texts is often also destroyed. However, it is not difficult to imagine that agreements will be formulated whereby some portions of this resource will become available in the future as full-text data bases (Lancaster, 1977).

More immediately, the wide use of computers in file management and the development of magnetic and optical devices for capturing written material in machine readable form has raised the question as to whether computer techniques can be developed for accessing and processing the information in large natural language files, such as are now found in quantity in medicine, government, business, and scientific research. If

routine clerical operations could be performed automatically on the original natural language form of the information, it could become worthwhile for institutions to have their document files stored in machine readable form.

In some cases, natural language files have already been put into machine readable form, not for purposes of processing the contents by computer, but for convenience of access and storage. The existence of the files in computerized form then raises the question as to whether further data processing operations can be carried out on them without having to transfer the information manually into preset formats. An example of this process is the case of a hospital which computerizes its patient files for quick back-up to the written charts and for transactional purposes; then, finding itself with this large natural language data base, seeks computer techniques for processing the contents of the documents to obtain the summaries and other information required for health care evaluation and for clinical research. We are bound to see a pressure of this sort arising wherever natural language files are computerized, since once information is available in computer readable form, users inevitably want the computer to process it.

1.4 Automatic Information Formatting

This article will survey the methods and computer programs by means of which the information in natural language documents in a given subject area can be converted into a structured data base so that subsequent computer programs can retrieve specific facts and summarize the different types of information present in the original documents. The technique is based upon the prior development of a natural language parsing system equipped with a comprehensive grammar of English. To this have been added procedures which reduce paraphrastically the variety of syntactic forms found in sentences and align parts of different sentences which have similar grammatical and informational standing in their respective sentences. When these procedures are applied to documents in the same subject area, the result is a table-like structure, called an information format, which contains the same information as the original documents, but arranged in a structured, rather than narrative, form.

To begin with an example, suppose the given document collection is a set of medical records of the type shown in Fig. 1, which is a typical hospital discharge summary. We would like the computer program to arrange the information in this document, and many others like it, so that specific information can be automatically retrieved. A system for formatting textual information has been developed by the New York Uni-

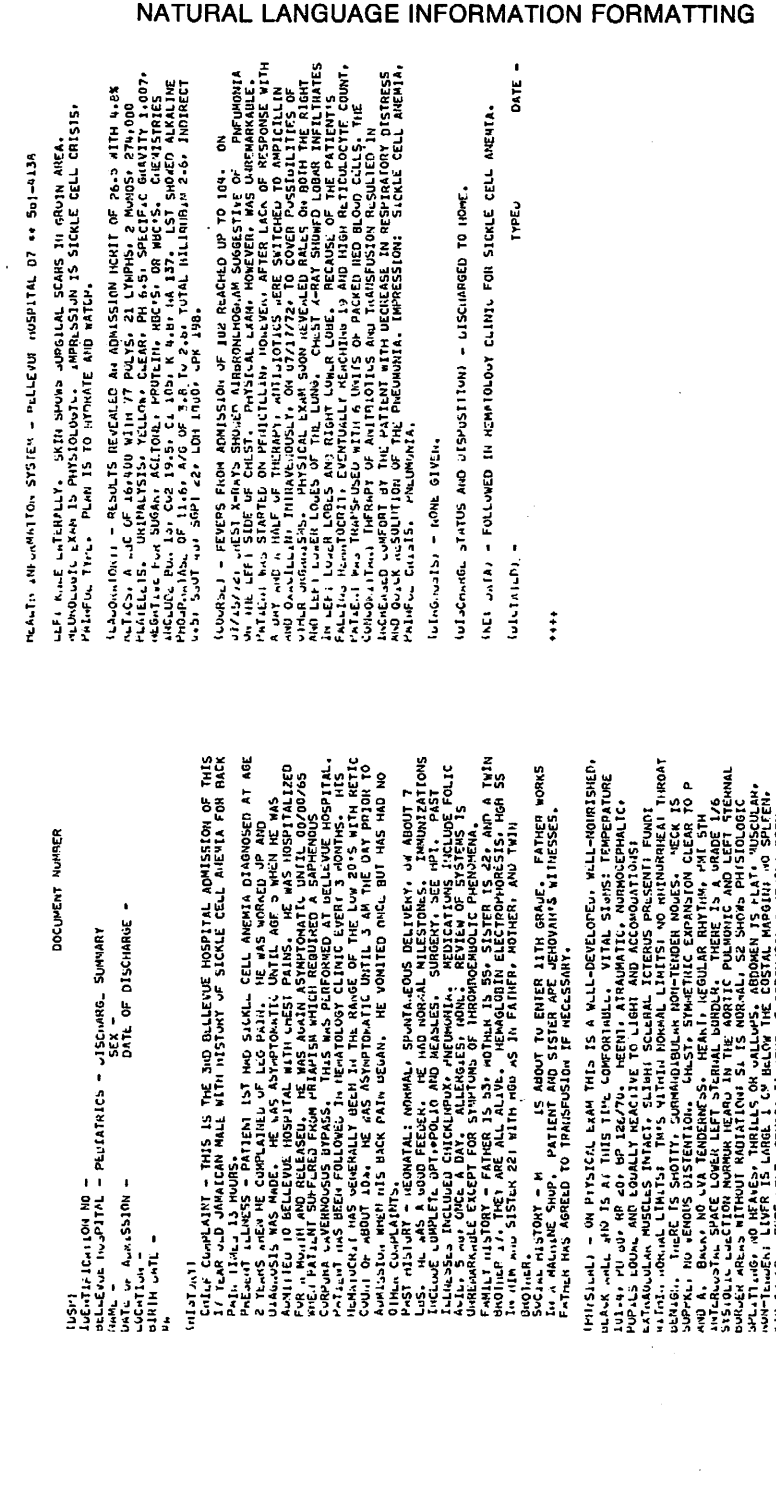


FIG. 1. A sample text.

PATIENT 1ST HAD SICKLE CELL ANEMIA DIAGNOSED AT AGE 2 YEARS WHEN HE COMPLAINED OF LEG PAIN. HE WAS WORKED UP AND DIAGNOSIS WAS MADE. HE WAS ASYMPTOMATIC UNTIL AGE 5 WHEN HE WAS ADMITTED TO BELLEVUE HOSPITAL WITH CHEST PAINS. HE WAS HOSPITALIZED FOR A MONTH AND RELEASED.

	CONJ	PATIENT	TREATMENT		PATIENT STATE					TIME				
			INST	V-MD	V-PATIENT	BODY PART	NORM	SIGN/ SYMPT	DIAGNOSIS	P1	#	UNIT	P2	REF.PT
1		PATIENT		1ST HAD DIAGNOSED					SICKLE CELL ANEMIA				AT	AGE 2 YEARS
2	WHEN	HE			COMPLAINED OF	LEG		PAIN						(AGE 2 YEARS)
3		HE		WAS WORKED UP										
4	AND			DIAGNOSIS WAS MADE										
5		HE			WAS		ASYMPTO-MATIC						UNTIL	AGE 5
6	WHEN	HE	BELLEVUE HOSPITAL	WAS AD-MITTED TO	WITH*	CHEST		PAINS						(AGE 5)
7		HE		WAS HOS-PITALIZED						FOR	A	MONTH		
8	AND	(HE)		(WAS) RELEASED										

*THE WORD "WITH" IS TRANSFORMATIONALLY RELATED TO THE VERB "HAVE"; PATIENT HAS SYMPTOM → PATIENT WITH SYMPTOM.

FIG. 2. An information format.

versity Linguistic String Project. A simplified version of the information format obtained for hospital discharge summaries is shown in Fig. 2, with several sentences from the HISTORY paragraph of the document in Fig. 1 shown as they are mapped by the programs into the medical information format.

As can be seen in Fig. 2, the column headings of an information format are specific to the subject matter. But it should be noted from the outset that these categories are not obtained by an *a priori* semantic analysis of the documents or from knowledge of the subject matter. Rather, they are determined by an analysis of word distribution patterns in sample texts from the field, as described in Section 2 of this article. Likewise, the computer programs for transferring the words of text sentences into the information formats are based on grammatical and word-usage regularities characteristic of the textual material. If it were not for the utilization of the inherent regularities in the language material, it would be impossible for a computer program to recognize and arrange the information contained in freely written English sentences.

Some general properties of information formats can be seen in Fig. 2. (Details of the particular formats obtained for medical records are given in Sections 3.6 and 3.7.)

(1) All of the words in the original sentences are placed somewhere in the format. While content words play a crucial role, function words like prepositions and conjunctions also carry information and are formatted. Some verbs, like *be*, are simply carriers of the subject–predicate relation and could have been dropped as far as the content is concerned, but the placing of all words in the format serves as a check that no information has been lost in the formatting process.

(2) Some linguistic transformations that do not change the information are performed during the processing so as to regularize the English syntactic structures. An example is the expansion of the conjunctional construction in lines 7 and 8: *He was hospitalized for a month and released* becomes *He was hospitalized for a month and (he was) released*, where the words in parenthesis are supplied by an EXPAND CONJUNCTION transformation that operates on the output of a syntactic analysis of the sentence. (The stages of processing are described in Section 3.)

As a result of features (1) and (2), the original sentences or paraphrases of them can be reconstructed from the format entries. Thus, while the form of the sentence is changed, no information is lost, and because only English transformations that are meaning-preserving and valid for the whole language are applied, no information that was not present in the original sentence is added.

(3) Words carrying the same kind of information within the given type of documents are aligned; they are placed in the same column and an appropriate column heading is supplied. (The procedures for determining which words carry the same type of information in the given sublanguage are described in Section 2.4.) This feature is central to the use of information formats in practical text-processing applications. It means that when particular kinds of information are being asked for, e.g., in the discharge summaries the presence of particular symptoms, the information can be found by scanning a particular column of the format for these terms instead of scanning the entire body of texts. Thus, *pain* would be located by scanning the SIGN/SYMPTOM column. In addition, because the context of each occurrence of the terms has also been analyzed, a precise answer can be returned. For example, if evidence of pain was being queried (present in lines 2 and 6 of Fig. 2), the location of the pain and the time at which the pain occurred could also be retrieved by reference to the columns BODY PART and TIME in the same format line as the SIGN/SYMPTOM entry. The ability to draw upon the syntactic context is especially important when the finding is qualified or

negated (*mild pain, pain was not present*). Such information is difficult to supply reliably without an analysis of the sentence.

2. Principles and Methods of Analysis

2.1 The Form–Content Relation in Language

The form–content relation in language—how much there is of it and how it can be established—has occupied the attention of linguists since the beginning of modern liguistics. Language description made a great step forward when language forms were considered to be objects of study in their own right and not only as the derivative of meanings, assumed to be known. [See, e.g., Sapir (1925) and Bloomfield (1926), both reprinted and discussed in Joos (1957).] Pioneers in this development were often accused of ignoring the importance of meaning, though their purpose was rather to arrive at an explanation of how language, viewed as a system, carries out its function of conveying meaning. In any case, the success of this approach in describing the phonemic, morphologic, and syntactic structures of many languages, including those for which Latin-based grammars were inapplicable, showed that this methodology was fruitful, even though many questions concerning meaning remained unanswered.

Within this approach, the method of immediate constituent analysis, first introduced under that name by Leonard Bloomfield (Bloomfield, 1933) and systematized by Harris (Harris, 1951), provided the basis for a formalization of the generative description of sentence structure (Chomsky, 1957) and for a procedural approach to sentence analysis. In immediate constituent analysis, the structure of a sentence is described as a sequence of certain kinds of segments (e.g., noun phrase plus verb phrase) each of which is characterized as being composed in turn of certain segment sequences, down to the words or morphemes of the language. While it was seen that this type of grammar is readily expressed as a set of context free productions, not all grammatical constraints that contribute to sentence well-formedness fit easily into the system. Nor does the analysis carry one very far toward a structure that correlates with meaning, even though a certain amount of information important to the meaning of the sentence is conveyed by its constituent structure. But the importance of immediate constituent analysis was only partly what it showed about sentence structure. Mainly, it demonstrated that grammar could be formulated as a system for analyzing (work of Bloomfield, Harris) or generating (Chomsky's work) sentences, in contrast to previous grammars, which were at best detailed descriptions of grammatical

phenomena (e.g., as in Jespersen, 1914–1929), or more often simply episodic collections of facts and rules.

The discovery of linguistic transformations in the early 1950's was a major advance toward correlating language structure with meaning. Sentences of different forms which were known to contain the same information (e.g., active and passive sentences) were now described as stemming grammatically from the same source sentence. The process of sentence construction could now be viewed as the successive transformation of underlying "kernel" sentences, the elementary assertions contained in the final sentence.[1] Each transformation either combined components, rearranged the sentence paraphrastically, or added a fixed increment of meaning, the same increment for all sentences on which the transformation operated (e.g., *seem* in *They seem to like it, They seem dissatisfied,* etc.) Not only was it now possible by transformational analysis to show a common source element in every sentence which contained the effect of that source element in its meaning, but transformational analysis clarified many linguistic phenomena. For example, it was shown that many ambiguities come about because of degeneracies in the transformational history of sentences, i.e., two different transformational histories resulting in the same final form.

It was immediately clear that transformational analysis had a great potential for language information processing. If sentences could be reduced to a canonical form of primitive assertions plus a certain few operators, where each operator could be associated with an increment of meaning or a paraphrastic change in form, then clearly such operations as recognizing the presence of a particular informational component in a sentence or comparing sentences for overlap in content could be carried out in a systematic, perhaps even computable, way. It also appeared that a formal theory of language structure, showing how language carries its semantic burden, was in the offing.[2]

2.2 Sublanguage Grammars

Regularities in language exist on two levels, one common to the language as a whole and one specific to the subject matter. The regularities

[1] The term "kernel sentences" arose from Harris' initial algebraic formulation of the transformational relation; they were the sentences in the kernel of the defined natural mapping [cf. Harris (1957, Sect. 5.4) and Harris (1968, Sect. 4.3.2)].

[2] With regard to developments in linguistic theory since the discovery of transformations, major works in Chomsky's theory, which combines some of Harris' transformations with other features in a formal generative system, are Chomsky (1957, 1964, 1972, 1975). For the extension of Harris' transformational analysis toward a theory of information in language, see Harris (1976) and papers in Harris (1970).

holding for the whole language are summarized in its grammar (phonology, morphology, and syntax) while the features pertaining to the specific content of discourses are often referred to as the semantic component of the description. Linguists for the most part have been concerned only with the former, and as was sketched above, progress relevant to computational goals has been made on this level. Computer scientists dealing with language, on the other hand, have for the most part dealt with the language material around specific structured data bases, or with instructions or conversations of a restricted type. Here it is sometimes possible to come to grips with the semantic categories and relations of the data as they appear in the language material without the use of exhaustive syntactic analysis (Charniak and Wilks, 1976) and without a general method for arriving at the relevant semantic categories. However, when the subject matter and the language material is more complex, as in the case of science writing, it is difficult, and would be costly, to proceed without a more general methodology suited to the purpose.

Within both linguistics and computer science some attempts are currently being made to formulate semantic theories and systems valid for the whole language, for example, to state a set of semantic primitives and rules of combination from which the meanings of sentences would be generated. For science information, however, it is hardly possible to conceive of a system of this type that would generate the meanings of science sentences. A practical consideration is that specialists on the level needed for such specification (assuming the task doable) are not available for this work. Where the goal is to process the information in text sentences there is the additional constraint that a computer program must be able to map text sentences into the defined structures. And in view of the effort required to program a natural language processor, it is also important that there be methods to adapt the system for use on different subject matters.

These considerations led to the search for general methods of determining the appropriate semantic structures for text processing in science subfields. It was clear that communications within one universe of discourse differed linguistically from those in another. This is especially marked in scientific fields, where the vocabulary of content-words differs sharply from field to field and where whole sentences which are mainly composed of common English words are intelligible only to persons engaged in that particular discipline. This specialized use of language in science subfields in discussed by Bross (Bross *et al.*, 1972) using the notion of a science sublanguage, first introduced in conjunction with the definition of sublanguage grammars (Harris, 1968, Sect. 5.9).

In the case of a whole language we know that there are grammatical

rules because some word sequences are accepted by native speakers of the language as grammatical while other sequences are rejected as ungrammatical. A similar situation exists in a community of individuals engaged in a specialized field of science. Certain statements will be accepted as possible within the discipline while others will be rejected as impossible or outlandish. What is involved is not truth versus falsity or even accepted versus unconventional formulations, but rather whether the statements would be nonsensical or run counter to fundamental knowledge in the discipline. Thus, for example, the statement the *ion crosses the membrane* would be acceptable to a cell biologist—it may or may not be true in a given case—whereas *the membrane crosses the ion* would be rejected as unsayable in the science. This linguistic behavior on the part of the scientist indicates that rules analogous to the rules of grammar for the whole language are operating in the language of a particular scientific discipline. These rules are called a sublanguage grammar.

The idea of a sublanguage grammar suggests a possible method for determining semantic structures in a science subfield. If a grammar summarizes the restrictions on occurrence that characterize a language, then a sublanguage grammar should capture the restrictions on occurrence that distinguish one area of scientific discourse from another. One would expect sublanguage grammatical categories and rules to have semantic standing, since what is captured in the sublanguage grammar over and above ordinary grammatical regularities is precisely what is special to the subject matter. This has proved to be the case. If one applies descriptive linguistic methods similar to those used in developing a grammar for a whole language to a corpus of texts in a science subfield, one obtains detailed patterns of word co-occurrence from which characteristic word subclasses and word-subclass sequences can be stated (i.e., a grammar). These word categories and syntactic formulas of the sublanguage grammar correlate closely with the classes of real-world objects and relations that are of special interest in the subfield. They thus provide a set of semantic structures for representing subfield information. And because the structures are based on syntactic regularities in the textual material (of both the general and sublanguage types) it is then possible to define procedures that locate occurrences of the structures in the sentences of subfield texts.

2.3 Information Structures in Science Subfields

The sublanguage-grammar hypothesis was first tested by applying the methods of distributional linguistics to a corpus of texts in a subfield of

pharmacology, the mechanisms of action of digitalis. In this study, nineteen texts were used, about 200 journal pages, divided into two sets, one to establish the grammar and the other to test it. Syntactic analysis was applied (manually) to the sentences of the first set, including the application of a small number of well-established paraphrastic transformations to regularize the sentence representation. Word classes were established on the basis of similarity of co-occurrence, for example, grouping together nouns which occurred as the subject of the same verb, and similarly verbs which occurred with a particular selection of nouns as their subject. It was found that there were distinct classes of this type in the texts, and it was possible, as conjectured, to write a specialized grammar summarizing the co-occurrence patterns of the classes (Sager, 1972a, 1975).

The word classes in the digitalis sublanguage grammar correlated with recognizable semantic classes in the textual material, as verified by a consulting cardiologist. The analysis produced noun classes corresponding to: cardiac glycosides, cations, contractile proteins, enzymes (ATPase), heart muscle, cell, cell substructure, etc., and verb classes corresponding to specific relations among the noun classes (e.g., *move* verbs connecting *cations* to *cell*), quantitative relations (e.g., *increase, decrease*), causal relations (e.g., *affect, produce*), as well as relations of the human investigator to all of the above (e.g., *observe, report, study*).

The grammatical structure of the sublanguage was found to consist of[3]:

(1) a set of elementary sentence types composed of the subfield-specific word classes (e.g., N_{ION} V_{MOVE} N_{CELL}, corresponding to such occurrences as *Sodium flows out of the cell, Potassium moves into the cell.*[4]

(2) aspectual operations on verbs (e.g., *not, fail to, appear to, tend to, persist, continue, commence*, etc.);

(3) quantifiers Q (e.g., *amount of, rate of*) on certain verbs or nouns, and quantifying verbs V_q (e.g., *increase, decrease*) operating on Q or V_q;

(4) the *wh* connective (relative clause), as in all of English;

(5) conjunctions of English, and conjunctional verbs that connect nominalized sentences (e.g., *affect, be concerned in, cause, produce, accompany, interfere with, involve*, as in *calcium exchange accompanies the excitation*);

[3] This brief summary is elaborated in Sager (1972a).

[4] There were about 20 of the elementary sentences types with some elementary sentences in the texts not repeated with sufficient frequency to constitute a type. The unique elementary sentences were often drawn from neighboring fields, referring to phenomena without discussing them.

(6) verbs and predicate sequences that operate on sentences (e.g., *It is not certain whether, We observe that*, etc.).

The pharmacology sublanguage study gave rise to the notion of an information format, since it was found that the grammatical structure of the sublanguage could also be presented as a prototype sentence form in terms of which each successive text sentence could be seen as the realization of certain particular options within the overall hierarchical organization of the sublanguage grammar. The choice of options in each case showed precisely which types of information were present in the sentence.

The overall structure of the format is given by English grammar. That is, the underlying grammatical relations of English determine the major columns in the format. These major relations are the subject–verb–object structure of elementary sentences; the fact that for certain verbs the subjects and objects consist of whole sentences (in nominalized form); and the fact that certain distinguished adjuncts, e.g., quantifiers and time expressions, have special syntactic status in sentences. The specialization of language usage in the given subfield, i.e., the sublanguage grammatical relations, shows up in the following: Particular sublanguage word-classes in the subject–verb–object positions constitute particular subtypes of sentences that occur frequently in the material. These subtypes have a specific semantic character, enabling us to attach sublanguage-specific labels to the major columns and subcolumns of the format. The information carried by sentences of these types is thereby classified and can be mapped into the appropriate columns of the format. This can best be illustrated by examples of formatted sentences.[5]

Elementary Fact Units

The formats obtained in the pharmacology sublanguage study contained one case or another of the basic unit illustrated in the format in Fig. 3 for the sentence *Calcium uptake into liver mitochondria appears not to be affected by cardiac glycosides*. In this sentence, as in most others in this material, there is a "bottom level," elementary assertion (shown in the format between double bars) consisting of a verb with its subject and object, which are concrete nouns.[6] In the pharmacology texts, this assertion described an elementary physiological or biochemical event, which in the case of Fig. 3 is the uptake of calcium into the

[5] The remainder of this section and Figs. 5–7 are from Sager (1977).

[6] "Bottom level" refer to its position in the operator–operand hierarchy obtained by syntactic decomposition of the sentence (see Fig. 7).

GL641 13.6.11 CALCIUM UPTAKE INTO LIVER MITOCHONDRIA APPEARS NOT TO BE AFFECTED BY CARDIAC GLYCOSIDES.

DRUG	V-CAUSE	ARG1	V-PHYS	ARG2	CONJ
CARDIAC GLYCOSIDES	AFFECT (APPEARS NOT TO)	CALCIUM	UPTAKE INTO	MITOCHONDRIA (LIVER)	

FIG. 3. Formatted pharmacology sentence (1).

mitochondria of the liver. The elementary assertion here is an instance of a subclass sequence that recurred over and over in these texts, N_{ION} V_{MOVE} N_{CELL}, in which a noun in the ion class is connected to a noun in the cell or cell substructure class by a verb in a class whose common semantic feature is movement. Despite the clear meaning features of these subclasses, it should be kept in mind that they are defined syntactically, by their position of occurrence *vis a vis* other classes. Examples of other elementary assertions encountered in this literature were those covering ion interactions, enzyme activity, tissue contraction or contractility, protein behavior, and ions binding to molecules.

Operating on the elementary assertion, very often, was a noun–verb pair, shown in Fig. 3 to the left of the double bars, consisting of a drug word and a verb of roughly causal character (*affect, influence,* etc.) possibly negated or quantified, as in this sentence. The causal pair is said to operate on the elementary assertion because the latter appears as the object of the causal verb. A paraphrastic transformation (passive → active) was required in order to reveal that *uptake* is the object of *affect*, since *uptake* appears in the sentence as the subject of the passive construction *appears not to be affected.* Also, while *uptake* appears in the sentence as a noun, it is in fact a nominal form of the verb *take up,* so it is put in the verb column. The subject and object of this verb occur in the sentence as adjuncts of the nominal form (*uptake*) but in the format they are restored to verb-argument status. As illustrated in the format of this simple sentence, the major fact type in this pharmacology material was composed of an elementary assertion drawn from a prior science (cell physiology, biochemistry), with the pharmacological agent entering only on a higher grammatical level, as an operator on the elementary assertion.

Fact versus "Meta-Fact"

The somewhat longer sentence formatted in Fig. 4 utilizes format columns that were not shown in Fig. 3 because they were empty there. Two new columns appear on the left, labelled HUMAN and V-STUDY.

GL 641 2.2.1 MORE DETAILED STUDIES OF THE EFFECTS OF CARDIAC GLYCOSIDES ON SODIUM AND POTASSIUM MOVEMENTS IN RED CELLS HAVE BEEN MADE BY KAHN AND ACHESON (99), SOLOMON ET AL (168) AND GLYNN (67).

HUMAN	V-STUDY	DRUG	V-CAUSE	ARG1	V-PHYS	ARG2	CONJ
K AND A (99) S ET AL (168) AND G (67)	HAVE MADE MORE DETAILED STUDIES OF	CARDIAC GLYCOSIDES	EFFECT	SODIUM	MOVE IN	RED CELLS	AND
				POTASSIUM	[MOVE IN]	[RED CELLS]	

FIG. 4. Formatted pharmacology sentence (2).

Factual assertions involving only the concrete objects of investigation in the science and their interrelations (the two inner sections of the format) are syntactically separable from the words describing the scientist's relation to the fact. Verbs like *study, present, discuss, assume, report,* which have exclusively human subject nouns and carry the connotation of the scientists' intellectual activity, appear as higher level operators in the operator-structure already built up from the words in the "object language" of the science.

Another new feature in Fig. 4 is the conjunction column CONJ on the right which contains words that connect one line (or several grouped lines) of the format to another line or lines. This is a major departure from tables for quantitative data. In Fig. 4 the conjunction is *and,* but in other cases the conjunction may have the form of a verb or a phrase (e.g., *is associated with, is the basis for*). Apart from grammatical conjunctions, only words which have the grammatical property of operating on a pair of nominalized sentences are accepted in the CONJ column. The words in the CONJ column are much the same in different subfields; whereas the words in the innermost columns are highly specific to the field.

A last point to notice in Fig. 4 is the presence of reconstructed word occurrences, shown in square brackets. *Sodium and potassium movements in red cells* is expanded by paraphrastic transformation to *sodium movements in red cells and potassium movements in red cells.* The expansion of the phrase into two assertions does not imply that the events are independent of each other; only that the connection between them is not more explicit here than their conjoining by *and.*

Data Structures versus Argument

A third formatted sentence, shown in Fig. 5, is sufficiently complex so that it illustrates some of the regularizing effect that formatting achieves

LA 721 1.1.5 THE POSSIBILITY THAT ADMINISTRATION OF DIGITALIS, THROUGH ITS INHIBITION OF THE NA+ - K+ COUPLED SYSTEM, PRODUCES AN INCREASE IN NA+ - CA++ COUPLED TRANSPORT AND THEREBY AN INCREASE OF INFLUX OF CA++ TO THE MYOFILAMENTS IS DISCUSSED AND IS PRESENTED AS A POSSIBLE BASIS FOR THE MECHANISM OF DIGITALIS ACTION.

HUMAN	V-STUDY	DRUG	V-CAUSE	V-QUANT	ARG1	V-PHYS	ARG2	CONJ
[AUTHOR]	DISCUSSES 1 2	DIGITALIS (ADMINISTRATION OF)	PRODUCES POSSIBLY	INCREASE	NA+ - CA++ COUPLED	TRANSPORT		AND THEREBY
		[DIGITALIS (ADMINISTRATION OF)]	[PRODUCES]	INCREASE	CA++	INFLUX TO	MYOFILAMENTS 2	THROUGH
		[DIGITALIS] = ITS	INHIBITION		NA+ - K+ COUPLED SYSTEM		1	AND
[AUTHOR]	PRESENTS	<---- [1] ---->						AS BASIS FOR (POSSIBLE)
		DIGITALIS	ACTION MECHANISM					

FIG. 5. Formatted pharmacology sentence (3).

for a whole text. When the sentence is read without reference to the format, it is not apparent that it is composed of repeating sequences of similar elements. As the format shows, the sentence consists of four interconnected factual units of the same basic type. The texture, and the intellectual content, comes from the interrelations among these subtypes, and from several other linguistic features: the use of conjunctions at different levels of grouping, the introduction of qualifying modifiers and higher level operators, and the use of reference, either explicitly via pronouns or implicitly via ellipsis. These features belong to the argument or reasoning in the text, which can be separated from the factual units shown in the inner portions of the format lines.

Turning first to the individual fact units in Fig. 5, the inner portion of the first line, stripped of its qualifiers, says that digitalis produces an increase in Na^+–Ca^{++} coupled transport. In this unit, *Na^+–Ca^{++} coupled transport* is an instance of the formula N_{ION} V_{MOVE} N_{CELL} seen previously, even though the *cell* word is not present here. In often-repeated material, the subject or object of the verb is frequently dropped; sometimes the verb is dropped if it is unique to the stated subject or object. This is the case in the third line, where *transport* is suppressed but easily reconstructed because of the subject, *Na^+–K^+ coupled system*.

Figure 5 introduces a new column V-QUANT between the innermost assertion and the columns DRUG, V-CAUSE. The V-QUANT column was not shown in the preceding formats because no words like *increase* or *decrease* were present in the sentences. In line 3, the V-QUANT column is empty. Another possibility is to split the word *inhibition* into two column entries, V-CAUSE and V-QUANT, since we find in the texts that *inhibit, produce a decrease*, and *cause a decrease* occur in similar environments.

The format in Fig. 5 also illustrates the use of pronouns and other devices of reference. In the third line, the antecedent of the pronoun its, namely *digitalis*, has been reconstructed as the subject of *inhibit*. This follows the pattern throughout that the class of pharmacological agents occurs as the subject of verbs in the V-CAUSE class.

The fourth format line has the interesting property that the whole object language, or factual, portion of the format is empty of physically occurring words. The three preceding format lines, seen as a unit, are repeated implicitly as the first operand (subject) of the binary relation *is a basis for*, where the second operand (object) is *mechanism of digitalis action*. Reasoning in science writing is characterized by devices of this sort. A single assertion becomes a nominalized sentence within another sentence; a sequence of interconnected sentences becomes an element of a later sentence by implicit repetition or by pronominal reference (*this, this process*, etc.) In this way it becomes possible for complicated interrelations to be expressed in the physically linear medium of language.

Properties of Science Information

From a study of information formats in different subfields, one gets a picture of how scientific information is carried by language both with respect to the unique informational characteristics of each science and with respect to the general properties of information viewed over science as a whole.

First, the information formats of a particular science reflect the properties of information in that particular science in contrast with other sciences. The pharmacology formats, for example, displayed a characteristic predicational hierarchy in which the word for the pharmacological agent occurred in the predicate on an embedded sentence of the type found in cell physiology or biochemistry, thus reflecting the role of the drug as an outside element that affects ongoing processes. Quantity and quantity-change were important in the pharmacology formats (not all quantity columns are shown in the example formats, e.g., dosage) reflecting the importance of quantity relations in this science. This type of format contrasts with one that was obtained for medical records, illustrated in Fig. 2. In the medical format, columns for time words are essential to the information, whereas they were almost entirely absent in the pharmacology formats. In the medical formats, there is very little

predicational hierarchy and virtually no argument, both of which were present in the pharmacology formats. The structure of the clinical information, displayed in the formats, is an interplay between columns containing treatment words and columns containing words that describe the patient's state; successive rows are linked primarily through time sequence, with the conjunction columns playing a secondary role.

While the formats for different subfields differ, as they should, to capture the specific character of information in each field, they have certain properties in common that appear to hold for all of science writing. To mention just a few:

(1) Statements about science facts are separated by the grammar from the science facts proper; the role of the human investigator is carried by a grammatically distinct class of verbs and is syntactically separable from the report of factual events.

(2) The report of a complex event has a structure composed of a hierarchy of different types of operators, the "bottom level" operand being the carrier of the most elementary objects and events. When a given science draws upon a prior science, the material from the prior science appears as the operand of material from the given science.

(3) Argument is carried by connectives between the data structures built up in this hierarchial fashion. Whole units are carried forward by the telescoping of an operator-hierarchy into a single noun phrase, pronoun or "pro-sentence," or by the controlled dropping of words permitted by the grammar.

(4) A surprising amount of repetition and regularity is found in all science writing, once stylistic variations and equivalent grammatical forms are eliminated. Every individual piece of writing contains some repetition (or it would not be connected discourse). Across a single specialized discipline, the same items repeat in different combinations and with variations, as though all the texts were part of a single extended discourse. Although the texts each bring in some new feature they are sufficiently similar as to fit into an overall structural characterization. These structures, or information formats, are then a powerful tool for organizing the information in subfield texts.

2.4 Automatic Generation of Subfield Word Classes

The manual study of a pharmacology sublanguage, described above, showed that sublanguage word classes can be established on the basis of co-occurrence similarities of words in grammatically analyzed subfield sentences, and that these word classes constitute a basis for defining information formats for the textual material. However, the work of manually grouping together words on the basis of co-occurrence similarities proved a tedious task. And although only words which occur frequently in the same environments were put in one class, the results could not be as precise as if numerical similarity coefficients were calculated. It was therefore felt that a clustering program should be written to automate this tedious step and to provide a more rigorous demonstration of the relevance of distributional analysis.

The steps in the procedure are summarized schematically in Fig. 6. Steps 2–5 constitute the clustering program proper, which was written especially for linguistic data. It is intended that the program operate from the output of the LSP sentence analyzer (parser + transformational component). However, at the time the experiment described here was performed, the transformational component was in process of implementation, and the input was obtained manually by applying standard English transformations to text sentences.

The procedure can be described informally with reference to Fig. 7 and Tables I and II, which carry through the analysis for several sample

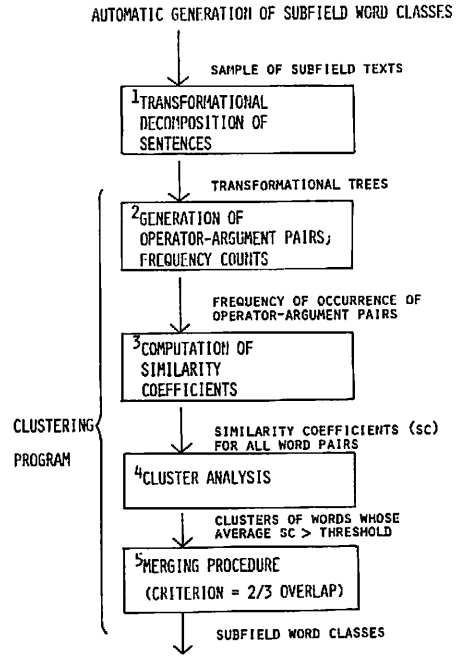
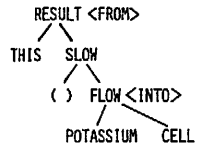
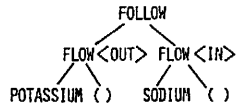


FIG. 6. Automatic generation of subfield word classes.

S1. THIS RESULTS FROM THE SLOWING OF THE INFLUX OF POTASSIUM INTO THE CELL.



S2. THE INFLUX OF SODIUM IS FOLLOWED BY AN EFFLUX OF POTASSIUM.



S3. HAVING ESTABLISHED IN BROAD OUTLINE THE MOVEMENTS OF SODIUM AND CALCIUM, ...

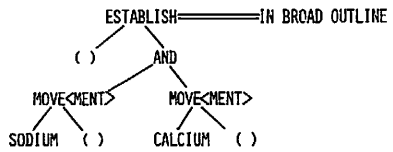


FIG. 7. Transformational trees.

sentences from the corpus of pharmacology sentences which was used in the experiment. Fig. 7 shows the transformational trees for the sample sentences. Each tree is made up exclusively of terminal nodes labelled with the base forms of the lexical items arranged in an operator–argument hierarchy; the verb dominates its subject and object(s), conjunctions dominate the verbs of the conjoined sentences, etc. Adjuncts (i.e., modifiers, as in S3) are shown connected to the element they modify by a double line. Transformations are used to denominalize a verb in nominalized sentences, e.g., *slow* in *the slowing of the efflux of potassium* (S2), *move* in *movements of sodium and calcium* (S3), to undo passives (e.g., in S2 *is followed by* → *follow* with reversal of subject and object), and to expand conjunctions, as in S3 where *the movements of sodium and calcium* → *the movement of sodium and the movement of calcium* and thence by denominalization to *sodium moves and calcium moves*, and finally to *sodium move and calcium move*, since tenses are removed. The purpose is to bring all occurrences of the same base morphemes into a single form so that they can be counted, and to do a similar job for all

TABLE I

OPERATOR–ARGUMENT OCCURRENCES

Operator	First argument	Second argument	Third argument
S1. RESULT ⟨FROM⟩	THIS	SLOW	
SLOW		FLOW ⟨INTO⟩	
FLOW ⟨INTO⟩	POTASSIUM	CELL	
S2. FOLLOW	FLOW ⟨OUT⟩	FLOW ⟨IN⟩	
FLOW ⟨OUT⟩	POTASSIUM		
FLOW ⟨IN⟩	SODIUM		
S3. ESTABLISH		MOVE	
MOVE	SODIUM		
MOVE	CALCIUM		

occurrences of a particular operator–argument relation (verb-subject, verb-object).

The input to the culstering program consists of a linearized form of the transformational trees obtained for the sentences. From these, all operator–argument pairs can be listed and their frequency of occurrence tabulated, as illustrated in Tables I and II for the data in sentences S1–S3. In reading off operator–argument pairs from the transformational trees, the conjunction nodes (e.g., *and* in S3) are "transparent." Thus, in S3, *move* is obtained as argument of *establish*.

To compute similarity coefficients, each word W_i is assigned a characteristic vector V_i which has $6n$ components if there are n distinct words

TABLE II

FREQUENCY OF OPERATOR–ARGUMENT PAIRS
(Noun Arguments Only)

	Flow	Move
1st argument POTASSIUM	2	0
1st argument SODIUM	1	1
1st argument CALCIUM	0	1
1st argument CELL	0	0
2nd argument POTASSIUM	0	0
2nd argument SODIUM	0	0
2nd argument CALCIUM	0	0
2nd argument CELL	1	0
3rd argument POTASSIUM	0	0
3rd argument SODIUM	0	0
3rd argument CALCIUM	0	0
3rd argument CELL	0	0

in the corpus. The number $6n$ arises because there are 6 possible relations which each W_i could have to another word W_j:

(1) W_i could be an operator with W_j as its first argument
(2) W_i could be an operator with W_j as its second argument
(3) W_i could be an operator with W_j as its third argument
(4) W_j could be an operator with W_i as its first argument
(5) W_j could be an operator with W_i as its second argument
(6) W_j could be an operator with W_j as its third argument.

The value of the component is the number of occurrences in which W_i and W_j have the stated relation. Thus, in Table II, which lists the components of *slow* and *move* (only as operators) with respect to the four nouns in S1–S3 (as arguments), the number 2 under FLOW in the row labelled 1st ARGUMENT POTASSIUM is the number of occurrences in S1–S3 of *flow* as operator (= verb) with *potassium* as its first argument (= subject). As Table II illustrates, the characteristic vectors are sparse; only a few of the components are nonzero.

In the clustering procedure, each characteristic vector is normalized to unit length, and multiplied by a weighting factor to reduce the effect of infrequently occurring words. The similarity coefficient between two words is then calculated by taking the inner product of the normalized, weighted characteristic vectors of the two words.

Clusters are built up one word at a time. Two words form a cluster if their similarity coefficient exceeds a threshold value, which is a parameter of the program. A word W may be added to a cluster if and only if the average of the similarity coefficient of W with each word in the initial cluster exceeds the threshold. The reason that clusters are built up one word at a time is to avoid the situation of grouping together unrelated small clusters due to high similarity coefficients within the small clusters, which might be high enough to compensate for the lower similarity coefficients between words from the different component clusters. The output of the clustering program generally contains a number of clusters with overlapping membership. Clusters with 2/3 or more members in common are merged to form the final output.

Experimental Results[7]

The clustering program was run on a set of 400 sentences taken from 6 articles on the mechanism of action of digitalis. Sentences were not specially selected, except that the Methods sections of the articles were excluded. The sentences yielded approximately 4000 operator–argument

[7] Summarized from Hirschman *et al.* (1975).

pairs and a vocabulary of some 750 words. The similarity coefficients between all pairs of words were computed and the words were grouped into clusters by the algorithm described above. While 400 sentences is a small corpus, it turned out, rather surprisingly, that the main subfield word classes and the main members in each class were obtained by the computer program.

Table III shows one part of the clustering output, the noun classes. It can be seen that the classes are coherent and very few words are in the wrong class. However, to evaluate the results further, the same 400

TABLE III

CLUSTERING PROGRAM OUTPUT
(Noun Classes Only)

Merged classes, Run of 11.13.74, $t = 0.250$

Noun classes:

CG class	Cation class		
agent	Ca	ion	ion
cardiotonic glycoside	Ca^{++}	K^+	substance
CG	calcium		
compound	electrolyte		
digitalis	glucose		
drug	ion		
erythrophleum alkaloid	K		
inhibitor	Na^+		
ouabain	potassium		
strophanthidin	sodium		
strophanthidin 3 bromoacetate			
strophanthin	**Protein class**		
Muscle class	actomyosin		
	cardiac		
atrium	fiber		
heart muscle	protein		
muscle			
ventricle	**SR class**		
Enzyme class	sarcoplasmic reticulum		
	SR		
Na + K + ATPase			
ATPase			
enzyme			
False clusters			
Myocardium ADP			
cell EI			

sentence corpus was analyzed manually and the computer-generated classes were compared with those obtained manually to determine two points: (1) How many of the classes recognized as significant by the human analyzer were represented in the computer output? (2) What proportion of the words in each manually prepared class (which can be assumed to be relatively complete) were present in the corresponding computer-generated class?

The results of the comparison of the noun classes in answer to Question 1 are shown in Table IV. Of the 11 major noun classes found manually, 10 are accounted for by the computer: 6 by merged clusters and 4 by single member classes. One major class recognized manually (phosphorylated compounds) did not appear, due to a minor mistake in the program. On the average the computer classes accounted for 84% of the nouns in each manual class. Overall the computer classes + single member classes account for 1335 of 2016 occurrences = 66% of pair-occurrences of concrete nouns in the corpus.

Table V shows the results of comparing the membership of the computer-generated class of cardiac glycosides (CG) with the manual obtained CG class, in answer to Question 2. In the case of the CG class, and others, the computer-generated class includes the major nouns of the class, and with minor exceptions does not include nouns from other classes. The computer CG class accounted for 89% of the pair occurrences of words in that class. In the case of the cation class the corresponding number was 96%.

2.5 The Sublanguage Method Summarized

Table VI summarizes the sublanguage method as it has evolved since the first study. In approaching a new subfield, first a linguistic analysis is performed on a sample of the subfield texts in order to determine the word classes and information structures in the material (Part A of Table VI). As we have seen, this analysis can be done on the basis of word distribution patterns in grammatically analyzed sentences. While some of the steps have been automated, and more may be in the future, this remains an essentially human task.

Once the subfield information format (or formats) has been defined, subfield texts can be converted to a structured data base by the six-step procedure outlined in Part B of Table VI. Of these, the first two steps produce the computer lexicon to be used in the processing, and the remaining four carry out the processing of the text sentences. The text processing procedures have been fully automated and are described in detail in the next section.

TABLE IV

COMPARISON OF NOUN CLASSES OBTAINED MANUALLY AND BY COMPUTER

Class	Manual[a]	Computer[a,b]	%COMP/MAN
	No. OCC/No. N	No. OCC/No. N	
major classes[c]:			
CG	442/22	395/11	89
Cation	412/14	394/ 9	96
Enzyme	192/13	157/ 3	82
Protein	136/21	63/ 3	45
SR	101/ 5	97/ 2	97
Cell	82/ 6	77/ 1	94
Phosph. Cmpds.[d]	66/10	xxxxxx	xx
Membrane	55/ 5	42/ 1	76
Heart	53/ 3	39/ 1	74
Heart parts	44/ 3	35/ 1	80
Muscle	45/ 6	38/ 2	84
minor classes[c]:			
Human agent	95/54		
Drug, not incl. CG	88/25		
Ultrastructure, not incl. SR	42/15		
Native org. sub.	33/12		
Organism	23/ 9		
Tissue	20/ 3		
Organ not heart	17/ 8		
Inorg. molecule not incl. cation	15/ 6		
Expt. medium	13/ 3		
Physical forces	12/ 5		
Miscellaneous	52/12		

[a] Entries are: total number of pair occurrences of nouns in class/number of nouns in class.

[b] Single member classes are shown in correspondence to manual classes if the single word in question accounts for two-thirds or more of the pair occurrences of words in the manual class. In almost all cases, this word is identical to the name of the class.

[c] Major classes are classes which have 50 or more total occurrences, and at least one member with more than 8 occurrences. Minor classes have either less than 50 occurrences total, or no member with more than 8 occurrences, as in the human agent class.

[d] A *phosphorylated compounds* class was obtained on previous runs (five nouns, with 71% coverage of the manual class). Due to a small error, this class did not appear in this run.

With regard to the preparation of a suitable computer lexicon, this is an essential and nontrivial operation. To format the text sentences a correct parse must be delivered to the transformational component. This requires that the words (or most of the words—cf. below) have correct English syntactic classifications down to the subclass level. To proceed

TABLE V

PROPORTION OF WORD CLASS MEMBERS IN COMPUTER OUTPUT
(Cardiac Glycoside Class Only)

CG class computer[a]	Manual[a]	Number occurrences in pairs[b]	
CG	CG	156	
digitalis	digitalis	118	
ouabain	ouabain	70	
drug	drug	15	
agent	agent	8	
strophanthidin	strophanthidin	5	$\frac{395}{442} = 89\%$
strophanthidin 3 bromoacetate	strophanthidin 3 bromoacetate	4	
strophanthin	strophanthin	4	
cardiotonic glycoside	cardiotonic glycoside	3	
compound	compound	7	
inhibitor	inhibitor	5	
erythrophleum alkaloid[c]		6	
	glycoside	11	
	digoxin	7	
	acetyl strophanthidin	7	
	cardioactive glycoside	6	
	digitalis glycoside	6	
	digitoxigenin	3	
	sprophanthoside	2	
	cardiac glycoside	2	
	digitoxin	1	
	digitalis compound	1	
	strophanthin K	1	
		442	

[a] *Agent, drug,* and *compound* are classifiers for words of the CG class, as well as of the more general DRUG class. *Inhibitor* is also a classifier, which classifies according to function.

[b] An occurrence of a word either as the operator or operand in a pair. Pair-occurrences are more numerous than text occurrences for several reasons. Recoverably zeroed material is reconstructed and contributes to pair formation. Also each operator can appear in a pair as the operand of its operator, as well as with each one of its arguments. (Thus a two-argument verb can appear in three pairs.) For concrete nouns however this does not occur, and the pair-occurrences correlate more closely with the number of actual occurrences in the text.

[c] *Erythrophleum alkaloid* does not belong in the CG class; it is a drug whose effect is compared to that of the cardiac glycosides.

TABLE VI

SUBLANGUAGE METHOD

A. Discovery of information structures
1. Select a representative sample of subfield texts.
2. Determine the sublanguage word classes based on similarity of word environments in text sentences.
3. Determine the sublanguage grammar based on co-occurrence patterns of sublanguage word classes.
4. Define the subfield information format based on the sublanguage grammar.

B. Automatic conversion of subfield texts to a structured data base.
Preliminary
1. Look up text words in LSP lexicon; print list of new words.
2. Prepare lexical entries for new words[a], update lexicon.
Text processing
3. Parse sentences using LSP parser, lexicon, and English grammar.
4. Regularize parse trees using LSP English transformations.
5. Map regularized parse trees into information format using sublanguage formatting transformations.
6. Regularize formats using procedures that recover implicit material.

[a] Currently a manual step.

further, i.e., to map the words into their correct information–format slots, requires that the content words also have correct sublanguage classifications. Together, these requirements present a considerable burden of lexical classification. With training, coders can prepare a lexicon for each subfield of application, and once the job is done the lexicon can be used for many subfield texts, with only minor updating required. However, automation of this stage would clearly be desirable, and at this writing, research on automating the lexical coding is underway.[8]

3. Computer Programs for Information Formatting

Computer programs for processing English sentences have been under development for almost 20 years. Advances in the computer field, par-

[8] The fact that 50–80% of the words in any new text are found in the LSP English lexicon (The percentage depends on how many previous texts in the subfield have been processed) suggests that a combination of morphological clues and parse tree context may make it possible to achieve correct parses of sentences containing new words, without hand coding all the new words. Further, the words in the environment of the new word which already have sublanguage classifications may in some cases determine the sublanguage class of the new word. These conjectures are being tested.

ticularly higher level languages and syntax driven compilers, along with bigger, faster machines, have aided in this development, but they have not in themselves solved the special problems of natural language. Among these problems are the inherent syntactic ambiguity of sentences, the existence of implicit elements (ellipsis), the rich network of conjunction and comparative constructions, the many detailed constraints that apply to word subclasses, the large amount of lexical variation (every word is different), the need for a semantic representation suited to the application, and the very size and complexity of the system needed to cope with all these features in algorithmic terms.

3.1 Linguistic Framework

The choice of linguistic framework plays an important role in how these problems are solved. In the case of the LSP system, the use of linguistic string analysis as the grammatical framework for the programs[9] has led to an economical organization of the grammar and a special method of dealing with linguistic relations. As will be seen below, the components of sentences under linguistic string analysis are such that all syntactic and semantic constraints on the words of a sentence operate locally, either within one component, or between contiguous components in the string analysis of the sentence. (A few constraints operate between elements separated by a chain of such units.) All linguistic operations can thus be carried out using a small number of basic routines corresponding to the simple grammatical relations which hold in the local case. This results in very great economy in specifying the grammar and appears to be the mechanism whereby language carries its enormous amount of detail without burdening the processor (human in this case) with thousands of complicated rules (Sager, 1967, 1972b).

String Analysis

Briefly, under string analysis, a sentence consists of an elementary central sentence (the main clause, stripped of modifiers) called the **center string,** and an optional number of **adjunct strings** (modifiers) and **conjunctional strings,** each of which occurs in the sentence at a stated position in the string it adjoins, usually to the left or right of a particular element, such as a noun or verb. For example, in *during the first admission,* the adjective *first* is a single-element adjunct string adjoined to the left of the noun *admission* in the prepositional string (PN: preposition + noun) *during admission.* Also, a sentence may contain a **sentence nominalization string** occurring in the position of a noun and its adjuncts, e.g. *Your being present* in *Your being present would be helpful,* parallel to *Your presence would be helpful.* Strings combine to form sentences in accord with their membership in sets of the above types; thus, for example in constructing a sentence, strings in the set left adjuncts of N (LN) are inserted to the left of the noun they are to adjoin, strings in the set right adjuncts of the verb (RV) are inserted to the right of the verb, and so forth.

[9] A summary of this theory is given in Harris (1968, Sects. 3.4 and 3.5).

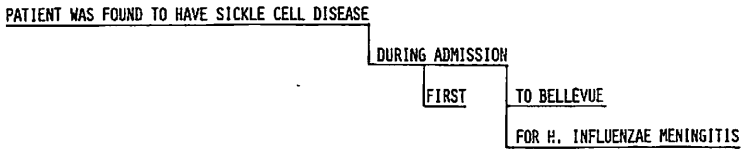


FIG. 8. Simple string diagram.

The string analysis of a sentence is very similar to what many of us were taught in grade school under the name of sentence diagramming. Fig. 8 shows a simple string diagram of a typical sentence from a hospital discharge summary, *Patient was found to have sickle cell disease during first admission to Bellevue for H. influenzae meningitis.* (The dropping of the definite article is a regular feature of the laconic style of notes and records.) In the diagram, each linguistic string is written on a separate line, and its point of adjunction in the host string is indicated by a vertical line drawn to that point in the host–string line. The compound nouns *sickle cell disease* and *H. influenzae meningitis* in this sentence are treated as single units, although linguistically they could be further analyzed into noun adjuncts on a host noun (e.g., *disease* of the *sickle cell* type).

Relation to Information

The simplicity of string analysis and its regular rules of sentence composition are strong recommendations for its use in computerized sentence analysis. Another feature, which is important for information processing, is the fact that the linguistic strings are informational units as well as grammatical units of the sentence. Thus, in the example of Fig. 8, the center string *patient was found to have sickle cell disease* is one asserted fact in the sentence. The prepositional strings *during admission, to Bellevue,* and *for H. influenzae meningitis* each add a unit of information to the sentence. Transformational analysis shows further that these prepositional strings are part of a single larger informational unit: the connec-

tive *during* followed by a nominalized form of the assertion: *patient was admitted to Bellevue for H. influenzae meningitis*, another asserted fact in the sentence. The adjective string *first* adds the information that the mentioned admission was the first such event.

Relation to Transformations

The fact that linguistic strings are at the same time grammatical and informational units in the sentence is explained linguistically by the relation of linguistic strings to transformations (Section 2.1 above). The linguistic strings in a sentence are closely related to the elementary assertions and operators in the transformational decomposition of the sentence. Aside from the purely paraphrastic operators, these elementary assertions and operators are the individual informational components of the sentence. In the course of constructing a sentence out of such components, the elementary assertions and operators are transformed (into linguistic strings), and in this form they combine by simple rules of adjunction and substitution[10] to form the final sentence.

When the relation of linguistic strings to transformations is understood, it is not hard to see also why grammatical and semantic constraints apply locally to the components of the sentence under string analysis. Most of these constraints, for example the fact that only certain nouns are appropriate subjects of particular verbs in a given sublanguage, are initially constraints within the elementary component assertions of the sentence. As an example, *patient complained of pain* is an acceptable sentence in the medical sublanguage, whereas *pain complained of patient* is not. In a sentence, this assertion might occur in the form *the complaint of pain by the patient*, having been transformed into a noun with prepositional adjuncts so as to fit into a noun position. However, transformations do not move the parts of the original assertion into arbitrary or distant positions. In the case of a nominalized sentence, the transformed arguments of the verb (here, *of pain, by the patient*) are in adjunct positions near the noun-form of the verb (*complaint*). This makes it relatively straightforward to recover these underlying grammatical relations and apply the appropriate constraints (such as, for example, here, that *pain* is not an acceptable sublanguage subject for *complain*).

[10] A particularly simple statement of the rules of combination is obtained by considering the sentence nominalization strings to have entered the sentence by substitution for a subject or object noun. For example, *I know that he was here* would be formed from two source sentences N-TV-N (*I know something*) and N-TV-ADJ (*He was here*), by nominalizing the second sentence (*He was here* → *that he was here*) and substituting it for the object N (*something*) in the first sentence.

3.2 Representation of the Grammar

As seen above, string analysis is a grammatical theory which has the advantages of simplicity and of providing relevant units of description for information processing. Nevertheless, the number of grammatical facts needed for sentence analysis (in any framework) is such that a very real question is how to represent these facts in a form suited to computation.

In the LSP case, the English grammar is divided into several components each of which is written in an appropriate formalism. The major syntactic constructions of the language, in our case the linguistic strings of the language and certain auxiliary constructs, are specified by context-free productions. For the convenience of parsing, these are written as a set of Backus–Naur Form (BNF) definitions. The parser uses these definitions to construct a parse tree for the input sentence, drawing upon a lexicon which gives the parts of speech and subclass memberships of the input vocabulary. The lexical entries are also written in a BNF formalism.

A second component of the grammar, the restrictions, states conditions on the parse tree which must be met in order for the parse tree to constitute a correct analysis of the input sentence. This component carries the many detailed grammatical constraints (e.g., number agreement and the like) that are not easily expressed in a context-free formalism. The restrictions are procedures which test parse subtrees and attributes of the sentence words as the parse tree is being constructed. If the test succeeds, parsing continues as though no interruption had occurred. If the test fails, the parser backs up and tries to rebuild the parse tree using other options of the grammar. The procedures are written in a special programming language, the Restriction Language (RL), which was developed by the LSP for the writing of computer grammars of natural languages and other language-like systems (Sager and Grishman, 1975).

A third component of the grammar is a set of routines, described in Section 3.4, which are used by the restrictions and also by the transformational component of the grammar. These routines embody the basic linguistic relations of string and transformational grammar as they appear in the parse tree. Their use in restrictions and transformations and in the definitions of the routines themselves shortens the grammar by thousands of lines. The routines are also written in the Restriction Language.

The last, and most recently implemented component of the grammar is a set of English transformations (Hobbs and Grishman, 1976; Raze, 1976b). These are procedures for restructuring the parse tree in accordance with stated conditions so as to eliminate alternative grammatical forms for the same information. The procedures are written in the Restriction Language, utilizing routines of the grammar and operations for

replacing and inserting nodes in the parse tree and for sequencing the transformations under user control.

Lastly, a part of the LSP system that is used in information formatting is a set of sublanguage transformations. These are not part of the English grammar proper. However, like the parsing grammar and the English transformations, they are written in Restriction Language and are a necessary component of the information-formatting system in any application.

3.3 Parsing Program

When a sentence is read into the computer for processing, the program first looks up the sentence words in the computer lexicon and associates with each word of the input string its major classifications and subclassifications. For example, in the LSP lexicon, the major classes associated with each word of the following sentence are those shown beneath the word:

Patient was admitted to hospital for meningitis.

N/ADJ TV TV/VEN P N P N.

(Here, N stands for noun, ADJ for adjective, TV for a verb with tense suffix, VEN for a past particle, P for preposition.) Each major-class entry X of a word may have an attribute list associated with it giving the subclasses of X that the word belongs to; e.g., *patient* as N has the attribute SINGULAR. There are 115 attributes defined for the LSP English grammar (Fitzpatrick and Sager, 1974).

Parsing Algorithm

The syntactic structure is obtained by a parsing algorithm that draws upon the lexical classifications of the sentence words and a grammar which specifies the well-formed sentence structures of the language in terms of the lexical classifications and other defined grammatical constructs. A number of such algorithms have been developed for natural language parsing (Grishman, 1975). The LSP system uses as the core of the analysis algorithm a top-down serial parser with automatic back-up. It builds a parse tree of the input sentence and, if the sentence is ambiguous, generates the different parse trees sequentially. The current implementation of the LSP system in FORTRAN is described in Grishman (1973) and Grishman *et al.* (1973).

Briefly, the top-down parser generates a parse tree from the context-free productions and attempts to match each successive terminal node of the tree with a word class assignment of the current sentence word,

stepping from left-to-right through the sentence. If a terminal node X matches the X category of the current sentence word, a pointer is created from X in the parse tree to X in the lexical entry for that word. The subcategories of X in the lexicon thereby become attributes of the associated terminal node X in the parse tree. They can then be tested by procedures executed on the parse tree (restrictions).

Components of the Program

In the LSP system the algorithm that produces the parse tree(s) for a sentence from the BNF definitions is actually a small part of the total program, as can be seen in Table VII. Out of a total of the 12,721 lines of the program, which does not include the grammar or lexicon, the parsing algorithm proper along with the trace mechanism accounts for 966 lines, or 7.5% of the program. The largest program component is the Restriction Interpreter (2257 lines). Next in size (1417–1001 lines) are the

TABLE VII

LSP PROGRAM COMPONENTS

LSP System (as of August 1977)		
Size of program	12,721 lines	
Size of grammar	3,400 lines	
Size of lexicon (ca. 5,000 English words)	8,000 lines	
Program (including comments)		**Number of Lines**
Largest component		
Restriction interpreter		2257
Next largest components		
Lexical processors		1417
—for compiler	423	
—for English parser	994	
Saving mechanism		1290
Word dictionary update program		1148
Loading grammar (housing, directories, . . .)		1001
Smaller components		
Parser, including trace		966
Main program		902
Code generating routines of RL compiler		887
Transformational mechanism		475
Left recursion		164
Print trees		128
Trace (restrictions, BNF)		included above

lexical processors, the subprogram for saving reusable parse-subtrees, the dictionary update program, and the loading program. The main program and the parsing algorithm proper are thus among the smaller components of the system.

Among the more interesting devices which have been developed as part of the system are the following: a system of interrupts and dynamically generated definitions for conjunction strings; nondeterministic procedures for executing restrictions on conjunctional strings with implicit elements; node attributes with automatic erasure for cross-referencing linguistically related nodes and updating the linkage if the parse tree is changed; switches to control which portions of the grammar are to be used under different circumstances; a mechanism for saving subtrees that keeps track of portions of restrictions which must be re-executed when a saved subtree is inserted into a new parse tree context; a method of distinguishing different types of ambiguity and printing out only alternative analyses of a specified type. These devices are described in a number of LSP papers and reports, chiefly (Sager, 1967, 1973; Raze, 1967, 1976a; Grishman *et al.*, 1973; Sager and Grishman, 1975).

Form of the Output

Figure 9 shows the output parse obtained for the same sentence that was analyzed informally in Fig. 8, namely, *Patient was found to have sickle cell disease during first admission to Bellevue for H. influenzae meningitis*. In this type of output, sibling nodes are connected by a horizontal line and the parent node is attached to the left-most daughter node only; branches end in terminal nodes (e.g., N, TV) or literals (e.g., "to") associated with sentence words or in NULL (not shown). This way of drawing the tree is appropriate for displaying a string analysis of the sentence.

It will be seen in Fig. 9 that certain nodes in the parse tree immediately dominate a sequence of nodes rather than a single element, for example, ASSERTION, PN. These nodes are in the class LINGUISTIC STRING. The sentence words subsumed by a LINGUISTIC STRING type node are those which would be recognized in string analysis as constituting a linguistic string in the sentence. Thus, associated with the terminal symbols under ASSERTION in Fig. 9, one reads the same sequence of words (*patient was found to have sickle cell disease*) as appear on the center string line of the string diagram of Fig. 8. If one reads the words associated with terminal symbols under each PN node (up to but not including the words subsumed by the next PN node) it is seen that each of these word sequences also corresponds to a component of the string diagram of Fig. 8.

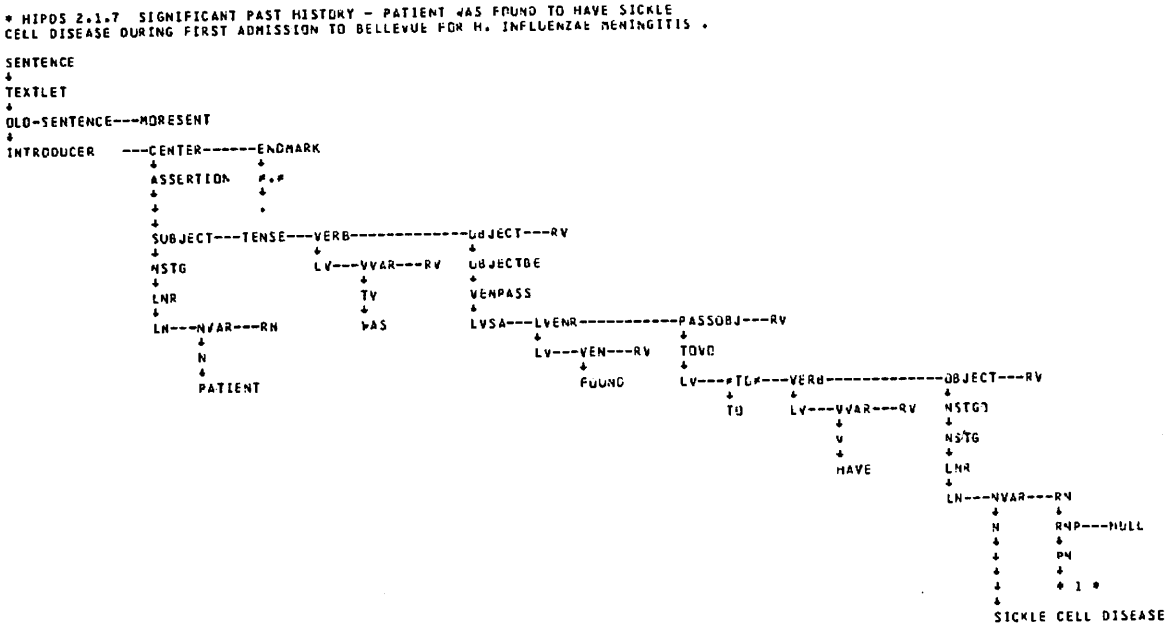
Thus the string character of the analysis, which we saw was important for further information processing, is preserved in the parse tree by the fact that the structures generated by a distinguished subset of the BNF definitions are in direct correspondence with the word sequences constituting the linguistic strings in the sentence. The reason, of course, why the computer-generated parse tree is so much more complicated than the string diagram is that it records every choice made by the parser from among the grammatical alternatives specified by the BNF portion of the grammar. In the case of the LSP grammar the number of alternatives is large since the grammar covers virtually all the sentence forms one is likely to encounter in scientific writing.

3.4 Restrictions and Routines

Just as the context-free parsing algorithm is a small part of the total parsing program, so the context-free portion of the grammar is a small part of the total grammar. Some 200 BNF definitions (about 250 lines) suffice for specifying the syntactic structures of English, exclusive of conjunction strings, which are dynamically generated. This constitutes about 7% of the parsing grammar, and if the transformational component is included in the tally, then the percentage is much smaller. The remainder of the grammar (about 3000 lines, written in RL) consists mainly of routines and restrictions.

Restrictions

As noted earlier, restrictions are procedures that test the parse tree and attributes of the sentence words that have been associated with terminal nodes of the parse tree. In the LSP grammar, for convenience of reference, the restrictions have been divided into functional groups, as shown in Table VIII. Referring to the numbering in Table VIII, Sections 2, 3, 4, 6, 9, 11, 12, 13 are of the type that concern a particular linguistic element or construction. Thus are covered the comma as punctuation, comparatives, coordinate conjunctions, the noun phrase, quantifiers, sentence nominalizations (exclusive of *wh*-complements), tense-verb constraints, and *wh*-strings of all kinds. Sections 1, 8, 10, 14 each concern a particular type of linguistic constraint. Thus, all agreement restrictions are in one section, including those that concern the noun phrase or other constructions named by special sections. The so-called position restrictions check for particular subclasses in particular syntactic positions (e.g., a particular preposition depending on the governing verb). Selection restrictions concern the appropriateness of the combinations of word choices in given syntactic relations. Sublanguage constraints are of



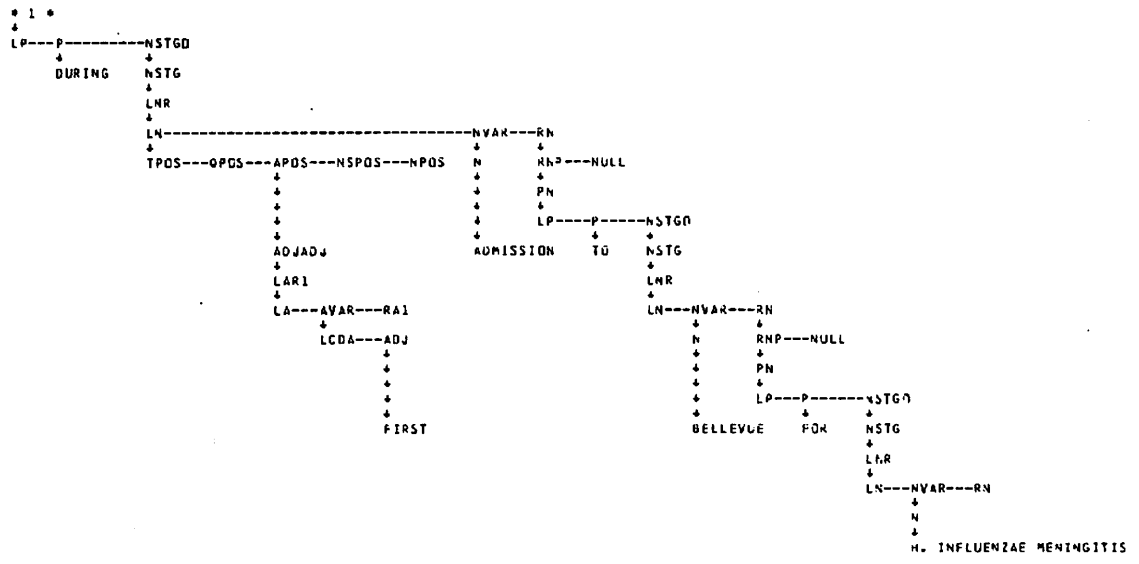


FIG. 9. Parse Tree Output. **Node Names:** *Terminal Symbols:* N noun, P preposition, ADJ adjective, TV tensed verb, VEN past participle. *Types of Non-terminal Symbols:* For X = a terminal symbol: LX *l*eft adjuncts of *X*; RX *r*ight adjuncts of *X*; LXR a sequence of LX + X + RX or of LX + XVAR + RX; XVAR local *var*iants of *X*; XPOS *pos*ition of *X*-occurrence among ordered adjuncts. *Other Node Names*: SA *s*entence *a*djunct; NSTG *n*oun *s*tring; NSTGT *n*oun *s*tring of *t*ime; NSTGO *n*oun *s*tring in *o*bject; PN prepositional phrase; ADJADJ repeating adjectives; LCDA *l*eft adjuncts of *c*ompound *a*djective; OBJECTBE *object* of *be*; VENPASS passive string; LVSA *l*eft adjuncts of *v*erb in participial *SA* string; PASSOBJ *obj*ect in *pass*ive string. **Output Conventions:** A prepositional phrase PN which has several possible positions of adjunction is assigned in the parse tree to the nearest PN slot (here BELLEVUE FOR MENINGITIS rather than ADMISSION FOR MENINGITIS). The later stages of processing correct the assignment on the basis of word co-occurrence classes.

TABLE VIII

LSP ENGLISH GRAMMAR COMPONENTS

Component of grammar		Number of lines
BNF definitions		250
Type-lists		150
Routines		500
Restrictions		2500
1. Agreement restrictions	250	
2. Comma restrictions	100	
3. Comparative restrictions	150	
4. Conjunction restrictions	300	
5. Min-word restrictions	50	
6. Noun phrase restrictions	250	
7. Optimization restrictions	150	
8. Position restrictions	400	
9. Quantifier restrictions	200	
10. Selection restrictions	50	
11. Sentence nominalization restrictions	250	
12. Verb and center string restrictions	100	
13. *Wh*-string restrictions	150	
14. Zeroing restrictions	100	

this type. Zeroing restrictions control the acceptance of NULL elements in conjunctional strings with ellipsis and lay the groundwork for recovering the implicit word-occurrences in these strings. Minword and optimization restrictions (Sections 5 and 7) increase the efficiency of parsing and limit the number of alternative analyses that are printed. A description of many of the restrictions in the form in which they appeared in earlier implementations of the grammar is given in Sager (1968) and Salkoff and Sager (1969). A fuller and more up-to-date documentation of the grammar is in preparation.

From the very start of efforts in computerized language processing it was clear that restrictions, or their equivalent in other terms, were an essential ingredient of the grammar. Parses that are syntactically correct according to the context-free component of the grammar but are clearly not correct syntactic analyses of the given sentence must be eliminated. For a grammar of any respectable size, there may be hundreds of such false parses for sentences even of modest length. The main constraints on the syntactic level that are needed to eliminate false parses are the well-known grammatical rules of agreement in number, case, etc. In addition, for applications, one needs sublanguage constraints to distinguish the intended reading from among alternative correct syntactic parses.

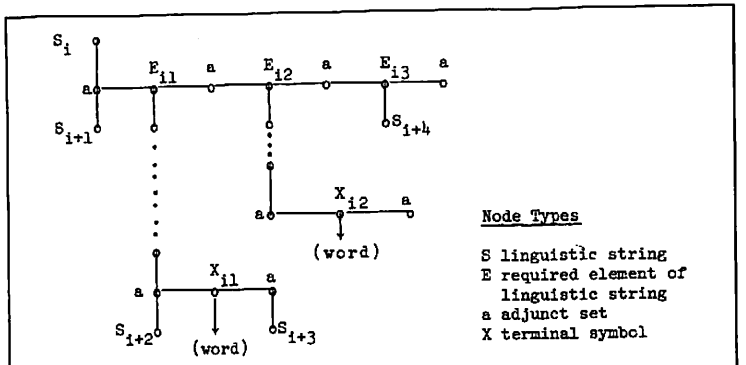
The Need for Global Routines

In applying constraints to putative parse trees the basic operation is to move a pointer from node to node in order to locate the nodes to which the constraint applies. These nodes are then tested to see if they have the desired attributes. By far the largest part of these procedures is devoted to locating the argument nodes of the test to be performed. Even the judicious use of registers does not eliminate the large number of tree-climbing operations that are required in order to cover all the different cases in all the restrictions. When these parse tree paths are specified in detail, the bulk and opacity of the grammar increases until it becomes clear that some more general solution is desirable.

Our solution has been to define a set of global tree-climbing routines based on the linguistic relations of string analysis. The BNF definitions of the grammar are divided into types based on their role in the string grammar, and each type is written in a standard form. This results in a parse tree which has a modular structure. The global tree-climbing routines are then defined in terms of these node types and operate in a standard way in all modules of the parse tree. This eliminates the need to specify the parse tree paths in detail, since all restrictions can be expressed in terms of the relations defined for a standard module.

Modular Structure of Parse Tree

The main types of nodes are LINGUISTIC STRING, ADJUNCT SET, and TERMINAL SYMBOL. A generalized parse tree module illustrating the relations of these types in the parse tree is shown in Fig. 10. A module has as its root a node of type LINGUISTIC STRING (S_i). The BNF definition of a LINGUISTIC STRING consists of a sequence of required elements $E_{i1} \cdots E_{in}$, (such as the SUBJECT, TENSE, VERB, and OBJECT nodes of an ASSERTION), interspersed with optional elements of the type ADJUNCT SET (a in Fig. 10). ADJUNCT SET nodes may be empty (because adjunct string occurrence is optional) or else they terminate in a terminal symbol X or a literal W of the grammar, or, as shown in Fig. 10 under the first a, in a node of the LINGUISTIC STRING type. If the latter, then this node is the root of another module. Required elements E_{ij} of a string S_i originate branches which either terminate in a LINGUISTIC STRING type node (again, the root of another module) or, as is more frequently the case, a three-element sequence consisting



LINGUISTIC RELATIONS

IMMEDIATE STRING	S_i is the immediate string of every node dominated by S_i in the module
ELEMENT	S_i has required elements E_{i1}, E_{i2}, E_{i3} and adjunct elements a
COELEMENT	E_{i1} has coelements E_{i2} and E_{i3}
CORE	E_{i1} has core X_{i1}
	E_{i3} has core S_{i+4}
LEFT ADJUNCT	X_{i1} has left adjunct S_{i+2}
RIGHT ADJUNCT	X_{i1} has right adjunct S_{i+3}
HOST STRING	S_i is the host string of S_{i+1}
HOST	X_{i1} is the host of S_{i+2}, S_{i+3}

FIG. 10. Parse tree module and linguistic relations.

of a TERMINAL SYMBOL X_{ij} flanked on either side by ADJUNCT SET nodes, corresponding to the left and right adjunct sets of X.

To illustrate how parse trees for text sentences are composed of modules, the ASSERTION subtree of Fig. 9 (up to the branch labeled *1*) is redrawn node-for-node in Fig. 11, deleting the node names and labelling those nodes which are of the types noted above. In practice, it has been convenient in some cases to define local variants of a terminal symbol (e.g., pronouns as variants of nouns), giving rise to a type of definition XVAR that is seen in the module in the position usually occupied by X. X or one of its local variants is then the value (i.e., immediate descendent) of the XVAR node. In the sentence of Fig. 9 there are no adjunct occurrences of the type that occur between string elements, and the nodes corresponding to these adjunct set positions are suppressed in the output. Thus no interelement a's are shown in the node-for-node module repre-

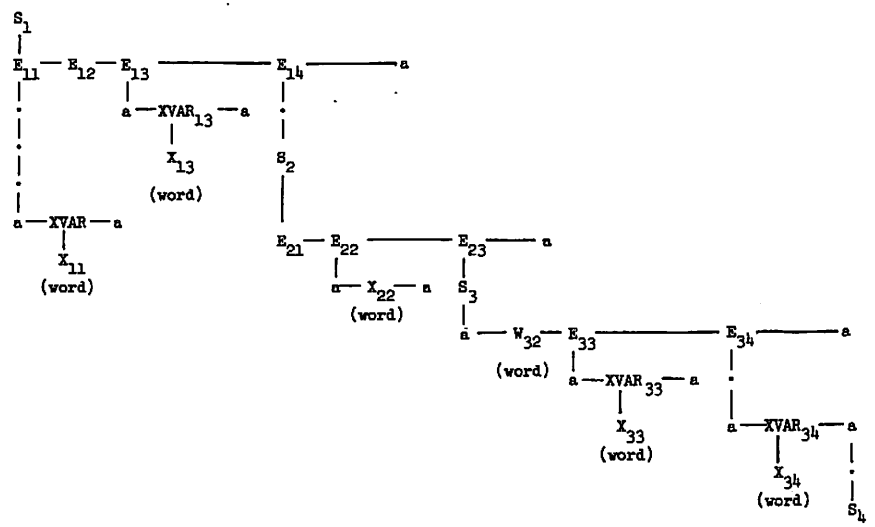


FIG. 11. String modules in the parse tree (ASSERTION subtree of Fig. 9, relabelled).

sentation of the ASSERTION parse tree. Apart from this convention it will be seen that the actual parse tree is faithful to the modular representation.

The node types in a module have a certain few relations in terms of which all linguistic constraints and conditions on transformations can be stated very succinctly. These relations are listed below the parse tree module in Fig. 10 and are illustrated by statements applying to the module. The relations are, in summary, that a linguistic string has ELEMENTSs, each of which is a COELEMENT to the others. Every string element has a CORE value which corresponds to the word or word-string (exclusive of modifiers) which satisfies that element in the sentence. In Fig. 10, the CORE values of E_{i1}, E_{i2}, and E_{i3} are, respectively, X_{i1}, X_{i2}, and S_{i+4} (the numbering of S-nodes is arbitrary). In Fig. 9 the CORE of SUBJECT in the ASSERTION subtree is N (*patient*). A CORE which is a terminal symbol can have LEFT-ADJUNCTs or RIGHT-ADJUNCTs or both. Usually the adjuncts are themselves linguistic strings, (e.g., S_{i+3}, right adjunct of X_{i1} in Fig. 10) and the CORE they adjoin is their HOST. Likewise, a whole linguistic string can have adjuncts (e.g., *usually* in *Usually he goes alone*) in which case the string it adjoins is its HOST STRING. All of the nodes in a module have the same IMMEDIATE STRING, the root node of the module.

Linguistic Routines

Corresponding to each of the relations described above is a routine of the grammar. When executed at a node N of the module the routine locates the node having the stated relation to N. Since the routines are written in terms of the node-types, they apply to any module. For example to find the CORE of any node N (except nodes of types LINGUISTIC STRING and TERMINAL SYMBOL for which the relation is not defined), the routine descends to the first node of type LINGUISTIC STRING or TERMINAL SYMBOL, not entering nodes of type ADJUNCT SET. By invoking the CORE routine at any element position one reaches the word (or word string) which centrally satisfies the element in the given sentence, regardless of how many intervening nodes may have been constructed in the parse tree or whether there are modifiers of that word in the sentence.

Thus, for example, to check agreement between the subject and verb of a declarative sentence, the restriction, executed at, say, the VERB element of the ASSERTION string would invoke the CORE routine to reach the verb that carries the number attribute, the COELEMENT routine (with argument SUBJECT) to reach the SUBJECT node, and again the CORE routine at SUBJECT to reach the noun (or other core element) carrying the subject number attribute. It is then a simple matter to check that the attributes of the core words are compatible.

Different restrictions test modules corresponding to different definitions of the grammar, and even a single restriction like subject–verb agreement applies to a variety of parse tree structures (e.g., different values of SUBJECT, the QUESTION versus ASSERTION string). Yet the same small set of routines suffices to locate the argument nodes for all cases of linguistic constraints and saves the user from having to specify the parse tree paths in detail.

The definition of parse tree modules and the use of higher level parse tree routines based on the relations of node-types in the module greatly simplifies and shortens the parsing grammar. Its most important effect, however, is that it provides regular structures and global operators for the definition of yet more complex linguistic operations. For example, the expansion of conjunctional strings to full assertions (filling in implicit elements) is carried out in a general way based on the modular structure of the parse tree (Raze, 1976b). The expansion procedure locates an element in the host string which is to become the "filled-in" value of the corresponding element in the conjunction string. The ability to accomplish this task in all linguistic cases by means of a single procedure rests on the regularity of the representation of sentence structure.

For another example, the denominalization transformation is a sizeable body of RL code with numerous calls on routines of the grammar. Without the aid of the generalized linguistic routines the code would be many times longer and much more difficult to comprehend. Under such circumstances it would hardly be possible to write a full transformational component of the grammar, nor to write sets of information formatting transformations for different sublanguages.

3.5 English Transformations

The need for a further stage of grammatical processing after a parse is obtained arises because language provides alternative grammatical forms for the same substantive information. In order to obtain a uniform representation of the information, it is desirable to reduce the number of different forms which have to be dealt with and to maximize the repetition of standard types. The most common among equivalent forms, therefore, is chosen as the base form and the program is equipped with procedures which transform occurrences of the equivalent forms into the base forms.

Transformational Decomposition

When the purpose of the sentence analysis is to obtain a decomposition from which tokens of a root word-form can be counted, then a relatively complete transformational analysis is required. The denominalization transformation, as well as many others, must be carried out in full. This is the case in preparing textual input for word cluster analysis as was described in Section 2.4. There, it will be recalled, the transformational decomposition was represented as a hierarchy of operators (mostly verbs and conjunctions) each operating on an n-tuple of arguments which were themselves either operators or concrete nouns (Fig. 7). In this form the number of occurrences of each operator–argument pair can be counted, providing the basis for a calculation of similarity coefficients.

An example of a computer output for this type of sentence decomposition is shown in Fig. 12. The sentence in Fig. 12, *This results from the slowing of the influx of potassium into the cell* is the same one (S1) for which a tree was drawn in Fig. 7. In the computer output every operator with its arguments appears under a node ASSERTION; the operator appears first, as in Polish notation, under the node VERB, with the ordered arguments following, under the nodes SUBJECT, OBJECT, OBJECT··· as needed. The indication that a transformation was performed is given by the presence of a T-node ("T" followed by the name of the transformation) dominating the structure which results from the execution of the transformation. In Fig. 12, the first such node is T-THIS, standing for a transformation which recognized the referential *this* (here in the subject position of *results*) and places a node NULLN under

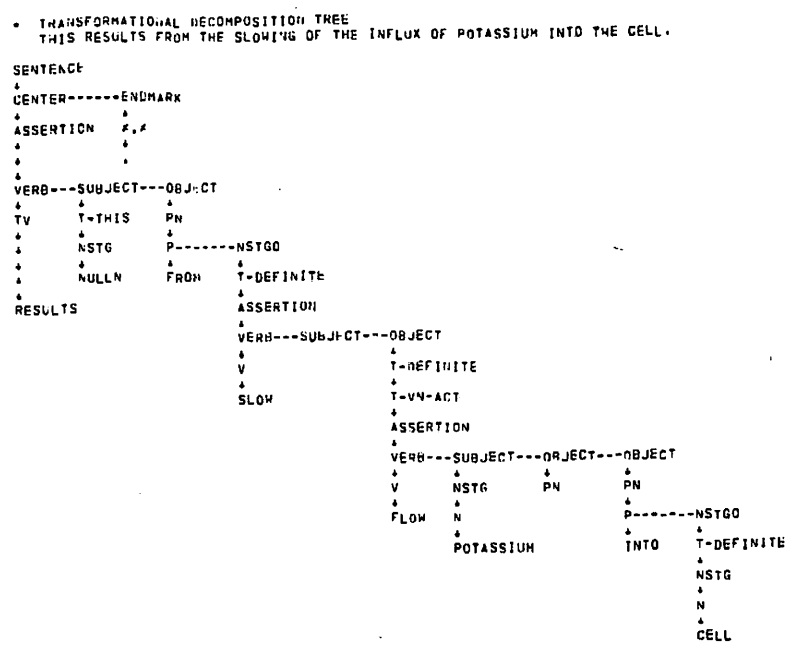


FIG. 12. Transformational decomposition output.

NSTG under SUBJECT to stand for the referred-to noun or nominalized sentence. Further down in the tree, there are two instances of a T-DEFINITE transformation, which records the presence of a definite article, and one instance of a denominalizing transformation T-VN-ACT, which operates on nominal forms of verbs (VN) of the type which undergo "action nominalization."[11] Aside from the form in which the operator–argument relations are presented in the output, the same operator–operand hierarchy which was pictured in the hand-drawn tree of Fig. 7 is seen in the computer output in Fig. 12 for the same sentence. In addition, in the computer decomposition a record is kept of the transformations which operated, since in some cases the transformation carries a certain increment of information.[12]

[11] See Hobbs and Grishman (1976) for a description of "action" and "argument" nominalization types, and for references to the linguistic literature on nominalization.

[12] One transformation which operated on the sentence of Fig. 12 is not represented by a T-node. That is the transformation which converted *the slowing of the influx of potassium into the cell* into an ASSERTION with the verb *slow* and the two arguments: (1) an unstated subject and (2) an object *the influx of potassium into the cell*. The latter is also transformed into an ASSERTION by the denominalizing transformation T-VN-ACT.

Transformational Regularization

The transformational decomposition of sentences is in itself an informational representation of a very general kind. In science sentences it separates the description of the scientists' activities from those of the objects of study and displays the predicational hierarchy. It does this for all sentences, without relying on prior knowledge of sublanguage word classes or sublanguage information structures.[13] However, in information formatting, the transformational decomposition is not the final form of the analysis. Rather, we use transformations to regularize parse tree structures, without changing the information, so that the mapping of the text sentences into a more specific information structure can be done with uniform procedures.

The main English transformations which have been needed in formatting applications to date are the conjunction expansion, relative clause expansion, passive → active transformation, and denominalization. The conjunction and relative clause transformations are important because they make explicit certain implicit word occurrences so as to provide two full assertions in place of one assertion with adjoined material of a different form. Thus, the relative clause transformation expands a sentence like *Was treated with erythromycin which she took for five days* to (in effect) *Was treated with erythromycin such that she took erythromycin for five days*. The second occurrence of *erythromycin* appeared in the original sentence in pronoun form in the word *which*.

In cases where the relative pronoun (e.g., *which*) is not present, the relative clause expansion is triggered by the presence of adjuncts on the noun. Thus, in the sentence seen earlier, *Patient was found to have sickle cell disease during first admission to Bellevue for H. influenzae meningitis,* the presence of adjuncts on the noun *admission* triggers the relative clause expansion, as though the sentence read *Patient was found to have sickle cell disease during an admission which was the first (admission) to Bellevue for H. influenzae meningitis*. Since the noun in this case (*admission*) is a nominalized verb, the final form of the expanded relative clause (as in the format of Fig. 17, below) will show *admission* in the verb position of the expanded relative clause due to the effect of denominalization.[14]

In the medical information format of Fig. 2, the effect of the execution of the conjunction–expansion transformation was seen in lines 7 and 8,

[13] But disambiguation may require the feedback of sublanguage selectional constraints.

[14] Currently, only those portions of the denominalization transformation that identify the arguments of the nominalized verb are executed; the nominal form of the verb is not changed into a verb. Thus, for example, *admission* remains *admission* in the format while its subject and object (i.e., of *admit*) are mapped into the correct argument slots.

where a single assertion containing a conjoined verb phrase (*He was hospitalized and released*) was expanded to two full assertions (*He was hospitalized and he was released*) as a step towards obtaining two complete format lines. An example of the results of executing the passive → active transformation is seen in Fig. 17 below (*find* in place of *was found*, with reversal of subject and object). The effect of the passive → active transformation is also seen in Fig. 2, though there the original passive forms of the verbs are retained in the table for readability.

3.6 Formatting Transformations

As previously outlined in Table VI, text sentences which are being information-formatted are first parsed using the LSP parsing program and English grammar (Sections 3.2–3.4). Then the parse tree structures are regularized by executing paraphrastic English transformations (Section 3.5). The regularized parse trees are then mapped into the information format for the given kind of textual material by means of a set of tree transformations that make use of the sublanguage word classes. These transformations will be described in this section. The final step (Section 3.7) is to normalize the formats by filling out format lines with word occurrences which can be reconstructed by reference to preceding formatted sentences (e.g., by resolving pronoun references). The data base is then ready for use (Section 4).

Information-formatting techniques have been under development by the LSP for several years. Formatting transformations were first written for a corpus of natural language radiology reports (Hirschman *et al.*, 1976). The corpus consisted of 159 consecutive (i.e., not specially selected) follow-up X-ray reports on 13 patients who had had surgery for breast cancer. The reports contained a total of 188 sentential units ranging in complexity from short partial sentences (e.g., *X-rays negative*), to full sentences of moderate length (e.g., *chest film 6-5 shows enlargement of lesion on right hilum since film of 4–17*), with occasional very long run-on sentences that could not always be handled by the parser. To parse the partial sentences, a FRAGMENT definition was added to the BNF grammar. The options of FRAGMENT cover the deletion forms found in notes and records, which, it turns out, are only a few types, mainly due to the dropping of the verb *be* or *show* or particular noun subjects (Anderson *et al.*, 1975). With the addition of the FRAGMENT portion of the grammar and the formatting transformations, the program (parser + English transformations + formatting transformations) successfully formatted 176 of the original 188 sentences (94%).

From the radiology data base, we moved on to consider the more

complex material represented by hospital discharge summaries, the type of document shown in Fig. 1. While this material contains a much greater variety of information and is more varied in style and content than the X-ray reports, it still has the restricted and repetitive features of a sublanguage. This has enabled us to formulate the word subclasses and syntactic regularities that are the basis for the information format for this type of document.

An information format in its entirety is a schematic representation of all the possible grammatical sequences of sublanguage word classes which might be encountered in sentences of the subfield texts. As a rule, we reserve a separate column of the format for each major sublanguage word class, since each such word class has been found to correlate with a particular kind of subfield information. For example, we define a column DIAG to house diagnosis words (*measles, sickle cell disease*, etc.), another column S–S for sign and symptom words, and so forth. This elaboration of format columns facilitates later retrieval. In any one instance of mapping a sentence into the format, only a few of the defined columns will be filled.

The formatting transformations use the same transformational mechanisms that were implemented for the English transformations in the LSP system. Because these mechanisms are designed to map trees into trees, the format is first created as a tree. After the sentences have been mapped into the format trees, the trees can be compressed and written out in the tabular form shown in Fig. 2.

FORMAT Tree for Discharge Summaries

An overview of the FORMAT tree for sentences of hospital discharge summaries (as of Fall 1977) is shown in Fig. 13.[15] An initial transformation sets up this structure and assigns the value NULL to certain nodes, as shown. Later, a number of the NULL nodes are replaced by parse tree substructures that are transferred into the format by the formatting transformations. Only subtrees terminating in non-NULL nodes are printed in the formatting output.

To explain the FORMAT tree a few preliminary remarks on the source of data are in order. We received our corpus of medical documents in machine readable form from the Pediatrics Service of Bellevue Hospital. The information system in use there at the time provided that each type of document in the system (about 70 types in all) would have preset

[15] Figure 13 was provided by Lynette Hirschman, who wrote and tested the formatting transformations for the hospital discharge summaries.

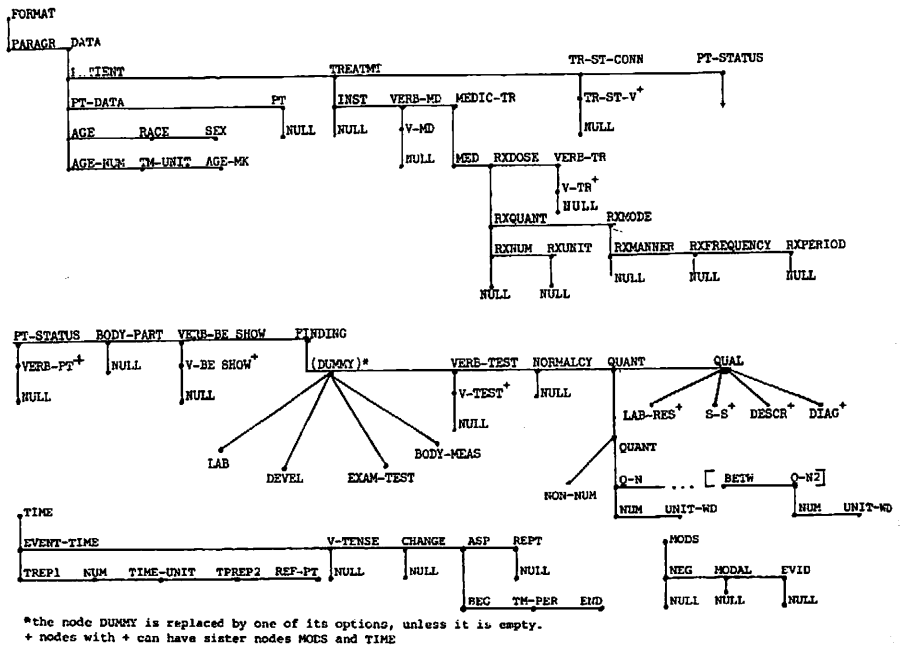


FIG. 13. Format tree for hospital discharge summary.

paragraph and subparagraph headings, within which any amount of free narrative was accepted (Korein, 1970). Correspondingly, the FORMAT tree developed for these documents has at the highest level a node for the paragraph heading (PARAGR) and a sibling node for all the data (DATA).

Under DATA, there are four nodes: PATIENT, TREATMT, TR-ST-CONN and PATIENT STATUS. The material under these nodes identifies the patient (PATIENT), states medical actions taken in treating the patient (TREATMT), states a connection between the treatment given and the patient's status (TR-ST-CONN), and describes the patient's status (PT-STATUS). In some sentences, all four nodes are represented, e.g., *Patient was hospitalized for meningitis*; in others only PATIENT and TREATMENT (*Patient has been followed in Hematology Clinic every three months*); in others only PT-STATUS (*There is tenderness to pressure at the left knee laterally*), etc.

The substructure of each of the main nodes provides slots for all the words in occurrences of the given type, following the syntactic order of

transformationally regularized sentences. Thus, under TREATMT, the node sequence is INST (medical institution personnel), VERB-MD (medical action verb), MEDIC-TR (medication or treatment). These nodes, together with the PATIENT node, which has been moved to the front, correspond to a SUBJECT + VERB + OBJECT + MEANS sequence of a prototype sequence "Physician treats patient with X." Variants of this basic form occur frequently in the material.

Examples of the substructure of the TREATMENT portion of the FORMAT tree are seen in the formatting outputs in Fig. 14. The format for the upper sentence, *Treatment of ampicillin 200 mg/kg I.V. was given* shows the VERB-MD slot filled by *give* (with PAST tense information given under TIME) and the MEDIC-TR slot filled by both a treatment verb (in this case the nominalized verb *treatment*) and a medication noun (*ampicillin*, under MED), associated with a dose (under RXDOSE).

The format for the second sentence in Fig. 14, *A transfusion of 100 cc packed red blood cells was given for extreme anemia* shows MEDIC-TR under TREATMT filled by *transfusion* with its associated information. It also illustrates the use of TR-ST-CONN (*for*) and PATIENT STATUS. Here the nodes under PATIENT STATUS indicate that a FINDING of a QUALitative type, namely a Sign or Symptom (S–S: *anemia*) was recorded, associated with QUANTitative information of a NON-NUMerical type (*extreme*).

Further options of PATIENT STATUS are illustrated in Fig. 15, by the formatting output for the sentence *There was mild mucous discharge from the nose, lungs were clear, grade 2 over 6 systolic murmur along the left sternal border.* This sentence was parsed into two ASSERTION's and a FRAGMENT, and each part was mapped into a FORMAT tree. All three FORMAT trees contain entries for BODY-PART (*nose, border, lungs*).[16] The first FORMAT shows *there be* under the EVIDential node of MODS, associated with the FINDING, whereas in the third FORMAT the verb *be* appears in the VERB-BE-SHOW slot connecting the BODY-PART entry (*lungs*) to the FINDING entry (*clear*). The way *be* is formatted in the two cases accords with the two syntactic roles of *be* in the

[16] A more detailed treatment of *along the left sternal border* would take *sternal* (→ *sternum*) as the entry under BODY-PART, associated with locative information *along the left border*. Words which are syntactically connected to the central word under a format node are placed in associated LEFT-ADJUNCT or RIGHT-ADJUNCT slots according to their position vis a vis the central word, (e.g., in the first FORMAT, *mild mucous* as LEFT-ADJUNCT of *discharge*). But it should be noted that LEFT-ADJUNCT and RIGHT-ADJUNCT in the format have been defined in a manner slightly different from their use in the string grammar, e.g., in the same FORMAT, the LEFT-ADJUNCT *from the* on *discharge*.

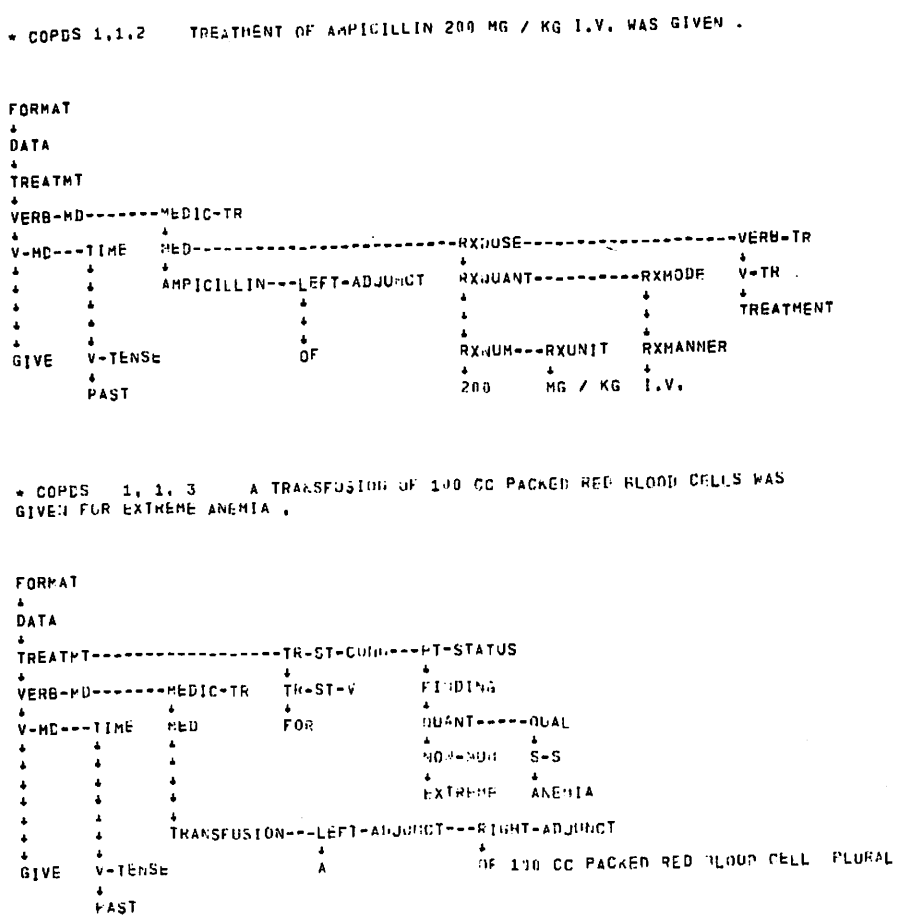


FIG. 14. Formatting output showing treatment subtree of format.

sentence. The FINDING node in the first and third FORMAT trees is, as in Fig. 14, of the QUALitative type; in the first format it contains a Sign–Symptom word (discharge) and in the third a more neutral DESCRiptive term (*clear*). The FINDING node of the second FORMAT tree has a combined QUANT QUAL value, with *systolic murmur* under QUAL and *grade 2 over 6* under QUANT.

FORMAT trees can be connected to each other via a CONNECTIVE node, which was not shown in Fig. 13. The connective may be one of several types. One possibility is a conjunction, such as *when* and *and* as shown in Fig. 2. Another possibility is a relational verb, such as *confirm*, as shown in the formatting output in Fig. 16, for the sentence *Impression*

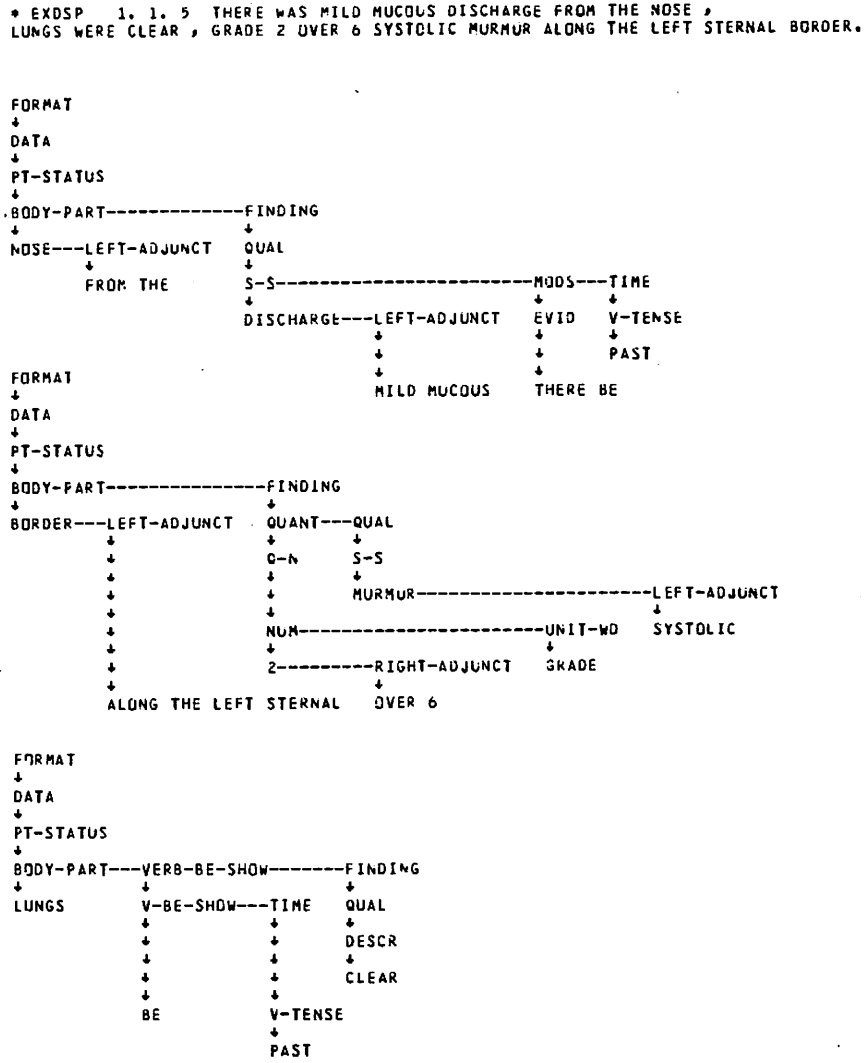


FIG. 15. Formatting output showing PT-status subtree of format.

of meningitis was confirmed by finding cloudy cerebrospinal fluid. Here the CONN node (*confirm*) joins the FORMAT tree for *finding cloudy cerebrospinal fluid* to the FORMAT tree for *impression of meningitis*. The arguments of CONN are written in Polish notation. Also in Fig. 16, the formatting of *cloudy cerebrospinal fluid* illustrates the LAB option of

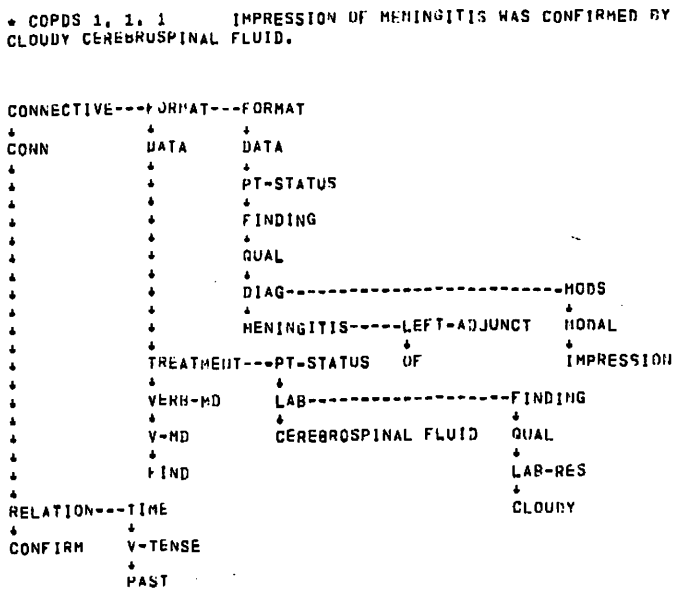


FIG. 16. Formatting output showing connective between two format trees.

PT-STATUS (containing the laboratory material *cerebrospinal fluid*[17]) associated with the QUALitative LAB-RESult (*cloudy*) under FINDING.

Other possible values for the CONNECTIVE node that links FORMAT trees include the marker of an embedded construction (EMBEDDED) and the marker of a relative clause (REL-CLAUSE). Both of these connectives are seen in Fig. 17, the formatting output for the sentence whose parse tree was exhibited in Fig. 9. The first CONN node in Fig. 17 (value EMBEDDED with value in turn NTOVO) joins the FORMAT that contains the medical action verb *find* (V-MD) to the conjunction of two FORMAT trees that contain the rest of the information in the sentence. The CONN node in this latter case has the value REL-CLAUSE with value EXPAND-REFPT which indicates that a word which is a time reference point in the first FORMAT under CONN (here, the word *admission* under REF-PT) is expanded in the manner of a full relative clause in the second FORMAT under CONN. Thus, in the case of Fig.

[17] Presently, as a short cut in retrieval for the frequent case where *culture* is omitted from *cerebrospinal fluid* (or *CSF*) *culture positive* (or *negative*, etc.), these BODY-PART words are also classed as LAB words, and are put in the LAB column if no other test-word is given.

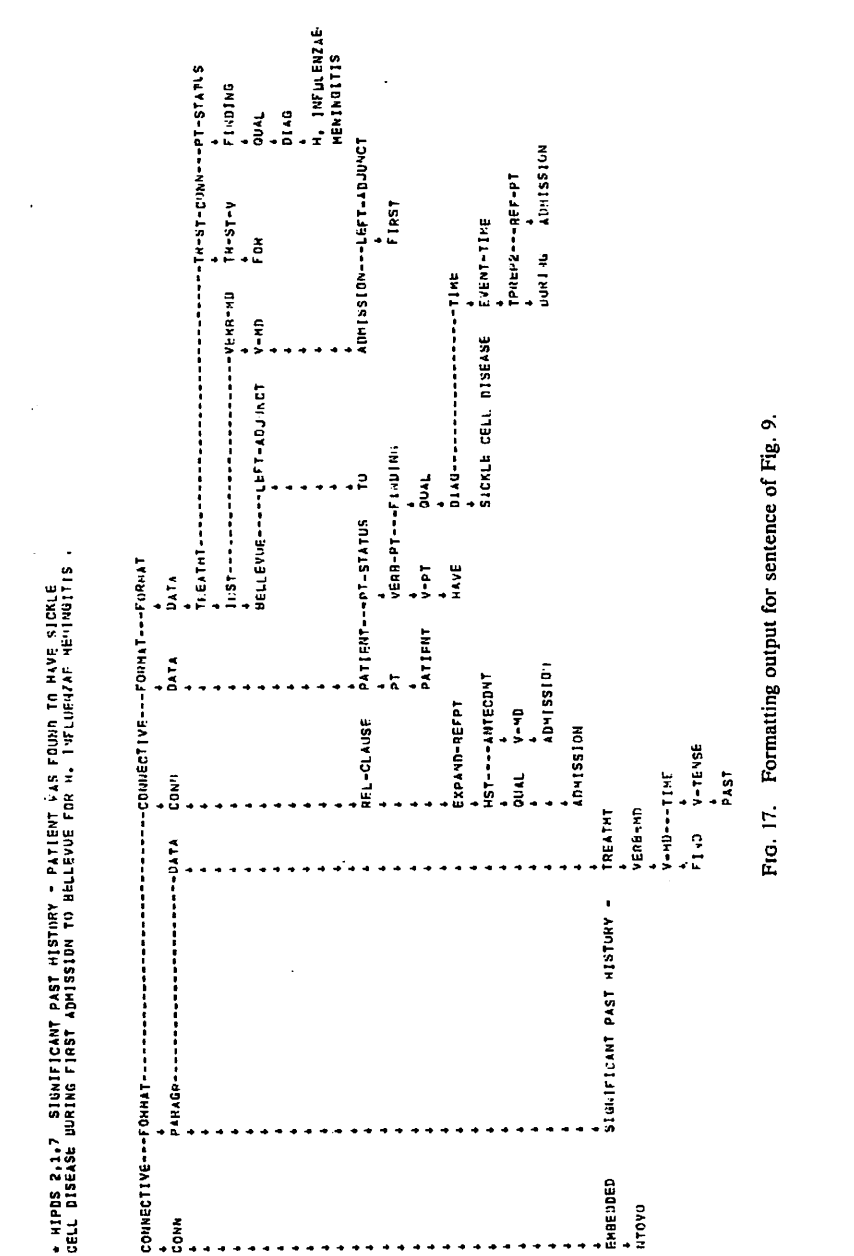


FIG. 17. Formatting output for sentence of Fig. 9.

17, *admission* appears a second time (in the last FORMAT tree), this time not as a time reference word but as a medical action verb, under V-MD, indicating that a medical event (the first admitting of the patient to Bellevue) is also noted in the text. Further details about both CONN and time reference words arise in connection with format normalization, described in Section 3.7 below.

A Formatting Transformation

Both English transformations and information formatting transformations are written in the Restriction Language. RL statements may be in declarative or command syntax, or a mixture of both, as the following fairly typical medical formatting transformation illustrates.

```
T-DIAG = IN LNR, NNN, LAR, LAR1:
  IF BOTH CORE IS H-DIAG AND $CHECK-COOC
  THEN ALL OF $PUT-IN-DIAG, $TFORM-LADJ-RADJ, $ERASE,
$PUT-IN-DIAG = BOTH FIND QUAL ×30 IS EMPTY
               AND BOTH REPLACE VALUE OF ×30
                   BY ⟨DIAG⟩ ×50 (PRESENT-ELEMENT-)
                 AND AT VALUE OF ×50 STORE IN ×51.
$CHECK-COOC = IF ASCEND TO LN
              → THEN HOST-IS NOT H-TEST OR H-RX.
```

The T-DIAG transformation transfers words that are in the hospital-records class for diagnosis words (H-DIAG) from stated positions in the parse tree to the DIAG column of the medical information format, provided certain conditions are met.

The transformation is executed at the nodes in the parse trees which are named following the word IN in the first line of the transformation:

T-DIAG = IN LNR, NNN, LAR, LAR1:

LNR and NNN are noun-centered structures, LAR and LAR1 are adjective-centered structures. The transformation states that if the core of the subtree dominated by the node in question is in the class H-DIAG and if the check on co-occurrence constraints ($CHECK-COOC) for words in this class succeeds, then three steps are carried out: (1) the subtree whose core is H-DIAG is copied into the appropriate slot in the format ($PUT-IN-DIAG); (2) the left and right adjunct subtrees of the core are transformed ($TFORM-LADJ-RADJ); and (3) the parse tree structures of which copies have been made are erased from the parse tree ($ERASE). The latter two procedures (not defined here) are global substatements used in many transformations whereas the $CHECK-COOC

procedure has local scope and is specific to the particular transformation in which it occurs.

In this case the intent of the co-occurrence check is to prevent the placing of a diagnosis-type word in the DIAG column if that word is functioning as a modifier of a test (H-TEST) or treatment (H-RX) word, rather than as a stated diagnosis, as in, e.g., *sickle cell preparation, tuberculosis immunization*. The test is simply stated in RL:

```
IF ASCEND TO LN
→ THEN HOST- IS NOT H-TEST OR H-RX.
```

Translating, this states that if an ascent to LN can be made, which indicates that the H-DIAG noun or adjective is occurring as a modifier of another noun, then the noun which is modified should not be an H-TEST or H-RX word, as in the above examples. The path from the node LN to the noun which is to be tested is given by the routine HOST-.[18]

The steps in transferring words from the parse tree to the information format, in this case via the procedure stated in $PUT-IN-DIAG, are illustrated in Fig. 18, which shows the successive transformations of the relevant portion of the FORMAT tree for the sentence in Fig. 17. The QUAL node under FINDING in the FORMAT tree initially has a NULL value (Fig. 17a). The parse tree for this sentence, shown previously in Fig. 9 has a subtree (Fig. 17b) to which the transformation applies. The first part of $PUT-IN-DIAG

FIND QUAL ×30 IS EMPTY

calls on a FIND routine with argument QUAL which searches the FORMAT tree for QUAL. The remainder of the statement causes QUAL to be stored in register ×30 and tests that the node is empty. The statement

REPLACE VALUE OF ×30 BY ⟨DIAG⟩ ×50 (PRESENT-ELEMENT-)

[18] The HOST- routine is distinguished from the HOST routine by the fact that it does not invoke an automatic re-execution of the restriction or transformation on conjoined structures as would otherwise be the case due to a device called STACKING (Raze, 1976a). Every routine R in the grammar which calls on the STACK operator has a nonstacking counterpart routine R-. The STACKING device is what makes it possible to obtain correct parses for sentences containing truncated conjunctional strings due to ellipsis. In order to obtain a correct parse the truncated string should be expanded so that restrictions can be executed on it, but it is inefficient, if not impossible, to expand all putative parse trees containing conjunctional strings before a correct parse of the sentence is obtained. The STACKING device is an answer to this dilemma. It causes restrictions to be executed as though the conjunctional string were expanded, without restructuring the parse tree, and leaves pointers which make a later expansion (of the correct parse) a relatively straightforward matter.

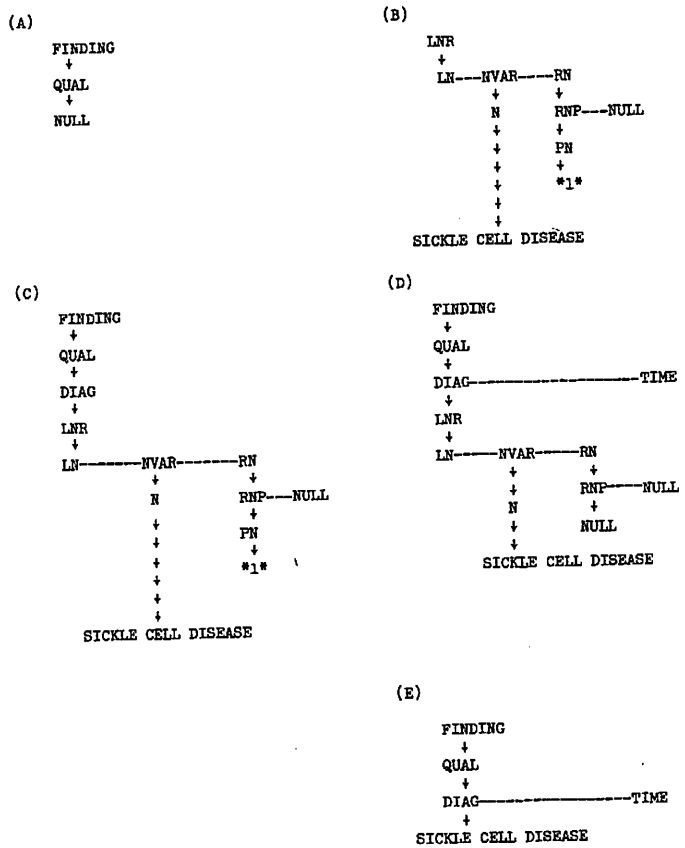


FIG. 18. Operation of a formatting transformation.

causes the FINDING subtree of FORMAT to appear as in Fig. 17c. The node DIAG has been created and appears in place of NULL under QUAL; its value is a copy of the node located by the routine PRESENT-ELEMENT- in the parse tree, which in this case is the LNR node shown in Fig. 17b. Register assignments are also made in this and the subsequent statement for use in the two procedures which conclude the transformation. The procedure $TFORM-LADJ-RADJ transforms the LN and RN nodes of the LNR structure (the RN prepositional phrase in *1* becomes a TIME entry), and $ERASE erases the LNR structure from the parse tree since all its parts have been transferred to the FORMAT tree. The final form of the FINDING subtree is shown in Fig. 17d, and

as it appears in the output (suppressing certain nonessential nodes) in Fig. 17e.

3.7 Format Normalization

The last step in preparing a structured data base from input texts is to complete each line of the format table (each FORMAT tree) with information which can be supplied by reference to other format lines of the data base. For example, in the data base of radiology reports referred to earlier, each format line had slots for the type of test, the location of the test, the date of the test, the medical finding, etc. In formatted sentences where only the finding slot was filled, but the preceding sentence of the report named both the test and a finding (e.g., *Chest films 10–15 RLL infiltrate clear. LUL scarring still present.*), it could be inferred that the format line for the second sentence should contain the same test information as its predecessor (e.g., in this case, *chest films 10–15* to be associated with *LUL scarring still present* as well as *RLL infiltrate clear*). Also, in the radiology data base, missing test locations and test dates could sometimes be supplied from the context of formatted sentences. The procedure for filling out the format in this way was called format normalization and is described in application to the radiology data base in Hirschman and Grishman (1977).

In the data base of hospital discharge summaries, the major format normalization problem is to assign to each format line a unique event time. This is important not only because the time of occurrence of an event is essential medical information but because in searching the data base we must be sure not to consider a reference to a previously noted event to be the record of a second event of the same kind. For example, if the report notes that a transfusion was given and later describes events that were "prior to the transfusion," we must not mistakenly take the two mentions of transfusion to indicate that there were two transfusion events. In this example, the referential status of *transfusion* in *prior to the transfusion* is clearly indicated by the presence of the definite article (*the* = the previously mentioned), and the occurrence of the word in a time prepositional phrase tells us explicitly that it is being used as the time reference point for another event. In other cases, the analysis is more complicated.

Representation of Time in the FORMAT

Time expressions can be divided linguistically (with some overlap) into those that specify or refer to the location in time of an event (e.g., *yesterday, 10-3-66, on admission*) and those that describe the time aspect

TABLE IX

STRUCTURE OF EVENT-TIME

TPREP1	NUM	TIME-UNIT	TPREP2	REF-PT
			on	admission
	one	day	prior to	admission
				10-24-72
			at	present
since	several	days	before	visit (*the visit to the Emergency Room*)
			at	age 2 years
			until	age 5
for	a	month		
			during	admission (*the first admission*)

of the event, i.e., its duration, frequency, way of beginning or ending, etc. Accordingly the TIME portion of the medical information format (bottom section, Fig. 13) consists of an EVENT–TIME node and an ASPectual node, with several sibling nodes that contribute to either or both types of information (V-TENSE, CHANGE, REPetition). TIME subtrees can be attached to any node which carries a verb (V-MD, V-PT, V-BE-SHOW, V-TEST, TR-ST-V, and CONN in certain cases) and independently to FINDING nodes (LAB-RES, S–S, LAB-RES, DIAG). The TIME subtree is only created when a time expression is present.

Within EVENT–TIME the ordered slots correspond to the word order in the most common form of event–time expression, the prepositional phrase, e.g., *on admission, one day prior to admission,* etc. As illustrated in Table IX, the noun under REF-PT provides the reference point; the immediately preceding preposition (TPREP2) tells whether the event time is before, at, or after the REF-PT time; the sequence preceding this preposition (if present) provides data for a quantitative adjustment of the REF-PT time in the direction given by TPREP2.

Time Normalization Problem

The time normalization problem is twofold. Given a complete statement of the event time in terms of a given nonreferential REF-PT the event time which is expressed in words should be translated into an absolute time. This is not difficult when the REF-PT is the current *admission, discharge,* or *hospitalization,* since these dates are supplied in the document header. Nevertheless, the adjustment of the REF-PT in accord with an analysis of the prepositional phrase(s) in EVENT–TIME is not trivial.

By far the harder problem is to assure that every format line has a REF-PT and to find the antecedent REF-PT when the given REF-PT is referential. The algorithm which is being implemented[19] has the following general features. It proceeds in a forward direction from format line to format line to determine the correct event–time for each format line. It first examines the EVENT–TIME entries and if sufficient information is present performs the necessary calculations. If no REF-PT in EVENT–TIME is given, the AGE column in PT-DATA is examined; e.g., in the occurrence *a well developed, well nourished one-year old female,* the time associated with the information *well developed, well nourished* is given indirectly by *one year old.* If no AGE entry is present, the algorithm looks at TENSE, since some tenses (e.g., perfect, future, progressive) provide time information of a general kind. Then PARAGR is examined, since some paragraph headings give time information (e.g., EXAMINATION ON ADMISSION, STATUS AT DISCHARGE).[20]

If the event–time cannot be established by the above steps, then preceding format lines must be consulted. If the REF-PT entry of a format line is empty then, again, the paragraph that the format line occurs in is important. In narrative paragraphs, such as HISTORY or COURSE IN HOSPITAL, it can be assumed that unless specific time information is given, the time of each successive statement is the same or later than that of its predecessor (ignoring intervening subordinated format lines). In the case of subordinated format lines, e.g., those which are conjoined to a previous format line by a relative clause connective, it may be assumed that the event time of the subordinated line is the same as that of its host line unless contrary information is given.

Where the time is given relative to a particular event (e.g., *at admission, two weeks following the last transfusion*) we try to establish a (more) exact time by locating in one of the preceding format lines the original mention of the event referred to. Since time normalization proceeds on the individual format lines in a forward direction, this original mention will have had its time normalized. There will therefore be a time associated with this event that can be used to establish the time of the subsequent format line in which the reference occurs.[21]

[19] The design of the time normalization algorithm described here and its implementation are the work of Lynette Hirschman.

[20] In EXAMINATION and LAB paragraphs, the tense employed does not give time information.

[21] Although one would expect in normal discourse not to find a reference to an event without there having been a prior mention of the event in the discourse, this sometimes occurs in this material; e.g., *posttransfusion hematocrit,* with no earlier mention of *transfusion.* Fortunately, such occurrences are rare.

When the reference is to the current admission, hospitalization or discharge, there is no need to search for the original mention since the exact dates for these events are given in the header information. A search for an antecedent is required in the case where the REF-PT word is not one mentioned in the header information, or in the case where such a word is mentioned but the adjuncts of the word indicate a hospital stay other than the present one (e.g., *the previous hospitalization, the admission for septicemia*).

The adjuncts of the event in the time expression determine the search strategy. There are four principal cases:

(1) *n*th: The adjunct or the associated REPT (repetition) column is an ordinal: *first, second, 3rd,* etc. The search for the *n*th event begins at the start of the subsection or paragraph where the reference occurs. It counts mentions of events of the appropriate type until it gets to *n*. It makes use of the adjuncts and the normalized time on each mention to distinguish mentions of the same event from all others, and to aid in counting.

(2) Same: The adjunct or associated TPREP2 of EVENT–TIME has a word indicating "sameness" of reference, e.g., *this transfusion, the same visit*. We search backward to the most recent mention of this event (where the time has already been normalized).

(3) Previous: TPREP2 of the associated EVENT–TIME has a word of type "previous" in it (*previous admission, the earlier transfusion*). We search backwards for the most recent mention of the appropriate event that is not a mention of the same event; "same" events will have the same normalized time, whereas distinct events will show a difference in time (at least relative to other events).

(4) Subsequent: TPREP2 of the associated EVENT–TIME has a word of type "subsequent" in it (*the next admission, the repeat determination*). The time information that these words provide is that the event occurred *after* a previous occurrence. Therefore we search (as in (3)) for the previous event and note the time as subsequent to this event—unless we have a future tense or a section heading indicating a future event (PLAN ON DISCHARGE, RETURN APPOINTMENT, etc.)—in which case we mark the time as *post discharge*.

Once the type of search is determined (*n*th, Same, Previous, Subsequent) and the word to be searched for is identified,[22] the search for an antecedent begins. When a candidate word is found it must be established

[22] Details of the identification procedure are omitted here.

that the word is not negated or otherwise modified in a way that disqualifies it as an antecedent. Therefore each time the procedure finds a format node with the word (or synonym of the word) that it is searching for, it checks that the word is not in the scope of negation (NEG in MODS on the noun or on an associated verb) and that it is not in the scope of a "nonreal" marker (e.g., *future, possible* in INDEF in MODS). Finally the procedure checks that the time of the proposed antecedent is consistent with the time information established so far: e.g., Format F_n has TIME t_1 associated with it; format F_{n+1} has a REF-PT (e.g., *previous transfusion*) in its TIME slot; the antecedent is in F_{n-3}, with Time t_0. To be consistent with the information so far, $t_0 \leq t_1$ (otherwise F_{n-3} has a time later than F_n, a very unlikely occurrence in a linearly progressing narrative).

These checks ensure that the event chosen as an antecedent will have the appropriate properties: it refers to the desired *type* of event; it will be *unnegated, real,* and have a time associated with it that is consistent with the accumulated time information so far.

3.8 Performance

In this section some preliminary results of the automatic formatting of discharge summaries will be presented. Table X summarizes the performance of the parsing and information-formatting components of the program on the first small set of documents received from Bellevue Hospital.[23] The nine documents were divided into two sets. The first five were

TABLE X

SUMMARY OF PRELIMINARY RESULTS

Parsing results	1st 5 documents	2nd 4 documents
Number of sentences	261	198
OK for formatting	88%	90%
Wrong parse	4%	5%
No parse	8%	5%
Average parsing time	3.5 sec	less than 1.5 sec
Formatting results	**1st 5 documents**	
Correct format	85%	
Wrong format	0%	
No format	15%	

[23] The parsing results in Table X are from Insolio and Sager (1977).

used mainly as a source of sentences to test and debug the additions to the system for medical records. The second set was run in a more routine manner, in batches, as a test of the system, though still on a very preliminary basis.

The top half of Table X gives parsing results. The category "OK for formatting" arises because the test of success in parsing in this application is precisely that the parsing output should be such that when the English and formatting transformations operate on it they produce correctly formatted sentences. Thus, some parses which are syntactically correct but do not display the intended reading of the sentence (e.g., a prepositional adjunct is shown as modifier of a near noun when in the intended reading it should be a modifier of a more distant noun or of the verb or of the whole sentence) are nevertheless adequate input to the transformations, since the transformations compensate for a certain number of such syntactic variations. The denominalization transformation, for example, hunts for its arguments in all possible neighboring adjunct slots of the nominalized verb and thus compensates for the ambiguity of adjunct PN's. Also, the formatting transformations require that words which are to be moved from the parse tree to particular format slots have particular sublanguage attributes, thereby supplying in effect a selectional (i.e., semantic) constraint on the parse tree output.

As a result of this interdependence of components, demands on the parser are to some extent lessened. However, because the parsing is only part of a larger process, there is a contrary pressure to produce a useable output as a first parse. (The figures in Table X are for the first parse only.) Formerly we were satisfied if the intended analysis was among the first three parses obtained. But in information–formatting, we want to avoid having to return to the parsing stage to obtain a second or third parse once the formatting has begun. We therefore attempt to order the options of the grammar and add selectional constraints so that a useable analysis is obtained as the first parse.

In unedited note-based material there are further difficulties in obtaining a correct parse due to the variety of incomplete sentence constructions and the many functions of the comma: punctuation, conjunction, end-of-sentence marker, deletion-marker, and sometimes no function at all, leading to misinterpretation. Special constraints of various kinds are needed, and a certain failure rate is to be expected and should be used to signal the need for either editing or manual analysis. In a routine application of the program to documents, one would expect to coordinate document analysis with data capture so as to avoid spending program time on problems that are not germane to the narrative analysis. In

medical records this means mainly establishing conventions for punctuation and for reporting laboratory data and medications.

With regard to the formatting results, a relevant question is whether failure signifies inability to format a sentence or the obtaining of a wrong result. At a later time a reliability index, such as the ratio of incorrect to correct analyses, can be measured. At this stage, since we are aiming at a zero or near-zero value of such an index, every wrong result is used as a signal to tighten the formatting constraints. In fact, there have been few such cases, partly because the parser screens out certain kinds of deviant sentences, and partly because the formatting transformations have been purposely written with tight constraints from the start.

With regard to processing times, figures are presented in Table X as a part of the report on processing results, but it should be borne in mind that the programs exist in a research environment and have not been designed for efficient routine operation. The figures on processing times thus give little indication as to what could be achieved with an optimized, application-tailored system, and, like the other numerical data in Table X, are offered in the spirit of informing the reader of the current state of an experiment rather than as a measure of performance of a finished system, it being too early in the development of the system to make such a test.

With all the limitations on the data of Table X noted above, one conclusion stands out: Automatic information–formatting can be done. A crucial fact is that the program and grammar were not written for these particular documents, nor even, except for a relatively small addition to the system, for this subject matter or type of document. The English parsing grammar, for example, which is the crucial element in the successful processing, was implemented (in three successive versions) and used to parse texts well before its application in information–formatting was envisioned. Of course, many problems remain to be solved, and the application to a new subject area would require a considerable tooling-up effort. Nevertheless, the novel notion that freely written text in delineated subject areas can be automatically converted into a structured data base is being demonstrated by this experiment.

4. Applications

If indeed it is possible to obtain structured data bases from textual material then the potential for new computer applications is large. In this section a few applications are sketched with reference to how they would

be realized on a data base of computer-formatted medical records. Analogous functions for other types of documents can be inferred. Thus, quality assessment by computer is illustrated below by the application of criteria for the evaluation of medical care to information-formatted hospital documents, but it is equally possible to imagine quality assessment being carried out on formatted documents in a different subject area by the application of criteria that are appropriate to that data base.

In all the applications, the essential point is that the information in sentences of the documents is arranged in labelled columns of a table in such a way that assertions of a factual kind can be recognized (rather than the occurrence of a term apart from its context) and that the co-occurrence of particular features can also be tested for, permitting complex informational queries to be answered. A retrieval program operating on formatted discharge summaries, for example, can establish not only that a given diagnosis word was mentioned, but that a diagnosis was asserted. And it can, if requested, also supply the age at which the diagnosis was made, the method by which the diagnosis was confirmed, the symptoms that were manifested, etc., all by reference to the entries in the relevant columns of the format.

4.1 Fact Retrieval

In the information field, the ability of a system to supply a user with specific information in response to an information request, as opposed to a complete document or citation that may be relevant to the request, is sometimes referred to as "fact retrieval" in contradistinction to "document retrieval" or "bibliographic retrieval." By linking the natural language information–formatting system described above to a retrieval program that can search the format columns of the data base for particular combinations of entries, we achieve a fact retrieval program that operates on natural language information. Such a program might be used interactively to obtain quickly some particular item of information from a document somewhere in the data base. Or it might be used retrospectively along with a counting and summarization procedure to obtain a profile of the contents of the documents in the data base with respect to particular variables. Examples of applications of this type have been demonstrated using the data base of automatically formatted radiology reports referred to earlier (Hirschman and Grishman, 1977; Sager *et al.*, 1977). A natural language question–answering program that operates on this type of data base is also being developed (Grishman and Hirschman, 1978).

Reuse of Information

A hypothetical situation illustrates an important feature of the information-formatted data base. Suppose, as has happened a number of times in recent medical history, some element in the past environment, treatment or habits of patients later becomes suspect. It could be desirable to search retrospectively in a given population for those whose medical history might have mentioned such facts, even though the facts may not have been considered particularly important at the time they were recorded. Such items might be the taking of a particular drug (e.g., a fertility drug considered harmless at the time), employment in a particular industry, amount of smoking, or the like. The purpose of the search might be to identify the high risk population so that an attempt could be made to contact the individuals, or it might be to establish a correlation of the implicated factor with later symptoms, or to provide background for a policy decision such as whether a mass screening of a segment of the population would be worthwhile.

What is significant here is that the perspective with which the recorded facts are viewed has changed. Most systems for storing textual information are based on a selection of content items which serve to identify the source documents and provide the points of access to the documents in the future. In a data base built up for research purposes, these selected content items may actually constitute the coded data base, i.e., they may be all that was asked for on a check list, or all that was kept when information was extracted from a set of source documents. In either case, there is very little room for reinterpreting the data, for accessing it from a different point of view, or, in economic terms, for amortizing the data base by making the information available for reuse in different contexts.

In contrast, the data base built up by information–formatting has the feature that it is neutral with regard to the purpose for which the information is to be used; and it is relatively complete with regard to the information originally contained in the source documents. Though it would undoubtedly be expensive to create such a data base on a large scale, that expense would have to be weighed in any real situation against the expense of acting without the information that such a data base could supply.

Case Matching

Given a large enough corpus of information-formatted medical documents, another application which would make use of the refinement in

search criteria that is possible with such a data base is to locate cases which can serve as a control population in clinical studies where ethical considerations preclude setting up a controlled experiment. It might be possible to match cases in the data base with regard to a sufficiently refined set of features so that for each patient who is being treated in a given manner in the existing study a "similar" patient who was not treated in the given manner can be located to serve as a control. Again, here, as in the former sketched application a large data base is assumed. While this may be far in the future, the example illustrates how a data base of information-formatted documents can be used retrospectively in different ways for different purposes.

4.2 Quality Assessment

As an example of fact retrieval from formatted discharge summaries, the LSP is engaged in showing that some types of questions that arise in assessing whether proper medical care was administered in a hospital can be answered by computer programs operating on information-formatted discharge summaries (Sager and Lyman, 1978). While there is discussion in the medical community as to what criteria should be used to evaluate care (see, e.g., McDermott, 1975; Brook, 1977), certain elementary questions regarding the procedures that were carried out and the reasons for doing them are likely to be asked in many evaluation frameworks. To the extent that these questions could be answered automatically by reference to the formatted documents, the human investigator would be freed from routine screening of documents to turn attention to cases that are not of a routine nature. The automation of the steps in a health care audit of hospital documents could also contribute to the study of different sets of criteria for evaluating the process of health care in relation to the outcome of the process in terms of changes in the patients' health. The number of cases that could be examined would be increased, and once the data base was established, the different sets of criteria could be applied, at least in part, simply by changing the retrieval queries applied to the data base.

Applying Audit Criteria to the Data Base

Table XI shows the first 3 of 14 audit criteria for meningitis and/or septicemia in sickle cell disease, prepared for use in the Performance Evaluation Procedure (PEP) forms of the Joint Committee on Accreditation of Hospitals ("PEP Primer," 1974). A LISP program is being written by the LSP to apply the full set of criteria to information-formatted discharge summaries. It is not difficult to see how the retrieval

TABLE XI

EXCERPTED AUDIT CRITERIA

Diagnosis of meningitis

1. Positive CSF culture (or A + B)
   - A. Admission history contains all of the following:
     - (1) Fever
     - (2) Stiff neck
     - (3) Vomiting or headache
   - B. First CSF shows 2 of the following:
     - (1) Positive smear
     - (2) WBC greater than 10/cmm
     - (3) Glucose less than 30 mg%
     - (4) Protein greater than 40 mg%

Diagnosis of septicemia

2. Positive blood culture

Diagnosis of sickle cell disease

3. One of the following:
   - (1) Positive sickle cell preparation
   - (2) Hemoglobin electrophoresis = HgS
   - (3) Statement in history "known sickler" or equivalent

procedure works if one keeps in mind the format structure for discharge summaries, described in Section 3.6 above. To illustrate, the logic for applying the criteria for a diagnosis of meningitis (item 1 of Table XI) will be described.

According to the criteria in Table XI, a diagnosis of meningitis is considered to be correctly established if there was a positive CSF culture or if the criteria stated in A and B under 1 are met as indicated. To establish that a positive CSF culture was reported, the retrieval procedure searches the LAB column (under FINDING under PATIENT STATUS) for *CSF* or *cerebrospinal fluid,* and if it finds such an entry, looks under the QUAL node of the same FINDING for LAB-RES, which should contain *positive* or a synonym of *positive* in this context. (The synonyms to look for are supplied by medical personnel in the form of directives to the clerks who presently carry out the manual procedure.) The list of synonyms of *positive* in this case includes the word *cloudy*. Thus, for example, the formatted sentence in Fig. 16, which shows LAB = *cerebrospinal fluid* and LAB-RES = *cloudy*, is an example of a formatted sentence that meets the test for a diagnosis of meningitis, up to this point.

If the test of the LAB and LAB-RES columns in FINDING is suc-

cessful, two more tests must be made. It must be established that the finding is not negated, and that the time of the finding is within the present hospitalization. The information as to whether the finding is negated or not is found in the columns NEG and MODAL under MODS. The entries in NEG indicate negation (*no, not,* etc.) and those in MODAL some indefiniteness (e.g., *impression*). Hence to establish a definite finding both these nodes should be empty. Syntactically, negation and modal information may attach to the finding itself or to a verb governing the finding (e.g., *It was not established that . . . , The culture did not grow out anything*). Therefore the MODS subtrees in PATIENT-STATUS or associated with V-MD are all examined for the presence of NEG or MODAL entries.

To establish that the time of the finding is within the hospitalization covered by the discharge summary the EVENT–TIME entry in the TIME subtree is examined. As in the case of the MODS, above, the TIME subtree may be associated directly with the finding (in PATIENT-STATUS) or may be associated with a verb (V-MD). The time normalization procedure will have supplied an EVENT–TIME entry to every format line during the last step in preparation of the data base (Section 3.7). For example in the formatted sentence in Fig. 16, where no event–time is stated, the fact that the sentence occurs in the paragraph COURSE IN HOSPITAL (indicated by the first two letters CO of the serialization) provides the normalization procedure with a default event–time entry "during this hospitalization."

Thus, in regard to criterion 1 of Table XI for the diagnosis of meningitis, the retrieval procedure finds that the sentence whose formatting output is shown in Fig. 16, gives a positive answer: a positive CSF culture was reported, with no negation or doubt expressed, having been obtained at a time within the hospitalization described in the document.

An alternative basis for the diagnosis of meningitis is given in 1A and 1B of Table XI, since the lack of a positive CSF culture does not rule out the possibility of meningitis. Criterion 1A states characteristic symptoms and 1B states significant laboratory findings. These are treated by the retrieval program in a similar manner to the CSF-culture criterion, above. In the case of 1A, S–S (Sign–Symptom) and BODY-PART columns are examined for the appropriate entries; then MODS and TIME are examined, the MODS to check that there is no negation or doubt (as above) and the TIME, in this case, to test for a time "at admission" (or within 48 hours of admission).

The criteria in 1B are applied by first testing the columns LAB and QUAL: LAB-RES (in a manner similar to the case above) for the qualitative finding 1B(1) "CSF shows positive smear," or LAB and QUANT

for the quantitative findings 1B(2)–(4) regarding the white blood cell, glucose, and protein determinations. Note that the criterion specifies "first CSF." To establish that a given CSF finding is "first" the search strategy for the *n*th event, described in Section 3.7 above, is applied to the document.

4.3 Data Summarization

Since, after information–formatting, the information in a document collection is arranged in columns that are labelled as to the type of information they contain, it is possible to count the number of occurrences of the different types of information and generate statistical summaries covering the contents of the data base. In situations where each document corresponds to a case treated by the institution (as in medical records) such summaries could provide those who are responsible for policy decisions with a current and cummulative profile of the types of cases that are entering the institution and the outcome of actions taken in each type of case.

In the case of a hospital record system such data might help to identify patient management problems and could be used to provide a summary of the hospital's experience as background for staff conferences and other activities in the area of continuing medical education. Continuing medical education, i.e., formal, systematic and on-going review of diagnosis and treatment and preventive health care—has become an important element in all efforts to improve the quality of medical care. Government legislation and regulations require that hospitals receiving Federal reimbursement for patient care monitor the quality of care rendered and take appropriate action when deficits are identified. One such action is to hold a teaching session or Hospital Staff Conference.

Teaching conferences commonly have a standard format: one or more case presentations, a review of the hospital's own experience in the subject of the conference, and a discussion which draws heavily on the experience of others as reflected in the medical literature. The use of a case presentation as a focal point for review seems well established in medical training programs.

There is a heavy burden of work on the persons conducting the conference, even for the specialist who is concentrating his reading in relatively small areas and keeping records of cases which have come to his attention. Most often it is a brief, intensive effort and rarely includes a complete review of the hospital's own cases. The number of such formal conferences that can be given appears to be as much limited by the work involved as by the time available to the staff for attending conferences.

In this context, the availability of an automated method for analyzing patient case reports could make a significant difference both in the number of staff conferences that would be possible and in the relevance of the contents of the conference to ongoing patient care in the hospital. Suppose, for example, that a conference is to be devoted to the treatment of bacterial meningitis. Discharge summaries for cases of bacterial meningitis treated in the hospital over a certain period could be information-formatted and a summary of features of these cases could be generated in response to questions posed by the person preparing the conference. Such questions might include: How many patients with this diagnosis stayed longer than two weeks? In this set, what were the signs or symptoms or additional diagnoses reported? In how many cases did the CSF culture show no growth? Of those patients who did not have positive CSF cultures, what were the symptoms on admission? How many patients were on antibiotics before diagnosis was established? For how many days? How many patients received antibiotics intravenously (for how many days?) and how many orally? How many patients were released with no residual sequelae?

These questions, and others, can be answered from the formatted discharge summaries, by a retrieval program similar to the one developed for quality assessment (Section 4.2). The answers to the questions can then be tabulated to provide a summary of the hospital's own experience in the form of a table as a basis for discussion at a staff conference or other forum. The hospital's performance can then be compared with that of other institutions as reported in such sources as "Medical Clinics of North America," and with up to date clinical review articles in the area. It is even conceivable that questions that arise during such a conference could be answered if an on-line query capability were added to the fact retrieval and data summarization system.

4.4 Data Alert

If the information–formatting process is further developed so that it can be performed routinely within 48 hours of the receipt of the source document a valuable service would be to alert the data collector of deficiencies in the source document. In many research situations much good data must be discounted at the end of the study because certain items are found to be missing which invalidate the inclusion of the case in the final summary. For example, in a long-term drug study if the patient was to have responded at periodic intervals to a particular question, but at various times in the study no response to the question was recorded, this may invalidate the patient's inclusion in the final tally of

responses. In a large expensive long-term study, it might be worthwhile to monitor the incoming data soon after it is recorded so that there is still time to contact the subject (or in cases not involving persons, to perform the test or whatever was omitted) so as to remedy the deficiency. The economic saving in terms of research time and effort that would otherwise be wasted might be far greater than the cost of automatically converting the documents into a structured data base by the formatting program. Once the information is formatted, it would be straightforward to detect missing format entries and issue an alert. It has also been pointed out that a periodic summary of results to date over the whole study population, which could also be generated from the formatted documents, might indicate desirable changes in research strategy that otherwise might not emerge until the end of the study.

Acknowledgments

I wish to acknowledge the contribution of Dr. Margaret Lyman to the research on processing hospital records. In particular, the medical applications sketched in Section 4 are based on suggestions of Dr. Lyman, who also answered many questions throughout the study. This research was supported in part by NIH Grant LMO2616 from the National Library of Medicine, and in part by Research Grant SIS-75-22945 from the National Science Foundation, Division of Science Information. I am grateful to Edmond Schonberg for helpful comments on the manuscript.

References

Anderson, B., Bross, I. D. J., and Sager, N. (1975). Grammatical compression in notes and records: Analysis and computation. *Am. J. Comput. Linguist* **2**, No. 4.

Bloomfield, L. (1926). A set of postulates for the science of language. *Language* **2**, 153–164.

Bloomfield, L. (1933). "Language." Holt, New York.

Brook, R. H. (1977). Quality—Can we measure it. *N. Engl. J. Med.* **296**, 170–172.

Bross, I. D. J., Shapiro, A., and Anderson, B. B. (1972). How information is carried in scientific sublanguages. *Science* **176**, 1303–1307.

Charniak, E., and Wilks, Y. (1976). "Computational Semantics." North-Holland Publ., Amsterdam.

Chomsky, N. (1957). "Syntactic Structures." Mouton, The Hague.

Chomsky, N. (1964). "Aspects of the Theory of Syntax." MIT Press, Cambridge, Massachusetts.

Chomsky, N. (1972). "Studies on Semantics in Generative Grammar." Mouton, The Hague.

Chomsky, N. (1975). "Reflections on Language." Pantheon Books, New York.

Damerau, F. J. (1976). Automated language processing. *Annu. Rev. Inf. Sci. Technol.* **11**, 107–161.

Fitzpatrick, E., and Sager, N. (1974). The lexical subclasses of the linguistic string parser. *Am. J. Comput. Linguist.* **2**, (microfiche).

Grishman, R. (1973). The implementation of the string parser of English. *In* "Natural Language Processing" (R. Rustin, ed.), pp. 90–109. Algorithmics Press, New York.

Grishman, R. (1975). "A Survey of Syntactic Analysis Procedures," Courant Computer Science Rep. No. 8, New York University, New York.
Grishman, R., and Hirschman, L. (1978). Question answering from natural language medical data bases. *Artif. Intell.* 7 (in press).
Grishman, R., Sager, N., Raze, C., and Bookchin, B. (1973). The linguistic string parser. *Proceedings 1973 Nat. Comput. Conf.* pp. 427-434.
Harris, Z. S. (1951). "Methods in Structural Linguistics." Univ. of Chicago Press, Chicago, Illinois.
Harris, Z. S. (1957). Cooccurrence and transformation in linguistic structure. *Language* **33**, 283-340.
Harris, Z. S. (1968). "Mathematical Structures in Language." Wiley (Interscience), New York.
Harris, Z. S. (1970). "Papers in Structural and Transformational Linguistics." Reidel Publ., Dordrecht, Netherlands.
Harris, Z. S. (1976). A theory of language structure. *Am. Philos. Q.* **13**, 237-255.
Harris, Z. S. (1959). "The 1959 Computer Sentence Analyzer," Transf. Discourse Anal. Pap. 15-19. Department of Linguistics, University of Pennsylvania, Philadelphia. (Reprinted in part in Harris, 1970.)
Hill, D. R. (1971). Man-machine interaction using speech. *Adv. Comput.* **11**, 166-230.
Hirschman, L., and Grishman, R. (1977). Fact retrieval from natural language medical records. *In* "Medinfo 77" (D. B. Shires and H. Wolf, eds.), pp. 247-251, North-Holland Publ., Amsterdam.
Hirschman, L., Grishman, R., and Sager, N. (1975). Grammatically-based automatic word class formation. *Inf. Process. Manage.* **11**, 39-57.
Hirschman, L., Grishman, R., and Sager, N. (1976). From text to structured information: Automatic processing of medical reports. *AFIPS Nat. Comput. Conf. Proc.* **45**, 267-275.
Hobbs, J., and Grishman, R. (1976). The automatic transformational analysis of English sentences: An implementation. *Int. J. Comput. Math.* **5**, 267-283.
Insolio, C., and Sager, N. (1977). Parsing free narrative. *Annu. Meet. Assoc. Comput. Linguist. Georgetown Univ., Washington, D.C.*
Jespersen, O. (1914-1929). "A Modern English Grammar on Historical Principles." (Reprinted, Allen Unwin, London, 1961.)
Joos, M., ed. (1957). "Readings in Linguistics." American Council of Learned Societies, New York.
Josselson, H. H. (1971). Automatic translation of language since 1960: A linguist's view. *Adv. Comput.* **11**, 2-58.
Keyser, S. J., and Petrick, S. R. (1967). "Syntactic Analysis," Rep. AFCPL-67-0305. Air Force Cambridge Research Laboratory, Bedford, Massachusetts.
Korein, J. (1970). The computerized medical record: The variable field length format system and its applications. *In* "Information Processing of Medical Records" (J. Anderson and J. M. Forsythe, eds.), pp. 259-291. North-Holland Publ., Amsterdam.
Kuno, S., and Oettinger, A. G. (1963). Multiple-path syntactic analyzer. *In* "Information Processing 1962" (C. M. Popplewell, ed.), pp. 306-312. North-Holland Publ., Amsterdam.
Lancaster, F. W. (1977). The relevance of linguistics to information science. *Proc. 1976 FID/LD Workshop Linguist. Inf. Sci. Int. Fed. Doc. Fid* **51**, 19-43.
Lehmann, W. (1978). Machine translation. *In* "Encyclopedia of Computer Science and Technology" (J. Belzer, A. G. Holzman, and A. Kent, eds.), Vol. X, pp. 151-164 Dekker, New York.

Lindsay, P. H., and Norman, D. A. (1977). "Human Information Processing." Academic Press, New York.
McDermott, W. (1975). Evaluating the physician and his technology. *Daedalus* Winter, pp. 135-157.
Otten, K. W. (1971). Approaches to the machine recognition of conversational speech. *Adv Comput* **11**, 127-163.
"PEP Primer" (1974). 4th Ed. Joint Commission on Accreditation of Hospitals, Chicago, Illinois.
Petrick, S. R. (1975). On natural language based computer systems. *IBM J. Res. Dev.* **20**, 314-325.
Plath, W. J. (1975). "The REQUEST System" IBM RC 5604. IBM T. J. Watson Research Center, Yorktown Heights, New York.
Raze, C. (1967). "The FAP Program for String Decomposition of Scientific Texts," String Program grep. S.P.R. No. 2. Linguistic String Project, New York University, New York.
Raze, C. (1976a). A compoutational treatment of coordinate conjunctions. *Am. J. Comput. Linguist.* **52**, (microfiche).
Raze, C. (1976b). "The Parsing and Transformational Expansion of Coordinate Conjunction Strings," String Program Rep., S.P.R. No. 11. Linguistic String Project, New York University, New York.
Reeker, L. H. (1976). The computational study of language acquisition. *Adv. Comput.* **15**, 181-239.
Sager, N. (1967). Syntactic analysis of natural language. *Adv. Comput.* **8**, 153-188.
Sager, N. (1968). "A Computer String Grammar of English," String Program Rep., S.P.R. No. 4. Linguistic String Project, New York University, New York.
Sager, N. (1972a). Syntactic formatting of scientific information. *Proc. Fall Jt. Comput. Conf., 1972, AFIPS Conf. Proc.* **41**, 791-800.
Sager, N. (1972b). A Two-stage BNF specification of natural language. *J Cybern.* **2**, 39-50.
Sager, N. (1973). The string parser for scientific literature. *In* "Natural Language Processing" (R. Rustin, ed.), pp. 61-85. Algorithmics Press, New York.
Sager, N. (1975). Sublanguage grammars in science information processing. *J. Am. Soc. Inf. Sci.* **26**, 10-16.
Sager, N. (1977). Information structures in the language of science. *In* "The Many Faces of Information Science," AAAS Selected Symposium 3 (E. C. Weiss, ed.), pp. 53-73. Westview Press, Boulder.
Sager, N., and Grishman, R. (1975). The restriction language for computer grammars of natural language. *Commun. ACM* **18**, 390-400.
Sager, N., and Lyman, M. (1978). Computerized language processing: Implications for health care evaluation. *Med. Rec. News* (JAMRA), Vol. 49, No. 3, pp. 20-30.
Sager, N., Hirschman, L., Grishman, R., and Insolio, C. (1977). Computer programs for natural language files. *Proc. 40th Annu. Meet. Am. Soc. Inf. Sci., ASIS, Washington, D.C.* (in press).
Salkoff, M., and Sager, N. (1969). "Grammatical Restrictions on the IPLV and FAP String Programs," String Program Rep., S.P.R. No. 5. Linguistic String Project, New York University, New York.
Sapir, E. (1925). Sound patterns in language. *Language* **1**, 37-51.
Schank, R. (1975). "Conceptual Information Processing." North-Holland Publ., Amsterdam and Elsevier, New York.
Simmons, R. F. (1970). Natural language question answering systems: 1969. *Commun. ACM* **13**, 15-30.

Thompson, F. B., and Thompson, B. H. (1975). Practical natural language processing: The REL system as prototype. *Adv. Comput.* **13**, 110–170.
Walker, D. E. (1973). Automated language processing. *Annu. Rev. Inf. Sci. Technol.* **8**, 69–119.
Walker, D. E. (1975). "Speech Understanding Research," SRI Annu. Tech. Rep.
Waltz, D. J. (1977). Natural language interfaces. *ACM SIGART Newsl.* **61**, 16–65.
Winograd, T. (1972). "Understanding Natural Language." Academic Press, New York.
Woods, W. A., Kaplan, R. M., and Nash-Webber, B. (1972). "The Lunar Sciences Natural Language Information System; Final Report," BBN Rep. 2378. Botl Beranek and Newman, Cambridge, Massachusetts.
Zwicky, A., Friedman, J., Hall, B., and Walker, D. (1965). The MITRE syntactic analysis procedure for transformational grammars. *Proc. AFIPS Fall Jt. Comput. Conf.*, **1**, 317.

ADVANCES IN COMPUTERS, VOL. 17

Distributed Loop Computer Networks*

MING T. LIU

Department of Computer and Information Science
The Ohio State University
Columbus, Ohio

* This work was supported in part by the National Science Foundation under Grant No. US NSF MCS-7723496.

ISBN 0-12-012117-4