AUTOMATIC ENGLISH-TO-LOGIC TRANSLATION IN A SIMPLIFIED MODEL

A Study in the Logic of Grammar

by

H. G. Bohnert and P. O. Backer

Final Report Under Contract AF 49(638)-1198
Logico-Linguistic Studies for Machine Text Perusal
March 1966

Principal Investigator
Herbert G. Bohnert

Submitted to

The Air Force Office of Scientific Research
Washington 25, D. C.

The International Business Machines Corporation
Thomas J. Watson Research Center
P. O. Box 218
Yorktown Heights, New York

CONTENTS

PREFACE

This report describes the latest in a series of Englishlike languages translatable into forms of first order predicate calculus notation. It includes the features illustrated in all previous languages. The recognition and translation program, written in the string manipulating SNOBOL 3 system[1], is also described. Illustrative printout is included in an appendix. Sufficient detail is given so that interested linguists and logicians will be able to study the program with understanding, experiment with it, and perhaps build upon it toward better logic-based models of English.

This and earlier reports[2], have been based on work jointly sponsored by IBM and the Air Force Office of Scientific Research under contract AF49(638)-1198.

In the earlier reports, the present language has usually been referred to, prospectively, as LOGOS IV. Here, however, it will be called English II.

The logic-like language into which it is translated, preparatory to representation in parenthesis and Polish notation is here called English I.

I. The Logic of Grammar: Motives and Methods

The grammar of any natural language appears exasperatingly illogical to any but the most unreflective native speaker. Therefore, the phrase in our subtitle, "The Logic of Grammar", may sound naive. Yet underlying the evolutionary welter of conflicting rules, we sense the existence of basic communicative tasks, independent of any language in detail, but important for each language to find some way of doing: tasks such as naming, predicating, negating and so on.

With a systematic inventory of what these basic tasks are, we could proceed in our study of various languages by studying the corresponding devices which each had developed, evaluating the efficiency and economy of each. Even when conflicting ways of doing the same task occur within a language, consistent rules can be separated out, made explicit, and studied as to the consequences of following each exclusively within the general framework, noting advantages and disadvantages, and possibly gaining some insight into why each has survived.

Thus, both in the generic communicational tasks and in the specific consequences of limited grammatical rules, there seems a basis for a rational analysis of grammar. Justification of the term "logic" in a narrower sense may be momentarily postponed.

The idea of analyzing a language in this way is now new. The very terms of ancient grammatical analysis, "dative", "ablative", "genitive", reflect an attempt to provide a task, or role, analysis of grammatical features. In more recent times, the Norwegian philologist, Otto Jespersen,

made a serious effort of this sort, perhaps the last within the framework of linguistics proper, which he called the notional, as opposed to the formal, approach to grammar.[3] The difficulty of carrying out such ideas bred an understandable skepticism among later linguists. The pendulum swung to de facto description with behaviorists such as Bloomfield and structuralists such as Harris.

The current broad drive toward exact syntactic description via generative and transformational grammars, initiated by Chomsky, inherited this skepticism, and at the outset stressed the independence of syntax from questions of communicative function or meaning.[4] More recently, however, there has been increasing concern with the problem of relating each syntactic structure to a so-called deep structure which is intended to bear some closer relation to meaning than the apparent, or surface syntax admits.[5] This suggests that the pendulum may have begun a swing back toward something like Jespersen's notional analysis.

Thus "Barking dogs don't bite" and "Parallel lines don't meet" appear to have the same surface syntax. Yet the transformation which changes the first sentence into the equivalent "No barking dog bites" yields "No parallel line meets" when applied to the second, which is unintelligible. This is said to reveal a difference in deep structure in the originally given strings. But, in order to say exactly what the difference is, a system for representing deep structures must be found.

It is a thesis of the present study that an appropriate system of representation is already at hand. It is the notation of modern logic. It provides the needed inventory of basic tasks naturally and comprehensively. Indeed, the very terms used earlier to suggest the existence of basic tasks - "naming", "predicating", "negating" - belong to the basic vocabulary of logic, and its notation provides a systematic representation of them. Lest it be thought suitable only for such elementary notions as those mentioned, it should be borne in mind that it suffices for the formulation of all mathematics and hence for any formalized scientific theory. Even aspects of grammar which are not purely logical (but still notional in Jespersen's broader sense), such as tense, which is based on the physical concept of time, can be represented in logical notation with the help of explicit time symbolism, as will later be seen. It is in this sense that our use of the term "logic" seems appropriate.

This is not to say that current logical theory is adequate to represent all known grammatical devices. There are well marked problem areas remaining, e.g., modalities, indirect discourse, causal locutions, etc. But its representational powers are, for example, quite equal to exhibiting the difference in deep structure in the two sample sentences given earlier. "Barking dogs don't bite" can be paraphrased:

1. For every x, if x is a dog and x is barking, then x does not bite (anything).

"Parallel lines don't meet" can be paraphrased:

2. For every x, for every y, if x is a line and y is a line and x is parallel to y, then x does not meet y.

Logicians will recognize these reformulations as corresponding to a familiar logical notation, revealing the difference to involve, among other things, the relational demands of "parallel" and "meets".

As the example suggests, our proposal is not merely the modest one that logical symbolism be used to systematize talk about the structure of sentences[6], but the more ambitious one that the sentence's deep structure be represented by an actual paraphrase in the notation of logic.

This idea, per se, is also not new. Not only have logicians from Aristotle, through the Stoics, Descartes, and Leibniz, analyzed specific sentence structures in the logical terms of their day, but Russell's writings give linguistic analyses in the modern symbolism he helped create; and later logicians, e.g., Quine, have contributed greatly to this development.[7]

While linguists have hardly begun to tap logic's potential even at this level, it remains true that such analyses so far have remained piecemeal. A grammar is, after all, a system of rules, and the logician's suggestions can only be evaluated adequately in terms of their operation within a system. Rudolf Carnap was apparently the first to suggest the possibility of approximately representing some part of a natural grammar by a logistic system,[8] and was followed by Hans Reichbenbach who made important suggestions for such work in this direction.[9]

To the extent that a natural grammar can be approximated by a logistic model, an additional advantage is gained in being able to carry over to that part of natural grammar the integrated system of semantical concepts that have been developed for such systems by Tarski, Carnap, Bar-Hillel (1952) and others, e.g., designation, truth, consequence, information content, generality ("width") of predicates, and, of course, sameness of meaning, or synonymity, a concept pre-supposed in any discussion of deep structure.

Y. Bar-Hillel (1954) urged these advantages upon linguists but was rebuffed by Chomsky (1955). Later, however, Chomsky has taken a more moderate wait-and-see attitude, along with his increasing interest in the deep structure problem.

Logistic modelling is admittedly a formidable task and even with its rationale fully accepted, it might normally be expected to be undertaken with reluctance. The advent of computers, however, has provided new incentive. There is obvious need to communicate with machines in as natural and flexible way as possible. This need has, together with the possibilities of machine translation and retrieval of information from natural language text, already fostered a surge of research in natural language. So far, this has been almost entirely in the purely syntactic, transformational grammar tradition, with little attempt to relate the machine parses produced to logical representations. At the same time, however, there has been an impressive development of machine deduction based directly on logic (an early, and still impressive example: H. Wang's

program (1960) which proved all three hundred and fifty of the first order theorems of Principia Mathematica in 8.4 minutes). The potentialities of a computer system which would translate from English, not just to a linguistic parse, but to logical notation thus became apparent to many writers independently in a fairly short-time period.[10]

There have, in fact, already been several computer programs written which translate from English-like sentences into logic; some linked with a deduction program permitting deductive solution of problems posed in English. Perhaps the most advanced so far is that of J. Darlington (1965). In most of these, however, the focus has been on the practical possibilities with the analysis of grammar a secondary consideration. Typically, the exact part of grammar being modelled is not explicitly specified (though implicit in the program, after a fashion). While further development of these practical possibilities will in itself call for a deeper analysis of the logic of grammar, with embodiments in computer programs, there are strong reasons why the purely scientific analysis of grammar, logical as well as purely syntactic, should organize itself around the computer, as astrophysics once organized itself around the spectograph.

Quite aside from its obvious advantages of speed, it provides an almost indispensable check on the operation of proposed rules within a system. As might be expected, proposed grammatical rules interact, often in ways difficult to foresee. The computer, doing exactly what it is told, acts as a merciless critic. By the same token, when a program

finally checks out over an intended range of sentences, it constitutes evidence of rule compatibility beyond that of even the most scrupulous human examination. A further, less direct, reason is that technical features of the underlying programs such as the specific sorts of string manipulations or list structures required, seem rich in insight into basic features of linguistic information handling in machines, and perhaps in the human brain. While proper caution is called for here, such considerations might, in the long run, prove at least equally important.

With these motivations, the present study has embodied its basic logico-grammatical models in computer programs. The procedure has been to begin with a very roughly Englishlike language, English I, whose grammar is essentially that of elementary logic itself (similar to the representation provided for in the barking dogs - parallel lines comparison). This is in accord with the thesis that the tasks performed by elementary logical notation do, in fact, represent a linguistically basic repertoire. For this language, the principle task is providing a program which recognizes grammaticality (translation to logical symbolism being a simple dictionary operation.) From there we proceeded to construct languages (i.e. by giving exact definition of sentencehood in terms of appropriate auxiliary concepts) which were progressively more Englishlike, each one accompanied not only by a recognition program, but by a program which translates it back to a standard logical notation. Thus our procedure might be described by saying that we start with deep structure and work toward surface syntax, rather than the other way around.

It may be felt that although an algorithm which translates to standard logical notation might provide a useful deep structure representation, and even a logic (as it does, if one rules that the logical relations which hold between sentences are just those that hold between their logical translations), it falls short of what one might expect of a logistic model of L. One might expect a logistic model of a natural language to resemble it not only in its basic syntactic categories and formation rules, but also in its logic, or transformation rules. There should be a "natural" logic, it might be urged, formulated in terms of the natural language forms, not their translations, and these should lead to a "natural" deep structure representation quite different from the artificial notation of logic. This is a reasonable aspiration and our more indirect procedure may be regarded as a roughhewn prolegomenon to such a finer modelling. This first step seems an almost mandatory one, however. We know that standard logic works, so to speak, and we can feel confident that the logic of a certain English device has been really understood if its translation to standard notation can be shown to be uniformly possible. It is hard to see how such confidence could be otherwise obtained. Indeed, any proposed more natural logic could hardly be justified itself except by some proof that it also provided representation, logically equivalent with the standard notation version.

From another viewpoint, the use of standard logic notation provides a valuable generality in its very non-resemblance to natural languages.

It is strongly neutral with respect to the grammatical features that typically distinguish one natural language from another. Such notation does not distinguish between nouns, verbs, or adjectives. It has no declensions, genders, or rules of agreement. It has no tense, person, mood, or number. (Leibniz was perhaps the first to insist on the logical dispensability of such devices.) Yet the intended effect of these features can be paraphrased within the framework of this one, exact, very general grammar, as we shall try to show. It has often been remarked that grammatical studies have been badly warped by a tendency to force Latin tense and case paradigms on languages which had no corresponding mechanisms. Logic may suffer from its own provincialism but, if so, it is at least a far broader one.

Our treatment of syntactic ambiguity is another feature which may at first repel linguists. Our policy has been, when faced with such an ambiguity, to rule somewhat arbitrarily in favor or a single interpretation. More exactly, our formation rules along "generate" well-formed strings which are ambiguous in the syntactic sense that their generation could have been accomplished in more than one way. Our recognition and translation rules, however, involve a transformation of each string to an unambiguous form. Our selection of an interpretation in this sense is guided not only by consideration of what the most "natural" interpretation might be, but also by whether there exists in our grammar an alternative, reasonably natural, way of expressing the rejected interpretation. This is to give any given model language as much referential capacity as

possible, admittedly a normative rather than a descriptive consideration. The policy of "artificial" univocality, however, is in keeping with the methodological attitude expressed in the opening paragraphs: When faced by conflicting rules, make each explicit and follow out the consequences of adhering to each exclusively. Our present policy amounts to the initial step of following one rule in each case. Nothing prevents later exploration of other consistent readings, nor of constructing programs which successively yield a given range of possible readings. A more basic, though probably more arguable, reason is the belief that the vexing problems of ambiguity are not especially profound in principle and that they tend to confuse and obscure deeper syntactic or logical problems.

The present approach, then, represents a confluence of linguistics, logic, philosophical language analysis, and computer science, and it is addressed to specialists in all of these fields. With these distinct audiences in mind, care is taken to make the main line of discussion understandable to those without special training in logic or computer science. In fact, the grammar of logical notation will itself be informally developed first, so that the linguistic reader may have a fair opportunity to evaluate for himself the claims here put forward for it. Similarly, informal descriptions of the program algorithms will precede more technical treatments; informal characterizations of the model languages constituted will accompany formal constructions, etc.

These procedures are, of course, not intended to substitute for courses in logic, training in programming, nor even as full preparation for following the technical portions in detail.

II. The Grammar of Logic: Predicating, Compounding, Generalizing

This section gives an informal overview of the syntax of typical logical notation and some of the questions it raises for the logical analysis of natural language.

Elementary Predication

A central grammatical conception in all language, natural or artificial, is that of applying a predicate to a subject, or subjects, i.e., of "saying something about something". Its clearest manifestation is in ascribing some property to a single named object, e.g., "David is hungry". When the predicate is used to assert a relation between two objects, (i.e. when it is dyadic in the terminology of logic), e.g., "David killed Goliath", "Boulogne is north of Paris", or among three objects (triadic), "Cleveland is between New York and Chicago", the situation need not be much more complex syntactically. All that is required is to specify the relation, name the participants in the tableau, and distinguish their roles in it.

In a typical logic notation (one we will use), the attribute or relation is represented by a single capital letter (subscripted when the alphabet gives out) while the participants in the tableau (always a fixed number for a given predicate, called its degree) are named by single small letters (also possibly subscripted) trailing the predicate letter. The role of the participants in the tableau is fixed by the order in which the letters occur. Thus "a is between b and c" could be rendered as "Babc". I.e., if "a is between b and c" is rendered as "Babc", then "Bbac" would

have to mean "b is between a and c", and "Bcba" would have to mean "c is between b and a". A predicate of degree n, followed by n names (small letters) is said to be an elementary (atomic) sentence (of the given notation).

Compounding

Compound sentences are built up by either of two methods: (1) (parenthesis, or infix, notation) introducing symbols for the connectives "or", "and", etc. with parentheses to avoid ambiguous grouping; (2) (Polish prefix notation) introducing symbols corresponding somewhat to English groupers "Either", "Both", etc. The two methods are illustrated below using small letters, beginning with "p", to stand for whole subsentences. Note that negation, acting on a single sentence, can be regarded as either connective or grouper.

Polish Notation	Parenthesis	Routh Englishlike Analogy
Np	$\sim$p	Not p
Bpq	(p . q)	Both p (and) q
Epq	(p v q)	Either p (or) q
Ipq	(p $\supset$ q)	If p (then) q
BpEqr	(p . (q v r))	Both p (and) Either q (or) r
BEpqr	((p v q) . r)	Both Either p (or) q (and) r
EBpqr	((p . q) v r)	Either both p (and) q (or) r

The grouping letters here have been changed from Lukasiewicz's original 'Polish' notation to provide a mnemonic correlation with the English words on the right, to which we shall assign similar grouping functions in English-like languages.

Generalizing

Next we consider logic's means of obtaining general sentences, i.e., those which in English are typically expressed with the help of terms like "every" and "some". The required indefiniteness is obtained by introducing a new syntactic category: (individual) variables: x, y, z, x_1, y_1, z_1, ... These act like names syntactically. More exactly, if we lump names and variables together as (individual) terms, an n-degree predicate followed by n terms constitutes an elementary (or atomic) formula. These "indeterminate" formulas, in turn, can be compounded, as atomic sentences were, into compound (or molecular) formulas. Formulas containing variables can be transformed into meaningful sentences of the system by applications of the universal quantifier, "A", and the existential quantifier, "E", by admitting a rule that one may prefix any formula, e.g., (Px v Qxy)", by a quantifier followed by a variable e.g., "Ax (Px v Qxy)", and still have a formula provided the original formula did not already contain a subformula in which a quantifier is immediately followed by the same variable, (which would cause an ambiguity). The significance of the resulting formula is most easily conveyed by the following paradigms:

Ax_____ For every x, _____

Ex_____ There exists (at least one) x such that _____

or For some x, _____

Where the dash stands for the original formula, whose reading is, in itself, not altered by prefixing the quantifier. In the paradigm shown,

the variable "x" is used but only as illustration. Any variable is admitted. The variable following the quantifier is said to be bound in the formula to which it is prefixed. Here parentheses again serve to prevent ambiguity. Thus in "(Ax(Px v Qx) v Rx)" the "x" in "Rx" is not bound. It is said to be outside the scope of the quantifier. The other occurrences of "x" are bound. A variable unbound in a given formula is said to be free in that formula. In order for a formula to have determinate meaning, then, all its variables must be bound, in which case it is called a sentence of the system.

Interpreting a formalized language of that sort, i.e. giving it a semantics as well as a syntax, requires among other things, that a universe of discourse be specified (e.g. connected material objects space-time regions, mass-points) so that, e.g., when a sentence is prefixed by 'Ax' its universal claim is indeed definite. The variables in such a notation are then said to range over the universe of discourse.

By careful attention to the degree of predicates, the exact reading of connectives, quantifiers, and their scopes, these simple syntactical means can be combined to express complex sentences. Initial illustrations have already been given in the barking dogs, parallel lines examples. A further example will be given here.

Consider the sentence "A chain is no stronger than its weakest link". We can symbolize the needed vocabulary and make clear the assignment of roles to places in the predicational sequence by presenting each predi-

cate in a full atomic formula with distinct variables and matching it with a quasi-English paraphrase in which the same (variable) letters appear in the intended role:

Cx	x is a chain
Lxy	x is a link of y
Sxy	x is stronger than y

We shall need no other terms. In particular, we shall need no special symbolization for the superlative nor for the effect of the possessive pronoun. The latter is implicit in the decision to regard "link of" as a dyadic relation. The sense of the superlative "weakest" can be spelled out in terms of the comparative "stronger than". The given sentence can be rendered:

"$\wedge x \wedge y (Cx . (Lyx . \sim \vee z (Lzx . Syz)) \supset \sim Sxy)$". This can be stiffly but intelligibly read as follows:

"For every x for every y (if x is a chain and (y is a link of x and not (there is at least one z such that (z is a link of x and y is stronger than z)) then not x is stronger than y"

Parentheses are retained to keep the grouping obvious.

A somewhat more natural version would be:

"For any chain, x, and any of its links, y, such that there exists no link z, of x, stronger than y, chain x is not stronger than link y."

And of course the most natural of all is the original sentence.

The system we have so far described is referred to by logicians by various phrases: The first order predicate calculus, quantification theory, the restricted, or lower, functional calculus. While a more exact characterization will be possible presently, it is hoped that enough has been said to convey an idea of its basis syntactical resources: (1) fixed-degree-fixed-order predication involving names and variables, (2) negating and compounding formulas according to some unambiguous scheme of grouping, (3) expressing generalized propositions by binding variables with quantifiers.

Each of these features presents problems to the logical analyst of natural language:

(1) How does a given language indicate role in a predication? How and to what extent does it achieve flexibility in predicational word order?

How can apparent variations in predicate degree be assimilated to a fixed degree system?

(2) What devices does a given language have to indicate grouping? When a language compounds not only sentences, but subjects, objects, predicates, etc., can its syntax be analyzed sufficiently to be translated into logic's simpler system?

(3) By what means does a given language express generality? Natural languages do not have anything quite like variables. Even pronouns, which have often been likened to variables, often fail to appear in general sentences, e.g., "Bees buzz" (universal) "Mary had a little lamb" (existential)

These are among the questions with which this study will be concerned. Though fundamental, they do not, of course, exhaust the problems of language nor the resources of logic. In particular, it may help to round out this preliminary survey of the grammar of logic to remark that logical systems more powerful than the first order calculus ((which are needed in varying degree of strength for various parts of mathematics (and for analysis of even very elementary quantitative locutions in natural language)) will usually differ syntactically from the first order calculus, if at all, only in introducing new classes of variables (ranging over more abstract entities; sets, functions, numbers, etc.).

Certain forms of logic may employ a few other syntactical forms such as term-forming operators or function-symbols, but these are, in principle, eliminable. While we anticipate calling upon all the logical and syntactical resources of advanced logic (in analyzing quantitative locutions, abstract terms, modal auxiliaries ("may", "must", etc.)), the three features we have described, i.e., predicating, compounding, and generalizing, remain basic.

III The Grammar of English I and II.

This central section of the report describes the grammar of our model, English II, alongside that of the more logic like English I. Some of the notation is that of the SNOBOL program which processes the sentences of English II and translates them into English I preparatory to representation in parenthesis and Polish notation. The discussion is divided into grammatical topics. In each part a preliminary informal treatment is given to motivate the formalism of the rules which follow. The rules themselves constitute a self contained recursive definition of the well formed expressions of the language.

Predication

English I and II are restricted to third person singular forms. They use a single tense, typically present, though simple past is sometimes used in the printout examples. Predicates are assigned a degree as in logic. In English II, predicates are also assigned a traditional category: Verb, Adjective, Noun. In elementary predication, these take the forms displayed in the following paradigm:

verb	Don runs	Don does not run
adj.	Don is tall	Don is not tall
noun	Don is a runner	Don is not a runner

After translation from English II to English I these settings are lost. The uncategoried English I equivalents are:

Don runs	Not Don runs
Don tall	Not Don tall
Don runner	Not Don runner

When a predicate (of any category) is of higher than first degree it often occurs in a pattern with other words, here called the placers of the given predicate, shown underlined below:

verb:	George gives Fido to Don	x gives y to z
adjective:	George is taller than Don	x (is) taller than y
	A is between B and C	x (is) between y and z
noun:	Fido is a gift from George to Don	x (is a) gift from y to z

Placers will often belong to the traditional category of prepositions, but they need not (as in the case of 'than' and 'and' above). Webster's classifies "than" as a conjunction (1) but its defining entry is "Indicating the second member of a comparison expressive of inequality". I.e., its function is simply that of a placer. Note also its typical role in "A writer than whom no sage was wiser wrote 'Isagoge'", an example whose translation may be seen in the printout. (Example 15)

It will be seen that the use of placers to distinguish participants in the predicational tableau is a significant syntactical alternative to the use of order alone. Like the case system of an inflected language, e.g. Russian or Latin, it permits more flexible word order. English, indeed, does not take full advantage of its possibilities, since subject, and usually object, are not marked by placers and hence are frozen in position, in

ordinary usage. Such fuller advantage could be seen, e.g., in a form of logical notation which introduced a special set of placer terms, e.g., $a._1$, a_2, ... a_i, rewriting four place predicate M, say, as follows

$$M\,x\,y\,z\,w \quad M\ a_1 x\ a_2 y\ a_3 z\ a_4 w$$

With these placers, order could be shifted so that, e.g., "a_3 George a_1 John M a_2 William" would mean the same as "M John William George". Even here, however, complete freedom is not attained since positionals (placer-with-argument) of one predicate could not be allowed to stray among those of another. In contemplating the bookkeeping that would be required to permit complete freedom one quickly sees it would be prohibitively complex, if possible at all.

The advantage of some flexibility i n word order, on the other hand, are by no means limited to rhetorical considerations. Some of its abbreviatory value will become clear in the later discussion of "factoring". A glimpse of its full theoretical significance may be seen in Quine (1960b) and in any treatment of combinatorial logic. Other devices for achieving flexibility in English and English II include the active-passive option, the use of converses ("husband of", "wife of") etc.

In English I and II only single-word placers and single-word proper names are admitted. Also, only a single category, degree, and placer pattern are allowed for a given predicate. The first two restrictions are minor programming conveniences, easily relaxed. The restrictions on category, degree, and placer pattern, however, can only be removed at the last stage of model language construction, since only then will all

the contexts be known by which ambiguity, e.g., in category, can be resolve Since English II is no final model, this step has been postponed.

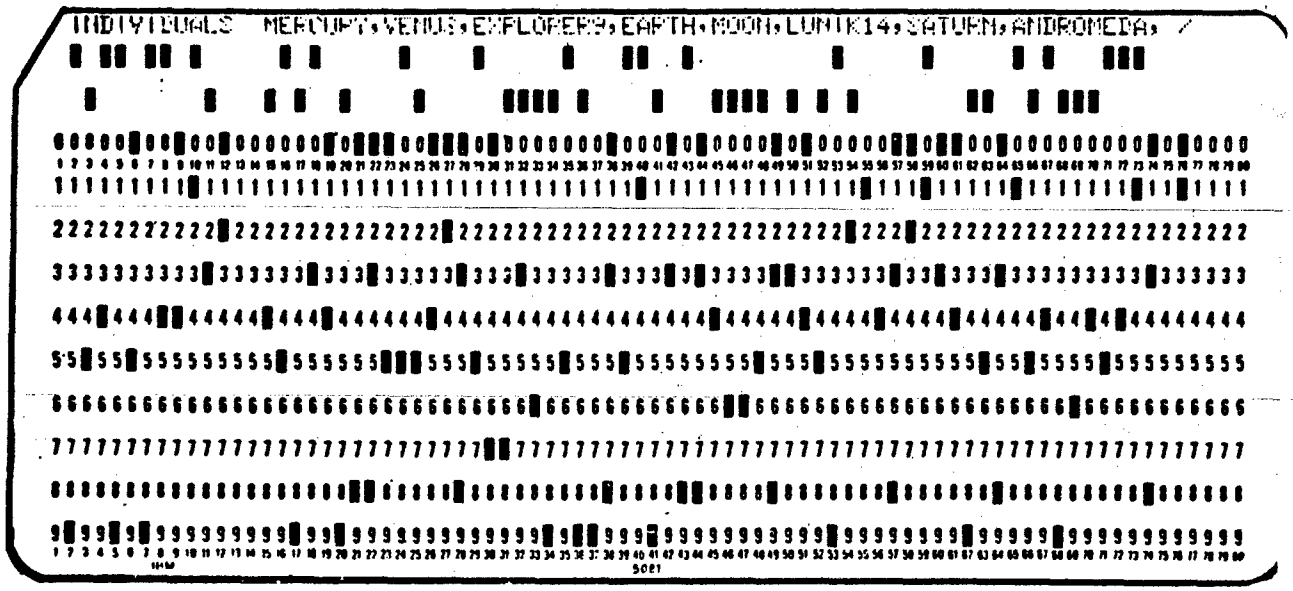
Proper names and predicates (unlike logical terms such as "or", "ever etc., which determine fixed grammatical structure) can be chosen by the program user at the time of a machine run. The card formats to be used for names are illustrated in Figure 1. Formats for predicates, showing synonymous variants in parentheses, grammatical category, degree, and predicational pattern, are shown in Figures 2 and 3. A sample input sentence is shown in Figure 4. Fuller specifications are given in Appendix I. The program constructs dictionaries from these cards, assigning the symbols "y1", "y2", etc. to the names as encountered. Symbols beginning with "H", e.g. "H1.2", "H2.1", H3.3", "H4." are assigned to predicates, where the first numeral is a sequentially assigned identification and the second, if any, categorizes it as verb (1), adjective (2), noun (3).

Placers, as found on predicate cards (no separate cards are required), are assigned symbols beginning with "P1." i.e. "P1.1", "P1.2", etc.

To each predicate symbol the dictionary, built by the program, assigns a placer set representing the pattern shown on the card. In the case of "gives", shown in Figure 1, the card pattern "3. X1, G, X2, TO X3" is changed to "3// PO.1/ PO.2/ P1.5/*". In this scheme, the first number is the degree. The symbols beginning with "PO.", are called null placers, with the second numeral indicating "normal" place. Thus "PO.2" as a null object placer. "P1.5" would be the symbol for "to" if it were the

fifth new placer word encountered in the predicate cards during a run. Null placers perform an essential function in keeping track of subject, predicate and other unmarked arguments during transformations required in the program. Henceforth "placer" will also be understood to refer to the null placers.

While linguists sometimes speak informally of the syntactical demand of a given structure we shall, in speaking of the demand of a predicate, mean just the machine version of the string above but with degree and asterisk deleted. E.g., "PO.1/PO.2/P1.5/" is the demand of "gives". Presently, we shall see how demands of more complex expressions are "computed" from their configurations together with the dictionary-given demands of the predicates occurring in them. We are, in fact, in the early stages of a recursive definition of "Sentence of English II".



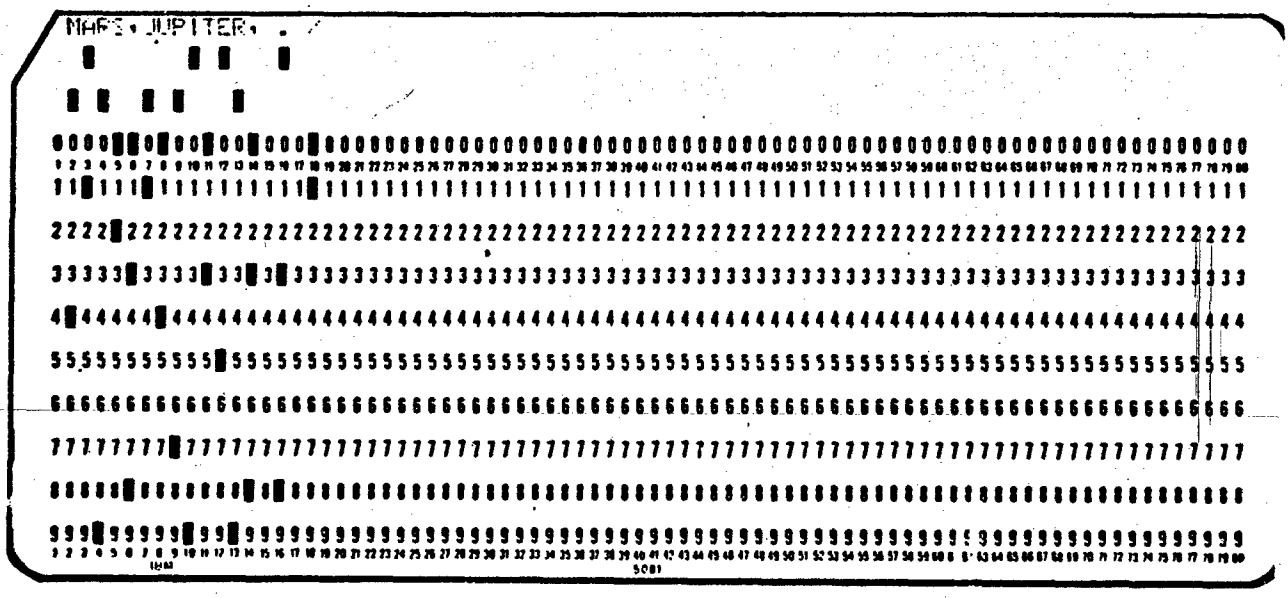


Figure 1.

Name entries can be entered many to a card and may extend beyond one card. Last card has period. All cards have slashes.

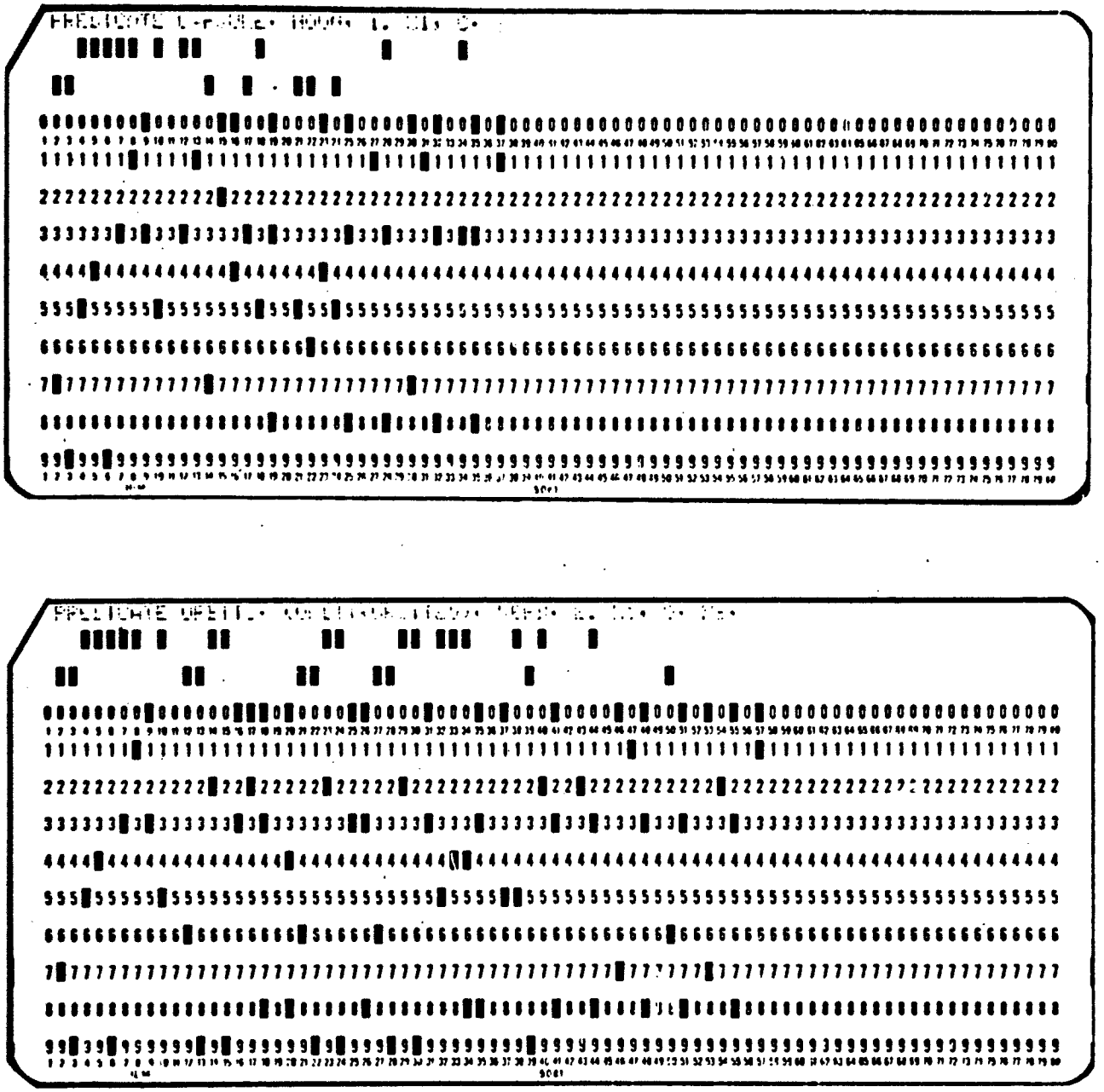


Figure 2.

Predicate cards: one per predicate. Synonymous forms, category optional. Degree. Pattern. Slash.

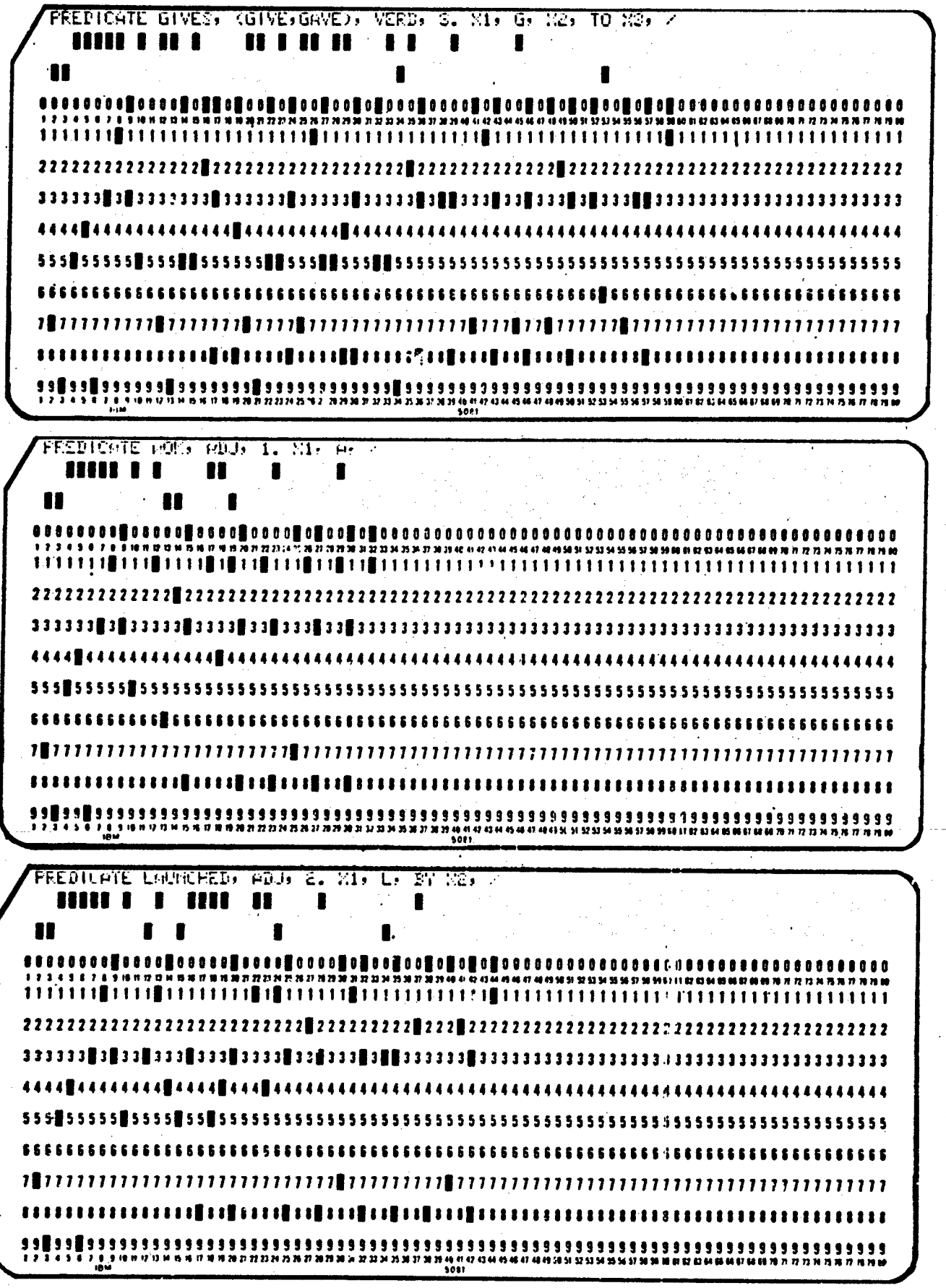


Figure 3. Predicates

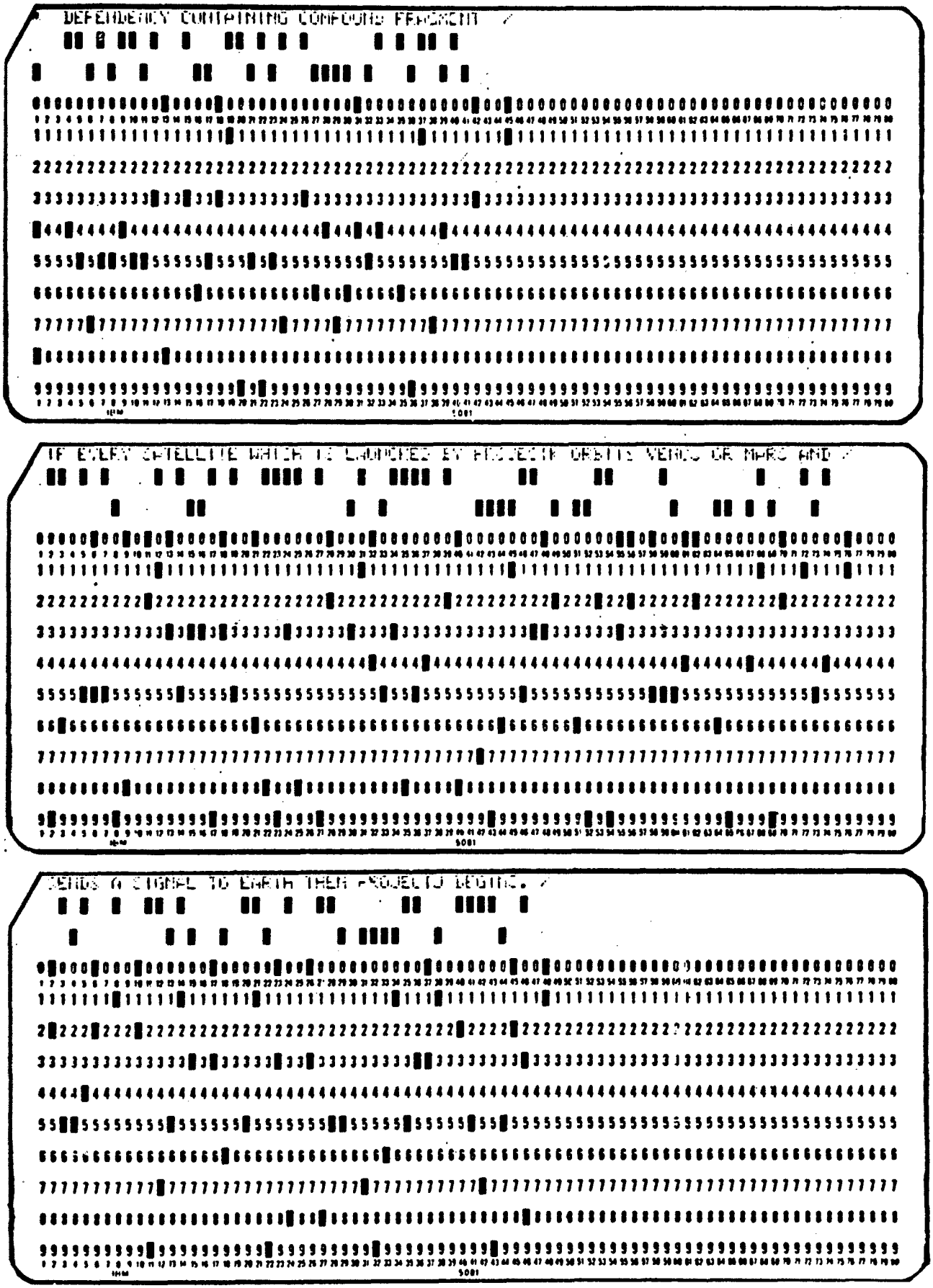


Figure 4. Comment card and input sentence extending beyond one card.

Compounding: The Grouping Problem

English I, like logic, can compound only formulas. There are no compound subjects, predicates, etc. To achieve unambiguous grouping it simply uses "not", "both", "either" and "if" like Polish groupers, but it retains the redundant associated connectives "and", "or", "then", as shown on the notation, comparison table of the foregoing logic section. (Page 16)

There is often a clumsy pile-up of groupers in English I, as can be seen in the printout, but the system is, of course, unambiguous.

English II, like natural English, permits connectives unaccompanied by groupers. This leads to syntactic ambiguity, but it need not lead to semantic ambiguity if rules are applied during the reading which supply missing groupers. This is the method of the present program, which supplies groupers according to a precedence system.

Such techniques are familiar not only to logicians but more recently to programmers, in their handling of algebraic expressions where parentheses are omitted.[11]

In the basic method, each connective is given a number called its precedence strength (in order from strongest to weakest: "not", "and", "or", (if) "then", "if and only if"), with the proviso, roughly speaking, that an ungrouped sentence "breaks" at its weakest connective. Thus "p and q or r" would break at 'or' rather than at 'and'. That is, it would be grouped "((p and q) or r)". English I groupers would accordingly be

inserted: "Either both p and q or r".

In the case of repeated connectives, grouping is to the right, e.g., "p v q v p" becomes "(p v (p v r))" or "either p or either q or r". This treatment of 'or' and 'and' as strictly dyadic may seem an artificiality of logic forced on language by the present approach. It may be artificial but it is not forced by logic per se. Systems of logic which permit non-dyadic, expanding 'and' and 'or' expressions have been developed,[12] and may be resorted to when a finer modelling of natural language is called for, e.g. when measures of syntactic complexity in the sense of Bar-Hillel et al. (1963) become a focus of attention.

While the precedence system described above provides the general principle for the English II compounding-grouping system, there is a further refinement.

The precedence system is extended by introducing two additional connectives whose logical roles are still those of 'or' and 'and' but whose strength in the precedence system is altered. Thus, "or else" is introduced as a weaker "or" and "and furthermore" is introduced as a weaker "and". The grouping effect is shown in the following comparisons:

"p and q or r" is construed as "((p . q) v r)"

while

"p and furthermore q or r" is construed as "(p . (q v r))".

"If p then q or r" is construed as "(p ⊃ (q v r))"

while

"if p then q or else r" is construed as "((p ⊃ q) v r)".

These additional low strength connectives help in avoiding pile-ups of groupers. Without them, the grouping in last example above, would have to be made explicit by groupers:

"Either if p then q or r".

The rendition of these low precedence connectives by longer phrases is prompted by two plausible linguistic conjectures.

The first is that in spoken English, grouping is often effected by shifts in speed or by use of pauses. That is, if, in reading "p and q and r or t" we read it "p and q" pause "and r or t" the hearer tends to understand it as the independent assertion of p and q with an assertion of the disjunction "r or t", e.g., (p . q) . (r v t); while if we read it "p" pause "and q and r or t" the hearer understands it as p . ((q . r) v t), especially if "q and r" are read quickly.

The second conjecture is that words like "else", "furthermore", "moreover", etc., act as written counterparts of verbal pauses, allowing a certain mental "closure" to set in; something like mentally adding a right-hand parenthesis, sealing off what has been said from involvement with any connectives to follow.

The method of multiple precedences can be extended further, of course, by introducing, e.g., "it is not the case that" as a low precedence negation, etc. And it can be altered to study the naturalness of other rankings. The basic ranking actually used was: "not", "and", "or", "then", "or else", "and furthermore".

The machine representation of groupers and connectives can be seen in Table I, where they are displayed in order of their precedence. The precedence number is the first numeral, the function number the second. Thus the representation of "and furthermore" as C3.8 shows it to have low precedence, 3, but to function like "and", 8.

Table I
Logical Terms

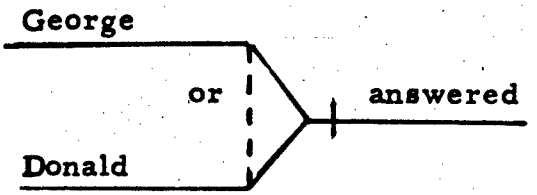
English I or II	Machine Symbolic	Text Symbolic	SNOBOL
Not	N	~	G9.9
Both	B	B	G8.8
And	.	.	C8.8
Either	E	E	G7.7
Or	V	v	C7.7
If	I	I	G6.6
Then	=*	⊃	C6.6
Or else	V*	v*	C4.7
And furthermore	.*	.*	C3.8
For every	A	A	Q1
For some	S	E	Q2
(variables)	Z1, Z2	x, y, z, x, y	Z1, Z2
Is		is	H1.1
Was		is	H1.1
Every	D1	e	D1
A	D2	a	D2
An	D2	"	D2
Any	D3	u	D3
Some	D4	s	D4
No	D5	n	D5
Which	W1	s	W1
Who	W2	w	W2
Whom	W3	w	W3

The general principle of the grouping program (a SNOBOL 3 function GRU(X)) is easily given. Each time a grouper is encountered its function number is entered in a pushdown. When a connective is encountered its function number is compared to that in the pushdown. On match, the pushdown is popped up (i.e. the demand of the grouper is satisfied) and the string between the grouper and connective is enclosed by parenthesis. On no match, the connective goes on a "hunt" leftward through the string matching its precedence with that of each connective encountered (but skipping parenthesis-enclosed strings) until either (1) it comes to the beginning of the sentence, whereupon its corresponding grouper is placed at the head of the sentence, enclosing the traversed string by parentheses, or (2) it encounters a weaker connective whereupon it inserts its own corresponding grouper to the immediate right of the weaker connective, and encloses the traversed string with parentheses. After each such operation a new sentence is sought. If, when the end of the string is reached, there are no claims left in the pushdown, the string is well formed and fully grouped. The parentheses or groupers are then edited out depending on whether polish or parenthesis notation is desired.

Since the grouping-by-precedence system described is quite general, English II uses its GRU function to group compound subjects, predicates, etc. and it is to this topic we next turn.

Compounding Terms

The sentence "George or Donald answered", in the diagramming method of high school memory,[13] would be rendered as follows:



Logical notation does not admit compound subjects, and it pays for the simplicity of its rules in the clumsiness of its sentence structures. The sample sentence must be changed to "George answered or Donald answered" before direct translation into such notation (e.g., as "Ag v Ad") is possible.

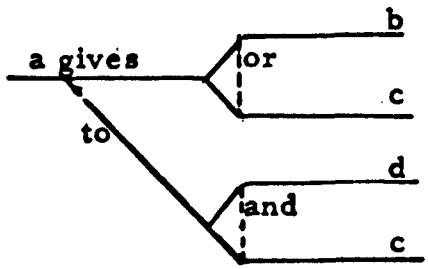
If we grant the primacy of logic's grammar, the English compound subject may be thought of as the result of factoring out the repeated predicate "answered". And the more primitive sentence may be thought of as attained from the more compact English sentence by distribution of the predicate over the compound subject.

This algebraic analogy may be seen more graphically if we imagine introducing a notation for compound subjects into logic in such a way that a "factoring law" establishes the equivalence:

$$A(g \vee d) \equiv Ag \vee Ad$$

Compound direct and indirect objects also occur in English, of course, and can be similarly handled. A sentence of the form "a gives

b or c to d and e" with the traditional diagram:



can be thought of as the result of successive factorings. The original "primitive" sentence can be recovered by successive distributions, as follows:

1. a gives b to d and e or a gives c to d and e.
2. a gives b to d and a gives b to e or a gives c to d and a gives c to e.

In the augmented logical notation, letting "G" stand for "gives", we would have the equivalence:

$$G(a, (bvc), (d \,.\, e)) \equiv ((Gabc \,.\, Gabe) \vee (Gacd \,.\, Gace))$$

Formally, to construct a logic notation of this sort in which factoring and distribution would be possible we should have to alter the usual logic syntax so as to make compound individual expressions acceptable arguments for predicates, e.g. "(a v (b . c))" as an argument to a predicate P, making "P (a v (b . c))" a well-formed formula.

We now take the corresponding step in our recursive construction of English II. Instead, however, of merely defining "compound name" we introduce a more general word, "term" whose meaning will be extended by later steps in the recursion. Informally speaking, "term" will ultimately embrace all expressions which form acceptable arguments for

English II predicates.

The required clauses for the recursion are the following:

1. Names are terms.

2. If a and b are terms, then so are the following:

not a, both a and b, either a or b, (grouped forms)

a and b, a or b, (ungrouped forms)

a and furthermore b, a or else b (low precedence forms)

In this formulation, and others like it to follow, any letters, capital or small, or strings of them, may be used (as "a" and "b" have been here) as metalinguistic variables ranging over strings. In certain contexts some words will be used without quotes to refer to themselves. The context will always make clear what is meant.

Also, further statements belonging to the recursion will be abbreviated by use of the symbols shown in Table I. Thus, the preceding list of forms could have been presented as

Na, Ba.b, Eavb, a.b, avb, a.*b, av*b

with the agreement that a symbol corresponds to the spelled word followed by a blank.

Terms, then, so far, consist of names and combinations of names, e.g., "George", "Charles and either Donald or Estelle and furthermore Arnold" are counted among the terms of English II.

Compounding Fragments

Besides compound subjects, objects, and indirect objects, traditional grammar also recognizes compound predicates as in "George runs or walks". Indeed all traditional categories are presumably compoundable.

Natural language factoring can, however, produce compounds which correspond to no traditional category. Thus when an indirect object is factored out as in:

"George sent the card and Tom wired the bouquet to Estelle",

the string "George sent the card and Tom wired the bouquet" does not constitute a compound subject, or predicate, but a compound "sentence-with-missing-indirect-object".

The corresponding logical notation would seem to be "(Sgc. Wtb)e" which should distribute to "Sgce . Wtbe".

To handle factoring in general, then, we appear to need a way of handling fragments, i.e. expressions which would be well-formed sentences if they were not defective in certain argument positions.

It was for the purpose of setting up such a needed calculus of fragments that we introduced our concept of demand, which we can now extend to fragments by a recursive process based on the demands assigned to the predicates by the dictionary.

Similarly, the categories are extended to strings other than predicates. Thus "either boy or girl" will be called a noun. "Noun phrase"

may be used in informal discussion, but it is not required in the recursion.

Throughout the following recursive stipulations it should be born in mind that "predicate" is reserved just for the words entered on dictionary cards. Also, by the noun, "demand" we always refer to a string of machine symbols, possibly null which the recursion proceeds to associate with the strings called fragments, (though we shall continue to use "demand" more informally, both as noun and verb in accompanying explanations.)

One null placer will be said to be less than another if its place number is less. E.g., PO.2 will be said to be less than PO.4. In this sense, we can also speak of the lowest null placer in a given demand.

We now proceed with the recursion, numbering the stipulations, 3, 4...etc., as sequels to stipulations 1 and 2 given earlier.

3. If P is a predicate of demand D and category c then it is also a fragment with the same demand and category.

4. If t is a term and F is a fragment whose demand contains PO.1 then

tF, t does not F	(if F is a verb fragment)
t is F, t is not F	(if F is an adj fragment) are fragments wh
t is a F, t is not a F	(if F is a noun fragment)

demands are obtained by deleting PO.1/ and whose category remains that of F in each case, provided that such deletion does not leave the demand null; if null demand does result, the "filled" fragment is assigned the category of sentence.

5. If t is a term, p a non-null placer, F a fragment whose demand contains p then Fpt is a fragment whose demand is obtained by deleting p/ from the demand of F, and whose category is that of F, provided that such deletion does not leave the demand null, otherwise, sentence.

6. If t is a term, F a fragment in whose demand a lowest null placer, p, exists but with $p \neq PO.1$ then Ft is a fragment whose demand is obtained from that of F by deleting p/, and whose category is that of F, provided that such deletion does not leave the demand null, otherwise, sentence.

7. If t is a term, F a fragment whose demand does not contain PO.1, p the lowest null placer in the demand of F, then tF is a fragment whose demand is obtained from that of F by deleting p/ and whose category is that of F provided that such deletion does not leave the demand null, otherwise, sentence.

8. If t is a term, p a non-null placer, F a fragment whose demand contains p but not PO.1 then ptF, Fpt are fragments whose demand is obtained from that of F by deleting p/ and whose category is that of F provided that such deletion does not leave the demand null, otherwise, sentence.

9. If F and G are fragments of identical category and demand then

NF, BF.G, EFvG, F.G, FvG, F.*G, Fv*G

are fragments of the same category and demand, provided that this does not result in a final term of F being separated from a first term of G by a connective. In such a case, the same compounds may be formed but with a comma preceding the connective.

Stipulation 3. simply includes predicates among the fragments and thus establishes some fragments, at least, as having a given demand.

4. says in effect that if you prefix a term to a fragment which demands a subject, in the copular setting required by the given category, you get a fragment which does not demand a subject. In this way, the concept of demand is extended from predicates such as "beats", with the demand PO. 1/PO. 2/ to strings such as "John beats" which would get the reduced demand PO. 2/.

Examination of later clauses will reveal that no other way of deleting a subject demand is provided other than use of the proper copular setting specified in stipulation 4.

5. says that terms with a demanded non-null placer can always be added behind any fragment; e. g. suppose "takes" has the pattern x takes y from z to w", then to the fragment "John takes George" one may add to "to NY and then "from Chicago" to get the permuted, but understandable, pattern "John takes George to NY from Chicago".

6. says that placerless terms can also be added behind but they will be interpreted as filling the lowest placerless position demanded, but never that of subject. E. g., assume the placerless pattern "John

gives George Fido" for "gives". Then adding Fido to the fragment "John gives" whose demand is PO. 2/PO. 3/, deletes the lowest placer, PO. 2, i.e. that which accompanies the indirect object in this pattern.

7. says that when a fragment has a subject already, placerless terms may be added in front if they can be regarded as satisfying a lowest placerless demand at the fragment.

8. says that placed terms may be added before or after a fragment demanding them, provided it already has a subject.

With each concatenation of the above sorts to a fragment, its demand becomes less. This permits a certain "adjustment" of fragments so that they can form compounds under stipulation 9. Thus, "x gives y to z" is a triadic pattern with demand PO. 1/PO. 2/TO/ while "x waves to y" is a dyadic pattern with demand PO. 1/TO/ (Where "to" has been left uncoded for readability.) But "gives Fido" is a fragment with demand PO. 1/TO/ which is the same as that of "waves" which permits the compound "gives Fido and waves" with the same demand. With the remaining demands satisfied, e.g. "George gives Fido and waves to Mary", we have an English II sentence.

The clause in 9. about the comma is to avoid ambiguity in a sentence such as "Albert likes Betty and Cathy and Dora likes Ernest."

In view of the characterization of "sentence" as a fragment with null demand in the foregoing stipulations, it might seem that "sentence" was now defined and that the characterization of English II was therefore completed. We would indeed have defined, at this point, a language of

names and predicates, with factoring and grouping. This language, called LFG (LOGOS with factoring and Grouping), was programmed separately and reported on in an earlier report.[2] Actually, of course, the stipulations have only established that filled fragments are among English II sentences; further stipulations could specify other forms. The extension to general sentences (e.g., universal and existential) undertaken in the next section, however, proceeds instead to extend the concepts term and (noun) fragment.

Before proceeding with the extension to general sentences, it may be helpful to sketch the procedure whereby sentences involving just factoring with names and predicates would be transformed to English I and to discuss some general points of interest in factoring.

The transformation takes place, roughly speaking, in the following steps:

1. Recognition of compound individual expressions, assigning working names to each so that the sentence is rewritten with these names, without apparent compound individuals.
2. Build-up of a dictionary of recognized fragments with their demands until the sentence can be represented as a truth-function of filled fragments.
3. Distribution within each filled fragment of the individual "working names" so that each simple predicate is finally the sole predicate in its own filled fragment.

4. Successive distribution of each simple predicate over the compound individuals expressions which are its actual arguments. (The first two steps are actually carried out in a single sweep of the sentence.)

It is worth pausing at this point to present some general observations on factoring.

One point of interest in factoring is that it represents a purely syntactical abbreviatory device. Throughout the history of logic, the central abbreviatory device has been that of defining a new term to stand for a longer expression. In factoring, no new term is introduced. Instead, rules are added which permit expansion with the help of the already available logical connectives. Furthermore, this purely syntactical expansion has no pre-set limit. A defined term, on the other hand, permits only a single fixed saving.

A second point is that factoring can often replace the need for variables in logic or pronouns in natural language. Thus "$Ax(Px \supset Qx)$" can be abbreviated to $A(P \supset Q)$[10]. Also, in "You saw John and I saw him", the "him" is eliminated in the factorization "You and I saw John". Factoring when coupled with some way of representing permutations of predicates, (e.g., converses) can, in fact, obviate variables altogether.[9]

Factoring, as mentioned, can also act as a grouping device. Thus, "A is P or A is Q and B is Q" is ambiguous without a precedence system. With the present precedence system, it would be grouped "A is P or (A is Q and B is Q)". But, if the grouping (A is P or A is Q) and B is Q"

were intended, at least one explicit grouper would be needed. "(Both) Either A is P or A is Q and B is Q". However, factoring can achieve this grouping without using groupers "A is P or Q and B is Q".

This grouping function of factoring may well play a vital role in human comprehension of linguistic communication, since abstract, thought quite apparently operates by bunching complexities into simpler units for batch processing. For example, the (factored) input of example 21 in the Printout Appendix seems definitely easier to understand than the distributed output.

In natural language, factoring produces many more kinds of fragments than those mentioned. There are, as already remarked, compounds belonging to every syntactical category and every fragment of such categories. Even proper names are fragmented and factored: "Dr., Mrs., and Joanna Brown".

Factoring occurs also with the logical connectives acting like relations having whole sentences as arguments. Thus "A if and only if B" can be regarded as a factored form of "A if (B) and (A) only if B". It is even possible to regard the legal "A and/or B" as short for "A and or or B" which could be a factorization of "(A and B) or (A or B)" where the "or" is read in the exclusive sense. That is, in symbols:

$$A((.) \not\equiv (\not\equiv)) B \quad \equiv \quad (A.B) \not\equiv (A \not\equiv B) \quad \equiv \quad A \vee B$$

Logicians sometimes informally factor out quantifiers, e.g., "AxyzEwuv Rxyzwuv" for "AxAyAzEwEuEvRxyzwuv".

It is apparent that no syntactical device is immune to factorization, whether it plays any independent semantic role or not. It is not known whether a general rule could be given, even for a relatively simple language, that would accommodate every possible unambiguous factorization.

When iterations grow very large, even factoring is not enough and the sentence form itself tends to be abandoned in favor if lists, tables, etc., in which the factored predicate, or magnitude designation, appears only as a heading. Nevertheless, even here an understanding of the factoring phenomenon may at least throw some light on the exact communicational role of lists and tables and correctly indicate their place in the more general framework of communication in sentence form.

We now return from these informal observations on factoring to the methods used for the formulation of universal and existential sentences in English I and II.

General Sentences

To express general propositions (universal and existential), English I uses variables, "z_1", "z_2", "z_3"... and quantifiers, "For-every" and "For-some", symbolized and coded as shown in Table I, so that English I is essentially a spelled-out version of predicate calculus in Polish notation, but with the retained connectives and predicational pattern described earlier.

English II uses no variables. It expresses generality with determiners, "every", "no", "any", "a", "some", and the relative pronouns, "which", "who", "whom". The latter permit formation of dependent clauses, nested and compounded.

We proceed forthwith to give the formational stipulation for determiners, followed by examples and a preliminary discussion of the intended logico-semantical functions of determiners.

Determiners

10. if d is a determiner and N is a noun fragment whose demand consists just of PO.1/ then dN is a term.

Such new sorts of terms include "every nation", "a pilot", "some friend of George", "any gift from Schmidt to Casey", "no shipment from either Tabu or Uuno to Manila or Djakarta". Such terms can, under the already given stipulations, fill any argument position in a predicate or a compound fragment. We also get such terms as "every associate of any Senator", "no flight between Tabu and any entrepot of an ally of any nonSEATOnation. Such examples can be manufactured for any given length by iterated use of predicates of other than first degree, though of course human users would not tend to avail themselves of such possibilities. With respect to the run-together name "nonSEATOnation", it should be remarked that not only does the present program require single-word names but that it can also not admit hyphens due to a conflict between our dictionary handling technique and a certain restriction in the SNOBOL 3 systems (string names may not use hyphens). But an easy, though slightly machine-time-expensive, elaboration of technique can remove both this and the single word-name restriction.

In order to translate English II sentences containing general terms involving determiners into English I sentences which express generality by the quantifier variable system, it is clear that the translating program must, when encountering a determiner, generate a variable as yet unused

in the given translation process, substitute it in the argument place where the general term appeared, and then, in some way, insert both (1) a quantifier followed by the variable, and (2) a qualifying phrase containing the new variable specified by the (possibly very complex) noun which follows the determiner, somewhere in the sentence, together with appropriate scope indications.

As may be expected, each determiner requires a special rule.

While the full treatment of determiners in English II can not be described until the grammar specification is completed, the basic principles of their use can be illustrated in the framework of a simpler language in which neither copular settings, factoring, nor relative pronouns are admitted. We shall call this simplification of English II L II. For its exact characterization see Bohnert (1962b).

For diagrammatic clarity, we use the determiner symbols shown in Table I, e.g.,

e	every
a	a
u	any
s	some
n	no

in conjunction with the other "Text symbolic" symbol, shown there, and with informally chosen predicate letters. Thus,

"Every businessman gambles"

might be represented as

eBG .

In English I, the corresponding sentence

"For every z_1 if z_1 businessman then z_1 gambles could be represented as

$$Az_1Iz_1B \supset z_1G$$

or in parenthesis notation

$$Az_1(Bz_1 \supset Gz_1)$$

As shown in these representations, "Every man gambles" has the form of a simple subject predicate sentence with the general term "every man" as subject. The logic versions, on the other hand, show it as a generalized compound sentence, in this case the universalization of an if-then, (or conditional or implication) formula.

Following the steps outlined, the transformation may be carried out by

1. generating a variable, e.g., z_1
2. substituting it for the general term, getting z_1G.
3. in placing the new variable, as subject, in a qualifying phrase provided by the general term, giving us z_1B
4. and, following the special rule for e, placing these pieces, with I and $\supset$, in the conditional word order shown.

The rule for e will also require that the insertion of this qualifying quantified clause - take place within whatever context the whole predicational unit which contains the general term itself occurs in. Thus, suppose we provide a context consisting only of a preceding negation, e.g.

"Not every businessman gambles" or "NeBG". The stated context rule requires the insertion of "Az_1Iz_1B" within the context of "eBG", i.e., after the negation. This gives

$$NAz_1Iz_1B \supset z_1G$$

or in parenthesis form

$$\sim Az_1(z_1B \supset z_1G)$$

which is, of course, the normal logical rendition of the given sentence.

In order to state the last rule more fully, and similar rules for the other determiners, it will be convenient to symbolize the (possibly null) leading and following contexts as M and W respectively, and let dRS stand for a single predicational pattern, without inversions of the dictionary-given word order, in which dR is the first general term to appear (where d is the determiner being studied). This notation may be illustrated by the following:

Suppose "If George rejects every best seller then contract 17 terminates", is symbolized as

$$IgReB \supset cT$$

Then eB is the first general term in the predicational pattern gReB, which is, then, represented eBS, where S does not represent a predicate, but rather a metalinguistic transformation on eB. Applying the same transformation to z_7 for instance, we would have

$$z_7S \rightarrow gRz_7$$

or in words, the S-transform of "z_7" is the string "George rejects z_7.

The whole sentence, then, may be symbolized as M eBS W, where

M, the leading context, stands just for "if", and W, the following context, stands for "then contract 17 terminates".

In stating the rules with the help of this symbolism we shall assume that the leading context, M, is either the null string or free of general terms, having already passed through the determiner elimination process which is being recursively characterized. In the same vein, in the transformation symbolism, dRS, in which the general term dR is the first such term to occur, we admit the possibility of earlier argument positions being occupied by variables resulting from earlier determiner elimination steps (whose quantifiers and qualifying clauses are already in the leading context string represented by M). Thus, the rules to be stated picture, so to speak, a moment in a left-to-right sweep of a given sentence when "the next" general term to be eliminated is encountered and then exhibit the result of the described single elimination step. Such elimination always involves generation of a variable not yet used in previous eliminations. (The present program does this simply by counting steps and concatenating the next numeral to "z", e.g. "Z1", "Z2", etc.) The generated variable (letter and numeral) is represented by "z" in the schematism below.

The rules for determiner elimination in the simplified illustrative language are the following:

Rule for e (every)

M eRS W → M AzIzR⊃z S W

M Az(zR⊃zS) W

Rule for u (any)

M uRS W → AzIzR ⊃ M zS W

Az(zR ⊃ M zS W)

Rule for a (a, an)

M aRS W → M EzBzR . zS W

M Ez(zR . zS) W

Rule for n (no)

M nRS W → M AzIsR ⊃ NzS W

M Az(zR ⊃ ∼ zS)W

Rule for s (some, in the sense of "some certain")

M sRS W → EzBzR . M zS W

Ez(zR . M zS W)

The two readings on the right correspond to English I and parenthesis notation, respectively, except for the leading argument variation in the parenthesis notation.

We now study the effect of these rules in a number of examples, showing first a possible input sentence, which we will call stage 1, s1, shown both in symbolized and spelled form. Then come the transformations s2, s3, ... which occur each time a determiner is encountered in reading the latest transform from the left. The last numbered transform in each sequence is the resulting parenthesis form sentence. Subsequent unnumbered transformations, marked with an arrow, are sometimes carried out according to transformation rules of logic itself to bring the result into a more readable form. Occasionally a further

transformation will be given, back into English II or natural English for purposes of comparison, but this is meant informally since the reverse translation algorithms have not been stated.

1.	s1:	eBG	Every businessman gambles.
	s2:	Ax(xB ⊃ xG)	
2.	s1:	mHaL	Mary has a lamb.
	s2:	Ex (xL. mHx)	
3.	s1:	jReB	John reads every bestseller.
	s2:	Ax(xB ⊃ jRx)	
4.	s1:	jRuB	John reads any bestseller.
	s2:	Ax(xB ⊃ jRx)	(Equivalent to 3)
5.	s1:	~(jReB)	John does not read every bestseller.
	s2:	~Ax(xB ⊃ jRx)	i.e. John reads no bestseller.

In 3 and 4, "every" and "any" seem to have the same meaning. But in other contexts, such as the negative forms of 5 and 6, their meanings diverge. It is the merit of our rules that this seeming discrepancy is shown to be the result of the straightforward operation of simple rules for each sign. The essential distinction between the two rules concerns the scope of the implied operator and the location of the qualifying clause. The idea harks back to Russell, who likens the behavior of "any" to free variables and that of "every" to bound variables. The idea has been touched upon and developed somewhat by various writers since then (Quine, Carnap) but apparently has not been incorporated in any formalized natural language model.

As a further example of a context in which the meanings of "every" and "any" diverge, but in a way accounted for by our rules, consider 7 and 8 below. Here a small liberty is taken with the symbolism in that "W" is meant to stand for the whole sentence "War occurs" or "There is a war". The analysis of such a sentence is inessential to the point of being illustrated.

7. s1: $eSH \supset \sim W$ — If every soldier stays home, there is no w

   s2: $Ax(xS \supset xH) \supset \sim W$

8. s1: $uSH \supset \sim W$ — If any soldier stays home, there is no war

   s2: $Ax(xS \supset (sH \supset \sim W))$

   $\rightarrow Ax(xS.xH \supset \sim W)$

   $\rightarrow Ex(xS.xH) \supset \sim W)$

   $\rightarrow aSH \supset \sim W$ — i.e. If a soldier (at least one) stays home, there is no war.

Both in English and in the model language, 7 is a truism and 8 is a "falsism". These sentence forms are shown in the printout in the variation "If every guest declines theparty fails", "If any guest..etc.".

Just as the narrow scope "every" is accompanied by the broadscope "any", we would expect the narrow-scope "a" to have a broadscope correlate with the existential operator. As shown in the rules given earlier, "some", symbolized "s", has been given this role. This reads well in many test contexts. However, actual English often prefers other locutions, such as "a certain" to effect the lengthening of scope. Some of the examples below may sound more clearcut if "a certain" is substituted for "some".

9. s1: $\sim$jKaG — John does not know a girl

s2: $\sim$ (Ex(xG. jKx))

→ Ax(xG ⊃ $\sim$jKx)

→ jKnG — John knows no girl

10. s1: jKsG — John does not know some (certain) girl.

s2: Ex(Gx. $\sim$ jKx) — There is a girl John does not know.

Examples 9 and 10 exhibit the effect of short and long scope in a and s, as examples 5 and 6 did for e and u, in the simple context of negation. It is admitted that English is less compelling in its scope readings for these words than for "every" and "any", but they seem to hold up in many complex contexts such as those next considered. (Part of the problem of a word like "a" is that it presumably has other functions to perform, such as that of providing reference for later occurrences of definite singulars as described in Quine (1960)).

11. s1: eFaWFsG — Every friend of a ward boss is a friend of some gangster (or of a certain gangster).

s2: Ax(xFaW ⊃ xFsG)

s3: Ax(Ey(yW. xFy) ⊃ xFsG)

s4: Ex(xG. Ax(Ey(yW. xFy) ⊃ xFz) — i.e. There is at least one gangster z such that every friend of a ward boss knows z.

12. s1: eFsWFaG — Every friend of some ward boss is a friend of a gangster.

s4: Ey(yW. Ax(xFy ⊃ Ez(zG. xFz)) By similar steps.

Examples for n:

13. s1: jRnB John reads no bestseller

s2: Ax(xB ⊃ ∼jRx) equivalent to example 6

14. s1: nSRuB No student reads any bestseller

s2: Ax(xS ⊃ ∼xRuB)

s3: Ay(yB ⊃ Ax(xS ⊃ ∼xRy))

→ Ay(Ax(yB. xS ⊃ ∼ xRy))

15. s1: nSRaB No student reads a bestseller.

s2: Ax(xS ⊃ ∼ xRaB)

s3: Ax(xS ⊃ ∼(Ey(yB. xRy)

→ Ax(Ay(xS. yB ⊃ ∼ xRy)) equivalent of 14

16. s1: nSRsB No student reads some (certain) bestselle

s2: Ax(xS ⊃ ∼ xRsB)

s3: Ey(yB. Ax(xS ⊃ ∼ xRy)

17. s1: nSRnB No student reads no bestseller.

(awkward but intelligible)

s2: Ax(xS ⊃ ∼ xRnB)

s3: Ax(xS ⊃ ∼ (Ay(yB ⊃ ∼ xRy))

→ Ax(xS ⊃ Ey(yB. xRy)) i.e. Every student reads a bestseller.

It should be emphasized that the above determiner rules apply only to languages simplified in the way described. When factoring is admitted, for example, the order in which grouping, distribution, and determiner elimination steps are carried out is crucial to the "reading" of the sentence.

Choosing the best possible sequence of operations becomes, indeed, one of the most complex problem areas in this approach to the logic of grammar.

Nevertheless, the rules shown illustrate the basic conception to be followed, and already shed some light on some interesting linguistic questions. We have already seen in the foregoing, the "every-some" ambivalence of "any" accounted for in terms of scope; also the scope lengthening effect of phrases like "a certain" or "some certain".

Now we turn to the further questions: one concerning an important ambiguity of "is" and the other an odd-phenomenon associated with the active-passive transformation.

"Is": Predication and Identity

"Is" is notoriously ambiguous. We shall not attempt a catalogue of all the meaning variations linguists, lexicographers, and logicians have found in it (e.g. identity, genidentity, class membership, class inclusion, existence, location, synonymy, etc.) but shall consider just the conflict between the "is" of identity (as in "Venus is (identical with) the nearest planet") and the predicational "is" used in ascribing a property to something, e.g. "Mt. Everest is high" (not identical with high). This clash may already have disturbed the reader of this study in finding "a star" treated as a term in a context such as "A star rose", while in "Betelgeuse is a star" the "is a" is swept away as mere syntactical setting, leaving "Betelgeuse star", in English I, as a simple predication with star as predicate.

Suppose that in the latter example "is" is read as identity (symbolized by "=" and that "a star" is treated as a term, i.e., as the second argument in the dyadic identity relation. Then "Betelgeuse is a star" may be symbolized "b=aS". Using the determiner elimination rule for "a" we have

$$Ez(zS.\ b=z)$$

or "For some z, z is a star and b is (identical with) z". This can be shown to be logically equivalent to the simpler predication bS, or "Betelgeuse (is a) star".

Such a proof, interestingly enough, can not be carried out in the simple first order logic we have so far used but only in first order logic with identity, a somewhat stronger system which, while still syntactically first order, requires a special set of axioms for identity. Taking the obvious equivalence of meaning for granted, however, we can verify that the proposed identity reading for "is" consistently reacts with general terms in a way consistent with our decision to treat them as terms. That is, in a language without adjectives (which require the predicational reading) we could always read "is" as identity. The point may be illustrated by recasting earlier examples.

s1: $eB = aG$ Every businessman is a gambler.

s2: $Ax(xB \supset x = aG)$

s3: $Ax(xB \supset Ey(yG.x = y))$

$\rightarrow$ $Ax(xB \supset xG)$ Which is equivalent to example 1, as it should be.)

s1: eFaW = aFaG Every friend of a ward boss is a friend of a gangster.

s2: Ax(xFaW ⊃ x = aFaG)

s3: Ax(Ey(yW.xFy) ⊃ x = aFaG)

s4: Ax(Ey(yW.xFy) ⊃ Ex(xFaG.x=z))

s5: Ax(Ey(yW.xFy) ⊃ Ez(Ew(wG.zFw).x=z))

→ Ax(Ay(yW.xFy ⊃ Ew(wG.xFw))) i.e. Every friend of any ward boss is a friend of a gangster.

This representation also has a certain historical interest for logicians. (others are invited to skip this brief digression.) Ancient laws of the syllogism are phrased with the concept of a distributed term. A class term is said to be distributed in one of the four traditional propositional forms, if the proposition says something about each member of the class. It is also said that it is the subjects of universal sentences and the predicates of negative sentences which are the distributed terms, that is, the underlined terms below, in the traditional AEIO scheme.

A. Positive Universal	E. Negative Universal
All P's are Q's	No P's are Q's
I. Positive Particular	O. Negative Particular
Some P's are Q's	Some P's are not Q's

Just why the underlined terms are said to be distributed in the sense that the proposition says something about each member is not equally clear in all cases. It is obvious enough for the A case. It becomes clear in the E case for the P term at least when it is rendered

in modern symbolism as $Ax(xP \supset xQ)$, since this makes the universal quantifier explicit. This then clarifies the Q term also when the equivalent contrapositive, "no Q is P" is symbolized in turn.

But the Q term in the O case is symbolized as $Ex(xP. \sim xQ)$ and no usual transformation turns up a prefixed universal quantifier, one does show up, however, if we express the four propositions in our English II singular general terms with determiners, with "is" read as identity, and with a rule (adopted in English II) that when a negated term (syntactically permitted in our earlier recursive stipulations) is encountered at a certain argument position the negation is transferred from the term to the predication in which it then stands before its determiner is eliminated. We illustrate the effect of our general-terms-with-identity parse on all four cases, but direct special attention to the O case.

A. s1: $eP = aQ$ Every P is a Q

s2: $Ax(xP \supset x=aQ)$

s3: $Ax(xP \supset Ey(yQ. x=y))$

→ $AxEy(xP \supset (yQ. x=y))$

E. s1: $nP = aQ$ No P is a Q

s2: $Ax(xP \supset \sim x = aQ)$

s3: $Ax(xP \supset \sim Ey(yQ. x=y))$ (short scope rule for a)

→ $AxAy(xP. yQ \supset \sim x=y)$

I. s1: aF = a — a P is aQ

s2: Ex(xP.x = aQ) — (Same transformations occur for

s3: Ex(xP.Ey(yQ.x=y)) — "some P is a Q" sP = aQ.)

→ ExEy(xP.(yQ.x=y))

O. s1: aP = ∼ aQ — a P is not aQ

s2: Ex(xP. = ∼ aQ)

s3. Ex(xP. ∼ x=aQ) — Transfer of negation when negated term is encountered)

s4: Ex(xP. ∼ Ey(yQ.x = y)) (Short scope rule for a)

→ ExAy(xP.(yQ ⊃ ∼x=y)) (Same transformation results from Some P is not aQ)

It will be noticed that in each of the final formulas a universal quantifier appears in the prefix corresponding exactly to those terms, which have been called distributed.

Unfortunately, "is" is ambiguous and presents an obstacle. It might be urged that since "is" acts predicationally before adjectives and could be consistently, if clumsily, interpreted as "=" before general terms, proper names (and other items not yet entering our analysis such as "the" phrases, pronouns, functors), that a systematic dependence on context could be built into a sufficiently orderly model of English. This may be so but there would be at least considerable clumsiness in devising a system which would take a compound of adjectives and general terms, such as "Smith is experienced, able, a negro, and a war veteran" and shift treatments of "is" during distribution.

It is because of such difficulties that English II has admitted certa crudities in its handling of "is a".

When the Active-Passive Transformation Involves General Terms

"John likes Mary" means the same as "Mary is liked by John". This seems, at first glance, an instance of a general rule that "x likes y" means the same as "y is liked by x". But we then note that "No girl is liked by every boy" is different in meaning from "Every boy likes no girl". At first glance, it appears we must either abandon the general rule of active-passive equivalence or attempt to force the same meaning on the two general term sentences. Luckily there is a way in which we can have both our rule and the apparent exceptions too. Since our underlying logic uses variables, our rule could be stated using them just as given above. But we have not specified a logic for English II except by saying that the logical relations which hold between English II sentences are just those which hold between their translations into logic (by the program). In particular, we have not broadened the underlying logic by adding a rule that general terms of English II may be substituted for variables (e.g. in any law of logic or any definition).

Thus, in the formula "x likes y if and only if y is liked by x" we can not substitute the general terms "no girl" or "every boy". The only way to study the logical relation between the two sentences is to translate each into its logical equivalent by sequential elimination of determiners

and compare the results. When we do this, we discover that we have exhibited the very difference of meaning we instinctively felt in the first place. In the following elimination process we symbolize the passive formation of L (likes) by $\breve{L}$ (is liked by).

s1: nG$\breve{L}$eB — No girl is liked by every boy

s2: Ax(xG ⊃ ∼ x$\breve{L}$eE)

s3: Ax(xG ⊃ ∼ (Ay(By ⊃ x$\breve{L}$y)))

→ Ax(xG ⊃ (Ey(By. ∼ x$\breve{L}$y)))

s1: eBLnG — Every boy likes no girl

s2: Ax(xB ⊃ xLnG)

s3: Ax(xB ⊃ Ay(Gy ⊃ ∼xLy))

Ax(Ay(xB. yG ⊃ ∼ xLy))

A similar explanation (within a higher order logic) can be given for the variation between: "Everyone in this room speaks two languages", "Two languages are spoken by everybody in this room". In these, and similar examples, we see evidence that the order of occurrence of indefinite singular terms or other quantificational idioms in English sentences has much the same significance as the order of operators in a sentence of formal logic, and that the present left-to-right scan technique accurately preserves this parallelism.

Relative Pronouns and Dependent Clauses

The grammatical "power" of English II will now be advanced by the incorporation of dependent clauses introduced by the relative pronouns "who", "whom", and "which". These in turn, are then used to make a further broadening of the concept of term. As in earlier sections, the stipulations are given first, and then illustrated and discussed.

11. If w is a relative pronoun and F is a fragment whose demand consists just of PO.1/ then

    wF if F is a verb
    w is F if F is an adj
    w is a F if F is a noun

    is a relative clause.

12. If w is a relative pronoun and F is a fragment whose demand consists just of p/, $p \neq$ PO.1, then

    pwf if p is not a null placer
    wF is p is a null placer

    is a relative clause.

13. If R and S are relative clauses then

    R.S, RvS are relative clauses.

14. If d is a determiner, n a name, F a fragment of the noun category whose demand is just PO.1/, and R a relative clause, then

    dPR, nR are terms

These stipulations may be illustrated as follows: Simple applications of 11: "which hurts", "who is present ", "who is a musician".

Simple applications of 12: "over whom Napoleon triumphed."

"which George bought", "to whom George gave a toy"

Simple applications of 13: "from which coffee is exported and to which iron is imported", "which radios or from which a flare is fired".

Simple applications of 14: "every bill which is outstanding"

"no student who lacks a pass", "George who is a musician"

More general relative clause: "To whom George or Anne gave a toy which buzzes or an instrument which plonks or toots or who was backstage"

More general term: every man or woman or child who was present or to whom an invitation to every theater which participated was sent"

It should be noted that the relative clause rules extend the term concept but do not, independently, extend the fragment concept. They do not, for example, admit a monadic fragment such as "civilian who informed or soldier who deserted or official against whom a complaint was filed." Such a combination would seem to be admitted by a rule of the following form, acting together with the fragment combination rule.

If F is a fragment with demand PO. 1, R a relative clause then FR is a fragment with demand PO. 1.

While such fragments occur in English, their analysis within the present system raises certain problems akin to the scope problems within general terms discussed in Quine (1960a). The above rule, in any case, will not do, since while one application might produce the permissible "civilian who informed", iterated application would produce, e.g., "civilian who informed who informed who deserted against whom .....etc."

Prepositional Phrases

Before leaving the syntax of relative clauses, it should be remarked that the familiar grammatical category of prepositional phrases consists largely, if not entirely, of relative clauses from which the relative pronoun has been dropped along with any copular setting with the placer, if any, which had accompanied the relative pronoun removed to the end of the clause.

Consider the example: Every stoplight between a school crossing intersection and a certain intersection near a department store is on auto-control." The prepositional phrases are easily, if inelegantly, transformed into relative pronoun introduced relative clauses as follows: "Every stoplight which is between an intersection which is a school-crossing and a certain intersection which is near a department store is on auto-control."

Just as the grouping of the program is able to insert missing groupers, so an additional branch could either convert prepositional

phrases to dependent clauses or, working directly, subject them to the analogous transformation.

The details must, however, be left for future investigation.

Translational Principle for Relative Pronouns

The principle underlying our translation technique for relative pronouns can be put informally as follows. When a relative pronoun possibly preceded by a placer is encountered in a left to right sweep during what we shall call the determiner elimination phase of the processing, its immediate predecessor will be either a formula in which a variable has just replaced a determiner by a preceding determiner elimination or it will be a proper name. Following the relative pronoun there will be a fragment whose demand (always a single placer) will already have been computed. The relative pronoun is, then, replaced by the preceding name or variable. This with the accompanying placer, if any, is then shifted to the argument position indicated by the demand of the following fragment (if it is a simple predicate, otherwise it, with the fragment, form a new fragment with null demand to await a later distributional phase). This well-formed formula inserting the "both" grouper and the "and" connective.

Consider the sentence "every child to whom a toy is given is happy" or eC to w aT is G is H.

The first determiner elimination step yields:

AxIxC to w aT is G ⊃ xH.

Preceding the "to w" is a formula in which "x" has just replaced the determiner "e". The fragment "a toy is given" demands a "to y" completion. More exactly, its demand is TO/, with the dictionary revealing that in the natural order "TO" is the third position marker. The preceding x then replaces the w and the positional "to x" is shifted to its natural position, giving the well-formed formula

a toy is given to x, or, aT is G to x.

Forming the conjunction, the sentence becomes

AxIBxC. aT is G to x $\supset$ xH

Elimination of the determiner a now yields

AxIBxC. EyByT. y is G to x $\supset$ xH

i.e., for every x if both x child and for some y both y toy and y is given to x then x happy"

The rules for eliminating relative pronouns can be given in the symbolism earlier employed to describe determiner elimination:

$$M\ zS_1pwS_2\ W \rightarrow M\ BzS_1.pzS_2\ W$$

$$M\ zS_1wS_2\ W \rightarrow M\ BzS_1.zS_2\ W$$

where both S_1 and S_2 are predicational transformations, as S alone was in the determiner elimination rules.

The Recognition-Translation Method

The task of constructing a mechanical recognition method for a recursively defined set of strings is seldom easy (and may be impossible: the class of first order theorems is recursively definable but it is known that the problem of finding a first order decision method, i.e., a mechanical method for recognizing theorems, is recursively unsolvable). For English II the problem is solved, but the method requires techniques and concepts not obvious from the recursive stipulations given.

The method used involves several major phases, most of them involving iterations or recursions of levels corresponding to the various sorts of nesting that may appear in an English II sentence.

The first phase is simple enough. It translates the spelled input into the SNOBOL code (thereby checking that all the words encountered belong to the logical vocabulary or the temporary dictionary). The SNOBOL code is needed not only for its brevity but for the syntactical information that is packed into each code word, precedences, categories, etc.

The second, or parse, phase analyzes the input string by an aggregating process in which it

1. lumps terms together, giving each maximal string of terms it encounters an auxiliary symbol beginning with an "I" (for individual) which it thereafter treats as a proper name. This is, of course, a

recursive process since terms may contain general terms and dependencies which may contain further general terms, etc. Therefore, the auxiliary I symbols are coded according to a level system.

2. Picks out at a given momentary level a sequence of substrings which we shall call fractions. Each fraction consists of a single dictionary predicate accompanied by whatever arguments and groupers are not separated from the predicate by connectives. Thus in

A both gave B and sold C to D

"A both gave B" and "sold C to D" are fractions.

By consulting the dictionary given demands of the predicates in each fraction and computing the demands of the fractions (disregarding imbedded groupers), it identifies which fractions can merge with which to form fragments whose demands it computes in turn until the string under consideration can be regarded as a compound of minimal length null demand fragments called L atoms to each of which the parser assigns an auxiliary symbol beginning with an L and numbered according to its level and position.

The result of this phase is to exhibit the sentence as a Boolean combination of simple predications but with the "names" and "predicates" involved being auxiliary symbols actually representing e.g., compounded general terms, fragments, etc. A temporary dictionary is built up for such auxiliary predicates, assigning them (computed) demands.

The third phase eliminates determiners and relative pronouns according to the general principles stated earlier, but working from the

inmost and lowest level components outward. Copulas are tailored away in the process.

The fourth is the distribution phase. Within each filled fragment the individual "working names" are distributed to the components of the fragment, until each simple predicate is finally the sole predicate in its own filled fragment. The individual terms represented by the working names are now Boolean combinations of proper names and variables. The predicates are distributed over these compound individuals.

The fifth, or output phase, by minor trimmings and replacements translates the transformed SNOBOL string into the format desired: English I, parenthesis notation, Polish notation, or parenthesis notation in which the elementary predications (atoms) are spelled out.

CONCLUSION

In this final section we review the accomplishments of the project, the insights gained, and the difficulties encountered, and comment on the theoretical and practical potential of logic-based, machine-implemented analyses of natural grammar of the type here exemplified.

Speaking first in a general way, we have shown by example that syntactical ambiguity (in the sense of current linguistics, i.e., the well-formedness of a string can be established by more than one sequence of formation rule applications) need have little relation with semantic ambiguity for a machine, provided that its read-parse algorithm is equipped with adequate resolution rules (e.g., of grouping, scope, distribution, etc.) This raises a serious question as to whether formation rules alone provide an adequate explication of "grammar". It may be claimed that such ambiguity-resolving parse algorithms step beyond syntax into semantics, but actually they make use only of syntactical information (if it be granted that the association of numbers such as degree and precedence with elements of vocabulary is not a semantical step.)

We have demonstrated to our own satisfaction that the embodiment of parsing algorithms in a computer program is an almost indispensable heuristic procedure. Not only have program runs repeatedly revealed subtle errors in algorithms which showed up only in examples too complicated to have been analyzed by hand, but it has, on occasions, provided valuable positive suggestions as to possible simplifying paraphrases.

One example concerned the ambiguity in the scope of a relative pronoun following a compound expression. In the following two sentences, for example, the scope of "who" varies.

1. Any Ph D or applicant who has seven years experience is eligible for the job.
2. Any instructor or professor who has taught seven years is eligible for a sabatical.

In the first, "who" refers only to the applicant; in the second, "who" distributes to both instructor and professor. In accord with the policy of selecting only one out of several competing rules for English II, we chose the first sentence as our paradigm, but then had difficulty in finding an economical paraphrase for the idea expressed in the second. Later, a slight change in the program for another reason had the unintended effect of reversing our decision on this point, and an example involving compound general terms provided us with a possible paraphrase for the first sentence, i.e.,

Any PhD and any applicant who has seven years experience is eligible for the job.

This is not graceful English, but it is understandable, not cumbersome, and translates to the correct logical formulation, and thus serves our approximative purpose for the present stage of modelling.

At other times, we have been led to more deliberate heuristic use of the program, e.g., when the grammar becomes too complex to permit examination of all consequences of a new change. This was

particularly the case in testing rules governing negation, since in English II negation appears in many contexts, negated names, negated predicates, negated general terms, negated compounds, negated fragments, negations imbedded in dependent clauses, etc.

Speaking more particularly, we have explored the natural language correlates of the logical concepts of degree, grouping, quantification, scope. We have embodied and coordinated in a single machine-parse system, several longstanding suggestions of logicians:

(1) That the logical concept of predicate degree offers a basis for a unified understanding of the role of cases, and prepositions, the transitive-intransitive distinction, the active-passive relationship, and related phenomena (especially stressed by Reichenbach);

(2) That words like "if", "either", "both", act like truth-functional groupers (most recently remarked on by Quine);

(3) That "any P" behaves logically like an unquantified variable restricted to the domain P (Russell);

(4) That pronouns perform, to some extent, the function of variables in logic.

In following the first line of suggestions (concerning degree), we have been led to the concept of placer as a basic grammatical function category as yet little recognized by grammarians. The phrase "grammatical function category" may be understood, for present purposes, by reference to familiar remarks of grammarians such as that in such-and-such a context a certain phrase functions adverbially. With the

placer concept, we may say that prepositions often function as placers (as do words belonging to other traditional categories: the "and" of "between x and y") but that they also may function as independent dyadic predicate adjectives ("x is in y").

We have amplified the second line of suggestions with our grouper-avoiding system of precedences, extended by our conjecture that the grouping effect of such phrase as "and furthermore" may be modelled by a system of multiple precedences.

The third line of suggestions, concerning "any", has been amplified so as to make variations of quantifier scope a primary consideration in the analysis of natural language. Thus the effect of "some", in the sense of "some certain", is attained with our long-scope rule.

Indeed, the very idea of translating general terms into expression involving quantifiers and variables raises the question of the order of quantification, in a way little suggested by traditional grammar itself.

Our scope rules, together with our rule for quantifying according to the left-to-right order in which general terms are encountered, have enabled us to demonstrate a unified algorithm which automatically and correctly interprets (1) the "puzzling" variation in the meaning of "any", depending on context, and (2) the "puzzling" variation in meaning when the active-passive transformation is carried out with general terms instead of names (as in the "No girl is liked by every boy"-"Every boy likes no girl" example).

The fourth area of suggestion (concerning pronouns and variable has been entered only to the extent of incorporating the relative pronoun "who", "whom", and "which", thus permitting the formation of (indefinitely nested) dependent clauses. What Quine has called the "cross-referencing" function of variables is exemplified here to the extent that a relative pronoun triggers pick-up of a variable already generated by a preceding general term.

Other relative pronouns, such as "when", "where" can be handled similarly in simple extensions of English II in which time and place variables are admitted along with time-dependent predicates. In present English II, they can be paraphrased by "time which", "place which", etc.

Indefinite pronouns, such as "something", "everybody", etc. can be included by an extra step in the scan process which would divide such words, making them into general terms, e.g., "some thing", "every body", etc., where, of course, "thing", "body", etc., would be included as monadic predicates in the dictionary. Since this method produces redundant clauses, however, a more direct method should be used.

The successful handling of relative pronouns places us not too far from being able to accept and analyze a large class of prepositional phrases, namely, those which can be regarded as formed by dropping a relative pronoun and its setting, as in "the boy (who was) by the window" Analyzing such forms is, of course, a restoration process, and hence m not be trivial.

It must be admitted, however, that since (third person) personal pronouns are not included in English II, it has less expressive power than English I (and the predicate calculus). That is, there are formulas of logic which have no English II paraphrase, e.g., (AxAy(Rxy ⊃ Sxy). The exact class of formulas which English II can paraphrase has not yet been exactly characterized. It is obviously not limited to the monadic calculus since De Morgan's relational "horse's head" argument can be expressed in it. This sort of program, which may be called that of the articulateness of a given language is little investigated by linguists, apparently because they are committed to the view that any natural language can express any thought. In this, however, they seem not to be considering a synchronic or "snapshot" account of a given language, but rather to be reflecting on the capacity of native speakers to stretch their linguistic resources as needed.

A characterization of the articulateness of English II has not been attempted since its articulateness may be easily increased in many ways (e.g., by permitting it to include English I as a sublanguage). Steps toward including personal pronouns have been taken but will not be described here.

Besides the four areas of logical suggestion, there have been developments of more purely syntactical interest. Our grammar has been based on the observation, already tacitly made by Lewis Carroll in having Alice commended for her eyesight in being able to see nobody on the road, that general terms act syntactically like proper names (perhaps because of a

historically natural syntactical inertia). This treatment of general terms proved to shed considerable light on the variation in the meaning of "is", as between identity and predication, and upon some long puzzling terminology in traditional syllogistic, concerning "distributed terms". The ambiguity in "is" did, however, prevent the inclusion in English II of certain factored expressions occurring in normal English in which general terms and predicates are mixed together.

In constructing the system, we were forced to the recognition of natural syntactic units, the fragments, not heretofore recognized, and to develop a calculus of demand computation to account for the ways in which they can be combined and analyzed. The concept of demand and its calculus, applied here only to English II, may prove a valuable paradigm for the analysis of a broad class of natural and artificial grammars. (The system has a certain resemblance to cancellation grammars of the Ajdukiewicz, Lambek, Bar-Hillel-Gaifman types, but it is more closely tied to the logical import of the expressions analyzed.)

In order to carry out the computation of demands in a way requiring no backtracking, a further grammatical unit was recognized: the apparent fragment, together with a technique for grouping these together into minimal distribution units (the L-atoms). This technique also should be of value for parsing systems of this sort.

The demand concept may also be easily extended to embrace sort distinctions, e.g., permitting "John admires courage" but ruling out "Courage admires John". Such a system was developed for English II

but not incorporated in the actual program since rejection of such categorial errors did not, at this stage, seem an important enough objective to justify the additional storage space and machine time required.

For theoretical linguistics, then, we hope to have made a respectable case for the existence of a promising field of investigation: the logic of natural grammar, and to have provided by example some worthwhile methodological principles and procedures, emphasizing the role of the computer and the value of logical notation as a consistent, broadly articulate, indeed almost inevitable, form of deep-structure representation.

The practical significance of logic-based, English-like languages will, we feel sure, ultimately be great--in law, in computer-assisted teaching and research, and in information retrieval. Its nearer term importance depends on factors hard to predict: the progress of machine inference techniques, the progress of computer software and hardware technology, and the progress of logic itself.

PRINTOUT APPENDIX

```
DICTIONARY          ISAGOGE    =  Y1
DICTIONARY          SINGAPORE  =  Y2
DICTIONARY          TOBU       =  Y3
DICTIONARY          ARNOLD     =  Y4
DICTIONARY          BETTY      =  Y5
DICTIONARY          CHARLES    =  Y6
DICTIONARY          DONALD     =  Y7
DICTIONARY          ESTELLE    =  Y8
DICTIONARY          FRANK      =  Y9
DICTIONARY          GEORGE     =  Y10
DICTIONARY          FIDO       =  Y11
DICTIONARY          ROVER      =  Y12
DICTIONARY          COFFEE     =  Y13
DICTIONARY          BEANS      =  Y14
DICTIONARY          FINLAND    =  Y15
DICTIONARY          BRAZIL     =  Y16
DICTIONARY          GEURGE     =  Y17
DICTIONARY          JONES      =  Y18
DICTIONARY          THEPARTY   =  Y19
DICTIONARY          ARENAA     =  Y20
DICTIONARY          KARL       =  Y21
DICTIONARY          JOHN       =  Y22
```

```
DICTIONARY      PREDICATE   AGENT          =  H14.
                  PLACER SET PI.H14.       =   1//PO.1/*

DICTIONARY      PREDICATE   REPRESENTATIVE =  H15.
                  PLACER SET PI.H15.       =   2//PO.1/PI.5/*.

DICTIONARY      PREDICATE   LIES           =  H16.
                  PLACER SET PI.H16.       =   2//PO.1/PI.6/*

DICTIONARY      PREDICATE   ZREGION        =  H17.
                  PLACER SET PI.H17.       =   1//PO.1/*

DICTIONARY      PREDICATE   READS          =  H18.
                  PLACER SET PI.H18.       =   2//PO.1/PO.2/*

DICTIONARY      PREDICATE   CONTACT        =  H19.
                  PLACER SET PI.H19.       =   2//PO.1/PO.2/*

DICTIONARY      PREDICATE   NATION         =  H20.
                  PLACER SET PI.H20.       =   1//PO.1/*

DICTIONARY      PREDICATE   KSHIPMENT      =  H21.
                  PLACER SET PI.H21.       =   1//PO.1/*

DICTIONARY      PREDICATE   CONTACTS       =  H22.
                  PLACER SET PI.H22.       =   2//PO.1/PO.2/*

DICTIONARY      PREDICATE   SENT           =  H23.
                  PLACER SET PI.H23.       =   3//PO.1/PI.7/PI.3/*

DICTIONARY      PREDICATE   ORIGINATES     =  H24.
                  PLACER SET PI.H24.       =   2//PO.1/PI.6/*

DICTIONARY      PREDICATE   ALERTS         =  H25.
                  PLACER SET PI.H25.       =   2//PO.1/PO.2/*

DICTIONARY      PREDICATE   MSTATION       =  H26.
                  PLACER SET PI.H26.       =   1//PO.1/*

DICTIONARY      PREDICATE   CONTROLS       =  H27.
                  PLACER SET PI.H27.       =   2//PO.1/PO.2/*

DICTIONARY      PREDICATE   FLIGHT         =  H28.
                  PLACER SET PI.H28.       =   1//PO.1/*

DICTIONARY      PREDICATE   PART           =  H29.
                  PLACER SET PI.H29.       =   2//PO.1/PI.5/*

DICTIONARY      PREDICATE   CARBONATOM     =  H30.
                  PLACER SET PI.H30.       =   1//PO.1/*

DICTIONARY      PREDICATE   SINGLYBONDED   =  H31.
                  PLACER SET PI.H31.       =   2//PO.1/PI.3/*

DICTIONARY      PREDICATE   MMOLECULE      =  H32.
                  PLACER SET PI.H32.       =   1//PO.1/*

DICTIONARY      PREDICATE   HYDROXYL       =  H33.
                  PLACER SET PI.H33.       =   1//PO.1/*
```

DICTIONARY	PREDICATE RING	= H34.
	PLACER SET PI.H34.	= 1//P0.1/*
DICTIONARY	PREDICATE MAGNESIUMATOM	= H35.
	PLACER SET PI.H35.	= 1//P0.1/*
DICTIONARY	PREDICATE LAUGHS	= H36.
	PLACER SET PI.H36.	= 1//P0.1/*
DICTIONARY	PREDICATE PLAYS	= H37.
	PLACER SET PI.H37.	= 1//P0.1/*
DICTIONARY	PREDICATE SINGS	= H38.
	PLACER SET PI.H38.	= 1//P0.1/*
DICTIONARY	PREDICATE PUSHES	= H39.
	PLACER SET PI.H39.	= 2//P0.1/P0.2/*
DICTIONARY	PREDICATE HITS	= H40.
	PLACER SET PI.H40.	= 2//P0.1/P0.2/*
DICTIONARY	PREDICATE GIVES	= H41.
	PLACER SET PI.H41.	= 3//P0.1/P0.2/P1.3/*
DICTIONARY	PREDICATE SELLS	= H42.
	PLACER SET PI.H42.	= 3//P0.1/P0.2/P1.3/*
DICTIONARY	PREDICATE WAVES	= H43.
	PLACER SET PI.H43.	= 2//P0.1/P1.3/*
DICTIONARY	PREDICATE SIGNALS	= H44.
	PLACER SET PI.H44.	= 2//P0.1/P1.3/*
DICTIONARY	PREDICATE RECEIVES	= H45.
	PLACER SET PI.H45.	= 3//P0.1/P0.2/P1.2/*
DICTIONARY	PREDICATE TAKES	= H46.
	PLACER SET PI.H46.	= 4//P0.1/P0.2/P1.2/P1.3/*

1. POSITIVE UNIVERSAL

```
INPUT IN ENGLISH II
   EVERY MAN IS MORTAL .

INPUT IN SNOBOL
   01 M51. M1.1 M52.

          THE TRANSFORMED SENTENCE

IN ENGLISH I
   FOR EVERY Z1 IF Z1 MAN THEN Z1 MORTAL

IN PARENTHESIS NOTATION WITH ATOM DISPLAY
   A Z1 ( Z1 MAN =* Z1 MORTAL )

IN PARENTHESIS NOTATION
   A Z1 ( M51. Z1 =* M52. Z1 )

IN POLISH NOTATION
   A Z1 I Z1 M51. Z1 M52.
```

2. NEGATIVE UNIVERSAL (SUPPRESS SNOBOL CODE)

```
INPUT IN ENGLISH II
   NO ELECTRON DECAYS .

          THE TRANSFORMED SENTENCE

IN ENGLISH I
   FOR EVERY Z1 IF Z1 ELECTRON THEN NOT Z1 DECAYS

IN PARENTHESIS NOTATION WITH ATOM DISPLAY
   A Z1 ( Z1 ELECTRON =* N Z1 DECAYS )

IN PARENTHESIS NOTATION
   A Z1 ( M26. Z1 =* N M19. Z1 )

IN POLISH NOTATION
   A Z1 I Z1 M26. N Z1 M19.
```

3. POSITIVE EXISTENTIAL

INPUT IN ENGLISH II
SOME PRIME IS EVEN .

THE TRANSFORMED SENTENCE

IN ENGLISH I
FOR SOME Z1 BOTH Z1 PRIME AND Z1 EVEN

IN PARENTHESIS NOTATION WITH ATOM DISPLAY
E Z1 (Z1 PRIME . Z1 EVEN)

IN PARENTHESIS NOTATION
E Z1 (H60. Z1 . H30. Z1)

IN POLISH NOTATION
S Z1 B Z1 H60. Z1 H30.

4. NEGATIVE EXISTENTIAL

INPUT IN ENGLISH II
SOME SENTENCE IS NOT DECIDABLE .

THE TRANSFORMED SENTENCE

IN ENGLISH I
FOR SOME Z1 BOTH Z1 SENTENCE AND NOT Z1 DECIDABLE

IN PARENTHESIS NOTATION WITH ATOM DISPLAY
E Z1 (Z1 SENTENCE . N Z1 DECIDABLE)

IN PARENTHESIS NOTATION
E Z1 (H70. Z1 . N H20. Z1)

IN POLISH NOTATION
S Z1 B Z1 H70. N Z1 H20.

COMPARISON OF DETERMINERS (SUPPRESS POLISH AND PAREN.)

5. EVERY -

INPUT IN ENGLISH II
IF EVERY GUEST DECLINES THEN THEPARTY FAILS .

THE TRANSFORMED SENTENCE

IN ENGLISH I
IF FOR EVERY Z1 IF Z1 GUEST THEN Z1 DECLINES THEN THEPARTY FAILS

IN PARENTHESIS NOTATION WITH ATOM DISPLAY
(A Z1 (Z1 GUEST =* Z1 DECLINES) =* THEPARTY FAILS)

6. ANY -

INPUT IN ENGLISH II
IF ANY GUEST DECLINES THEN THEPARTY FAILS .

THE TRANSFORMED SENTENCE

IN ENGLISH I
FOR EVERY Z1 IF Z1 GUEST THEN IF Z1 DECLINES THEN THEPARTY FAILS

IN PARENTHESIS NOTATION WITH ATOM DISPLAY
A Z1 (Z1 GUEST =* (Z1 DECLINES =* THEPARTY FAILS))

7. A - (IN THE DIRECT OBJECT)

INPUT IN ENGLISH II
EVERY FRIEND OF A BOOKIE IS A FRIEND OF A GANGSTER .

THE TRANSFORMED SENTENCE

IN ENGLISH I
FOR EVERY Z1 IF FOR SOME Z2 BOTH Z2 BOOKIE AND Z1 FRIEND OF Z2 THEN FOR SOME Z3 BOTH Z3 GANGSTER AND Z1 FRIEND OF Z3

IN PARENTHESIS NOTATION WITH ATOM DISPLAY
A Z1 (E Z2 (Z2 BOOKIE . Z1 FRIEND OF Z2) =* E Z3 (Z3 GANGSTER . Z1 FRIEND OF Z3)
)

8. SOME - (IN THE DIRECT OBJECT)

INPUT IN ENGLISH II
EVERY FRIEND OF A BOOKIE IS A FRIEND OF SOME GANGSTER .

THE TRANSFORMED SENTENCE

IN ENGLISH I
FOR SOME Z3 BOTH Z3 GANGSTER AND FOR EVERY Z1 IF FOR SOME Z2 BOTH Z2 BOOKIE AND Z1 FR IEND OF Z2 THEN Z1 FRIEND OF Z3

IN PARENTHESIS NOTATION WITH ATOM DISPLAY
E Z3 (Z3 GANGSTER . A Z1 (E Z2 (Z2 BOOKIE . Z1 FRIEND OF Z2) =* Z1 FRIEND OF Z3)
)

9A. TWO CASES WITH EVERY AND ANY EQUIVALENT.

INPUT IN ENGLISH II
JOHN READS EVERY BESTSELLER .

THE TRANSFORMED SENTENCE

IN ENGLISH I
FOR EVERY Z1 IF Z1 BESTSELLER THEN JOHN READS Z1

IN PARENTHESIS NOTATION WITH ATOM DISPLAY
A Z1 (Z1 BESTSELLER => JOHN READS Z1)

9B. TWO CASES WITH EVERY AND ANY EQUIVALENT.

INPUT IN ENGLISH II
JOHN READS ANY BESTSELLER .

THE TRANSFORMED SENTENCE

IN ENGLISH I
FOR EVERY Z1 IF Z1 BESTSELLER THEN JOHN READS Z1

IN PARENTHESIS NOTATION WITH ATOM DISPLAY
A Z1 (Z1 BESTSELLER => JOHN READS Z1)

10A. TWO CASES WITH EVERY AND ANY NOT EQUIVALENT.

INPUT IN ENGLISH II
JOHN DOES NOT READ EVERY BESTSELLER .

THE TRANSFORMED SENTENCE

IN ENGLISH I
NOT FOR EVERY Z1 IF Z1 BESTSELLER THEN JOHN READS Z1

IN PARENTHESIS NOTATION WITH ATOM DISPLAY
N A Z1 (Z1 BESTSELLER =* JOHN READS Z1)

10B. TWO CASES WITH EVERY AND ANY NOT EQUIVALENT.

INPUT IN ENGLISH II
JOHN DOES NOT READ ANY BESTSELLER .

THE TRANSFORMED SENTENCE

IN ENGLISH I
FOR EVERY Z1 IF Z1 BESTSELLER THEN NOT JOHN READS Z1

IN PARENTHESIS NOTATION WITH ATOM DISPLAY
A Z1 (Z1 BESTSELLER =* N JOHN READS Z1)

11A SOME AND A WHEN NOT EQUIVALENT.

INPUT IN ENGLISH II
JOHN DOES NOT READ A BESTSELLER .

THE TRANSFORMED SENTENCE

IN ENGLISH I
NOT FOR SOME Z1 BOTH Z1 BESTSELLER AND JOHN READS Z1

IN PARENTHESIS NOTATION WITH ATOM DISPLAY
N E Z1 (Z1 BESTSELLER . JOHN READS Z1)

11B SOME AND A WHEN NOT EQUIVALENT.

INPUT IN ENGLISH II
JOHN DOES NOT READ SOME BESTSELLER .

THE TRANSFORMED SENTENCE

IN ENGLISH I
FOR SOME Z1 BOTH Z1 BESTSELLER AND NOT JOHN READS Z1

IN PARENTHESIS NOTATION WITH ATOM DISPLAY
E Z1 (Z1 BESTSELLER . N JOHN READS Z1)

12. THE SHORT SCOPE DETERMINER NO.

INPUT IN ENGLISH II
JOHN READS NO BESTSELLER .

THE TRANSFORMED SENTENCE

IN ENGLISH I
FOR EVERY Z1 IF Z1 BESTSELLER THEN NOT JOHN READS Z1

IN PARENTHESIS NOTATION WITH ATOM DISPLAY
A Z1 (Z1 BESTSELLER =* N JOHN READS Z1)

THREE EXAMPLES OF RELATIVE PRONOUNS.

13. EXAMPLE 1.

INPUT IN ENGLISH II
EVERY PRIME WHICH IS ODD IS GREATER THAN TWO .

THE TRANSFORMED SENTENCE

IN ENGLISH I
FOR EVERY Z1 IF BOTH Z1 ODD AND Z1 PRIME THEN Z1 GREATER THAN TWO

IN PARENTHESIS NOTATION WITH ATOM DISPLAY
A Z1 ((Z1 ODD .& Z1 PRIME) =* Z1 GREATER THAN TWO)

14. EXAMPLE 2.

INPUT IN ENGLISH II
EVERY CHILD TO WHOM GEORGE GIVES A TOY WHICH BUZZES IS HAPPY .

THE TRANSFORMED SENTENCE

IN ENGLISH I
FOR EVERY Z1 IF BOTH FOR SOME Z2 BOTH BOTH Z2 BUZZ AND Z2 TOY AND GEORGE GIVES Z2
Z1 AND Z1 CHILD OF Z1 THEN Z1 HAPPY

IN PARENTHESIS NOTATION WITH ATOM DISPLAY
A Z1 ((E Z2 ((Z2 BUZZ .& Z2 TOY) .& GEORGE GIVES Z2 TO Z1) .& Z1 CHILD OF Z1) =*
HAPPY)

15. EXAMPLE 3.

INPUT IN ENGLISH II
A WRITER THAN WHOM NO SAGE WAS WISER WROTE ISAGOGE .

THE TRANSFORMED SENTENCE

IN ENGLISH I
FOR SOME Z1 BOTH BOTH FOR EVERY Z2 IF Z2 SAGE THEN NOT Z2 WISER THAN Z1 AND Z1 WRITER AND Z1 WRITES ISAGOGE

IN PARENTHESIS NOTATION WITH ATOM DISPLAY
E Z1 ((A Z2 (Z2 SAGE =⊕ N Z2 WISER THAN Z1) . Z1 WRITER) . Z1 WRITES ISAGOGE)

COMPOUND FORMS

16. COMPOUND SUBJECT.

INPUT IN ENGLISH II
GEORGE OR DONALD PLAYS .

THE TRANSFORMED SENTENCE

IN ENGLISH I
EITHER GEORGE PLAYS OR DONALD PLAYS

IN PARENTHESIS NOTATION WITH ATOM DISPLAY
(GEORGE PLAYS V DONALD PLAYS)

17. COMPOUND DIRECT OBJECT.

INPUT IN ENGLISH II
ARNOLD PUSHES BETTY OR CHARLES .

THE TRANSFORMED SENTENCE

IN ENGLISH I
EITHER ARNOLD PUSHES BETTY OR ARNOLD PUSHES CHARLES

IN PARENTHESIS NOTATION WITH ATOM DISPLAY
(ARNOLD PUSHES BETTY V ARNOLD PUSHES CHARLES)

18. COMPOUND INDIRECT OBJECT.

INPUT IN ENGLISH II
BETTY GIVES ROVER TO CHARLES OR DONALD .

THE TRANSFORMED SENTENCE

IN ENGLISH I
EITHER BETTY GIVES ROVER TO CHARLES OR BETTY GIVES ROVER TO DONALD

IN PARENTHESIS NOTATION WITH ATOM DISPLAY
(BETTY GIVES ROVER TO CHARLES V BETTY GIVES ROVER TO DONALD)

19. COMPOUND FRAGMENT

INPUT IN ENGLISH II
DONALD GIVES ROVER , OR GEORGE GIVES FIDO TO ESTELLE .

THE TRANSFORMED SENTENCE

IN ENGLISH I
EITHER DONALD GIVES ROVER TO ESTELLE OR GEORGE GIVES FIDO TO ESTELLE

IN PARENTHESIS NOTATION WITH ATOM DISPLAY
(DONALD GIVES ROVER TO ESTELLE V GEORGE GIVES FIDO TO ESTELLE)

20. ABBREVIATING POWER OF FACTORING

INPUT IN ENGLISH II
ARNOLD OR BETTY OR CHARLES GIVES OR SELLS FIDO AND ROVER TO DONALD OR ESTELLE OR FRAN K .

THE TRANSFORMED SENTENCE

IN ENGLISH I
EITHER EITHER BOTH EITHER ARNOLD GIVES FIDO TO DONALD OR EITHER ARNOLD GIVES FIDO TO ESTELLE OR ARNOLD GIVES FIDO TO FRANK AND EITHER ARNOLD GIVES ROVER TO DONALD OR EITH ER ARNOLD GIVES ROVER TO ESTELLE OR ARNOLD GIVES ROVER TO FRANK OR EITHER BOTH EITHER BETTY GIVES FIDO TO DONALD OR EITHER BETTY GIVES FIDO TO ESTELLE OR BETTY GIVES FIDO TO FRANK AND EITHER BETTY GIVES ROVER TO DONALD OR EITHER BETTY GIVES ROVER TO ESTEL LE OR BETTY GIVES ROVER TO FRANK OR BOTH EITHER CHARLES GIVES FIDO TO DONALD OR EITHE R CHARLES GIVES FIDO TO ESTELLE OR CHARLES GIVES FIDO TO FRANK AND EITHER CHARLES GIV ES ROVER TO DONALD OR EITHER CHARLES GIVES ROVER TO ESTELLE OR CHARLES GIVES ROVER TO FRANK OR EITHER BOTH EITHER ARNOLD SELLS FIDO TO DONALD OR EITHER ARNOLD SELLS FIDO TO ESTELLE OR ARNOLD SELLS FIDO TO FRANK AND EITHER ARNOLD SELLS ROVER TO DONALD OR E ITHER ARNOLD SELLS ROVER TO ESTELLE OR ARNOLD SELLS ROVER TO FRANK OR EITHER BOTH EIT HER BETTY SELLS FIDO TO DONALD OR EITHER BETTY SELLS FIDO TO ESTELLE OR BETTY SELLS F IDO TO FRANK AND EITHER BETTY SELLS ROVER TO DONALD OR EITHER BETTY SELLS ROVER TO ES TELLE OR BETTY SELLS ROVER TO FRANK OR BOTH EITHER CHARLES SELLS FIDO TO DONALD OR EI THER CHARLES SELLS FIDO TO ESTELLE OR CHARLES SELLS FIDO TO FRANK AND EITHER CHARLES SELLS ROVER TO DONALD OR EITHER CHARLES SELLS ROVER TO ESTELLE OR CHARLES SELLS ROVER TO FRANK

IN PARENTHESIS NOTATION WITH ATOM DISPLAY
((((ARNOLD GIVES FIDO TO DONALD V (ARNOLD GIVES FIDO TO ESTELLE V ARNOLD GIVES FIDO TO FRANK)) . (ARNOLD GIVES ROVER TO DONALD V (ARNOLD GIVES ROVER TO ESTELLE V ARN OLD GIVES ROVER TO FRANK))) V (((BETTY GIVES FIDO TO DONALD V (BETTY GIVES FIDO TO ESTELLE V BETTY GIVES FIDO TO FRANK)) . (BETTY GIVES ROVER TO DONALD V (BETTY GIV ES ROVER TO ESTELLE V BETTY GIVES ROVER TO FRANK))) V ((CHARLES GIVES FIDO TO DONAL D V (CHARLES GIVES FIDO TO ESTELLE V CHARLES GIVES FIDO TO FRANK)) . (CHARLES GIVE S ROVER TO DONALD V (CHARLES GIVES ROVER TO ESTELLE V CHARLES GIVES ROVER TO FRANK))))) V (((ARNOLD SELLS FIDO TO DONALD V (ARNOLD SELLS FIDO TO ESTELLE V ARNOLD SELL S FIDO TO FRANK)) . (ARNOLD SELLS ROVER TO DONALD V (ARNOLD SELLS ROVER TO ESTELLE V ARNOLD SELLS ROVER TO FRANK))) V (((BETTY SELLS FIDO TO DONALD V (BETTY SELLS F IDO TO ESTELLE V BETTY SELLS FIDO TO FRANK)) . (BETTY SELLS ROVER TO DONALD V (BET TY SELLS ROVER TO ESTELLE V BETTY SELLS ROVER TO FRANK))) V ((CHARLES SELLS FIDO TO DONALD V (CHARLES SELLS FIDO TO ESTELLE V CHARLES SELLS FIDO TO FRANK)) . (CHARLE S SELLS ROVER TO DONALD V (CHARLES SELLS ROVER TO ESTELLE V CHARLES SELLS ROVER TO F RANK))))))

21. SECOND ABBREVIATING EXAMPLE.

INPUT IN ENGLISH II
IF ARNOLD AND BETTY AND CHARLES GIVE OR SELL ROVER OR FIDO TO FRANK OR GEORGE , THEN FRANK OR GEORGE SIGNALS OR WAVES TO DONALD OR ESTELLE .

THE TRANSFORMED SENTENCE

IN ENGLISH I
IF EITHER BOTH EITHER EITHER ARNOLD GIVES ROVER TO FRANK OR ARNOLD GIVES ROVER TO GEORGE OR EITHER ARNOLD GIVES FIDO TO FRANK OR ARNOLD GIVES FIDO TO GEORGE AND BOTH EITHER EITHER BETTY GIVES ROVER TO FRANK OR BETTY GIVES ROVER TO GEORGE OR EITHER BETTY GIVES FIDO TO FRANK OR BETTY GIVES FIDO TO GEORGE AND EITHER EITHER CHARLES GIVES ROVER TO FRANK OR CHARLES GIVES ROVER TO GEORGE OR EITHER CHARLES GIVES FIDO TO FRANK OR CHARLES GIVES FIDO TO GEORGE OR BOTH EITHER EITHER ARNOLD SELLS ROVER TO FRANK OR ARNOLD SELLS ROVER TO GEORGE OR EITHER ARNOLD SELLS FIDO TO FRANK OR ARNOLD SELLS FIDO TO GEORGE AND BOTH EITHER EITHER BETTY SELLS ROVER TO FRANK OR BETTY SELLS ROVER TO GEORGE OR EITHER BETTY SELLS FIDO TO FRANK OR BETTY SELLS FIDO TO GEORGE AND EITHER EITHER CHARLES SELLS ROVER TO FRANK OR CHARLES SELLS ROVER TO GEORGE OR EITHER CHARLES SELLS FIDO TO FRANK OR CHARLES SELLS FIDO TO GEORGE THEN EITHER EITHER EITHER FRANK SIGNALS TO DONALD OR FRANK SIGNALS TO ESTELLE OR EITHER GEORGE SIGNALS TO DONALD OR GEORGE SIGNALS TO ESTELLE OR EITHER EITHER FRANK WAVES TO DONALD OR FRANK WAVES TO ESTELLE OR EITHER GEORGE WAVES TO DONALD OR GEORGE WAVES TO ESTELLE

IN PARENTHESIS NOTATION WITH ATOM DISPLAY
(((((ARNOLD GIVES ROVER TO FRANK V ARNOLD GIVES ROVER TO GEORGE) V (ARNOLD GIVES FIDO TO FRANK V ARNOLD GIVES FIDO TO GEORGE)) . (((BETTY GIVES ROVER TO FRANK V BETTY GIVES ROVER TO GEORGE) V (BETTY GIVES FIDO TO FRANK V BETTY GIVES FIDO TO GEORGE)) . ((CHARLES GIVES ROVER TO FRANK V CHARLES GIVES ROVER TO GEORGE) V (CHARLES GIVES FIDO TO FRANK V CHARLES GIVES FIDO TO GEORGE)))) V (((ARNOLD SELLS ROVER TO FRANK V ARNOLD SELLS ROVER TO GEORGE) V (ARNOLD SELLS FIDO TO FRANK V ARNOLD SELLS FIDO TO GEORGE)) . (((BETTY SELLS ROVER TO FRANK V BETTY SELLS ROVER TO GEORGE) V (BETTY SELLS FIDO TO FRANK V BETTY SELLS FIDO TO GEORGE)) . ((CHARLES SELLS ROVER TO FRANK V CHARLES SELLS ROVER TO GEORGE) V (CHARLES SELLS FIDO TO FRANK V CHARLES SELLS FIDO TO GEORGE))))) =* (((FRANK SIGNALS TO DONALD V FRANK SIGNALS TO ESTELLE) V (GEORGE SIGNALS TO DONALD V GEORGE SIGNALS TO ESTELLE)) V ((FRANK WAVES TO DONALD V FRANK WAVES TO ESTELLE) V (GEORGE WAVES TO DONALD V GEORGE WAVES TO ESTELLE))))

22. GENERAL TERMS IN COMPOUND FRAGMENTS.

INPUT IN ENGLISH II
EVERY SENTENCE IS TRUE OR FALSE .

THE TRANSFORMED SENTENCE

IN ENGLISH I
FOR EVERY Z1 IF Z1 SENTENCE THEN EITHER Z1 TRUE OR Z1 FALSE

IN PARENTHESIS NOTATION WITH ATOM DISPLAY
A Z1 (Z1 SENTENCE =* (Z1 TRUE V Z1 FALSE))

23. GENERAL TERMS IN COMPOUND FRAGMENTS.

INPUT IN ENGLISH II
GEORGE PUSHED OR PULLED EVERY LEVER .

THE TRANSFORMED SENTENCE

IN ENGLISH I
FOR EVERY Z1 IF Z1 LEVER THEN EITHER GEORGE PUSHES Z1 OR GEORGE PULL Z1

IN PARENTHESIS NOTATION WITH ATOM DISPLAY
A Z1 (Z1 LEVER =* (GEORGE PUSHES Z1 V GEORGE PULL Z1))

24. GENERAL TERMS IN COMPOUND FRAGMENTS.

INPUT IN ENGLISH II
GEORGE OR AN ASSISTANT PUSHED A BUTTON , OR PULLED A LEVER .

THE TRANSFORMED SENTENCE

IN ENGLISH I
FOR SOME Z1 BOTH Z1 ASSISTANT AND FOR SOME Z2 BOTH Z2 BUTTON AND FOR SOME Z3 BOTH Z3 LEVER AND EITHER EITHER GEORGE PUSHES Z2 OR Z1 PUSHES Z2 OR EITHER GEORGE PULL Z3 OR Z1 PULL Z3

IN PARENTHESIS NOTATION WITH ATOM DISPLAY
E Z1 (Z1 ASSISTANT . E Z2 (Z2 BUTTON . E Z3 (Z3 LEVER . ((GEORGE PUSHES Z2 V Z1 PUSHES Z2) V (GEORGE PULL Z3 V Z1 PULL Z3))))

25. DETERMINER, SIMPLE PREDICATE, RELATIVE CLAUSE.

INPUT IN ENGLISH II
EVERY MAN OR EVERY WOMAN WHO APPLIES RECEIVES A COUPON FROM JONES .

THE TRANSFORMED SENTENCE

IN ENGLISH I
FOR EVERY Z1 IF Z1 MAN THEN FOR EVERY Z2 IF BOTH Z2 APPLIES AND Z2 WOMAN THEN FOR SOM
E Z3 BOTH Z3 COUPON AND EITHER Z1 RECEIVES Z3 FROM JONES OR Z2 RECEIVES Z3 FROM JONES

IN PARENTHESIS NOTATION WITH ATOM DISPLAY
A Z1 (Z1 MAN =* A Z2 ((Z2 APPLIES . Z2 WOMAN) =* E Z3 (Z3 COUPON . (Z1 RECEIVES
Z3 FROM JONES V Z2 RECEIVES Z3 FROM JONES))))

26. DETERMINER, COMPOUND FRAGMENT, RELATIVE CLAUSE.

INPUT IN ENGLISH II
EVERY MAN OR WOMAN WHO APPLIES RECEIVES A COUPON FROM JONES .

THE TRANSFORMED SENTENCE

IN ENGLISH I
FOR EVERY Z1 IF BOTH Z1 APPLIES AND EITHER Z1 MAN OR Z1 WOMAN THEN FOR SOME Z2 BOTH Z
2 COUPON AND Z1 RECEIVES Z2 FROM JONES

IN PARENTHESIS NOTATION WITH ATOM DISPLAY
A Z1 ((Z1 APPLIES . (Z1 MAN V Z1 WOMAN)) =* E Z2 (Z2 COUPON . Z1 RECEIVES Z2 FROM
JONES))

27. COMPOUND RELATIVE CLAUSE.

INPUT IN ENGLISH II
EVERY ENTREPOT WHICH IS NOT STRIKEBOUND AND FROM WHICH COFFEE IS EXPORTED TO FINLAND AND TO WHICH BEANS ARE SHIPPED FROM BRAZIL IS WATCHED BY A DETECTIVE .

THE TRANSFORMED SENTENCE

IN ENGLISH I
FOR EVERY Z1 IF BOTH BOTH NOT Z1 STRIKEBOUND AND BOTH COFFEE EXPORT FROM Z1 TO FINLAND AND BEANS SHIP FROM BRAZIL TO Z1 AND Z1 ENTREPOT THEN FOR SOME Z2 BOTH Z2 DETECTIVE AND Z1 WATCH BY Z2

IN PARENTHESIS NOTATION WITH ATOM DISPLAY
A Z1 (((N Z1 STRIKEBOUND . (COFFEE EXPORT FROM Z1 TO FINLAND . BEANS SHIP FROM BRAZIL TO Z1)) . Z1 ENTREPOT) =* E Z2 (Z2 DETECTIVE . Z1 WATCH BY Z2))

28. COMPOUND RELATIVE CLAUSE.

INPUT IN ENGLISH II
EVERY CARBONATOM WHICH IS PART OF A MMOLECULE AND WHICH IS SINGLYBONDED TO A HYDROXYL IS PART OF A RING OF WHICH A MAGNESIUMATOM IS PART .

THE TRANSFORMED SENTENCE

IN ENGLISH I
FOR EVERY Z1 IF BOTH BOTH FOR SOME Z2 BOTH Z2 MMOLECULE AND Z1 PART OF Z2 AND FOR SOME Z3 BOTH Z3 HYDROXYL AND Z1 SINGLYBONDED TO Z3 AND Z1 CARBONATOM THEN FOR SOME Z4 BOTH BOTH FOR SOME Z5 BOTH Z5 MAGNESIUMATOM AND Z5 PART OF Z4 AND Z4 RING AND Z1 PART OF Z4

IN PARENTHESIS NOTATION WITH ATOM DISPLAY
A Z1 (((E Z2 (Z2 MMOLECULE . Z1 PART OF Z2) . E Z3 (Z3 HYDROXYL . Z1 SINGLYBONDED TO Z3)) . Z1 CARBONATOM) =* E Z4 ((E Z5 (Z5 MAGNESIUMATOM . Z5 PART OF Z4) . Z4 RING) . Z1 PART OF Z4))

USER'S APPENDIX

Further Remarks on Table 1

In Table I, the entries from "Not" to "Then" are common to English I and II. The symbolic forms shown belong either to parenthesis or Polish notation in an obvious way.

"For every" and "For some" are peculiar to English I and their symbolizations vary in that "S" is used for the existential quantifiers in the Polish output so as not to conflict with the 'E' used for 'either', while parenthesis notation output gives "E in its more normal usage as existential quantifier. "Is" and "was" are given the symbolic representation "is". They, the determiners, and relative pronouns shown are peculiar to English II.

Formats for Dictionary Cards

In entering non-logical words on dictionary cards as shown in Figures 1, 2, 3, in the text, the following points should be observed.

1. Colum 1 must be blank.

2. Proper names may be listed several to a card, each followed by a comma. Eighty columns are read but each card must have a terminal slash. The last name card of a sequence of name cards must have a period before the slash. All such cards may have a period before the slash.

3. Predicate cards permit only one predicate per card

4. The words in parentheses are words which the read-in is to recognize as synonymous variants. The English I output gives only the main term. This item may be omitted.

5. The category designations permitted are "verb", "adj", "noun"; they may be omitted if rejection of ill-formed strings is not required.

6. The number preceding the period is the degree of the predicate.

7. Any placers appearing in the right hand "placer pattern" of a predicate card are incorporated by the program in a "placer dictionary" as a "P" followed by an assigned numeral; no special cards for placers are required.

8. Prepositions and passive voice verbs used as predicates (e.g. "A is inside of B', "A is hit by B") are regarded as adjectives (because of their similar relation to the copula).

9. The order and punctuation of the sample cards should be carefully followed. Where spaces are shown, they may be of any positive length.

As name and predicate cards are read by the program, it sets up English SNOBOL, SNOBOL English dictionaries. Names are represented as "Y1", "Y2", "Y3", etc.

Predicates are represented by "H" followed by a numeral followed by a period followed by a second numeral, e.g. "H15.2". The first numeral is assigned in order of read-in. The second numeral corresponds to the category, as follows:

Verb 1

Adj 2

Noun 3

If no category is specified, the last numeral is omitted, e.g., "H15."

Card Formats in Data Input

1. It is often convenient to preface a given example by a comment. This can be done by placing a comment card before the example. Comment cards must not have a blank column zero. Any numeral, character or symbol will do. See asterisk example in Figure 4. A comment may continue for several cards provided all have non-blank zero columns. Any dictionary or data cards accidentally punched in column 1 are printed, as is, as comments and not proces sed.
2. Input test sentences may begin anywhere except in column 1. Commas may immediately follow a word, or be spaced. All input sentences must end in a period (spaced or not). No other period may occur in a sentence. Input sentences may run for several cards. All 80 columns are read but each card must end in '/'.

General Format Remarks

All input cards (dictionary, comment, sentence) must end in '/'. When a card lacking a slash is encountered the run is ended. Thus a blank card is placed at the end of a set of examples to lead to the normal SNOBOL exit.

Control of Output Variations

At the end of the program deck just before the END card, following a card labeled FRMT= (FORMAT), come a series of three-card sequences each of which causes print-out of the transformed sentence in a given format. E. G.

(1) SYSPOT = 'IN ENGLISH I'

(2) SYSPNT (ENGI(TSTRING= PNM)) /F(FOUT2)

(3) SYSPOT = BLNK

causes printout in English I. Others give printout in parenthesis notation, etc., as seen in the printout samples. Any combination or order of formats can be chosen for a given run by shifting, removing or inserting these three-card sequences as wholes.

Notes

1. Developed by Farber, Griswold, and Polonsky (1964). (Last names with parenthesized dates refer to publications in the list of References.) The more powerful SNOBOL3 has not appeared in publicly available form at this writing.
2. See Bohnert items and Backer (1965) in References
3. Jespersen (1927)
4. Chomsky (1961) p. 2
5. Postal (1964), Chomsky (1965)
6. Fodor and Kate (1964)
7. Quine (1960a and c)
8. Carnap (1937). The famous quote from the Introduction is reproduced once again:

   "The method of (logical) syntax... will not only prove useful in the logical analysis of scientific theories - it will also help in the logical analysis of the word languages... The direct analysis of these, which has been prevalent hitherto, must inevitably fail, just as a physicist would be frustrated were he from the outset to attempt to relate his laws to natural things - trees, stones, and so on. In the beginning, the physicist relates his laws to the simplest of constructed forms; to a thin straight lever, to a simple pendulum, to punctiform masses, etc. Then, with the help of the laws relating to these constructed forms, he is later in a position to analyze into

suitable elements the complicated behavior of real bodies, and thus to control them. One more comparison: the complicated configurations of mountain chains, rivers, frontiers, and the like, are most easily represented and investigated by the help of geographical coordinates - or, in other words, by constructed lines not given in nature. In the same way, the syntactical properties of a particular word-language, such as English, or of a particular sub-language of a word-language, are best represented and investigated by comparison with a constructed language which serves as a system of reference."

While this quote has received shrewd criticism from Chomsky (1955), the basic point that analysis of grammatical forms may profit from idealization seems sound.

9. Reichenbach (1947), Chapter 7.

10. Williams (1956), Bohnert (1961), Cooper (1963).

11. Floyd (1963)

12. Cook (1965)

13. Originating, apparently, in Reed, A. and Kellogg, B., Higher Lessons in English, N. Y., Clark and Maynard, 1888. We are indebted to Professor D. W. Emery, University of Washington, for this historical reference.

References

Backer, P., and Bohnert, H. (1965) "Computer Analysis of Compound Expressions. IBM (Unpublished)

Bar-Hille, Y. and Carnap, R. (1952) An Outline of a Theory of Semantic Information, Technical Report No. 247, Research Laboratory of Electronics, Massachusetts Institute of Technology

Bar-Hillel, Y. (1954). Logical syntax and semantics. Language, Vol. 30, pp. 230-237

Bar-Hille, Y., M. Perles and E. Shamir (1961). "On formal properties of simple phrase structure grammars." First appeared in Zeitschrift fur Phonetic, Sprachwissenschaft und Kommunikations forschung, vol. 14, pp. 143-172. Reprinted in Yehoshua Bar-Hillel (1964). Language and Information.

Bar-Hillel, Y., Kasher, A. Shamir, E. (1963) Measures of Syntactic Complexity Technical Report No. 13 - The Hebrew University of Jerusalem

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

Bohnert, H. (1961). "Project LOGOS." IBM (unpublished).

Bohnert, H. (1962a). "Formation Rules for LAMB." IBM (unpublished.

Bohnert, H. (1962b). "An Englishlike extension of an applied predicate calculus." IBM (unpublished).

Bohnert, H. (1962c). "The logic of the relative pronoun 'that'." IBM (unpublished).

Bohnert, H. (1962d), "A System of Grouping for English-Like Languages", IBM N8 122.

Bohnert, H. (1963). "Englishlike systems of mathematical logic for content retrieval." In Proceedings of American Documentation Institute, Annual Meeting, Vol. 2, pp. 155-156.

Carnap, Rudolf (1937). The Logical Syntax of Language. New York: Harcourt, Brace and Company.

Carnap, Rudolf (1942). Introduction to Semantics. Cambridge: Harvard University Press.

Carnap, Rudolf (1947). Meaning and Necessity: a Study in Semantics and Modal Logic. Chicago.

Carnap, Rudolf, and Yehoshua Bar-Hillel (1952). "An outline of a theory of semantic information" Technical Report No. 247, Research Laboratory of Electronics. Cambridge: M. I. T.

Carnap, Rudolf (1958). Introduction to Symbolic Logic and its Applications. New York: Dover.

Chomsky, N. (1955). "Logical syntax and semantics-their linguistic relevance." Language, Vol. 3, pp. 36-45.

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

Chomsky, N. (1961). "Some methodological remarks on generative grammar." Word, Vol. 17, pp. 219-239.

Chomsky, N., and M. P. Schutzenberger (1963). "The Algebraic Theory of Context-Free Languages", P. 118, Computer Programming and Formal Systems, Edited by P. Braffort and D. Hurschberg, N. Holland Publishing Co.

Chomsky, N. (1965). Aspects of the Theory of Syntax. Cambridge: The M. I. T. Press

Church, Alonzo (1956). Review of Wundheiler, L. and A. (1955), Journal of Symbolic Logic, Vol. 21, pp. 312-313.

Cohen, L. J. (1963). The Diversity of Meaning. New York: Herder and Herder.

Cook, S. (1965). "Algebraic Techniques and the Mechanization of Number Theory" The Rand Corporation - RM-4319 - PR

Cooper, William S. (1963). "Fact retrieval and deductive question-answering information retrieval systems." Journal of the Association for Computing Machinery, Vol. 11, No. 2, April, 1964

Darlington, J. L. (1965). "Machine Methods for proving logical arguments expressed in English." Mechanical Translation, Vol. 8, June-October, 1965, pp. 41-67.

Farber, Griswold and Polonsky (1964). "SNOBOL, a string manipulation language." ACM Journal, vol. 11, no. 1.

Floyd, R. W. (1963) "Syntactic Analysis and Operation Precedence" J. Assoc. for Computing Machines, 10, p. 316-333

Fodor, J. (1964), review of Cohen, L. J. (1963), Journal of Philosophy, Vol. LXI, No. 11: May 21

Fodor, J. A. and J. J. Katz (eds.) (1964). The Structure of Language: Readings in the Philosophy of Language. Englewood Cliffs, N. J.: Prentice-Hall.

Geach, P. T. (1965). "Complex Terms Again." The Journal of Philosophy, Vol. LXII, No. 23: December 2, 1965.

Gilbert, Philip (1966). "On the syntax of algorithmic languages". Journal of the Association for Computing Machinery, vol. 13, no. 1.

Harris, Z. S. (1951). Methods of Structural Linguistics. Chicago: University of Chicago Press.

Gorn, Saul (1962). "The treatment of Ambiguity and paradox in mechanical languages." Reprinted from "Recursive function theory." Proceedings of Symposia in Pure Mathematics, Vol. 5. American Mathematical Society.

Jespersen, O. (1927) Analytic Syntax

Kirsch, R. (1963). "The application of automata theory to problems in information retrieval." National Bureau of Standards Report 7882.

Knuth, Donald E. (1965) "On the Translation of Languages from Left to Right", Information and Control, 8, P. 607-639

Postal, P. M. (1964). "Underlying and superficial linguistic structure." Harvard Educational Review, Vol. 34, pp. 246-266.

Quine, W. V. (1940), Mathematical Logic. Harvard Press.

Quine, W. V. (1950). Methods of Logic. New York: Henry Holt and Company.

Quine, W. V. (1960a). "Logic as a source of syntactical insights", Structure of Language and Its Mathematical Aspects, ed. by R. Jakobson, American Mathematical Society, 1961.

Quine, W. V. (1960b). "Variables explained away." Proceedings of the American Philosophical Society, Vol. 104, No. 3, June, 196

Quine, W. V. (1960c). Word and Object. Cambridge, Mass.: M. I. T. Press, and New York: Wiley.

Reichenbach, H. (1947). Elements of Symbolic Logic, Chapter 7. New York: Macmillan.

Rosenbloom, P. C. (1950) The Elements of Mathematical Logic. New Yorl

Sillars, Walter (1963). "An algorithm for representing English sentences in a formal language." National Bureau of Standards Report 7884. U. S. Department of Commerce.

Tarski, A. (1956). Logic, Semantics, Metamathematics." Oxford.

Tarski, A. (1959). "What is elementary geometry?" The Axiomatic Method, eds. L. Henkin, P. Suppes, A. Tarski.

Wang, Hao (1960). "Toward mechanical mathematics." IBM Journal of Research and Development, Vol. 10, pp. 2-22.

Williams, T. (1956). "Translating from ordinary discourse into symbolic logic." ACF Industries.

Wundheiler, L. and A. (1955). "Some logical concepts for syntax." Machine Translation of Languages, Fourteen Essays, ed. by W. Locke and D. Booth, pp. 194-207, New York: John Wiley and Sons.

Unclassified

Security Classification

DOCUMENT CONTROL DATA - R&D

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

1. ORIGINATING ACTIVITY *(Corporate author)*

IBM Corp
T. J. Watson Research Center

2a. REPORT SECURITY CLASSIFICATION

Unclassified

2b GROUP

3. REPORT TITLE

Automatic English to Logic Translation in a Simplified Model
A Study in the Logic of Grammar

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

Logical - Linguistic study; Final report; 1961 - 1966

5. AUTHOR(S) *(Last name, first name, initial)*

Bohnert, Herbert G. (Principal Investigator)
Backer, Paul O.

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
March 1966	112	52

8a. CONTRACT OR GRANT NO.	9a. ORIGINATOR'S REPORT NUMBER(S)
AF 49(638) 1198	
b. PROJECT NO.	
c. Project Task No. 9769-06	9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)*
d.	AFOSR 66-1727

10. AVAILABILITY/LIMITATION NOTICES

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
	Air Force Office of Scientific Research Washington 25, DC

13. ABSTRACT

The study has been concerned with the relation between natural language and symbolic logic. This final report describes the latest in a series of Englishlike languages translatable into forms of first order predicate calculus notation. The recognition and translation program, written in the string manipulating SNOBOL 3 system, is also described. (U)

DD FORM 1 JAN 64 1473

Unclassified

Security Classification

Unclassified

Security Classification

14. KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Logic						
Grammar						
Mechanical Translation						
Predication						
Compounding						
Quantification						

INSTRUCTIONS

1. ORIGINATING ACTIVITY: Enter the name and address of the contractor, subcontractor, grantee, Department of Defense activity or other organization *(corporate author)* issuing the report.

2a. REPORT SECURITY CLASSIFICATION: Enter the overall security classification of the report. Indicate whether "Restricted Data" is included. Marking is to be in accordance with appropriate security regulations.

2b. GROUP: Automatic downgrading is specified in DoD Directive 5200.10 and Armed Forces Industrial Manual. Enter the group number. Also, when applicable, show that optional markings have been used for Group 3 and Group 4 as authorized.

3. REPORT TITLE: Enter the complete report title in all capital letters. Titles in all cases should be unclassified. If a meaningful title cannot be selected without classification, show title classification in all capitals in parenthesis immediately following the title.

4. DESCRIPTIVE NOTES: If appropriate, enter the type of report, e.g., interim, progress, summary, annual, or final. Give the inclusive dates when a specific reporting period is covered.

5. AUTHOR(S): Enter the name(s) of author(s) as shown on or in the report. Enter last name, first name, middle initial. If military, show rank and branch of service. The name of the principal author is an absolute minimum requirement.

6. REPORT DATE: Enter the date of the report as day, month, year, or month, year. If more than one date appears on the report, use date of publication.

7a. TOTAL NUMBER OF PAGES: The total page count should follow normal pagination procedures, i.e., enter the number of pages containing information.

7b. NUMBER OF REFERENCES: Enter the total number of references cited in the report.

8a. CONTRACT OR GRANT NUMBER: If appropriate, enter the applicable number of the contract or grant under which the report was written.

8b, 8c, & 8d. PROJECT NUMBER: Enter the appropriate military department identification, such as project number, subproject number, system numbers, task number, etc.

9a. ORIGINATOR'S REPORT NUMBER(S): Enter the official report number by which the document will be identified and controlled by the originating activity. This number must be unique to this report.

9b. OTHER REPORT NUMBER(S): If the report has been assigned any other report numbers *(either by the originator or by the sponsor)*, also enter this number(s).

10. AVAILABILITY/LIMITATION NOTICES: Enter any limitations on further dissemination of the report, other than those imposed by security classification, using standard statements such as:

(1) "Qualified requesters may obtain copies of this report from DDC."

(2) "Foreign announcement and dissemination of this report by DDC is not authorized."

(3) "U. S. Government agencies may obtain copies of this report directly from DDC. Other qualified DDC users shall request through _____."

(4) "U. S. military agencies may obtain copies of this report directly from DDC. Other qualified users shall request through _____."

(5) "All distribution of this report is controlled. Qualified DDC users shall request through _____."

If the report has been furnished to the Office of Technical Services, Department of Commerce, for sale to the public, indicate this fact and enter the price, if known.

11. SUPPLEMENTARY NOTES: Use for additional explanatory notes.

12. SPONSORING MILITARY ACTIVITY: Enter the name of the departmental project office or laboratory sponsoring *(paying for)* the research and development. Include address.

13. ABSTRACT: Enter an abstract giving a brief and factual summary of the document indicative of the report, even though it may also appear elsewhere in the body of the technical report. If additional space is required, a continuation sheet shall be attached.

It is highly desirable that the abstract of classified reports be unclassified. Each paragraph of the abstract shall end with an indication of the military security classification of the information in the paragraph, represented as *(TS), (S), (C), or (U)*.

There is no limitation on the length of the abstract. However, the suggested length is from 150 to 225 words.

14. KEY WORDS: Key words are technically meaningful terms or short phrases that characterize a report and may be used as index entries for cataloging the report. Key words must be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location, may be used as key words but will be followed by an indication of technical context. The assignment of links, rules, and weights is optional.

GPO 886-551