There are a number of reasons why I chose the context of algebra story problems in which to embed an English language question-answering system. First, we know a good type of data structure in which to store information needed to answer questions in this context, namely, algebraic equations. There exist well known algorithms for deducing information implicit in the equations, that is, values for particular variables which satisfy the set of equations.

In addition, I felt that there was a manageable subset of English in which many types of algebra story problems were expressible. A large number of these story problems are available in first year high school text books, and I have transcribed some of them into STUDENT's input English. Since this question-answering task is one performed by humans, and since the entire process from input to solution of the equations was programmed, we can obtain a measure of comparison between the performance of STUDENT and of a human on the same problems. In fact, this program on an IBM 7094, answers most questions that it can handle as fast as or faster than humans trying the same problem. In judging this comparison, one should remember the base speed of the IBM 7094, which can perform over one hundred thousand additions per second.

II. SEMANTIC GENERATION AND ANALYSIS OF DISCOURSE

The purpose of this section is to put the techniques of analysis embedded in the STUDENT program into a wider context, and indicate how they would fit into a more general language processing system. We will describe a theory of semantic generation and analysis of discourse. STUDENT can then be considered a first approximation to a computer implementation of the analytic portion of the theory, with certain restrictions on the interpretation of a discourse to be analyzed. It will be evident from the theory why analysis is so greatly simplified by the imposed restrictions.

A. *Language as Communication*

Language is an encoding used for communication between a speaker and a listener (or writer and reader). To transmit an "idea", the speaker must first encode it in a message, as a string in the transmission language. In order to understand this message, a listener must decode it and extract its meaning. The coding of a particular message, M, is a function of both its global context and local context. The global context of a message is the background knowledge of the speaker and the listener, including some knowledge of possible universes of discourse, and codings for some simple ideas.

The local context of a message, M, is the set of messages temporally adjacent to M. M may refer back to earlier messages. M may even be just a modification of a previous message, and only understandable in this context. For example, consider the second sentence of the following discourse: "How many chaplains are in the U.S. Army? How many are in the Navy?"

In order for communication to take place, the information map of both the listener and the speaker must be approximately the same, at least for the universe of discourse; also the decoding process of the listener must be an approximate inverse of the encoding process of the speaker. Education in language is, in large part, an attempt to force the language processors of different people into a uniform mold to facilitate successful communication. We are not proposing that identity in detail is achieved, but as Quine [9] put it:

> "Different persons growing up in the same language are like different bushes trimmed and trained to take the shape of identical elephants. The anatomical details of twigs and branches will fulfill the elephantine form differently from bush to bush, but the overall outward results are alike."

As a speaker transmits successive messages concerning some portion of his information map, the listener who understands the messages constructs a model of a "situation". The relation between the listener's model and the speaker's information map is that from each can be extracted the transmitted information relevant to the universe of discourse, including information deducible from the entire set of messages. The internal structure of the listener's model need bear no resemblance to that of the speaker, and may in general contain far less detail.

B. Definition of Coherent Discourse

The theory of language generation and analysis which we shall describe below is designed to handle what we call *coherent discourse*. A discourse is a sequence of sentences, and the meaning of a discourse for a listener is the model of a situation he derives from this discourse. Determination of the meaning of each sentence of the discourse may sometimes involve knowledge of the meanings of other sentences of the discourse. A discourse is coherent if it has a complete and consistent interpretation within the model of the situation being built by the listener. A listener understands the discourse if his model of the situation is isomorphic to the speaker's model.

A listener's ability to build a model of a situation from a discourse is dependent on information available to him from his general store of knowledge. Therefore it is quite possible for a discourse to seem coherent to one listener and not another. A writer, reading his own writing, may feel that he has generated a coherent sequence of sentences, which, in fact, is incoherent to all other readers. This is, unfortunately, not a rare occurrence in the scientific literature. Conversely, a listener who is a psychiatrist, for example, may find coherence in a sequence of remarks which a patient thinks are entirely unrelated.

The STUDENT system utilizes an expandable store of general knowledge to build a model of a situation described in a member of a limited class of discourses. The form of the model of a situation built by STUDENT will be discussed in detail below.

C. *The Use of Kernel Sentences in Our Theory*

A basic postulate of our theory of language analysis is that a listener understands a discourse by transforming it into an equivalent (in meaning) sequence of simpler kernel sentences. A *kernel sentence* is one which the listener can understand directly; that is, one for which he knows a transformation into his information store. Conversely, a speaker generates a set of kernel sentences from his information map, and utilizes a sequence of transformations on this set to yield his spoken discourse. This set of kernel sentences is not invariant from person to person, and even varies for a single individual as he learns.

Although we are not proposing our theory as a basis for a psychological model, it has been useful, to avoid circumlocutions, to describe the theory in terms of the properties and actions of a hypothetical speaker and listener. All statements about speakers and listeners should be interpreted as referring to computer programs which respectively, generate and analyze coherent discourse.

D. *Generation of Coherent Discourse*

1) The Speaker's Model of the World. We assume that a speaker has some model of the world in his information store. We shall not be concerned here with how this model was built, or its exact form. Different forms for the model will be useful for different language tasks, but they must all have the properties described below.

The basic components of the model are a set of objects, $\{O_i\}$, a set of functions $\{F_i^n\}$, a set of relations $\{R_i^n\}$, a set of propositions $\{P_i\}$, and a set of semantic deductive rules. A *function* F_i^n is a mapping from ordered sets of n objects, called the arguments of F_i^n, into the $\{O_i\}$. The mapping may be multivalued and is defined only if the arguments satisfy a set of conditions associated with F_i^n. A condition is essentially membership in a class of objects, but is defined more precisely below. A relation R_i^n is a special type of object in the model, and consists of a label (a unique identifier), and an ordered set of n conditions, called the argument conditions for the relation. Functions of relations are again relations.

An *elementary proposition* P_i consists of a label associated with some relation, R_i^n, and an ordered set of n objects satisfying the argument conditions for this relation. One may think of these propositions as the beliefs of a speaker about what relationships between objects he has noticed are true in the world. *Complex propositions* are logical combinations (in the usual sense) of elementary propositions.

The *semantic deductive rules* give procedures for adding new propositions to the model based on the propositions now in the model. In addi-

tion to the ordinary rules of logic, these rules include axioms about the relationships of the relations in the model. The semantic deductive rules also include links to the senses of the speaker. For example, one such deductive rule for adding a proposition to the model might be (loosely speaking) "Look in the real world and see if it is true." These rules essentially determine how the model is to be expanded, and are the most complex part of a complete system. However, from our present point of view, we need only consider these rules as a black box which can extend the set of propositions in the model.

A *closed question* is a relational lable for some R_i^n and an ordered set of n objects. The *answer* to this question is affirmative if the proposition, consisting of this label and the n objects, is in the model (or can be added to it according to the semantic deductive rules). If the negation of this proposition is in the model (or can be added), the answer is negative. Otherwise the answer is undefined.

An *open question* consists of a relational label for an n-argument relation, R_i^n, and a set of objects corresponding to n-k of these arguments, where $n \geq k \geq 1$. An answer to an open question is an ordered set of k objects, such that if these objects are associated with the k unspecified arguments of R_i^n, the resulting proposition is in the model, or can be added to it. An open question may have no answers, or may have one or more answers. A *condition* is an open question with k=1, and an object satisfies a condition if it is an answer to the question.

2) *Generation of Kernel Sentences.* We have described the logical properties of the speaker's model of the world. We shall now consider how strings in a language, words, phrases, and sentences, are associated with the model. Corresponding to the set of objects $\{O_i\}$ there is a set $\{N_{ij}\}$ of strings (in English in our case), called the *names* of the objects. There is a many-one mapping from $\{N_{ij}\}$ onto $\{O_i\}$. It is many-one because one object may have more than one name, e.g. frankfurter and hot dog both map back into the same object in the model.

Recall that functions map n-tuples of objects into objects. Thus a function name and an n-tuple can specify an object. We can derive a name for this object from the function name and the names of its n arguments. Associated with each function is at least one linguistic form, a string of words with blanks in which names of arguments of the function must be inserted. Examples of linguistic forms associated with a model are "number of _____", "father of _____", and "the child of _____ and _____". There is a many-one mapping from the set of linguistic forms $\{L_{ij}^n\}$ cnto the set of functions. Two examples of multiple linguistic forms for the same function are: "father of _____" and "_____'s father"; and "_____ plus _____" and "the sum of _____ and _____". Thus, if objects x and y have names "the first number" and "the second number" and associated with the function "*" is the linguistic form "the product of _____ and _____", then the name of the object produced by applying the function "*" to x and y is "the product of the first number and the second number". A parsing of a name must decompose it into the part which is the linguistic form, and the parts which are names of arguments of the corresponding function. We shall call objects defined in terms of a function and an n-tuple of objects a *functionally defined object*, and those which are not functionally defined we shall call *simple objects*. Simple objects have *simple names* and functionally defined objects have *composite names*.

In addition to linguistic forms associated with functions, there are linguistic forms associated with relations. For an n argument relation there are n blanks in the linguistic form. Examples of relational linguistic forms are: "_____ equals _____", "_____ gave _____ to _____" and "_____ speaks". These linguistic forms, corresponding to the relations in the model, serve as frames for the kernel sentences.

In a manner similar to the way composite names are built, a kernel sentence corresponding to an elementary proposition is constructed by inserting names corresponding to each argument in the appropriate blank. Names may be simple or composite. An example of a kernel sentence for a proposition built from such a relational linguistic form is "John's father gave .3 times the salary of Bill to Jack." which con-

tains the simple names "John", ".3", "bill", and "Jack". It contains the functional linguistic forms "_____'s father", "_____ times _____" and "salary of _____" and the relational linguistic form "_____gave to_____".

A kernel sentence corresponding to a complex proposition is constructed recursively from the kernel sentences corresponding to its elementary propositional constituents by placing them in the corresponding places in the linguistic forms "_____ and _____", "_____ or _____", "not _____" etc.

The kernel sentence corresponding to a closed question is constructed from the kernel of the corresponding proposition by placing it in the linguistic form "It is true that _____?" For an open question, dummy objects are placed in the open argument positions to complete a propositional form. These dummy arguments have names "who", "what", "where", etc., and which dummy objects are used depends on the condition on that argument position. A question mark is placed at the end of the kernel sentence constructed in the usual way from the relational linguistic form and the names of the arguments.

In generating a coherent discourse, a speaker chooses a number of propositions in his model and/or some open or closed questions. He then uses linguistic information associated with the model to construct the set of kernel sentences corresponding to this set of chosen propositions. In the next section we will discuss how he generates his discourse from this set of kernels.

3) *Transformations on Kernel Sentences.* The set of kernel sentences is the base of the coherent discourse. The meaning of a kernel sentence is the proposition into which it maps, and similarly, the meaning of any name is the object which is its image under the mapping. To this set of kernels we apply a sequence of meaning preserving transformations to get the final discourse. We use the word "transformation" in its broad general sense, not in the narrow technical sense defined by Chomsky.[10]

There two distinct types of transformations, structural and definitional. A structural or syntactic transformation is only dependent on the structure of the kernel string(s) on which it operates. For example, one syntactic transformation takes a kernel in the active voice to one in the passive voice. Another combines two sentences into a single complex coordinate sentence.

One large class of syntactic transformations is used to substitue pronominal phrases for names. Pronominal phrases may be ordinary pronouns such as "he", "she", or "it". They may be referential phrases such as "the latter", "the former" or "this quantity". They may also be truncations of a full name such as "the distance" for "the distance between New York and Los Angeles". In cases where such pronominal reference is made, the coherence of the final discourse is dependent on the order in which the resultant strings appear.

The second type of transformation is definitional. It involves substitutions of linguistic strings and forms for ones appearing in the kernel sentences. For example, for any appearance of "2 times" we may substitute "twice", and for ".5 times" substitute "one half of". In addition to this string substitution, some transformations perform form substitution and rearrangement. For example, for a kernel sentence of the form "x is y more than z", where x, y, and z are any names, one definitional transformation can substitute "x exceeds z by y".

Some transformations are optional, and some may be mandatory if certain forms are present in the kernel set. Certain transformations are used by a speaker for stylistic purposes, for example, to emphasize certain objects; other syntactic transformations, such as those which perform pronominal substitutions, are used because they decrease the depth of a construction, in the sense defined by Yngve.[11]

Let us review the steps in the generation of a coherent discourse. The speaker chooses a set of propositions, the "ideas" he wishes to transmit. He then encodes them as language strings called kernel sentences in the manner described above. He then chooses a sequence of structural and definitional transformations which are defined on this set of kernels or on the ordered set of sentences which result from applications of the first transformations. The resulting sequence of sentences will be a co-

herent discourse to a listener if he knows all the definitional transformations applied. In addition, every pair of distinct names which the speaker maps back into the same object, the listener must also map into a single object.

E., *Analysis of Coherent Discourse*

Generation of coherent discourse consists of two distinguishable steps. From propositions in the speaker's model of the world, he generates an ordered set of kernel sentences. He then applies a sequence of transformations to this kernel set. The resulting discourse is a coded message which is to be analyzed and decoded by a listener. The listener's problem can be loosely characterized as an attempt to answer the question, "What would I have meant if I said that?"

To analyze a discourse the listener must find the set of kernel sentences from which it was generated; one way to do this is to find a set of inverse transformations which when applied to the input discourse yield a sequence of kernel sentences. The listener must then transform these kernel sentences to an appropriate representation in his information store. The appropriateness of a representation is a function of what later use the listener expects to make of the information contained in the discourse. The listener may simultaneously transform a given kernel sentence into a number of different representations in his information store. On a level of pragmatic analysis, statements require only storage of information. Questions and imperatives require appropriate responses from the listener. The difficulties in analysis dichotomize into those associated with finding the kernel sentences which are the base of the discourse, and those associated with transforming the kernel sentences into representations in the information store.

STUDENT'S analytical program utilizes a set of inverse analytic transformations. If T_i is a transformation that may be used in generating a discourse, and T_i (S) = S, where S and S are sets of sentences, then the analytic transformation T_i^{-1} is the inverse of T_i if and only if T_i^{-1} (S) = S. The choice of which inverse transformations to apply and the order of their application may be restricted by utilizing heuristics concerned with features of the input.

Once the base set of kernel sentences for a given discourse is determined, there remains the problem of entering representations of these sentences in the listener's information store. The major problem in accomplishing this step involves the separation of those words which are part of linguistic forms for relations, and those which are part of a name. This is difficult because the same word (lexicographic symbol) may have multiple uses in a language. Having separated the relational form from the names which represent the arguments of this relation, one can then analyze the name in terms of components which are functional linguistic forms and others which are simple names. From this parsing in terms of relational linguistic forms, functional linguistic forms and simple names, the discourse can be transformed into a canonical representation in the information store of the listener.

F. *Limited Deductive Models*

A complete understanding of a discourse by a listener would imply that the representation of the discourse in his information store is essentially isomorphic to the speaker's model of the world, at least for the universe of discourse. The listener's representation must preserve all information implicit in the discourse.

If the listener is only interested in certain aspects of the discourse, he need only preserve information relevant to his interest, and discard the rest. Within his area of interest the listener's model is isomorphic to the speaker's model in the sense that all relevant deductions which can be made by the speaker on the basis of the discourse can also be made by the listener. Outside this area of interest, the listener will be unable to answer any questions. We call such restricted information stores *limited deductive models.*

The question-answering programs of Lindsay and Raphael, and the STUDENT system, all utilize limited deductive models. For the area of interest in each of these programs there is a "natural" representation for the information in the allowable input. These representations are natural in that they facilitate the deduction of implicit information. For example, Lindsay's

family tree representation makes it easy to compute the relationship of any two individuals in the tree, independent of the number of sentences necessary to build the tree.

Because the number of relations and functions expressible in the models in all three systems is very limited, there is a corresponding limitation on the number of linguistic forms that may appear in the input. This greatly simplifies the parsing problem by restricting alternatives for forms in the input text.

G. *The STUDENT Deductive Model*

The STUDENT system is an implementation of the analytic portion of our theory. STUDENT performs certain inverse transformations to obtain a set of kernel sentences and then transforms these kernel sentences to expressions in a limited deductive model. Utilizing the power of this deductive model, within its limited domain of understanding, it is able to answer questions based on information implicit in the input information.

The analytic and transformational techniques utilized in STUDENT are described in detail in the next section. We shall describe here the canonical representation of objects, relations, and functions within the model. STUDENT is restricted to answering questions framed in the context of algebra story problems. Algebraic equations are a natural representation for information in the input.

The objects in the model are numbers, or numbers with an associated dimension. The only relation in the model is equality, and the only functions represented directly in the model are the arithmetic operations of addition, negation, multiplication, division and exponentiation. Other functions are defined in terms of these basic functions, by composition, and/or substitution of constants for arguments of these functions. For example, the operation of squaring is defined as exponentiation with "2" as the second argument of the exponential function; subtraction is a composition of addition and negation.

Within the computer, a parenthesized prefix notation is used for a standard representation of the equations implicit in the English input. The arithmetic operation to be expressed is made the first element of a list, and the arguments of the function are succeeding list elements. The exact notation is given in Figure 1. In the figure, A, B, and C are any representations of objects in the model, either composite or simple names. The usual infix notation for these functional expressions is given for comparison. Because this is a fully parenthesized notation, no ambiguity of operational order arises, as it does, for example, for the unparenthesized infix notation expression A*B+C, or its corresponding natural language expression "A times B plus C". Note also that in this prefix notation *plus* and *times* are not strictly binary operators. Indeed, in the model they may have any finite number of arguments, e.g. (TIMES A B C D) is a legitimate expression in the STUDENT model.

Representations of objects in the STUDENT deductive model are taken from the input. Any strings of words *not* containing a linguistic form associated with an arithmetic function expressible in the model are considered simple names for objects. Thus, "the age of the child of John and Jane" is considered a simple name because it contains no functional linguistic forms associated with functions represented in STUDENT's limited deductive model. In a more general model it would be considered a composite name, and the functional forms "age of _____" and "child of _____ and _____" would be mapped into their corresponding functions in the model.

Figure 1: Notation Within the STUDENT Deductive Model

Operation	*Infix Notation*	*Prefix Notation*
Equality	A = B	(EQUAL A B)
Addition	A + B	(PLUS A B)
	A + B + C	(PLUS A B C)
Negation	− A	(MINUS A)
Subtraction	A − B	(PLUS A (MINUS B))
Multiplication	A * B	(TIMES A B)
	A * B * C	(TIMES A B C)
Division	A / B	(QUOTIENT A B)
Exponentiation	A^B	(EXPT A B)

Figure 1. Flow Chart of the Student Program

Because such complex strings are considered simple names in the model, and objects are distinguished only by their names, it is important to determine when two distinct names actually refer to the same object. In fact, answers to questions in the STUDENT system are statements of the identity of the object referenced by two names. However, one of the names (the desired one) must satisfy a certain lexical condition. Most often this condition is just that the name be a numeral. For a more general model this restriction could be stated as requiring a simple name corresponding to some functionally defined name—because, for example, "number of _____" would be a functional linguistic form in the general model, and the only simple name for such an object would be the numeral corresponding to this number. An answer consists of a statement of identity e.g. "The number of customers Tom gets is 162."

The other lexical restriction on answers sometimes used in the STUDENT system is insistence that a certain unit (corresponding to a dimension associated with a number) appear in the desired answer. For example, *spans* is the unit specified by the question "How many spans equals 1 fathom?", and the answer given by STUDENT is "1 fathom is 8 spans."

The deductive model described here is useful for answering questions because we know how to extract implicit information from expressions in this model. More explicity, we know how to solve sets of algebraic equations to find numerical values which satisfy these equations.

III. *Transformation of English to the STUDENT Deductive Model*

Our question-answering system contains two main programs which process English input. One is a program called STUDENT which accepts the statement of an algebra story problem and attempts to find the solution to the particular problem. STUDENT does not store any information, nor "remember" any thing from problem to problem. The information obtained by STUDENT is the local context of the question.

The other program is called REMEMBER and it processes and stores facts not specific to any one problem. These facts make up STUDENT's store of "global information" as opposed to the "local information" contained in the statement of any one problem. This information is accepted in a subset of English which overlaps but is different from the subset of English accepted by STUDENT. REMEMBER accepts statements in certain fixed formats, and for each format the information is stored in a way that makes it convenient for retrieval and use within the STUDENT program. The following examples indicate some of the acceptable formats for global information:

"Distance equals speed times time."
"Feet is the plural of foot."
"Bill is a person."
"Times is an operator of level 1."
"One half always means .5."

We will not give any details of the processing done by REMEMBER, except to note that some of the global information is stored by actually adding statements to the STUDENT program, and other information is stored on dictionary entries for the individual words.

A. *Outline of the Operation of STUDENT*

To provide perspective by which to view the detailed heuristic techniques used in the STUDENT program, we shall first give an outline of the operation of the STUDENT program when given a problem to solve. This outline is a verbal description of the flow chart of the program found in the appendix.

STUDENT is asked to solve a particular problem. We assume that all necessary global information has been stored previously. STUDENT will now transform the English input statement of this problem into expressions in its limited deductive model, and through appropriate deductive procedures attempt to find a solution. More specifically, STUDENT finds the kernel sentences of the input discourse, and transforms this sequence of kernels into a set of simultaneous equations, keeping a list of the answers required, a list of the units involved in the problem (e.g. dollars, pounds) and a list of all the variables (simple names) in the equations. Then STUDENT invokes the SOLVE program to solve this set of equations for the desired unknowns. If a solution is found, STUDENT prints the values of the un-

knowns requested in a fixed format, substituting in "(*variable IS value*)" the appropriate phrases for *variable* and *value*. If a solution cannot be found, various heuristics are used to identify two variables (i.e. find two slightly different phrases that refer to the same object in the model). If two two variables, A and B, are identified, the equation A = B is added to the set of equations. In addition, the store of global information is searched to find any equations that may be useful in finding the solution to this problem. STUDENT prints out any assumptions it makes about the identity of two variables, and also any equations that it retrieves because it thinks they may be relevant. If the use of global equations or equations from identifications leads to a solution, the answers are printed out in the format described above.

If a solution is not found, and certain idioms are present in the problem (a result of a definitional transformation used in the generation of the problem), a substitution is made for each of these idioms in turn and the transformation and solution process is repeated. If the substitutions for these idioms do not enable the problem to be solved by STUDENT, then STUDENT requests additional information from the questioner, showing him the variables being used in the problem. If any information is given, STUDENT tries to solve the problem again. If none is given, it reports its inability to solve this problem and terminates. If the problem is ever solved, the solution is printed and the program terminates.

B. *Categories of Words in a Transformation.*

The words and phrases (strings of words) in the English input can be classified into three distinct categories on the basis of how they are handled in the transformation to the deductive model. The first category consists of strings of words which name objects in the model; I call such strings, *variables*. Variables are identified only by the string of words in them, and if two strings differ at all, they define distinct variables. One important problem considered below is how to determine when two distinct variables refer to the same object.

The second category of words and phrases are what I call *substitutors*. Each substitutor may be replaced by another string. Some substitutions are mandatory; others are optional and are only made if the problem cannot be solved without such substitutions. An example of a mandatory substitutions is "2 times" for the word "twice". "Twice" always means "2 times" in the context of the model, and therefore this substitution is mandatory. One optional "idiomatic" substitution is "twice the sum of the length and width of the rectangle" for "the perimeter of the rectangle". The use of these substitutions in the transformation process is discussed below. These substitutions are inverses of definitional transformations as defined earlier.

Members of the third category of words indicate the presence of functional linguistic forms which represent functions in the deductive model. I call members of this third class *operators*. Operators may indicate operations which are complex combinations of the basic functions of the deductive model. One simple operator is the word "plus", which indicates that the objects named by the two variables surrounding it are to be added. An example of a more complex operator is the phrase "percent less than", as in "10 percent less than the marked price", which indicates that the number immediately preceding the "percent" is to be subtracted from 100, this result divided by 100, and then this quotient multiplied by the variable following the "than".

Operators may be classified according to where their arguments are found. A prefix operator, such as "the square of ...." precedes its argument. An operator like "....percent" is a suffix operator, and follows its argument. Infix operators such as ".....plus....." or "....less than ....." appear between their two arguments. In a split prefix operator such as "difference between ..... and .....", part of the operator precedes, and part appears between the two arguments. "The sum of .....and.....and....." is a split prefix operator with an indefinite number of arguments.

Some words may act as operators conditionally, depending on their context. For example, "of" is equivalent to "times" if there is a fraction immediately preceding it; e.g., ".5 of the profit" is equivalent to ".5 times the profit"; however, "Queen of England" does not imply

a multiplicative relationship between the Queen and her country.

C. *Transformational Procedures.*

Let us now consider in detail the transformation procedure used by STUDENT, and see how the three categories of phrases interact. Let us consider the following example, which has been solved by STUDENT.

(THE PROBLEM TO BE SOLVED IS)
(IF THE NUMBER OF CUSTOMERS TOM GETS IS *TWICE* THE *SQUARE OF* 20 *PER CENT* OF THE NUMBER OF ADVERTISEMENTS HE RUNS, AND THE NUMBER OF ADVERTISEMENTS HE RUNS IS 45, WHAT IS THE NUMBER OF CUSTOMERS TOM GETS Q.)

Shown below are copies of actual printout from the STUDENT program, illustrating stages in the transformation and the solution of the problem. The parentheses are an artifact of the LISP programming language, and "Q." is a replacement for the question mark not available on the key punch.

The first stage in the transformation is to perform all mandatory substitutions. In this problem only the three phrases underlined (by the author, not the program) are substitutors: "twice" becomes "2 times", "per cent" becomes the single word "percent", and "square of" is truncated to "square". Having made these substitutions, STUDENT prints:

(WITH MANDATORY SUBSTITUTIONS THE PROBLEM IS)
(IF THE NUMBER OF CUSTOMERS TOM GETS IS *2 TIMES* THE *SQUARE* 20 *PERCENT* OF THE NUMBER OF ADVERTISEMENTS HE RUNS, AND THE NUMBER OF ADVERTISEMENTS HE RUNS IS 45, WHAT IS THE NUMBER OF CUSTOMERS TOM GETS Q.)

From dictionary entries for each word, the words in the problem are tagged by their function in terms of the transformation process, and STUDENT prints:

(WITH WORDS TAGGED BY FUNCTION THE PROBLEM IS)
(IF THE NUMBER (OF / OP) CUSTOMERS (GETS / VERB) IS 2 (TIMES / OP 1) THE (SQUARE / OP 1) 20 (PERCENT / OP 2) (OF/OP) THE NUMBER (OF / OP) ADVERTISEMENTS (HE / PRO) RUNS, AND THE NUMBER (OF / OP) ADVERTISEMENTS (HE / PRO) RUNS IS 45, (WHAT / QWORD) IS THE NUMBER (OF / OP) CUSTOMERS TOM (GETS / VERB) (QMARK / DLM))

If a word has a tag, or tags, the word followed by "/", followed by the tags, becomes a single unit, and is enclosed in parentheses. Some typical taggings are shown above. "(OF/OP)" indicates that "OF" is an operator and other taggings show that "GETS" is a verb, "TIMES" is an operator of level 1 (operator levels will be explained below), "SQUARE" is an operator of level 1, "PERCENT" is an operator of level 2, "HE" is a pronoun, "WHAT" is a question word, and "QMARK" (replacing Q.) is a delimiter of a sentence. These tagged words will play the principal role in the remaining transformation to the set of equations implicit in this problem statement.

The next stage in the transformation is to break the input sentences into "kernel sentences". As in the example, a problem may be stated using sentences of great grammatical complexity; however, the final stage of the transformation is only defined on a set of kernel sentences. The simplification to kernel sentences as done in STUDENT depends on the recursive use of format matching. If an input sentence is of the form "IF" followed by a substring, followed by a comma, a question word and a second substring then the first substring (between the IF and the comma) is made an independent sentence, and everything following the comma is made into a second sentence. In the example, this means that the input is resolved into the following two sentences, (where tags are omitted for the sake of brevity):

"The number of customers Tom gets is 2 times the square 20 percent of the number of advertisements he runs, and the number of advertisements he runs is 45." and "What is the number of customers Tom gets?"

This last procedure effectively resolves a problem into declarative assumptions and a question sentence. A second complexity resolved by STUDENT is illustrated in the first

sentence of this pair. A coordinate sentence consisting of two sentences joined by a comma immediately followed by an "and" will be resolved into these two independent sentences. The first sentence is therefore resolved into two simpler sentences.

Using these two inverse syntactic transformations, this problem statement is resolved into "simple" kernel sentences. For the example, STUDENT prints

(THE SIMPLE SENTENCES ARE)
(THE NUMBER (OF / OP) CUSTOMERS TOM (GETS / VERB) IS 2 (TIMES / OP 1) THE (SQUARE / OP 1) 20 (PERCENT / OP 2) (OF / OP) THE NUMBER (OF / OP) ADVERTISEMENTS (HE / PRO) RUNS (PERIOD / DLM)
(THE NUMBER (OF / OP) ADVERTISECENTS (HE / PRO) RUNS IS 45 (PERIOD / DLM)
((WHAT / QWORD) IS THE NUMBER (OF / OP) CUSTOMERS TOM (GETS / VERB) (QMARK / DLM))

Each simple sentence is a separate list, i.e., is enclosed in parentheses, and each ends with a delimiter (a period or question mark). Each of these sentences can now be transformed directly to its interpretation in the model.

D. *From Kernel Sentences to Equations*

The transformation from the simple kernel sentences to equations uses three levels of precedence for operators. Operators of higher precedence level are used earlier in the transformation. Before utilizing the operators, STUDENT looks for linguistic forms associated with the equality relation. These forms include the copula "is" and transitive verbs in certain contexts. In the example we are considering, only the copula "is" is used to indicate equality. The use of transitive verbs as indicators of equality, that is, as relational linguistic forms, will be discussed in connection with another example. When the relational linguistic form is identified, the names which are the arguments of the form are broken down into variables and operators (functional linguistic forms). In the present problem, the two names are those on either side of the "is" in each sentence.

The word "is" may also be used meaningfully within algebra story problems as an auxiliary verb (*not* meaning equality) in such verbal phrases as "is multiplied by" or "is divided by". A special check is made for the occurrence of these phrases before proceeding on to the main transformation procedure. The transformation of sentences containing these special verbal phrases will be discussed later. If "is" does not appear as an auxiliary in such a verbal phrase, a sentence of the form "P1 is P2" is interpreted as indicating the equality of the objects named by phrases P1 and P2. No equality relation will be recognized within these phrases, even if an appropriate transitive verb occurs within either of them. If P1* and P2* represent the arithmetic transformations of P1 and P2, then "P1 is P2" is transformed into the equation "(EQUAL P1* P2*)".

The transformation of P1 and P2 to give them an interpretation in the model is performed recursively using a program equivalent to the table in Figure 2. This table shows all the operators and formats currently recognized by the STUDENT program. New operators can easily be added to the program equivalent of this table.

In performing the transformation of a phrase P, a left to right search is made for an operator of level 2 (indicated by subscripts of "OP" and 2). If there is none, a left to right search is made for a level 1 operator (indicated by subscripts "OP" and 1), and finally another left to right search is made for an operator of level 0 (indicated by a subscript "OP" and no numerical subscript). The first operator found in this ordered search determines the first step in the transformation of the phrase. This operator and its context are transformed as indicated in column 4 in the table. If no operator is present, delimiters and articles (*a, an* and *the*) are deleted, and the phrase is treated as an indivisible entity, a variable.

In the example, the first simple sentence is

(THE NUMBER (OF/OP) CUSTOMERS TOM (GETS/VERB) IS 2 (TIMES/OP 1) THE (SQUARE/OP 1) 20 (PERCENT/OP 2) (OF/OP) THE NUMBER (OF/OP) ADVERTISEMENTS (HE/PRO) RUNS (PERIOD/DLM))

Figure 2

Operator	Precedence Level	Context	Interpretation in the Model		
PLUS	2	P1 PLUS P2	(PLUS P1* P2*)	(a)	
PLUSS	0	P1 PLUSS P2	(PLUS P1* P2*)	(b)	
MINUS	2	P1 MINUS P2 MINUS P2	(PLUS P1* (MINUS P2*)) (MINUS P2*)	(c)	
MINUSS	0	P1 MINUSS P2	(PLUS P1* (MINUS P2*))	(b)	
TIMES	1	P1 TIMES P2	(TIMES P1* P2*)		
DIVBY	1	P1 DIVBY P2	(QUOTIENT P1* P2*)		
SQUARE	1	SQUARE P1	(EXPT P1* 2)	(d)	
SQUARED	0	P1 SQUARED	(EXPT P1* 2)		
**	0	P1 ** P2	(EXPT P1* P2*)		
LESSTHAN	2	P1 LESSTHAN P2	(PLUS P2* (MINUS P1*))		
PER	0	P1 PER K P2 P1 PER P2	(QUOTIENT P1* (K P2)*) (QUOTIENT P1* (1 P2)*)	(e)	(f)
PERCENT	2	P1 K PERCENT P2	(P1 (K/100) P2)*	(f)	(g)
PERLESS	2	P1 K PERLESS P2	(P1((100–K)/100) P2)*	(f)	(g)
SUM	0	SUM P1 AND P2 AND P3 SUM P1 AND P2	(PLUS P1* (SUM P2 AND P3)*) (PLUS P1* P2*)		
DIFFERENCE	0	DIFFERENCE BETWEEN P1 AND P2	(PLUS P1* (MINUS P2*))		
OF	0	K OF P2 P1 OF P2	(TIMES K P2*) (P1 OF P2)*		

(a) If P1 is a phrase, P1* indicates its interpretation in the model.
(b) *PLUSS* and *MINUSS* are identical to PLUS AND MINUS except for precedence level.
(c) When two possible contexts are indicated, they are checked in the order shown.
(d) SQUARE P1 and SUM P1 are idiomatic shortenings of SQUARE OF P1 and SUM OF P1.
(e) * outside a parenthesized expression indicates that the enclosed phrase is to be transformed.
(f) K is a number.
(g) / and – imply that the indicated arithmetic operations are actually performed.

This is of the form "P1 is P2", and is transformed to (EQUAL P1* P2*). P1 is "(THE NUMBER (OF/OP) CUSTOMERS TOM (GETS/VERB))". The occurrence of the verb "gets" is ignored because of the presence of the "is" in the sentence, meaning "equals". The only operator found is "(OF/OP)". From the table we see that if "OF" is immediately preceded by a number (*not* the word "number") it is treated as if it were the infix "TIMES". In this case, however, "OF" is not preceded by a number; the subscript OP, indicating that "OF" is an operator, is stripped away, and the transformation process is repeated on the phrase with "OF" no longer acting as an operator. In this repetition, no operators are found, and P1* is the variable

(NUMBER OF CUSTOMERS TOM (GETS/VERB)).

To the right of "IS" in the sentence is P2:

(2 (TIMES/OP 1) THE (SQUARE/OP 1) 20 (PERCENT/OP 2) (OF/OP) THE NUMBER (OF/OP) ADVERTISEMENTS (HE/PRO) RUNS (PERIOD/DLM)) The first operator found in P2 is PERCENT, an operator of level 2.

From the table in Figure 2, we see that this operator has the effect of dividing the number immediately preceding it by 100. The "PER-

CENT" is removed and the transformation is repeated on the remaining phrase. In the example, the ". . . .20 (PERCENT/OP 2) (OF/OP) . . . ." becomes ". . . .2000(OF/OP) . . . .".

Continuing the transformation, the operators found are, in order, TIMES, SQUARE, OF and OF. Each is handled as indicated in the table. The "OF" in the context ". . . .2000(OF/OP) THE . . . ." is treated as an infix TIMES, while at the other occurrence of "OF", the operator marking is removed. The resulting transformed expression for P2 is:

(TIMES 2 (EXPT (TIMES .2 (NUMBER OF ADVERTISEMENTS (HE/PRO) RUNS)) 2))

The transformation of the second sentence of the example is done in a similar manner, and yields the equation:

(EQUAL (NUMBER OF ADVERTISEMENTS (HE/PRO) RUNS) 45)

The third sentence is of the form "What is P1?". It starts with a question word and is therefore treated specially. A unique variable, a single word consisting of an *X* of *G* followed by five integers, is created, and the equation (EQUAL Xnnnnn P1*) is stored. For this example, the variable X00001 was created, and this last simple sentence is transformed to the equation:

(EQUAL X00001 (NUMBER OF CUSTOMERS TOM (GETS/VERBS))

In addition, the created variable is placed on the list of variables for which STUDENT is to find a value. Also, this variable is stored, paired with P1, the untransformed right side, for use in printing out the answer. If a value is found for this variable, STUDENT prints the sentence (P1 is *value*) with the appropriate substitution for *value*. Below we show the full set of equations, and the printed solution given by STUDENT for the example being considered. For ease in solution, the last equations created are put first in the list of equations.

(THE EQUATIONS TO BE SOLVED ARE)
(EQUAL X00001 (NUMBER OF CUSTOMERS TOM (GETS/VERB))) (EQUAL (NUMBER OF ADVERTISEMENTS (HE/PRO) RUNS) 45) (EQUAL (NUMBER OF CUSTOMERS TOM (GETS/VERB)) (TIMES 2 (EXPT (TIMES .2000 (NUMBER OF ADVERTISEMENTS (HE/PRO) RUNS)) 2))) (THE NUMBER OF CUSTOMERS TOM GETS IS 162)

In the example just shown, the equality relation was indicated by the copula "is". In the problem shown below, solved by STUDENT, equality is indicated by the occurrence of a transitive verb in the proper context.

(THE PROBLEM TO BE SOLVED IS)
(TOM HAS TWICE AS MANY FISH AS MARY HAS GUPPIES. IF MARY HAS 3 GUPPIES, WHAT IS THE NUMBER OF FISH TOM HAS Q.)
(THE NUMBER OF FISH TOM HAS IS 6)

The verb in this case is "has". The simple sentence "Mary has 3 guppies" is transformed to the "equivalent" sentence "The number of guppies Mary has is 3" and the sentence is processed as before. This transformation rule may be stated generally as: anything (a subject) followed by a verb followed by a number followed by anything (the unit) is transformed to a sentence starting with "THE NUMBER OF" followed by the unit, followed by the subject and the verb, followed by "IS" and then the number. In "Mary has 3 guppies" the subject is "Mary", the verb "has", and the units "guppies". Similarly, the sentence "The witches of Firth brew 3 magic potions" would be transformed to

"The number of magic potions the witches of Firth brew is 3."

In addition to a declaration of number, a single-object transitive verb may be used in a comparative structure, such as exhibited in the sentence "Tom has twice as many fish as Mary has guppies." STUDENT transforms this sentence into the equivalent sentence.

"The number of fish Tom has is twice the number of guppies Mary has."

Transformations of new sentence formats to formats previously "understood" by the program can be easily added to the program, thus extending the subset of English "understood" by STUDENT.

The word "is" indicates equality only if it is not used as an auxiliary. The example below shows how verbal phrases containing "is", such

as "is multiplied by", and "is increased by" are handled in the transformation.

(THE PROBLEM TO BE SOLVED IS)
(A NUMBER IS MULTIPLIED BY 6. THIS PRODUCT IS INCREASED BY 44. THIS RESULT IS 68. FIND THE NUMBER.)

(THE EQUATIONS TO BE SOLVED ARE)
(EQUAL X00001 (NUMBER))
(EQUAL (PLUS (TIMES (NUMBER) 6) 44) 68)

(THE NUMBER IS 4)

The sentence "A number is multiplied by 6" only indicates that two objects in the model are related multiplicatively, and does not indicate explicitly any equality relation. The interpretation of this sentence in the model is the prefix notation product "(TIMES (NUMBER 6))". This latter phrase is stored in a temporary location for possible later reference. In this problem, it is referenced in the next sentence, with the phrase "this produce". The important word in this last phrase is "this"—STUDENT ignores all other words in a variable containing the key word "this". The last temporarily stored phrase is substituted for the phrase containing "this". Thus, the first three sentences in the problem shown above yield only one equation, after two substitutions for "this" phrases. The last sentence "Find the number." is transformed as if it were "What is the number Q.", and yields the first equation shown.

The word "this" may occur in a context where it is not referring to a previously stored phrase. Below is an example with such a context.

(THE PROBLEM TO BE SOLVED IS)
(THE PRICE OF A RADIO IS 69.70 DOLLARS. IF THIS PRICE IS 15 PERCENT LESS THAN THE MARKED PRICE, FIND THE MARKED PRICE.)

(THE MARKED PRICE IS 82 DOLLARS)

In such contexts, the phrase containing "THIS" is replaced by the left half of the last equation created. In this example, STUDENT breaks the last sentence into two simple sentences, deleting the "IF". Then the phrase "THIS PRICE" is replaced by the variable "PRICE OF RADIO", which is the left half of the previous equation.

This problem illustrates two other features of the STUDENT program. The first is the action of the complex operator "percent less than". It causes the number immediately preceding it, i.e., 15, to be subtracted from 100, this result divided by 100, to give .85. Then this operator becomes the infix operator "TIMES". This is indicated in the table in Figure 2.

This problem also illustrates how units such as "dollars" are handled by the STUDENT program. Any word which immediately follows a number is labeled as a special type of variable called a *unit*. A number followed by a unit is treated in the equation as a product of the number and the unit, e.g., "69.70 DOLLARS" becomes "(TIMES 69.70 (DOLLARS))". Units are treated as special variables in solving the set of equations; a unit may appear in the answer though variables cannot. If the value for a variable found by the solver is the product of a number and a unit, STUDENT concatenates the number and the unit. For example, the solution for "(MARKED PRICE)" in the problem above was (TIMES 82 (DOLLARS)) and STUDENT printed out:

(THE MARKED PRICE IS 82 DOLLARS)

There is an exception to the fact that any unit may appear in the answer, as illustrated in the problem below.

(THE PROBLEM TO BE SOLVED IS)
(IF 1 SPAN EQUALS 9 INCHES, AND 1 FATHOM EQUALS 6 FEET, HOW MANY SPANS EQUALS 1 FATHOM Q.)

(THE EQUATIONS TO BE SOLVED ARE)
(EQUALS X00001 (TIMES 1 (FATHOMS))) (EQUAL (TIMES 1 (FATHOMS)) (TIMES 6 (FEET))) (EQUAL (TIMES 1 (SPANS)) (TIMES 9 (INCHES)))

THE EQUATIONS WERE INSUFFICIENT TO FIND A SOLUTION (USING THE FOLLOWING KNOWN RELATIONSHIPS)
((EQUAL (TIMES 1 (YARDS)) (TIMES 3 (FEET))) (EQUAL (TIMES 1 (FEET)) (TIMES 12 (INCHES))))

(1 FATHOM IS 8 SPANS)

If the unit of the answer is specified, in this problem by the phrase "how many *spans*"—then *only* that unit, in this problem "spans",

may appear in the answer. Without this restriction, STUDENT would blithely answer this problem with "(1 FATHOM IS 1 FATHOM)".

In the transformation from the English statement of the problem to the equations, "9 INCHES" became (TIMES 9 (INCHES)). However, "1 FATHOM" became "(TIMES 1 (FATHOMS))". The plural form for fathom has been used instead of the singular form. STUDENT always uses the plural form if known, to ensure that all units appear in only one form. Since "fathom" and "fathoms" are different, if both were used STUDENT would treat them as distinct, unrelated units. The plural form is part of the global information that can be made available to STUDENT, and the plural form of a word is substituted for any singular form appearing after "1" in any phrase. The inverse operation is carried out for correct printout of the solution.

Notice that the information given in the problem above was insufficient to allow solution of the set of equations to be solved. Therefore, STUDENT looked in its glossary for information concerning each of the units in this set of equations. It found the relationships "1 foot equals 12 inches." and "1 yard equals 3 feet." Using only the first fact, and the equation it implies, STUDENT is then able to solve the problem. Thus, in certain cases where a problem is not *analytic*, in the sense that it does not contain, explicitly stated, all the information needed for its solution, STUDENT is able to draw on a body of facts, picking out relevant ones, and use them to obtain a solution.

In certain problems, the transformation process does not yield a set of solvable equations. However, within this set of equations there exists a pair of variables (or more than one pair) such that the two variables are only "slightly different", and really name the same object in the model. When a set of equations is unsolvable, STUDENT searches for relevant global equations. In addition, it uses several heuristic techniques for identifying two "slightly different" variables in the equations. The problem below illustrates the identification of two variables where in one variable a pronoun has been substituted for a noun phrase in the other variable. This identification is made by checking all variables appearing *before* one containing the pronoun, and finding one which is identical to this pronoun phrase, with a substitution of a string of any length for the pronoun.

(THE PROBLEM TO BE SOLVED IS)

(THE NUMBER OF SOLDIERS THE RUSSIANS HAVE IS ONE HALF OF THE NUMBER OF GUNS THEY HAVE. THE NUMBER OF GUNS THEY HAVE IS 7000. WHAT IS THE NUMBER OF SOLDIERS THEY HAVE Q.)

THE EQUATIONS WERE INSUFFICIENT TO FIND A SOLUTION (ASSUMING THAT)
((NUMBER OF SOLDIERS (THEY/PRO) (HAVE/VERB)) IS EQUAL TO (NUMBER OF SOLDIERS RUSSIANS (HAVE/VERB)))

(THE NUMBER OF SOLDIERS THEY HAVE IS 3500)

If two variables match in this fashion, STUDENT assumes the two variables are equal, prints out a statement of this assumption, as shown, and adds an equation expressing this equality to the set to be solved. The solution procedure is tried again, with this additional equation. In this example, the additional equation was sufficient to allow determination of the solution.

The example below is again a non-analytic problem. The first set of equations developed by STUDENT is unsolvable. Therefore, STUDENT tries to find some relevant equations in its store of global information.

(THE PROBLEM TO BE SOLVED IS)
(THE GAS CONSUMPTION OF MY CAR IS 15 MILES PER GALLON. THE DISTANCE BETWEEN BOSTON AND NEW YORK IS 250 MILES. WHAT IS THE NUMBER OF GALLONS OF GAS USED ON A TRIP BETWEEN NEW YORK AND BOSTON Q.)

THE EQUATIONS WERE INSUFFICIENT TO FIND A SOLUTION (USING THE FOLLOWING KNOWN RELATIONSHIPS)
((EQUAL (DISTANCE) (TIMES (SPEED) (TIME))) (EQUAL (DISTANCE) (TIMES (GAS CONSUMPTION)

(NUMBER OF GALLONS OF GAS USED))))

(ASSUMING THAT)
((DISTANCE) IS EQUAL TO (DISTANCE BETWEEN BOSTON AND NEW YORK)
(ASSUMING THAT)
((GAS CONSUMPTION) IS EQUAL TO (GAS CONSUMPTION OF MY CAR))
(ASSUMING THAT)
((NUMBER OF GALLONS OF GAS USED) IS EQUAL TO (NUMBER OF GALLONS OF GAS USED ON TRIP BETWEEN NEW YORK AND BOSTON))

(THE NUMBER OF GALLONS OF GAS USED ON A TRIP BETWEEN NEW YORK AND BOSTON IS 16.66 GALLONS)

It uses the first word of each variable string as a key to its glossary. The one exception to this rule is that the words "number of" are ignored if they are the first two words of a variable string. Thus, in this problem, STUDENT retrieved equations which were stored under the key words *distance, gallons, gas,* and *miles*. Two facts about *distance* had been stored earlier; "distance equals speed times time" and "distance equals gas consumption times number of gallons of gas used". The equations implicit in these sentences were stored and retrieved now—as possibly useful for the solution of this problem. In fact, only the second is relevant.

Before any attempt is made to solve this augmented set of equations, the variables in the augmented set are matched, to identify "slightly different" variables which refer to the same object in the model. In this example "(DISTANCE)," "(GAS CONSUMPTION)", and "(NUMBER OF GALLONS OF GAS USED)" are all identified with "similar" variables. The following conditions must be satisfied for this type of identification of variables P1 and P2:

1) P1 must appear later in the problem than P2.
2) P1 is completely contained in P2 in the sense that P1 is a contiguous substring within P2.

This identification reflects a syntactic phenomenon where a truncated phrase, with one or more modifying phrases dropped, is often used in place of the original phrase. For example, if the phrase "the length of a rectangle" has occurred, the phrase "the length" may be used to mean the same thing. This type of identification is distinct from that made using pronoun substitution.

In the example above, a stored schema was used by identifying the variables in the schema with the variables that occur in the problem. This problem is solvable because the key phrases "distance", "gas consumption" and "number of gallons of gas used" occur as substrings of the variables in the problem. Since STUDENT identifies each generic key phrase of the schema with a particular variable of the problem, any schema can be used only once in a problem. Because STUDENT handles schema in this *ad hoc* fashion it cannot solve problems in which a relationship such as "distance equals speed times time" is needed for two different values of distance, speed, and time.

E. *Possible Idiomatic Substitutions*

There are some phrases which have a dual character, depending on the context. In the example below, the phrase "perimeter of a rectangle" becomes a variable with no reference to its meaning, or definition, in terms of the length and width of the rectangle. This definition is unneeded for solution.

(THE PROBLEM TO BE SOLVED IS)
(THE SUM OF THE PERIMETER OF A RECTANGLE AND THE PERIMETER OF A TRIANGLE IS 24 INCHES. IF THE PERIMETER OF THE RECTANGLE IS TWICE THE PERIMETER OF THE TRIANGLE, WHAT IS THE PERIMETER OF THE TRIANGLE Q.)

(THE PERIMETER OF THE TRIANGLE IS 8 INCHES)

However, the following problem is stated in terms of the perimeter, length and width of the rectangle. Transforming the English into equations is not sufficient for solution. Neither retrieving and using an equation about "inches", the unit in the problem, nor identifying "length" with a longer phrase serve to make the problem solvable. Therefore, STUDENT looks in its dictionary of possible idioms, and

finds one which it can try in the problem. STUDENT

(THE PROBLEM TO BE SOLVED IS)
(THE LENGTH OF A RECTANGLE IS 8 INCHES MORE THAN THE WIDTH OF THE RECTANGLE. ONE HALF OF THE PERIMETER OF THE RECTANGLE IS 18 INCHES. FIND THE LENGTH AND THE WIDTH OF THE RECTANGLE.)

THE EQUATIONS WERE INSUFFICIENT TO FIND A SOLUTION TRYING POSSIBLE IDIOMS
(THE PROBLEM WITH AN IDIOMATIC SUBSTITUTION IS)

(THE LENGTH OF A RECTANGLE IS 8 INCHES MORE THAN THE WIDTH OF THE RECTANGLE. ONE HALF OF TWICE THE SUM OF THE LENGTH AND WIDTH OF THE RECTANGLE IS 18 INCHES. FIND THE LENGTH AND THE WIDTH OF THE RECTANGLE.)

(ASSUMING THAT)
((LENGTH) IS EQUAL TO (LENGTH OF RECTANGLE))

(THE LENGTH IS 13 INCHES)
(THE WIDTH OF THE RECTANGLE IS 5 INCHES)

actually had two possible idiomatic substitutions which it could have made for "perimeter of a rectangle"; one was in terms of the length and width of the rectangle and the other was in terms of the shortest and longest sides of the rectangle. When there are two possible substitutions for a given phrase, one is tried first, namely the one STUDENT has been told about most recently. In this problem, the correct one was fortuitously first. If the other had been first, the revised problem would not have been solvable, and eventually the second (correct) substitution would have been made. Only one non-mandatory idiomatic substitution is ever made at one time, although the substitution is made for all occurrences of the phrase chosen.

In this problem, the idiomatic substitution made allows the problem to be solved, after identification of the variables "length" and "length of rectangle". The retrieved equation about inches was not needed. However, its presence in the set of equations to be solved did not sidetrack the solver in anyway.

This use of possible, but non-mandatory idiomatic substitutions can also be used to give STUDENT a way to solve problems in which two phrases denoting one particular variable are quite different. For example, the phrase, "students who passed the admissions test" and "successful candidates" might be describing the same set of people. However, since STUDENT knows nothing of the "real world" and its value system for success, it would never identify these two phrases. However, if told that "successful candidates" sometimes means "students who passed the admissions test", it would be able to solve a problem using these two phrases to identify the same variable. Thus, possible idiomatic substitutions serve the dual purpose of providing tentative substitutions of definitions, and identification of synonomous phrases.

F. *Special Heuristics.*

The methods thus far discussed have been applicable to the entire range of algebra problems. However, for special classes of problems, additional heuristics may be used which are needed for members of the class, but not applicable to other problems. An example is the class of age problems, as typified by the problem below.

(THE PROBLEM TO BE SOLVED IS)
(BILL S FATHER S UNCLE IS TWICE AS OLD AS BILL S FATHER. 2 YEARS FROM NOW BILL S FATHER WILL BE 3 TIMES AS OLD AS BILL. THE SUM OF THEIR AGES IS 92. FIND BILL S AGE.)

(BILL S AGE IS 8)

Before the age problem heuristics are used, a problem must be identified as belonging to that class of problems. STUDENT identifies age problems by any occurrence of one of the following phrases, "as old as", "years old" and "age". This identification is made immediately after all words are looked up in the dictionary and tagged by function. After the special heuristics are used the modified problem is transformed to equations as described previously.

The need for special methods for age problems arises because of the conventions used for denoting the variables, all of which are ages. The word age is usually not used explicitly,

but is implicit in such phrases as "as old as". People's names are used where their ages are really the implicit variables. In the example, for instance, the phrase "Bill's father's uncle" is used instead of the phrase "Bill's father's uncle's age".

STUDENT uses a special heuristic to make all these ages explicit. To do this, it must know which words are "person words" and therefore, may be associated with an age. For this problem STUDENT has been told that *Bill, father,* and *uncle* are person words. The "spaces -s" following a word is the STUDENT representation for possessive, used instead of "apostrophe -s" for programming convenience. STUDENT inserts a "S AGE" after every person word *not* followed by a "S" (because this "S" indicates that the person word is being used in a possessive sense, not as an independent age variable). Thus, as indicated, the phrase "BILL S FATHER S UNCLE" becomes "BILL S FATHER S UNCLE S AGE".

In addition to changing phrases naming people to ones naming ages, STUDENT makes certain special idiomatic substitutions. For the phrase "their ages", STUDENT substitutes a conjunction of all the age variables encountered in the problem. In the example, for "THEIR AGES" STUDENT substitutes "BILL S FATHER S UNCLE S AGE AND BILL S FATHER S AGE AND BILL S AGE". The phrases "as old as" and "years old" are then deleted as dummy phrases not having any meaning, and "will be" and "was" are changed to "is". There is no need to preserve the tense of the copula, since the sense of the future or past tense is preserved in such prefix phrases as "2 years from now", or "3 years ago".

The remaining special age problem heuristics are used to process the phrases "in 2 years", "5 years ago" and "now". The phrase "2 years from now" is transformed to "in 2 years" before processing. These three time phrases may occur immediately after the word "age", (e.g., Bill's age 3 years ago") or at the beginning of the sentence. If a time phrase occurs at the beginning of the sentence, it implicitly modifies all ages mentioned in the sentence, *except* those followed by their own time phrase. For example, "In 2 years Bill's father's age will be 3 times Bill's age" is equivalent to "Bill's father's age in 2 years will be 3 times Bill's age in 2 years". However, "3 years ago Mary's age was 2 times Ann's age now" is equivalent to "Mary's age 3 years ago was 2 times Ann's age now". Thus prefix time phrases are handled by distributing them over all ages not modified by another time phrase.

After these prefix phrases have been distributed, each time phrase is translated appropriately. The phrase "in 5 years" causes 5 to be added to the age it follows, and "7 years ago" causes 7 to be subtracted from the age preceding this phrase. The word "now" is deleted.

Only the special heuristics described thus far are necessary to solve the first age problem. The second age problem, given below, requires one additional heuristic not previously mentioned. This is a substitution for the phrase "was when" which effectively decouples the two facts combined in the first sentence. For "was when", STUDENT substitutes "was K years ago. K years ago" where K is a new variable created for this purpose.

(THE PROBLEM TO BE SOLVED IS)
(MARY IS TWICE AS OLD AS ANN WAS WHEN MARY WAS AS OLD AS ANN IS NOW. IF MARY IS 24 YEARS OLD, HOW OLD IS ANN Q.)

(THE EQUATIONS TO BE SOLVED ARE)
(EQUAL X00008 ((ANN / PERSON) S AGE))
(EQUAL ((MARY / PERSON) S AGE) 24)
(EQUAL (PLUS ((MARY / PERSON) S AGE) (MINUS X00007))) ((ANN / PERSON) S AGE))
(EQUAL ((MARY / PERSON) S AGE) (TIMES 2 (PLUS ((ANN / PERSON) S AGE) (MINUS X00007)))))

(ANN S AGE IS 18)

In the example, the first sentence becomes the two sentences: "Mary is twice as old as Ann X00007 years ago. X00007 years ago Mary was as old as Ann is now." These two occurrences of time phrases are handled as discussed previously. Similarly the phrase "will be when" would be transformed to "in K years. In K years".

These decoupling heuristics are useful not only for the STUDENT program but for people trying to solve age problems. The classic age problem about Mary and Ann, given above, took an MIT graduate student over 5 minutes to solve because he did not know this heuristic. With the heuristic he was able to set up the appropriate equations much more rapidly. As a crude measure of STUDENT's speed, note that STUDENT took less than one minute to solve this problem.

G. *When All Else Fails.*

For all the problems discussed thus far, STUDENT was able to find a solution eventually. In some cases, however, necessary global information is missing from its store of information, or variables which name the same object cannot be identified by the heuristics of the program. Whenever STUDENT cannot find a solution for any reason, it turns to the questioner for help. As in the problem below, it prints out "(DO YOU KNOW ANY MORE RELATIONSHIPS BETWEEN THESE VARIABLES)" followed by a list of the variables in the problem. The questioner can answer "yes" or "no". If he says "yes", STUDENT says "TELL ME", and the questioner can append another sentence to the statement of the problem.

(THE PROBLEM TO BE SOLVED IS)
(THE GROSS WEIGHT OF A SHIP IS 20000 TONS. IF ITS NET WEIGHT IS 15000 TONS, WHAT IS THE WEIGHT OF THE SHIPS CARGO Q.)
THE EQUATIONS WERE INSUFFICIENT TO FIND A SOLUTION
(DO YOU KNOW ANY MORE RELATIONSHIPS AMONG THESE VARIABLES)
(GROSS WEIGHT OF SHIP)
(TONS)
(ITS NET WEIGHT)
(WEIGHT OF SHIPS CARGO)
yes
TELL ME
(the weight of a ships cargo is the difference between the gross weight and the net weight)
THE EQUATIONS WERE INSUFFICIENT TO FIND A SOLUTION

(ASSUMING THAT)
((NET WEIGHT) IS EQUAL TO (ITS NET WEIGHT)
(ASSUMING THAT)
((GROSS WEIGHT) IS EQUAL TO (GROSS WEIGHT OF SHIP))
(THE WEIGHT OF THE SHIPS CARGO IS 5000 TONS)

In this problem, the additional information typed in (in lower case letters) was sufficient to solve the problem. If it was not, the question would be repeated until the questioner said "no", or provides sufficient information for solution of the problem.

In the problem below, the solution to the set of equations involves solving a quadratic equation, which is beyond the mathematical ability of the present STUDENT system. Note that in this case STUDENT reports that the equations were unsolvable, not simply insufficient for solution. STUDENT still requests additional information from the questioner. In the example, the questioner says "no", and STUDENT states that "I CANT SOLVE THIS PROBLEM" and terminates.

(THE PROBLEM TO BE SOLVED IS)
(THE SQUARE OF THE DIFFERENCE BETWEEN THE NUMBER OF APPLES AND THE NUMBER OF ORANGES ON THE TABLE IS EQUAL TO 9. IF THE NUMBER OF APPLES IS 7, FIND THE NUMBER OF ORANGES ON THE TABLE.)
UNABLE TO SOLVE THIS SET OF EQUATIONS

TRYING POSSIBLE IDIOMS
(DO YOU KNOW ANY MORE RELATIONSHIPS AMONG THESE VARIABLES)
(NUMBER OF APPLES)
(NUMBER OF ORANGES ON TABLE)
no

I CANT SOLVE THIS PROBLEM

H. *Summary of the STUDENT Subset of English*

The subset of English understandable by STUDENT is built around a core of sentence and phrase formats which can be transformed into expressions in the STUDENT deductive model. On this basic core is built a larger set of formats. Each of these are first transformed into a string built on formats in this basic set and then this string is transformed into an ex-

pression in the deductive model. For example, the format ($ IS EQUAL TO $) is changed to the basic format ($ IS $), and the phrase "IS CONSECUTIVE TO" is changed to "IS 1 PLUS". The constructions discussed earlier involving single object transitive verbs could have been handled this way, though for programming convenience they were not.

The basic linguistic form which is transformed into an equation is one containing "is" as a copula. The phrases "is equal to" and "equals' are both changed to the coupla "is". The auxiliary verbal constructions "is multiplied by", "is divided by" and "is increased by" are also acceptable as principal verbs in a sentence. As discussed in detail earlier, a sentence with no occurrence of "is" can have as a main verb a transitive verb immediately followed by a number. This number must be an element of the phrase which is the direct object of the verb, as in "Mary has three guppies". This type of transitive verb can also have a comparative structure as direct object, e.g., "Mary has twice as many guppies as Tom has fish".

This completes the repertoire of declarative sentence formats. Any number of declarative sentences may be conjoined, with ", and" between each pair, to form a new (complex) declarative sentence. A declarative sentence (even a complex declarative) can be made a presupposition for a question by preceding it with "IF" and following it with a comma and the question.

Questions, that is, requests for information from STUDENT, will be understood it they match any of the following patterns (where $ will match any string, and $1 any one word):

(WHAT ARE $ AND $)	(WHAT IS $)
(FIND $ AND $)	(FIND $)
(HOW MANY $ DO $ HAVE)	(HOW MANY $ DOES $ HAVE)
(HOW MANY $1 IS $)	

This completes the summary of the set of input formats presently understood by STUDENT. This set can be enlarged in two distinct ways. One is to enlarge the set of basic formats, using standard subroutines to aid in defining, for each new basic format, its interpretation in the deductive model. The other method of extending the range of STUDENT input is to define transformations from new input formats to previously understood basic or extension formats.

Even if a story problem is stated within the subset of English acceptable to STUDENT, this is not a guarantee that this problem can be solved by STUDENT (assuming it to be solvable). Two phrases describing the object must be at worst only "slightly different" by the criteria prescribed earlier. Appropriate global information must be available to STUDENT, and the algebra involved must not exceed the abilities of the solver. However, though most algebra story problems found in the standard texts cannot be solved by STUDENT exactly as written, the author has usually been able to find some paraphrase of almost all such problems which is solvable by STUDENT.

I. *Limitations of the STUDENT Subset of English*

The techniques presented here are general and can be used to enable a computer program to accept and understand a fairly extensive subset of English for a fixed semantic base. However, the current STUDENT system is experimental and has a number of limitations.

STUDENT's interpretation of the input is based on format matching. If each format is used to express the meaning understood by STUDENT, no misinterpretation will occur. However, these formats occur in English discourse even in algebra story problems, in semantic contexts not consistent with STUDENT's interpretation of these formats. For example, a sentence containing", and" is always interpreted by STUDENT as the conjunction of two declarative statements. Therefore, the sentence "Tom has 2 apples, 3 bananas, and 4 pears." would be incorrectly divided into the two "sentences", "Tom has 2 apples, 3 bananas." and "4 pears."

Each of the operator words shown in Figure 2 must be used as an operator in the context as shown or a misinterpretation will result. For example, the phrase "the number of times I went to the movies" which should be inter-

preted as a variable string will be interpreted incorrectly as the product of the two variables "number of" and "I went to the movies", because "times" is always considered to be an operator. Similarly, in the current implementation of STUDENT, "of" is considered to be an operator if it is preceded by any number. However, the phrase "2 of the boys who passed" will be misinterpreted as the product of "2" and "the boys who passed".

These examples obviously do not constitute a complete list of misinterpretations and errors STUDENT will make, but it should give the reader an idea of limitations on the STUDENT subset of English. In principle, all of these restrictions could be removed. However, removing some of them would require only minor changes to the program, while others would require techniques not used in the current system.

For example, to correct the error in interpreting "2 of the boys who passed", one can simply check to see if the number before the "of" is less than 1, and if so, only then interpret "of" as an operator "times". However, a much more sophisticated grammer and parsing program would be necessary to distinguish different occurrences of", and" and correctly extract simpler sentences from complex coordinate and subordinate sentences.

Because of limitations of the sort described above, and the fact that the STUDENT system currently occupies almost all of the computer memory, STUDENT serves principally as a demonstration of the power of the techniques utilized in its construction. However, I believe that on a larger computer one could use these techniques to construct a system of practical value which would communicate well with people in English over the limited range of material understood by the program.

IV. CONCLUSION

A. *Results*

The purpose of the research reported here was to develop techniques which facilitate natural language communication with a computer. A semantic theory of coherent discourse was proposed as a basis for the design and understanding of such man-machine systems. This theory was only outlined, and much additional work remains to be done. However, in its present rough form, the theory served as a guide for construction of the STUDENT system, which can communicate in a limited subset of English.

The language analysis in STUDENT is an implementation of the analytic portion of this theory. The STUDENT system has a very narrow semantic base. From the theory it is clear that by utilizing this knowledge of the limited range of meaning of the input discourse, the parsing problem becomes greatly simplified, since the number of linguistic forms that must be recognized is very small. If a parsing system were based on any small semantic base, this same simplification would occur. This suggests that in a general language processor, some time might be spent putting the input into a semantic context before going ahead with the syntactic analysis.

The semantic base of the STUDENT language analysis is delimited by the characteristics of the problem solving system embedded in it. STUDENT is a question-answering system which answers questions posed in the context of "algebra story problems." We shall use four general criteria for evaluating this question-answering system.

1) *Extent of Understanding.* Other question-answering systems analyze input sentence by sentence. Although a representation of the meaning of all input sentences may be placed in some common store, no syntactic connection is ever made between sentences.

In the STUDENT system, an acceptable input is a sequence of sentences, such that these sentences cannot be understood by just finding the meanings of the individual sentences, ignoring their local context. Inter-sentence dependencies must be determined, and inter-sentence syntactic relationships must be used in this case for solution of the problem given. This extension of the syntactic dimension of understanding is important because such inter-sentence dependencies (e.g., the use of pronouns) are very commonly used in natural language communication.

The semantic model in the STUDENT system is based on one relationship (equality) and

five basic arithmetic functions. Composition of these functions yield other functions which are also expressed as individual linguistic forms in the input language. The input language is richer in expressing functions than Lindsay's or Raphael's system. Some logic-based question-answering systems may have more relationships (predicates) allowable in the input, but do not allow any composition of these predicates. The logical combinations of predicates used are only those expressed in the input as logical combinations (using and, or, etc.)

The deductive system in STUDENT, as in Lindsay's and Raphael's programs, is designed for the type of questions to be asked. It can only deduce answers of a certain type from the input information, that is arithmetic values satisfying a set of equations. In performing its deductions it is reasonably sophisticated in avoiding irrelevant information, as are the other two mentioned. It lacks the general power of a logical system, but is much more efficient in obtaining its particular class of deductions than would be a general deductive system utilizing the axioms of arithmetic.

2) *Facility for Extending Abilities.* Extending the syntactic abilities of most other question-answering systems would require reprogramming. In the STUDENT system new definitional transformations can be introduced at run time without any reprogramming. The information concerning these transformations can be input in English, or in a combination of English and METEOR, if that is more appropriate. New syntactic transformations must be added by extending the program.

The semantic base of the STUDENT system can be extended only by adding new program, as is true of other question-answering systems. However STUDENT is organized to facilitate such extensions, by minimizing the interactions of different parts of the program. The necessary information need only be added to the program equivalent of the table of operators in Figure 2.

Similarly, the deductive portion of STUDENT, which solves the derived set of equations, is an independent package. Therefore, a new extended solver can be added to the system by just replacing the package, and maintaining the input-output characteristics of this subroutine.

3) *Knowledge of Internal Structure Needed by User.* Very little if any internal knowledge of the workings of the STUDENT system need be known by the user. He must have a firm grasp of the type of problem that STUDENT can solve, and a knowledge of the input grammar. For example, he must be aware that the same phrase must always be used to represent the same variable in a problem, within the limits of similarity defined earlier. He must realize that even within these limits STUDENT will not recognize more than one variation on a phrase. But if the user does forget any of these facts, he can still use the system, for the interaction discussed in the next section allows him to make amends for almost any mistake.

4) *Interaction with the User.* The STUDENT system is embedded in a time-sharing environment and this greatly facilitates interaction with the user. STUDENT differentiates between its failure to solve a problem because of its mathematical limitations and failure from lack of sufficient information. In case of failure it asks the user for additional information, and suggests the nature of the needed information (relationships among variables of the problem). It can go back to the user repeatedly for information until it has enough to solve the problem, or until the user gives up.

STUDENT also reports when it does not recognize the format of an input sentence. Using this information as a guide, the user is in a teaching-machine type situation, and can quickly learn to speak STUDENT's brand of input English. By monitoring the assumptions that STUDENT makes about the input, and the global information it uses, the user can stop the system and reword a problem to avoid an unwanted ambiguity, or add new general information to the global information store.

B. *Extensions*

The present STUDENT system has reached the maximum size allowable in the LISP system on a thirty-two thousand word IBM 7094. Therefore, very little can be added directly to the present system. All the programming extensions mentioned here are predicated on the

existence of a computer with a much larger memory.

Without inventing any new techniques, I think that the STUDENT system could be made to understand most of the algebra story problems that appear in first year high school text books. If new operators, new combinations of arithmetic operations occur, they can easily be added to the subroutine which maps the kernel English sentences into equations. The number of formats recognizable in the system can be increased without reprogramming through the machinery available for storing global information. The problems it would not handle are those having excessive verbiage or implied information about the world not expressible in a single sentence.

As mentioned earlier, the system can now make use of any given schema only once in solving a problem. This is because the schema equation is added to the set of equations to be solved, and the variables in the schema only identified with one another set of variables appearing in the problem. For example, if "distance equals speed times time" were the schema, the "distance", as a variable in the schema might be set equal to "distance traveled by train" or "distance traveled by plane", but not both in the same problem. This problem could be resolved by not adding the schema equation directly to the set of equations to be solved, but by looking for consistent sets of variables to identify with the schema variables. Then STUDENT could add an instance of the schema equations, with the appropriate substitutions, for each consistent set of variables found which are "similar" to the schema variables.

At the moment the solving subroutine of STUDENT can only perform linear operations on literal equations, and substitutions of numbers in polynomials and exponentials. It would be relatively easy to add the facility for solving quadratic or even higher order solvable equations. One could even add, quite easily, sufficient mechanisms to allow the solver to perform the differentiation needed to do related rate problems in the differential calculus.

The semantic base of the STUDENT system could be expanded. In order to add the relations recognized by the SIR system of Raphael, for example, one would have to add on the lowest level of the STUDENT program the set of kernel sentences understood in SIR, their mapping to the SIR model, and the question-answering routine to retrieve facts. Then the apparatus of the STUDENT system would process much more complicated input statements for the SIR model. One serious problem which arises when the semantic base is extended is based on the fact that one kernel may have an interpretation in terms of two different semantic bases. For example, "Tom has 3 fish." can be interpreted in both SIR and the present STUDENT system. To resolve this semantic ambiguity, the program can check the context of the ambiguous statement to see if there has been one consistent model into which all the other statements have been processed. If the latter condition does not determine a single preferred interpretation for the statement, then both interpretations can be stored.

One use for our model for generation and analysis of discourse would be as a hypothesis about the linguistic behavior of people. STUDENT may be a good predictive model for the behaviour of people when confronted with an algebra problem to solve. This can be tested, and such a study may lead to a better understanding of human behaviour, and/or a better reformulation of this theory of language processing.

I think we are far from writing a program which can understand all, or even a very large segment of English. However, within its narrow field of competence, STUDENT has demonstrated that "understanding" machines [12] can be built. Indeed, I believe that using the techniques developed in this research, one could construct a system of practical value which would communicate well with people in English over the range of material understood by the program.

REFERENCES

1. BOBROW, D. G., "METEOR: A LISP Interpreter for String Transformations"; in (13)
2. MCCARTHY, J., et. al., *LISP 1.5 Programmers Reference Manual;* MIT Press, Cambridge, Mass.; 1963

3. CORBATO, F. J., et. al., *The Compatible Time-Sharing System;* MIT Press, Cambridge, Mass.; 1963
4. GREEN, B. F., et. al., "Baseball: An Automatic Question Answerer"; *Proc. WJCC;* May 1961
5. LINDSAY, R. K., "Inferential Memory as the Basis of Machines Which Understand Natural Language," in (14)
6. RAPHAEL, B., "A Computer Program Which 'Understands' "; *Proc. FJCC;* Spartan Press, Baltimore, Maryland; 1964
7. SIMMONS, R. F., "Answering English Questions by Computer—A Survey"; SDC Report SP–1556, Santa Monica, California; April 1964
8. BOBROW, D. G., "Natural Language Input for a Computer Problem Solving System", Ph.D. Thesis, Mathematics Department, M.I.T., Cambridge, Mass.; 1964
9. QUINE, W. V., *Word and Object;* MIT Press, Cambridge, Mass.; 1960
10. CHOMSKY, A. N., *Syntactic Structures;* Mouton and Co.; S-Grauenhage; 1957
11. YNGVE, V., "A Model and an Hypothesis for Language Structure," *Proceedings of the American Philosophical Society,* vol. 104, No. 5; 1960
12. MINSKY, M., "Steps toward Artificial Intelligence" in (14)
13. BERKLEY, E. C. and BOBROW, D. G. (editors) *The Programming Language LISP: Its Operation and Applications,* Information International Inc., Cambridge, Mass.; 1964
14. FEIGENBAUM, E. and FELDMAN, J. (editors) *Computers and Thought,* McGraw Hill, New York; 1963

THE UNIT PREFERENCE STRATEGY IN THEOREM PROVING*

Lawrence Wos, Daniel Carson, and George Robinson
Argonne National Laboratory
Argonne, Illinois

Unit Preference and Set of Support Strategies

The theorems, axioms, etc., to which the algorithm and strategies described in this paper are applied are stated in a normal form defined as follows: A *literal* is formed by prefixing a predicate letter to an appropriate number of arguments (constants, variables, or expressions formed with the aid of function symbols) and then perhaps writing a negation sign (−) before the predicate letter. For example:

P(b,x) −P(b,x) Q(y) R(a,b,x,z,c) S

are all literals if P, Q, R, and S are two-, one-, five-, and zero-place predicate letters, respectively. The predicate letter is usually thought of as standing for some n-place relation. Then the literal P(a,b), for example, is thought of as saying that the ordered pair (a,b) has the property P. The literal −P(a,b) is thought of as saying that (a,b) does not have the property P.

One may build a *clause* by writing a sequence of literals separated by disjunction (logical "or") signs. Logical "or" will be symbolized by a small letter "v" (distinguishable from possible uses of "v" as a variable from context). Where it is desired to indicate dependence of a particular argument of a predicate letter upon one or more other variables, function symbols are employed. The clause

P(x,y,z) v Q(x,f(x,y))

thus says that either the ordered triple (x,y,z) has the property P or the ordered pair (x,f(x,y)) has the property Q (or both). Each of the variables in a clause is then thought of as being universally quantified (that is, if the variable x occurs in a clause, the clause is assumed to be preceded by an implicit quantifier, "for each x." Functional expressions such as f(x,y) are then treated as existentially quantified variables. Roughly speaking, f(x,y) in the example above stands for an element (depending on x and y) which forms, when used as the second element with x as the first element, an ordered pair which has the property Q. A *unit clause* is a clause composed of a single literal.

Finally, one may consider a sequence of clauses implicitly joined by logical "and" (conjunction). Such a sequence of clauses will be said to be in *(conjunctive) normal form.*

Instantiation, as applied to this normal form, can be thought of as the forming of a possibly less general (more specific) *instance* of a clause by performing a systematic replacement of variables by constants, new variables, or by expressions formed with the aid of function symbols. Substituting b for x and f(d,u) for y in the clause

P(x,y) v Q(b,y)

would yield a less general instance of that clause:

P(b,f(d,u)) v Q(b,f(d,u)).

* Work performed under the auspices of the U. S. Atomic Energy Commission.