mt

mechanical translation

DEVOTED TO THE TRANSLATION OF LANGUAGES BY THE AID OF MACHINES

Vol. 7, No. 2

August, 1963

Syntactic Dependence and the Computer Generation of Coherent Discourse

Sheldon Klein and Robert F. Simmons

AN INTERNATIONAL JOURNAL PUBLISHED AT THE MASSACHUSETTS INSTITUTE OF TECHNOLOGY

Syntactic Dependence and the Computer Generation of Coherent Discourse

by Sheldon Klein and Robert F. Simmons,* System Development Corporation, Santa Monica, California

An experiment in the computer generation of coherent discourse was successfully conducted to test a hypothesis about the transitive nature of syntactic dependency relations among elements of the English language. The two primary components of the experimental computer program consisted of a phrase structure generation grammar capable of generating grammatical nonsense, and a monitoring system which would abort the generation process whenever it was apparent that the dependency structure of a sentence being generated was not in harmony with the dependency relations existing in an input source text. The final outputs of the system were coherent paraphrases of the source text. An implication of the hypothesis is that certain types of dependency relations are invariant under a variety of linguistic transformations. Potential applications include automatic kernelizing, question answering, automatic essay writing, and automatic abstracting systems.

The question of the validity of transitive dependency models for languages other than English should be explored.

Introduction

This paper sets forth the hypothesis that there is in the English language a general principle of transitivity of dependence among elements and describes an experiment in the computer generation of coherent discourse that supports the hypothesis.

The hypothesis of transitive dependency, simply stated, is that if a word or element *a* modifies a word *b* and *b* modifies *c*, it may be said that *a* transitively modifies, or is dependent on, *c*. Based on this principle it was found possible to design and program a system to generate coherent discourse using both the AN/FSQ-32 (a large IBM military computer) and the IBM 7090. The input to the coherent discourse generator consists of written English text which has been analyzed in terms of syntactic dependency relations. The output is a large set of sentences generated by the computer, each of which is a coherent paraphrase of some portions of the input text.

We treat syntactic dependency as a primitive relation which is transitive in some environments, intransitive in others. While dependency may always be transitive in a system of formal logical syntax for English, results indicate that this is not always true for a semantic interpretation of that system. The totality of the conditions under which dependency is treated as intransitive is subject to empirical determination by analysis of the output of the discourse generator.

One of the components of the system is a phrase structure generation grammar which can generate grammatically correct nonsense. The vocabulary of a source text is placed in the vocabulary pool of this program, and the generation of grammatical nonsense is initiated.

At the same time, a monitoring program inspects the sentence being generated and aborts the generation process whenever it is apparent that such a sentence would have dependency relations incompatible with those of the source text. The only permissible final output is a coherent paraphrase of the source text.

From one point of view, the system functions as a decision procedure for determining whether or not a sentence is the result of an application of legitimate transformations upon other sentences. The implication is that dependency, with its transitive and intransitive aspects, may be an invariant under many linguistic transformations. Also, the coherent discourse generator can be modified to act as an automatic kernelizer of English sentences.

It is also possible to describe the operation of the system in terms of the Stratificational Grammar of Sydney Lamb[8]. By relying upon constancies of word co-occurrence, the system provides a method of going from the morphemic stratum of a source to the morphemic stratum of an output, bypassing the sememic stratum.

BACKGROUND

In attempting to discover a logic that would allow a computer to answer questions from a natural English language text[11], we observed early that an acceptable answer could take many forms. Words different from those in the question would sometimes be the most

* This research was sponsored by the Advanced Research Projects Agency, under contract SD-97.

natural for an answer. This could be taken care of by a thesaurus or synonym dictionary. But often, even where all the words of the question were represented in the answer, the syntactic structure was remarkably different. It became apparent very quickly that in addition to the well-known fact of synonymy of different words in English, there existed a considerable degree of syntactic synonymy in which the same words in different syntactic structures could nevertheless carry essentially the same meaning.

For example, the question "Where do large birds live?" transforms into a bewildering complexity of answering forms: "Living large birds are found (where)." "Birds that are large live (where)." "Living in (where), large birds, etc." These examples are of course just a few and are only those in which the words of the question occur in the answer.

Syntactic analysis of the answers showed that there was less variation in syntactic form than we had originally thought. But the fact that a word belonged in a particular type of phrase in the question gave no assurance that the same structure would be present in an answer. However, as we studied the syntactic trees of the potential answers we gradually realized that there was a relationship that appeared to be invariant.

The relative order of words on the syntactic dependency tree was approximately the same in every acceptable answer as it was in the question. Thus "Where do birds live?" gave a tree structure in which "live" is dependent on "birds," "where" is dependent on "live" and "birds" is the subject dependent upon itself. Each of the answers maintains these dependency relationships although there may be other words inserted at nodes on the syntactic tree between them. For example in the sentence "Birds that are large live (where)," "large" is dependent on "are," which is dependent on "that," which is finally dependent on "birds." Thus, in a transitive sense, "large" is dependent on "birds."

The relative invariance of order of words on the syntactic tree gave rise to the concept of *transitive dependency*. If there exists a wholly upward path between two words in a dependency tree, the two words are transitively dependent. As a further result of this idea, the concept of *syntactic synonymy* came to have quantifiable meaning. If two statements containing the same vocabulary tokens (excepting only inflectional changes) contain the same transitive dependencies, they are syntactically synonymous.

If these two ideas are generally valid, they imply that many operations on language hitherto impossible for computers at once become practical. For example, meaning-preserving transformations of the type that Harris[5] performs, automatic essay writing, question answering and a whole realm of difficult language processing tasks all require (among other things) that syntactic synonymy be recognized and dealt with.

To study the generality of the principle of transitive dependency we began some programs that use this principle for generating coherent discourse. The hypothesis underlying these computer experiments may be stated roughly as follows. Given as input a set of English sentences, if we hold constant the set of vocabulary tokens and generate grammatical English statements from that vocabulary with the additional restriction that their transitive dependencies agree with those of the input text, the resulting sentences will all be truth-preserving paraphrases derived from the original set. To the extent that they are truth-preserving derivations, our rules for the transitivity of dependence are supported.

In order to accomplish this experiment we borrowed heavily from the program for generating grammatically valid but semantically nonsensical sentences described in detail by Yngve.[13, 14] We modified his approach by selecting words as each part of the sentence became defined rather than waiting until the entire pattern was generated. In addition, we devised a system that adds the restriction that each word selected must meet the transitive dependency relations of the input text.

The coherent discourse generator, the dependency rules underlying it, its outputs, its applications and implications form the body of this paper.

Dependency

Before describing the design and operation of the coherent discourse generator, it is first necessary to explain the ground rules of dependency—the primitives on which the system is based. If English were a language in which each word necessarily modified the following word, the dependency structure would be immediately obvious—each word would be dependent on the succeeding word. Unfortunately, English, though highly sequential in nature, is not completely so; in order to uncover the relation of *modification* or *dependency,* a syntactic analysis is first necessary. (Such dependency analysis systems as that of D. Hays[6], which go directly from word class to dependency links include in their computer logic most of the rules necessary to an immediate constituency analysis.) The result of a syntactic analysis is a tree structure whose different levels include word class descriptions, phrase names and clause designations.

The dependency analysis of these tree structures is simply a convenient notation that emphasizes one feature of the syntactic analysis. The feature emphasized is the relation of modification or dependency. A word at any node of the dependency tree is directly dependent on another word if and only if there is a single line between the node of the first word and that of the second. For our purpose the strength of dependency notation lies in the fact that it facilitates expression of transitive relations. The precise form that our rules of dependency take was determined empirically; the rules chosen were those that facilitated the selection of answers to questions and the generation of coherent discourse. We have attempted to state

these rules as generally as possible in order to allow compatibility with a variety of syntactic analysis systems.

The elements of a sentence in our pilot model were taken to be words. (A more sophisticated model might include dependency relations among morphemes.) The rules of dependency are as follows:

1. The head of the main verb phrase of a sentence or clause is dependent upon the head of the subject.
2. The head of a direct object phrase is dependent upon the head of the governing verb phrase.
3. Objects of prepositions are dependent upon those prepositions.
4. Prepositions are dependent upon the heads of the phrases they modify. Prepositions in the predicate of a sentence are dependent upon the head of a verb phrase and also upon the head of an intervening noun phrase if one is present.
5. Determiners and adjectives are dependent upon the head of the construction in which they appear.
6. Adverbs are dependent upon the head of the verb phrase in which they appear.
7. Two-way dependency exists between the head of a phrase and any form of the verb "to be" or the preposition "of." This rule holds for the heads of both phrases linked to these forms.
8. Two-way dependency within or across sentences also exists between tokens of the same noun and between a pronoun and its referent.
9. Dependencies within a passive sentence are treated as if the sentence were an active construction.
10. The head of the subject is dependent upon itself or upon a like token in a preceding sentence.

In both the computer program and the following examples the dependencies are expressed in the form of a list structure. The words in the text are numbered in sequential order; where one word is dependent on another, the address of that other word is stored with it as follows:

0	.	0
1	John	1
2	eats	1
3	fish	2
4	.	4

A more complex example is the following:

0	.	0	
1	The	2	Rule 5
2	man	2, 19, 3, 8	Rules 10, 8, 8, and 8
3	who	2	Rule 8
4	has	3	Rule 1
5	the	6	Rule 5
6	book	4, 15	Rule 2, 8
7	rode	2	Rule 1
8	his	9, 2	Rules 5, 8
9	bicycle	7	Rule 2
10	in	9, 7	Rules 4, 4
11	the	12	Rule 5
12	afternoon	10	Rule 3
13	.	13	
14	The	15	Rule 5
15	book	6, 17	Rules 8, 9, 2
16	was		
17	written	19	Rules 9, 1
18	by		
19	him	19, 2	Rules 10, 8
20	in	17, 15	Rules 4, 4
21	two	22	Rule 5
22	hours	20	Rule 3
23	.	23	

The rules for determining the antecedents of pronouns across sentences are not perfect. In general it is assumed that the referent of a pronoun occupies a parallel syntactic function. For this purpose all sentences are treated as if they were active constructions, and special rules for case are also taken into consideration. Nevertheless, the style of some writers will yield constructions that do not fit the rules. In such cases, it is usually only the context of the real world which resolves the problem for live speakers, and sometimes not then, e.g., "The book is on the new table. It is nice."

THE TRANSITIVITY OF DEPENDENCE

The dependency relationships between words in English statements are taken as primitives for our language processing systems. We have hypothesized that the dependency relation is generally transitive; that is, if *a* is dependent on *b*, and *b* is dependent on *c*, then *a* is dependent on *c*. The purpose of the experiment with the coherent discourse generator is to test this hypothesis and to explore its limits of validity.

It was immediately obvious that if dependency were always transitive—for example, across verbs and prepositions—the discourse generator would occasionally construct sentences that were not truth-preserving derivations of the input text. For example, "The man ate soup in the summer" would allow the generation of "The man ate summer" if transitivity were permitted across the preposition. As a consequence of our experimentation, the following rules of non-transitivity were developed:

1. No transitivity across verbs except forms of the verb "to be."
2. No transitivity across prepositions except "of."
3. No transitivity across subordinating conjunctions (if, although, when, where, etc.).

There are no doubt additional exceptions and additional convenient rules of dependency, such as the two-way linkage that we use for "to be" and "of," which will improve the operation of language process-

ing systems. However, we have noticed that each special rule eliminates some errors and causes others. The problem is very similar to the completeness problem for most interesting systems of formal logic (Gödel[10]) in which the unattainable goal is to devise a system in which all and only true theorems are derivable. Gödel proved that one has a choice of obtaining only true theorems but not all true theorems in the system, or all true theorems in the system at the cost of also obtaining false theorems.

The Generation Process

PHRASE STRUCTURE GENERATION OF GRAMMATICAL NONSENSE

The basis for computer generation of grammatically correct nonsense via a phrase structure generation grammar has been available for several years. The program design of the generation grammar used in this system was initially modeled after one developed by Victor Yngve[13, 14]. The recursive phrase structure formulas that such a grammar uses were first crystallized by Zellig Harris[4] in 1946. The purpose of the formulas in Harris's formulation, however, was to describe utterances in terms of low-level units combining to form higher-level units. Chomsky[1] later discussed these types of rules in application to the generation of sentences.

The phrase structure generation grammar uses such rules to generate lower-order units from higher-order units. As an example, consider the following short set of rules which are sufficient to generate an indefinitely large number of English sentences, even though the rules themselves account for only a very small portion of the structure of English:

1. $N_2 = Art_0 + N_1$
2. $N_1 = Adj_0 + N_1$
3. $N_1 = N_0$
4. $V_2 = V_1 + N_2$
5. $V_1 = V_0$
6. $S = N_2 + V_2$

where "Art" stands for article, "Adj" for adjective, "N" for noun phrase, "V" for verb phrase, "S" for sentence. To accomplish the generation, substitution of like form class types is permitted but with such substitution controlled by subscripts. For example, the right half of an N_1 formula or an N_2 formula could be substituted for an N_2, but the right half of an N_2 formula could not be substituted for an N_1. The use of subscripts is not the only way to control the order of application of formulas. It is a modification of a method used by Yngve[13] and was suggested as one of several methods by Harris[4].

In the usual description of a phrase structure generation grammar, left-most entities are expanded first and an actual English word is not substituted for its class descriptor until the subscript of that class marker reaches a certain minimal value.

For example:

S
$N_2 + V_2$ (rule 6)
$Art_0 + N_1 + V_2$ (rule 1)

Here "Art" has a minimal subscript and one may pick an English article at random.

The $+ N_1 + V_2$
The $+ Adj_0 + N_1 + V_2$ (rule 2)

Another zero subscript permits a random choice of an adjective.

The + tall $+ N_1 + V_2$
The tall $+ Adj_0 + N_1 + V_2$ (rule 2)

Note that formula 2 might be applied recursively *ad infinitum.*

The + tall + dark $+ N_1 + V_2$
The + tall + dark $+ N_0 + V_2$ (rule 3)
The + tall + dark + elephant $+ V_2$
The + tall + dark + elephant $+ V_1 + N_2$ (rule 4)
The + tall + dark + elephant $+ V_0 + N_2$ (rule 5)
The + tall + dark + elephant + eats $+ N_2$
The + tall + dark + elephant + eats $+ N_0$ (rule 3)
The + tall + dark + elephant + eats + rocks

In Yngve's program particular rules were chosen at random, as were vocabulary items.

Agreement of number can be handled in several ways. One could build rules that dealt with individual morphemes rather than word classes as terminal outputs; one might make use of duplex sets of rules for singular and plural constructions accompanied by singular and plural vocabulary lists; or one might have a special routine examine the output of a generation process and change certain forms so that they would agree in number.

Table 1 shows a sample output of the generation grammar which puts only grammatical restrictions on the choice of words. All sentences are grammatically correct according to a simple grammar, but usually nonsensical.

THE GENERATION OF COHERENT DISCOURSE

Description of the System

The basic components of the coherent discourse generator are a phrase structure grammatical nonsense generator which generates randomly and a monitoring system which inspects the process of sentence generation, rejecting choices which do not meet the dependency restrictions.

A source text is selected and analyzed in terms of dependency relations. In the system under discussion this has been accomplished by hand. The vocabulary

TABLE 1

COMPUTER-GENERATED GRAMMATICAL NONSENSE

TOBACCO	OF	THE	POTATOES	INCLUDE	CHOICE	.			
GENERAL	SUGAR	CROPS	UPON	DIFFERENT	FINANCING	DEPENDS	SYSTEMS	.	
MARKET	ARE MIXED	UPON	CROPS	.					
RICE	INCLUDE	THE	RICE	.					
FARM	DRY	FARM	SUGAR	CROPS	ARE MIXED	BEETS	.		
BEETS	INCLUDE	SUGAR	MARKET	.					
SOIL	ARE MIXED	THE	FARM	DRY	GENERAL	BEETS	.		
THE	SYSTEMS	ARE MIXED	GENERAL	CLIMATE	.				
CROPS	INCLUDE	SUGAR	CROPS	.					
PEANUTS	ARE MIXED	UPON	POTATOES	.					
THE	TOBACCO	ARE MIXED	UPON	BEETS	.				
CLIMATE	DEPENDS	SOIL	.						
BEANS	IN	DIFFERENT	SYSTEMS	INCLUDE	MARKET	OF	DRY	GENERAL	DRY
CLIMATE	.								
DIFFERENT	CANE	ARE MIXED	THE	GENERAL	PEANUTS	.			
GENERAL	POTATOES	DEPENDS	SUGAR	CANE	OF	THE	CASH	BEANS	.
DIFFERENT	CASH	CHOICE	INCLUDE	THE	CLIMATE	.			
THE	FINANCING	INCLUDE	UPON	GENERAL	FARM	DRY	BEETS	.	
THE	BEANS	UPON	THE	SUGAR	DRY	MARKET	INCLUDE	THE	CHOICE
DIFFERENT	CROPS	UPON	DIFFERENT	FINANCING	INCLUDE	DIFFERENT	SYSTEMS	.	
MARKET	DEPENDS	DIFFERENT	CROPS	.					
SOIL	ARE MIXED	DIFFERENT	FARM	DRY	MARKET	.			
THE	FARM	SUGAR	SOIL	OF	FARM	DRY	CROPS	INCLUDE	OF
MARKET	.								
MARKET	ARE MIXED	SUGAR	TOBACCO	.					
DIFFERENT	SUGAR	DRY	FINANCING	INCLUDE	GENERAL	TOBACCO	.		
DRY	RICE	DEPENDS	SYSTEMS	.					
DIFFERENT	CANE	ARE MIXED	DRY	POTATOES	UPON	DIFFERENT	CASH	TOBACCO	.
DIFFERENT	SOIL	DEPENDS	CASH	CHOICE	OF	TOBACCO	.		
SUGAR	CROPS	OF	CASH	PEANUTS	INCLUDE	MARKET	.		
FINANCING	ARE MIXED	CHOICE							
THE	BEETS	DEPENDS	CANE	.					
SYSTEMS	OF	DIFFERENT	SUGAR	SOIL	ARE MIXED	THE	PEANUTS	.	
DIFFERENT	CHOICE	ARE MIXED	THE	TOBACCO	.				
POTATOES	INCLUDE	THE	BEANS	.					
BEETS	INCLUDE	GENERAL	CHOICE	.					
GENERAL	DRY	TOBACCO	INCLUDE	THE	SUGAR	CASH	DRY	CHOICE	.
THE	CASH	GENERAL	SUGAR	CROPS	UPON	CLIMATE	DEPENDS	THE	CASH
GENERAL	CANE	.							
FARM	CASH	CHOICE	INCLUDE	THE	FARM	BEANS	.		
THE	SOIL	ARE MIXED	THE	SYSTEMS	.				
DIFFERENT	SYSTEMS	INCLUDE	DIFFERENT	CASH	CROPS	.			
POTATOES	DEPENDS	THE	BEANS	UPON	CASH	CROPS	.		
FARM	CASH	POTATOES	ARE MIXED	POTATOES	OF	DRY	BEANS	.	
FARM	BEANS	ARE MIXED	CHOICE	IN	SYSTEMS	.			
THE	PEANUTS	IN	THE	GENERAL	SUGAR	BEETS	INCLUDE	CASH	CLIMATE
POTATOES	DEPENDS	DIFFERENT	CANE	.					
FINANCING	INCLUDE	SOIL	OF	SUGAR	CROPS	.			
MARKET	UPON	FARM	GENERAL	SOIL	ARE MIXED	DIFFERENT	DRY	PEANUTS	.
CROPS	IN	DIFFERENT	DRY	POTATOES	DEPENDS	THE	FINANCING	.	
CHOICE	IN	DIFFERENT	POTATOES	INCLUDE	IN	RICE	.		
DIFFERENT	CROPS	UPON	THE	CLIMATE	DEPENDS	OF	GENERAL	RICE	.
SOIL	DEPENDS	BEETS	.						
CANE	DEPENDS	FINANCING	.						
FARM	RICE	OF	DIFFERENT	CROPS	DEPENDS	GENERAL	FINANCING	UPON	PEANUTS
DRY	GENERAL	SYSTEMS	INCLUDE	GENERAL	RICE	.			
SUGAR	CHOICE	INCLUDE	MARKET	.					
BEANS	ARE MIXED	THE	SUGAR	MARKET	.				
CHOICE	DEPENDS	MARKET	.						
GENERAL	SYSTEMS	IN	GENERAL	PEANUTS	DEPENDS	IN	CROPS	.	
GENERAL	CLIMATE	DEPENDS	THE	BEETS	.				
RICE	INCLUDE	OF	DIFFERENT	DRY	CANE	.			
CROPS	DEPENDS	DIFFERENT	CHOICE	.					
SUGAR	MARKET	DEPENDS	DIFFERENT	BEANS	.				
SUGAR	PEANUTS	INCLUDE	DIFFERENT	GENERAL	PEANUTS	.			
MARKET	INCLUDE	CANE	.						
CHOICE	ARE MIXED	TOBACCO	.						
DIFFERENT	PEANUTS	ARE MIXED	CROPS	.					
THE	BEETS	ARE MIXED	TOBACCO	OF	THE	POTATOES	.		
SYSTEMS	DEPENDS	THE	FINANCING	OF	SUGAR	CASH	RICE	.	
THE	PEANUTS	INCLUDE	OF	BEETS	.				
FARM	POTATOES	OF	FINANCING	ARE MIXED	DIFFERENT	FARM	CANE	UPON	CROPS
BEANS	ARE MIXED	GENERAL	CANE	IN	FARM	BEANS	.		
FINANCING	INCLUDE	DIFFERENT	FARM	CASH	FARM	PEANUTS	UPON	THE	MARKET
TOBACCO	ARE MIXED	THE	CROPS	.					
RICE	INCLUDE	UPON	CLIMATE	.					
FINANCING	OF	GENERAL	SUGAR	POTATOES	INCLUDE	CANE	.		
DRY	SYSTEMS	OF	SYSTEMS	DEPENDS	FINANCING	OF	DIFFERENT	PEANUTS	.
CROPS	INCLUDE	RICE	.						
RICE	DEPENDS	IN	DIFFERENT	GENERAL	SOIL	.			
TOBACCO	DEPENDS	POTATOES	.						
THE	CROPS	ARE MIXED	IN	DRY	CANE	.			
FARM	BEANS	INCLUDE	GENERAL	CASH	DRY	FARM	CLIMATE	.	
CANE	INCLUDE	UPON	SUGAR	GENERAL	SUGAR	POTATOES	.		
THE	TOBACCO	ARE MIXED	THE	GENERAL	SUGAR	MARKET	.		
CASH	FARM	SOIL	OF	THE	MARKET	DEPENDS	FARM	SYSTEMS	IN
SYSTEMS	.								
DIFFERENT	CASH	SUGAR	MARKET	UPON	CROPS	ARE MIXED	OF	DIFFERENT	MARKET
DIFFERENT	POTATOES	DEPENDS	THE	SOIL	.				
CROPS	UPON	CASH	BEETS	INCLUDE	THE	CASH	CROPS	.	
THE	DRY	BEETS	ARE MIXED	OF	CASH	CLIMATE	.		
CANE	INCLUDE	SYSTEMS	.						
SOIL	DEPENDS	THE	CANE	.					
THE	BEANS	DEPENDS	MARKET	.					
DIFFERENT	SUGAR	CASH	CANE	IN	GENERAL	SUGAR	FINANCING	DEPENDS	SOIL
OF	BEETS	.							
GENERAL	CASH	BEETS	IN	DIFFERENT	DRY	CROPS	INCLUDE	CASH	SUGAR
FINANCING	OF	THE	MARKET	.					
DIFFERENT	FARM	GENERAL	RICE	OF	THE	FARM	SOIL	INCLUDE	DRY
SYSTEMS	.								
PEANUTS	ARE MIXED	IN	THE	SUGAR	CROPS	.			
FARM	GENERAL	CHOICE	IN	FINANCING	INCLUDE	TOBACCO	IN	THE	SUGAR
BEANS	.								
CROPS	IN	BEETS	DEPENDS	BEETS	.				
THE	MARKET	DEPENDS	RICE	OF	CASH	FINANCING	.		
DIFFERENT	SOIL	IN	DIFFERENT	SOIL	ARE MIXED	GENERAL	RICE	.	
DIFFERENT	CLIMATE	INCLUDE	THE	SOIL	.				
THE	FINANCING	ARE MIXED	DIFFERENT	TOBACCO	OF	THE	SYSTEMS	.	
DIFFERENT	DRY	FINANCING	INCLUDE	THE	CASH	GENERAL	SOIL	.	
THE	CASH	CANE	UPON	RICE	INCLUDE	CROPS	.		

of the source text is placed in the vocabulary pool of the phrase structure nonsense generator. The process of generation is then initiated. Each time an English word is selected, the monitoring system checks to see if the implied dependency relations in the sentence being generated match the dependency relations in the source text. When no match is observed, the monitoring system either selects a new word or aborts the process of generation and starts over.

One of the requirements for matching is that the dependencies must refer to particular tokens of words. For example, given a text such as:

"The man works in a store.
The man sleeps in a bed."

if "in" is determined to be dependent on "works," it is only the token of "in" in the first sentence that is dependent on "works." Similarly, having selected this particular token of "in," it is only "store" that is dependent on it. In the second sentence "bed" is dependent on another token of "in." Were it not for this restriction it would be possible to generate sentences such as "The man works in a bed."

The phrase structure generator in this system differs from the type described in the preceding section in one important way: the English vocabulary items are chosen as soon as a class name appears, regardless of subscript value. This permits a hierarchical selection of English words, i.e., the heads of constructions are selected first. Also, the generation process builds a tree; by selecting English words immediately, the words whose dependency relations are to be monitored are always at adjacent nodes in the tree when the monitoring takes place. If the English words are selected at a later time the problem of determining dependency becomes more complex, especially if the words involved in a direct dependency relation have become separated by other items.

For example:

S

$N_2 + V_2$

$N_2 + V_2$
Cats eat

$Adj_0 + N_1 + V_2$
Tall cats eat

$Adj_0 + Adj_0 + N_1 + V_2$
Tall black cats eat

Note that "tall" is no longer adjacent to "cats."

$Adj_0 + Adj_0 + N_0 + V_2$
Tall black cats eat

Because the English word has already been selected, a zero subscript means only that this item can be expanded no further.

$Adj_0 + Adj_0 + N_0 + V_1 + N_2$
Tall black cats eat fish

$Adj_0 + Adj_0 + N_0 + V_0 + N_2$
Tall black cats eat fish

$Adj_0 + Adj_0 + N_0 + V_0 + Adj_0 + N_1$
Tall black cats eat stale fish

etc.

Note the separation of "eat" and "fish" which are in a dependency relation.

This example should make it clear that the monitoring of dependencies is greatly facilitated if the English words are chosen as early as possible. There is also another advantage to early selection. It permits the determination of heads of phrases before attributes. Note in the preceding example that the main subject and verb of the sentence were selected first. In a system which generates randomly, this yields a faster computer program. Consider an alternative. If one were to start with

$Adj_0 + Adj_0 + N_0 + V_0 + N_2$
Tall black cats

and be unable to find a verb dependent on "cats," the entire sentence would have to be thrown out. Then the computation involved in picking adjectives dependent on "cats" would have been wasted.

Detailed Analysis of the Generation Process

The best way to explain the process of generation is to examine a sentence actually generated by the computer, along with its history of derivation which was also a computer output. The experiment on which this paper is based used as an input the following segment of text from page 67 of Compton's Encyclopedia[2] which was hand analyzed in terms of dependency relations,

> "Different cash crops are mixed in the general farm systems. They include tobacco, potatoes, sugar beets, dry beans, peanuts, rice, and sugar cane. The choice of one or more depends upon climate, soil, market. . . , and financing."

The word "opportunities" occurred after market in the original text and was deleted because it contained more than 12 letters. This deletion results from a format limitation of a trivial nature; it can easily be overcome although it was not thought necessary to do so in the pilot experiment.

The text analyzed in terms of dependency is contained in Table 2. The vocabulary of the phrase structure nonsense generator, sorted according to grammatical category, is contained in Table 3. The grammar rules listed in the format of their weighted probability of selection are contained in Table 4. Each class of rule—noun phrase rule, verb phrase rule, etc.—has ten slots allotted to it. Probability weighting was achieved

by selected repetitions of various rules. Inspection of Table 4 will show the frequency with which each rule is represented.

Consider the generation of an actual sentence as accomplished by a computer. The program starts each sentence generation with:

$N_4 + V_4$ as the sentence type.

In our example,

N_4 + V_4
Choice

a verb dependent on "choice" is now selected.

TABLE 2

DEPENDENCY ANALYSIS OF SOURCE TEXT

Sequence	*Word*	*Dependency*
0	.	0
1	Different	3
2	cash	3
3	crops	5, 12
4	are	
5	mixed	5
6	in	3, 5
7	the	10
8	general	10
9	farm	10
10	systems	6
11	.	11
12	They	3, 12, 34, 36
13	include	12
14	tobacco	13
15	,	15
16	potatoes	13
17	,	17
18	sugar	19
19	beets	13
20	,	20
21	dry	22
22	beans	13
23	,	23
24	peanuts	13
25	,	25
26	rice	13
27	and	27
28	sugar	29
29	cane	13
30	.	30
31	The	32
32	choice	33, 32
33	of	32, 34, 36
34	one	33, 12
35	or	35
36	more	33, 12
37	depends	32
38	upon	37
39	climate	38
40	,	40
41	soil	38
42	,	42
43	market	38
44	,	44
45	and	45
46	financing	38
47	.	47

TABLE 3

VOCABULARY POOL

Token	*Word Class*
CANE CHOICE CLIMATE SOIL MARKET FINANCING BEANS PEANUTS BEETS CROPS SYSTEMS TOBACCO RICE POTATOES	N (noun)
ARE MIXED INCLUDE DEPENDS	V (verb)
CASH GENERAL FARM DRY SUGAR	ADJ (adjective)
IN OF UPON	PREP (preposition)
THE DIFFERENT	ART (article)

N_4 + V_4
Choice include

N_2 + Mod_1 + V_4
Choice include

The N_4 of the preceding step was expanded by selection of rule (3) Table 4.

N_0 + Mod_1 + V_4
Choice include

By selection of rule 4, Table 4. Now the Mod_1 remains to be expanded since it is the leftmost entity with a subscript greater than zero.

N_0 + $Prep_0$ + N_2 + V_4
Choice include

N_0 + $Prep_0$ + N_2 + V_4
Choice in include

"In" is dependent on choice.

N_0 + $Prep_0$ + N_2 + V_4
Choice in systems include

"Systems" is dependent on "in."

N_0 + $Prep_0$ + N_0 + V_4
Choice in systems include

TABLE 4

GENERATION GRAMMAR RULES

Rule No.	*Formula*	*Rule No.*	*Formula*	*Rule No.*	*Formula*
1	$N_2 = ART_0 + N_1$	5	$V_2 = V_1 + N_2$	8	$MOD_1 = PREP_0 + N_2$
1	$N_2 = ART_0 + N_1$	5	$V_2 = V_1 + N_2$	8	$MOD_1 = PREP_0 + N_2$
1	$N_2 = ART_0 + N_1$	6	$V_3 = V_2 + MOD_1$	8	$MOD_1 = PREP_0 + N_2$
2	$N_1 = ADJ_0 + N_1$	6	$V_3 = V_2 + MOD_1$	8	$MOD_1 = PREP_0 + N_2$
2	$N_1 = ADJ_0 + N_1$	6	$V_3 = V_2 + MOD_1$	8	$MOD_1 = PREP_0 + N_2$
3	$N_3 = N_2 + MOD_1$	7	$V_1 = V_0$	8	$MOD_1 = PREP_0 + N_2$
3	$N_3 = N_2 + MOD_1$	7	$V_1 = V_0$	8	$MOD_1 = PREP_0 + N_2$
4	$N_1 = N_0$	7	$V_1 = V_0$	9	$S_1 = N_4 + V_4$
4	$N_1 = N_0$	8	$MOD_1 = PREP_0 + N_2$	9	$S_1 = N_4 + V_4$
4	$N_1 = N_0$	8	$MOD_1 = PREP_0 + N_2$	9	$S_1 = N_4 + V_4$
5	$V_2 = V_1 + N_2$	8	$MOD_1 = PREP_0 + N_2$	9	$S_1 = N_4 + V_4$
5	$V_2 = V_1 + N_2$			9	$S_1 = N_4 + V_4$

N_2 reduced to N_0 by rule (4) Table 4. The V_4 now remains to be expanded.

N_0 + $Prep_0$ + N_0 +
Choice in systems
V_1 + N_2
include

N_0 + $Prep_0$ + N_0 +
Choice in systems
V_1 + N_2
include cane

"Cane" is dependent on "include".

N_0 + $Prep_0$ + N_0 +
Choice in systems
V_0 + N_0
include cane

And finally (two steps involved here) all zero subscripts by rules 4 and 7 of Table 4.

Table 5 contains 102 sentences generated by the coherent discourse generator using as input the analyzed paragraph (Table 2). For comparison Table 1 contains the output of the same system, except for the dependency monitoring routine.

Comments on Linguistic Methodology of the Program

The grammar used in the pilot model of this program is an extremely simple one. The parts of speech (Table 3) include only article, noun, verb, adjective, and preposition. The grammatical rules used are a tiny subset of those necessary to account for all of English. Accordingly, the hand analysis of the vocabulary into parts of speech required a number of forced choices. Namely, "different" was classified as an article, and "farm" and "sugar" were classified as adjectives. A larger grammar including several adjective classes would permit more adequate classification.

An interesting strategy has been developed for the handling of intransitive verbs. As noted above, the system does not distinguish between transitive and intransitive verbs. The running program demands that an attempt be made to find a direct or indirect object for every verb. Only if no such object is found to be dependent on the verb in the source text is the generation of a verb with no object permitted. In effect, a bit of linguistic analysis involving the source text is done at the time of generation.

A system to automatically perform dependency analysis on unedited text is currently being developed. Part of it (a system which performs a phrase structure analysis of English text) is completely programmed and operative on the IBM 7090. A system for converting the phrase structure analysis into a dependency analysis is currently being programmed.

Theoretical Discussion

What is the significance of the transitivity of dependency? Granted, our rules for manipulating dependency to generate coherent discourse can be viewed as a clever engineering technique. However, we feel that the transitive nature of dependency is of greater theoretical significance. A language is a model of reality. To the extent that it is a good model, its speakers are able to manipulate it in order to draw valid inferences about reality. The rules of dependency are part of a model of language, a model which is in turn a second-order model of reality. The value of any model is determined by the truth of its predictions. In this case the value of our transitive dependency model is determined by the validity of the output of the coherent discourse generator in terms of its input.

If we have uncovered some of the mechanisms involved in the logical syntax of English,* then dependency is a primitive for our model of that system, and the rules about its transitivity and intransitivity are axioms. Whether or not concepts of transitive dependency might be important components in logical-syntactic models of other languages can be tested easily.

* We have made no attempt to deal with conditionals or negation in the present experiment.

TABLE 5

COMPUTER-GENERATED COHERENT DISCOURSE

CASH CROPS IN THE FARM GENERAL FARM SYSTEMS DEPENDS UPON CLIMATE .
CROPS INCLUDE TOBACCO .
DIFFERENT CASH CROPS INCLUDE TOBACCO .
DIFFERENT CASH CROPS DEPENDS UPON MARKET .
CROPS DEPENDS UPON MARKET . .
CROPS IN SYSTEMS DEPENDS UPON SOIL .
CHOICE INCLUDE TOBACCO .
CHOICE INCLUDE POTATOES .
CROPS DEPENDS UPON MARKET .
CROPS DEPENDS UPON MARKET .
DIFFERENT CROPS INCLUDE BEETS .
CHOICE DEPENDS UPON CLIMATE .
CHOICE DEPENDS UPON SOIL .
DIFFERENT CASH CHOICE IN GENERAL SYSTEMS DEPENDS UPON CLIMATE .
CASH CHOICE INCLUDE TOBACCO .
CHOICE INCLUDE PEANUTS .
CHOICE INCLUDE BEETS .
DIFFERENT CHOICE DEPENDS UPON MARKET .
CHOICE DEPENDS UPON SOIL .
CASH CHOICE DEPENDS UPON FINANCING .
CROPS IN THE SYSTEMS INCLUDE RICE .
DIFFERENT CROPS DEPENDS UPON MARKET .
THE CASH CROPS IN SYSTEMS INCLUDE BEETS .
CASH CROPS DEPENDS UPON FINANCING .
CROPS INCLUDE SUGAR BEETS .
DIFFERENT CHOICE INCLUDE SUGAR CANE .
CHOICE INCLUDE RICE .
CROPS INCLUDE PEANUTS .
CASH CROPS DEPENDS UPON MARKET .
CASH CHOICE INCLUDE BEETS .
DIFFERENT CHOICE INCLUDE SUGAR CANE .
CASH CHOICE INCLUDE BEETS .
THE CASH CHOICE IN THE SYSTEMS DEPENDS UPON FINANCING .
CROPS IN SYSTEMS INCLUDE SUGAR CANE .
CHOICE DEPENDS UPON MARKET .
THE CHOICE DEPENDS UPON MARKET .
CASH CHOICE INCLUDE PEANUTS .
CASH CROPS DEPENDS UPON SOIL .
DIFFERENT CROPS INCLUDE CANE .
CHOICE INCLUDE TOBACCO .
CHOICE INCLUDE CANE .
CROPS IN THE SYSTEMS DEPENDS UPON CLIMATE .
CHOICE IN SYSTEMS DEPENDS UPON FINANCING .
CROPS DEPENDS UPON SOIL .
THE CASH CROPS DEPENDS UPON SOIL .
DIFFERENT CHOICE IN SYSTEMS DEPENDS UPON FINANCING .
DIFFERENT CROPS INCLUDE DRY BEANS .
CHOICE INCLUDE SUGAR CANE .
CASH CHOICE INCLUDE POTATOES .
DIFFERENT CASH CHOICE INCLUDE SUGAR CANE .
THE CASH CROPS DEPENDS UPON FINANCING .
DIFFERENT CHOICE INCLUDE PEANUTS .
CHOICE DEPENDS UPON CLIMATE .
CHOICE IN SYSTEMS INCLUDE CANE .
CASH CHOICE DEPENDS UPON MARKET .
CASH CHOICE IN GENERAL SYSTEMS INCLUDE DRY BEANS .
CROPS DEPENDS UPON CLIMATE .
THE CASH CROPS DEPENDS UPON CLIMATE .
CASH CHOICE INCLUDE RICE .
CHOICE INCLUDE SUGAR CANE .
CASH CROPS DEPENDS UPON MARKET .
CHOICE IN THE SYSTEMS DEPENDS UPON CLIMATE .
CHOICE DEPENDS UPON MARKET .
DIFFERENT CROPS DEPENDS UPON CLIMATE .
CASH CHOICE INCLUDE POTATOES .
CHOICE DEPENDS UPON SOIL .
DIFFERENT CASH CHOICE INCLUDE PEANUTS .
CROPS IN THE FARM GENERAL FARM SYSTEMS DEPENDS UPON CLIMATE
CASH CHOICE INCLUDE POTATOES .
THE CROPS INCLUDE RICE .
CROPS DEPENDS UPON FINANCING .
CROPS INCLUDE TOBACCO .
CROPS DEPENDS UPON SOIL .
CHOICE INCLUDE PEANUTS .
THE CASH CROPS INCLUDE PEANUTS .
CHOICE DEPENDS UPON SOIL .
CHOICE DEPENDS UPON CLIMATE .
CASH CROPS IN SYSTEMS INCLUDE BEANS .
CASH CHOICE IN THE FARM GENERAL SYSTEMS INCLUDE CANE .
CHOICE DEPENDS UPON FINANCING .
THE CHOICE DEPENDS UPON FINANCING .
CROPS DEPENDS UPON FINANCING .
THE CHOICE DEPENDS UPON CLIMATE .
DIFFERENT CHOICE DEPENDS UPON MARKET .
CASH CHOICE INCLUDE DRY BEANS .
CROPS DEPENDS UPON MARKET .
CHOICE DEPENDS UPON MARKET .
CROPS DEPENDS UPON CLIMATE .
THE CHOICE INCLUDE TOBACCO .
CHOICE INCLUDE BEETS .
CHOICE DEPENDS UPON SOIL .
DIFFERENT CHOICE INCLUDE CANE .
CASH CROPS INCLUDE BEETS .
CASH CHOICE DEPENDS UPON FINANCING .
THE CROPS IN SYSTEMS DEPENDS UPON CLIMATE .
CASH CHOICE DEPENDS UPON CLIMATE .
CASH CHOICE DEPENDS UPON MARKET .
CASH CROPS IN SYSTEMS DEPENDS UPON MARKET .
CASH CHOICE INCLUDE RICE .
CROPS IN GENERAL FARM GENERAL FARM GENERAL SYSTEMS DEPENDS UPON MARKET .
THE CASH CROPS DEPENDS UPON CLIMATE .
CROPS DEPENDS UPON SOIL .

One has only to conduct new experiments in the generation of coherent discourse.

Our coherent discourse generation experiment has interesting implications for transformational theory[5, 1]. The experiment involved control of co-occurrence; that is, the vocabulary of the output was limited to the vocabulary of the source text. It was demanded that pertinent transitive or intransitive dependency relations be held constant from source to output. The fact that the output sentences of Table 5 look like a set that might have been derived from the source text (page 56) by a series of truth-preserving transformations, suggests that dependency, in its transitive and intransitive aspects, is an invariant under a large number of transformations.

Also of great importance is the fact that these sentences were produced without the use of a list of transformations.* The implication here is that the coherent discourse generator contains a decision procedure for determining whether a sentence could have been derived from source text by an appropriate choice of transformations.

One might also note in passing that an automatic kernelizer would not be a difficult application of the principles involved. What is necessary is to adjust the sentence pattern rules of the coherent discourse generator so that only kernel type sentences can be generated. Inspection of Table 5 will reveal that a number of kernels derived from the source text have indeed been generated.

With respect to the Stratificational Theory of Language as propounded by Sydney Lamb[8], our rules of transitive dependency permit the isolation of syntactic synonymy. It would seem that given control over co-occurrence of morphemes and control over syntactic synonymy, one has control over remaining sememic co-occurrence. This would suggest that our rules provide a decision procedure for determining the co-occurrence of sememes between one discourse and another, without need for recourse to elaborate dictionaries of sememes and sememic rules.

Potential Applications

The principles of transitive dependency and of syntactic synonymy lend themselves very readily to a number of useful language processing applications. Among these are the recognition of answers to questions, a computer essay writing system, and some improvements in automatic abstracting.

QUESTION ANSWERING

Given an English question and a set of statements some of which include answers and some of which do not, the dependency logic is very helpful in eliminating statements which are not answers. The logic involved is similar to that used in the part of the coherent discourse generator which rejects sentences whose dependency relations are not in harmony with those in a source text. In the case of a question answering system, the question is treated as the source text. Instead of generating sentences for comparison of dependencies, a question answering system would inspect statements offered to it as potential answers, and reject those with dependencies whose inconsistencies with those of the question fall above a minimum threshold.

The primary set of potential answers might be selected through statistical criteria which would insure presence of terms which also occurred in the question. This set would then be subjected to analysis by a dependency comparison system. Such an approach is used in the protosynthex question answering system which is currently being programmed, and is partly operative on the IBM 7090.[7, 11, 12]

For an example of this application, consider the question in Table 6 and some potential answers. Each of the potential answering sentences was selected to contain almost all of the words in the question. In the first potential answer, Answer 1, "cash" is dependent on "crops"; "are" is equivalent to "include" and is dependent on "crops"; "bean" is dependent on "are"; and "soy" is dependent on "bean." Thus the potential answer matches the question in every dependency link and, for this type of question, can be known to be an answer. The second example, Answer 2, also matches the question on every dependency pair and is also an answer.

In Answer 3, the importance of some of the rules which limit transitivity can be seen. For example, transitivity across a verb is not allowed. Thus "beans" is dependent on "eat" which is dependent on "people" which is dependent on "includes." Because "eat" is a verb, the dependency chain is broken and the match with the question fails. In a similar manner "cash" in the same sentence would be transitively dependent on "crops" via "bring," "to," and "fails" except that the chain breaks at "bring." In Answer 3, every dependency link of the question fails and the statement can unequivocally be rejected as a possible answer even though it contains all the words of the question.

In general, for question answering purposes, the matching of dependencies between question and answer results in a score. The higher this score value the greater the probability that the statement contains an answer.

AUTOMATIC ESSAY WRITING

A system to generate essays on a computer might make use of a question answering system and a coherent discourse generator. The input to such a system would be a detailed outline of some subject. Each topic in the

* One transformation, however, was used implicitly, in that passive construction dependencies were determined as if the construction had been converted to an active one.

TABLE 6

DEPENDENCIES OF A QUESTION AND SOME ANSWERS

QUESTION

Sequence		*Dependency*
1	Do	
2	cash	3
3	crops	3
4	include	3
5	soy	6
6	beans	4

ANSWER 1

Sequence		*Dependency*
1	Soy	2
2	beans	2,3
3	are	2,6,7
4	a	6
5	cash	6
6	crop	3
7	in	6,3
8	the	9
9	South	7

ANSWER 2

Sequence		*Dependency*
1	Cash	2
2	crops	2
3	include	2
4	peanuts	3
5	and	5
6	soy	7
7	beans	3

ANSWER 3

Sequence		*Dependency*
1	The	2
2	state	2
3	includes	2
4	people	3,5
5	who	4
6	eat	5
7	soy	8
8	beans	6
9	when	6
10	the	12
11	peanut	12
12	crop	9
13	fails	12
14	to	13
15	bring	14
16	a	18
17	cash	18
18	return	15

outline would then be treated as if it were a retrieval question. This would mean that statements pertinent to the outline topic would be retrieved from a reference corpus. The vocabulary of the retrieved statements would then be placed in the vocabulary pool of the coherent discourse generator. Those words which were common to both the outline topic and the retrieved statements might be weighted so as to maximize the probability of their occurrence in the sentences that would be generated.

After a number of paraphrases had been produced, the one or ones which had the maximum number of words in common with those in the outline topic would be printed out as the computer-generated expansion for that topic. The process would continue for all the topics in the outline. The output would then be some form of essay—an essay with fuzzy transitions between ideas and perhaps occasional irrelevancies, but nevertheless a computer-generated essay.

AUTOMATED ABSTRACTING

A procedure similar to the one outlined for the computer production of essays might be expected to offer considerable improvement over the present state of automatic abstracting or extracting as discussed by Luhn[9] and Edmundson[3] and others. One approach might take a set of the most pertinent typical words represented in the article as estimated by frequency counting techniques and use a coherent discourse generator to construct a set of sentences, each of which included a maximal set of those words. The same process would be followed on the words which had not yet been built into sentences, until the list of words was exhausted. Since only dependencies present in the original article would be allowed, the resulting abstracted paraphrase could not deviate from the relationships set up by the author.

Another approach might impress on each article a frame of words pertinent to a particular literature searcher. For example, a chemist looking at an article would be interested in those statements that included chemical terms and chemical relationships. A physician looking at the same article would be interested in a medical vocabulary. By extracting from the article those words which matched the vocabulary of a particular discipline and weaving these into a set of sentences, we would expect to construct meaningful, specialized abstracts.

Summary and Conclusions

In this paper, the hypothesis that the dependency relationship between words is largely transitive has been supported by an experiment in producing coherent discourse on a computer. The experiment showed that dependency is *not* always a transitive relationship; some significant exceptions include the lack of transitivity across verbs, prepositions and many subordinating conjunctions. If these exceptions were ignored, the discourse generator could produce statements that were not truth-preserving derivations from the original text.

The rules of dependency and transitivity of dependence may be considered as the beginnings of a

transitive dependency model of English. Experiment has supported this model to the extent that the use of such rules allows a computer to automatically generate coherent discourse. Other experiments in answering questions, generating essays and automatic abstracts have been planned and begun in order to further develop the model and to continue testing the hypothesis of transitive dependencies.

The question of the validity of transitive dependency models for other languages needs also to be considered. Since the technique for dependency analysis originated in Europe and is currently finding use on at least one French language project and several efforts at Russian translation, it is reasonable to suspect that transitivity is as important in these languages as it appears to be for English. An important linguistic question whose answer depends on many more experiments is "To what extent is transitive dependency a feature of the logical syntax of all human languages?"

Received October 8, 1962

References

1. Chomsky, N. Syntactic Structures. 's-Gravenhage:Mouton and Co., 1957.
2. Compton's Encyclopedia—F. E. Compton & Co., Chicago, Vol. 1, 1962.
3. Edmundson, H. P., Oswald, V. A., Jr., and Wyllys, R. E. Automatic Indexing and Abstracting of the Contents of Documents, PRC-126 ASTIA AD No. 231606. Los Angeles: Planning Research Corporation, 1959.
4. Harris, Zellig. From Morpheme to Utterance. *Language,* Vol. 22, pp. 161-183, 1946.
5. Harris, Zellig. Co-occurrence and Transformation in Linguistic Structure. *Language,* Vol. 33, No. 3, pp. 283-340, 1957.
6. Hays, D. G. Studies in Machine Translation—10: Russian Sentence-Structure Determination, RM-2538. The RAND Corporation, 1960.
7. Klein, S., and Simmons, R. F. A Computational Approach to Grammatical Coding of English Words. SDC document SP-701. 25 pp., February 22, 1962.
8. Lamb, Sydney M. Outline of Stratificational Grammar. University of California, Berkeley, 1962. (mimeo)
9. Luhn, H. P. The Automatic Creation of Literature Abstracts. *IBM Journal of Research and Development,* Vol. 2(2), pp. 159-165, 1958.
10. Nagel, E. and Newman, J. R. Gödel's Proof. New York University Press, 1958.
11. Simmons, R. F., Klein, S., and McConlogue, K. Toward the Synthesis of Human Language Behavior. *Behavioral Science,* Vol. 7, No. 3, pp. 402-407, 1962.
12. Simmons, R. F., and McConlogue, K. Maximum-depth Indexing for Computer Retrieval of English Language Data. SDC document SP-775. 22 pp., 1962. *American Documentation* (in press), January 1963.
13. Yngve, Victor H. A Model and an Hypothesis for Language Structure. *Proceedings of the American Philosophical Society,* Vol. 104, No. 5, pp. 444-466, 1960.
14. Yngve, Victor H. Random Generation of English Sentences. *Proceedings of the 1961 Conference on Machine Translation of Languages and Applied Language Analysis,* National Physical Laboratory Symposium No. 13, pp. 66-80, 1962.