Reprinted from LANGUAGE
Vol. 41, No. 4, October–December, 1965
*Printed in U.S.A.*

# CONTROL OF STYLE WITH A GENERATIVE GRAMMAR[1]

SHELDON KLEIN
*Carnegie Institute of Technology; System Development Corporation*

**1.** INTRODUCTION. In the course of building an automatic essay-paraphrasing computer program[2] it became apparent to me that certain quantitative features of style could be controlled in the paraphrasing process. This paper presents the results of several experiments in the control of two quantitative measures of style, frequency of occurrence of syntactic structures and frequency of occurrence of pronouns. The results suggest that any well-defined quantitative measure of style is subject to machine control.

The complete paraphrasing system performs both an immediate-constituency and a dependency analysis of an input text and uses these to monitor the output of a generation grammar as part of the task of producing an essay-type paraphrase of the source text. The generation grammar chooses randomly among immediate constituency rules. Syntactic style of the output is controlled by weighting the probability of selecting a given rule. If the weight assigned to each generative rule is made a function of the frequency of application of that rule in the parsing of a source text, then the paraphrase may manifest a frequency of occurrence of syntactic structures similar to that of the source. Vocabulary frequency can be controlled in a like manner.

A detailed description of the operation of the computer system that serves as a basis for the style control experiments is published elsewhere.[3] §2 of this paper provides an abbrevaited explanation of the concepts underlying the system's basic components. The program is written in JOVIAL, an ALGOL language; it is at present operational on the Philco 2000 computer and potentially operational on any computer for which a JOVIAL compiler exists.

I have deliberately kept the level of description relatively free of considerations of hardware and program details. The language in which the program is written is very much like mathematical logic, and is almost independent of the machine. While early writers on computational linguistics may have felt compelled to state the intimate details of coding and hardware, I feel that such matters not only are unnecessary, but hinder the use of the system by other researchers. Accordingly, I hope that the level of presentation in this paper is such that the basic concepts can readily be transferred to programs written in any language for any computer.

**2.** AUTOMATIC DISCOURSE-PARAPHRASING SYSTEM. The essay-paraphrasing system contains a number of components embodying concepts that require some explanation. §2.1 first describes a phrase-structure parsing system, and then a

[1] This research is supported in part by Public Health Service Research Grant MH 07722, from the National Institute of Mental Health (to the Carnegie Institute of Technology).

[2] Sheldon Klein, 'Automatic paraphrasing in essay format', *Mechanical translation* 8:3/4.68–83 (1965).

[3] Ibid.

1. $Art_0 + N_1 = N_2$
2. $Adj_0 + N_1 = N_1$
3. $V_1 + N_2 = V_2$
4. $N_2 + V_2 = S_1$

TABLE 1. ILLUSTRATIVE PHRASE-STRUCTURE GRAMMAR

system for performing a simultaneous phrase-structure and dependency analysis. §2.2 describes the use of dependency information in generating sentence paraphrases of an input text. §2.3 describes the use of the components in a larger system which produces paraphrases in an essay format.

RUCTURE PARSER. A phrase-structure analysis
d with the application of appropriate rules in
Table 1. The same type of rule can be modified
as well. The relation between dependency and
language has been described in the work of
.[5] Most automatic parsing systems using binary
ltiple analyses for a given sentence. The routine
ussion arbitrarily yields only one analysis per
ave subscripts that are used both to determine
es and to avoid some of the problems created by
sentence. This use of subscripts is derived from
iig Harris.[6] A design for a parsing system using
found in the work of Victor Yngve.[7]
Table 1 is best described by example. Consider
. Assume that unique grammar codes have been

| $dj_0$ | $N_0$ | $V_0$ | $N_0$ |
|---|---|---|---|
| *all* | *man* | *eats* | *fish* |

odes is searched for among the pairs to the left
Table 1. When a match is found, the system
appropriate elements in the sentence analysis
he rules. The first pair of codes in the sentence,
rules. The next pair, $Adj_0 + N_0$, matches the
subscripts of the sentence codes are zero—less
cordingly, the elements are united to form an

**2.1.** DEPENDENCY PHRASE-ST
of English text may be obtaine
binary format, such as those in
to yield a dependency analysis
phrase-structure descriptions of
David Hays[4] and Haim Gaifman
phrase-structure rules obtain mu
used in the program under disc
sentence. The rules in Table 1 h
the order of application of the rul
recognizing only one analysis per
a notation first suggested by Zel
rules without subscripts is to be

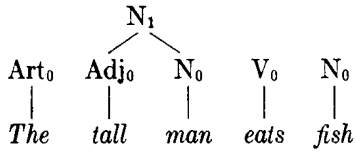
The operation of the rules in
the sentence, *The tall man eats fish*
assigned to each word.

| $Art_0$ | A |
|---|---|
| *The* | *t* |

Each adjacent pair of grammar c
of the equal-signs in the rules of
inquires if the subscripts of the
are less than or equal to those in
$Art_0 + Adj_0$, is not found in the
left half of rule 2, Table 1. The
than those of the table rule. Ac
$N_1$ unit.

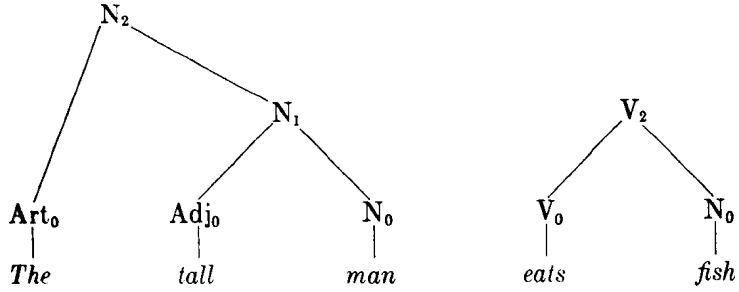
data processing', *Computer applications in the behav-*
ood Cliffs, 1962).
ms and phrase-structure systems', The RAND Cor-
Monica, California, 1961.
utterance', *Lg.* 22.161–83 (1946).
ypothesis for language structure', *Proceedings of the*
4–6 (1960).

[4] D. G. Hays, 'Automatic language
*ioral sciences* (H. Borko, ed.; Englew
[5] H. Gaifman, 'Dependency syste
poration memorandum P-2315, Santa
[6] Z. S. Harris, 'From morpheme to
[7] V. H. Yngve, 'A model and an h
*American Philosophical Society* 104.44

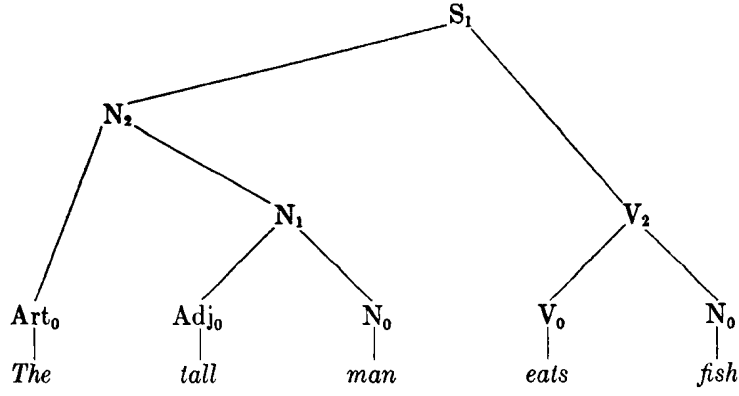



The next pair of elements, $N_0 + V_0$, matches rule 4 of Table 1, yielding $S_1$, or 'sentence'. The grouping is rejected because of criteria having to do with the precedence value of constructions, a feature connected with the value of subscripts when the grammar tags are identical, and determined by definition when they are not.

The pair $V_0 + N_0$ is united by rule 3, yielding a $V_2$ unit. Another pass is now made through the sentence, and the pair $Art_0 + N_1$ is found to match rule 1 of Table 1.




The only remaining uncombined units are $N_2 + V_2$, which match rule 4 of Table 1. The final analysis is as follows.



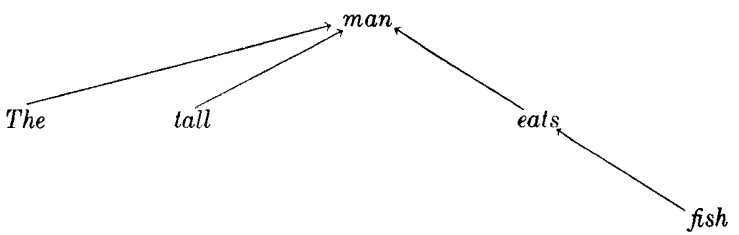

A dependency analysis may also be performed with the same rules slightly modified. The rules in Table 2 are identical with those in Table 1, except that one element in the left half of each rule is prefaced by an asterisk. The unstarred partner of the form is dependent on the starred form. In general, attributes are dependent on heads of constructions. With dependency indicated for each rule, the dependency relations of the total sentence can be computed by cumulatively

1. $Art_0 + *N_1 = N_2$
2. $Adj_0 + *N_1 = N_1$
3. $*V_1 + N_2 = V_2$
4. $*N_2 + V_2 = S_1$

TABLE 2. DEPENDENCY PHRASE-STRUCTURE GRAMMAR

keeping track of the local dependencies as each parsing rule is applied. The program yields as output a dependency tree superimposed on a phrase-structure tree.

The dependency analysis alone is represented in diagram form as follows.




It may also be expressed in a list structure format, which is the way it may be stored in a computer:

| 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| *The* | *tall* | *man* | *eats* | *fish* |
| 2 | 2 | | 2 | 3 |

**2.2.** GENERATION. The use of binary phrase-structure rules to generate grammatically correct nonsense is well known from the work of Chomsky[8] and Yngve.[9] If the output of such a program is monitored so that it contains only the vocabulary of an input text and so that only direct or certain transitive dependency relations are common both to source text and generation output, then the generated sentences will conform to the meaning of the input text.[10] For the purpose of making a single dependency analysis of a whole text, like tokens of the same noun are assigned two-way dependency links.

For example, consider the text, *Dogs are canines. Canines eat meat.* Assume that the vocabulary of a grammatically correct nonsense generator is limited to the vocabulary of this text. Also add the following to the rules of Table 1:

5. $N_1 = N_0$
6. $V_1 = V_0$

Assume that parts of speech are uniquely assigned to the vocabulary and that in the rewrite rules a symbol with a zero subscript may be rewritten with an appropriate lexical item.

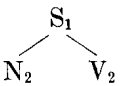
[8] N. Chomsky, *Syntactic structures* ('s-Gravenhage, 1957).

[9] Yngve, op.cit.

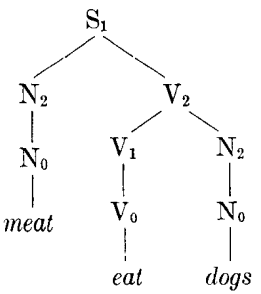
[10] Klein, op.cit.; Klein, 'Automatic decoding of written English' (dissertation, University of California, Berkeley, June 1963); Klein and R. F. Simmons, 'Syntactic dependence and the computer generation of coherent discourse', *Mechanical translation* 7:2.50–61 (1963).

In the course of generation, the use of subscripts to monitor the applicability of rules is the converse of the usage in parsing. If the right half of a grammar rule has the same literal value as a node in the generation tree, its left half may be the expansion of that node only if the node subscript is greater than or equal to the subscript in the right half of the grammar rule. Thus an $N_1$ may be rewritten by a rule with an $N_1$ right half, but not an $N_2$. In the course of generation, leftmost entities are expanded first.

Starting with $S_1$, rule 4 of Table 1 yields

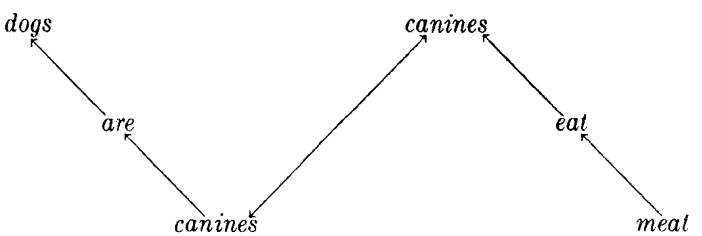



Let the $N_2$ be rewritten $N_0$ after rule 5, and the $V_2$ be rewritten $V_1 + N_2$ after rule 3. If the $V_1$ and $N_2$ are replaced by $V_0$ and $N_0$, and lexical items are chosen, the production might be




This is grammatically correct within the limits of the grammar of Table 1, which does not distinguish number. Other strings that might be produced would include *Canines eat dogs*, *Dogs eat canines*, *Canines eat canines*, and *Dogs eat meat*, as well as the original text.

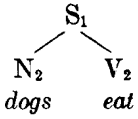
Dependency relations can be used to restrict the output to sentences that conform somewhat to the meaning of the input text. Assume that, in addition to the information already suggested, the text has the following dependency analysis plus the additional information that *are* is in a special verb class, *to be*—a class different from the class of *eat*.



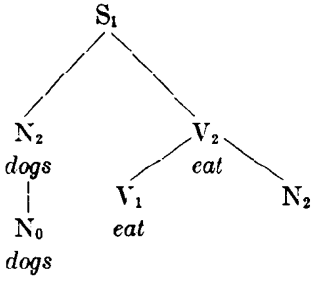

In the generation process, lexical items are now chosen as soon as the head of a new construction is defined. The choice of a lexical item is rejected unless a path between it and its governor can be plotted in the network representing the text.

Dependencies are transitive across nouns and across verbs of the class *to be*. A path is assumed not to exist if an intransitive form is in the path of linkage.

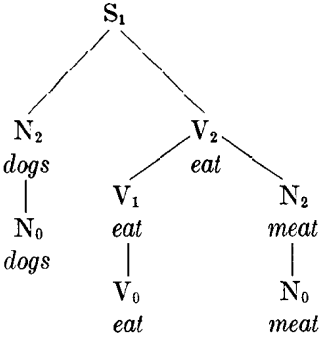
Consider an example. Let $S_1$ be rewritten by $N_2 + V_2$, as in the last example. In this case, however, vocabulary items must be picked at once for the heads of the two new constructions. Assuming *meat* is picked for $N_2$, the program must pick a suitable verb. Neither *eat* nor *are* can be chosen, because there is no path from either of these to *meat*. Since the verb possibilities are exhausted, the system determines that *meat* cannot be the subject of the sentence, and another noun is selected, say, *dogs*. *Eat* can be selected as the verb because a path exists between *eat* and *dogs* that does not cross an intransitive link.




Assume that $N_2$ is rewritten $N_0$ by rule 5. If it had been rewritten $Art_0 + N_1$, the system would search for an article; finding none, it would prune the tree of the $Art_0$ node. Let the $V_2$ be rewritten by rule 3.




It is now necessary to pick a noun dependent on *eat*. Both *dogs* and *canines* would be rejected because no path exists from either of them to *eat*. *Meat*, however, is acceptable and is chosen. Let the subscripts of $V_1$ and $N_2$ be reduced to zero by rules 5 and 6, with the result,




Given appropriate grammar rules, the program can also convert a verb into a participle. If a production has a slot for a participle, but none with appropriate dependencies appears in the source text, the system may select a verb with ap-

propriate dependency relations, converting it to a form in *ing*. Thus, *Dogs eating meat are canines* might be generated if the grammar of Table 2 included rules for a participial phrase.

The success of the coherent paraphrasing system suggests that if the vocabulary and dependency relations in one text are included in another, then the meaning of the first text is more or less included in the meaning of the latter. To the extent that this suggestion is valid, I regard the system as an approximation of a stratificational model of grammar, which makes reference to semantic units in a network. The requirement for cooccurrence of lexical items in both source text and paraphrase makes them serve as approximations to the semantic entities they represent.[11]

**2.3.** PARAPHRASING IN ESSAY FORMAT. §2.2 described a grammatically correct nonsense generator monitored by a program that requires a sentence in generation to preserve the dependency relations in a source text. The essay-paraphrasing system makes use of additional controls on the output of the coherent discourse-generation component to insure that the paraphrases will occur in a particular order and with limited redundancy. It also inserts subject and object pronouns into the output text, even though such forms do not occur in the input text; and it is capable of converting a verb with appropriate dependencies to a form in *ing*. The frequency of occurrence of subject pronouns can be controlled by a program input parameter. Object pronouns are used in mandatory fashion to avoid repetition of nouns.

The system also permits the inclusion of an outline that partially determines the structure of the essay paraphrase. The outline consists of sentences. A program prepares a list of outline nouns in the sequence of their occurrence, plus a list of verbs in sequence, from both the outline and the main text. No noun is placed on the noun list if it is either directly or transitively dependent on a noun occurring before it in the outline.

The essay-paraphrasing program uses the noun list and the verb list in the generation process in major and minor cycles respectively. A major cycle involves a single scan through the noun list. A minor cycle involves a complete scan of the verb list each time a new noun is encountered in the course of a major-cycle pass through the noun list. The system produces a new paragraph for each new minor cycle. For each new paragraph, the system uses a forced choice of a particular noun and, cyclically, all verbs, for the main subject and predicate of possible sentences. If a verb is dependent on the subject noun, a sentence is produced; otherwise, the next verb on the list is selected. Once a verb is used in a successful production, even if it occurs outside the main predicate of a sentence, it is deleted from the verb list. The result is an essay in which each paragraph says everything that can be said about a particular subject. Other essay formats can be incorporated in the program. Again, for details, the reader is referred to the basic document describing this system.[12]

[11] Klein and Simmons, ibid.; D. G. Hays, 'Dependency theory: A formalism and some observations', *Lg.* 40.525 (1964).

[12] Klein, 'Automatic paraphrasing in essay format'.

| | Rule | Weight |
|---|---|---|
| 1. | $Art_0 + N_1 = N_2$ | .1 |
| 2. | $Adj_0 + N_1 = N_1$ | .1 |
| 3. | $N_2 + SbCon_1 = N_3$ | .3 |
| 4. | $N_3 + Mod_1 = N_4$ | .4 |
| 5. | $N_0 = N_1$ | .1 |
| 6. | $V_1 + N_4 = V_2$ | .6 |
| 7. | $V_0 = V_1$ | .4 |
| 8. | $Part_0 + N_3 = Mod_1$ | .9 |
| 9. | $Prep_0 + N_3 = Mod_1$ | .1 |
| 10. | $N_4 + V_4 = S_1$ | 1.0 |
| 11. | $PnRcn_0 + V_2 = SbCn_1$ | 1.0 |

Table 3. Generation grammar

**3.** Control of style. The information supplied to the program in the style-control experiments consisted of a text with unique grammar codes for each lexical item, a grammar of binary phrase-structure dependency rules, and frequency-of-selection weightings for both these rules and the portion of the program determining pronoun usage. The operation of the program beyond this was fully automatic, including the phrase-structure dependency analysis for the total text, and including the construction of the essay format from the sentences in the outline.

The use of random selections is an integral part of the paraphrasing system. The phrase-structure rules, as well as vocabulary items, are selected randomly. Equal probabilities, however, need not be assigned to all items. In fact, the probabilities assigned can control style. For example, if rule 2 of Table 1, $Adj_0 + N_1$, were given an extremely high probability of occurrence, the generated text would be very rich in nouns preceded by many adjectives. Similarly, particular vocabulary items may receive special probability weightings.

One program method of selection control consists of testing whether or not a program-generated random number falls within a certain range. For example, the program may have the option of using a pronoun instead of a noun in a particular construction. A random number is generated within the range from 0 to 99; if the random number is less than some constant in this range, a pronoun is generated. If the constant is 29, there is then a 30% probability that the program will select a pronoun rather than a noun. Other random selection devices are also used.

Table 3 contains generation rules with the frequency weightings pertaining to Output Texts 1–3. The frequencies within each category of rules add up to unity, e.g. the sum of the N weightings is 1. Note that these initial settings of frequency are only indirectly connected with the frequency with which the rules are actually selected. The weightings refer only to the frequency with which the rules are presented for consideration to various parts of the generation program. Some selected rules may be rejected by the program at one stage and accepted at another. For example a rule $N_0 = N_1$ would be rejected unless the use of an $Art_0 + N_1 = N_2$ had already been tried. Special abbreviations in the rules may

need explanation: 'SbCon$_1$' refers to a relative clause, 'Part' refers to a form ending in *ing*, and 'PnRcn' refers to *who* or *which*.

The illustrated Input Text was specially constructed so as to be readily parsed by a simple recognition grammar. Unique grammatical codes for individual words were part of the data, e.g. '*The* (art.) *tall* (adj.) *man* (noun, sg., masc.) *eats* (verb, 3rd pers. sg.) *fish* (noun, sg., neut.)'.[13] Although the grammar rules recognize only one noun class, the program recognizes an additional class whose members are preceded by a plus sign, representing capitalization of the first letter. Such forms are not permitted to take articles or modifying phrases. Because of this not very satisfactory rule, *bentley* was left uncapitalized in both input and output text. Also, *married* was given the ad-hoc classification of noun.

Output Texts 1–3 contain essays generated by a system that was varied only with respect to the parameter controlling subject-pronoun frequency. The essays vary also in other respects, for the altering of any random selection factor affects the sequence in which the program-generated random numbers were presented to the decision-making routines of the program. Note also that only the spacing between paragraphs in the output essays corresponds to the spacing in the actual computer outputs. The actual output as printed resembled the following.

| | | | | |
|---|---|---|---|---|
| THE | POOR | HOUSEWIFE | WHO | WAS |
| A | COUNTESS | IS | HER | . |

The essays occurring in Output Texts 4–6 have a constant 50% factor controlling subject-pronoun selection. Tables 4–6 and Output Texts 4–6 present, in alternation, a set of generation rules with varied selection weightings and a set of essays produced with the rules thus weighted.

INPUT TEXT

Outline:

The poor housewife is Mary. The homewrecker is Helen. The unfaithful husband is John who commits murder. The hero is Peter.

Main text:

John is a successful businessman who works for a corporation. A businessman is a man who likes money. John was a gangster. Peter was a bullfighter. Mary was a countess. Helen was a dancer. A gangster commits crimes. A bullfighter fights bulls. Bulls are dangerous animals. The gangster drives a bentley. The dancer has many admirers. The dancer wears a hat. The countess owns a castle. John secretly loved Helen who was beautiful. Helen who also loved John was married to Peter. John wanted Helen. Helen wanted John. Divorce was impossible. The solution was simple. John killed Peter. Helen killed Mary. The end was happy.

[13] This information was punched on IBM cards; numerical codes represented the grammatical information. The cards were read onto a tape, which served as the input to the program. There were no restrictions on the spacing of the information except that at least one blank space must occur between lexical item and grammar code. The information for a single sentence can be spread over several cards when necessary. A lexical item and its associated grammatical code could appear on different cards without affecting the program.

OUTPUT TEXT 1

Pronoun weighting = .10; phrase-structure rules weighted as in Table 3

The poor housewife who was a countess is her. The poor housewife owns a castle.

A beautiful homewrecker who was a dancer wearing a hat is beautiful Helen. A homewrecker has many admirers. A homewrecker loved John. The beautiful homewrecker was married. A beautiful homewrecker wanted successful John. The homewrecker killed Mary.

An unfaithful husband committing murder is the gangster committing crimes. The husband working is a successful businessman who is a man. The unfaithful husband likes money. The unfaithful husband was a gangster driving a bentley. The husband loved a dancer. He wanted her. The unfaithful husband killed Peter who fights bulls.

The hero is a bullfighter. A hero was him.

OUTPUT TEXT 2

Pronoun weighting = .70; phrase-structure rules weighted as in Table 3

The poor housewife who was a countess is her. The poor housewife owns a castle.

A beautiful homewrecker who was a dancer wearing a hat is beautiful Helen. A homewrecker has many admirers. She loved successful John who wanted beautiful Helen who was married. The beautiful homewrecker who killed Mary wanted a businessman working.

A successful unfaithful husband is John. The unfaithful husband who was a gangster who commits crimes commits murder. He is a businessman who is a man who likes money. He drives a bentley. The husband loved beautiful Helen. He killed Peter.

The hero is him. He was a bullfighter fighting bulls.

OUTPUT TEXT 3

Pronoun weighting = .90; phrase-structure rules weighted as in Table 3

The poor housewife who was a countess is her. She owns a castle.

The beautiful homewrecker who was a dancer wearing a hat is beautiful Helen

| | RULE | WEIGHT |
|---|---|---|
| 1. | $Art_0 + N_1 = N_2$ | .1 |
| 2. | $Adj_0 + N_1 = N_1$ | .1 |
| 3. | $N_2 + SbCon_1 = N_3$ | .1 |
| 4. | $N_2 + Mod_1 = N_4$ | .2 |
| 5. | $N_0 = N_1$ | .5 |
| 6. | $V_1 + N_4 = V_2$ | .6 |
| 7. | $V_0 = V_1$ | .4 |
| 8. | $Part_0 + N_3 = Mod_1$ | .9 |
| 9. | $Prep_0 + N_3 = Mod_1$ | .1 |
| 10. | $N_4 + V_4 = S_1$ | 1.0 |
| 11. | $Pn\ Rcn_0 + V_2 = SbCn_1$ | 1.0 |

TABLE 4. GENERATION GRAMMAR

| | Rule | Weight |
|---|---|---|
| 1. | $Art_0 + N_1 = N_2$ | .5 |
| 2. | $Adj_0 + N_1 = N_1$ | .1 |
| 3. | $N_2 + SbCon_1 = N_3$ | .1 |
| 4. | $N_2 + Mod_1 = N_4$ | .2 |
| 5. | $N_0 = N_1$ | .1 |
| 6. | $V_1 + N_4 = V_2$ | .6 |
| 7. | $V_0 = V_1$ | .4 |
| 8. | $Part_0 = N_3 = Mod_1$ | .9 |
| 9. | $Prep_0 + N_3 = Mod_1$ | .1 |
| 10. | $N_4 + V_4 = S_1$ | 1.0 |
| 11. | $Pn\ Rcn_0 + V_2 = SbCn_1$ | 1.0 |

Table 5. Generation grammar

who has many admirers. She loved successful John. She was married. She wanted John who was a gangster who commits crimes. She killed a countess.

The husband is a gangster driving a bentley. He commits murder. He is a successful businessman working. He is a man who likes money. He loved a dancer. He wanted her. He killed Peter who was a bullfighter who fights bulls.

The hero is Peter.

### Output text 4

Pronoun weighting = .50; phrase-structure rules weighted as in Table 4

A housewife is a countess owning a castle. The poor housewife was a countess.

The homewrecker who has many admirers is beautiful Helen. She was a dancer wearing a hat. She loved successful John who commits murder. She was married. She wanted John. She killed Mary.

The unfaithful husband working is John. The husband is a businessman. The husband liking money is a man. He was a gangster. He commits crimes. The husband who killed Peter drives a bentley. A husband loved Helen. He wanted her.

The hero fighting bulls is Peter. A hero was a bullfighter.

### Output text 5

Pronoun weighting = .50; phrase-structure rules weighted as in Table 5

A housewife is the countess. She was her. She owns a castle.

A homewrecker is a dancer. She was her. She has many admirers. She wears a hat. The homewrecker loved successful John. She was married. She wanted a gangster. The beautiful homewrecker killed Mary.

A husband is successful John. The husband commits murder. He is a successful businessman working. He is a man who likes money. A successful husband com-

| | Rule | Weight |
|---|---|---|
| 1. | $Art_0 + N_1 = N_2$ | .1 |
| 2. | $Adj_0 + N_1 = N_1$ | .1 |
| 3. | $N_2 + SbCon_1 = N_3$ | .3 |
| 4. | $N_2 + Mod_1 = N_4$ | .4 |
| 5. | $N_0 = N_1$ | .1 |
| 6. | $V_1 + N_4 = V_2$ | .6 |
| 7. | $V_0 = V_1$ | .4 |
| 8. | $Part_0 + N_3 = Mod_1$ | .5 |
| 9. | $Prep_0 + N_3 = Mod_1$ | .5 |
| 10. | $N_4 + V_4 = S_1$ | 1.0 |
| 11. | $Pn\ Rcn_0 + V_2 = SbCn_1$ | 1.0 |

Table 6. Generation grammar

mitting crimes was a gangster driving a bentley. He loved Helen. He wanted her. He killed Peter.

A hero is him. He was a bullfighter fighting bulls.

Output text 6

Pronoun weighting = .50; phrase-structure rules weighted as in Table 6

The poor housewife who was a countess is her. The poor housewife owns a castle.

The homewrecker wearing a hat is beautiful Helen. The beautiful homewrecker was a dancer who has many admirers. The beautiful homewrecker loved John who wanted beautiful Helen who was married to Peter who was a bullfighter who fights bulls. The homewrecker wanted successful John who is a businessman working. She killed Mary.

The unfaithful husband committing murder is successful John who killed Peter. The husband who loved a dancer is a man who likes money. He was a gangster who commits crimes. He drives a bentley.

The hero is a bullfighter.

**4.** Discourse generation in the style of an input text. Assuming the existence of an adequate grammar, it should be possible, for almost any quantitative measure of style that can be defined, to produce automatic paraphrases that are stylistically similar to an input text. This might be accomplished by making generation-rule frequency weightings correspond to the frequency of pertinent features in the source text. The successful operation of the system would be complicated by the occasional absence of a direct correspondence between generation-rule weightings and the desired frequency of occurrence of the corresponding constructions in paraphrased texts. This lack of correspondence is attributable to the interdependence among the rules. The relation between input parameters of style and the output style might be predicted by some complex mathematical function to be approximated by trial and error.

The problem of obtaining empirical approximations to functions determining correspondence between input and output style parameters can undoubtedly be solved. However, even if every conceivable quantitative measure of style were

the same for a given input text and its paraphrase, human reviewers might still judge the texts to be of different authorship. If this were to happen, factors not really pertinent to stylistic differences might be identified and eliminated, leading to a new theory of style.

One might also control features of style usually considered nonquantitative by building a paraphraser that incorporates a stratificational model of grammar.[14] Such a system might use the frequency of sememic rather than lexemic entities in the generation process to monitor the use of stylistic attributes, like simile and metaphor.

The system described in this paper works with a very small grammar and with what can only be called a cooked up text. While the suggestions for making output style conform to input style might be implemented within these constraints, a system that would do the same for unrestricted English text must wait for solutions to major problems in the treatment of meaning and syntactic ambiguity.

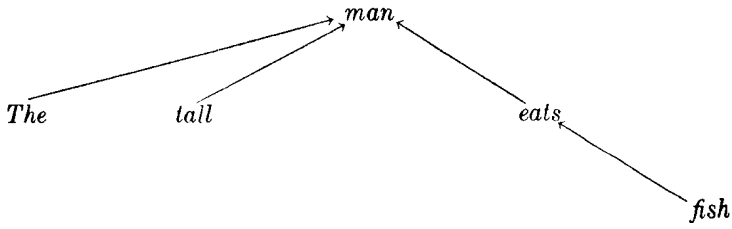
[14] Klein, ibid.; S. M. Lamb, 'The sememic approach to structural semantics', *Transcultural studies in cognition* 57–78 (A. K. Romney and R. G. D'Andrade, eds.; *American anthropologist* 66:3 part 2, 1964).

1. $Art_0 + *N_1 = N_2$
2. $Adj_0 + *N_1 = N_1$
3. $*V_1 + N_2 = V_2$
4. $*N_2 + V_2 = S_1$

TABLE 2. DEPENDENCY PHRASE-STRUCTURE GRAMMAR

keeping track of the local dependencies as each parsing rule is applied. The program yields as output a dependency tree superimposed on a phrase-structure tree.

The dependency analysis alone is represented in diagram form as follows.




It may also be expressed in a list structure format, which is the way it may be stored in a computer:

| 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| *The* | *tall* | *man* | *eats* | *fish* |
| 2 | 2 | | 2 | 3 |

**2.2.** GENERATION. The use of binary phrase-structure rules to generate grammatically correct nonsense is well known from the work of Chomsky[8] and Yngve.[9] If the output of such a program is monitored so that it contains only the vocabulary of an input text and so that only direct or certain transitive dependency relations are common both to source text and generation output, then the generated sentences will conform to the meaning of the input text.[10] For the purpose of making a single dependency analysis of a whole text, like tokens of the same noun are assigned two-way dependency links.

For example, consider the text, *Dogs are canines. Canines eat meat.* Assume that the vocabulary of a grammatically correct nonsense generator is limited to the vocabulary of this text. Also add the following to the rules of Table 1:

5. $N_1 = N_0$
6. $V_1 = V_0$

Assume that parts of speech are uniquely assigned to the vocabulary and that in the rewrite rules a symbol with a zero subscript may be rewritten with an appropriate lexical item.

[8] N. Chomsky, *Syntactic structures* ('s-Gravenhage, 1957).

[9] Yngve, op.cit.

[10] Klein, op.cit.; Klein, 'Automatic decoding of written English' (dissertation, University of California, Berkeley, June 1963); Klein and R. F. Simmons, 'Syntactic dependence and the computer generation of coherent discourse', *Mechanical translation* 7:2.50–61 (1963).