DOCUMENT RESUME

ED 022 148 AL 001 359

By-Chapin, Paul G.; Norton, Lewis M.
A PROCEDURE FOR MORPHOLOGICAL ANALYSIS.
Pub Date Jul 68
Note-20p.
Available from-The MITRE Corporation, Box 208, Bedford, Massachusetts 01730 (Information System Language Studies No. 18).
EDRS Price MF-$0.25 HC-$0.88
Descriptors-*COMPUTATIONAL LINGUISTICS, COMPUTER PROGRAMS, ENGLISH, *LANGUAGE PATTERNS, MORPHEMES, *MORPHOLOGY (LANGUAGES), PLURALS, SPELLING, *STRUCTURAL ANALYSIS, *SUFFIXES, SURFACE STRUCTURE, VERBS
Identifiers-*MORPH, TREET Programming System

A procedure, designated "MORPH," has been developed for the automatic morphological analysis of complex English words. Each word is reduced to a stem in canonical or dictionary form, plus affixes, inflectional and derivational, represented as morphemes or as syntactic features of the stem. The procedure includes the task of analyzing as many nested levels of affixation as a word may contain so that each morpheme of the input string will have a distinct representation. The overall strategy includes--(a) a set of analysis rules, (b) a set of morpheme-combinatorial rules, and (c) a set of redundancy rules. Spurious analyses are dealt with by certain modifications in these sets of rules. The procedure has been implemented on the IBM System 360 in TREET as part of an experimental text-processing system in which it provides the input to a transformational syntactic analysis procedure. The present sets of rules are quite small, removing only the most common inflectional suffixes (-s, -ed, -ing). Using these rules on a small vocabulary, MORPH has performed accurately at an average speed of 0.7 seconds per word. It was discovered in experimenting with this program that proper names require special treatment to avoid spurious analysis and that there is far greater homography with English prefixes than with suffixes, to the extent that the value of prefixational analysis is questionable. (JD)

A PROCEDURE FOR MORPHOLOGICAL ANALYSIS

by

Paul G. Chapin
University of California, San Diego
La Jolla, California

and

Lewis M. Norton
The MITRE Corporation
Bedford, Massachusetts

July 1968

A classical problem for natural language processing systems is the high redundancy, in terms of dictionary entries, of the set of words encountered in raw text. According to highly productive rules, for example, all but a handful of English nouns have plural forms differing (in spelling) from the singular only in the presence of a final *(e)s*; third person singular verb forms differ from the infinitive in the same way; most adjectives have regular comparative and superlative forms. Somewhat less productive rules generate various affixational derivatives of lexical stems, such as *happiness*, *modernize*, *continental*. A dictionary which contained separate entries for all these forms would clearly be highly inefficient. Some sort of morphological analysis must be employed by any system general enough to be useful.

In the literature of computational linguistics, three general approaches to the morphological analysis problem have developed. In projects directed toward content analysis [1,2,3,4] affixes (generally suffixes) have been removed and in effect discarded, leaving only stems for consideration. Where the orientation was syntactic [5,6,7] affixes have been treated as heuristic clues for automatic part-of-speech classification, in an attempt to avoid dictionary look-up altogether. The most sophisticated and ambitious attempts at a generalized natural language processing capability [8,9,10] have recognized the ultimate necessity of considering stems and affixes combinatorially as syntactically and semantically functioning elements, and have programmed dictionary look-ups which assign syntactic (and semantic) codes to stems and modify the syntactic code according to the final suffix, if any.

Even the latter efforts, however, produced as output encodings of words. A generative grammar as generally conceived has morphemes for terminal elements, and a syntactic analysis procedure based on such a grammar has as its first task the decomposition of an input string into its component morphemes. The present procedure is one approach to the performance of that task. It is designed to reduce morphologically complex English words to stems in canonical or dictionary form, plus affixes, inflectional and derivational, represented as morphemes or as syntactic features of the stem. Each morpheme of the input string should have a distinct representation for purposes of further analysis; consequently the task of the procedure includes analyzing as many nested levels of affixation as a word may contain. In this respect also it differs from the procedures described in references 1-10.

The overall strategy is as follows. A set of analysis rules (AN) decompose words of the input string into presumable stems and presumable affixes.[1] These rules are based on characteristic spelling of the affixes. The analysis rules also restore the stem to its dictionary form if its spelling was deformed by affixation (for example, skating would be analyzed as skate+(ING)).[2] The presumable stem is then

[1] The linguistic nature of English affixation apparently makes it possible to analyze as many nested levels of affixation as a word may contain in a single pass through the analysis rules, if provision is made for internal looping in the case of certain suffixes. See [11] for discussion.

[2] An alternative to restoring analyzed stems to their canonical forms would be to use a lexicon of deformed stems, with lexical look-up

looked up in the lexicon, and all of its lexical entries are associated with it. Thus *skate* might be categorized as a count noun and an intransitive verb. Next a set of *morpheme-combinatorial rules* (MC) apply. The MC relate the set of lexical entries assigned to the stem to the presumable affixes discovered by the AN, and modify the lexical entries accordingly. So if the input word had been *skates*, analyzed as *skate*+(S), one MC rule would modify the noun entry for *skate* to indicate plurality, and another would modify the verb entry to (third person) singular. Finally a set of *redundancy rules* (RD) fill in additional syntactic information not specified by the lexical entry or the MC. Thus if the input word was *skate*, one RD rule would indicate that the noun entry should be marked singular, and another would mark the verb as plural. The output of the RD is a fully categorized string of morphemes which serve as input to the syntactic analysis procedure.

In the ideal case application of the AN would yield exactly the right morphological analysis of every input string--that is, the presumable stems and affixes always would be the actual stems and affixes.

sometimes operating on the basis of partial rather than complete match with the input word. Thus instead of restoring the stem-final *e* in the analysis of *skating*, the AN would yield *skat*+(ING) and the matching lexical entry would be *skat*. *Skate* as an input word would then also match *skat*. This is essentially the procedure followed in [9]. Such an approach is especially attractive in the analysis of languages whose canonical forms are bimorphemic, e.g., Russian, Spanish, etc. It is not without its problems, however. For example, how are *wag* and *wage*, *fin* and *fine*, *mat* and *mate*, etc., kept distinct?

In practice, of course, the ideal is unattainable because of the problem of homography. Spurious analyses will result when a word has a spelling characteristic of a class of derivatives without being a member of that class, e.g., _herring_. The AN can be designed to avoid making a certain number of wrong analyses by specification of minimal stem length and inadmissible stem spelling. Thus if the AN for -_ing_ only applies when at least two letters precede the final -_ing_, analysis of _sing_, _wing_, _ring_, etc., is avoided, and removal of the final -_s_ from _fuss_, _mess_, _buss_, etc. is avoided by not allowing the AN for -_s_ to apply when the penultimate letter is also _s_. However, a large number of spurious analyses cannot be avoided by such qualifications.

There are two mechanisms within the present procedure for rejecting spurious analyses. The first and simpler rejects an analysis when the presumable stem is not found in the lexicon. Thus if one of the AN detects the final -_ly_ characteristic of adverbs derived from adjectives, some of the words to which it will apply are _philately_, _contumely_, _homily_, and _family_. However, their presumable stems--*_philate_, *_contume_, *_homy_, and *_famy_ (the _y_ for _i_ substitution by the rule which gives _happy_ as the stem of _happily_)--are not English words and will not be found in the lexicon. The presumable analyses will therefore be rejected.

It can also happen that a spurious presumable analysis yields a stem which _is_ in the lexicon. We may take _witness_ as an example. The analysis rule which removes the -_ness_ characteristic of nominalized adjectives (e.g., _rudeness_) will analyze _witness_ as _wit_+(NESS). _Wit_ is a valid English stem, a noun. After it has been so categorized by lexical look-up, the resulting

N + NESS goes to the MC. There is an MC rule for ADJ + NESS, but none for N + NESS. Since no MC rule applies, the analysis is rejected as spurious.

Each time a presumable analysis is made, the configuration immediately prior to the analysis is saved as an alternative presumable analysis. The result of application of all the AN is thus a set of presumable analyses, some of which will be rejected.

Implementation

The procedure, designated MORPH, has been implemented on the IBM System 360 in TREET, a list-processing system [12]. It has been designed to serve as part of an experimental text-processing system, in which it provides the input to a transformational syntactic analysis procedure. Since the grammar on which this procedure is based deals with grammatical strücture below the word level entirely in terms of syntactic features, the present version of MORPH represents morphemically complex words as stems with associated feature-value pairs, which may include inherent features of the stem as well as features corresponding to the various discovered affixes. A fairly trivial modification of MORPH would be required to represent words as strings of morphemes.

The operation of the analysis rules can be indicated in detail by a description of the syntax of a rule. Each AN rule is actually a statement in a variant of the string-processing language METEOR, for which an interpreter has been written in TREET, and the set of rules comprises a METEOR program. METEOR has been adapted for this use in MORPH by eliminating those capabilities not needed for the task at hand, and by including some new features which prove useful. Thus while the reader may profit by referring to the detailed description of the METEOR syntax [13], differences will be apparent in the discussion of the analysis rule format, which follows.

```
(INGS $SAVE $REV ($Ø (S) G N I $2) (1 2 (ING) 6) EINSDF END)
```

Figure 1
Sample Analysis Rule

An analysis rule (Figure 1) consists of seven fields, only two of which are obligatory. The first field is an optional name for the rule, by which it can be referenced in another rule. The second is an optional occurrence of the symbol $SAVE (this symbol cannot be used as a name) which, if present, causes the program to save the partial results of the analysis prior to the application of the rule. Saving does not occur if the rule does not apply. The third field is an optional occurrence of another special symbol, $REV, which, if present, specifies reversal of the elements to be analyzed, before application of the rule. This results in a right-to-left analysis. The fourth and fifth fields are lists used for the actual specification of the rule. These are required, and will be described shortly. The sixth field is an optional name of some other rule (i.e., a symbol in the first field of another rule) which will be tried next if the present rule applies. If the symbol END is in this field, no more analysis rules will be tried if the present rule successfully applies. The seventh field, also optional, cannot be present unless the sixth field is included. It specifies a rule to be branched to if the present rule fails. END may be used in this field also. In the absence of direction from these last two fields, control passes to the next rule in sequence, if any.

The rule specification fields describe a "before" and "after" situation, respectively. Thus the second specifies operations to be performed if the first field finds a match. Elements of these fields are called constituents, and the two fields are called the left-half and the right-half of the rule. These fields are processed from left to right.

Possible left-half constituents include:

a) an element (a letter or a list not specifically described below) -- This must match identically, as a single constituent. A letter matches a letter of the input word. A list matches a previously analyzed affix.

b) a list of elements, headed by the symbol $OR -- One element must match.

c) a list of elements, headed by the symbol $NOT -- Any element not in the list will match.

d) a symbol of the form $n, where n is a digit -- Any consecutive n elements will match. This is used to require at least n elements before a suffix, for instance. n may be Ø; $Ø constrains the match to start at the left boundary of a word.

e) the symbol $$ -- This matches the right terminal boundary of a word. ("Right", and "left" in the above discussion of $Ø, is defined after the effect of an occurrence of $REV. Due to the availability of the $REV option, $$ is seldom needed.)

f) the symbol $ -- This matches any number of arbitrary elements.

g) a digit n -- This matches whatever the nth constituent matched.

Obviously it must be at least the n+1st constituent.

h) a list whose first member is $FN -- This is a provision to allow more complex restrictions than $OR or $NOT. See [13] for details.

Right-half constituents specify the new appearance of matching structure. They include

a) letters or lists, which are inserted as constituents.

b) digits n, which specify the retention (and new location) of the nth left-half constituents. The absence of a digit corresponding to some left-half constituent indicates the deletion of that constituent.

```
((V) NIL ((D .)) (V (TNS PST)) (ADJ) )
```

Figure 2
Sample Morpheme-combinatorial Rule

A morpheme-combinatorial rule (Figure 2) begins with a categorization, which presently is restricted to a category label and zero or one feature-value pairs indicating subcategorization (e.g., a rule may apply only to transitive verbs). The second and third fields are lists of prefixes and suffixes, respectively. One of these fields must be NIL; i.e., a rule cannot refer to both prefixes and suffixes. These three fields specify the stem categorization and affix structure to which the rule applies. The non-NIL affix list has the following form: The list begins with the affix nearest the stem. Each affix is represented by a list (e.g. "(D)" for -ed), and each list may have an optional second member of "*" or ".". The "*" specifies a match with a flagged (i.e., previously matched--see discussion below) affix only; the "." specifies a match with an unflagged affix only. Omission of this second member allows the indicated affix to match in either event. The remaining elements of the MC rule form the fourth field, a list of categorizations, possibly with subcategorizations (feature-value pairs), to be assigned to a successfully matched stem-affix combination.

During the course of applying the MC, intermediate categorizations of the stem in combination with some of its affixes are obtained. Each of these partial results is marked with the number of affixes thus far accounted for. Each successful application of an MC rule produces intermediate categorizations accounting for exactly one more affix. For each tentative decomposition of the input word produced by the analysis rules, the process of applying the MC is basically a cycle through intermediate categorizations.

Given a decomposition whose stem appears in the lexicon, the initial steps are to count the number of affixes in it (call this N) and to set the intermediate categorizations to be the lexical categorizations of the stem, each with an indication that no affixes have been accounted for. Once this is done, MORPH begins considering the intermediate categorizations.

For each intermediate categorization, a check is first made to see if the number of affixes accounted for is equal to N. If this is so, the categorization becomes input to the next set of rules, the redundancy rules. For example, in the case of the trivial decomposition where $\underline{N} = 0$, the lexical categorizations of the simple stem are the end result of the (vacuous) application of the MC.

If affixes remain to be accounted for, the MC rules applying to the category label of this particular intermediate categorization are obtained. Each such rule is checked, in order, to see if a match can be found. The feature-value pair, if present in the rule, must be found among the feature-value pairs of the intermediate categorization. If this requirement is met, attention turns to the affixes. The prefixes or suffixes specified by the rule must match an unbroken string in the decomposition, starting at the position adjacent to the stem. Flagging restrictions in the ru'e must be met by the match. Further affixes may be present in the decomposition; all such would be farther from the stem than all affixes participating in a match.

If no MC rule for the category label produces a successful match, the intermediate categorization is discarded; i.e., the proposed analysis path contains an illegal stem-affix combination. If a successful match is made, the last affix of the decomposition which participated in the match is flagged, if it had not been flagged previously by successful application of an MC rule to another intermediate categorization. Flagging remains in effect while all intermediate categorizations of a given decomposition are processed (thus flagging alone cannot serve to keep track of how many affixes have been accounted for in a given intermediate categorization).

Successful application of an MC rule yields, from the rule's last field, new intermediate categorizations, each of which is marked as having accounted for one more affix than the old intermediate categorization. If a new categorization has the same category label as the old, feature-value pairs of the two are merged. The new intermediate categorizations are added to the top of the list of those to be considered.

At the end of the application of the MC, zero or more of the decompositions of the input word produced by the analysis rules will have one or more categorizations assigned to them. MORPH retains the information as to which stem underlies each such categorization. If no categorizations appear at this stage, it is as if the stem(s) were not found in the lexicon--MORPH cannot analyze the input word. Categorizations produced by the use of the MC are now sent through the redundancy rules.

((V) (TNS PRES)(NUM PL))

Figure 3
Sample Redundancy Rule

The role of the redundancy rules is to complete the categorization of a word by specifying the values of those of its features which have not already been specified in the lexicon or by the MC. Alternatively they may be thought of as filling in the morphemes with zero graphemic reflexes. The particular information supplied by the RD in the present implementaticn is in the form of feature-value pairs. Each RD rule (Figure 3) begins with a categorization indicating the lexical category to which it applies. As with MC rules, this categorization may include one subcategorizing feature-value pair. The remainder of the rule is a list of feature-value pairs. When a rule is found to be applicable to a categorization, these pairs are considered, and if <u>none</u> of their feature labels are present in the categorization of the word (i.e., no conflicts are found), the pairs are added to the categorization. The result of applying the RD is the set of final categorizations for an input word.

We now give the present AN, MC, and RD, with comments on each set. An example of the procssing of an input word by MORPH using these sets of rules is then given.

Analysis rules:

```
1)   ($SAVE $REV ($Ø S ($NOT I O S U)) (1 (S) 3) INGS)
2)   ($SAVE $REV ($Ø D E) (1 (D)) EINSB)
3)   ($SAVE $REV ($Ø G N I) (1 (ING)) EINSB END)
4)   (INGS $SAVE $REV ($Ø (S) G N I) (1 2 (ING)) EINSDF)
5)   ($REV ($Ø (S) E ($OR I O)) (1 2 4) Y END)
6)   (EINSB $REV ($Ø ($OR (D)(ING))($OR L R)($OR D T Z)) (1 2 E 3 4) END)
7)   ($REV ($Ø ($OR (D)(ING))($OR C U V Z)) (1 2 E 3) END)
8)   (EINSDF $REV (($OR (D)(ING))($OR N T)($OR A I)($NOT A)) (1 E 2 3 4) END)
9)   ($REV (($OR (D)(ING))($OR L R) U ($NOT A O)) (1 E 2 3 4) END)
10)  (Y $REV ($Ø ($OR (S)(D)) I) (1 2 Y) END)
11)  ($REV (($OR (D)(ING))($OR L M N) 2) (1 2) END)
```

Rule 1) detaches the suffix -*s*, reducing nouns to the singular form and (third-person) verbs to the present plural or infinitive form. The restriction on the letter preceding the *s* eliminates many incorrect analyses, such as for *analysis*, *grass*, etc. At the same time some correct analyses are prevented, e.g. *taxis*.

Rules 2) through 4) detach the suffixes -*ed* and -*ing*, rule 4) covering the -*ing* + -*s* case. These are the only affixes which this set of analysis rules processes; since all are suffixes, the rules all make use of $REV, specifying right-to-left processing. Rules 5) through 11) attempt to restore stems to their correct form. For instance, rule 9) restores the *e* on the stem of such inputs as *manufacturing*, *ruled*, etc. The restriction prevents incorrectly adding the *e* to the stem of such words as *hauling*, *pouring*, etc. Only two known incorrect analyses are made, for *auguring* and *murmuring*. Rule 10) replaces *i* with *y*, and rule 11) removes the second of doubled consonants, as in *programming*.

Morpheme-combinatorial rules:

1) ((V) NIL ((ING .)) (V (ING PLUS)) (ADJ (ING PLUS)) (N (CMNF CMN) (ANIM MINUS)(ING PLUS)))

2) ((V) NIL ((S .)) (V (TNS PRES)(NUM SG)))

3) ((V (TRANS MINUS)) NIL ((D .)) (V (TNS PST)))

4) ((V) NIL ((D .)) (V (TNS PST)) (ADJ))

5) ((N) NIL ((S)) (N (NUM PL)))

6) ((N) NIL ((ING *)(S .)) (N (NUM PL)))

For this set of rules, the legal combinations are: a) from rule 1), verb + -*ing*, yielding a verb, adjective or noun with appropriate feature-value pairs; b) verb + -*s*, with the information obtained from the suffix retained by encoding it in feature-value pairs; c) verb + -*ed*, except that for intransitive verbs the participle (adjective) form is not allowed, so two rules are needed; d) noun + -*s*, as expected, where rule 6) covers the -*ings* case, which involves a path through rule 1)'s categorization of verb + -*ing* as a noun. The use of the "." denoting a match with only an unflagged affix in the first four rules assumes that the lexical categorizations of a word have been ordered in the lexicon so that any one with the category label V is first. Since an entry in the lexicon might have categories V and/or N, rule 5) must allow a match with either a flagged or an unflagged affix.

Redundancy rules:

1) ((V) (TNS PRES)(NUM PL))

2) ((N) (NUM SG))

3) ((N (HUM PLUS)) (ANIM PLUS))

4) ((N (ANIM MINUS)) (HUM MINUS))

These rules specify information as follows: If a verb has neither tense nor number specified, it is taken as present plural; nouns not marked plural are assumed singular; appropriate interpretations are made for the features of humanity and animacy for nouns.

Consider the input word holdings. MORPH first applies the analysis rules to the string of its letters, (H O L D I N G S). Rule 1) produces (H O L D I N G (S)), and $SAVE causes retention of the prior analysis (HOLDINGS). Control is passed to rule 4), INGS, which applies, detaching the second suffix to produce (H O L D (ING) (S)), and saving (HOLDING (S)) as a possible analysis. Control passes to EINSDF, rule 8), and subsequently to rules 9), 10), and 11), none of which apply (correctly, since HOLD is a correctly spelled stem). Thus there are three tentative decompositions: (HOLDINGS), (HOLDING (S)), and (HOLD (ING) (S)).

Neither holdings nor holding are lexical stems, so the only tentative decomposition of interest is (HOLD (ING) (S)). A possible lexical entry for hold is (HOLD (V (TRANS PLUS)) (N (CMNF CMN))), which indicates that hold is a transitive verb or a common noun. (Note the ordering of the categorizations, with the V before the N.) Given these lexical categorizations, the application of the MC begins with the intermediate categorizations (Ø V (TRANS PLUS)) and (Ø N (CMNF CMN)). The zeros indicate that no affixes have been accounted for.

The first intermediate categorization is examined. Its category label is V, and the four MC rules pertaining to this label are considered. The first of these, rule 1), matches (the (ING) of the decomposition is unflagged, as required). This causes flagging of the (ING) so that the tentative decomposition now appears as (HOLD (ING *) (S)). The three categorizations of MC rule 1)'s last field become intermediate categorizations with 1's associated with them to indicate that they have accounted for one affix. Thus the list of intermediate categorizations is now (1 V (ING PLUS)), (1 ADJ (ING PLUS)), (1 N (CMNF CMN)(ANIM MINUS) (ING PLUS)) and (Ø N (CMNF CMN)). Each of these must be examined. The category label of the first is V, but MC rule 1) does not match this time, since the (ING) is now flagged. No other rules apply and the counter, 1, does not equal N (N = 2, since there are two suffixes), so this path is discarded (note that the flagging operation is crucial here). The second intermediate categorization is also discarded, since there are no MC rules applying to the category ADJ and its counter is also 1. Rejection of these two paths corresponds to the fact that the progressive verb and participial adjective forms holding do not take an -s.

The next intermediate categorization has category label N, and there are two MC rules for this label. The first, rule 5), does not apply because the (S) is not adjacent to the stem in the decomposition. However, the next applies, changing the decomposition to (HOLD (ING *) (S *)) and yielding the intermediate categorization (2 N (NUM PL) (CMNF CMN)(ANIM MINUS)(ING PLUS)). The 2 indicates that this categorization has accounted for both suffixes and is to be sent to the RD.

The remaining intermediate categorization, (Ø N (CMNF CMN)), triggers an attempt to apply rules 5) and 6) again. 5) does not match, as before, and now 6) does not either, due to the flag on the (S *). Therefore, this categorization is discarded, as it should be since nouns do not take the suffix -*ing*. Note again the importance of the flagging operation.

The RD rules associated with N, namely 2), 3), and 4), are now applied to the categorization (N (NUM PL)(CMNF CMN)(ANIM MINUS)(ING PLUS)). Rule 2) has no effect since number has already been specified. Rule 3) does not apply since the feature-value pair restriction is not met, but rule 4) applies, yielding as final output from MORPH the categorization (N (HUM MINUS)(NUM PL)(CMNF CMN)(ANIM MINUS)(ING PLUS)). MORPH also reports that the stem associated with this analysis of *holdings* is *hold*.

Discussion

The present sets of rules are quite small, removing only the most common inflectional suffixes. Using these rules on the small test vocabulary given it so far, MORPH has performed accurately at an average speed of about 0.7 seconds per word on the 360/40. The questions of accuracy and efficiency become more critical, of course, as more comprehensive morphological analysis is attempted. In an earlier implementation [14], a considerably larger set of AN was programmed in FORTRAN on the 7090, but without any corresponding MC or RD. This set of AN included rules for the suffixes -*ly*, -*ness*, -*or*, -*er*, -*est*, -*ible*, -*able*, and -*less*, and the prefixes *un*-, *in*-, and *non*-, as well as the rules for -*s*, -*ed*, and -*ing*. This program analyzed a vocabulary of 112 words in about 5 1/2 seconds, but with many spurious analyses, as would be expected (some interesting examples: (IN+*cape*+(ABLE) 'incapable'; (UN)+(LESS) 'unless'; (IN)+tege+(ER) 'integer'; *eith*+(ER) 'either').

Inspection of the results of this and subsequent experiments with this set of rules disclosed no spurious analyses which could not, in principle, be rejected by use of the complete procedure as described.[3]

An important result of the experiments with this program was the realization, obvious in retrospect, that proper names require special treatment, perhaps a distinguishing diacritic upon input. Otherwise many spurious analyses result, e.g., Socrate+(S), Grell+(ING), Schill+(ER). Another result was the discovery that there is far greater homography with English prefixes than with suffixes, to the extent that the value of prefixational analysis is questionable.

[3] There are, of course, cases of genuine orthographic ambiguity, words for which there are two or more valid morphological analyses, one of which will always be spurious in a given context. For example, number will always be analyzed as a comparative adjective as well as a noun.

References

1. Simmons, R., and K. McConlogue, "Maximum-depth Indexing for Computer Retrieval of English Language Data". American Documentation, Vol. 14, No. 1 (January 1963), pp. 68-73.

2. Sedelow, S. Y., Stylistic Analysis--Report on the First Year of Research, SDC TM-1908/100/00, System Development Corporation, Santa Monica, Calif., March 1965.

3. Stone, P., and E. Hunt, "A Computer Approach to Content Analysis: Studies Using the General Inquirer System". AFIPS Conference Proceedings, Vol. 23, 1963 Spring Joint Computer Conference, pp. 241-256.

4. Coyaud, M., "L'analyse Morphologique en Documentation Automatique". La Traduction Automatique, Vol. 5, No. 3 (September 1964).

5. Klein, S., and R. Simmons, "A Computational Approach to Grammatical Coding of English Words". Journal of the ACM, Vol. 10 (1963), pp. 334-347.

6. Stolz, W., P. Tannenbaum, and F. Carstensen, "A Stochastic Approach to the Grammatical Coding of English". Communications of the ACM, Vol. 8, No. 6 (June 1965), pp. 399-405.

7. Dolby, J., L. Earl, and H. Resnikoff, The Application of English-Word Morphology to Automatic Indexing and Extracting, Annual Summary Report to the Office of Naval Research, Contract Nonr 4440(00), April 1965.

8. Thorne, J., Automatic Language Analysis, Final Technical Report to U. S. Air Force Systems Command, Contract No. AF 30(602)-2185, December 1962. See especially page 67.

9. Salton, G., Information Storage and Retrieval, Harvard Computation Laboratory Scientific Report No. ISR-5 to the National Science Foundation, January 1964. See especially pp. I-10 to I-15.

10. Meyers, L., "Morphological Classification in the National Bureau of Standards Machine Translation System". Journal of the ACM, Vol. 12, No. 4 (October 1965), pp. 437-472.

11. Chapin, P., On the Syntax of Word-Derivation in English, Information System Language Studies Number Sixteen (MTP-68), The MITRE Corporation, Bedford, Mass., September 1967. See Section III.

12. Haines, E. C., The TREET-360 Programming System: Reference Manual. (forthcoming)

13. Bobrow, D., "METEOR: A LISP Interpreter for String Transformations". In Berkeley, E., and D. Bobrow (eds.), The Programming Language LISP: Its Operation and Applications. Cambridge, Mass.: The MIT Press, March 1967.

14. Friedman, J., Programming Lexical Grapho-Morphemic Analysis. Working paper, Computational Linguistics Project, Stanford University, Palo Alto, Calif., November 1966.