Printed at the Mathematical Centre, 49, 2e Boerhaavestraat, Amsterdam.

The Mathematical Centre, founded the 11-th of February 1946, is a non-profit institution aiming at the promotion of pure mathematics and its applications. It is sponsored by the Netherlands Government through the Netherlands Organization for the Advancement of Pure Research (Z.W.O), by the Municipality of Amsterdam, by the University of Amsterdam, by the Free University at Amsterdam, and by industries.

AUTOMATIC ANALYSIS OF DUTCH COMPOUND WORDS

BY W.A. VERLOREN VAN THEMAAT

MATHEMATICAL CENTRE TRACTS 38

MATHEMATISCH CENTRUM AMSTERDAM 1972

ACKNOWLEDGEMENTS

I wish to express my gratitude to the Mathematical Centre for the opportunity of performing the research dealt with in this paper. I am deeply indebted to Prof.dr. A. van Wijngaarden and Dr. H. Brandt Corstius for their many valuable suggestions and criticisms during this study. To Mr. J. Alanen I am grateful for the correction of the English text.

I would also like to thank Mrs. S. Hillebrand and Miss O. de Jong for the typing of the manuscript and Mr. J. Suiker, Mr. D. Zwarst and Mr. J. Schipper for the printing.

CONTENTS

1. THE PURPOSE OF THE ANALYTIC PROGRAM

1.0. INTRODUCTION

This publication is intended as a contribution to the construction of translation programs for computers.

Dutch has a large capacity to form new words by combination of language elements already existing. Some of these language elements are independent words; some are language elements not occurring as independent words, among which the most important are the affixes like *ing* (a suffix to derive nouns from verbs; its sphere of meaning is large; sometimes it corresponds to English *ing* (as gerund ending), sometimes to *-ion*).

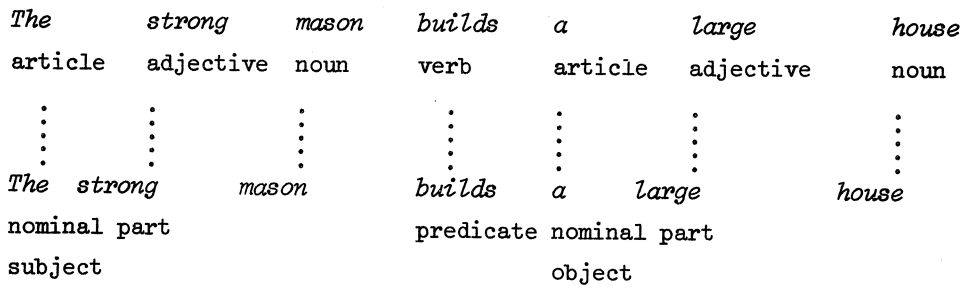
Because such compounds are continuously newly formed, not all compounds can be adopted into a lexicon and so for mechanic translation one must be able to determine their structure mechanically.

In linguistics there are several definitions of morphemes, but one which will be accepted by at least some linguists is: morphemes are the smallest elements which one must consider as a unity to be able to construct the words from them and to be able to formulate the construction rules. We adopt a more opportunistic point of view and say: the morphemes are the word constituents whose adoption in the analytic lexicon (the lexicon in which all constituents are looked up) is necessary or useful for a translation program as simple as possible. So the morphemes are not only the constituents truly without constituents, but also the constituents whose meaning cannot be deduced from those of their constituents (e.g., German *Hochzeit* from *hoch* (= *high*) and *Zeit* (= *time*), but meaning *wedlock*) or whose compounding rules are too complicated, e.g. that of the compound affix *erig* with *er* and *ig*.

In linguistics compounding (formation of a word from two or more independent words) is sometimes distinguished from derivation (formation of a word from one word with dependent morphemes). But since words are also often constructed from words and affixes, in the sequel I shall call all words consisting of more than one morpheme compounds (thus also grammatically inflected forms, as far as obtained by the addition of grammatical affixes to the word stem. Inflectional forms obtained by change of the stem, e.g., the past tense *viel* of *vallen* (English *fall - fell*) are con-

sidered as morphemes).

The structure of a compound COMPRISES its dissection into morphemes, considered as mere strings of letters, but does not COINCIDE with it. Just as one has not described the structure of a sentence completely by its dissection into words, but has also to indicate all intermediate constituents with their grammatical functions, e.g.,



, so the structure of a compound word is not completely described by its dissection into morphemes, but their intermediate constituents and their grammatical functions must be indicated too (the part of speech of constituents also occurring as words; which indications will be added to each constituent, will be treated in more detail later on). E.g., the word *onder-district-s-hoofd* has an entirely other meaning and is therefore otherwise translated into English, if *onder-district* or *district-s-hoofd* is considered as a constituent (respectively *sub-district chief* and *district vice-chief*).

The computer analyses the words occurring in the WORD-LIST with the aid of a LEXICON (the set of the words and morphemes with indications about the way they can occur in compounds) and the ANALYTIC PROGRAM. So the report contains four sections besides this introducing section:

2. The word-list
3. The lexicon
4. The analytic program
5. Results: error-analysis.

The word-list is treated before the lexicon, because the instructions for its punching are simpler.

1.1. THE FORM OF THE OUTPUT AND THE TASKS OF THE ANALYTIC PROGRAM

The analytic program is a program in ALGOL 60.

The complete description of the structure of a compound word W, as given as the result of the analytic program, is a so-called structural tree, indicating the so-called constituents. These constituents have the following properties:

1) W is the constituent of the 1-st order;
2) A constituent of the n-th order (n > 0) contains at least 1, but at most 3 constituents of the n+1-th order, unless all morphemes are constituents of the n-th order (the limitation, that a constituent has no more than 3 immediate constituents, is not an aprioristic postulate, but is based on specific properties of Dutch; see 4.1.4.);
3) Each constituent of the n+1-th order is contained in one and only one constituent of the n-th order;
4) All constituents of the order n+1 contained in one constituent of the order n follow each other immediately.

The order of the constituents of the highest order (the morphemes) is called hoogte (the Dutch word for heighth).

Every constituent is noted on at most 4 lines: on the first line stand its letters, on the second line its indexes, on the third "koppel", if it is united in a pseudo-concatenation with the leading constituent of the next-higher constituent (for the meaning of "pseudo-concatenations" see 4.1.4.), on the line immediately under "koppel" (if present, otherwise under the indexes) "leider", if it is the leading constituent of the next-higher constituent and does not coincide with it.

The indexes of each constituent form a number of 4 or 5 digits. The precise information they convey and the code for its notation will be explained in 3.2. Provisorily I satisfy myself with saying, that these indexes indicate the kind of constituent (word, affix, inflectional morpheme or link), the part of speech of words, and the possible position of the constituent in compounds (at the beginning, in the middle or at the end).

Intuitively the leading constituent of a string of constituents of the same order forming together one next-higher constituent is the constituent of which the other constituents are determiners. So in *huis-deur* (*house-*

door) *deur* is the leading constituent. In cases in which this semantic criterion does not yield sufficiently clear results, the constituent determining the *part of speech* of the compound is considered as the leading constituent, e.g. *ing* in *wandel-ing* (= *walk* (noun), *wandel* = *walk* (verb); all compounds formed by *ing* are nouns). In 4.1. it will be indicated precisely, which constituents are considered as the leading constituents of the next-higher constituents.

The constituents of the same order stand on the same line and are separated from each other by tabulation(s).

Three or four such lines, together with the following void line, are called a constituent row. Then a structural tree looks as follows:

1) On the upper constituent row stand the morphemes.
2) I suppose the constituent rows hoogte,..., n+1 to be constructed.

Then the constituents of the order n stand under them in a row in the order in which they occur in W. The first letter of each constituent stands perpendicularly under the first letter of the first morpheme belonging to it.

An example of such a structural tree is (under each constituent its English translation or an indication of its function is put - in deviation of the true output of the analytic program -):

reger	*ing*	*s*	*verantwoordelijk*	*heid*
govern	*~ion*	meaningless	*responsible*	*ness, ity*
3023	1060	22024	21	1040
		leider		
regering		*s*	*verantwoordelijk*	*heid*
government		meaningless	*responsible*	*ness, ity*
20		22024	21	1040
leider				leider
regerings			*verantwoordelijkheid*	
governmental			*responsibility*	
3020			20	
			leider	

regeringsverantwoordelijkheid
governmental responsibility
20

So for the calculation of the structural tree we must know:

1) Which are the morphenes;
2) Which constituents can be united to higher constituents;
3) How the indexes of the higher constituents can be calculated from those of the lower ones;
4) How pseudo-concatenations can be recognised;
5) How the leading sub-constituent of a constituent is determined.

1.2. THE DISSECTION INTO MORPHEMES: THE REIFLER CALCULUS

Though the dissection into morphemes is only part of the determination of the structure, it will be treated separately, because it has already been treated by Reifler ([10]).

The word to be analysed is called W and its number of letters, letter. The longest initial segment of a string of letters S is the longest substring of S beginning with the first letter of S which has an equivalent in the lexicon.

Then a simplified version of the Reifler calculus for the dissection of words into morphemes proceeds as follows (actually this is the calculus given by Reifler himself for compounds with more than two constituents, p. 13):

The 1-st, ..., n-th morpheme are determined by complete induction as follows:

1) If W has no longest initial segment the word is declared unanalysable; otherwise the longest initial segment of W is the first morpheme.
2) I suppose the 1-st, ..., n-th morphemes to be determined. If the last letter of the n-th morpheme is the last letter of W, then the morpheme dissection has been completed.

Otherwise, the longest initial segment of T, the segment of W acquired by the subtraction of the 1-st up to the n-th morpheme is the n+1-th mor-

pheme. If T has no longest initial segment, the longest initial segment of the n-th morpheme after subtraction of the last letter becomes the new n-th morpheme (in ALGOL 60 value assignments to identifiers can be undone by later assignments).

This simplified version of the Reifler calculus could be written as a program fragment in ALGOL 60 (if we neglect many declarations and input and output procedures, not relevant here). The following identifiers will be explained:

The ordinal number of the first letter of the k-th morpheme as a letter of W will be called unua [k], the ordinal number of the last letter of the k-th morpheme lasta [k].

lv(m,n) is the ordinal number of the last letter of the longest initial segment of the string from the m-th until the n-th letter. If that string of letters has no longest initial segment, lv(m,n) is put to 0.

The ordinal number of the morpheme being treated is called woord.

SUBJAS is the label immediately after the program fragment for the dissection into morphemes; this label is preserved for the sake of comparison with a fragment of the true analytic program.

```
   unua [1] : = 1; r : = lv(1, letter); woord : = 1;
L: if r = 0 then PUTEXT (⟨ unanalysable ⟩) else lasta [1] : = r;
   if r = letter then goto SUBJAS else
   begin    woord : = 2;
M:          q : = unua [woord] : = lasta [woord-1] + 1;
            lasta [woord] : = r : = lv(q, letter);
F:          if r = 0 then goto G else
   begin    if letter = r then goto SUBJAS else
            begin woord : = woord + 1; goto M end
end end;
G: woord : = woord - 1; q : = unua [woord]; r : = lv(q, lasta [woord] - 1);
   goto (if woord = 1 then L else F);

SUBJAS:
```

The characteristic feature of this dissection calculus is, that it takes into account only, WHETHER a certain string of letters can occur as constituent of a word, not IN WHICH POSITION IN THE WORD it can occur. E.g. the dissection *ing/reep* (*ingreep = intervention, operation* from *in* and *greep = grasp; reep = roll* (e.g. of chocolate); *ing~ing, -ion*) would be unreproachable according to this dissection calculus, though *ing* as a suffix can never occur at the beginning of a word. Reifler, indeed, has also gone beyond this primitive stage. For compounds of two constituents his dissection calculus is more refined than the one applied here, since it takes into account the possibility that a compound has the structure LT - X - RT, in which LT, LT - X, X - RT and RT are all possible constituents. He provided some nouns with indexes indicating, that they could not occur as first constituents and rejected the dissection *Literat/urwelt* (*literary man's primeval world*) instead of *Literatur/welt* (*literary world*), because *Literat* cannot occur as the first constituents of a compound. In cases in which this test did not apply, he gave two dissections, e.g. *Wacht/raum* (*guard room*) against *Wach/traum* (*day-dream*). In my dissection calculus this possibility is ignored, because according to Reifler himself "Such composita, are, however, extremely rare coincidences" ([10] p. 13).

Reifler's dissection method was usable for the limited purpose Reifler put to himself: the dissection of compounds of more nouns. This method is not usable for the far more comprehensive purpose of this study: the analysis of compounds of all possible morphemes: words of all parts of speech, affixes and inflectional morphemes (and links, as we shall see).

In some cases in the analytic program a completed dissection into morphemes is rejected for the impossibility to assign a structure to the word with this dissection into morphemes (see 4.3.), but even apart from that it is easy to see, that the program fragment above does not quite coincide with the fragment l. 888 - 935 of the analytic program, most nearly analogous to it. Apart from some differences concerning the connection with the rest of the program, the most relevant differences are:

1) The dropping of the hyphens, tremas and apostrophs behind morphemes (l. 902 - 903);

2) The examination for each morpheme, whether it can occur at the given

position and the rejection of the dissection into morphemes, if the morpheme cannot occur there (by the statements depending on conditional clauses containing ind1 and ind2: l. 892 - 893, 898 - 899, 906 - 907 and 911 - 912) and the statements positie, assigning values to ind1 and ind2. But the explanation of these statements will be postponed until 4.3., because the meaning of ind1, ind2 and positie cannot be explained without a previous treatment of other properties of the lexicon and the program. The statements depending on conditional clauses containing ind1 and ind2 serve to exclude morpheme dissections such as *ing/reep* treated in this section.

2. THE WORD-LIST

The word-list is not allowed to contain a space or NLCR before the first letter.

The words may contain letters of the Latin alphabet, with diacritical signs or not, points, hyphens, apostrophs, but no digits. The letters with diacritical signs are typewritten according to an especial code.

Both in the lexicon and in the word-list we put as many words as possible behind each other on one line, but each word entirely on one line. Two consecutive words in the word-list on the same line are separated by a space. The last word is followed by a space and a closing punching, for which ¬ is elected.

The words in the word-list need not stand in a definite order.

3. THE LEXICON

3.0. THE PURPOSE AND DIVISION OF SECTION 3

The questions section 3 tries to answer are:

1) Into which kinds of constituent are the morphemes and constituents divided;

2) Which morphemes and other constituents have to be adopted into the lexicon;

3) In which form(-s) words with more inflectional forms and other morphemes, which occur in GRAPHEMICALLY different forms in compounds, though they must be considered as one morpheme LINGUISTICALLY, have to be adopted, e.g. the suffix *eer* at the end of a word against *er* before a suffix or inflectional morpheme beginning with a vowel (Dutch has added this morpheme to the stem of many verbs borrowed from Romance languages, e.g. *copul-er-en* = French *copuler = to copulate);*

4) Which indications have to be added with each constituent.

Question 1 is answered in section 3.1., question 2 in 3.4. - 3.8. for each kind of constituent, question 3 partially in 3.3. (with respect to modifications determined exclusively by preceding or following letters and sometimes by the kind of constituent of a COCONSTITUENT, but never by the kind of constituent of the constituent itself), partially in 3.4. - 3.8. for each kind of constituent separately, question 4 in 3.2.

3.1. THE CLASSIFICATION OF THE MORPHEMES

The following kinds of constituents are distinguished for the morphemes:

1) WORDS and VIRTUAL WORDS. In 3.4. I shall discuss, which form(s) of inflected words will be adopted.

The so-called virtual words are put on a level with words. " *"tuintje"* (= *little garden*, from *tuin = garden*), *tuinier* (*gardener*), *beplanten* (*to plant*, from *planten = to plant*), *beplanting* (= *planting*) are formed with elements not occurring as words: *-tje*, *-ier*, *be-*, *-ing*. They are called PREFIXES and SUFFIXES; the words formed by them are called derivatives." ([11], p. 130).

Certainly all PRODUCTIVE morphemes must occur in the lexicon. Strictly sticking to this definition would yield very strange analyses for many compounds. The strangest consequence is, that some compounds would consist of only affixes without stems.

An example is the word *kinder-achtig (= child-ish)*. Its first constituent *kinder* does not occur as an independent word. LINGUISTICALLY it must be considered as consisting of a word-morpheme *kind (= child)* and an inflectional morpheme *er*, but for our analytic program this dissection would only yield unnecessary complications, i.a. because then in compounds like *kind-er-lijk (= child-ish)* an inflectional morpheme would be followed by a suffix, what fairly rarely occurs in Dutch and which possibility we have therefore excluded systematically in the analytic program. But it is certainly productive ([5] mentions 280 compounds of *kinder*). From the definition above it would then follow, that *kinder-achtig* would consist of two affixes.

A morpheme like *kinder* is not called an affix, but a VIRTUAL WORD. I define a virtual word as a CONSTITUENT, NOT OCCURRING AS AN INDEPENDENT WORD, BUT ACTING IN THE FORMATION OF COMPOUNDS AS IF IT OCCURRED AS AN INDEPENDENT WORD AND TO WHICH THE MEANING OF AN INDEPENDENT WORD (THE "TRANSLATION") CAN BE ASSIGNED IN SUCH A WAY, THAT THIS MEANING TOGETHER WITH THE MEANINGS OF THE OTHER CONSTITUENTS OCCURRING IN THE COMPOUND YIELDS THE CORRECT MEANING OF THE COMPOUND.

The word with which *kinder* agrees in meaning is of course *kind*.

2) AFFIXES are morphemes not occurring with the same meaning as words or virtual words, but yielding a word together with one word or virtual word. E.g. *ig* in *blauw-ig (blu-ish)*. This does not exclude, that affixes can be HOMONYMOUS with independent words (e.g. *in (= in)* as a preposition and *in* as a suffix for female living beings: *koning-in = queen* from *koning = king)*. In that case the meaning of a compound formed by that affix cannot be derived from the meaning of a homonymous word as an independent word (of course with the aid of the other morphemes occurring in the word).

The affixes are divided into two subspecies: DOMINANT and RECESSIVE ones. An affix A is called DOMINANT, if it imparts a fixed part of speech independent of the part of speech of W to every compound A-W (if A is a

prefix) or W-A (if A is a suffix). This part of speech is called the part of speech of the affix itself.

A prefix P is called RECESSIVE (recessive suffixes do not exist), if every compound P-W has the same part of speech as W.

These definitions leave us the choice to consider prefixes P connectible only to words W of one part of speech w, for which P-W always has the part of speech w, as dominant or recessive. We consider them as recessive, because this simplifies the analysis and is semantically most acceptable (in other languages such prefixes are mostly translated by determiners to the second constituent, e.g. *schoon-moeder = mother-in-law*).

3) INFLECTIONAL MORPHEMES. By this I understand morphemes, which can yield inflectional forms of a word together with words (or virtual words), e.g. *t* as the inflectional morpheme of the 3-rd person singular of the present tense of the verb. They have to be distinguished from the INFLECTIONAL FORMS, which are words, e.g. *loop-t (run-s)*.

The inflectional morphemes cannot be dropped from the lexicon, because inflected compound words do not always occur in the text to be analysed in the form in which they would be mentioned in the lexicon (e.g. a compound noun may occur in the text in plural, while it would be mentioned in its singular form in the lexicon) and must also then be analysable.

4) LINKS, not having a clear own syntactic or semantic function and always standing between two other constituents, e.g. the *e* in *eik-e-blad (oak leaf)*.

3.2. THE FORMAL STRUCTURE OF THE LEXICON AND THE MEANING OF THE SETS OF INDEXES

The lexicon is a series of morphemes (i.e. strings of letters or other writing signs, which can occur in words), each followed by one or more numbers of at most five digits, called SETS OF INDEXES (if the number of digits $n < 5$, then the first $5-n$ digits are supposed to be 0). The 1-st digit, the 2-nd digit, the number formed by the 3-rd and 4-th digits and the 5-th digit are called, respectively, the 1-st, 2-nd, 3-rd and 4-th index. These sets of indexes give roughly much information: how the morpheme concerned can occur in compounds; its kind of constituent, its part of speech,

its possible position in compounds, etc. A mere string of letters in the lexicon is called a LEXICAL MORPHEME. A lexical morpheme together with one of its (perhaps more)sets of indexes is called a SYNTACTIC MORPHEME. Syntactic morphemes with the same lexical morpheme, but different sets of indexes, are called HOMONYMOUS morphemes (e.g. *in* as a preposition and *in* as a suffix for female living beings). A lexical morpheme together with all its sets of indexes is called a COMPLETE MORPHEME.

The lexical morphemes are ordered in the lexicon according to increasing length and alphabetically for equal length. The capitals are supposed to follow the lower-case *z* in the alphabet.

The point, hyphen, apostroph and the diacritic signs follow:

¨

cedille (with c) or ¸

´

ˆ

Apostroph
Hyphen
Point

For the determination of the position of a word in an alphabetised lexicon, the diacritic signs are supposed to stand before the letter over or under which they stand.

The lexical morphemes are ordered according to increasing length in order to facilitate the comparison of segments of the words to be analysed with lexical morphemes. Segments are only compared with morphemes of the same length and this comparison is facilitated, if all morphemes of the same length stand together in the lexicon. The conventional position of the capitals behind the lower-case letters is chosen for the sake of adaptation to the representation of the letters in the X8-code (the X8 is the computer at the Mathematical Centre, Amsterdam, on which the analytic program was first executed).

Each set of indexes is written behind the lexical morpheme to which it belongs. Each set of indexes is preceded by a space and followed by a comma. The sets of indexes of one lexical morpheme are ordered according to decreasing frequency of the syntactic morphemes to which they belong.

As we shall see in 4.3., the computer executing the analytic program, having dissected the word into morphemes, first tries to construct a structural tree, in which each lexical morpheme has the first set of indexes standing behind it in the lexicon and only if this turns out to be impossible, it explores consecutively the following sets of indexes (a certain choice among sets of indexes of a lexical morpheme on the base of probabilistic considerations occurs in the analytic program l. 443 - 444 and is explained in 4.2.). The ordering of the sets of indexes according to decreasing frequency promotes the choice of the most probable structural tree.

In order to save paper we put as many complete morphemes as possible on one line, but each complete morpheme entirely on one line. The different complete morphemes are separated from each other by tabulation or transition to a new line. The first complete morpheme has to be preceded by a NLCR on the punch-tape. The lexicon is opened and closed by special punchings; in the analytic program < is elected for the opening and ¬ for the closing.

Homonymes in the usual sense, differing ONLY in their MEANING (e.g. *kool* as *coal* and *kool* as *cabbage*, both nouns), are not distinguished in the analytic program, but morphemes differing in one of their indexes (e.g. *acht* as a numeral *(eight)* and as a noun *(attention)*, *ver* as an adjective *(far)* and *ver* as a prefix (a very wide sphere of meaning, sometimes it corresponds to the English suffix *ify: eenvoudig = simple; ver-eenvoudig = simpl-ify))* are distinguished.

The 1-st index indicates the possible position at the beginning of the word, the 2-nd index the possible position at the end of the word.

The 3-rd index indicates the kind of constituent and for affixes moreover the parts of speech with which they can be connected and the order of the parts of speech of decreasing so-called affinity of the affix to the part of speech for affixes connectible with more parts of speech. There are also affinities among parts of speech for words; they will be treated in 4.2.2. In the analytic program the affinities of parts of speech to affixes are represented by array-elements with two indexes. But in their linguistic meaning they are integer functions of two parameters: a part of speech and an affix. Two affixes B_1 and B_2 are called AFFINITY-EQUIVALENT,

if every part of speech has the same affinity to B_1 as to B_2. The classes of affinity-equivalent affixes are called AFFINITY CLASSES. In the report (not in the analytic program !) if a word or affix A belongs to a part of speech a and an affix B to an affinity class b, the affinity of a to B will also be called the affinity of A to B, or that of a to b. Two other uses of the affinities in the analytic program, and the criteria for the determination of their values, will be treated in 4.2.3. But the affinities serve mainly for the determination of the structure of tripartite compounds A-B-C. If A is a recessive prefix, B a word and C a word or a suffix, A and B together form a constituent, if the affinity of B to A $\geq$ the affinity of C to A and B and C, if the affinity of B to A < the affinity of C to A. If A is a dominant prefix, B a word and C is a suffix, A and B together form a constituent, if the affinity of B to A $\geq$ the affinity of B to C and B and C form a constituent, if the affinity of B to A < the affinity of B to C. E.g. the prefix *on (= un)* has a greater affinity to adjectives than to nouns, and on the base of that *on-gevoel-ig (gevoel = sentiment, ig~ish, y, al)* is analysed as *on-gevoelig (un-sentimental)* and not as *ongevoel-ig*.

The 4-th index indicates the part of speech. By the part of speech of an inflectional morpheme I understand the part of speech of the words with which it can be connected.

The part of speech of a dominant affix has been defined in 3.1. The recessive prefixes get a 4-th index 7.

The links obtain an entirely arbitrary 4-th index, having nothing to do with any part of speech (see 3.8.).

Now follows a table of the meanings of the indexes.

Possible position at the beginning of the word	1-st index
Can occur at the beginning of the word, but also in other positions	0
Can only occur at the beginning of the word	1
Cannot occur at the beginning of the word	2
Possible position at the end of the word	2-nd index
Can occur at the end of the word, but also in other positions	0
Can only occur at the end of the word	1

Possible position at the end of the word	2-nd index
Cannot occur at the end of the word, but does not impose phonetical restrictions on the following constituent	2
Must be followed by a suffix, inflectional morpheme or link beginning with a vowel	3

Kind of constituent	3-rd index
Link	0
Inflectional morpheme	1
Word or virtual word	2

Affix connectible with (ordered according to decreasing affinity)	3-rd index
Noun	3
Adjective	4
Cardinal numeral	5
Verb	6
Preposition - adverb	7
Adjective - adverb	8
Noun - adjective	9
Adjective - noun	10
Verb - noun	11
Noun - adjective - adverb	12
Noun - adjective - verb - other parts of speech	13
Noun - verb - adjective - other parts of speech	14
Adjective - noun - verb - other parts of speech	15
Verb - noun - adjective - other parts of speech	16
Verb - adjective - noun - other parts of speech	17
Adjective - verb - noun - other parts of speech	18
Noun = verb - adjective - other parts of speech (i.e. equal affinity to nouns and verbs)	19

Part of speech (except for links)	4-th index
Noun or substantive pronoun	0
Adjective or adjunct pronoun	1

Part of speech (except for links)	4-th index
Cardinal numeral	2
Verb	3
Adverb	4
Preposition	5
Conjunction	6
Recessive prefix	7

By a SUBSTANTIVE pronoun I understand for the determination of the 4-th index a pronoun which can ONLY occur as a substantive pronoun, by an adjunct pronoun a pronoun which can occur both as an adjunct pronoun and as a substantive pronoun, because pronouns only occurring as adjunct pronouns do not exist in Dutch. Compare [1] pp. 87 - 93, the distinction of actual and potential function.

The analytical program does not take account of the possibility of occurrence of articles and interjections in compounds. Pronouns rarely occur in compounds and completely behave like nouns and adjectives. Therefore they are put on a level with them.

Because the purpose of the analytic program is analysis, for each of these indexes it has to be proved, that it is necessary for the analysis at least in some cases.

The 1-st and 2-nd indexes exclude analyses in which a morpheme occurs in a position in which it is not allowed to stand by its indexes.

A 2-nd index 3 occurs with by-forms of morphemes before vowels (see 3.3.2.; e.g. *huiz* as a by-form of *huis* (= *house*) occurring in *huiz-en* (= *house-s*)). It excludes analyses in which a morpheme with this index is followed by a consonant, e.g. *nav-looi-en* (= *to nave-tan*) instead of *na-vlooi-en* (= *catch-ing fleas afterwards*).

The 3-rd index can sometimes help to distinguish homonymous constituents of different kinds of constituent.

The 4-th index sometimes helps to distinguish homonymous constituents, especially affixes and inflectional morphemes, which can be connected only with certain parts of speech. E.g. *drinker = drink-er* (English *drinker*), while there are eight homonymous constituents *er*:

1) The adverb *er* (= *there*),
2) The prefix *er*, forming so-called pronominal adverbs: *er-mee* (= *with it*), *er-tegen* (= *against it*), etc.,
3) The suffix *er*, occurring i.a. in *lop-er* (= *runn-er*),
4) The suffix *er* for the formation of the comparative of adjectives (as in English),
5) The suffix *er* for the formation of the comparative of adverbs,
6) *er* as an allomorph of the stem of the verb *eren* (= *to honour*),
7) *er* as an allomorph of the noun *eer* (= *honour*),
8) *er* as an allomorph of the suffix *eer* often added to verbs borrowed from French (*adopt-er-en* = *to adopt*).

The syntactic morphemes 4 and 5 are excluded here, because *drink* is a verbal stem and *er* as a comparative suffix can only be added to adjectives and adverbs (so because of the 3-rd index of *er* and the 4-th index of *drink*). The syntactic morphemes 2, 6, 7 and 8 are excluded by their 2-nd index 2 or 3. This requires that *er* be followed by other constituents, while in *drinker* *er* stands at the end of the word.

Finally, the syntactic morpheme 1 is excluded by the 1-st index 1.

3.3. ALLOMORPHS

3.3.0. INTRODUCTION

In linguistics the notion ALLOMORPH is often taken in a wider sense and all morphs (strings of letters in written language, strings of phonemes in spoken language) expressing the same syntactic or semantic function in connection with different coconstituents (e.g. *en* en *s* as plural morphemes in Dutch) are considered as allomorphs. I take the word ALLOMORPH in a more restricted sense and consider as allomorphs only morphemes B_1, ..., B_n alternating according to FIXED RULES on the base of the letters of the preceding constituent A and the following constituent C and sometimes on the base of the kinds of constituent of A and C, but never on the base of the kind of constituent of B_1, ..., B_n themselves (so what Nida calls PHONOLOGICALLY DEFINABLE ALLOMORPHS, [9] p. 14). We ought to speak of GRAPHEMICALLY definable allomorphs, since we analyse WRITTEN language. The

allomorphs are treated in a separate section, because for morphemes of different kinds of constituent the choice among allomorphs is often determined by the same rules.

3.3.1. ALLOMORPHS CONDITIONED BY PRECEDING CONSTITUENTS

In general the superlative is formed by addition of *st* to the positive. But if the positive ends by *s*, only *t* is added. Therefore we also adopt *t* as a superlative ending into the analytic lexicon.

Though there are two homonymous morphemes *t*, namely the ending *t* of the 2-nd and 3-rd person singular of the present tense of the verb and the affix *t* of the passive participle (for reasons we shall see we do not consider the participles as inflectional forms, but as derivatives of the verb, see 3.5.), this solution does not yield ambiguities where they would not also rise with separate adoption of these superlatives in the analytic lexicon. If the constituent A preceding *t* is an adjective, *t* is a superlative ending; if A is a verbal stem, *t* is one of the two other morphemes. A homonymy, because a word may be both a verbal stem and an adjective, is not eliminated by separate adoption of the superlative (*wijs-t* is the superlative of the adjective *wijs* (= *wise*) or the 2-nd or 3-rd person singular of the present tense of the verb *wijzen* (= *to show*)).

Yet the adoption of the superlative ending *t* in the lexicon necessitates a procedure mistest (l. 426 - 430) in the procedure declaration compl to eliminate analyses, in which the superlative ending *t* would follow an adjective not ending by *s*.

The apostroph as a genitive ending is adopted into the lexicon (the apostroph is considered as a letter in the lexicon).

[14] mentions the following cases of insertion of an apostroph (the first constituent is called A, the second one B):

1) If A is a noun ending with *a*, *i*, *o* or *u* and B is the inflectional morpheme *s*, or A is a noun ending with *y*, having the phonetic value *i* and B is the inflectional morpheme *s* or the diminutive suffix *tje* (*auto* - *auto'tje* = *auto* - *little auto*),

2) If A ends with an open *e* written with one *e* and B is the plural ending *s* (*aloë* - *aloë's*),

3) If A ends with a vowel + *h* and B is the genitive or plural ending *s* (*fellah - fellah's*).

In these cases it is convenient to consider *'s* and *'tje* as allomorphs of *s* and *tje*.

3.3.2. ALLOMORPHS CONDITIONED BY FOLLOWING CONSTITUENTS

The treated constituent is called B; its by-form is D. The basic form of the morpheme B is the one, in which it occurs at the end of the word. All other forms are called by-forms.

For the determination of the by-form, B must be dissected into B_1, B_2 and B_3, if it ends by a consonant and is no bastard word ending by *ief* or *ies* (e.g. *actief = active, precies = precise)*, in which the *f* or *s* is false (i.e. is changed to *v* or *z* before a vowel). B_3 is the final consonant block of B, i.e. the longest continuous string of only consonants containing the final consonant of B. *j* is not considered as a consonant here, because a *j* followed only by consonants in the same syllable can only be the second part of the digraph *ij* (pronounced approximately as the *ei* in French *abeille*).

B_2 is the last vowel block in B. A RAW VOWEL BLOCK is a string of writing signs maximal with respect to the properties that

1) Every writing sign is a vowel or *j*;
2) Every *j* follows an *i*;
3) Either only the first writing sign or no writing signs at all has a trema.

The second condition is theoretically not entirely correct, because there are also cases, in which *i* and *j* occur together and do not form together the digraph *ij* (*ski-jumper = ski jacket*). But since this case is difficult to recognise for a computer and has no importance for the determination of the by-form, I do not take account of it.

The VOWEL BLOCK is obtained from it by dropping the trema on the first letter, if any.

B_1 is everything in B preceding B_2. The form obtained by juxtaposition of X and Y is noted as X+Y. Then $D = B_1+D_2+D_3$, in which D_3 is the final consonant block and D_2 is the last vowel block in D.

The construction of D proceeds in five stages:

1) The determination of B_3,

2) The determination of B_2,

3) The construction of D_3,

4) The construction of D_2,

5) The construction of $D = B_1+D_2+D_3$.

I suppose the constructions 1) and 2) to have been completed.

3) There are three cases, in which D_3 differs from B_3:

31) By the transition of a false *f* into *v* (*actief - activ-iteit*)

32) By the transition of a false *s* into *z* (*buis - buiz-en = tube - tube-s*).

33) By the duplication of the final consonant (*blok - blokk-en = block - block-s*).

4) In two cases D_2 can differ from B_2, namely:

41) By the transition of *aa*, *ee*, *oo*, *uu* respectively into *a*, *e*, *o*, *u*, if B ends with xy+xy+yz, where xy represents *a*, *e*, *o* or *u* and yz is one consonant (*haar - har-en = hair - hair-s*),

42) By the transition of *iee* into *ië*, if B ends with *iee*+xy, where xy is a consonant (*definieer - definiër-en = define - to define*).

A completely mechanical determination of the by-forms is impossible by the completely irregularly inflected words.

But if one has a lexicon indicating

1) The part of speech of each word,

2) The plural of each noun,

3) The 2-nd inflectional form (with added *e*: *groen - groene = green*) of each adjective, if it is irregular,

4) The past of each verb,

(the last three requirements are fulfilled e.g. by [13]), it is easy to write a program for the determination of the by-forms of the overwhelming majority of the words.

The by-form of the noun is found by subtraction of *en*, *'s* or *s* from the plural; the by-form of the adjective is found by subtraction of *e* from the 2-nd inflectional form, if it differs from the 1-st inflectional form.

The basic form of the verb is found by subtraction of *de* or *te* from the past. The by-form is formed starting from the infinitive I according to the following instructions (the string of letters formed by subtracting

the final string of letters B from the string of letters A is called A - B):

1) I - *n* is called J. If the last letter of J is not an *e*, J is the by-form.

2) If the last letter of J is *ë*, J - *ë* is the by-form.

3) In the sequel the last letter of J is *e*. J - *e* is called K. If the last letter of K is a consonant, *u* or *y*, K is the by-form.

4) If the last letter of K is *a*, *e* or *o*, J is the by-form.

5) In the sequel the last letter of K is *i*. The penultimate letter of K is called vl. If vl is a consonant, the by-form is J. If vl is a vowel, the by-form is K.

Then we explore,whether the relation between basic form and by-form belongs to one of the cases just mentioned. In the cases in which this relation does not belong to them, the by-form so found is rejected and afterwards the by-form is determined "by hand", because the mechanical determination of the by-form fails in the case of completely irregular inflection, e.g. *komen - kwam (come - came)*.

It is true, that the by-forms of constituents ending with consonants are often followed in compounds by the link *e* or by affixes or inflectional morphemes beginning with *e*. Yet it is not a good solution to adopt D+*e* and all compounds formed by suffixes beginning with another vowel than *e* instead of D in the lexicon. For the number of affixes beginning with another vowel than *e* is very great and even if the number of suffixes with which each stem can be connected would be small, the total number of suffixes involved in compounding is far too large to enable a simple procedure to decide, with which suffixes a certain word-stem can be connected. Moreover there are many compounds not sufficiently usual to be adopted into the lexicon themselves and yet sufficiently conformous to the structure of Dutch to be formed, e.g. *viss-ig (= fish-y?)*.

Six classes of bastard words have besides (or instead of it) another by-form exclusively on the base of their ending (and in one case on the base of their form in the original language), which can be obtained by substitution of the ending. A translation is superfluous, since these words differ very little from the same words in English.

Bastard word X (basic form)	Example of X	By-form Y	Example of compound of Y
X = A+*eer* (X is a verb)	*ador-eer*	Y = A	*ador-atie*
X = A+*ie*, derived from a French word on *ion* or a Latin word on *io*	*natie*	Y = A+*ion*	*nation-aal*
X = A+*teit*	*qualiteit*	Y = A+*tat*	*qualitat-ief*
X = A+*air*	*vulg-air*	Y = A+*ar*	*vulg-ar-iteit*
X = A+*eel*	*univers-eel*	Y = A+*al*	*univers-al-iteit*
X = A+*ie*+B_3	*reflex-ie-f*	Y = A+*i*+D_3	*reflex-iv-iteit*

(D_3 differs from B_3 in the same way as on p. 31).

[13] treats only ORTHOGRAPHICAL modifications, i.e. different ORTHOGRAPHY of morphemes with unchanged PRONUNCIATION (except the alternation of *f* with *v* and *s* with *z*). But for us this limitation is useless, because the computer program treats only WRITTEN words.

Further [14] mentions the following morphographemic modifications in compounding:

1) The transition of *ine* into *ien* before the diminutive suffix *tje* *(machine - machientje, machine - little machine)*,

2) The transition of *é* and *ée* into *ee* before the diminutive suffix *tje* *(logée - logeetje, guest - little guest)*,

3) The duplication of the final vowel of A, if

31) A is a noun ending with *a, o* or *u*, B is the diminutive suffix *tje* or the suffix (not the inflectional morpheme!) *s* *(Wolvega - Wolvegaas;* Wolvega is a Dutch village),

32) A is an adjective ending with *a, o* or *u*, B is the superlative ending *st* or the ending *s* of the 3-rd inflectional form *(cru - cruus; cru = crude;* about the 3-rd inflectional form see 3.5.),

33) A is a verbal stem ending with *a, o* or *u*, B is the ending *t* of the 2-nd and 3-rd person singular of the present tense or the ending *n* of the plural, the affix *n* of the infinitive, *nd* of the active participle or *d* of the passive participle (in the lexicon the verbs are not adopted with their infinitive, but with their stem. In deviation of traditional grammar

the infinitive and the participles are not considered as inflectional forms, but as derivatives: *gaa - gaa-nd = go - go-ing*),

4) That the diminutive of *parachute* is *parachuutje*,

5) The transition of *ng* into *nk* before *elijk* and the diminutive suffix *je (afhang-en - afhank-elijk, to depend - depend-ent)*,

6) The irregular diminutives

dejeuner - dejeuneetje,

diner - dineetje,

souper - soupeetje,

7) ([14], pp. LXVIII - LXIX) The compound has a capital, while the first constituent does not have it, or inversely:

a) Some proper nouns, etymologically compounds, have a capital, while their constituents do not have it (*Kalverstraat* lit. *Calf Street*, *Koningsplein* lit. *King's Place)*,

b) Compounds with a proper noun as first member which can be considered as common nouns are written with a small letter *(aagt-appel* lit.*Aagt's apple*, *adams-appel = Adam's apple*,...)

c) "But one should write with a minuscle all compounds with as first member a person's name indicating the inventor, discoverer etc. of the thing indicated by the whole *(dieselmotor = Diesel motor, erlenmeierkolf = Erlenmeier recipient, newtonringen = Newton rings, priesnitzverband = Priesnitz bandage, schakellijm = Schakel glue, vanderwaalskrachten = Van der Waals forces)*and their shortened forms (a *diesel*, an *erlenmeier* etc.).

. . . . . .

d) In derivatives like *dahlia, darwinisme, flamingant, fuchsia, guillotine, jeremiade* the thought of the proper noun notion is so weakened, that the capital is no longer required."

e) Some titles, which are compounds etymologically, like *Hoogheid (= Highness)*, are written with a capital, while their constituents are not.

8) "Spellings like *rij-dier* (from *rijd-en = to ride* and *dier = animal*), *lei-draad (= introduction*, from *leid-en = to lead* and *draad = file)* must be compared with *rij-broek (= riding trousers), lei-boom (= tree trained on trellis-work)* etc. (without *d* between diphthong and consonant)." ([14], pp. XLIII and XLIV).

In the cases 1-6 the morphographemic alternation of a constituent only occurs in compounding with one or two other constituents (the verbs

mentioned in 33 are also irregular for other reasons). But adoption of a by-form for only one or two compounds does not save memory space in the computer. So in all these cases the compounds mentioned are adopted into the lexicon themselves.

In all sub-cases of case 7 the connection in meaning between the compound and its constituents is so obscure, that adoption of the compound into the lexicon is anyhow inevitable. So we shall not pay further attention to them.

In case 8 the compounds *rij-dier* and *lei-draad* are considered as compounds of the virtual words *rij* and *lei* for the large number of compounds of *rijd* and *leid* without *d*.

3.4. WORDS

This section will answer the following questions:

1) Which words has the analytic lexicon to mention?
2) Which inflectional form(s) of inflected words has it to mention?

1) The answer to this question has been given by [10] p. 6.

The words to be adopted are:

a) Non-compound words,

b) Established compounds (i.e. rather usual ones found in many lexica) whose meaning cannot be derived from those of the constituents.

The words not to be adopted are:

a) Improvised compounds (i.e. compounds made by the author of the text to be analysed, which cannot be adopted into a lexicon in principle),

b) Established compounds whose meaning can be derived from those of the constituents.

There are many so-called equiradical words which are homonymous, belong to different parts of speech, but whose meanings are very similar, may sometimes even almost be called synonymous. The clearest example is the stems of many verbs used as nouns indicating the action of that verb. But such words have to be mentioned separately in the lexicon, because in some compounds the word can only be assigned one of the two parts of speech. E.g. in *meester-zet (master's move) zet*, and consequently also

meesterzet, can only be considered as a noun and not as a verb. But this distinction produces linguistic pseudo-problems in many cases. E.g. it is not possible to decide, whether *zet* in *zet-dwang* is a verbal stem or a verbal noun. The meaning can be paraphrased both as "compulsion to move" and as "compulsion to a move". We shall see (4.3.), that on the basis of the order of the sets of indexes in the lexicon one analysis is chosen (in some cases arbitrarily from a linguistic point of view).

In order to make the Reifler calculus applicable (which instructs first to look up the longest constituent at the beginning of each compound, [10] p. 11) I adopt inflectional forms homonymous to the stem of another word themselves into the lexicon (e.g. *wagen* is a singular noun *(= car)*, the plural of *waag (= weighing-house)* or the plural form of the verb *wagen (= to risk))*.

For inflected words for each part of speech it is determined, which is the most frequent ("regular") system of inflectional morphemes, by which the inflectional forms are formed from the stem. Different inflectional morphemes, frequently used for the same function, are each adopted into the lexicon, e.g. *en* and *s* as plural morphemes for nouns. But if an inflectional morpheme only occurs with few words, e.g. *eren* as plural morpheme for nouns, it is not adopted into the lexicon, but the inflectional forms formed by them are adopted into the lexicon themselves. In 3.5. it will be indicated exactly, which inflectional morphemes are adopted into the lexicon.

For words, all whose inflectional forms are obtained by compounding these inflectional morphemes with the stem or one of its allomorphs, only the stem and its by-forms are adopted.

For the choice of the form(s) (basic form) in which a regularly inflected word is adopted into the lexicon, the principle is adopted, that the basic form must as much as possible enable one to obtain all inflectional forms, derivatives and compounds by adding morphemes and as little as possible by first subtracting letters and only afterwards adding morphemes.

This principle does not entail changes for most nouns and adjectives (nouns are adopted with their singular form , adjectives with their 1-st declension form (without added final *e*)), but entails for verbs, that they

are not adopted with their infinitive, as usual in lexica, but with their 1-st person singular present tense form. For separable compound verbs (compounds of a verb with a preposition or adverb, in which the preposition or adverb can come behind the root-verb in the case of inversion of the sentence, e.g. *op-zoek-en - ik zoek een boek op = to look up - I look up a book)* the form, in which the 1-st person singular present tense form occurs at the end of a subordinate clause, is adopted. See [2], especially pp. 34-36. An equally detailed description of the syntax of Dutch does not exist as far as I know, but the word order in Dutch is very similar to that in German. Bierwisch takes the subordinate clause order (with the finite verb at the end of the sentence) as the basic order. For this he gives many arguments, but the argument most relevant here, valid also for Dutch, is that subordinate clauses do not separate the parts of a separable compound verb.

Nouns usually have only two inflectional forms: singular and plural. The regular plurals of nouns are formed by addition of *en* to the stem.

Among the four traditional cases of Dutch the *accusative* does not differ morphologically from the nominative for any noun.

The *genitive*, if differing morphologically from the nominative, is adopted into the lexicon with a 3-rd index 8.

The *dative* (by a dative I understand only a dative differing morphologically from the nominative) practically only occurs in fixed expressions.

For *adjectives* and *verbs* things are slightly more intricate. As we shall see in 3.5., in the analytic program the comparative and superlative are not considered as inflectional forms, but as derivatives of the positive. The positive and the comparative have each three inflectional forms, the superlative two. The 2-nd inflectional form of regularly inflected adjectives is formed by suffixing *e*, the 3-rd inflectional form (not occurring with superlatives) by suffixing *s*.

The genitives and datives *(op heter daad*, lit. *on hot deed, in flagrant delict, in koelen bloede = in cool blood)*, only occurring in fixed expressions, are considered as irregularities. Their adoption is still more necessary, because many of these genitives and datives could be confused morphologically with comparatives.

The Dutch verb has two stem forms: the stem and the preteritum stem, because the passive participle is not considered as an inflectional form (see 3.5.).

The preteritum has two inflectional forms, the singular form and the plural form. The preteritum stem is the singular form. The plural is usually formed by adding *en* or *n* to the singular. If the plural form of the preteritum is irregular with regard to the singular form, that does not influence the 2-nd index of the stem, but the preteritum stem gets a 2-nd index 1, so e.g.

was 21023
waren 21023

All other inflectional forms (PRESENT STEM FORMS) are formed from the main stem (in theory an exception would perhaps have to be made for the singular form of the conjunctive of the preteritum, but because it coincides with the indicative of the preteritum in most cases and in the other cases it has a by-form coinciding with the indicative of the preteritum, I shall not pay further attention to it).

By the conjunctive of present I understand the form of the 3-rd person singular of the conjunctive of present, because the other forms coincide with those of the indicative. The inflectional forms for *gij* (a rather archaic 2-nd person personal pronoun, used both for one person and for more) and the plural form of the imperative (Dutch has an especial plural form of the imperative, but the imperative coinciding with the stem is now most usual also in addressing more persons) are neglected.

The stem of the verb is indicated by st. Then the inflectional forms of the regular verbs have the following forms:

Indicative present singular 1-st person = st
Indicative present singular 2-nd person = st+*t*
Indicative present singular 3-rd person = st+*t*
Indicative present plural = st+*en*
Indicative preteritum singular = st+*de* or st+*te*
Indicative preteritum plural = st+*den* or st+*ten*
Conjunctive present = st+*e*
Imperative = st

The following cardinal numerals are adopted into the lexicon: the numerals from 1 until 20 successively: *een, twee, drie, vier, vijf, zes, zeven, acht, negen, tien, elf, twaalf, dertien, veertien, vijftien, zestien, zeventien, achttien, negentien, twintig; dertig (= thirty), veertig (= forty), vijftig (= fifty), zestig (= sixty), zeventig (= seventy), tachtig (= eighty), negentig (= ninety), honderd (= hundred), duizend (= thousand), miljoen (= million), miljard (= thousand million), biljoen (= million times million)* (higher numerals like *triljoen* (= 10^{18}), *quadriljoen* (= 10^{24}), do occur, but do not belong to common speech, and are mostly replaced by indications by digits in publications, mainly scientific ones, where they would be necessary).

In theory the by-forms of *vijf, zes, elf* and *twaalf* are, respectively, *vijv, zess, elv* and *twaalv*, but they are not adopted into the lexicon, because their only compounds are *vijv-en, zessen, elven* and *twaalven (we zijn met ons vijven = we are five).*

Only the irregularly formed ordinal numerals are adopted: *eerste (= first), derde (= third).*

In theory the numerals have genitives *(enerzijds = on one hand, enerlei = of one kind, tweeërlei = of two kinds, drieërlei = of three kinds)*, but they are not used productively in compounds. So these genitives are not mentioned and the compounds formed by them are separately adopted into the lexicon.

beide (= both) is not considered as a numeral in deviation from [11] (p. 122), because it lacks the only compounding potentiality characteristic for numerals, that of the formation of ordinal numerals.

Only the following pronouns are productive in compounds:

1) The nominatives of the personal pronouns (*iktaal (= I-language)* and *hij-taal (= he-language)*, [8] p. 31),
2) *zelf (= self),*
3) *al, alles, alle (= all),*
4) *aller,*
5) *niets (= nothing), niemand (= nobody).*

So the compounds of all other pronouns are adopted into the lexicon and the words themselves get 1-st and 2-nd indexes 1.

aller, though an independent word in origin (genitive plural of *alle*), is especially frequent in compounds of superlatives and so it must rather be considered as a prefix than as an independent word, moreover because it often changes the meaning of the superlative from a relative superlative into an absolute one *(de erbarmelijk-ste toestand = the most miserable situation; een aller-erbarmelijk-ste toestand = a most miserable situation)*. So I adopt *aller* into the lexicon as *aller* 2087.

This entails, that all compounds of *aller* with other adjectives and adverbs, e.g. *aller-heilig-en* (= *All Saints' Day*; this must be rather considered as a compound of *aller* with the noun *heilige-n*, but *heilig* (= *saint*) is an adjective) must be adopted into the lexicon separately.

3.5. INFLECTIONAL MORPHEMES

The inflectional morphemes are treated before the virtual words and affixes, because they are relatively few and their previous treatment makes that of the virtual words and affixes much clearer.

This section starts with a discussion of the definition of inflectional morphemes, then decides, which forms are considered as inflectional forms of one word in virtue of this definition, and ends by an enumeration of the inflectional morphemes.

With regard to the criteria for the distinction of inflection and derivation, and in connection with that the distinction between affixes and inflectional morphemes, the authoritative linguists do not quite agree.

[9] gives on p. 99 as a criterion for inflectional forms, that they do not have the same external distribution as the simplest members of their class (EXTERNAL DISTRIBUTION CRITERION; from the opposition *een groen huis = (a green house)* against *de groene boom (= the green tree)* it appears clearly, that *groene* must be considered as an inflectional form of *groen* in virtue of this criterion) and gives moreover as a criterion, that inflection cannot transform a word into a word of another part of speech (CRITERION OF PART OF SPEECH).

[3] gives as a criterion on pp. 222-224, that inflection leads to CLOSURE: after addition of inflectional morphemes either no bound morphemes at all can be added, or only some from a well-defined set of bound

morphemes (morphemes not occurring as independent words) and then the word-form so formed is closed (CLOSURE CRITERION).

The part of speech criterion and the closure criterion are most relevant for our morphological analysis, because the part of speech and the closedness are relevant for the possibility of combination of a constituent with other constituents to still longer compounds. Therefore they are given priority above the external distribution criterion. This has the following consequences:

1) The grades are considered as derivatives of the positive in virtue of the closure criterion *(ver-slecht-er-ing = deter-ior-is-ation; slecht = bad; er* = comparative suffix, *ver* = prefix denoting here *to become, ing* = nominalisation suffix for verbs *(~ing* or *ation))*, though they would be inflectional forms in virtue of the external distribution criterion (a comparative followed by a *than*-clause cannot be replaced by a positive).

2) The ordinal numeral is considered as a derivative of the cardinal numeral in virtue of the part of speech criterion, because the ordinal numeral quite behaves as an adjective, while the cardinal numeral can act both as a substance word (with the grammatical function of a noun, e.g. *Drie passeerden de straat = three passed the street)* and as a determiner of a noun.

3) The INFINITIVE and PARTICIPLES are considered as derivatives of the verb in virtue of both the part of speech and the closure criterion, the infinitive as a noun, the participles as adjectives.

All passive participles beginning with *ge* are adopted into the lexicon, because the Reifler calculus would otherwise produce wrong analyses far too easily, even in the case of adoption of the prefix *ge* into the lexicon. In a test it appeared, that 14 among 45 passive participles found in two fragments, one newspaper article (Nieuwe Rotterdamse Courant, 21 September 1963) and one fragment from [12], III p. 306, are analysed erroneously by the Reifler calculus, i.a.
gewei-ger-d (correct analysis: *ge-weiger-d = refuse-d; gewei = antlers; ger = gore; d* =passive participle suffix); *over-gel-ever-d (over = over; gel = yellow; ever = wild boar; d* = passive participle suffix; correct

analysis: *over-ge-lever-d = surrender-ed*); *ger-e-d-ig-eer-d* (*ger = gore*; *e* has many different functions; *d* = passive participle suffix; *ig* (~y) = suffix forming adjectives with a wide sphere of meaning; *eer* = suffix added to verbal stems of Romance origin; *d* = passive participle suffix; correct analysis: *ge-redigeer-d = redact-ed*). This is caused by the fact, that because *ge* is such a short morpheme, very many words among those beginning with *ge* have a longer initial segment also occurring in the lexicon.

In virtue of these criteria the following forms are considered as inflectional forms of one word:

1) All forms of indicative, conjunctive and imperative of the verb,
2) Singular and plural of the noun,
3) The cases of the nouns, adjectives, articles and pronouns,
4) The 1-st inflectional form (without *e*), the 2-nd inflectional form (with *e*) and the 3-rd inflectional forms (with *s*) of the adjectives and the analogous inflectional forms of some adjunct pronouns.

The endings *er* for genitive or dative and *en* for dative or accusative are not adopted into the lexicon, because they are used far too sporadically.

From a linguistic point of view it is preferable to consider the final *n* in plural past forms after a past morpheme *de* or *te* as a morpheme: *zij klaag-de-n = they complain-ed* (against *hij klaag-de = he complain-ed*). Yet *den* and *ten* are considered as morphemes, because this simplifies the analysis (the rule, that in Dutch two inflectional morphemes never follow each other, then is without exception).

Then the list of inflectional morphemes is as follows:

de	21013	*e*	21013	*n*	20010	*s*	20011	*ten*	21013
den	21013	*en*	21010	*n*	21013	*t*	21013	*'*	20010
e	20011	*en*	21013	*s*	20010	*te*	21013	*'s*	20010
								's	21011

3.6. VIRTUAL WORDS

Among the virtual words there are many "international" ones, like *theo-*, *bio-*, *-loog* (= *-logist*) with an origin in Latin, Greek or the Romance languages. Especially for them one should realise, that they serve for the analysis of *Dutch* words and are no fragments of e.g. a Greek, Latin or French lexicon. This is relevant for both the form in which they are adopted into the lexicon and their indexes.

As to the *form* an appeal to the language of origin is already impossible merely because one would not know for a word of Latin or Greek origin, which of the inflectional forms of the word in the language of origin one should take.

Some international words are adopted twice, once as a noun and once as an adjective, e.g. *morf* (noun) = *form* and *morf* (adjective) = *with a ... form*, since the most frequent type of compound in Dutch is that, in which the preceding constituents are determiners of the last one and so the compound has also the part of speech of the last constituent.

But there are many virtual words of Greek origin, which were nouns in Greek and act as nouns in many compounds, but also occur as last constituents of adjectives. In order to create as few exceptions as possible to the main rule, that the compound has the part of speech of the last constituent, we adopt them into the lexicon not only as virtual nouns, but also as virtual adjectives.

The 1-st, 2-nd and 3-rd indexes do not need further explanation. The 4-th index is determined by the part of speech of the "translation". E.g. the translation of *mono-* is *one*, because this gives the correct meaning to the compounds formed by it (*mono-the-ism = worship of one god*). Therefore the 2-nd index of *mono* is 2.

Virtual words with any connection in form with their "translation" are in one of the 4 following cases:

1) Formally the translation is a regular inflectional form (*financi - financiën = finances, hersen - hersenen = brains*), derivative (*bruik - gebruik = use, weduw - weduwe = widow*) or compound (*kerst - kerstmis = christmas, lief - liefheb = love*.) of the virtual word;

2) Formally irregular inflectional forms (*beender - beender-en = bones, schep - schepen = ships* with *schip = ship*) or compounds (*glij = slide* (verb) - *glij-baan = slide* (noun), with *glijd-en = slide*) can be considered as inflectional forms (respectively compounds) of the virtual word;
3) Formally the virtual word is a compound which does not occur alone but does occur as a constituent of still longer compounds: (*sterkstroom - sterke stroom = strong current*);
4) Formally the virtual word can be considered as a derivative by internal change, analogous to derivatives truly occurring: *braak - breek-ing = break-ing, name - neem-ing = tak-ing* (compare *wraak* (= *revenge* as a noun) as a derivative of *wreek* (= *revenge* as a verb)).

On the basis of [13] I made a slip system of virtual words. The only virtual words not adopted were:

kolder V [13], VII p. 5114, since it has only four compounds:
kolder-gat = oblong aperture in the ship's deck in which the kolder-stok can be moved going and returning
kolder-stok = kolder-stang = stick moved in the kolder-gat
kolder-schijf = wheel fixed excentrically on an ax converting the rotating movement of an ax into a going and returning one
kolder VI ([13], VII p. 5114), since it has only four compounds:
kolder-gang = kolder-molen = kolder-werk = mill in which the place of the upper grind-stone is occupied by two rollers whose axes, which are the continuation of each other are fixed in the mid to a vertical ax
kolder-steen = cylindrical stone serving as a runner in a kolder-mill

3.7. AFFIXES

By a nominal, adjectival or adverbial affix I mean an affix forming, respectively, nouns, adjectives or adverbs.

The compounds formed by a nominal suffix are nouns and so most of them have plurals. If we call the stem A and the suffix B, the plural of A+B can usually be written A+C, in which C only depends on B and not on A (an exception is e.g. *kind-er-tje-s = little children; kind = child, kind-er-en = children, kind-je = child, tje* is a diminutive suffix and *s* a plural end-

ing). C can then very conveniently be called the *plural* of B. If C is "irregular" with respect to B (i.e. is not obtained from B by addition of *en* or *s*), it must be adopted into the lexicon separately (e.g. *heden* besides *heid*; *heid* is a nominalising suffix added to adjectives, roughly corresponding with English *ness*).

While some virtual words, like *bruik* (= *use*) can occur both at the beginning and the end of a word (*bruik-baar* = *us-able; mis-bruik* = *ab-use*), all affixes are either prefixes or suffixes. It is true, there are HOMONYMOUS affixes, one of which is a prefix and one a suffix, e.g. *in* as a negative prefix (as in English: *in-congruent*) and *in* as a suffix indicating a female being (*koning* = *king, koning-in* = *queen*). But they have so entirely different meanings, that it is clear, that one should speak of homonymous affixes and not of one and the same affix.

Prefixes and suffixes are distinguished by the 1-st index. For prefixes the 1-st indes is 0, for suffixes 2. Because it does not occur, that a prefix has an allomorph, whose occurrence depends on preceding constituents. But there are a few suffixes like *abil* and *ation* only occurring before following constituents (since they are allomorphs of respectively *abel* (= *able*) and *atie* (= *ation*)). For such suffixes the 2-nd index is 3.

For affixes the usefulness of the indexes has to be proved separately.

The indexes 3 and 4 help to distinguish homonymous constituents, if two affixes, the latter of which can only be connected to words of a certain part of speech, follow each other. E.g. *rek-baar-heid* = *dilat-abil-ity*. In general *baar* can be a noun (with several meanings), an adjective (= *bare*), a verbal stem (= *bear* in the sence of *bear children*) and an adjectival suffix behind verbs corresponding to *able*, but in this compound it can only be an adjective or an adjectival suffix behind verbs, because the suffix *heid* can only be connected with adjectives.

The 1-st and 2-nd indexes serve to distinguish homonymous affixes, one of which is a prefix and the other a suffix. So in *koning-in* (= *queen; koning* = *king*) *in* can only be the suffix *in* and not the prefix.

Only productive affixes are adopted. Therefore the following affixes are omitted:

de, occurring in *lief-de* (= *love* (noun); the virtual noun *lief* occurs in

ge-lief-d (= *love-d*) and *lief-heb* (*love* as verb)) and *vreug-de* (= *joy*); *egge*, occurring almost only in *diev-egge* (= *female thief; dief = thief*); the prefix *er* before verbs.

All the same the diminutive suffix *lijn* is not adopted for its small productivity and in order to prevent confusion with the other meaning of *lijn* (*line* or *linen*).

The dominant prefixes have the following sub-types (this sub-division is given, because it will play a role in some later considerations, see 4.2.3.):

1) Preverbal prefixes, prefixes connectible with verbs. They are:
be 2163
ge 2060
ont 2143
ver 2183
(*ge* as a formator of nouns from verbs (*ge-bouw = build-ing*) from *bouw = build*), not as morpheme for the formation of passive participles (see 3.5.)).

2) Formators of pronominal adverbs. They are connected with prepositions and adverbial by-forms of prepositions, moreover with *heen* (*daar-heen = thither*), and form so-called pronominal adverbs, e.g. *hier-voor = before this*. They are
daar 2074
er 2074
hier 2074
waar 2074

The pronominal adverbs are translated according to the scheme (the translation of X is indicated as Trad(X)):
Trad(A+B) = Trad(B) Subs(A),
in which
Subs (*daar*) = *that*
Subs (*er*) = *it*
Subs (*hier*) = *this*
Subs (*waar*) = *what*

3) Prefixoids.

If both A and B are true words, their compound C = A+B usually has the part of speech of B. But if A is a dominant prefix, A determines the part of speech of C.

Now, there are some adverbs and prepositions forming verbal compounds together with following nouns or adjectives. So in these compounds the adverb or the preposition determines the part of speech of the compound as a whole and so behaves as a prefix. Hence the name PREFIXOID. I call the compounds formed by them PREFIXOIDAL VERBAL COMPOUNDS.

The prefixoids are noted with 4-th index 3 and 2-nd index 2. The 1-st index indicates, whether they can be connected with nouns or with adjectives.

The prefixoids have to be adopted into the lexicon, since prefixoidal verbal composition is productive. In [4] p. 24 I found *om-stam-den = surrounded with stems*, the passive participle of a verb *om-stam = surround with stems*.

Before I shall treat the prefixoids further I have first to define, when a morpheme functions as a word and when as a prefixoid in the case, that B can be both a verb and a noun or adjective. For this I must first know, whether C is a compound of B as a verb or of B as a noun or adjective. I take as a general criterion, that C is considered as a compound of B in that meaning, by which its meaning can be paraphrased in the simplest way. In a non-formalised language this criterion is not entirely exact, but in the case, that the different B's are pure homonymes, it is yet sufficient for a decision.

fris (verb) = *melt cast iron in an open furnace and change into malleable iron by removal of carbon*

op-fris (transitive) = *make fresh* (*fris = fresh*)

op-fris (intransitive) = *become fresh*

The given definition of *op-fris* with the aid of the ADJECTIVE *fris* is simple, while a definition of *op-fris* with the aid of the VERB *fris* would be very intricate, even if it were possible.

In many cases this criterion is also sufficient for the distinction of equiradical words. There are also cases, in which a compound has two mean-

ings and is a compound of two words in one case and a prefixoidal compound in the other case, e.g.

om	*+hoepel*	→	*om-hoepel*
prefixoid	noun		transitive verb
	hoop		*surround with a hoop*
			prefixoidal compound

om	*+hoepel*	→	*om-hoepel*
preposition	intransitive verb		intransitive verb
around	*trundle a hoop*		*trundle a hoop around something*

From these definitions it appears immediately, that *om-hoepel* is a compound of the noun *hoepel* in the first meaning and a compound of the verb *hoepel* in the second meaning.

The prefixoidal compounds of all but eight prepositions and adverbial by-forms of prepositions are few and unproductive. So these compounds are adopted into the lexicon themselves and the prepositions and adverbs only as words, not as prefixoids.

Only eight prepositions and adverbial by-forms of prepositions have a relatively great number of prefixoidal compounds (behind each stands the number of compounds found in [5]), namely *af* (*~from*, 31), *door* (= *through*, 6), *in* (= *in*, 10), *om* (= *around*, 13), *op* (= *on*, 26), *over* (= *over*, 18), *toe* (= *to*, 5), *uit* (= *out*, 24). Only *toe* is not adopted into the lexicon as a prefixoid, because the relation in meaning between stem and compound is too different for different prefixoidal compounds.

erken-telijk (= *grateful*) and *erken-tenis* (= *acknowledge*), both derived from the verb *erken* = *acknowledge*, are irregularly formed and so they are adopted into the lexicon separately.

The 3-rd index contains information about two different things, namely the POSSIBILITIES OF COMBINATION of the affix with different parts of speech and the affinity of the affix to these parts of speech. The 4-th index and the possibilities of combination are determined on the base of relatively frequent and productive derivation types. It may happen, that a few compounds, in which either the part of speech of the coconstituent or the part of speech of the compound does not agree with the requirements of the

3-rd and 4-th indexes, are formed with the affix. These compounds have to be adopted into the lexicon separately.

E.g. the main function of *heid* is the formation of abstract nouns from adjectives. So *heid* gets a 3-rd index 4 and a 4-th index 0. Then e.g. *mens-heid* (= *man-kind; mens = man*) and *al-heid* (= *universe*) have to be adopted into the lexicon separately, also for the semantic irregularity of the first one (*mensheid* has not an abstract meaning, *"being a man"*, but a collective one *"the collectivity of all men"*).

For *ge* the 3-rd index is 6 and the 4-th index is 0. So all verbs formed by *ge* have to be adopted into the lexicon separately, also because no fixed rules for the relation in meaning between the original and the derived verb can be given.

Still more deviating are: *apart-je*, lit. *little apart = entre-nous; tegen-heid*, lit. *against-ness = contrast, resistance, repugnancy, antipathy, disaster; uit-je*, lit. *little out = trip.*

3.8. LINKS

The meaning of the 1-st, 2-nd and 3-rd indexes has been explained in 3.2.

The 1-st and 2-nd indexes are of course 2 for all links in the meaning in which they have been defined in 3.2.

For analysis of the compound words it is relevant to know, to which kind of constituent and part of speech the constituents, which precede and follow the links, can belong. Unlike for affixes and inflectional morphemes for links the usefulness of the 4-th index for this could be doubted, because the links are so few, that except *en* and *n*, which are allomorphs of each other, there would only be one link for each value of the 4-th index and so the condition "if the 4-th index is 0" could be replaced by "if A = *e*". Yet the 4-th index for links is conserved for three reasons:

1) Because after the dissection into morphemes the structure of the words is constructed exclusively by their indexes (apart from the local procedure "mistest" in the procedure compl in the analytic program l. 426 - 430 and the test, whether a morpheme with a 2-nd index 3 is followed by a vowel, l. 952 - 953) and so the program would become more complicated if one would

deviate from this in this case;

2) Because two sets of indexes are stored in one machine word, so that the omission of the 4-th index would not save memory space;

3) Because in the lexicon punching-tape each index has its array-index by its position in the set of indexes and so the omission of the 4-th index would entail, that the 3-rd index would be considered as the 4-th index.

There are five links (the indexes of each, inclusively the 4-th indexes formally assigned, are indicated)

e 22000

en 22003

n 22003

o 22001

s 22002

s between two nouns or behind a noun and before an adjective (e.g. *aanbevel-ing-s-waardig = recommend-able; aanbevel-en = to recommend; aanbevel-ing = recommend-ation; waardig = worthy*) is always considered as a link, because it cannot be considered as a genitive ending after historically feminine words (modern colloquial Dutch shows a strong tendency to treat almost all non-neuter inanimate nouns as masculine, but formerly and still in most written language the nouns formed by the suffix *ing* are feminine; a genitive of *de aanbeveling*, if any, would have to be *der aanbeveling* and never *des aanbeveling-s*) and it is then simplest never to consider it as a genitive ending in the middle of a word.

[14] p. LIX: "The intermediate sound *e(n)* in compounds with as first member a person name not indicating a determined female person, is written *en*.

Examples: *helden-daad* (= *heroic deed, held = hero, daad = deed*), *her-en-hoed* (= *gentleman's hat, heer = gentleman, hoed = hat*), *vorst-en-kroon* (= *princely crown; vorst = prince, kroon = crown*), *weduwe-n-kap* (= *widow's cap, weduwe = widow, kap = cap*), *zieke-n-troost* (= *consolation of ill people; ziek = ill, troost = consolation*), etc."

Adoption of *en* as a link is necessary, because it could otherwise only be analysed as a plural ending in such compounds, what would not yield the

correct meaning of the compound.

Behind the lexical morpheme *en* in the lexicon 22003 is put before 20010; so between two nouns *en* is always considered as a link, never as an inflectional morpheme.

4. THE ANALYTIC PROGRAM

4.0. INTRODUCTION

In order to be able to construct a program for the determination of the syntactic structure of the compounds, we must know:

1) Which results will be noted and how;
2) Which compounds are POSSIBLE;
3) Which among more structural descriptions of a compound is MOST PROBABLE (in order to choose the most probable structural description of a compound, for which more structural descriptions are possible).

The first question has been answered in 1.1. So section 4 consists of 3 subsections:

4.1. The possible compounds.
4.2. The relative probabilities of alternative structural description of compounds.
4.3. The arrangement of the analytic program.

4.1. THE POSSIBLE COMPOUNDS

4.1.0. INTRODUCTION

In section 4.1. I need only treat the relation between the constituents and their immediate sub-constituents, i.e. the constituents standing in the structural tree in the constituent row immediately above them and which are parts of them. The relation between a constituent and its mediate sub-constituents need not be treated separately, because all rules for the construction of compound words can be reduced to rules for the construction of constituents from their immediate sub-constituents.

The sub-sections of 4.1. are ordered according to increasing number of constituents. I make an exception only for the links, because it depends on an especial convention, whether the next-higher constituent of a link is considered as a compound of two or three constituents.

4.1.1. COMPOUNDS OF ONE CONSTITUENT OR IMMEDIATE DERIVATIVES

A complete description of the structure of compound words must also treat immediate derivation, i.e. the formation from a word of an equiradical word (e.g. the verb and the noun *val* (= *fall*)). This way of word-formation is productive too: as it appears from the fact, that the last edition of [5] does mention the noun *blunder*, but not the verb *blunder* (as in English), the latter one must have entered Dutch not long ago. So there is also a fair chance of new immediate derivations in the future. But the semantic relation between equiradical words is very irregular. Even apart from some entirely singular cases there are e.g. at least 16 types of semantic relation between equiradical nouns and verbs. So it is practically impossible to establish a mechanical procedure for the determination of the meaning of a new-formed immediate derivative, if that of the basic word is known. Therefore we adopt all sets of indexes assigning different parts of speech to a word separately into the lexicon as much as possible and do not take account of the possibility of the formation of new words equiradical with already existing ones in the analytic program.

4.1.2. COMPOUNDS OF LINKS

By definition a link B follows a constituent A and precedes a constituent C. In order to know the compounding potentialities of B we must know, to which kinds of constituent and parts of speech A and C can belong.

For the synthesis of the meaning it does not matter much, whether we consider A+B, A+B+C or B+C as the next-higher constituent B belongs to. Because it slightly simplifies the analytic program, I consider formally A+B as the next-higher constituent B belongs to. A is the leading constituent of A+B.

A and C cannot be links, since two links following each other do not occur.

C cannot be an inflectional morpheme either, because a word-stem is never connected with an inflectional morpheme with insertion of a link.

A cannot be an inflectional morpheme either. This requires an especial explanation only for the inflectional morpheme *e* of the 2-nd inflectional

form of the adjective.

One might be inclined to analyse *ziekentroost* as *ziek-e-n-troost*, in which

ziek (= *ill*) : adjective
e : inflectional morpheme of the 2-nd inflectional form
n : link
troost (= *consolation*) : noun.

But the incorrectness of this analysis appears immediately at a periphrasis of the meaning of the compound: *consolation to an ill man*. So if we should stick to the morpheme dissection *ziek-e-n-troost*, we should have to consider *e* not as an inflectional morpheme, but as an affix. Moreover it would only yield unnecessary complications for the analytic program, if we took account of a morpheme string *e-n* homonymous with the morpheme *en*, in which *n* is a link (unnecessary, because links can be neglected for the synthesis of the meaning).

So eventually we obtain the analysis (according to the instructions of 1.1., but with omission of "leider"; for the sake of readability the sets of indexes are replaced by kinds of constituents and parts of speech):

ziek	*en*	*troost*
adjective	affix	noun
zieken		*troost*
noun		noun
ziekentroost		
noun		

So only words, virtual words and affixes remain for further examination.

A cannot be a prefix, because a prefix is never connected with a word-stem with insertion of a link.

The last morpheme before B can be a suffix, e.g. *veilig-heid-s-gordel* = *secur-ity girdle*, but in such cases the highest preceding constituent, with which the link can be connected to the next-higher constituent, is not the suffix, but the word formed by the suffix. The structure is

veilig	*heid*	*s*	*gordel*
secur	*ity*		*girdle*
veiligheid		*s*	*gordel*
security			*girdle*
veiligheids			*gordel*

veiligheidsgordel
security girdle

So in the description of A we proceed, as if A were always a word.

If two words A and C are composed with insertion of a link, of course C can begin with a prefix, e.g. *aanval-s-ge-vecht = agressive fight-ing* (*aanval = attack*), but because the part of speech of C is decisive for the possibility of such compounds and not the character of the prefix with which it begins, we need not take that into account for the requirements to C.

It occurs, however, that B is a suffix, e.g. *aanschouw-e-lijk* (= *plastic* from *aanschouw*, a more sublime word for *see*).

Now follows the table of the constituents, with which the links can be connected.

A	B	C
noun, adjective, adverb	*e*	noun, adjective, verb, nominal suffix, adjectival suffix, adverbial suffix
noun	*en*	noun, adjective
noun	*n*	noun, adjective
noun, adjective, numeral, verb	*o*	noun, adjective, verb
noun	*s*	noun, adjective, nominal suffix, adjectival suffix, adverbial suffix

The few compounds, in which *e* follows an adjunct pronoun (*all-e-daag-s = dai-ly*) do not influence the analytic program, because adjunct pronouns have the same 2-nd index as adjectives.

Besides in the constructions permitted by this table *s* occurs:
behind an adjective: *bloot-s-hoofd-s* (= *bare-head-ed*);

behind an adjunct pronoun: *geen-s-zin-s* (= *by no means; geen = no, zin = sense*);
behind a verb: *betaal-s-heer* (= *pay-master*);
behind a preposition: *achter-s-kind* (= *grand-nephew, grand-niece; achter = behind, kind = child*), *voor-s-hand-s* (= *provisorily; voor = before, hand = hand*), but they do not represent productive compounding types.

4.1.3. COMPOUNDS OF TWO CONSTITUENTS

I call the first constituent A, the second one B. These compounds are first classified according to the kinds of constituent of A and B. The virtual words are once again comprised among the words (compare 3.1.).

The compounds of links have been treated in 4.1.2. The only case is: A is a word, B is a link.

A cannot be an inflectional morpheme, because an inflectional morpheme forms a higher constituent only with an immediately preceding constituent and with the list of inflectional morphemes given in 3.5. no Dutch word has two inflectional morphemes following each other.

So only seven cases remain for closer examination:

1) A is a word, B is a word
2) A is a word, B is an affix
3) A is a word, B is an inflectional morpheme
4) A is a word, B is a link
5) A is an affix, B is a word
6) A is an affix, B is an affix
7) A is an affix, B is an inflectional morpheme

Case 7 is eliminated, because though an inflectional morpheme can immediately follow an affix, the highest constituent, with which an inflectional morpheme can be connected to a next-higher constituent is always a word and never an affix.

Case 6 is eliminated. There are a few so-called "compound affixes", like *erig*. E.g. *branderig* (= *burnt*) can only be analysed as *brand-erig*. The analysis

brand *er* *ig*
burn

brander
(= *burner*)

branderig
(= *burnerish?*)

is false. But these compound affixes are adopted into the lexicon separately.

So only 5 cases remain and they really occur:

1)	word-word:	*huis-deur = house-door*
2)	word-affix:	*drink-baar = drink-able*
3)	word-inflectional morpheme:	*drink-t = drink-s*
4)	word-link:	*eik-e* (in *eik-e-blad = oak's leaf*)
5)	affix-word:	*on-geluk = mis-fortune*

A+B is called C. The 5 questions of 1.1. (except the 1-st, see 1.2.) are first treated for the most frequent (so-called normal) types of compounds. For them question 3 can be answered very generally: they are not pseudo-concatenations.

Now I shall give the instructions for the calculation of the indexes of C, if the set of indexes of A and the set of indexes of B have been determined. The choice among the perhaps more sets of indexes of one lexical morpheme will be treated in 4.2. and 4.3.

The indexes of A, B and C are called respectively:

A[1], ..., A[4],
B[1], ..., B[4],
C[1], ..., C[4].

All constituents of more than 1 morpheme are words or virtual words. So

C[1] = A[1]; C[2] = B[2]; C[3] = 2.

For the calculation of C[4] we have only to compute the leading constituent, because according to 1.2. C[4] always coincides with the 4-th index of the leading constituent. The most frequent cases are (the 5 cases

isolated above with a subdivision of case 5):

1) A is a word, B is a word, A is a determiner of B. B is the leading constituent. From a merely semantic point of view the COORDINATIVE compounds could be removed from this category, e.g. *geneesheer-directeur* (*physician-director*), but C has the part of speech of B in them too.
2) A is a word, B is an affix. B is the leading constituent.
3) A is a word, B is an inflectional morpheme. A is the leading constituent.
4) A is a word, B is a link. A is the leading constituent.
5) A is a dominant prefix, B is a word. A is the leading constituent.
6) A is a recessive prefix, B is a word. B is the leading constituent.

An immediate compound belonging to one of these six categories, in which every affix, inflectional morpheme or link follows or precedes a word it may follow or precede according to its indexes is called a NORMAL COMPOUND.

The indexes and the leading constituent of a normal compound can be computed on the base of these instructions.

A virtual word-formation (see the types 4 and 6 of abnormal compounds) is a word-formation F not occurring as such, but which could be used as an intermediate stage in the derivation of a word G from a word E in such a way, that both the derivation of F from E and that of G from F occur according to general derivation patterns in Dutch. The virtual word-formations should not be confused with the virtual words: they are not adopted into the lexicon.

Every compound which is not normal is called abnormal. There are a good deal of abnormal compounds. After subtraction of some entirely isolated word-formations and a far from complete scanning of [5] 9 types of abnormal compounds remained:

1) Pseudo-concatenations, i.e. compounds having at least the part of speech with the grammatical function which the phrase consisting of the constituents would have, if they were written as separate words and whose meaning is sometimes almost the same, as when the constituents would be written as separate words (*hand-vol = hand-ful, over-zee = over (the) sea*).
2) Adjectival compounds with a nominal last constituent and meaning:

"Having the thing called by the last constituent with its preceding determiners" (*acht-kant = eight-side-d, vol-bloed = pure-blood-ed,* lit. *full-blood*);

3) Nominal compounds with nominal last constituent and meaning: "Thing with the thing called by the last constituent with its preceding determiners" (*blauw-borstje = blue thrush, bleek-gezicht = pale-faced person*);

4) Derivatives of virtual direct derivatives (*beeld-end; beeld-end-e kunsten = plastic arts;* Dutch has a noun *beeld = image, statue,* but no verb *beeld*);

5) Compounds of verbal stems with agent-meaning, while the verbal stem itself does not have this meaning (*bedil-al = caviller, bedil = cavel, sta-in-de-weg* lit. *stand-in-the-way = stumbling-block*);

6) Direct derivatives of virtual compounds (*af-schuw = aversion, schuw* (verb) = *shun*);

7) Compounds, in which the second member determines the first one (*secretaris-generaal = secretary-general*);

8) Compounds indicating a thing with the property indicated by the compound, if the compound would be understood as a concatenation (*over-al = over-all*);

9) Compounds with abnormal insertion of affixes or links (*binnen-s-huis = in-door,* lit. *inside (the) house, voorzie-n-ing = provis-ion*);

The treatment of *sta-in-de-weg, binnen-s-huis* and *voorzie-n-ing* in 4.1.3. may seem strange from a compository point of view, because the words are considered in their relation to more than two constituents. Yet they are treated here, because *sta-in-de-weg* has a great structural similarity to *bedil-al* and the two other ones, if the program would take account of them, would have to be considered as mediate formations

binnen	*s*	*huis*		*voorzie*	*n*	*ing*
binnens		*huis*		*voorzien*		*ing*
binnenshuis				*voorziening*		

The types 4, 5, 6 and 8 are far too infrequent to be treated. All compounds belonging to these types are adopted into the lexicon separately. The analytic program does not take account of the possibility that new compounds of these types would be formed.

For type 3 only the relation in meaning between the compound and its constituents is abnormal, not the part of speech of the compound.

So only the types 1, 2, 7 and 9 remain for further examination. They are called respectively: *pseudo-concatenations*, *abnormal adjectival compounds*, *inverse compounds* and *compounds with abnormal insertions*.

I have only found one kind of compounds with abnormal insertions in [5], namely that with the scheme verb-*n*-postverbal suffix (*begaa-n-baar* = *pass-able*, *voorzie-n-ing* = *provis-ion*). These compounds are adopted into the lexicon separately.

One type of the abnormal adjectival compounds has the most representatives, namely the words *drie-kant* (= *tri-angul-ar*), *vier-kant* (lit. *four-side-d*, *quadrate*), *vijf-kant* (*penta-gon-al*), *zes-kant* (*sexa-gon-al*) and *acht-kant* (*octo-gon-al*).

As appears from the fact, that the analogous word-formations *zeven-kant* (*zeven* = *seven*), *negen-kant* (*negen* = *nine*) and *tien-kant* (*tien* = *ten*) are not used, this type is not productive. So the compounds mentioned are adopted into the lexicon separately and the analytic program does not take account of the possibility of the formation of analogous compounds.

Inverse compounds and pseudo-concatenations are always compounds of only words. So in the case, that at least one of A and B is not a word, the compound is always normal.

In inverse compounds the constituents are always separated by a hyphen ([14], p. LXIV). But inversely not all compounds, whose constituents are separated by a hyphen, are inverse.

The inverse compounds, in which neither constituent is a proper noun, e.g. *secretaris-generaal* (= *secretary-general*), *proces-verbaal* (= *official report*), are adopted into the lexicon themselves. This is possible, because such words are mostly imitations of French words and are not formed new. The few inverse compounds without an analogon in French, such as *kwartiermeester-generaal* (*quartermaster-general*), are also adopted in the lexicon.

The only productive type of inverse compounds is that, in which the first constituent is a noun and the second one a proper noun (recognisable by its capital). But [14] also gives a number of types of compounds with hyphens between the constituents, in which the second constituent has a

capital and which are not inverse compounds, namely (p. LXIV):

"2) Compounds with *Sint* (*St.* = *Saint*) as first member: *Sint-Nicolaas, St.-Nikolaas, Sint-Bernhard,* etc.

4) Some geographical names like *Nieuwpoort-Bad* ([14] also treats this under the head "inverse compounds", but because it is treated otherwise in a translation, if any, than the "true" inverse compounds (as a proper noun it is left unchanged), here we subtract it from the rubric "inverse compounds".

5) In names of married women: *Mevrouw* (= *Mrs.*) *A. Jansen-Smit.*

6) In geographical and other names, consisting of a proper noun with an uninflected word added before or behind. *Antwerpen-Oost, Nieuw-Zeeland, Oost-Indonesië* (= *East Indonesia*), *Voor-Indië* (= *Hindostan*), *Frans-Guyana* (= *French Guyana*), etc.

. . . . . .

7) The compound relies on a coordination still felt as such: *Belgisch-Nederlands* (= *Belgian-Dutch*), *Belgisch-Nederlands-Luxemburgs* (*Belgian-Dutch-Luxemburgian*), ..., station *Naarden-Bussem, Moeder-Maagd* (= *Mother-Virgin*), ...

8) In a bit unusual compounds, looking constructed, and other formations; further in general to explain the structure of the word or to prevent disguising or strange orthographical images: ..., *niet-Nederlander* (= *non-Dutchman*), ...; also in compounds whose first member consists of two parts written separately outside the compound, if one wants to stress, that these two parts belong together: ..., *Tweede-Kamerzitting* (= *House of Representatives Session*), ...; further in some words with *pro* and *anti*: *pro-Frans* (= *pro-French*), *anti-Duits* (= *anti-German*); ...

Remark 3: The orthography officially fixed of town names such as *'s-Gravenhage, 's-Gravenvoeren, 's-Herenelderen* can also be counted to this group".

The part of C before the first hyphen (this for compounds with more hyphens!) is called A, the part behind the first hyphen B. Then I give the rule:

C can only be an inverse compound, if A does not begin with a capital and B does, if A is a noun and B occurs in the lexicon either with indexes B[4] = 0 or not at all.

The condition, that A is not allowed to begin with a capital, excludes the cases 2), 4) and 5). The case 6) is excluded too, because every proper noun begins with a capital.

The town names *'s-Gravenhage, 's-Gravenvoeren* and *'s-Herenelderen* of remark 3 are excluded by the condition, that A must be a noun.

The coordinative compounds of case 7 are excluded, if A is not a noun. A noun can only be composed coordinatively with another noun. So our instruction would only wrongly classify a coordinative compound as inverse, if both A and B are nouns and A is written without a capital and B with one, e.g. *ingenieur-Nederlander* (= *engineer-Dutchman*). But in this case C[4] is also computed correctly. Only the leading constituent is determined wrongly.

Among the words of case 8 the words with *pro* and *anti* are eliminated by the condition, that A must be a noun. The "compounds whose first member consists of two parts written separately outside the compound, if one wants to stress, that these two parts belong together" are noted as A-B+D, in which A is the constituent before the hyphen. So this case would only wrongly classify a word as an inverse compound, if A were a noun and B would begin with a capital. That A and B form one phrase together, is only possible, if B is an apposition of A and so also a noun, more precisely a proper noun (because it is written with a capital), because determining words such as adjectives, articles, pronouns and numerals stand before the noun in Dutch. E.g. *koning-Davidster* (*King David's star*) is imaginable. But because such appositions are mostly written with a capital themselves (most people will write e.g. *Koning David* and not *koning David*), we need not take account of this either.

Only the use of the hyphen according to the rather vague rule "further in general to explain the structure of the word or to prevent disguising or strange orthographical images" can yield compounds which meet the formal requirements above and are not inverse compounds, e.g. *ras-Leemans* (*pure-blooded Leemans; ras = race*). But in that case C[4] is also determined unambiguously.

In an examination of 186 compounds (Nieuwe Rotterdamse Courant, 5 July 1963, 16 October 1963 and 17 October 1963; [7]) this procedure appeared to recognise 7 words correctly as inverse compounds, not to recognise 3 in-

verse compounds as such and to classify 3 compounds wrongly as inverse. This number of errors is too great to justify the complications in the analytic program by the incorporation of routines for the recognition of inverse compounds. So the inverse compounds are neglected in the analytic program.

Among the pseudo-concatenations of two constituents there is only one type with a great number of representatives, namely the type preposition + noun → adverb. There are moreover 9 pseudo-concatenations preposition + *een* → adverb. But for the small number of prepositions all these compounds can be adopted into the lexicon.

Even for *bij* (= *at*), which has very many pseudo-concatenations of the main form, in [5] I found 14 pseudo-concatenations of the form preposition + noun → adverb against 124 normal compounds with nouns. So I adopt all established pseudo-concatenations into the lexicon. The analytic program does not take into account the possibility of the formation of new pseudo-concatenations of two elements.

4.1.4. COMPOUNDS OF MORE THAN TWO IMMEDIATE CONSTITUENTS

I call the first constituent of the compounds with three immediate constituents A, the second one B and the third one C. A type of compound is noted as a number of three digits, in which the first digit is A[3], the second one B[3] and the third one C[3]. If A (respectively B, C) is an affix, the 1-st (respectively 2-nd, 3-rd) digit is always posed 3.

None of A, B and C can be a link on the basis of the convention of 4.1.2.

None of A, B and C can be an inflectional morpheme either.

A cannot, because all inflectional morphemes are suffixes.

B cannot be an inflectional morpheme either. The verbal inflectional morphemes never occur in the middle of the word and the morphemes *e*, *en* and *n* behind nouns and adjectives and before other constituents are always considered as links, affixes or words, never as inflectional morphemes (compare 3.8.).

C cannot be an inflectional morpheme either. There are of course plenty of inflectional forms of compounds, but they are so analysed, that either A and B, or B and C form a lower constituent together.

So the compound types possible a priori are: 333, 332, 323, 233, 223, 232, 322, 222.

The compound type 333 does not exist, since if two affixes cannot form a higher constituent together, of course three affixes can do so far less.

But a few representatives of the compound type 332 can be indicated, namely

on-ge-bloem-d (= *unveiled; gebloemd* is little usual)
on-ge-kurk-t (= *un-cork-ed; ge-kurk-t* is not mentioned)
on-ge-schoon-d (= *un-purifie-d;* a verb *schoon* does exist)
ver-on-zaad (= *exhaust (land) by always sowing the same seed;*
ver-on-zaad (adjective) = *corrupted by inchastity;* a verb *onzaad* does not exist)
ver-ont-heilig (= *profane* (verb); *ont-heilig* also exists)
ver-ont-reinig (= *dirty* (verb); *ont-reinig* is obsolete)
ver-ont-rust (= *disquiet; ont-rust* is literary)
ver-ont-schuldig (= *excuse; ont-schuldig* is obsolete)
ver-ont-waardig (= *make indignant; ontwaardig* does not exist)

The meaning of each of these compounds except *verontwaardig* can also be found by assigning them the following structural tree

A B C

A B+C

A+B+C

The connection in meaning between *verontwaardig* and *waardig* (= *worthy*) is so opaque, that it has anyhow to be adopted into the lexicon separately. So we do not take account of the possibility of the formation of new immediate compounds of the type 332 and give every compound of the type 332 a structural tree

A B C

A B+C

A+B+C

Compounds of the type 233 do not exist either. The only compounds which could be considered as such are the compounds with compound affixes,

e.g. *brand-er-ig* (4.1.3.). But in them B and C together are considered as one constituent and then they are compounds of two constituents.

The most numerous among the compounds of the type 323 are those, in which A is the prefix *ge*. They have to be adopted into the lexicon separately, since otherwise they would be analysed wrongly in far too many cases (3.5.). The other ones (*on-god-ist = a-the-ist, ont-zin-d = furious, ver-duivel-d = damned, ver-en-ig = unite, ver-eenzelv-ig = identify*) are so few and moreover partly so idiomatic in meaning, that they are adopted into the lexicon separately.

The same holds for the compounds of the type 322
(*ver-donkere-maan = annihilate,* lit. *make a dark moon,*
ver-halve-zool = half-sole, lit. *make a half sole*).

The compounds of the type 232 have only one productive sub-type. In order to understand this sub-type we must anticipate the results of the following.

A productive type of compound with 4 immediate constituents is that of the structure A+B+C+D, in which A is a noun or adjective, B is *be* or *ge*, C is a noun and D is *d* or *t*, e.g. *zwart-ge-laars-d = black-boot-ed.* If C ends with *d* or *t*, D is dropped and so a compound of the type A+*be*+C or A+*ge*+C is obtained. Then the entire word is an adjective. But these compounds are too complicated, so that the analytic program does not take account of them.

The number of types of compounds with three immediate constituents of the type 222 and 223 and compounds with more than three immediate constituents is very great, but the analytic program takes account of only one type of pseudo-concatenations, namely the compounds of the form A+*en*+C, in which A and C have the same part of speech. In that case C is appointed as the leading constituent (e.g. *drie-en-dertig = thirty-three*). The other types have few representatives and are not productive. So the compounds of these types are adopted into the lexicon and the analytic program does not take account of the formation of new compounds of these types. For the reason just explained I do the same for compounds with more than three immediate constituents.

4.2. THE RELATIVE PROBABILITIES OF ALTERNATIVE STRUCTURAL DESCRIPTIONS OF COMPOUNDS

4.2.1. THE SCOPE OF PROBABILISTIC CONSIDERATIONS

As we have seen in 1.1., the structural tree contains the following information:

1) The morphemes;
2) The intermediate constituents;
3) For each constituent:
3a) The set of indixes;
3b) (except for the morphemes), whether it is a pseudo-concatenation;
3c) (except for the morphemes) the leading sub-constituent.

So the analytic program has to accomplish the following tasks:

1) To dissect the words into morphemes;
2) To determine the sets of indexes of the morphemes;
3) To determine the intermediate constituents;
4) To compute the sets of indexes of the higher constituents from those of the lower ones;
5) To determine the leading sub-constituent of all constituents except the morphemes;
6) To determine, which constituents are pseudo-concatenations of their immediate sub-constituents.

The methods for 4, 5 and 6 have been described in 4.1.3. and 4.1.4.

As we have seen in 1.1., probabilistic considerations do not play a part in the dissection into morphemes. After a first dissection into morphemes (analytic program l. 881 - 934) the computer tries to construct a structural tree and only if this construction appears to be IMPOSSIBLE, it tries another dissection into morphemes.

So probabilistic considerations play a part only in the execution of the tasks 2 and 3.

The first application of probabilistic considerations is the ordering of the sets of indexes of one morpheme in the lexicon according to decreasing frequency (see 3.2.). This amounts to the calculation of the relative

probabilities of different sets of indexes of one lexical morpheme.

All other probabilistic considerations concern the relative probability of the CONNECTION of two constituents with given sets of indexes. The only type of compounds with three immediate constituents the analytic program takes account of is the pseudo-concatenation (4.1.4.) and this is recognised without probabilistic considerations. So the relative probability of combinations of three or more constituents is not incorporated into the analytic program.

Problems concerning inflectional morphemes and links are always treated without probabilistic considerations. So the only necessary probability array's are affin[0:6, 0:6] and affix[0:6, 3:19] . affin[a,b] is the probability of a constituent A+B, in which both A and B are words, a the 4-th index of A and b the 4-th index of B. affin[a,b] is called the AFFINITY of the part of speech belonging to the index a to that belonging to the index b. affix[a,b] is the probability of a compound of a word A with a 4-th index a with an affix B with 3-rd index b (B may be both a prefix preceding A and a suffix following A). If F is an affix with 3-rd index b, affix[a,b] is called the affinity of F to the part of speech with 4-th index a.

These "probabilities" do not have exactly the formal properties of the probabilities treated in statistics. Because in the analytic program only the relations <,> and = among them and their being 0 are used, it appeared to be convenient to give them integers as values.

A, B, C and the numerical indications of the types of tripartite compounds have the same meaning as in 4.1.4. If in the tripartite compound A+B+C A+B is an intermediate constituent, we speak of preconnection, if B+C is an intermediate constituent, of postconnection.

The a priori possible types of tripartite compounds of words and affixes are: 222, 223, 232, 322, 233, 323, 332, 333. The type 333 does not occur. For the determination of the intermediate constituent for the types 233 and 332 the array's affin and affix are superfluous, because all compounds of the type 233 have preconnection and all compounds of type 332 postconnection. Compounds of type 232 have preconnection if B is a suffix and postconnection, if B is a prefix.

So the only compound types, for which affin and affix can be used to determine the intermediate constituent,are 222, 223, 322 and 323.

Therefore section 4.2. has two sub-sections besides 4.2.1.:
4.2.2. The use of affin and the determination of its elements
4.2.3. The use of affix and the determination of its elements and the third indexes of the affixes.

4.2.2. THE USE OF AFFIN AND THE DETERMINATION OF ITS ELEMENTS

The array affin is used to determine the intermediate constituent in compounds of the types 222 and 223. I call the fourth indexes of A, B and C a, b and c. The subtype of the types 222 and 223, in which the 4-th indexes of A, B and C are respectively a, b and c is noted as a+b+c (for greater readability sometimes as K+L+M, in which K, L and M are the parts of speech indicated by a, b and c, e.g. adjective + numeral + numeral). Analogously a type of compound of two words A and B, in which A has the 4-th index a and B the 4-th index b, is noted as a+b.

The rule is, that if affin[a,b] $\geq$ affin[a,c], the word has preconnection and if affin[a,b] < affin[a,c], postconnection.

For the type 223 the array affix is irrelevant for the choice between preconnection and postconnection, because the 4-th index of the constituent with which C is connected (in that case A+B) is also b in the case of preconnection.

This rule entails, that if b=c, we always have preconnection. Indeed, all types of compound a+b+b have preconnection in the majority of the cases in the corpus scanned by me (1024 words from different sources) or at most as many cases of preconnection as of postconnection.

There are only two types with a majority of postconnections, namely adjective + numeral + numeral and adverb + adjective + adjective. I have only found one example of the type adjective + numeral + numeral in the corpus explored, namely *later-negentiende-eeuws* (= *in the later nineteenth centure*). The best solution seemed to me an especial rule, that a numeral is always connected with a numeral to the next-higher constituent, if it stands immediately besides it (the analytic program l. 372 - 373).

I found only one representative of the type adverb + adjective + adjective, occurring twice, namely *alleen-zalig-makend* (= *only making beate*). I thought I need not attribute value to such a small number.

For the determination of the affinities themselves I took as first criterion, that if the majority of the compounds of the type a+b+c have preconnection, then affin[a,b] $\geq$ affin[a,c] (b$\neq$c) and if the majority of the compounds of the type a+b+c have postconnection, then affin[a,b] < affin[a,c]. It has to be examined, whether this criterion does not produce contradictions.

The application of this criterion only determines the order among elements of affin sharing the first index. So in order to judge, whether this criterion produces contradictions, I have only to compare the statements about different elements of affin with the same first index. This criterion turns out not to produce contradictions. There are only two cases, in which both affin[a,b] $\geq$ affin[a,c] and affin[a,c] $\geq$ affin[a,b] hold. In these cases of course affin[a,b] = affin[a,c]. For this reason affin[4,0] = affin[4,3] and affin[5,0] = affin[5,3].

If there is not even one compound of the type a+b in the list of compounds of two constituents, I put affin[a,b] = 0. affin[0,0] and affin[4,4] get the value 5 for the great number of compounds of these types.

Further I assign values to affin[a,b] for each fixed a according to the following criteria:

1) If there is no c such that affin[a,c] < affin[a,b], then affin[a,b] = 1.
2) If affin[a,b] $\leq$ affin[a,c] and affin[a,b] $\geq$ affin[a,c], then affin[a,c] = affin[a,b].
3) If affin[a,b] $\leq$ affin[a,c] and on the other hand the tripartite compounds do not give reason to put affin[a,b] $\geq$ affin[a,c] and there is no d such that affin[a,b] $\leq$ affin[a,d] $\leq$ affin[a,c], then affin[a,c] = affin[a,b] + 1.

Here follows a table of the types of tripartite compounds, of which I found at least one representative in the corpus scanned, with their numbers of pre- and postconnections, the inequalities among the elements of affin and eventually the values of the elements of affin on their base. First I scanned 1024 words, which were pure word-compounds. For the types, for which I did not find pure word-compounds, I scanned the compounds, whose third constituents are suffixes, in the Nieuwe Rotterdamse Courant of 23

September 1967 (they were the types 0+3+1, 1+3+1, 4+3+1 and 5+3+1).

Subtype of tripartite compound	Number of preconnections	Number of postconnections	Inequalities among the elements of affin	Subtype of bipartite compound	Number of words	Elements of affin
0+0+0	223	83	No	0+0	889	affin[0,0]=5
0+0+1	1	0	affin[0,0]≥affin[0,1]	0+1	9	affin[0,1]=2
0+1+0	0	7	affin[0,0]>affin[0,1]			
0+1+1	1	1	No	0+2	0	affin[0,2]=0
0+3+0	3	23	affin[0,3]<affin[0,0]	0+3	50	affin[0,3]=1
0+3+1	0	33	affin[0,3]<affin[0,1]			
0+4+0	0	15	affin[0,4]<affin[0,0]	0+4	0	affin[0,4]=0
0+5+0	0	46	affin[0,5]<affin[0,0]	0+5	0	affin[0,5]=0
				0+6	0	affin[0,6]=0
1+0+0	48	19	No	1+0	71	affin[1,0]=2
1+0+1	1	1	No	1+1	33	affin[1,1]=2
				1+2	0	affin[1,2]=0
1+3+1	21	26	affin[1,1]>affin[1,3]	1+3	33	affin[1,3]=1
				1+4	5	affin[1,4]=1
1+5+0	0	1	affin[1,5]<affin[1,0]	1+5	0	affin[1,5]=0
1+6+1	8 compounds with 3 immediate constituents		No	1+6	0	affin[1,6]=0
2+0+0	17	0	No	2+0	8	affin[2,0]=1
				2+1	0	affin[2,1]=0
				2+2	0	affin[2,2]=0
				2+3	0	affin[2,3]=0
				2+4	0	affin[2,4]=0
				2+5	0	affin[2,5]=0
				2+6	0	affin[2,6]=0
3+0+0	55	3	No	3+0	87	affin[3,0]=2
3+0+1	4	0	affin[3,0]≥affin[3,1]	3+1	0	affin[3,1]=0
				3+2	0	affin[3,2]=0

Subtype of tri-partite compound	Number of pre-connec-tions	Number of post-connec-tions	Inequalities among the elements of affin	Subtype of bi-partite compound	Num-ber of words	Elements of affin
3+3+0	0	14	affin[3,3]<affin[3,0]	3+3	2	affin[3,3]=1
				3+4	0	affin[3,4]=0
				3+5	0	affin[3,5]=0
				3+6	0	affin[3,6]=0
4+0+0	16	0	No	4+0	52	affin[4,0]=2
4+0+1	4	1	affin[4,0]≥affin[4,1]	4+1	78	affin[4,1]=1
				4+2	0	affin[4,2]=0
4+0+3	1	0	affin[4,0]≥affin[4,3]	4+3	190	affin[4,3]=2
4+1+0	1	1	No			
4+1+1	0	2	No			
4+1+3	0	1	affin[4,1]<affin[4,3]			
4+3+0	3	0	affin[4,3]≥affin[4,0]			
4+3+1	45	5	affin[4,3]≥affin[4,1]			
4+4+0	1	0	affin[4,4]≥affin[4,0]	4+4	79	affin[4,4]=5
4+4+3	1	0	affin[4,4]≥affin[4,3]			
4+5+1	0	3	affin[4,5]<affin[4,1]	4+5	0	affin[4,5]=0
4+6+4	2 compounds with 3 immediate con-stituents		affin[4,6]<affin[4,4]	4+6	17	affin[4,6]=1
5+0+0	101	10	No	5+0	236	affin[5,0]=3
5+0+1	6	0	affin[5,0]≥affin[5,1]	5+1	84	affin[5,1]=2
				5+2	0	affin[5,2]=0
5+0+3	11	0	affin[5,0]≥affin[5,3]	5+3	492	affin[5,3]=3
5+1+0	0	4	affin[5,1]<affin[5,0]			
5+3+0	63	0	affin[5,3]≥affin[5,0]			
5+3+1	109	2	affin[5,3]≥affin[5,1]			
5+4+1	2	0	affin[5,4]≥affin[5,1]	5+4	12	affin[5,4]=4
5+4+3	5	0	affin[5,4]≥affin[5,3]			
5+5+0	11	2	affin[5,5]≥affin[5,0]	5+5	16	affin[5,5]=4
5+5+1	17	1	affin[5,5]≥affin[5,1]			

Subtype of tripartite compound	Number of preconnections	Number of postconnections	Inequalities among the elements of affin	Subtype of bipartite compound	Number of words	Elements of affin
5+5+3	16	0	affin[5,5]≥affin[5,3]			
				5+6	25	affin[5,6]=1
				6+0	0	affin[6,0]=0
				6+1	0	affin[6,1]=0
				6+2	0	affin[6,2]=0
				6+3	0	affin[6,3]=0
				6+4	0	affin[6,4]=0
				6+5	0	affin[6,5]=0
				6+6	5	affin[6,6]=1

130 of 1024 words are analysed incorrectly on the basis of these criteria, so 13%.

But if one assumes, that the lexicon contains all words of [5] and all proper nouns occurring in the text, only 691 words remain to be treated (the other ones cease to be compounds with more constituents, because their higher constituents stand in the lexicon), among which 37 are analysed incorrectly, so 5,3%.

4.2.3. THE USE OF AFFIX AND THE DETERMINATION OF ITS ELEMENTS AND THE 3-rd INDEXES OF THE AFFIXES

The determination of the third indexes of the affixes has to be adopted into the topic of this section, because affix[e,f] is defined as the probability of a compound of a word E with 4-th index e and an affix F with 3-rd index f, and f itself indicates the order of decreasing affinity of F to the different parts of speech.

affix is used:

1) To test the POSSIBILITY (not merely the probability) of the connection of E with F. The impossibility of the connection of E with F is indicated

by affix[e,f] = 0. In order to test the possibility of the connection of E with F the computer just tests, whether affix[e,f] = 0.

2) To determine the set of indexes of a word-morpheme connected with an affix, if the word-morpheme has more sets of indexes. So *werk-er* (= *work-er*) is analysed as a compound of the VERB *werk* and not of the NOUN *werk*, because affix[3,11] > affix[0,11] (the considered set of indexes of *er* is 20110; analytic program l. 443 - 444). So for this task elements of affix with the same SECOND index are compared. The inventory of all values of the 3-rd index of the affixes is made up on the basis of the compounds of ALL affixes.

3) To choose between preconnection and postconnection for the compound type 322 or 323, in which A is a recessive prefix (analytic program l. 470 - 471). The rule is: if affix[C[4],A[3]] > affix[B[4],A[3]], then postconnection, else preconnection.

4) To choose between preconnection and postconnection for the compound type 323, in which A is a dominant prefix. The rule is: if affix[B[4],C[3]] > affix[B[4],A[3]], then postconnection, else preconnection (analytic program l. 540 - 541). An exception has to be made for the case, that B[4] = 3, because the elements of affix cannot be assigned such values, that the correct analysis would then be given in the majority of the cases and no wrong analyses in other cases.

So the elements of affix occur in three kinds of operations except their value-assignments (l. 815 - 830):

1) Examination, whether an element of affix is 0;

2) Determination of the sequential order of two elements of affix with the same second index;

3) Determination of the sequential order of two elements of affix with the same first index.

To fit the elements of affix for the first task is simple: for this we have only to put affix[k,m] = 0 for all values of k and m, for which compounds of words with fourth index k and affixes with third index m are impossible.

Because the third index of F itself is defined by the order of decreasing probability of connection of F with words of different parts of

speech, only a few extra conventions would be necessary to assign the elements of affix values, by which they could accomplish their second task.

The execution of the third task consists of application of the rule: if affix[B[4],C[3]] > affix[B[4],A[3]], then postconnection, else preconnection.

The optimisation of the results of this rule requires statistic explorations of compounds of the type 323, in which A is a dominant prefix. Because C[3] and A[3] occur in this rule, these explorations require the previous determination of the 3-rd indexes of the affixes.

The determination of the first indexes of the affixes and the elements of affix is executed in four phases:

1) The determination of the 3-rd indexes of the recessive prefixes;
2) The ordering of the parts of speech according to decreasing affinity to F for each dominant affix F. According to the conventions of 3.2. this ordering immediately yields the values of the third indexes for the dominant affixes;
3) The determination of the sequential order of affix[b,a] and affix[b,c] for each triple (a,b,c), in which b is the fourth index of a word, a the third index of a dominant prefix and c the third index of a suffix;
4) The computation of the elements of affix on the basis of these facts.

1) For the determination of the third indexes of the recessive prefixes I first determined the number of compounds of each of them with words of different parts of speech (on the basis of an unpublished alphabetic frequency lexicon composed on the base of the corpus of [6]) and determined the sequential order of their affinities to the parts of speech by giving them the greatest affinity to the parts of speech with which most compounds occurred. If possible I tested the results by examining in tripartite compounds, with which parts of speech they are connected. The recessive prefixes connectible to more parts of speech are (their 3-rd index is provisorily set to x):

a	2×7	*ex*	2×7	*mis*	2×7	*oor*	2×7	*wan*	2×7
aarts	2×7	*her*	2×7	*oer*	2×7	*opper*	2×7		
aller	2×7	*in*	2×7	*on*	2×7	*re*	2×7		

Under the headings "noun - adjective", "noun - verb" and "adjective - verb" the number pairs x-y mean: in the cases, in which the prefix is followed by a noun and adjective (nominal and adjectival suffixes are bracketed with them), the prefix is connected with the noun (or a constituent with the noun as second member) in x cases and with the adjective (or a constituent with the adjective as second member) in y cases (and analogously for the two other headings).

Prefix	Translation	Number of compounds with Nouns	Adjectives	Verbs	Noun - adjective	Noun - verb	Adjective - verb	Order of affinity	3-rd index
a	*a*		1					Adjective-noun	10
aarts	*arche*	3						Noun-adjective	9
aller	absolute superlative		13					Adjective-adverb	8
ex	*ex*	0	0						9
her	*re*	1		61		11-52		Verb-noun	11
in	*in*		2		0- 1			Adjective-noun	10
mis	*mis*	7	0	7	1- 0	1- 3	0-3	Verb-noun-adjective	16
oer	*original*	1						Noun-adjective	9
on	*un*	14	125		2-46			Adjective-noun	10
oor	*original*	16			4- 0			Noun-adjective	9
opper	*supreme*	3			1- 0			Noun-adjective	9
re	*re*	5		13		5-10		Verb-noun	11
wan	*bad*	5	0	1		5- 1		Noun-verb-adjective	14

2) I ordered the parts of speech according to decreasing affinity to given dominant affixes by counting the compounds of words of given part of speech and the cases of preference of an affix F for a part of speech a above the part of speech b. If a word E has at least two sets of indexes, one assigning it the part of speech a and one assigning it the part of speech b, and the affix F is connected with E with the part of speech b in a certain compound, it is said, that F prefers b above a in that compound.

For the determination of the part of speech of E itself the criterion of 3.7., that the compound is considered as a compound of E in that meaning, by which its meaning can be paraphrased in the simplest way, is used. For equiradical words, for which the decision may be most difficult, this criterion is specified by the usage of so-called periphrastic schemes. Each affix F connectible to words E of part of speech e has one or more so-called periphrastic schemes, according to which the meaning of the compound can be paraphrased with the aid of the meaning of E. Such a periphrastic scheme must be based on unambiguous cases, e.g.
ver-A = *become* A (*ver-geel = become yellow, ver-groen = become green*).
If then there is a periphrastic scheme assigning the compound the correct meaning, if E has the part of speech a, but no such scheme, if E has the part of speech b, E is assigned the part of speech a.

E.g. the prefix *ver* has for the case, that A is a verb, i.a. the periphrastic scheme *ver*-A = *change by* A-*ing*, e.g. *ver-bouw* (*bouw = build*), *ver-buig* (*buig = bend*).

For the case, that E is a noun, *ver* has four periphrastic schemes:

1) A *ver*-B-*t* = A *becomes* B: *ver-ambtenaar* (*ambtenaar = officer*), *ver-kool* (*kool = coal*);
2) A *ver*-B-*t* = A *gets into* B: *ver-armoed* (*armoed = poverty*);
3) A *ver*-B-*t* C = A *makes* C B: *ver-afgood* (*afgood = idol*), *ver-alsem* (*alsem = wormwood*), *ver-slaaf* (*slaaf = slave*);
4) A *ver*-B-*t* C = A *provides* C *with* B: *ver-koper* (*koper = copper*), *ver-zilver* (*zilver = silver*).

To the case *ver-werk* (*werk = work*, both as a noun and as a verb) the periphrastic scheme *ver*-A = *change by* A-*ing* applied, but none of the four periphrastic schemes with A as a noun. So *ver-werk* is a compound of *werk* as a verb.

For this purpose I scanned the Nieuwe Rotterdamse Courant of 24 November 1967. Only for the affixes *be* 2×3, *ig* 20×1, *lijk* 20×1 and *ver* 2×3 did I stop counting on p. 4, because I had then already counted enough compounds to be able to draw conclusions.

For two parts of speech a and b I assigned an affix F a greater affinity to a than to b, if there were more compounds, in which F prefered a

above b than inversely. If that criterion failed, I gave F the greatest affinity to the part of speech, with which it has most compounds. These criteria turned out not to produce contradictions.

3) In the weekly supplement of the Nieuwe Rotterdamse Courant of 27 April 1968 I counted the compounds of words of given parts of speech with given dominant prefixes, the compounds of words of given parts of speech with suffixes of given 3-rd indexes and the tripartite compounds of type a-b-c, in which A is a dominant prefix with 3-rd index a (if A is a preverbal prefix, a is put equal to A), B a word with 4-th index b and C a suffix with 3-rd index c. For the tripartite compounds of each type I counted moreover the cases of preconnection and postconnection.

If a type of tripartite compounds a-b-c has more cases of preconnection, I put affix[b,a] > affix[b,c], else affix[b,a] < affix[b,c]. This is in theory the most reliable method. Another method is that, in which each part of speech gets the greatest affinity to the affixes, with which it has the most compounds. But this method yields gross errors. E.g. the corpus contains 3192 compounds of verbs with suffixes with 3-rd index 6 against 478 compounds of verbs with the prefix *be* 2163, while all compounds of the type *be*-3-6 have preconnection. So this criterion is only used to determine the order among elements of affix with the same 3-rd index, if that order cannot be determined with tripartite compounds.

If the choice between preconnection and postconnection in the type 323 has to be made only with affix, the inequalities given by the tripartite compounds (the 3-rd indexes of *be*, *ge* and *ont* are provisorily called *be*, *ge* and *ont*; the 3-rd index of *ver* is immediately called 18, because there is no other affix with 3-rd index 18):

affix[0, 14] > affix[0, *be*]
affix[0, 11] < affix[0, 18]
affix[0, 14] > affix[0, 18]
affix[0, 16] > affix[0, 18]
affix[0, 19] > affix[0, 18]
affix[1, 4] > affix[1, 18]
affix[1, 14] > affix[1, 18]
affix[2, 5] > affix[2, 18]
affix[2, 14] > affix[2, 18]
affix[3, 6] < affix[3, *be*]
affix[3, 11] < affix[3, *be*]
affix[3, 16] < affix[3, *be*]
affix[3, 19] < affix[3, *be*]

affix[3, 6] < affix[3, *ge*]
affix[3, 14] < affix[3, *ge*]
affix[3, 16] < affix[3, *ge*]
affix[3, 19] < affix[3, *ge*]
affix[3, 6] < affix[3,*ont*]
affix[3, 11] < affix[3,*ont*]
affix[3, 16] < affix[3,*ont*]
affix[3, 19] < affix[3,*ont*]
affix[3, 6] < affix[3, 18]
affix[3, 11] < affix[3, 18]
affix[3, 16] < affix[3, 18]
affix[3, 19] < affix[3, 18]

All these inequalities concern the order of the affinities of the preverbal prefixes and the suffixes. They leave the following gaps:

1) A flat contradiction: the inequalities affix[1,18] > affix[3,18], affix[3,18] > affix[3,14], affix[3,14] > affix[1,14] and affix[1,14] > affix[1,18] would hold at the same time;
2) The indeterminedness of the order between the affinities of the preverbal prefixes and the affinities of many suffixes;
3) The indeterminedness of the order of the affinities of the prefixoids and pronominal adverb-formators relative to the affinities of the suffixes.

1 and 2) For the triples (a,b,c), in which a is the 1-st index of a preverbal prefix and b ≠ 3, there are no strong independent reasons to fix the order between affix[b,a] and affix[b,c] (of course the integers affix[b,a] and affix[b,c] always have a sequential order, but this order has little influence on the result of the program). E.g. affix[0,9] and affix[0,10] would have to be smaller than affix[0,*be*] according to the numbers of the words of the compound types corresponding to them, but exactly for the types *be*-0-9 and *be*-0-10 preconnection is impossible, because suffixes with third index 9 or 10 cannot be connected with verbs. Therefore I do not require an especial order among these pairs of elements in the construction of the elements of affix, because a non-optimal analysis in very rare cases

(and the subtypes of tripartite compounds concerned are very rare - that is guaranteed by their absence in the corpus) is not a more serious defect than the assignment of an order to these elements on entirely insufficient grounds.

1) This contradiction is eliminated by dropping all inequalities of the form affix[3,c] > affix[3,d]. Then the instruction of preconnection in all tripartite compounds of the type 323, in which the second constituent is a verb and preconnection is not strictly impossible, is immediately given (analytic program 1. 541: ∨ (aff > 0 ∧ leider 2=3)) and not deduced from inequalities among elements of affix. So unique third indexes for *be*, *ge* and *ont* become superfluous.

3) As appears from their low frequency in the corpus (1 compound with *om* 2033, 2 with *over* 2033, none with the other ones) the prefixoids are not very productive. So I assign the lowest possible values to affix[0,3], affix[0,9] and affix[1,9].

Pronominal adverbs are very frequent and suffixes forming prepositions or adverbial by-forms of prepositions do not exist (the few derived prepositions, such as *niet-tegen-staande* (= *in spite of*, lit. *not-against-standing*) are adopted into the lexicon). So I assign higher values to affix[4,7] and affix[5,7] than to affix[4,c] and affix[5,c] for any value of c with $3 \leq c \leq 19$ and $c \neq 7$. This condition can be reduced to:

$c = 8$ or $12 \leq c \leq 19$ for adverbs

and

$13 \leq c \leq 19$ for prepositions,

because adverbs and prepositions can only be connected with affixes with these third indexes.

4) The set SI of inequalities which the elements of affix must fulfil is the following one:

1) (in virtue of the definitions of 3.2.; cet = 2, 4 or 5)

affix[0, 3] > 0	affix[5, 7] > affix[4, 7] > 0
affix[1, 4] > 0	affix[1, 8] > affix[4, 8] > 0
affix[2, 5] > 0	affix[0, 9] > affix[1, 9] > 0
affix[3, 6] > 0	affix[1,10] > affix[0,10] > 0

affix[3,11] > affix[0,11] > 0
affix[0,12] > affix[1,12] > affix[4,12] > 0
affix[0,13] > affix[1,13] > affix[3,13] > affix[cet,13] > 0
affix[0,14] > affix[3,14] > affix[1,14] > affix[cet,14] > 0
affix[1,15] > affix[0,15] > affix[3,15] > affix[cet,15] > 0
affix[3,16] > affix[0,16] > affix[1,16] > affix[cet,16] > 0
affix[3,17] > affix[1,17] > affix[0,17] > affix[cet,17] > 0
affix[1,18] > affix[3,18] > affix[0,18] > affix[cet,18] > 0
affix[3,19] = affix[0,19] > affix[1,19] > affix[cet,19] > 0
affix[m, n] = 0 for all combinations of m and n not mentioned in this list.

2) The inequalities given in this section 4.2.3.:

affix[0,18] > affix[0,11] affix[1,14] > affix[1,18]
affix[0,14] > affix[0,18] affix[1, 4] > affix[1,18]
affix[0,14] > affix[0,16] affix[2, 5] > affix[2,18]
affix[0,16] > affix[0,18] affix[2,14] > affix[2,18]
affix[0,19] > affix[0,18] affix[4, 7] > affix[4, 8]
affix[4, 7] > affix[4, c] for $12 \leq c \leq 19$
affix[5, 7] > affix[5, c] for $13 \leq c \leq 19$

I determine the elements of affix with value n by complete induction to n as follows.

1) It has been determined, which elements of affix are 0.

2) I suppose all elements of affix < n determined. Then every affix[a,b] which has not yet been assigned a value and for which in every inequality affix[a,b] > K in SI K has been assigned a value < n, gets the value n.

This gives the following values for the elements of affix.

2-nd index	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
1-st index																	
0	1	0	0	0	0	0	2	1	1	3	4	7	3	3	2	2	3
1	0	5	0	0	0	2	1	2	0	2	3	5	4	2	3	4	2
2	0	0	2	0	0	0	0	0	0	0	1	2	1	1	1	1	1
3	0	0	0	1	0	0	0	0	2	0	2	6	2	4	4	3	3
4	0	0	0	0	2	1	0	0	0	1	1	1	1	1	1	1	1
5	0	0	0	0	3	0	0	0	0	0	1	1	1	1	1	1	1
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

This table must be read as follows: for each m and n the number in the row with pre-index m and the column with superindex n is affix[m,n].

4.3. THE ARRANGEMENT OF THE ANALYTIC PROGRAM

The analytic program consists of three parts: the first reading of the lexicon (l. 1 - 50), the second reading of the lexicon (l. 51 - 169) and the reading and elaboration of the word-list (l. 170 - 1330).

The information of the lexicon is stored in four integer array's: DIC[1:dic], wodic[1:dicn+1], PAKET[1:paket] and SYNTAX[1:dicn+1] (l. 52 - 53).

DIC contains all lexical morphemes of the lexicon, wodic[n] is the ordinal number in DIC of the element of DIC, in which the first letters of the n-th word are stored. The elements of PAKET except the last one, if the lexicon contains an odd number of sets of indexes, each contain two sets of indexes following each other. SYNTAX[n] is the ordinal number in the series of sets of indexes in the lexicon of the 1-st set of indexes of the n-th complete morpheme.

dicn is the number of complete morphemes in the lexicon. 4 letters following each other of one lexical morpheme (or less at the end of a word, whose number of letters is not a multiple of 4) are stored together in one element of DIC. dicn is the number of elements DIC must have in virtue of this. paket is the number of elements of PAKET.

From this it follows, that the lexicon must be read twice: once to determine dic, dicn and paket (l. 1 - 50) and once to determine the values of the elements of DIC, wodic, PAKET and SYNTAX (l. 51 - 169).

The integer array's alv[120:127] and revers[66:70] (l. 20) are used respectively for the compression of the letters in the array DIC and the reduction of the modified encodation into the original one.

The array alv maps the integers representing ', - and | according to the X8-code to the numbers 70, 67 and 66 (so all signs occurring in words are mapped to numbers < 90, so that more letters can be stored in one machine word), revers reduces these number representations to the original representations according to the X8-code.

In the analytic program in the second reading of the lexicon two sets

of indexes are stored together in one element of PAKET. This necessitates the statement-part
REMAINDER(p,1000)+200×ind2+800×ind1 (l. 156)
in order to "compress" the sets of indexes and a procedure VALIND(k) (l. 119 - 138) to reobtain the indexes from an element of PAKET for the analysis of the word to be elaborated (see 4.1.3.). The parameter k is here the ordinal number of the morpheme being treated in the word to be analysed. The procedure VALIND uses the auxiliary procedure pakt in its turn, which makes available the correct set of indexes from two sets stored in one element of PAKET.

All words of the word-list are read and treated successively and independently of each other. The information about each word is stored in the integer array LETTER[1:45] (l. 24). The representation of the k-th letter is LETTER[k].

The word being analysed is called W. The number of morphemes of W is called woord. The number of letters of W is called letter.

The analysis of W is executed in four stages: the dissection into morphemes (l. 881 - 934), the determination of the so-called dominator of each morpheme and the determination of the pseudo-concatenations (l. 935 - 1094), the determination of the intermediate constituents (l. 1095 - 1202) and the printing or punching of the results (l. 1203 - 1300). The program has two versions: one in which the output is printed and one in which it is punched. The second one (with added line numbers) is printed in 6.1.

The ordinal number of the first letter of the k-th morpheme as a letter of W is called unua[k]; the ordinal number of the last letter of the k-th morpheme lasta[k] (l. 23).

The dissection into morphemes uses two procedures: Meq(m,n) (l. 200 - 243) and lv(m,n) (l. 246 - 293).

The ordinal number of the equivalent of the k-th morpheme of W in the lexicon is called WOORD[k] (l. 20; if the k-th morpheme has no equivalent in the lexicon, WOORD[k] is put 0). The procedure for the determination of the equivalent of the string from the m-th to the n-th letter is called Meq(m,n) (Meq does not need a parameter k, because the index of the element of WOORD being determined is always woord).

lv(m,n) is the ordinal number of the last letter of the longest ini-

tial segment of the string from the m-th to the n-th letter with an equivalent in the lexicon. If that string of letters does not have such an initial segment, lv(m,n) is put to 0.

The Reifler calculus and the deviation from that in the analytic program has largely been treated in 1.2. Only two deviations of this fragment from the Reifler calculus could not be explained there: the statements depending on conditional clauses containing ind1 and ind2 and the statements positie. ind1 and ind2 are parameters of each lexical morpheme provisorily found in W (the program is such, that ind1's and ind2's of different lexical morphemes need never to be considered at the same time; ind1 and ind2 themselves do not occur in the sets of indexes of the lexical morphemes, but can be calculated from them by the procedure positie, l. 296 - 308). The meaning of ind1 and ind2 is as follows (they occur in quite another sense on l. 154 - 157):

Occurrence of the morpheme as first morpheme	ind1
Can occur as the first morpheme of a word, but also in other positions	0
Can only occur as the first morpheme of a word	1
Cannot occur as the first morpheme of a word	2
Occurrence of the morpheme as last morpheme	ind2
Can occur as the last morpheme of a word, but also in other positions	0
Can only occur as the last morpheme of a word	1
Cannot occur as the last morpheme of a word, but may be followed both by vowels and consonants	2
Can only occur followed by a vowel	3

A morpheme is rejected as first morpheme, if its ind1=2; as last morpheme, if its ind2 > 1; in all other positions, if its ind1=1 or ind2=1.

At first view the definitions of ind1 and ind2 are very similar to those of INDEX[k,1] and INDEX[k,2] in 3.2. The difference between INDEX[k,1] and INDEX[k,2] on one hand, ind1 and ind2 on the other hand is,

that INDEX[k,1] and INDEX[k,2] are properties of SYNTACTIC morphemes, but ind1 and ind2 properties of COMPLETE morphemes. E.g. that INDEX[k,2] = 2, means, that the lexical morpheme A cannot occur as the last morpheme of W WITH THE SET OF INDEXES BEING CONSIDERED AT THAT MOMENT; that ind2 = 2 means, that A cannot occur as the last morpheme of W WITH ANY OF ITS (PERHAPS MORE) SETS OF INDEXES. E.g. for *in* 2107 INDEX[k,2] = 2; but for *in* ind2 = 0, because the lexical morpheme *in* can occur as the last morpheme of W with another set of indexes (e.g. *in* 20030).

Because the word-list is read only once, unua, lasta, WOORD, MORFEEM, LETTER and INDEX must have fixed upper bounds. They are chosen such, that for the overwhelming majority of the words occurring in practice the indexes of their elements fall inside the array-bounds.

In the determination of the dominator the array subj[1:12] (l. 23) is used. subj[k] = m means: the dominator of the k-th morpheme is the m-th morpheme. subj[k] = 0 means: the k-th morpheme has no dominator.

The determination of subj uses six auxiliary procedures: structur(k,m) (l. 356 - 416), compl(n,k,m) (l. 419 - 703), maxaffin(p,r) (l. 706 - 732), macrostructur(k,m) (l. 735 - 806), close (l. 311 - 317) and transform (change) (l. 331 - 353).

Formally the dominators fulfil the following conditions:

1) The word has exactly one morpheme without a dominator; each other morpheme has exactly one dominator.
2) There is no series of morphemes $a_1, \ldots, a_n$ such that for $2 \leq k \leq n$, a_k is always the dominator of a_{k-1} and $a_1 = a_n$.
3) Every set of morphemes which always contains b, if it contains a morpheme a and a is the dominator of b, is a continuous series of morphemes.

Starting from the complete structural tree one finds the LEADING MORPHEME of a constituent $C = C_0$ as follows:

We construct the series of constituents $C_0, \ldots, C_n$ by complete induction as follows:

If C_n is a morpheme, it is the leading morpheme of C and the construction is completed; if C_n is not a morpheme, the leading immediate sub-constituent of C_n is C_{n+1} (this construction serves to explain the linguistic meaning of the term LEADING MORPHEME; since it presupposes the construction

of the structural tree, it is nowhere executed in the analytic program).

If A is the largest constituent whose leading morpheme is a, B is the next-higher constituent of A and b the leading morpheme of B, then b is called the DOMINATOR of a.

structur(k,m) determines the structure of the series of the k-th up to the m-th morpheme, if they are all words (according to 3.2. this means: if INDEX[p,3] = 2 for $k \leq p \leq m$). The m-th morpheme is the leading morpheme of this series. So for the other morphemes the dominator is determined and whether the next-higher constituent, whose leading morpheme it is, is a normal compound or a pseudo-concatenation.

According to 4.1.3. and 4.1.4. the program takes only account of two possible relations between mere word-compounds and their immediate constituents:

1) Normal compounds of the form A+B, in which B is the leading constituent;
2) Pseudo-concatenations of the form A+*en*+C, in which C is the leading constituent (at least, in the analytic program it is considered as such; from a mere linguistic point of view A and C are equivalent members of a coordination).

en is recognised by its INDEX[k,3] = 6 (except in one place of the procedure compl (the local procedure mistest, l. 426 - 430) and the test, whether a morpheme with a 2-nd index 3 is followed by a vowel, the structure is only determined with the aid of the indexes of the morphemes, not their letters; only in the punching of the results (l. 1203 - 1330) are the letters used again).

In compounds without *en* the dominator of each morpheme is determined with the aid of the affinities of INDEX[p,4] (the values of the elements of affin are assigned on l. 831 - 839). If W does not contain *en*'s, in the execution of structur(k,m) initially the k+1-th morpheme is indicated as the dominator of the k-th morpheme and then all morphemes are scanned and each time, when
affin[INDEX[k,4],INDEX[r,4]] > affin[INDEX[k,4],INDEX[s,4]]
(s is the ordinal number of the morpheme last indicated as the dominator of the k-th morpheme), then the r-th morpheme is indicated as the dominator of

the k-th morpheme. Then the following morphemes are treated in the same way as the k-th one.

If the r-th morpheme is *en*, all morphemes from the r+1-th one up to the m-th one are scanned and the first one with the same part of speech as the r-1-th morpheme, the s-th morpheme of W, is indicated as the dominator of both the r-1-st and the r-th morpheme. Then the compound of the constituents with as leading morphemes, respectively, the r-1-th, the r-th and the s-th morpheme is a pseudo-concatenation. If such a morpheme is not found, the analysis is continued for other values of the sets of indexes or if all sets of indexes of all morphemes have been tried, another dissection into morphemes is tried and if that turns out to be impossible, the word is declared unanalysable.

normaal[k] (normaal[1:12] (l. 25) is a boolean array) means: the next-higher constituent of the constituent whose leading morpheme is the k-th morpheme, is a normal compound. Normal compounds are far more frequent than pseudo-concatenations. So for each combination of sets of indexes all elements normaal[1], ..., normaal[woord] are put true (l. 959) and in other places in the procedures structur, compl and macrostructur an element of normaal is put false, if the constituent with the morpheme concerned as leading morpheme appears to be connected with its co-constituents to the next-higher constituent in a pseudo-concatenation, but nowhere explicitly true.

compl(n,k,m) analyses the structure of the k-th up to the m-th morpheme. n is the ordinal number of the BLOCK to which the k-th up to the m-th morpheme belong (the meaning of the blocks will be explained on p. 78), compl(n,k,m) also determines lblok[n], the leading morpheme of the series.

compl is only applied to morpheme sequences consisting of at most three sub-sequences following each other: an initial segment, consisting only of prefixes, a central segment, consisting only of words and a final segment, consisting only of suffixes and links. The initial segment and the final segment may also miss.

The procedure compl has the local switch IkIm (l. 424) and four local procedures: P27op (l. 464 - 477), terugschuif (l. 480 - 487), mistest (l. 426 - 430) and optimum (l. 433 - 450).

The morpheme sequence to which compl is applied can be in four cases:

1) It has an initial segment and a final segment;
2) It has an initial segment, but no final segment;
3) It has a final segment, but no initial segment;
4) It has neither an initial segment, nor a final segment.

The elements of IkIm are successively followed by the instructions for these four cases.

The ordinal number of the last morpheme of the initial segment is called lpref, the ordinal number of the first morpheme of the final segment is called bsuff.

In the case Ik2Im2 two subcases are distinguished. In the first subcase

INDEX[bsuff,3] > 2^affix[INDEX[bsuff-1,4], INDEX[lpref,3]] ≠ 0.

In that case first the structure of the central segment is determined by the procedure structur (so every element of the central segment, except the last one, is assigned a morpheme of the central segment as dominator).

Then the structure of the series is further constructed concentrically with the aid of the labels open, openeind, openbegin and gesloten. At each moment I call the maximal sequence of morphemes in the sequence, all of which, except the leading morpheme and recessive prefixes, have been assigned a dominator, the EXPLORED FIELD. I call the ordinal number of the first morpheme of the explored field begin, that of the last morpheme of the explored field eind and that of the leading morpheme of the explored field leider. The explored field is continuously extended so, that either the longest preceding continuous sequence of recessive prefixes is added (by the procedure terugschuif), or the last morpheme before the explored field or the first morpheme behind it is assigned as the dominator of the leading morpheme, or the leading morpheme is assigned as the dominator of the first morpheme behind the explored field. At each moment of the elaboration four cases are possible:

1) k = begin, eind = m ("gesloten");
2) k = begin, eind < m ("openeind");
3) k < begin, eind = m ("openbegin");
4) k < begin, eind < m ("open").

The four labels mentioned are followed by the instructions for these four cases.

The execution of the program fragments Ik2Im2 and Ik2Im3 is completed by the execution of the procedures optimum and P27op; the execution of Ik3Im2 by the execution of optimum.

The procedure optimum changes the set of indexes of the last morpheme of the central segment, if in this way that gets a greater affinity to its dominator. The procedure P27op assigns dominators to the recessive prefixes.

A more detailed description of procedure compl would only be a useless paraphrase of the concerned program fragment.

A block is a continuous sequence of morphemes maximal with respect to the property to contain no more than one continuous sequence of prefixes at the beginning and one continuous sequence of suffixes, inflectional morphemes and links at the end. macrostructur assigns the dominators to the leading morphemes of the block (except the last one, which has no dominator). If the dominator of the leading morpheme of the p-th block is a morpheme of the r-th block and the r-th block begins with a prefix, then this dominator is ALWAYS the leading morpheme of the r-th block. If the dominator of the leading morpheme of the p-th block is a morpheme of the r-th block and the r-th block does not begin with a prefix, then this possible dominator is determined with the procedure maxaffin(p,r). The procedure macrostructur for the determination of the relation among the leaders of the blocks is very analogous to the procedure structur for the determination of the relation among a sequence of word-morphemes following each other.

By the ordinal number of a set of indexes I mean its ordinal number in the series of all sets of indexes in the lexicon. Then inf[p] (l. 22) is the ordinal number of the first set of indexes of the p-th morpheme of the word being analysed. sup[p] (l. 22) is the ordinal number of the last set of indexes of the p-th morpheme of the word being analysed. tract[p] is the ordinal number of the set of indexes assigned to the p-th morpheme at that moment. finit is the ordinal number of the last morpheme, for which tract[p] has not yet become equal to sup[p].

close modifies finit, if that has become necessary by modifications in tract. transform modifies the elements of tract, when the case, that tract

has a certain value, has been elaborated completely.

Four non-local switches are used: SIND, INDAF, SCHAK and Prenpo (l. 171 - 174). SIND is used on l. 968, SCHAK on l. 1003, INDAF on l. 970 - 974 and Prenpo on l. 1144 - 1168.

The elements of SIND are followed by the instructions for the cases, in which the morpheme being treated is, respectively, a link, an inflectional morpheme or a word.

The elements of SCHAK are followed by the instructions for the cases in which a link is followed, respectively, by a link, an inflectional morpheme or a word.

The elements of INDAF are followed by the instructions for the cases, in which the morpheme behind a certain prefix is, respectively, a link, an inflectional morpheme or a word.

The middle part of the program for the analysis of words consists itself of two parts: the division of the morpheme sequence into blocks (l. 935 - 1059) and the definitive determination of the elements of subj (l. 1060 - 1094).

The fragment of l. 935 - 1059 determines the elements of the integer array's eblok[0:12], thblok and lblok[1:12] (l. 22 - 23). The number of blocks in the word is called blokn. blokn is only determined in this program fragment. Therefore eblok and lblok have to be declared with a fixed upper bound. But values are only assigned to the elements of eblok and lblok with index $\leq$ blokn.

bblok(p) (l. 141) is the ordinal number of the first morpheme of the p-th block as a morpheme of W.

eblok[p] is the ordinal number of the last morpheme of the p-th block as a morpheme of W (eblok[0], which would be undefined according to this definition, is formally assigned the value 0 (l. 876); note, that this statement is executed before the analysis of the words). This serves to simplify some calculations.

lblok[p] is the ordinal number of the leading morpheme of the p-th block. Note, that the value assignment to lblok is provisory. It can be undone on l. 1063 - 1094.

blok[p] is the ordinal number of the block to which the p-th morpheme belongs.

thblok[p] is eblok[p], if the last morpheme of the p-th block is a word or an affix and eblok[p] - 1, if the last morpheme of the p-th block is an inflectional morpheme or a link.

The program l. 935 - 1094 determines the dominators of all morphemes, those of the morphemes of each block (except the leading morpheme) with the procedure compl (n is the ordinal number of the block), those of the leading morphemes of the blocks with the procedure macrostructur.

The program fragment l. 1095 - 1262 computes the constituents of W.

FIN gets the value 2, if the word is unanalysable, 3, if the word is analysable.

The program fragment l. 1095 - 1202 uses the procedure dom(A,B,m) (l. 92 - 99) and eind(n,k) (l. 76 - 77) and computes the elements of the array's lengte, begin, leider, diepte and naasthoger and the integer hoogte. As for the array's LETTER, WOORD, etc., values are often not assigned to all their elements. lengte[n] is the number of constituents of the n-th order. begin[m,n] is the ordinal number of the 1-st morpheme of the n-th constituent of the m-th order (formally begin[hoogte, lengte[m]+1] is put to woord + 1 for the determination of eind(m,lengte[m]). leider[m,n] is the ordinal number of the leading morpheme of the n-th constituent of the m-th order. naasthoger[n,m] is the ordinal number of the 1-st morpheme of the constituent of the n-th order, which is the leading constituent of the m-th constituent of the n+1-th order. Formally naasthoger[n, lengte[n+1]+1] is put to woord + 1. hoogte is the maximum of the orders of the morphemes as constituents. diepte[k] is the order as constituent of the greatest constituent, whose leading morpheme is the k-th morpheme.

In the determination of diepte and leider the switch Prenpo is used. The leading morphemes of the immediate subconstituents of a constituent with the OMK-th morpheme as leading morpheme always have the OMK-th morpheme as dominator and stand in the range between the leading morphemes of the two adjacent constituents of the same constituent as the constituent dominated by the OMK-th morpheme. The candidates for the function of leading morpheme of an immediate subconstituent are predom, the 1-st morpheme in the range before OMK immediately dominated by OMK, postdom , the last morpheme in the range behind OMK immediately dominated by OMK and endom,

the conjunction immediately dominated by OMK (each of these three is put 0, if the morpheme concerned is absent). The elements of Prenpo have the form Pkmn, in which k, m and n are each 0 if and only if predom, endom and postdom are, respectively, 0.

eind(m,n) is the ordinal number of the last morpheme of the n-th constituent of the m-th order.

In the punching of the results the procedures PUCAS(m) (l. 66 - 73) and notind(n,k) (l. 80 - 89) are used.

PUCAS serves to punch the elements of LETTER in the form of the letters, from which they have originally been obtained. Its parameter m is the ordinal number of the morpheme being treated (each time an entire morpheme is punched at the same time).

notind(n,k) computes (and punches!) the set of indexes of the k-th constituent of the n-th order.

dom(A,B,m) (l. 92 - 99) means: diepte[B] $\leq$ m and there is a chain $C_0, \ldots, C_n$ such that $A = C_0$, $B = C_n$ and for every k with $1 \leq k \leq n$ C_{k+1} = subj[C_k].

The program fragment from l. 1203 - 1330 becomes very complicated by the necessity to put the first letter of each constituent perpendicularly under the first letter of its leading morpheme. This makes an integer array kolon (l. 23) necessary. kolon[k] - 1 is the distance in tabulations between the first letter of the k-th morpheme and the beginning of the line. mod8 is the distance in carriage places between the letter just printed and the last tabulation. In this program fragment the sets of indexes of the constituents are also computed and punched (by the execution of notind).

5. RESULTS: ERROR ANALYSIS

I executed this program with as word-list the first 197 compounds with more than 2 morphemes in "De Groene Amsterdammer" of 12 November 1966.

The lexicon contained all lexical morphemes which should occur in a complete lexicon of Dutch according to the instructions of 3. and which, considered as mere strings of letters, occurred in at least one of the words in the word-list. Each was followed by all the sets of indexes which should follow it in the complete lexicon. So the lexicon had e.g. to contain the word *mees* (= *tit*) because of *stadsbouwmeester* (= *urban architect*) in the word list, though it is not a morpheme of *stadsbouwmeester*. This composition of the lexicon was necessary to control, in how far the analytic program itself could find the correct morpheme analysis instead of being forced to it by a restricted lexicon.

The lexicon, the word-list, the analytic program and the results are given in appendix 6.

Only 4 errors occurred, namely:

post/academi/aal (= *post-academi-al*), in which *post* was considered as a noun;

af/beeld/en (= *map* (verb)), in which *beeld* was considered as a noun;

winkel/bedrijv/ig/heid = *shop-busy-ness*, in which *winkelbedrijv* (= *shop enterprise*) was considered as a constituent;

tussen/verkiez/ing (= *by-election*), in which *tussenverkiez* was considered as a constituent.

The first two errors are a consequence of the fact, that the noun *post* is more frequent than the virtual preposition, respectively the noun *beeld* more frequent than the verb.

The erroneous analysis of *winkel/bedrijv/ig/heid* is caused by the general instruction of preconnection of tripartite compounds, whose last two constituents have the same part of speech, what is justified by statistical considerations (see 4.2.2.), that of *tussen/verkiez/ing* by the equal affinity of prepositions to verbs and to nouns.

So all errors could only be avoided by modifications in the analytic program, which would either be completely AD HOC, or cause far more errors in other cases. An error procentage of 2% seems reasonable to me.

6. APPENDICES

6.1. THE ANALYTIC PROGRAM

```
1   begin comment : Integrated program for the dissection into
2   morphemes and the construction of structural trees by Dr. W.A.
3   Verloren van Themaat, R1436. This program has been executed on the
4   computer X8 of the Mathematical Centre and combines the functions
5   of the programs R1079 and R1173. The words to be treated by this
6   program are allowed to contain capital and lower-case letters of
7   the Latin alphabet, also with special signs, hyphens, apostrophs
8   and full stops. The program contains two number-tapes. The first
9   one is the lexicon, the second one the word-list. The word-list
10  contains the words to be analysed. The lexicon is noted according
11  to the instructions of section 2. of the report. The output is a
12  structural tree of words according to the model described in
13  section 1.1. of the report;
14  integer k, m, n, p, q, r, s, dic, dicn, maxl, symbol, RETURN, mod4,
15  KOLON, paket, woord, blokn, hoogte, ind3, tractk, letter, FIN,
16  AP, EC, plaats, indn, indp, indr, afp, afp1, ind1, ind2, ebl,
17  lem, lem1, aff1, u, v, finit, FASE, P, Q, R, u1, OMK, predom,
18  postdom, endom, regel;
19  boolean lex, SCHOON, CHANGE, NORMAAL, NASCHAK;
20  integer array alv[120:127], revers[66:70], blok, WOORD[1:12],
21  INDEX[1:12,1:4], reg, lengte[1:7], begin[1:7,1:15],
22  leider[1:7,0:15], naasthoger[2:7,1:15], tract, inf, sup, thblok,
23  lblok, subj, kolon, unua, lasta[1:15], diepte, eblok[0:15],
24  LETTER[1:45], prod[0:8,1:4], affin[0:6,0:6], affix[0:6,3:19];
25  boolean array normaal[1:12], voorschak[0:6,0:3], naschak[0:3,0:6],
26  suffix[0:3,0:4];

28  integer procedure nexttape;
29  begin integer heptad;
30     heptad:= RESYM;
31     nexttape:= if heptad < 120 ∨ heptad > 127 then heptad else
32     alv[heptad]
33  end;

35  alv[126]:= 70; alv[127]:= 67; alv[120]:= 66; revers[70]:= 126;
36  revers[67]:= 127; revers[66]:= 120; dic:= dicn:= paket:= 0;
37  mod4:= 3; SCHOON:= true;
38 BEGIN: symbol:= nexttape;
39  goto if symbol = 72 then COUNT else BEGIN;
40 COUNT: symbol:= nexttape; if symbol ≠ 76 then
41  begin if symbol = 118 ∨ symbol = 119 ∨ symbol = 135 then
42     begin dicn:= dicn + 1; mod4:= 3 end
43     else if symbol = 93 then
44     begin symbol:= read; if SCHOON then paket:= paket + 1;
45   SCHOON:= ¬SCHOON
46     end
47     else if mod4 = 3 then
48     begin dic:= dic + 1; mod4:= 0 end
49     else mod4:= mod4 + 1; goto COUNT
50  end;
```

```
51   comment Leg de eerste band opnieuw in de bandlezer;
52   begin integer array DIC[1:dic], wodic, SYNTAX[1:dicn + 1],
53      PAKET[1:paket];

66      procedure PUCAS(m); integer m;
67      begin integer p, q, s;
68   s:= lasta[m];
69   for q:= unua[m] step 1 until s do
70   begin p:= LETTER[q];
71      if p < 66 ∨ p > 70 then PUSYM(p) else PUSYM(revers[p])
72   end
73      end;


76      integer procedure eind(n, k); integer n, k;
77      eind:= begin[n,k + 1] - 1;


80      procedure notind(n, k); integer n, k;
81      begin integer nk, IND1, IND2, IND3, IND4;
82   if begin[n,k] = eind(n, k) then
83   begin nk:= begin[n,k]; IND3:= INDEX[nk,3]; IND4:= INDEX[nk,4]
84   end
85   else
86   begin IND3:= 2; IND4:= INDEX[leider[n,k],4] end;
87   IND1:= INDEX[begin[n,k],1]; IND2:= INDEX[eind(n,k),2];
88   ABSFIXP(5, 0, 10000 × IND1 + 1000 × IND2 + 10 × IND3 + IND4)
89      end;


92      boolean procedure dom(A, B, m); value A, B, m; integer A, B, m;
93      begin integer C;
94   C:= A;
95      D: if diepte[C] < m then
96   begin dom:= if B = C then true else false end
97   else
98   begin C:= subj[C]; goto D end
99      end;


102     integer procedure len(n); integer n;
103     begin integer REST;
104  REST:= DIC[wodic[n + 1] - 1];
105  len:= (if REST < 8100 then (if REST < 90 then 1 else 2)
106  else if REST < 729000 then 3 else 4) + (wodic[n + 1] -
107  wodic[n] - 1) × 4
108     end;


111     integer procedure pakt;
112     begin integer PAKO;
```

```
113   PAKO:= PAKET[(tractk + 1) : 2];
114   pakt:= if EVEN(tractk) = -1 then PAKO : 4000 else
115   REMAINDER(PAKO, 4000)
116      end;


119      procedure VALIND(k); value k; integer k;
120      begin integer m, PAKT;
132   integer array PAK[1:3];
133   PAKT:= pakt; PAK[1]:= PAKT : 800; PAK[2]:= PAKT : 200;
134   PAK[3]:= PAKT : 10; INDEX[k,1]:= PAK[1];
135   INDEX[k,2]:= PAK[2] - 4 × PAK[1];
136   INDEX[k,3]:= PAK[3] - 20 × PAK[2];
137   INDEX[k,4]:= PAKT - 10 × PAK[3]
138      end;


141      integer procedure bblok(k); integer k; bblok:= eblok[k - 1] + 1;

143      k:= m:= n:= 0; mod4:= 3; wodic[dicn + 1]:= dic + 1;
144      SYNTAX[dicn + 1]:= 2 × paket + (if SCHOON then 1 else 0);
145      SCHOON:= true;
146   SKIP: symbol:= nexttape;
147      goto if symbol = 72 then ASSIGN else SKIP;
148   ASSIGN: symbol:= nexttape; if symbol = 76 then
149      begin if ¬SCHOON then PAKET[n]:= PAKET[n] × 4000 end
150      else
151      begin if symbol = 118 ∨ symbol = 119 ∨ symbol = 135 then
152   begin lex:= true; m:= m + 1; wodic[m]:= k + 1; mod4:= 3 end
153   else if symbol = 93 then
154   begin if SCHOON then n:= n + 1; p:= read; ind1:= p : 10000;
155      ind2:= p : 1000 - 10 × ind1;
156      PAKET[n]:= REMAINDER(p, 1000) + 200 × ind2 + 800 × ind1
157      + (if SCHOON then 0 else PAKET[n] × 4000); if lex then
158      begin SYNTAX[m]:= 2 × n - (if SCHOON then 1 else 0);
159         lex:= false
160      end;
161      SCHOON:= ¬SCHOON
162   end
163   else if mod4 = 3 then
164   begin mod4:= 0; k:= k + 1; DIC[k]:= symbol end
165   else
166   begin mod4:= mod4 + 1; DIC[k]:= DIC[k] × 90 + symbol end;
167   goto ASSIGN
168      end;
169      maxl:= len(dicn);
170      begin integer array ewl[0:maxl], bwl[1:maxl];
171   switch SIND:= IND0, IND1, IND2;
172   switch INDAF:= AFF0, AFF1, AFF2;
173   switch Prenpo:= P100, P101, P110, P111, P000, P001;
174   switch SCHAK:= U0, U1, U2;
```

```
176  integer procedure soort(n); value n; integer n;
177  begin integer p;
178     p:= LETTER[n];
179     soort:= if p < 10 then 2 else if p = 66 ∨ p = 65 then 3
180     else if p = 88 then 4 else 1
181  end;


184  boolean procedure sl(m, n); integer m, n;
185  sl:= (if m = 1 then true else soort(m - 1) = 3) ∧ (if n =
186  letter then true else soort(n + 1) = 3);


200  procedure Meq(m, n); integer m, n;
201  begin integer p, q, r, P, Q, R, s, AD, snee, rest, divid,
202     last, u, MEQ, morfeem;
203     morfeem:= n + 1 - m;
204     if morfeem > maxl then MEQ:= 0 else if bwl[morfeem] = 0
205     then MEQ:= 0 else
206     begin divid:= morfeem : 4; last:= divid × 4;
207        rest:= morfeem - last; snee:= divid + sign(rest);
208        begin integer array SNEE[1:snee];

210           integer procedure SEQ(p); integer p;
211           begin integer q;
212             q:= 1; s:= wodic[p];
213           F: AD:= sign(SNEE[q] - DIC[s + q - 1]);
214             if AD ≠ 0 then SEQ:= AD else if q = snee then
215             SEQ:= 0 else
216             begin q:= q + 1; goto F end
217           end;

219           for q:= 1 step 1 until divid do
220           begin u:= 4 × q + m - 1;
221             SNEE[q]:= LETTER[u - 3] × 729000 + LETTER[u - 2]
222             × 8100 + LETTER[u - 1] × 90 + LETTER[u]
223           end;
224           if rest ≠ 0 then
225           begin SNEE[snee]:= 0;
226             for q:= last + 1 step 1 until morfeem do
227             SNEE[snee]:= 90 ↑ (morfeem - q) × LETTER[m + q -
228             1] + SNEE[snee]
229           end;
230           p:= bwl[morfeem]; r:= ewl[morfeem]; P:= SEQ(p);
231           R:= SEQ(r);
232        G: if r < p + 1 then MEQ:= if P = 0 then p else if
233           R = 0 then r else 0 else
234           begin q:= (p + r) : 2; Q:= SEQ(q);
235             if Q = 0 then MEQ:= q else if Q = - 1 then
236             begin r:= q; goto G end
237             else
```

```
238             begin p:= q; goto G end
239           end
240         end
241       end;
242       WOORD[woord]:= MEQ
243   end;


246   integer procedure lv(m, n); value m, n; integer m, n;
247   if n < m then lv:= 0 else
248   begin integer q, t, v, MS, VMS, WO;
249       switch MSVMS:= VMS0, VMS1, VMS2, VMS3, VMS4, VMS5, VMS6,
250       VMS7, VMS8;

252       procedure goback; if m = v then lv:= 0 else

264       begin v:= v - 1; goto K end;

266       q:= m + maxl - 1; v:= n;
267   K:  t:= m; VMS:= soort(m);
268   W:  if t = n then
269       begin if VMS = 4 then goto VMS4 end
270       else
271       begin t:= t + 1; VMS:= prod[VMS,soort(t)]; goto W end;
272       WO:= LETTER[n];
273       if WO > 66 ∧ WO < 75 then MS:= 0 else goto MSVMS[VMS + 1];
274   VMS0: MS:= 0; goto EIND;
275   VMS1: MS:= 1; goto EIND;
276   VMS3: MS:= if m = n then 3 else 0; goto EIND;
277   VMS4: MS:= 0; goto EIND;
278   VMS2:
279   VMS5: MS:= if (if m = 1 then true else LETTER[m - 1] =
280       65) ∧ (if n = letter then true else LETTER[n + 1] = 65)
281       then 2 else 0; goto EIND;
282   VMS6: MS:= if soort(m) = 3 ∨ soort(n) = 3 then 0 else 1;
283       goto EIND;
284   VMS7: MS:= if ¬sl(m, n) ∨ soort(n) ≠ 4 then 0 else 2;
285       goto EIND;
286   VMS8: MS:= if ¬sl(m, n) ∨ LETTER[m] = 88 then 0 else 2;
287   EIND: if MS = 0 then goback else if MS = 1 then
288       begin if v > q then goback else
289           begin Meq(m, v);
290               if WOORD[woord] = 0 then goback else lv:= v
291           end
292       end
293   end;


296   procedure positie;
297   begin integer MEQ, synt1, synt2, test1, test2;
298       MEQ:= WOORD[woord]; tractk:= synt2:= SYNTAX[MEQ + 1] - 1;
```

```
299      if pakt = 0 then tractk:= synt2:= synt2 - 1;
300      VALIND(woord); synt1:= SYNTAX[MEQ]; ind1:= INDEX[woord,1];
301      ind2:= INDEX[woord,2];
302      for tractk:= synt2 - 1 step - 1 until synt1 do
303      begin VALIND(woord); test1:= INDEX[woord,1];
304         test2:= INDEX[woord,2]; if test1 ≠ ind1 then ind1:= 0;
305         if test2 × ind2 = 6 then ind2:= 2 else if test2 ≠ ind2
306         then ind2:= 0
307      end
308   end;


311   procedure close; if finit < woord then
312   begin if tract[finit] = sup[finit] then
313      begin
314      A: finit:= finit + 1; if finit ≠ woord + 1 then
315         begin if inf[finit] = sup[finit] then goto A end
316      end
317   end;


331   procedure transform(change); value change; integer change;
332   begin if finit = woord + 1 then
333      begin FIN:= FIN + 2; goto if FIN = 2 then G else BOOMBOUW
334      end
335      else
336      begin k:= change;
337      R1: if tract[k] ≠ sup[k] then
338         begin tractk:= tract[k]:= tract[k] + 1;
339            if FIN = 1 then
340            begin FIN:= 3; goto BOOMBOUW end;
341            VALIND(k);
342            for m:= k + 1 step 1 until woord do
343            begin tractk:= tract[m]:= inf[m]; VALIND(m) end;
344            close; goto PP
345         end
346         else if k = finit ∨ k = 1 then
347         begin FIN:= FIN + 2;
348            goto if FIN = 2 then G else BOOMBOUW
349         end
350         else
351         begin k:= k - 1; goto R1 end
352      end
353   end;


356   procedure structur(k, m); value k, m; integer k, m;
357   begin integer p, q, r, s, adj;
358      if k > m then goto END;
359      begin boolean array klaar[k:m];
```

```
360         for p:= k step 1 until m do klaar[p]:= false; p:= k;
361         q:= m;
362      G: if p < m then
363         begin integer r, indp, afp, p1, indr;
364            boolean INDP;
365            p1:= INDEX[p + 1,4]; indp:= INDEX[p,4];
366            subj[p]:= p + 1; afp:= affin[indp,p1];
367            INDP:= indp = 1 ∨ indp = 2;
368            for r:= p + 1 step 1 until q do
369            begin indr:= INDEX[r,4];
370              if indr = 0 ∧ r < m ∧ INDP then
371              begin for s:= p step 1 until r - 1 do
372                begin if INDEX[s,4] = 1 ∨ INDEX[s,4] = 2 then
373                  adj:= s
374                end;
375                subj[adj]:= r
376              end
377              else if indr ≠ 6 then
378              begin if affin[indp,indr] > afp then
379                begin subj[p]:= r; afp:= affin[indp,indr] end
380              end
381              else
382              begin normaal[r - 1]:= normaal[r]:= false;
383                s:= r + 1; NORMAAL:= false;
384                if s > m then goto END;
396              H: if INDEX[s,4] = INDEX[r - 1,4] then
397                begin subj[r - 1]:= subj[r]:= s;
398                  structur(r + 1, s); p:= s; goto G
399                end
400                else if s = m then CHANGE:= false else
401                begin s:= s + 1; goto H end
402              end
403            end;
404            klaar[subj[p]]:= klaar[p]:= true;
405            structur(p + 1, subj[p]); p:= subj[p];
406            if p = m then goto END else
407            begin q:= p + 1;
408            A: if klaar[q] then r:= q else if q = m then r:=
409              0 else
410              begin q:= q + 1; goto A end
411            end;
412            q:= if r = 0 then m else r; goto G
413         end
414      end;
415   END:
416   end;


419   procedure compl(n, k, m); value n, k, m; integer n, k, m;
420   begin integer d, ACHTER, bsuff, lpref, radik, indl, lpref2,
```

```
421     begin, eind, leider, kern, aff, bsuff1, leider2, eind2,
422     test, test1, d2, p27;
423     integer array P27[1:5];
424     switch Ik1m:= Ik2Im2, Ik2Im3, Ik3Im2, Ik3Im3;

426     procedure mistest;
427     begin tractk:= tract[test];
428        if LETTER[lasta[test - 1]] ≠ 28 ∧ pakt = 1641 ∧
429        LETTER[unua[test]] = 29 then transform(test)
430     end;


433     procedure optimum;
434     begin integer aff2, lpref2, subkern;
435        subkern:= subj[kern]; if subkern = 0 then goto Y;
436        lpref2:= INDEX[subkern,3];
437        aff:= affix[INDEX[kern,4],lpref2];
438        tractk:= tract[kern];
439     X: if tractk ≠ sup[kern] then
440        begin tractk:= tractk + 1; VALIND(kern);
441           if INDEX[kern,3] ≠ 2 then goto X else
442           begin aff2:= affix[INDEX[kern,4],lpref2];
443              if aff2 > aff then
444              begin aff:= aff2; tract[kern]:= tractk end;
445              goto X
446           end
447        end;
448        tractk:= tract[kern]; VALIND(kern);
449     Y:
450     end;


464     procedure P27op;
465     begin for d:= p27 step - 1 until 1 do
466        begin begin:= P27[d]; indl:= INDEX[begin,3]; aff:= 0;
467           eind:= begin + 1;
468           if INDEX[eind,4] = 7 then eind:= subj[eind];
469        EE: eind2:= INDEX[eind,4];
470           lpref:= affix[eind2,indl]; if lpref > aff then
471           begin subj[begin]:= eind; aff:= lpref end;
472           eind:= subj[eind]; if eind > begin then goto EE;
473           if subj[begin] = 0 then
474           begin CHANGE:= false; goto END end
475        end;
476        goto END
477     end;


480     procedure terugschuif;
481     begin
482     J: d:= begin - 1; if INDEX[d,4] = 7 then
```

```
483         begin begin:= d;
484            goto if begin = k then (if m = eind then gesloten
485            else openeind) else J
486         end
487      end;

489      CHANGE:= true; p27:= 0; if k = m then
490      begin lblok[n]:= m; goto END end;
491      if k > m then goto END;
492      for d:= k step 1 until m do
493      begin subj[d]:= 0 end;
494      lpref:= k;
495   F: if INDEX[lpref,3] > 2 then
496      begin if INDEX[lpref,4] = 7 then
497         begin p27:= p27 + 1; P27[p27]:= lpref end;
498         lpref:= lpref + 1; goto F
499      end
500      else lpref:= lpref - 1; bsuff:= m;
501   G: if INDEX[bsuff,3] = 2 then bsuff:= bsuff + 1 else
502      begin bsuff:= bsuff - 1; goto G end;
503      kern:= bsuff - 1;
504      goto IkIm[sign(2 - INDEX[k,3]) × 2 - sign(INDEX[m,3] - 2)
505      × sign(INDEX[m,3] - 2) + 4];
506   Ik2Im2: bsuff1:= INDEX[bsuff,3];
507      leider2:= INDEX[bsuff - 1,4];
508      if bsuff1 > 2 ∧ affix[leider2,bsuff1] ≠ 0 then goto H
509      else if bsuff1 = 0 then
510      begin if ¬voorschak[leider2,INDEX[bsuff,4]] ∧
511         affix[leider2,INDEX[lpref,3]] ≠ 0 then
512      H: begin for d:= k step 1 until m do
513            begin subj[d]:= 0 end;
514            begin:= lpref + 1; leider:= eind:= bsuff - 1;
515            structur(begin, eind);
516         open: terugschuif; leider2:= INDEX[leider,4];
528            test:= eind + 1; eind2:= INDEX[test,4];
529            lpref2:= INDEX[test,3];
530            if lpref2 > 2 then ACHTER:= affix[leider2,lpref2]
531            else if voorschak[leider2,eind2] then
532            begin if test = m then
533               begin subj[m]:= leider; goto openbegin end
534               else if suffix[eind2,INDEX[eind + 2,4]] then
535               begin subj[test]:= leider; eind:= test; goto open
536               end
537               else ACHTER:= 0
538            end;
539            d:= begin - 1; aff:= affix[leider2,INDEX[d,3]];
540            if ACHTER + aff = 0 then transform(d) else if aff
541            > ACHTER ∨ (aff > 0 ∧ leider2 = 3) then
542            begin begin:= d; subj[leider]:= begin;
543               leider:= begin; if begin ≠ k then goto open
544            end
```

```
545           else
546           begin mistest; eind:= test; subj[leider]:= eind;
547             leider:= eind;
548             goto if m = eind then openbegin else open
549           end;
550         openeind: test:= eind + 1; eind2:= INDEX[test,4];
551           leider2:= INDEX[leider,4]; lpref2:= INDEX[test,3];
552           if lpref2 > 2 then
553           begin if affix[leider2,lpref2] = 0 then
554             transform(eind) else
555             begin mistest; eind:= test; subj[leider]:= eind;
556               leider:= eind;
557               goto if m = eind then gesloten else openeind
558             end
559           end
560           else if voorschak[leider2,eind2] then
561           begin if test = m then
562             begin subj[m]:= lblok[n]:= leider; P27op end
563             else if suffix[eind2,INDEX[eind + 2,4]] then
564             begin subj[test]:= leider; eind:= test;
565               goto openeind
566             end
567             else transform(eind)
568           end;
569         openbegin: terugschuif;
570           if affix[INDEX[leider,4],INDEX[begin - 1,3]] = 0
571           then transform(m) else
572           begin begin:= begin - 1; subj[leider]:= begin;
573             leider:= begin; if begin ≠ k then goto openbegin
574           end;
575         gesloten: lblok[n]:= leider; optimum; P27op
576         end
577       end
578       else
579       begin compl(n, k, bsuff - 1); leider:= lblok[n]; d:= bsuff;
580         if ¬CHANGE then
581         begin CHANGE:= true; goto H end;
582       L: leider2:= INDEX[leider,4]; d2:= INDEX[d,4];
594         lpref2:= INDEX[d,3]; if lpref2 > 2 then
595         begin aff:= affix[leider2,lpref2];
596           if aff = 0 then goto H else
597           begin subj[leider]:= d; if d = m then
598             begin lblok[n]:= m; optimum; P27op end
599             else
600             begin leider:= d; d:= d + 1; goto L end
601           end
602         end
603         else if voorschak[leider2,d2] then
604         begin if d = m then
605           begin subj[m]:= lblok[n]:= leider; optimum; P27op
606           end
607           else if suffix[d2,INDEX[d + 1,4]] then
```

```
608            begin subj[d]:= leider; d:= d + 1; goto L end
609            else goto H
610         end
611         else goto H
612      end;
613   Ik2Im3: d:= k;
614   B: if d < lpref then
615      begin if INDEX[d,4] < 7 then
616         begin subj[d + 1]:= d end;
617         d:= d + 1; goto B
618      end;
619      radik:= lpref + 1; indl:= INDEX[lpref,4];
620   C: aff:= affix[INDEX[radik,4],INDEX[lpref,3]];
621      if aff ≠ 0 then
622      begin if indl < 7 then
623         begin subj[radik]:= lpref; lpref2:= INDEX[lpref,3] end
624      end
625      else if radik = m then
626      begin lblok[n]:= m; CHANGE:= false; goto END end
627      else
628      begin radik:= radik + 1; goto C end;
629      structur(lpref + 1, radik); lpref2:= d:= radik + 1;
630   D: if d = m + 1 then
631      begin if lpref2 > m then lpref2:= m;
632         structur(lpref2 + 1, m); structur(lpref + 1, lpref2);
633         if indl < 7 then
634         begin subj[lpref2]:= lpref end;
635         if lpref2 < m then
636         begin subj[lpref]:= m end;
637         for p:= k step 1 until m do
638         begin if subj[p] = 0 ∧ INDEX[p,4] < 7 then
639            begin lblok[n]:= p; optimum; P27op end
640         end
641      end
642      else
643      begin if affin[indl,INDEX[d,4]] >
644         affin[indl,INDEX[lpref2,4]] then lpref2:= d; d:= d + 1;
645         goto D
646      end;
647   Ik3Im3: structur(k, m); lblok[n]:= m; goto END;
648   Ik3Im2: bsuff1:= INDEX[bsuff,3]; radik:= kern;
660      if bsuff1 = 0 then
661      begin lpref2:= INDEX[bsuff,4];
662         if voorschak[INDEX[radik,4],lpref2] ∧
663         suffix[lpref2,INDEX[bsuff + 1,4]] then
664         begin subj[bsuff]:= radik; bsuff:= bsuff + 1;
665            goto Ik3Im2
666         end
667      end;
668      aff:= affix[INDEX[radik,4],bsuff1]; if aff = 0 then
669      begin if tract[radik] ≠ sup[radik] then transform(radik)
```

```
670         else transform(bsuff)
671       end
672       else
673       begin test:= bsuff; mistest; subj[radik]:= bsuff;
674          leider:= bsuff; lpref2:= INDEX[test,3]
675       end;
676       test1:= bsuff;
677       for test:= bsuff + 1 step 1 until m do
678       begin if INDEX[test,3] = 0 then
679          begin subj[test]:= test - 1 end
680          else
681          begin if affix[INDEX[test1,4],INDEX[test,3]] ≠ 0 then
682             begin mistest; subj[test1]:= test end
683             else if tract[test1] ≠ sup[test1] then
684             transform(test1) else transform(test); test1:= test
685          end
686       end;
687       indl:= INDEX[bsuff,4];
688       for test:= bsuff - 2 step - 1 until k do
689       begin aff:= affin[INDEX[test,4],indl] -
690          affin[INDEX[test,4],INDEX[radik,4]];
691          if aff > 0 ∨ (aff = 0 ∧ affin[INDEX[test,4],indl] >
692          affin[INDEX[test,4],INDEX[test + 1,4]]) then radik:=
693          test
694       end;
695       subj[radik]:= bsuff; structur(radik + 1, bsuff - 1);
696       structur(k, radik - 1); if radik > bblok(n) then
697       begin subj[radik - 1]:= if affin[INDEX[radik -
698          1,4],INDEX[radik,4]] > affin[INDEX[radik -
699          1,4],INDEX[bsuff,4]] then radik else bsuff
700       end;
701       lblok[n]:= m; optimum;
702    END:
703    end;


706    integer procedure maxaffin(p, r); value p, r; integer p, r;
707    begin integer indr, q, MAX;
708       indp:= INDEX[lblok[p],4]; indr:= INDEX[lblok[r],4];
709       if INDEX[bblok(r),3] > 2 then
710       begin MAX:= lblok[r]; afp1:= affin[indp,INDEX[MAX,4]];
711          maxaffin:= MAX
712       end
713       else
714       begin MAX:= bblok(r); afp1:= affin[indp,INDEX[MAX,4]];
726          for q:= bblok(r) + 1 step 1 until thblok[r] do
727          begin if affin[indp,INDEX[q,4]] > afp1 then
728             begin MAX:= q; afp1:= affin[indp,INDEX[q,4]] end
729          end;
730          maxaffin:= MAX
731       end
732    end;
```

```
735  procedure macrostructur(k, m); value k, m; integer k, m;
736  begin if k > m then goto END;
737     begin integer p, q, r, s, adj, MAX;
738        boolean array klaar[k:m];
739        for p:= k step 1 until m do klaar[p]:= false; p:= k;
740        q:= m;
741     G: if p < m then
742        begin integer r;
743           reg[p]:= p + 1; subj[lblok[p]]:= maxaffin(p, p + 1);
744           afp:= afp1;
745           for r:= p + 1 step 1 until q do
746           begin indr:= INDEX[lblok[r],4];
747              if indr = 0 ∧ r < m ∧ (indp = 1 ∨ indp = 2) then
748              begin for s:= p step 1 until r - 1 do
749                 begin if INDEX[lblok[s],4] = 1 ∨
750                    INDEX[lblok[s],4] = 2 then adj:= s
751                 end;
752                 subj[lblok[adj]]:= lblok[r]; reg[adj]:= r
753              end
754              else if indr ≠ 6 then
755              begin if INDEX[bblok(r),3] > 2 then
756                 begin if affin[indp,indr] > afp then
757                    begin reg[p]:= r; subj[lblok[p]]:= lblok[r];
758                       afp:= affin[indp,indr]
759                    end
760                 end
761                 else
762                 begin MAX:= maxaffin(p, r);
763                    if affin[indp,INDEX[MAX,4]] > afp then
764                    begin reg[p]:= r; subj[lblok[p]]:= MAX;
765                       afp:= afp1
766                    end
767                 end
768              end
769              else
770              begin normaal[lblok[r - 1]]:= normaal[lblok[r]]:=
771                 false; s:= r + 1; NORMAAL:= false;
772                 if s > m then goto END;
773              H: if INDEX[lblok[s],4] = INDEX[lblok[r - 1],4]
774                 then
775                 begin reg[r - 1]:= reg[r]:= s;
776                    subj[lblok[r - 1]]:= subj[lblok[r]]:= s;
777                    macrostructur(r + 1, s); p:= s; goto G
778                 end
779                 else if s = m then CHANGE:= false else
780                 begin s:= s + 1; goto H end
792              end
793           end;
794           klaar[reg[p]]:= klaar[p]:= true;
795           macrostructur(p + 1, reg[p]); p:= reg[p];
796           if p = m then goto END else
```

```
797             begin q:= p + 1;
798             A: if klaar[q] then r:= q else if q = m then r:=
799               0 else
800               begin q:= q + 1; goto A end
801             end;
802             q:= if r = 0 then m else r; goto G
803           end
804       end;
805   END:
806   end;

808   for k:= 0 step 1 until 8 do
809   begin for m:= 1 step 1 until 4 do prod[k,m]:= 0 end;
810   prod[1,1]:= 1; prod[2,2]:= 2;
811   prod[1,2]:= prod[2,1]:= prod[5,1]:= prod[5,2]:= 5;
812   prod[1,3]:= prod[3,1]:= prod[6,1]:= prod[6,3]:= 6;
813   prod[1,4]:= prod[7,1]:= prod[7,4]:= 7;
814   prod[2,4]:= prod[8,2]:= 8;
815   for k:= 0 step 1 until 6 do
816   begin for m:= 3 step 1 until 19 do affix[k,m]:= 0 end;
817   affix[0,3]:= affix[3,6]:= affix[4,8]:= affix[1,9]:=
818   affix[0,10]:= affix[0,11]:= affix[4,12]:= 1;
819   for k:= 2, 4, 5 do
820   begin for m:= 13 step 1 until 19 do affix[k,m]:= 1 end;
821   affix[2,5]:= affix[4,7]:= affix[1,8]:= affix[0,9]:=
822   affix[1,10]:= affix[3,11]:= affix[1,12]:= affix[3,13]:=
823   affix[2,14]:= affix[3,15]:= affix[1,16]:= affix[0,17]:=
824   affix[0,18]:= affix[1,19]:= 2;
825   affix[0,12]:= affix[1,13]:= affix[5,7]:= affix[0,15]:=
826   affix[0,16]:= affix[1,17]:= affix[3,18]:= affix[0,19]:=
827   affix[3,19]:= 3;
828   affix[0,13]:= affix[1,15]:= affix[3,16]:= affix[3,17]:=
829   affix[1,18]:= 4; affix[1,4]:= affix[1,14]:= 5;
830   affix[3,14]:= 6; affix[0,14]:= 7;
831   for k:= 0 step 1 until 6 do
832   begin for m:= 0 step 1 until 6 do affin[k,m]:= 0 end;
833   affin[0,3]:= affin[1,3]:= affin[1,4]:= affin[2,0]:=
834   affin[3,3]:= affin[4,1]:= affin[4,6]:= affin[5,6]:=
835   affin[6,6]:= 1;
836   affin[0,1]:= affin[1,0]:= affin[1,1]:= affin[3,0]:=
837   affin[4,0]:= affin[4,3]:= affin[5,1]:= 2;
838   affin[5,0]:= affin[5,3]:= 3; affin[5,4]:= affin[5,5]:= 4;
839   affin[0,0]:= affin[2,2]:= affin[4,4]:= 5;
840   for k:= 0 step 1 until 6 do
841   begin for m:= 0 step 1 until 3 do voorschak[k,m]:= false end;
842   voorschak[0,0]:= voorschak[1,0]:= voorschak[3,0]:=
843   voorschak[4,0]:= voorschak[0,1]:= voorschak[1,1]:=
844   voorschak[2,1]:= voorschak[3,1]:= voorschak[0,2]:=
845   voorschak[5,2]:= voorschak[0,3]:= true;
846   for m:= 0 step 1 until 3 do
858   begin for n:= 0 step 1 until 6 do naschak[m,n]:= false end;
```

```
859  naschak[0,0]:= naschak[0,1]:= naschak[0,3]:= naschak[1,0]:=
860  naschak[1,1]:= naschak[1,3]:= naschak[2,0]:= naschak[2,1]:=
861  naschak[3,0]:= naschak[3,1]:= true;
862  for m:= 0 step 1 until 3 do
863  begin for n:= 0 step 1 until 4 do suffix[m,n]:= false end;
864  for m:= 0, 2 do
865  begin for n:= 0, 3, 4 do suffix[m,n]:= true end;
866  r:= 0;
867  for p:= 1 step 1 until dicn do
868  begin q:= r; r:= len(p); s:= r - 1; if q ≠ r then
869     begin for m:= q + 1 step 1 until s do bwl[m]:= ewl[m]:= 0;
870        bwl[r]:= p; ewl[q]:= p - 1
871     end
872  end;
873  ewl[maxl]:= dicn;
874  for k:= 1 step 1 until 8 do PUNLCR; regel:= 9; PUTEXT(
875  ‡ 6.4.  The results‡); PUNLCR;
876  eblok[0]:= diepte[0]:= RETURN:= 0;
877  for k:= 1 step 1 until 7 do
878  begin leider[k,0]:= 0; begin[k,1]:= 1 end;
879  for k:= 1 step 1 until 15 do begin[7,k]:= leider[7,k]:= k;
880  lengte[1]:= begin[1,1]:= kolon[1]:= unua[1]:= 1;
881     LEES: letter:= FIN:= 0; woord:= 1;
882     AB: symbol:= nexttape;
883  if symbol = 76 then EXIT else if symbol = 93 ∨ symbol = 118
884  ∨ symbol = 119 ∨ symbol = 135 then RETURN:= if RETURN = 0
885  then 1 else 3 else
886  begin letter:= letter + 1; LETTER[letter]:= symbol; goto AB
887  end;
888  r:= lv(1, letter); woord:= 1;
889     L: if r = 0 then
890  begin FIN:= 2; lasta[1]:= letter; goto BOOMBOUW end
891  else lasta[1]:= r; positie; if r = letter then
892  begin if ind2 < 2 ∧ ind1 < 2 then
893     begin woord:= 1; goto SUBJAS end
894     else
895     begin r:= lv(1, letter - 1); goto L end
896  end
897  else
898  begin if ind1 > 1 then
899     begin r:= lv(1, r - 1); goto L end
900     else woord:= 2;
901  M: EC:= lasta[woord - 1] + 1; AP:= LETTER[EC];
902     q:= unua[woord]:= EC + (if AP = 65 ∨ AP = 66 ∨ AP = 70
903     then 1 else 0); lasta[woord]:= r:= lv(q, letter);
904  F: if r = 0 then goto G else
905     begin positie; if letter = r then
906        begin if ind2 > 1 then
907           begin r:= lv(q, letter - 1); goto F end
908           else goto SUBJAS
909        end
```

```
910         else
911         begin if ind2 = 1 ∨ ind1 = 1 then
912            begin r:= lv(q, r - 1); goto F end
924            else
925            begin lasta[woord]:= r; woord:= woord + 1; goto M
926            end
927         end
928      end
929   end;
930      G: FIN:= 0; woord:= woord - 1; q:= unua[woord];
931   if woord = 1 then
932   begin r:= lv(1, lasta[1] - 1); goto L end
933   else
934   begin r:= lv(q, lasta[woord] - 1); goto F end;
935      SUBJAS: lasta[woord]:= letter; finit:= 1;
936   for m:= 1 step 1 until woord do subj[m]:= 0;
937   for k:= 1 step 1 until woord do
938   begin inf[k]:= SYNTAX[WOORD[k]];
939      tractk:= sup[k]:= SYNTAX[WOORD[k] + 1] - 1;
940      if pakt = 0 then sup[k]:= tractk - 1
941   end;
942   k:= 1;
943      N: for u:= k step 1 until woord do
944   begin tractk:= tract[u]:= inf[u]; VALIND(u) end;
945   close;
946      PP: if INDEX[1,1] = 2 then transform(1);
947   if INDEX[woord,2] > 1 then transform(woord); k:= 1;
948   for u:= 1 step 1 until woord - 1 do
949   begin if INDEX[u,2] = 1 then transform(u) else if INDEX[u,2]
950      = 3 then
951      begin aff1:= LETTER[unua[u + 1]];
952         if (aff1 = 10 ∨ aff1 = 14 ∨ aff1 = 18 ∨ aff1 = 24 ∨
953         aff1 = 30 ∨ aff1 = 34) ∧ INDEX[u + 1,1] = 2 ∧ INDEX[u
954         + 1,3] ≠ 2 then  else transform(u)
955      end
956   end;
957   for u:= 2 step 1 until woord do
958   begin if INDEX[u,1] = 1 then transform(u) end;
959   for u:= 1 step 1 until woord do normaal[u]:= true;
960   NORMAAL:= true; aff1:= 0;
961      MM: ebl:= bblok(k); aff1:= aff1 + 1; FASE:= 0;
962   if k = 1 then goto QQ else if eblok[k - 1] = woord then
963   begin blokn:= k - 1; goto S end
964   else
965      QQ:
966   begin
967   K: lem:= INDEX[aff1,3];
968      goto if lem > 2 then IND3 else SIND[lem + 1]
969   end;
970      AFF0:
971      AFF1: transform(p);
```

```
972     AFF3: if INDEX[p,1] = 2 then transform(p) else
973  begin eblok[k]:= p; p:= p + 1; goto LL end;
974     AFF2: aff1:= p; goto K;
975     INDO: compl(k, bblok(k), thblok[k]);
976  if (bblok(k) = eblok[k] ∧ INDEX[eblok[k],3] ≠ 2) ∨ aff1 =
977  woord then transform(aff1) else if
978  ¬voorschak[INDEX[lblok[k],4],INDEX[aff1,4]] then
990  begin if tract[aff1 - 1] = sup[aff1 - 1] then transform(aff1)
991     else transform(aff1 - 1)
992  end
993  else
994  begin subj[aff1]:= lblok[k]; eblok[k]:= aff1; u:= aff1 + 1;
995     u1:= INDEX[u,3] + 1; if u1 > 3 then
996     begin if INDEX[u,1] = 2 then
997        begin if suffix[INDEX[aff1,4],INDEX[u,4]] then
998           begin aff1:= u; goto IND3 end
999           else transform(u)
1000       end
1001       else goto MM
1002    end
1003    else goto SCHAK[u1]
1004 end;
1005    UO:
1006    U1: transform(aff1);
1007    U2: k:= k + 1; goto MM;
1008    IND1: if aff1 > ebl then
1009 begin compl(k, bblok(k), thblok[k]);
1010    if INDEX[lblok[k],4] = INDEX[aff1,4] then
1011    begin subj[aff1]:= lblok[k]; eblok[k]:= aff1;
1012       if FASE < 4 then
1013       begin FASE:= FASE + 2 end
1014       else transform(aff1); if aff1 ≠ woord then
1015       begin aff1:= aff1 + 1;
1016          if INDEX[aff1,3] = 1 then transform(aff1) else
1017          goto K
1018       end
1019    end
1020    else transform(aff1)
1021 end
1022 else transform(aff1); goto RR;
1023    IND3: if INDEX[aff1,1] ≠ 2 then
1024 begin if aff1 ≠ 1 then k:= k + 1; lblok[k]:= aff1;
1025    p:= aff1 + 1; FASE:= 1;
1026 LL: if p > woord then transform(woord) else
1027    begin indp:= INDEX[p,3] + 1;
1028       goto if indp > 3 then AFF3 else INDAF[indp]
1029    end
1030 end
1031 else if aff1 ≠ ebl then
1032 begin if FASE > 3 then transform(aff1) else
1033    begin FASE:= 3 end;
1034    eblok[k]:= thblok[k]:= aff1; if aff1 < woord then
```

```
1035      begin aff1:= aff1 + 1; goto K end
1036   end;
1037   goto RR;
1038      IND2: if FASE > 1 then
1039   begin eblok[k]:= aff1 - 1; if aff1 > 1 then k:= k + 1;
1040      thblok[k]:= eblok[k]:= lblok[k]:= aff1;
1041      if aff1 ≠ woord then
1042      begin aff1:= aff1 + 1; goto K end
1043   end
1044   else
1056   begin thblok[k]:= eblok[k]:= aff1; FASE:= FASE + 2;
1057      if aff1 ≠ woord then
1058      begin aff1:= aff1 + 1; goto K end
1059   end;
1060      RR: if eblok[k] < woord then
1061   begin k:= k + 1; goto MM end
1062   else blokn:= k;
1063      S: for m:= 1 step 1 until blokn do
1064   begin compl(m, bblok(m), thblok[m]);
1065      if ¬CHANGE then transform(thblok[m]);
1066      if thblok[m] ≠ eblok[m] then
1067      begin subj[eblok[m]]:= lblok[m] end;
1068      if INDEX[eblok[m],3] = 0 then
1069      begin NASCHAK:= true;
1070         for k:= m + 1 step 1 until blokn do
1071         begin compl(k, bblok(k), thblok[k]);
1072            if ¬CHANGE then transform(thblok[k]);
1073            if naschak[INDEX[eblok[m],4],INDEX[lblok[k],4]]
1074            then NASCHAK:= false
1075         end;
1076         if NASCHAK then transform(thblok[blokn])
1077      end;
1078      if ¬CHANGE then
1079      begin k:= eblok[m];
1080      B: if k = woord + 1 then transform(woord) else if
1081         tract[k] = sup[k] then
1082         begin k:= k + 1; goto B end
1083         else
1084         begin tractk:= tract[k]:= tract[k] + 1; VALIND(k);
1085            if finit = woord + 1 then
1086            begin FIN:= FIN + 2; goto BOOMBOUW end;
1087            close; k:= k + 1; goto N
1088         end
1089      end
1090   end;
1091   FIN:= 1; macrostructur(1, blokn);
1092   for k:= 1 step 1 until blokn do
1093   begin for m:= bblok(k) step 1 until eblok[k] do blok[m]:= k
1094   end;
1095      BOOMBOUW: if FIN = 2 then
1096   begin lasta[1]:= letter; if regel > 52 then
```

```
1097      begin for k:= regel + 1 step 1 until 71 do PUNLCR;
1098         regel:= 5
1099      end;
1100      PUNLCR; PUNLCR; PUNLCR; regel:= regel + 3; PUTEXT(‡,,‡);
1101      PUCAS(1); PUTEXT(‡"niet ontleedbaar‡)
1102   end
1103   else if FIN ≠ 0 then
1104   begin hoogte:= 1;
1105      for k:= 1 step 1 until woord do
1106      begin if subj[k] = 0 then
1107         begin leider[1,1]:= k; diepte[k]:= 1 end
1108         else diepte[k]:= 20
1109      end;
1110      lem:= 1; begin[1,2]:= woord + 1;
1122   SS: if lem < woord then
1123      begin leider[hoogte,lem + 1]:= woord + 1;
1124         hoogte:= hoogte + 1; lem1:= 0;
1125         for n:= 1 step 1 until lem do
1126         begin OMK:= leider[hoogte - 1,n];
1127            predom:= endom:= postdom:= 0;
1128            for k:= leider[hoogte - 1,n - 1] + 1 step 1 until
1129            OMK - 1 do
1130            begin if subj[k] = OMK then
1131               begin predom:= k;
1132                  if normaal[k] then goto POSTDOM else
1133                  begin for k:= predom + 1 step 1 until OMK - 1
1134                     do
1135                     begin if subj[k] = OMK ∧ ¬normaal[k] then
1136                        begin endom:= k; goto POSTDOM end
1137                     end
1138                  end
1139               end;
1140            end;
1141         POSTDOM: for k:= OMK + 1 step 1 until leider[hoogte -
1142            1,n + 1] - 1 do
1143            begin if subj[k] = OMK then postdom:= k end;
1144            goto Prenpo[ - sign(predom) × 4 + sign(endom) × 2 +
1145            sign(postdom) + 5];
1146         P101: if blok[predom] = blok[OMK] ∧
1147            (INDEX[postdom,3] > 2 ∨ INDEX[OMK,3] > 2) then
1148            goto P001;
1149         P100: lem1:= lem1 + 2;
1150            leider[hoogte,lem1 - 1]:= predom;
1151            leider[hoogte,lem1]:= OMK; diepte[predom]:= hoogte;
1152            goto verhoog;
1153         P111: if blok[predom] = blok[OMK] ∧
1154            (INDEX[postdom,3] > 2 ∨ INDEX[OMK,3] > 2) then
1155            goto P001;
1156         P110: lem1:= lem1 + 3;
1157            leider[hoogte,lem1 - 2]:= predom;
1158            leider[hoogte,lem1 - 1]:= endom;
```

```
1159            leider[hoogte,lem1]:= OMK;
1160            diepte[predom]:= diepte[endom]:= hoogte;
1161            goto verhoog;
1162         POOT: lem1:= lem1 + 2;
1163            leider[hoogte,lem1 - 1]:= OMK;
1164            leider[hoogte,lem1]:= postdom;
1165            diepte[postdom]:= hoogte; goto verhoog;
1166         POOO: lem1:= lem1 + 1; leider[hoogte,lem1]:= OMK;
1167         verhoog:
1168         end;
1169         begin[hoogte,lem1 + 1]:= leider[hoogte,lem1 + 1]:=
1170         woord + 1; lem:= lengte[hoogte]:= lem1; goto SS
1171      end;
1172      for m:= 2 step 1 until hoogte do
1173      begin for n:= 2 step 1 until lengte[m] do
1174         begin P:= leider[m,n - 1]; R:= leider[m,n];
1175         TT: if R = P + 1 then
1176            begin begin[m,n]:= R end
1188            else
1189            begin Q:= (P + R) : 2;
1190               if dom(Q, leider[m,n - 1], m) then P:= Q else
1191               R:= Q; goto TT
1192            end
1193         end
1194      end;
1195      for m:= 2 step 1 until hoogte - 1 do
1196      begin k:= 1;
1197         for n:= 1 step 1 until lengte[m] do
1198         begin if diepte[leider[m,n]] < m then
1199            begin naasthoger[m,k]:= begin[m,n]; k:= k + 1 end
1200         end;
1201         naasthoger[m,k]:= woord + 1
1202      end;
1203   BLANK XXII: if regel + 4 × hoogte > 58 then
1204      begin for k:= regel + 1 step 1 until 71 do PUNLCR;
1205         regel:= 5
1206      end;
1207      PUNLCR; PUNLCR; PUNLCR; regel:= regel + 3;
1208      for k:= 1 step 1 until woord - 1 do
1209      begin for m:= begin[hoogte,k] step 1 until eind(hoogte, k)
1210         do
1211         begin PUCAS(m);
1212            symbol:= lasta[eind(hoogte,k)] -
1213            unua[begin[hoogte,k]] + 1;
1214            kolon[k + 1]:= kolon[k] + symbol : 8 + (if
1215            REMAINDER(symbol, 8) = 7 then 2 else 1)
1216         end;
1217         PUSYM(118)
1218      end;
1219      for m:= begin[hoogte,woord] step 1 until eind(hoogte,
1220      woord) do PUCAS(m); PUNLCR; regel:= regel + 1;
```

```
1221      notind(hoogte, 1);
1222      for m:= 2 step 1 until woord do
1223      begin if kolon[m - 1] + 1 = kolon[m] then PUSYM(93) else
1224         for k:= kolon[m - 1] + 2 step 1 until kolon[m] do
1225         PUSYM(118); notind(hoogte, m)
1226      end;
1227      if ¬NORMAAL then
1228      begin PUNLCR; regel:= regel + 1;
1229         if normaal[1] then PUTEXT(‹normaal›) else PUTEXT(
1230         ‹koppel›);
1231         for m:= 2 step 1 until woord do
1232         begin symbol:= kolon[m] - kolon[m - 1] - (if normaal[m
1233            - 1] then 1 else 0);
1234            for k:= 1 step 1 until symbol do PUSYM(118);
1235            if normaal[m] then PUTEXT(‹normaal›) else PUTEXT(
1236            ‹koppel›)
1237         end
1238      end;
1239      PUNLCR; regel:= regel + 1;
1240      for k:= 2 step 1 until kolon[leider[hoogte - 1,1]] do
1241      PUSYM(118); if begin[hoogte - 1,2] ≠ 2 then PUTEXT(
1242      ‹leider›);
1254      for m:= 2 step 1 until lengte[hoogte - 1] do
1255      begin for k:= kolon[leider[hoogte - 1,m - 1]] + 1 step 1
1256         until kolon[leider[hoogte - 1,m]] do PUSYM(118);
1257         if begin[hoogte - 1,m] ≠ eind(hoogte - 1, m) then
1258         PUTEXT(‹leider›)
1259      end;
1260      for n:= hoogte - 1 step - 1 until 2 do
1261      begin PUNLCR; PUNLCR; regel:= regel + 2;
1262         for k:= 1 step 1 until lengte[n] - 1 do
1263         begin indn:= begin[n,k]; plaats:= eind(n, k);
1264            KOLON:= kolon[indn];
1265            for m:= indn step 1 until plaats do PUCAS(m);
1266            symbol:= lasta[plaats] - unua[indn] + 1;
1267            KOLON:= KOLON + symbol : 8 + (if REMAINDER(symbol,
1268            8) = 7 then 1 else 0);
1269            for m:= KOLON + 1 step 1 until kolon[begin[n,k + 1]]
1270            do PUSYM(118)
1271         end;
1272         for m:= begin[n,lengte[n]] step 1 until eind(n,
1273         lengte[n]) do PUCAS(m); PUNLCR; regel:= regel + 1;
1274         notind(n, 1); KOLON:= 3;
1275         for k:= 2 step 1 until lengte[n] do
1276         begin if KOLON > kolon[begin[n,k]] then PUSYM(93) else
1277            for m:= KOLON step 1 until kolon[begin[n,k]] do
1278            PUSYM(118); notind(n, k);
1279            KOLON:= kolon[begin[n,k]] + 2
1280         end;
1281      BLANK XXIII: if ¬NORMAAL then
1282         begin PUNLCR; regel:= regel + 1;
```

```
1283            if normaal[leider[n,1]] then PUTEXT(
1284            ‹normaal›) else PUTEXT(‹koppel›);
1285            for m:= 2 step 1 until lengte[n] do
1286            begin symbol:= kolon[begin[n,m]] - kolon[begin[n,m
1287              - 1]] - (if normaal[leider[n,m - 1]] then 1 else
1288              0);
1289              for k:= 1 step 1 until symbol do PUSYM(118);
1290              if normaal[leider[n,m]] then PUTEXT(
1291              ‹normaal›) else PUTEXT(‹koppel›)
1292            end
1293          end;
1294          PUNLCR; regel:= regel + 1;
1295          for k:= 2 step 1 until kolon[naasthoger[n,1]] do
1296          PUSYM(118);
1297          if begin[n,2] ≠ begin[n - 1,2] then PUTEXT(‹leider›);
1298          p:= q:= 1;
1299          for k:= 2 step 1 until eind(n - 1, 1) do
1300          begin if k = begin[n,p + 1] then p:= q:= p + 1 end;
1301          for m:= 2 step 1 until lengte[n - 1] do
1302          begin for k:= kolon[naasthoger[n,m - 1]] + 1 step 1
1303            until kolon[naasthoger[n,m]] do PUSYM(118);
1304            p:= q:= p + 1;
1305            for k:= begin[n - 1,m] step 1 until eind(n - 1, m)
1306            do
1307            begin if k = begin[n,p + 1] then p:= p + 1 end;
1308            if p ≠ q then PUTEXT(‹leider›)
1320          end
1321        end;
1322        PUNLCR; PUNLCR; regel:= regel + 2;
1323        for m:= 1 step 1 until woord do PUCAS(m); PUNLCR;
1324        regel:= regel + 1; notind(1, 1)
1325    end;
1326    if RETURN = 1 ∨ RETURN = 3 then goto LEES
1327      end
1328    end
1329 end
```

6.2. The word-list
(this precedes the lexicon, because the lexicon is adapted to the word-list, 5.)

postchecks abonnementsvoorwaarden nationale gezondheidszorg

vooropstellen Nederlandse gezondheidszorg waaromheen

gezondheidstoestand ziekenfondswege wezenverzorging loonexplosies

vijfdaagse ingrijpende landelijke huisartsenvereniging

angstaanjagend ziekenfondsen uitoefenen gezondheidszorg

Nederlanders artsenstatus beeindiging omzetting verminderen

artsenberoep ziekenfondsen vergaande ziekenfondsen ziekenfondsen

gezondheidszorg bijverschijnselen gezondheidstoestand

ziekenfondsen ziekenfondsen gezondheidszorg uiteindelijk

nationalisering beoefenaar werktijden uitoefening postacademiaal

onvermijdelijk overgangsverschijnselen internationaal langlopende

ontwapening meningsvorming ontwapening internationale

ontwapening halfjaarlijks loonsverhogingen produktiviteit

werkgevers verplichtingen meerjarige loonovereenkomsten

vakbonden Westduitse afbeelden bedenken bedreigt kantonrechter

ontwapent rechtsvervolging provinciale veroordeeld onverbindend

strijdigheid mensenrechten plaatselijke provinciale verordeningen

navlooien mensenrechten ineenstorting bouwonderneming opblazen

regeringssteun gemeenschapshanden beleidsdaden werknemers

Amsterdamse misleidend Amsterdamse belangenspel Amsterdamse

Amsterdamse binnenstadsbeleid misgrepen goedvinden

Amsterdamse vertegenwoordiger architectuurmedewerker

opdrachtgevers loonslaven opdrachtgever grachtenhuizen
regentenpaleis Amsterdamse stadsregenten Amsterdammers
koopmansstad bouwkunstige regentenarchitectuur bouwkundigen
monumentale koopliedenregenten grachtenhuizen culturele
bedillerijen dagelijkse culturele vreemdelingen landgenoten
culturele beursnoteringen stedebouwkundige welstandscommissies
Nederlandse stadsruimte ruilvoorwaarden ongevoelig vakbladen
zesenzeventig woonruimte winkelbedrijvigheid stedebouwers
bouwkunstige internationale winkelhuizen bestuursbeslissingen
stadsbouwmeester beleidsbeslissingen verwerkelijking
oorlogsvoorbereiding regeringsverantwoordelijkheid taakstelling
verkiezingen bekrachtigen veelzeggende verantwoordingsplicht
partijraadsvergadering Amsterdamse houdbaarheid
confessionalisme kiezelstenen regentenmaatregel
vertegenwoordigers verwezenlijking vakbondspecialisten
besprekingen zwaargewichten overschatting invoering
aantrekkingskracht meningsverschillen opvolgingsveten
regeringsleider vooruitstrevende doorslaggevende
verkiezingscampagne verkiezingen beinvloedden
presidentsverkiezingen verzorgingsstaat gelijkberechtigder
verkiezingscampagne bestrijders volksvertegenwoordigers
presidentsverkiezingen verkiezingen gouverneurszetels
verschuivingen verkiezingen verschuivingen tussenverkiezing
toonaangevende veranderingen belastingverhogingen verhoging
consultatiebureau buitenechtelijk onvermijdelijk verkiezingen

verkiezingen geloofsbrieven melkbedrijven voetklachten

sokkenfabrikanten voetartsen partijstromingen schoolgaande

diplomatenkinderen ziekenfondspatient ziekenfondspremie ¬

6.3. The lexicon

<

a 1097, 2030, d 20061, e 22000, 20011, 21013, 20040,

n 22003, 21010, 21013, 21030, 21033, 20060, 20040, 20031, o 22001,

r 20041, 20084, 20120, 22123,

s 22002, 21010, 21011, 21031, 20141, 21154, t 21013, 20041,

u 11020,

ad 23110, af 24, 2033, ak 3020, al 23031, 24, 3020,

an 23030, ap 3020, ar 23031, 23110, 3020, 20,

as 20, at 23110, au 11024, az 3023, 3020, be 2163,

co 2024, de 21013, 20061, 11021, eb 20, 23, eg 20, 23,

ei 20, el 23031, 20063, 20,

en 22003, 21010, 21013, 21030, 21033, 20040, 20031, 20060, 26,

ep 3020,

er 20060, 20030, 20041, 20084, 3074, 11024, 23123, 3023, 3020,

es 20030, 20, et 3023, ex 2025, 2097, ga 23, ge 2060,

ha 11024, ho 11024, ie 20140, ig 20141,

ij 20030, in 25, 20030, 21017, 23, is 23093, 20023,

it 23090, iv 23061, iz 23093, ja 11024,

je 20030, 20, ka 20, ke 20030, la 20, ma 20,

me 11020, na 25, nd 20061, nu 24, of 1026,

og 3020, 3023, om 25, 2083, on 2107, op 25,

or 20060, 3020, os 23031, 20, pa 20, po 20,

ra 20, 11023, re 20041, 2117, st 20041, 20084,

te 21013, 20041, 20040, 20060, 11024, ui 20, un 3022,
ur 3020, 23160, uw 21, we 11020, ze 11020, zo 24, 20,
aak 20, aal 20031, 20, aan 25, 20030,
aar 20060, 20030, 20031, 20, aat 20110, ach 11024,
aks 20, ant 20061, art 3020, bed 20, 23, bei 20,
bek 20, 3020, 23, bel 20, beo 20, beu 1021,
bij 25, 20, bon 20, bui 20, 23, bur 3020,
con 2025, dad 3020, dag 20, 3023, dam 20, 23,
del 3020, 3023, 20, den 21013, 20061, 20, 3020,
der 20061, 11021, 3023, dip 23, dra 1024, een 22,
eer 20063, 20, 23, 21, end 20061, 20, ent 20060, 23,
ere 20041, 22163, erg 24, erv 3023, 3020, eur 20016,
fon 3020, gaa 2023, gel 3021, 3023, 20,
gen 23021, 20, ger 3023, gev 3023, gin 20,
hal 3023, 20, 3020, han 3020, hap 23, 20,
hat 3023, 3020, hee 11024, hei 20, 23, 11024, hij 20,
hit 20, hog 3021, 3023, hui 20, 11024, ial 23031,
iek 20031, iev 23161, ijk 23, 20, ing 20060,
inn 23030, 3023, ion 23060, 20, ist 20130, jag 3023,
jar 3020, jen 23, kan 1023, 20, kel 3020, 3023,
ken 23, 20030, ker 3020, 3023, kin 20, kom 23, 20,
kun 23, lad 3023, lag 3021, 10023, 3020, lan 3020,
las 23, lav 3023, lei 20, 2023, len 3023, 3020,
ler 3023, 3020, lij 20, lis 20, log 20, 21,
lom 3021, lop 3023, 3020, los 21, 23,

man 20, 3020, 3023, mat 3020, 23, 21, 20, mee 24, 20,
mel 3020, men 3023, 11020, mer 3020, 3023,
mij 11020, min 21, 23, 20, mis 2167, 23, 21, 20, mon 3022,
nar 23030, 3021, 20, nat 21, nav 3020, nee 11024,
nem 3023, not 3020, 3023, oef 11024, oer 3097,
oma 20, ons 11020, 20, ont 2143, ooi 20,
oor 20, 2097, oud 21, pal 3020, 21, par 3020, 3023,
pat 20, pel 23, 20, pen 20, 23, 3020, pre 2025,
pro 2025, ree 20, rei 20, 23, rek 23, 20, rel 20, 23,
rem 23, 20, ren 23, 20, rep 3020, 23, rev 3023, rij 20, 23,
roe 20, rom 3020, rov 3023, 3020, rui 20, 23,
sar 23, sel 2060, sla 23, 20, sok 20, sta 23,
sul 20, tal 20, 3020, 3023, tan 3023, tat 23040,
tel 23, 20, 3023, ten 21013, 3020, ter 3023, 3020, 25,
tij 20, tin 20, toe 24, ton 3023, 3020, 20,
tor 20, tri 2022, tur 23060, 3023, uil 20,
uit 25, 23, uur 20, 20160, vak 20, vee 20,
ven 3020, 20, ver 2183, 21, 3020, vet 20, 21, vin 20,
vit 23, vlo 20, vol 21, vor 3020,
wel 24, 20, wer 3023, 3020, win 23, zeg 23, 20,
zel 3020, zen 20, zes 22, zet 23, 20,
zev 3023, 3020, zie 23, zin 20, 23, zon 20, 23, 3020,
aard 20, 23, acht 22, 23, 20, adem 20, 23, ader 20,
akst 20, arch 20020, 3023, arts 20, atie 20060,
baar 20061, 23, 20, 21, bede 20, best 21, beur 23,

bind 23, blad 20, blaz 3023, 3020, bond 20, 1023,
bouw 20, 23, buit 20, 1023, cult 3020, daag 2020, 23,
deel 20, 23, denk 23, demi 2097, 20, ding 20, 23,
duit 20, echt 21, 20, 23, eens 24, egel 20,
eind 20, 23, erij 20060, even 24, ever 20, ezel 20,
gade 20, gang 20, ging 23, goed 21, 20, half 21, 20,
hand 20, heen 24, heid 20040, houd 23, huis 20, 23,
huiv 3020, 3023, huiz 3020, 3023, iaal 20031, iser 23123,
isme 20150, 20, jaar 20, kant 20, 23, kiez 3023, 3020,
kind 20, koop 20, 23, laat 21, 23, lach 23, 20, lade 20,
land 20, 23, lang 21, last 20, leid 23, lied 20,
lijk 20191, 23, 20, ling 20170, liss 3020, list 20,
loof 23, 20, looi 23, loon 20, 23, maat 20, mans 2020,
mede 24, 20, meen 23, meer 21, 20, 23, mees 20,
melk 20, 23, mens 20, ment 20060, mijd 23,
naar 20030, 10025, 21, oord 20, open 21, 23, orde 20,
orgi 3020, over 25, part 20, post 20, 23, 2025,
prek 3023, 3020, raad 20, 23, rand 20, 23, rekk 3023, 3020,
rijd 20, rijp 21, 23, 20, rijv 3023, ring 20, 23,
roep 23, 20, ruil 20, 23, ruim 21, 23, 20, ruit 20,
slag 20, 3023, slav 3020, 3023, slei 20, snot 20,
sokk 3020, spat 23, 20, spel 20, 23, 3023, staa 2023,
stad 20, sted 3020, sten 20040, 3020, 3023,
ster 20060, 20, stor 3023, stro 20, taak 20,
taan 23, tand 20, tart 23, teit 20040,

tell 3023, 3020, tien 22, tijd 20, tion 23060,
toon 20, 23, trek 23, 20, trom 20, tuur 20060, 23,
uwer 20, veel 21, 23, verg 23, vers 21, 20,
verv 3023, 3020, verz 3020, vete 20,
vijf 22, vind 23, ving 23, voel 23, voer 23, 20,
voet 20, 23, volg 23, volk 20, voor 25, 20, vorm 20, 23,
vér 21, waar 2074, 24, 21, 20, wege 2020, 2023, 21034,
werk 20, 23, wond 20, 23, woon 23, zegg 3023, 3020,
zett 3023, 3020, ziek 21, zing 23, zond 23, 3020,
zorg 20, 23, West 20, aarde 20, ander 21, angst 20,
ation 23060, bedil 23, beeld 20, 23, beurs 20, 21, briev 3020,
check 20, dreig 23, drijv 3023, duits 21, 20, elijk 20191,
eling 20170, ester 20, fonds 20, genot 20, 23030,
grijp 23, ijdel 21, ijver 20, 23, ineen 24,
inter 2025, 3023, iteit 20040, komst 20, kunst 20,
lazen 1023, licht 20, 21, 23, lijks 21191, meest 21,
noter 3023, oefen 23, onder 25, orden 23, parti 3020,
plaat 20, recht 20, 21, 23, regel 23, 20, regen 20, 23, 1023,
reger 3023, schap 20130, schat 23, 20, schil 20, 23,
selen 3020, 20060, sprek 3023, staat 20, 1023, stand 20,
stede 2020, stell 3023, 3020, steun 20, 23, stort 23,
strev 3023, strom 3023, 3020, stuur 23, 20, tegen 25, 1023,
trekk 3023, 3020, vloed 20, vlooi 23, waard 21, 20,
wapen 20, 23, wezen 20, wicht 20, zetel 20, 23,
zeven 22, 1023, 20, zwaar 21, aangev 3023, bedenk 23,

bedill 3023, belang 20, beleid 20, bereid 21, 23,
beroep 20, 23, beslis 23, binnen 25, buiten 25, 20, bureau 20,
consul 20, cultur 3020, dracht 20, gelijk 21, 20, 23,
geloof 23, 20, gemeen 21, gering 21, gevoel 20, 23, gezond 21,
gracht 20, grepen 1023, 20, intern 21, kanton 20,
kiezel 20, kinder 2020, 23, klacht 20, kracht 20,
kundig 21, lieden 20, mening 20, mental 3021,
minder 21, 23, missie 20, nation 3020, oorlog 20, 3023,
opvolg 23, paleis 20, partij 20, plaats 20, 23,
plicht 20, premie 20, regent 1023, 20, schatt 3023, 3020,
schijn 23, 20, schill 3020, 3023, school 20, 1023,
schuiv 3023, 3020, sering 20, status 20, strijd 20, 23,
tering 20, tussen 25, vreemd 21, waarde 20, 21,
winkel 20, 23, wonder 20,
academi 3020, bedrijv 3020, 3023, besliss 3023, besprek 3023,
bestuur 23, 20, consult 20, 3023, diploma 20,
genoten 20, 1023, gewicht 20, invloed 20, meester 20,
oordeel 20, 23, overeen 24, patient 20, produkt 20, 3023,
special 3021, verbind 23, verenig 23, verkiez 3023,
vermijd 23, vervolg 23, verzorg 23, vlooien 1023, 20,
vooruit 24,
aantrekk 3023, antwoord 20, 23, bestrijd 23, campagne 20,
diplomat 3020, doorslag 20, explosie 20, monument 20,
ondernem 3023, opdracht 20, overgang 20, provinci 3020,
resident 20, stroming 20, toestand 20, verander 23,

vergader 23, verorden 23, verschil 23, 20, welstand 20,
zeventig 22,
architect 20, belasting 20, berechtig 23, commissie 20,
fabrikant 20, maatregel 20, president 20, verschill 3023, 3020,
werkelijk 21, wezenlijk 21, Amsterdam 20, Nederland 20,
abonnement 20, bekrachtig 23, confession 3020, gouverneur 20,
specialist 20, vereniging 20, voorwaarde 20, Amsterdamm 3020,
bouwmeester 20, gemeenschap 20, verantwoord 23,
architectuur 20, tegenwoordig 21, verantwoordelijk 21,

6.4. The results

```
post    check   s
   20      20   21010
        leider

post    checks
   20    1020
        leider

postchecks
  1020


abonnement      s       voorwaarde      n
   20            22002      20           21010
leider                   leider

abonnements              voorwaarden
  2020                     1020
                         leider

abonnementsvoorwaarden
  1020


nation  al      e
  3020   23031   20011
        leider

national        e
  3021           20011
leider

nationale
   21
```

gezond	heid	s	zorg
21	20040	22002	20
	leider		

gezondheid	s	zorg
20	22002	20
leider		

gezondheids	zorg
2020	20
	leider

gezondheidszorg
20

voor	op	stell	en
25	25	3023	21013
	leider	leider	

voorop	stellen
25	1023
	leider

vooropstellen
1023

Nederland	s	e
20	20141	20011
	leider	

Nederlands	e
21	20011
leider	

Nederlandse
21

gezond heid s zorg
21 20040 22002 20
leider

gezondheid s zorg
20 22002 20
leider

gezondheids zorg
2020 20
leider

gezondheidszorg
20

waar om heen
2074 25 24
leider

waarom heen
24 24
leider

waaromheen
24

gezond heid s toestand
21 20040 22002 20
leider

gezondheid s toestand
20 22002 20
leider

gezondheids toestand
2020 20
leider

gezondheidstoestand
20

ziek en fonds wege
21 20040 20 21034
leider

zieken fonds wege
20 20 21034
leider

ziekenfonds wege
20 21034
leider

ziekenfondswege
1024

wezen verzorg ing
20 23 20060
leider

wezen verzorging
20 20
leider

wezenverzorging
20

loon explosie s
20 20 21010
leider

loon explosies
20 1020
leider

loonexplosies
1020

vijf daag s e
22 2020 20141 20011
leider

vijfdaag s e
2020 20141 20011
leider

vijfdaags e
21 20011
leider

vijfdaagse
21

in grijp end e
25 23 20061 20011
leider

ingrijp end e
23 20061 20011
leider

ingrijpend e
21 20011
leider

ingrijpende
21

land elijk e
20 20191 20011
leider

landelijk e
21 20011
leider

landelijke
21

```
huis     arts     en       vereniging
  20       20    22003        20
           leider

huis     artsen             vereniging
  20      2020                  20
           leider

huisartsen                  vereniging
  2020                          20
                              leider

huisartsenvereniging
  20


angst    aan      jag      end
  20       25     3023     20061
                 leider

angst    aanjag             end
  20      3023               20061
                            leider

angst    aanjagend
  20       21
           leider

angstaanjagend
  21


ziek     en       fonds    en
  21    20040       20    21010
           leider  leider

zieken           fondsen
  20               1020
                   leider

ziekenfondsen
  1020
```

uit oefen en
25 23 21013
leider

uit oefenen
25 1023
leider

uitoefenen
1023

gezond heid s zorg
21 20040 22002 20
leider

gezondheid s zorg
20 22002 20
leider

gezondheids zorg
2020 20
leider

gezondheidszorg
20

Nederland er s
20 20030 21010
leider

Nederlander s
20 21010
leider

Nederlanders
1020

arts en status
20 22003 20
leider

artsen status
2020 20
leider

artsenstatus
20

be	eind	ig	ing
2163	20	20141	20060
		leider	

be	eindig	ing
2163	21	20060
leider		

beeindig	ing
23	20060
	leider

beeindiging
20

om	zett	ing
25	3023	20060
	leider	

omzett	ing
3023	20060
	leider

omzetting
20

ver	minder	en	
2183	21	21013	
leider			

verminder	en
23	21013
leider	

verminderen
1023

arts	en	beroep
20	22003	20
leider		

artsen	beroep
2020	20
	leider

artsenberoep
20

ziek en fonds en
21 20040 20 21010
leider leider

zieken fondsen
20 1020
leider

ziekenfondsen
1020

vér gaa nd e
21 2023 20061 20011
leider

vér gaand e
21 21 20011
leider

vér gaande
21 21
leider

vérgaande
21

ziek en fonds en
21 20040 20 21010
leider leider

zieken fondsen
20 1020
leider

ziekenfondsen
1020

ziek en fonds en
21 20040 20 21010
leider leider

zieken fondsen
20 1020
leider

ziekenfondsen
1020

```
gezond  heid    s       zorg
    21   20040   22002      20
         leider

gezondheid      s       zorg
    20           22002      20
leider

gezondheids             zorg
  2020                      20
                          leider

gezondheidszorg
    20


bij     ver     schijn  selen
    25    2183      23   20060
          leider

bij     verschijn       selen
    25      23           20060
                        leider

bij     verschijnselen
    25      20
          leider

bijverschijnselen
    20


gezond  heid    s       toestand
    21   20040   22002      20
         leider

gezondheid      s       toestand
    20           22002      20
leider

gezondheids             toestand
  2020                      20
                          leider

gezondheidstoestand
    20
```

ziek en fonds en
21 20040 20 21010
leider leider

zieken fondsen
20 1020
leider

ziekenfondsen
1020

ziek en fonds en
21 20040 20 21010
leider leider

zieken fondsen
20 1020
leider

ziekenfondsen
1020

gezond heid s zorg
21 20040 22002 20
leider

gezondheid s zorg
20 22002 20
leider

gezondheids zorg
2020 20
leider

gezondheidszorg
20

uit eind elijk
25 20 20191
leider

uiteind elijk
20 20191
leider

uiteindelijk
21

```
nation  al      iser    ing
  3020   23031   23123   20060
        leider

national        iser    ing
  3021           23123   20060
                leider

nationaliser            ing
  3023                   20060
                        leider

nationalisering
    20


be      oefen   aar
  2163      23   20060
leider

beoefen         aar
    23           20060
                leider

beoefenaar
    20


werk    tijd    en
    20      20   21010
        leider

werk    tijden
    20    1020
        leider

werktijden
  1020


uit     oefen   ing
    25      23   20060
        leider

uitoefen        ing
    23           20060
                leider

uitoefening
    20
```

post academi aal
20 3020 20031
leider

postacademi aal
3020 20031
leider

postacademiaal
21

on vermijd elijk
2107 23 20191
leider

on vermijdelijk
2107 21
leider

onvermijdelijk
21

overgang s ver schijn selen
20 22002 2183 23 20060
leider

overgang s verschijn selen
20 22002 23 20060
leider leider

overgangs verschijnselen
2020 20
leider

overgangsverschijnselen
20

inter nation aal
2025 3020 20031
leider

internation aal
3020 20031
leider

internationaal
21

```
lang    lop     end     e
   21    3023   20061   20011
               leider

lang    lopend          e
   21      21           20011
        leider

lang    lopende
   21      21
        leider

langlopende
   21


ont     wapen   ing
  2143     20   20060
leider

ontwapen        ing
   23            20060
                leider

ontwapening
   20


mening  s       vorm    ing
   20   22002      23   20060
leider                  leider

menings         vorming
  2020              20
                leider

meningsvorming
   20


ont     wapen   ing
  2143     20   20060
leider

ontwapen        ing
   23            20060
                leider

ontwapening
   20
```

inter nation al e
2025 3020 23031 20011
leider

internation al e
3020 23031 20011
leider

international e
3021 20011
leider

internationale
21

ont wapen ing
2143 20 20060
leider

ontwapen ing
23 20060
leider

ontwapening
20

half jaar lijks
21 20 21191
leider

halfjaar lijks
20 21191
leider

halfjaarlijks
1021

loon s ver hog ing en
20 22002 2183 3021 20060 21010
leider

loon s verhog ing en
20 22002 3023 20060 21010
leider

loon s verhoging en
20 22002 20 21010
leider leider

loons verhogingen
2020 1020
leider

loonsverhogingen
1020

produkt iv iteit
3023 23061 20040
leider

produktiv iteit
3021 20040
leider

produktiviteit
20

werk gev er s
20 3023 20060 21010
leider

werk gever s
20 20 21010
leider

werk gevers
20 1020
leider

werkgevers
1020

ver plicht ing en
2183 20 20060 21010
leider

verplicht ing en
23 20060 21010
leider

verplichting en
20 21010
leider

verplichtingen
1020

meer jar ig e
21 3020 20141 20011
leider

meerjar ig e
3020 20141 20011
leider

meerjarig e
21 20011
leider

meerjarige
21

loon overeen komst en
20 24 20 21010
leider

loon overeen komsten
20 24 1020
leider

loon overeenkomsten
20 1020
leider

loonovereenkomsten
1020

```
vak     bond    en
    20      20   21010
        leider

vak     bonden
    20    1020
        leider

vakbonden
  1020


West    duits   e
    20      21   20011
        leider

West    duitse
    20      21
        leider

Westduitse
    21


af      beeld   en
    24      20   21010
        leider

af      beelden
    24    1020
        leider

afbeelden
  1020


bedenk  en
    23   21013
leider

bedenken
  1023
```

```
be      dreig   t
  2163      23   21013
leider

bedreig         t
    23           21013
leider

bedreigt
  1023


kanton  recht   er
    20      23   20060
                leider

kanton  rechter
    20      20
        leider

kantonrechter
    20


ont     wapen   t
  2143      20   21013
leider

ontwapen        t
    23           21013
leider

ontwapent
  1023


recht   s       vervolg         ing
    20   22002      23           20060
leider                          leider

rechts          vervolging
  2020              20
                leider

rechtsvervolging
    20
```

provinci al e
3020 23031 20011
leider

provincial e
3021 20011
leider

provinciale
21

ver oordeel d
2183 23 20061
leider

veroordeel d
23 20061
leider

veroordeeld
21

on verbind end
2107 23 20061
leider

on verbindend
2107 21
leider

onverbindend
21

strijd ig heid
20 20141 20040
leider

strijdig heid
21 20040
leider

strijdigheid
20

mens en recht en
20 22003 20 21010
leider leider

mensen rechten
2020 1020
leider

mensenrechten
1020

plaats elijk e
20 20191 20011
leider

plaatselijk e
21 20011
leider

plaatselijke
21

provinci al e
3020 23031 20011
leider

provincial e
3021 20011
leider

provinciale
21

verorden ing en
23 20060 21010
leider

verordening en
20 21010
leider

verordeningen
1020

```
na      vlooien
    25    1023
        leider

navlooien
  1023


mens    en      recht   en
    20   22003      20   21010
leider          leider

mensen          rechten
  2020            1020
                leider

mensenrechten
  1020


ineen   stort   ing
    24      23   20060
        leider

ineenstort      ing
    23           20060
                leider

ineenstorting
    20


bouw    ondernem        ing
    20    3023           20060
                        leider

bouw    onderneming
    20      20
        leider

bouwonderneming
    20
```

```
op      blaz    en
   25    3023   21013
         leider

op      blazen
   25    1023
         leider

opblazen
  1023


reger   ing     s       steun
  3023   20060   22002      20
         leider

regering        s       steun
    20           22002      20
leider

regerings               steun
  2020                      20
                          leider

regeringssteun
    20


gemeenschap     s       hand    en
    20           22002     20   21010
leider                  leider

gemeenschaps            handen
  2020                    1020
                          leider

gemeenschapshanden
  1020


beleid  s       dad     en
    20   22002    3020   21010
leider          leider

beleids         daden
  2020            1020
                  leider

beleidsdaden
  1020
```

```
werk    nem     er      s
    20   3023   20060   21010
                leider

werk    nemer           s
    20      20          21010
        leider

werk    nemers
    20   1020
        leider

werknemers
  1020


Amsterdam       s       e
    20           20141   20011
                leider

Amsterdams              e
    21                   20011
leider

Amsterdamse
    21


mis     leid    end
  2167      23   20061
        leider

misleid         end
    23           20061
                leider

misleidend
    21


Amsterdam       s       e
    20           20141   20011
                leider

Amsterdams              e
    21                   20011
leider

Amsterdamse
    21
```

belang en spel
20 22003 20
leider

belangen spel
2020 20
leider

belangenspel
20

Amsterdam s e
20 20141 20011
leider

Amsterdams e
21 20011
leider

Amsterdamse
21

Amsterdam s e
20 20141 20011
leider

Amsterdams e
21 20011
leider

Amsterdamse
21

binnen stad s beleid
25 20 22002 20
leider

binnen stads beleid
25 2020 20
leider

binnenstads beleid
2020 20
leider

binnenstadsbeleid
20

mis grepen
2167 1023
leider

misgrepen
1023

goed vind en
21 23 21013
leider

goed vinden
21 1023
leider

goedvinden
1023

Amsterdam s e
20 20141 20011
leider

Amsterdams e
21 20011
leider

Amsterdamse
21

ver tegenwoordig er
2183 21 20060
leider

vertegenwoordig er
23 20060
leider

vertegenwoordiger
20

```
architectuur    mede    werk    er
    20            24      23   20060
                        leider

architectuur    medewerk        er
    20            23            20060
                                leider

architectuur    medewerker
    20            20
                leider

architectuurmedewerker
    20


opdracht        gev     er      s
    20            3023  20060   21010
                        leider

opdracht        gever           s
    20            20            21010
                leider

opdracht        gevers
    20            1020
                leider

opdrachtgevers
  1020


loon    slav    en
    20    3020  21010
        leider

loon    slaven
    20    1020
        leider

loonslaven
  1020
```

opdracht gev er
20 3023 20060
leider

opdracht gever
20 20
leider

opdrachtgever
20

gracht en huiz en
20 22003 3020 21010
leider leider

grachten huizen
2020 1020
leider

grachtenhuizen
1020

regent en paleis
20 22003 20
leider

regenten paleis
2020 20
leider

regentenpaleis
20

Amsterdam s e
20 20141 20011
leider

Amsterdams e
21 20011
leider

Amsterdamse
21

```
stad    s       regent  en
   20   22002      20   21010
leider          leider

stads           regenten
  2020            1020
                leider

stadsregenten
  1020


Amsterdamm      er      s
  3020           20030   21010
                leider

Amsterdammer            s
   20                    21010
leider

Amsterdammers
  1020


koop    mans    stad
   20    2020      20
        leider

koopmans        stad
  2020             20
                leider

koopmansstad
   20


bouw    kunst   ig      e
   20      20   20141   20011
        leider

bouwkunst       ig      e
   20            20141   20011
                leider

bouwkunstig             e
   21                    20011
leider

bouwkunstige
   21
```

```
regent  en      architectuur
    20   22003      20
leider

regenten        architectuur
  2020              20
                  leider

regentenarchitectuur
    20


bouw    kundig  en
    20      21   20040
                leider

bouw    kundigen
    20      20
        leider

bouwkundigen
    20


monument        al      e
    20           23031   20011
                leider

monumental              e
  3021                   20011
leider

monumentale
    21


koop    lieden  regent  en
    20      20      20   21010
        leider  leider

kooplieden      regenten
    20            1020
                leider

koopliedenregenten
  1020
```

gracht en huiz en
20 22003 3020 21010
leider leider

grachten huizen
2020 1020
leider

grachtenhuizen
1020

cultur el e
3020 23031 20011
leider

culturel e
3021 20011
leider

culturele
21

bedill erij en
3023 20060 21010
leider

bedillerij en
20 21010
leider

bedillerijen
1020

dag elijk s e
20 20191 20141 20011
leider

dagelijk s e
21 20141 20011
leider

dagelijks e
21 20011
leider

dagelijkse
21

```
cultur  el      e
  3020   23031   20011
        leider

culturel        e
  3021           20011
leider

culturele
    21


vreemd  eling   en
    21   20170   21010
        leider

vreemdeling     en
    20           21010
leider

vreemdelingen
  1020


land    genoten
    20      20
        leider

landgenoten
    20


cultur  el      e
  3020   23031   20011
        leider

culturel        e
  3021           20011
leider

culturele
    21
```

beurs noter ing en
20 3023 20060 21010
leider

beurs notering en
20 20 21010
leider

beurs noteringen
20 1020
leider

beursnoteringen
1020

stede bouw kundig e
2020 20 21 20011
leider leider

stedebouw kundige
20 21
leider

stedebouwkundige
21

welstand s commissie s
20 22002 20 21010
leider leider

welstands commissies
2020 1020
leider

welstandscommissies
1020

Nederland s e
20 20141 20011
leider

Nederlands e
21 20011
leider

Nederlandse
21

```
stad    s       ruim    te
    20   22002      21   20040
leider                  leider

stads           ruimte
  2020              20
                  leider

stadsruimte
    20


ruil    voorwaarde      n
    20      20          21010
        leider

ruil    voorwaarden
    20    1020
        leider

ruilvoorwaarden
  1020


on      gevoel  ig
  2107      20   20141
                leider

on      gevoelig
  2107      21
        leider

ongevoelig
    21


vak     blad    en
    20      20   21010
        leider

vak     bladen
    20    1020
        leider

vakbladen
  1020
```

zes	en	zeventig
22	26	22
koppel	koppel	normaal
		leider

zesenzeventig
22

woon	ruim	te
23	21	20040
		leider

woon	ruimte
23	20
	leider

woonruimte
20

winkel	bedrijv	ig	heid
20	3020	20141	20040
	leider		

winkelbedrijv	ig	heid
3020	20141	20040
	leider	

winkelbedrijvig	heid
21	20040
	leider

winkelbedrijvigheid
20

stede	bouw	er	s
2020	23	20060	21010
		leider	

stede	bouwer	s
2020	20	21010
	leider	

stede	bouwers
2020	1020
	leider

stedebouwers
1020

```
bouw    kunst   ig      e
    20      20   20141   20011
        leider

bouwkunst       ig      e
    20           20141   20011
                leider

bouwkunstig             e
    21                   20011
leider

bouwkunstige
    21


inter   nation  al      e
  2025    3020   23031   20011
        leider

internation     al      e
  3020           23031   20011
                leider

international           e
  3021                   20011
leider

internationale
    21


winkel  huiz    en
    20    3020   21010
        leider

winkel  huizen
    20    1020
        leider

winkelhuizen
  1020
```

bestuur s besliss ing en
20 22002 3023 20060 21010
leider

bestuur s beslissing en
20 22002 20 21010
leider leider

bestuurs beslissingen
2020 1020
leider

bestuursbeslissingen
1020

stad s bouwmeester
20 22002 20
leider

stads bouwmeester
2020 20
leider

stadsbouwmeester
20

beleid s besliss ing en
20 22002 3023 20060 21010
leider

beleid s beslissing en
20 22002 20 21010
leider leider

beleids beslissingen
2020 1020
leider

beleidsbeslissingen
1020

```
ver     werkelijk       ing
  2183      21           20060
leider

verwerkelijk            ing
    23                   20060
                        leider

verwerkelijking
    20


oorlog  s       voor    bereid  ing
    20   22002      25      23   20060
                        leider

oorlog  s       voorbereid      ing
    20   22002      23           20060
leider                          leider

oorlogs         voorbereiding
  2020              20
                  leider

oorlogsvoorbereiding
    20


reger   ing     s       verantwoordelijk        heid
  3023   20060   22002      21                    20040
        leider

regering        s       verantwoordelijk        heid
    20           22002      21                    20040
leider                                          leider

regerings               verantwoordelijkheid
  2020                      20
                          leider

regeringsverantwoordelijkheid
    20
```

taak stell ing
20 3023 20060
leider

taak stelling
20 20
leider

taakstelling
20

verkiez ing en
3023 20060 21010
leider

verkiezing en
20 21010
leider

verkiezingen
1020

bekrachtig en
23 21013
leider

bekrachtigen
1023

veel zegg end e
21 3023 20061 20011
leider

veel zeggend e
21 21 20011
leider

veel zeggende
21 21
leider

veelzeggende
21

verantwoord	ing	s	plicht
23	20060	22002	20
	leider		

verantwoording	s	plicht
20	22002	20
leider		

verantwoordings	plicht
2020	20
	leider

verantwoordingsplicht
20

partij	raad	s	vergader	ing
20	20	22002	23	20060
	leider			

partij	raads	vergader	ing
20	2020	23	20060
	leider		leider

partijraads	vergadering
2020	20
	leider

partijraadsvergadering
20

Amsterdam	s	e
20	20141	20011
	leider	

Amsterdams	e
21	20011
leider	

Amsterdamse
21

houd baar heid
23 20061 20040
leider

houdbaar heid
21 20040
leider

houdbaarheid
20

confession al isme
3020 23031 20150
leider

confessional isme
3021 20150
leider

confessionalisme
20

kiezel sten en
20 3020 21010
leider

kiezel stenen
20 1020
leider

kiezelstenen
1020

regent en maatregel
20 22003 20
leider

regenten maatregel
2020 20
leider

regentenmaatregel
20

ver tegenwoordig er s
2183 21 20060 21010
leider

vertegenwoordig er s
23 20060 21010
leider

vertegenwoordiger s
20 21010
leider

vertegenwoordigers
1020

ver wezenlijk ing
2183 21 20060
leider

verwezenlijk ing
23 20060
leider

verwezenlijking
20

vak bond specialist en
20 20 20 21010
leider leider

vakbond specialisten
20 1020
leider

vakbondspecialisten
1020

besprek ing en
3023 20060 21010
leider

bespreking en
20 21010
leider

besprekingen
1020

zwaar gewicht en
21 20 21010
leider

zwaar gewichten
21 1020
leider

zwaargewichten
1020

over schatt ing
25 3023 20060
leider

overschatt ing
3023 20060
leider

overschatting
20

in voer ing
25 23 20060
leider

invoer ing
23 20060
leider

invoering
20

aantrekk ing s kracht
3023 20060 22002 20
leider

aantrekking s kracht
20 22002 20
leider

aantrekkings kracht
2020 20
leider

aantrekkingskracht
20

mening	s	verschill		en
20	22002	3020		21010
leider		leider		
menings		verschillen		
2020		1020		
		leider		
meningsverschillen				
1020				
opvolg	ing	s	vete	n
23	20060	22002	20	21010
	leider			
opvolging		s	vete	n
20		22002	20	21010
leider			leider	
opvolgings			veten	
2020			1020	
			leider	
opvolgingsveten				
1020				
reger	ing	s	leid	er
3023	20060	22002	23	20060
	leider			
regering		s	leid	er
20		22002	23	20060
leider				leider
regerings			leider	
2020			20	
			leider	
regeringsleider				
20				

vooruit 24	strev 3023 leider	end 20061	e 20011
vooruitstrev 3023		end 20061 leider	e 20011
vooruitstrevend 21 leider			e 20011
vooruitstrevende 21			
doorslag 20	gev 3023	end 20061 leider	e 20011
doorslag 20	gevend 21 leider		e 20011
doorslag 20	gevende 21 leider		
doorslaggevende 21			
verkiez 3023	ing 20060 leider	s 22002	campagne 20
verkiezing 20 leider		s 22002	campagne 20
verkiezings 2020			campagne 20 leider
verkiezingscampagne 20			

verkiez ing en
3023 20060 21010
leider

verkiezing en
20 21010
leider

verkiezingen
1020

be invloed den
2163 20 21013
leider

beinvloed den
23 21013
leider

beinvloedden
1023

president s verkiez ing en
20 22002 3023 20060 21010
leider

president s verkiezing en
20 22002 20 21010
leider leider

presidents verkiezingen
2020 1020
leider

presidentsverkiezingen
1020

verzorg	ing	s	staat
23	20060	22002	20
	leider		

verzorging	s	staat
20	22002	20
leider		

verzorgings	staat
2020	20
	leider

verzorgingsstaat
20

gelijk	berechtig	der
21	23	20061
		leider

gelijk	berechtigder
21	21
	leider

gelijkberechtigder
21

verkiez	ing	s	campagne
3023	20060	22002	20
	leider		

verkiezing	s	campagne
20	22002	20
leider		

verkiezings	campagne
2020	20
	leider

verkiezingscampagne
20

```
bestrijd        er      s
    23           20060   21010
                leider

bestrijder              s
    20                   21010
leider

bestrijders
  1020


volk    s       ver     tegenwoordig    er      s
    20   22002    2183       21           20060   21010
                leider

volk    s       vertegenwoordig         er      s
    20   22002       23                  20060   21010
                                        leider

volk    s       vertegenwoordiger               s
    20   22002       20                          21010
leider          leider

volks           vertegenwoordigers
  2020            1020
                leider

volksvertegenwoordigers
  1020


president       s       verkiez         ing     en
    20           22002    3023          20060   21010
                                        leider

president       s       verkiezing              en
    20           22002      20                   21010
leider                  leider

presidents              verkiezingen
  2020                    1020
                        leider

presidentsverkiezingen
  1020
```

verkiez ing en
3023 20060 21010
leider

verkiezing en
20 21010
leider

verkiezingen
1020

gouverneur s zetel s
20 22002 20 21010
leider leider

gouverneurs zetels
2020 1020
leider

gouverneurszetels
1020

ver schuiv ing en
2183 3023 20060 21010
leider

verschuiv ing en
3023 20060 21010
leider

verschuiving en
20 21010
leider

verschuivingen
1020

verkiez ing en
3023 20060 21010
leider

verkiezing en
20 21010
leider

verkiezingen
1020

ver schuiv ing en
2183 3023 20060 21010
leider

verschuiv ing en
3023 20060 21010
leider

verschuiving en
20 21010
leider

verschuivingen
1020

tussen verkiez ing
25 3023 20060
leider

tussenverkiez ing
3023 20060
leider

tussenverkiezing
20

toon aangev end e
20 3023 20061 20011
leider

toon aangevend e
20 21 20011
leider

toon aangevende
20 21
leider

toonaangevende
21

verander ing en
23 20060 21010
leider

verandering en
20 21010
leider

veranderingen
1020

belasting ver hog ing en
20 2183 3021 20060 21010
leider

belasting verhog ing en
20 3023 20060 21010
leider

belasting verhoging en
20 20 21010
leider

belasting verhogingen
20 1020
leider

belastingverhogingen
1020

ver hog ing
2183 3021 20060
leider

verhog ing
3023 20060
leider

verhoging
20

consult atie bureau
3023 20060 20
leider

consultatie bureau
20 20
leider

consultatiebureau
20

buiten echt elijk
25 20 20191
leider

buitenecht elijk
20 20191
leider

buitenechtelijk
21

on vermijd elijk
2107 23 20191
leider

on vermijdelijk
2107 21
leider

onvermijdelijk
21

verkiez ing en
3023 20060 21010
leider

verkiezing en
20 21010
leider

verkiezingen
1020

verkiez ing en
3023 20060 21010
leider

verkiezing en
20 21010
leider

verkiezingen
1020

geloof s briev en
20 22002 3020 21010
leider leider

geloofs brieven
2020 1020
leider

geloofsbrieven
1020

melk bedrijv en
20 3020 21010
leider

melk bedrijven
20 1020
leider

melkbedrijven
1020

voet klacht en
20 20 21010
leider

voet klachten
20 1020
leider

voetklachten
1020

```
sokk    en      fabrikant       en
  3020   22003      20           21010
leider          leider

sokken          fabrikanten
  2020            1020
                  leider

sokkenfabrikanten
  1020


voet    arts    en
    20      20   21010
        leider

voet    artsen
    20    1020
        leider

voetartsen
  1020


partij  stroming        en
    20      20          21010
        leider

partij  stromingen
    20    1020
        leider

partijstromingen
  1020


school  gaa     nd      e
    20    2023  20061   20011
                leider

school  gaand           e
    20      21          20011
        leider

school  gaande
    20      21
        leider

schoolgaande
    21
```

diplomat en kinder en
3020 22003 2020 21010
leider leider

diplomaten kinderen
2020 1020
leider

diplomatenkinderen
1020

ziek en fonds patient
21 20040 20 20
leider

zieken fonds patient
20 20 20
leider

ziekenfonds patient
20 20
leider

ziekenfondspatient
20

ziek en fonds premie
21 20040 20 20
leider

zieken fonds premie
20 20 20
leider

ziekenfonds premie
20 20
leider

ziekenfondspremie
20

LITERATURE LIST

1) Balk, Mrs. F. - Smit Duyzentkunst, De grammatische functie, (The grammatical function), dissertation, Groningen 1963.

2) Bierwisch, M., Grammatik des deutschen Verbes (grammar of the German verb), East Berlin 1963.

3) Bloomfield, L., Language, New York 1933.

4) Boutens, P.C., Carmina, Amsterdam 1924.

5) Van Dale, Groot Woordenboek der Nederlandse Taal (Big Lexicon of the Dutch Language), 's-Gravenhage 1960.

6) Formal properties of newspaper dutch, Mathematical Centre, Amsterdam 1965.

7) Geyl, P., Geschiedenis als medespeler (History as a playfellow), Utrecht/Antwerpen 1958.

8) Mannoury, G., Handboek der analytische significa (Manual of analytical significs), I, Bussum 1947.

9) Nida, E.A., Morphology, Ann Arbor 1949.

10) Reifler, E., Mechanical determination of the constituents of german substantive compounds, Mechanical Translation, Massachusetts 1955.

11) Tinbergen, D.C., Nederlandse Spraakkunst (Dutch Grammar), 13-th edition, Zwolle 1963.

12) De Vries, Th., Februari, Amsterdam 1962.

13) Woordenboek van de Nederlandse Taal (Lexicon of the Dutch Language).

14) Woordenlijst van de Nederlandse Taal (Word-list of the Dutch Language), 2-nd edition, 's-Gravenhage 1956.

INDEX OF TECHNICAL TERMS AND SYMBOLS

This list contains only the technical terms introduced or deviantly defined by me, with the page of definition behind it to facilitate the re-finding of the definition for the reader, together with some identifiers in the analytic program, used in a more or less constant meaning. These latter ones are followed by declarations between brackets.