CLOC mark 2Q

A Collocations and Concordance Package
by
Alan Reed*

February 7, 2016

Abstract

This guide describes a computer program which will help you analyse natural language text. The program is named CLOC and takes the form of a package of facilities designed for ease of use by people with little or no computer experience. The original development work has been done at Birmingham University Computer Centre in collaboration with Professor J. McH. Sinclair of the Department of English Language and Literature. The package has been improved from the original and currently contains facilities for producing sorted vocabulary lists, word indexes, concordances, and the automatic discovery of collocations. The name CLOC is an acronym taken from the term 'ColLOCation'.

Acknowledgements

The author would like to offer his thanks to friends and colleagues whose advice and criticism have aided the writing of this guide and the creation of the package. To Professor J. McH. Sinclair of the Department of English for requesting the program and suggesting several of the features; to Dr. J. L. Schonfelder for his enthusiasm and advice; and to Professors Greaves (York, Toronto), Benson (York,Toronto) and Brainerd (Toronto) whose interest and support was most welcome.

*A.REED@TALK21.COM

Contents

1 INTRODUCTION

CLOC is a package which will enable a novice computer user to analyse natural language text by computer. This guide explains what the package can do, and shows you how to instruct it to carry out various tasks.

The package can examine the vocabulary used by an author and be told to print it in several ways. The vocabulary, or a selected portion of it, can be printed in order, as in a dictionary, or according to how frequently a word is used.

The primary purpose of the CLOC package is to produce collocations of selected words. These are frequently occurring patterns of words which appear regularly within a text. The package is capable of discovering these patterns and will print the context of each occurrence in a style of your choice.

CLOC can also produce a concordance of words selected from the text, to show how an author uses the selected words. You can choose the amount of context that is printed as well as the style of concordance.

You guide and control the actions of the package by employing a few simple commands, which are supplied to the package by way of statements in a command language. The following sections explain how you tell the package what analyses you want it to do.

2 PREPARATION OF TEXT

2.1 THE CHARACTER SET

Before you can use the CLOC package to analyse some text it must be converted from the printed page or spoken word into a form that a computer can read. A printed page could contain many differing alphabets, use several type styles, and allow a large number of special symbols. As CLOC can only deal with (say)95 different characters, you must devise a consistent scheme to convert every letter, punctuation mark, diacritic and significant change of type style into one or more characters in this small and restricted alphabet.

This can be achieved by dividing the set of possible characters into several mutually exclusive categories. Use one category of characters to compose words, use another to separate one word from the next, and so on. For English, the first category could include the alphabet `A` to `Z`, while the second could include `? , ; .` , and 'space'. In this document it will be assumed that text is prepared using the 95 printable characters of the ASCII alphabet. This includes the upper and lower case letters together with a large variety of other symbols. For your information the ASCII alphabet is reproduced below:-

```
!"#$%&'()*+,-./0123456789:;<=>?@
ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_‘
abcdefghijklmnopqrstuvwxyz{|}~
```

These will be found on most computer keyboards but the graphic symbols printed on the keyboard may not look quite the same as the above. CLOC commands can be written using uppercase or lowercase letters or a mixture of the two. All the examples will show CLOC commands in uppercase letters for clarity. The CLOC package lets you choose which characters are 'letters' and which are not. Normally you would choose abcdefghijklmnopqrstuvwxyz as your alphabet. Additional 'letters' called 'padding' and 'deferred' can be defined to cope with accents, apostrophes, breathing marks, etc. If you are unlucky enough to prepare your text on punched cards you could represent the phrase 'Once upon a time' as follows:-

```
=ONCE␣UPON␣A␣TIME
```

Notice that before every capital letter you should place some symbol (say =) to indicate that a capital letter comes next. This can also be used when a capital letter occurs inside a word as in 'MacDonald'. This you could represent as :

```
=MAC=DONALD
```

The CLOC package could be told that = is a special kind of letter so it could read '=MAC=DONALD' as one word rather than the two words 'MAC' and 'DONALD'.

2.2 CAPITAL AND SMALL LETTERS

The CLOC package is designed to work with text containing a mixture of upper and lower case letters. Thus if you ask for a concordance of (say) 'the' you will get all the places where 'the' occurs even when 'The' is given in the text. Usually the case of the letters is not relevant, but if required you can tell CLOC that the case of letters is significant, in which case 'the' and 'The' will be counted as different words. (see the `ITEMIZE USING` command for details).

2.3 DIACRITICAL MARKS

Marks placed above or below letters to indicate stress can be represented by two characters, one for the letter and one for the particular mark. For example you could represent the following:

cliché as `cliche1` (or on punched cards as `CLICH1E`)

where the symbol 1 is used to represent the acute accent. This can be combined with capitalization as follows:

Écosse as `1Ecosse` (or on punched cards as `1=ECOSSE`)

Note that each diacritical mark must be considered as either a 'padding' or a 'deferred' 'letter'.

2.4 FOREIGN LANGUAGES

When the language of the text is not in the Roman alphabet, you must transliterate letters in the language to characters in the computer's alphabet. Using Greek as an example and a systematic conversion changing α to `A`, β to `B` and γ to `G` etc. we could write:-

$\pi o \lambda \epsilon \mu \iota \kappa o \sigma$ as `polemikos`

where each Greek letter is replaced by a Roman letter. When the natural language being used contains words from another language they should be carefully distinguished. This can easily be achieved by prefixing each rarely occurring foreign word with a special character. Thus when using English containing French words they could be distinguished by a `$` symbol. For example:-

'In French, lard means bacon' could be coded as:

```
In␣French,␣$lard␣means␣bacon
```

or on punched cards as

```
=IN␣=FRENCH,␣$LARD␣MEANS␣BACON
```

It is expensive and time-consuming to convert a large amount of text into computer readable form. It is well worth finding out from colleagues or published catalogues if your chosen text has already been prepared for computer use. If you have to do the job yourself do make sure you have coded all significant features in the text using a consistent and preferable 'standard' method. It is well worth doing a pilot study using a few hundred lines of text to get a feel for what the coding scheme should cover. Finally do not forget to obtain copyright clearance for storing the text on a computer.

2.5 TEXT REFERENCES

Text prepared for a computer is likely to contain information which is extra-textual such as the title, author, chapter, page number, and so on. CLOC can be told that a text has no references whatever, it which case to treats the text as a series of 'words' separated from each other by 'spaces' , 'fullstops', etc. When a concordance or word index is requested CLOC will use the line number to act as a reference for citations that are printed. It is much better though to pre-code text references into the text to begin with using some well known and simple convention such as that used in the COCOA or OCP packages.

2.5.1 Coding references

This feature allows you to include in your text data the name of the author, the section, chapter, and line number, etc. in a manner similar to the well known COCOA and OCP packages. Your text data could include say `<A␣DICKENS><P␣1><L␣1>` which to you would signify author Dickens, Page 1, Line 1. Subsequent lines would contain the text data for this section. For the next page it is sufficient for you to type `<P␣2><L␣1>` as the author has not changed. As the package reads each line of text the line number is increased by 1, so the line number should be reset to 1 when you start a new page. Note that `<L␣1>` refers to the *next line*, hence no text data should occur after it on the same line. Apart from this restriction, text references may be placed anywhere in your text data. It is, however, recommended that text references be placed on lines separate from the text data itself. Blank lines and lines containing text references only are ignored. This feature is only invoked when you supply an `INPUT DETAILS` command with the `REFS` option in the specification field (see section 5). Suppose you had supplied `REFS<>` this would tell the package that your text references were enclosed by the the two characters `<` and `>`.

The general form of a text reference is: a*letter gap reference*b
where:-

letter is `A` or `B` or `C` ... `Z`

reference is any sequence of characters

gap denotes one or more spaces

and ab are the two characters specified on the `REFS`ab part of the `INPUT DETAILS` command. (See section 5)

Note:-

- When L is used as a *letter*, the reference must be a number which will normally be 1.
- An incorrect text reference will be ignored - you can therefore include a title of the form: a *title*b which the package will ignore.

2.5.2 Printing references

Each line of a printed concordance can be prefixed with a detailed text reference. The `CONCORDANCE`, `COLLOCATIONS`, `CO-OCCURRENCE`, `INDEX` and `WRITETEXT` commands can contain the keyword `REFS` followed by *letter number* pairs, e.g. `REFS A4P2L3`. This example will cause the first 4 characters of the current `A` *reference*, the first 2 of the `P` *reference*, and a 3 figure line number to be placed before every printed citation. Each entry will be separated from the next by one space. The general rule is that for every *letter number* pair occurring after the `REFS` keyword the first *number* characters of the current value of the *letter* references are printed.

A minor variation occurs when a line number of say `L2` is asked for, and the given citation occurs at line 100 or greater. On the first occasion that this happens the package increases the request by 1, and subsequent references are now considered to have been requested by `L3`.

Note that in the first instance all reference letters are considered to contain 'spaces': thus a request of say `X4` will cause 4 spaces (+1 space) to be printed.

2.5.3 Example

The following verse is to be coded for storage on a computer:-

> *The Small Celandine*
> There is a Flower, the Lesser Celandine
> That Shrinks, like many more, from cold and rain;
> And, the first moment that the sun may shine,
> Bright as the sun itself, 'tis out again!
>
> When hailstones have been falling, swarm on swarm,
> Or blasts the green field and the trees distress'd,
> Oft have I seen it muffled up from harm,
> In close self-shelter, like a Thing at rest.

On a computer this could be coded as:-

```
<␣␣The␣Small␣Celandine>
<A␣Wordsworth><V␣1><L␣1>
```

```
There␣is␣a␣Flower,␣the␣Lesser␣Celandine
That␣Shrinks,␣like␣many␣more,␣from␣cold␣and␣rain;
And,␣the␣first␣moment␣that␣the␣sun␣may␣shine,
Bright␣as␣the␣sun␣itself,␣'tis␣out␣again!
<V␣2><L␣1>
When␣hailstones␣have␣been␣falling,␣swarm␣on␣swarm,
Or␣blasts␣the␣green␣field␣and␣the␣trees␣distress'd,
Oft␣have␣I␣seen␣it␣muffled␣up␣from␣harm,
In␣close␣self-shelter,␣like␣a␣Thing␣at␣rest.
```

3 OVERVIEW OF THE CLOC PACKAGE

You must supply instructions to the package to tell it how to read and analyse the text. These instructions are prepared according to rules described in section 4, and must be supplied to the package in the following order:

1. (Optional) `INPUT DETAILS` command
2. Word definition commands
3. Word selection commands
4. Task selection commands, e.g. `WORDLIST`, `INDEX`, `CONCORDANCE`, `CO-OCCURRENCE`, `COLLOCATIONS` and `WRITETEXT` commands
5. `FINISH` command

The precise order in which the individual commands should be presented is described in the Appendix. The layout and effect of each command will be described later.

The `INPUT DETAILS` command can be used to inform the package of any special characteristics in the way the text has been prepared, such as what kind of text references are present.

Word definition commands inform the package how to interpret the text data which is about to be read. They define the constitution of a *word* in terms of an alphabet of *characters*. The CLOC package assumes that a file of text is composed of a series of distinguishable words, clearly separated from each other by spaces, full stops, etc.

Words selection commands are used to instruct the package to choose a suitable collection of words, termed 'nodes', which will be used by the task commands during the analysis of the text.

The `WORDLIST` command causes the package to list the selected words sorted into a chosen order.

The `INDEX` command will produce for each word a list of references to it's position in the original text.

The `CONCORDANCE` command is used to command the package to print a concordance of the selected words. The style chosen for printing the concordance can be chosen from a menu of styles.

The `CO-OCCURRENCE` command will produce a concordance of given phrases or series of words.

The `COLLOCATIONS` command is used to command the package to search the context of the selected words, and to print the context of those which possess frequently occurring neighbours.

The `WRITETEXT` command will cause the original text to be printed out, but each line will start with a text reference.

The following example program illustrates how CLOC commands could appear in a typical task.

```
                    ITEMIZE USING  CLOC
Word definition     *LETTERS       abcdefghijklmnopqrstuvwxyz
                    SELECT WORDS
Word selection      *FREQUENCY     (100 TO 500)
                    EXCLUDING
                    *LIST OF WORDS i the but
Concordance command CONCORDANCE    KWIC, CITE 6 BY 6
FINISH command      FINISH
```

This example causes CLOC to do the following things:

- Read text composed of a series of words in the English alphabet

- Select a collection of words each of which has a frequency of occurrence lying in the range 100 to 500 inclusive, with the exception of the words 'i' 'the' and 'but'
- Produce a Key Word In Context concordance of the chosen words, where each word is surrounded by 12 words of context.

4 THE COMMAND LANGUAGE

The package needs to be told what actions to perform and in which order they should be done. It is the function of the 'command language' to supply this information in a clear and unambiguous way. Each of the following sections contains detailed information on how to instruct the package to carry out certain actions. The sections are described in the order in which they should be presented to the package, when they are phrased in terms of the 'command language'.

The package is controlled by a series of 'control statements' each of which contains a 'command'. These commands are obeyed by CLOC in the order in which they are given. Certain commands are optional and can be included when you require more facilities than those provided by default.

4.1 The control statement conventions

CLOC commands must be prepared in a standard format. A control statement is notionally divided into two distinct 'fields', each of which occupies a certain number of columns on a line. The fields for CLOC control statements are as follows:-

- The *control field* occupying columns 1 to 15 inclusive. Its function is to inform the package of an action to be performed, or to present the package with further information.
- The *specification field* occupying columns 16 to 80 inclusive. This portion of the line supplies extra information to the instruction in the control field.

An example of a command is

```
WORDLIST        ALPHA

control field   specification field
```

This command instructs the package to sort the words into alphabetic order, and to print them.

The command WORDLIST must be placed in columns 1 to 15 of the line. The specification field, columns 16 to 80, of this command contains the sorting criterion ALPHA, meaning 'alphabetic order'.

4.2 Rules for Control Statements

1. Each control field must be written within columns 1 to 15.
2. The commands must be spelt correctly. (Thus CONCORDENCE is not a valid command!)
3. When columns 1 to 15 contain spaces only, the specification field of the line is treated as a continuation of the specification field of the previous command.
4. Keywords in the specification field must not contain spaces,nor be split when continuing them onto the next line.
5. Some commands contain a star symbol (*) in column 1. This indicates that the command is subsidiary to the last unstarred command. The starred commands supply extra information to that supplied by the unstarred command upon which they are dependent. The star symbol is there for your convenience to remind you of the function of these commands, the symbol itself is optional.
6. The right parenthesis) can be used to terminate a control field. The specification field is then deemed to start immediately following it. Hence, you can write `WORDLIST)ALPHA`. This feature is intended for use when CLOC control statements are typed at a terminal rather than punched on cards. Note that you can use a ')' in column 1 to stand for the 15 space continuation field described above in part 3 above.

7. All CLOC commands and keywords can be written in upper or lower case (or a mixture of the two). The examples in this guide are written in upper case for clarity.

8. The pseudo CLOC commands -INSERT -SEND -NOSEND do not take any continuation lines. They can therefore be placed anywhere in a sequence of CLOC commands.

4.3 The -INSERT feature

This allows you to include a file of prewritten CLOC commands. You could, for example, have a set of files each containing a different sequence of word definition commands. Alternatively you could have files containing specific lists of words which you could -INSERT when required.

4.3.1 Examples

```
-INSERT        WORDDEF1
SELECT WORDS
*FREQUENCY     (1 TO 100)
EXCLUDING
-INSERT        VERBS
```

4.3.2 General form

-INSERT *filename*

The contents of *filename* are used as if they appeared in place of the -INSERT command.

4.3.3 Points to note

- There are no continuation lines for this (pseudo) CLOC command. This allows *filename* to contain lists of words on continuation lines only. For example, if a file A contains:

```
is are was were be
up down in out
```

  we could write

```
*LIST OF WORDS
-INSERT        A
               EXTRA-WORDS-HERE
```

  which would be interpreted as if you had written

```
*LIST OF WORDS
               is are was were be
               up down in out
               EXTRA-WORDS-HERE
```

- The syntax of *filename* depends on the computer that CLOC is implemented on.

- Lines taken from an -INSERT file will be copied onto the CLOC information and diagnostic file. An indication will be given as to where the lines came from.

- The contents of an -INSERT file may include other -INSERT commands. The depth of nesting is implementation dependent.

4.4 The -SEND feature

This (pseudo) CLOC command will cause subsequent CLOC commands to be sent to the CLOC diagnostic and information file.

4.4.1 Example and General form

-SEND

4.5 The -NOSEND feature

This (pseudo) CLOC command will stop the normal process of sending CLOC commands to the CLOC diagnostic and information file.

4.5.1 Example and General form

-NOSEND

5 The INPUT DETAILS command (optional)

This command allows you to specify the maximum width of lines of text data; to indicate the presence of text references; to determine which parts to skip; to define an explicit newline symbol; to include ignorable comments; and to specify rules about line continuation.

When this command is absent INPUT DETAILS WIDTH80 is assumed.

5.1 Examples

```
INPUT DETAILS       WIDTH72
INPUT DETAILS       NEWLINE/,REFS<>
INPUT DETAILS       WIDTH128,NEWLINE/,CONTINUE+
INPUT DETAILS       COMMENT(),NEWLINE/,CONTINUE+,REFS<>,ENDHYPHEN-
```

5.2 General Form

```
INPUTDETAILS   WIDTHnumber,SKIPab,COMMENTab,
               NEWLINEa,CONTINUEa,RUNOVER,REFSab,ENDHYPHENa
```

5.3 Default value

INPUT DETAILS WIDTH80

5.4 Parameters

- WIDTHnumber default value WIDTH80
  At most *number* characters will be read for each line of text data. Trailing spaces will be removed by the package. All characters which occur after column *number* will be ignored.

- SKIPab
  When present this instruction causes the package to ignore all characters between a and b inclusive. This option withdraws characters a and b from the available character set.

- COMMENTab
  Words which occur between the *pair of* characters aa and the *pair of* characters bb will not appear in the word count tables, but they will appear in the context when a citation is printed. This option withdraws the characters a and b from the available character set.

- NEWLINEa
  When present the character a represents a logical newline. This option allows more than one 'line of text' to be placed on the same line. Note that a will also be inserted automatically at the end of each line. This option withdraws character a from the available character set.

- CONTINUEa
  When CONTINUEa is present and a is found in the text, all characters remaining on the line are ignored. The next line is considered to replace the ignored part, and to be on the same line.

- RUNOVER
  When RUNOVER is present, and the text reading position is at the end-of-line (i.e. at the WIDTH number position), the end-of-line will *not terminate* a word. Hence the full width of line can be used to store text, and words can run over onto the next line.

- REFSab
  When present the package extracts text references of the form a*letter reference*b from the text file. This option withdraws characters a and b from the available character set.

- ENDHYPHENa
  Note that a is normally - . When present this feature affects the reading of text which splits words via a hyphen (-) at the end of a line. The character a (-) is removed from the word and it's two parts fused.

6 WORD DEFINITION COMMANDS

The CLOC package has been designed to read text coded according to many differing conventions. No matter how a text has been coded, the package interprets it as an arbitrary series of *words*. The composition of words is left up to you, but CLOC needs to know what rules are used for constructing words. These rules embody a strategy for extracting words from the characters in the text data. The process of combining characters in this way is called *itemization* and one must first select which itemizing strategy the package is to use; the rules of the strategy are supplied by subsidiary (starred) commands.

6.1 The ITEMIZE USING Command

This command is used to select a strategy for itemizing the text. Note that the -ISE form can also be used.

6.1.1 Example

ITEMIZE USING CLOC

6.1.2 General form

ITEMIZE USING *strategy name*

6.1.3 Default

When *strategy name* is absent CLOC is assumed.

Several possibilities are available at present. They are:-

- CLOC
  This ensures that words which differ only by the case of their letters, and/or contain *PADDING letters (q.v.) are counted as the same word

- CLOC UNCHANGED
  When you choose this strategy words will always be distinguished by the case of their letters and the presence of *PADDING letters.

For example, consider the sentence:-
The MacDonald Hotel is different from the Mac'donald Motel.
Assuming that the apostrophe ' has been designated a padding letter, then when the CLOC itemizing strategy is in use the word 'the' is deemed to occur twice, as does the word 'macdonald'. When a CONCORDANCE or COLLOCATIONS task (etc) is run, they too will treat the various forms as if they were the same word. The citations will of course look like the original text. The effect of the CLOC itemizing strategy is that :-

The	is mapped to	'the'
the	is mapped to	'the'
MacDonald	is mapped to	'macdonald'
Mac'donald	is mapped to	'macdonald'
Hotel	is mapped to	'hotel'
Motel	is mapped to	'motel'
is	is mapped to	'is'
different	is mapped to	'different'
from	is mapped to	'from'

When CLOC UNCHANGED is used all the above words are considered distinct.

The ITEMIZE USING CLOC command has a number of subsidiary commands. These commands tell the package how to interpret the characters it finds on the lines containing the text data.

6.2 The *LETTERS Command

This command is mandatory and must be the first command which follows the ITEMIZE USING CLOC command. This informs the package of the alphabet of characters out of which words are composed. A *word* is defined to be one or more consecutive letters. Every character which could form part of a word must be specified here. This includes characters used for accents, apostrophes, hyphenation, changes of type style etc.

6.2.1 Example

*LETTERS abcdefghijklmnopqrstuvwxzy

6.2.2 General form

*LETTERS *letter characters*

The *order* in which the *letter characters* appear in the command is significant. This order determines the way in which words will be alphabetically sorted. In the above example, those words beginning with 'a' will precede those starting with 'b', and so on. Thus 'alan' will sort before 'fred' which itself precedes 'freda'. Note that this command automatically caters for upper and lower case letters.

6.3 The *PADDING Command

This command is optional and when present informs the package of those *letter characters* which are to be ignored when words are placed in the vocabulary table. Usually this command will contain those *letter characters* used as apostrophes or hyphenation, but any characters specified on the above *LETTERS command could also be used.

When the CLOC itemizing strategy is chosen the *PADDING letters cannot appear in the vocabulary print-outs, because they are absent from the vocabulary table.

Note that if you were to choose an itemizing strategy (e.g. CLOC UNCHANGED) which allowed padding letters to appear in the vocabulary table, they would be ignored when sorting took place.

6.3.1 Example

*PADDING ' -
Words containing the apostrophe and/or the hyphen will have them removed before the word is stored in the vocabulary table.

6.3.2 General form

*PADDING *letter characters*
Where every character declared must be a *letter character* declared on the *LETTERS command. This is a deliberate design decision to emphasize that CLOC defines a word to be a sequence of *letter*s.

6.4 The *DEFERRED Command

This command is optional and when present informs the package of those *letter characters* which are to be ignored when words are sorted alphabetically. Usually this command will contain those *letter characters* used for accents and changes of type style, but any characters specified on the above *LETTERS command could also be used. Words which contain *DEFERRED letters will be counted separately.

6.4.1 Examples

- *DEFERRED -
  Hyphenated words will be separately indexed.
- *DEFERRED aeiou
  Words will be sorted alphabetically ignoring vowels.

6.4.2 General form

*DEFERRED *letter characters*
Where every character declared must be a *letter character* declared on the *LETTERS command. This is a deliberate design decision to emphasize that CLOC defines a word to be a sequence of *letter*s, and that the deferred feature only affects the sorting order.

This command ensures that words which differ only in (say) diacritical marks are adjacent in an alphabetically ordered dictionary. Words will always be distinguished by their *DEFERRED letters, each will have a separate entry in the vocabulary table. Note that whenever two words differ *only* in deferred letters, their sorting order is determined by the order of the deferred letters on the *LETTERS command.

6.5 The *SEPARATORS Command

This command is optional and when present informs the package of those characters which separate one word from the next. When this command is absent every character that is not declared by the *LETTERS command is automatically assumed to be a separator. The symbols one would use to separate one word from the next might be the fullstop, comma, semicolon, etc. The CLOC package always takes a 'space' to be a separator.

6.5.1 Example

*SEPARATORS ? ! ; .

6.5.2 General form

*SEPARATORS *separator characters*

The order in which *separator characters* appear on this command is of no significance. Note that a character must (and cannot) be declared both as a *letter character* and as a *separator character* at one and the same time. Those characters which are neither *letter characters* nor *separator characters* will be assumed to signify 'spaces' and will be interpreted as if they were declared on the following control statement.

6.6 The *READ AS SPACE Command

This command is optional and when present informs the package of those characters which signify a space. These characters although present in the text data will be assumed to stand for the space character and will be printed as such when concordances and collocations are produced.

6.6.1 Example

*READ AS SPACE

6.6.2 General form

*READ AS SPACE *space characters*

The order in which the *space characters* appear on this command is of no significance. This command can be used to remove punctuation marks from a text or to cause one word to be read as several. For example, if the text contained N'EST%PAS, the package would read it as two words N'EST and PAS, and would print it as N'EST PAS. If the % sign were declared as a (padding) *letter character* instead of on the *READ AS SPACE command, N'EST%PAS would be read as single *word*, and printed as N'EST%PAS.

6.7 The *IGNORE Command

This command is optional and when present informs the package of those characters which are to be *totally ignored* when the text is read.

6.7.1 Example

*IGNORE @ /

6.7.2 General form

*IGNORE *ignore characters*

The order in which the *ignore characters* appear on this command is of no significance. This command can be used to ignore characters which were placed in the text for special purposes. As an example one could cause 'house-wife' to be read as if it were 'housewife' by declaring '-' as an *ignore character*.

7 SAVING TEXT FILES

7.1 The Itemization Process

The package treats the text as a series of *words* separated from each other by *separators*. Thus:

text: *separator word separator ... word separator*

The composition of *words* and *separators*, and the method of extracting them from a text, are chosen by the ITEMIZE USING command and its subsidiary commands. Every time a CLOC job is run the text will be read *word* by *word*, and carefully saved in a special form which permits rapid production of concordances and collocations. Whenever the same text is to be examined several times it is clearly desirable to use this special form and save computer time by not reading the same text over and over again. They cause text to be stored in, and returned from, the computer's filing system. Their function is to bypass the text reading stage and so allow computer time to be saved when many analyses are performed on one file of text. Further information on the filing system can be obtained from your local computer centre.

7.2 The SAVE TEXT command

The SAVE TEXT command causes itemized text to be placed in a permanent file, named *filename*.

7.2.1 General form

SAVE TEXT *filename*

7.3 The GET TEXT command

The GET TEXT command causes itemized text to be retrieved from a permanent file, named *filename*, previously created by the SAVE TEXT command.

7.3.1 General form

GET TEXT *filename*

7.4 Examples

On one run of the package the following commands are sufficient to read the text and store it in the special form.

```
ITEMIZE USING   CLOC
*LETTERS        abcdefghijklmnopqrstuvwxyz
SAVE TEXT       MYFILE
FINISH
```

Once the file of text has been saved, the following jobs could be run in which the GET TEXT command replaces the itemizing instructions in the previous example.

```
GET TEXT        MYFILE
EVERY WORD
WORDLIST        ALPHA
FINISH
```

and on some other run one could write:

```
GET TEXT        MYFILE
SELECT WORDS
*PATTERN        *.ing
CONCORDANCE     KWIC, CITE 4 BY 4
FINISH
```

8 The OUTPUT DETAILS Command (optional)

This command allows you to choose the maximum line width for wordlists and citations to suit your particular lineprinter or terminal device.

When this command is absent OUTPUT DETAILS WIDTH 120 is assumed.

8.1 Example

OUTPUT DETAILS WIDTH80

8.2 General form

OUTPUT DETAILS WIDTH*number*
Where no more than *number* character positions will be reserved on the output device. All word lists and citations will be packed into a line of this width.

9 WORD SELECTION COMMANDS

This section describes how one can select, from the vocabulary of the text, a collection of words for analysis. This collection is used by subsequent commands when performing sorting, and producing concordances and collocations. One can choose the entire vocabulary or select a portion of it. An exclusion facility is provided which operates on the complete vocabulary or on the portion selected.

9.1 Set Descriptions

These commands allow you to specify portions of the vocabulary used in your text. These commands all have a star symbol(*) in column 1, showing they are subsidiary to the previous unstarred command. Several ways of choosing words are provided.

- By frequency of occurrence – using the `*FREQUENCY` command
- By an explicit list – using the `*LIST OF WORDS` command
- By a pattern – using the `*PATTERN` command

9.2 The *FREQUENCY command

This command is used to choose a set of words each member of which has a particular frequency of occurrence or lies in a given frequency range.

9.2.1 Examples

- `*FREQUENCY (100 TO 500)`
  This command will select only those words which occur between 100 and 500 times inclusive.
- `*FREQUENCY 1 OR 4 OR >50`
  This command will select words which occur exactly once, exactly four times, or more than 50 times.

9.2.2 General form

*FREQUENCY *expression*

where *expression* is one or more *term*s connected by OR symbols. And a *term* is one of the following:

- *integer* for example 10
  only words occurring exactly *integer* times will be selected.
- >*integer* for example >10
  only words occurring more than *integer* times will be selected.
- <*integer* for example <10
  only words occurring less than *integer* times will be selected.
- (*integer1* TO *integer2*) for example (100 TO 500)
  only words lying in the range *integer1* to *integer2* inclusive will be selected. Note that *integer1* must be smaller than *integer2*.

9.3 The *LIST OF WORDS command

This command allows one to specify a set of words of interest by supplying them explicitly. Note that when the CLOC itemizing strategy is in use, each item in the explicit list will be mapped to a 'word'. Thus you do not need to supply the exact case of the letters nor include padding letters.

9.3.1 Example

*LIST OF WORDS this that me you

9.3.2 General form

*LIST OF WORDS *list*

where *list* is one or more words separated from each other by one or more spaces.

9.4 The *PATTERN command

This command specifies a skeletal form of a word, and causes the package to select only those words which match the specified pattern. Two reserved characters are used within a pattern:-

- A dummy-symbol which is .
  The dummy-symbol stands for *any* letter.
- A variable-symbol which is *
  The variable-symbol stands for '*any sequence* of letters, including none at all'.

These reserved characters can be used in combination with the *letter characters* defined by the word definition commands, to construct a pattern.

9.4.1 Examples

- *PATTERN run*
  All words which start with 'run' are selected.
- *PATTERN *ing
  All words which end with 'ing' are selected.
- *PATTERN pre*ed
  All words which start with 'pre' and end in 'ed' are selected.
- *PATTERN *ing *ed
  All words ending with 'ing' together with all ending with 'ed' This is equivalent to having two *PATTERN commands one containing '*ing' and the other '*ed'
- *PATTERN a* b* c* All words starting with 'a' together with all starting with 'b' together with all starting with 'b' are chosen. This is equivalent to having three *PATTERN commands. One can use this feature to produce a full concordance in sections; first the 'a', 'b', and 'c's then 'd', 'e', 'f's etc.
- *PATTERN ....
  All four letter words will be chosen.
- *PATTERN .h.a...
  All six letter words with 2nd letter 'h' and 4th letter 'a' will be selected.
- *PATTERN *...ing All words of *at least* six letters which end in 'ing' will be picked out.

9.4.2 The DUMMYaVARIABLEb option

This option must be used whenever a given pattern is to contain '*' or '.' as letters. The revised symbols apply for the current *PATTERN command only.

- *PATTERN DUMMY?VARIABLE- *run-
  The '?' temporarily replaces '.' as the dummy-symbol; the '-' temporarily replaces '*' as the variable-symbol. All words which start with '*run' are selected.
- *PATTERN DUMMY.VARIABLE? ?...ing*
  All words *at least* seven letters long and ending with 'ing*' are selected.

9.4.3 General Forms

- *PATTERN pattern1 pattern2 pattern3 etc.
  Note that DUMMY.VARIABLE* is implied before pattern1.
- *PATTERN DUMMYaVARIABLEb pattern1 pattern2 pattern3 etc.
  The character 'a' becomes the new dummy-symbol, overriding '.', the character 'b' becomes the new variable-symbol, overriding '*'.

Notes:

- At least one pattern must appear on the command.
- A pattern consists of *letters*,the dummy-symbol, and the variable symbol in any combination.
- When the CLOC itemizing strategy is in use each explicit pattern will be carefully mapped to one which will match the various 'words' in the vocabulary. Thus the pattern 'run*' will match 'run' 'running' 'Run' 'RUNNING' etc. Padding letters will be ignored since the vocabulary words do not contain them. For example:-
  - if ′ was a padding letter then .... would match 'don′t'
  - if ′ was a deferred letter then .... would NOT match 'don′t'

  This is because padding letters are removed before the word is stored in the vocabulary table, so it looks like 'dont'. In practice it is sufficient for you to examine the vocabulary table printed using the WORDLIST command (q.v.) to find out what 'words' are in the vocabulary.

9.5 Combining Set Descriptions

Set descriptions can follow each other. The vocabulary selected is sum of the vocabularies chosen by the separate set descriptions. For example:-

```
*FREQUENCY          (100 TO 500)
*LIST OF WORDS      me you we they
*PATTERN            ...ing
```

The above sequence of commands defines a set of words which contain:- all words of frequency 100 to 500 inclusive, the words 'me' 'you' 'we' 'they', and all six letter words which end in 'ing'.

9.6 Choosing a vocabulary

9.6.1 The EVERY WORD Command

The command EVERY WORD specifies the entire vocabulary of the text.

9.6.2 The SELECT WORDS Command

The SELECT WORDS command is used, in conjunction with several subsidiary commands, to define a given portion of the vocabulary.

9.6.3 The EXCLUDING Command

The EXCLUDING command can be used to remove unwanted words and reduce the size of the above collection.

9.6.4 The INCLUDING Command

The command INCLUDING is provided in case your exclusion commands remove too many words.

9.6.5 Combining the above commands

The set of words defined using the SELECT WORDS, EXCLUDING, or INCLUDING commands is specified by way of several subsidiary commands. These are termed *set description* commands and are described in section 9.1.

To choose a collection of words one can use either of the following two constructions, without supplying an exclusion list.

(a) EVERY WORD	The entire vocabulary is selected
(b) SELECT WORDS	The following *set description*
set description	describes the words to be used.

The exclusion list can be placed after either of the above constructions
to give the following alternatives.

(c) EVERY WORD	The entire vocabulary
EXCLUDING	excluding
set description	this *set description* is selected.
(d) SELECT WORDS	The words specified in
set description1	*set description1* are used,
EXCLUDING	excluding those words in
set description2	*set description2*.
(e) EVERY WORD	The entire vocabulary
EXCLUDING	excluding
set description1	this *set description1*,
INCLUDING	but including

set description2 *set description2*

(f) SELECT WORDS The words specified in
set description1 this *set description1*
EXCLUDING excluding
set description2 this *set description2*
INCLUDING but including
set description3 the *set description3*

The commands EVERYWORD and SELECTWORDS *set description* define a working set of words. You can then use the EXCLUDING commands to remove words from the working set, and the INCLUDING commands to add words to the working set. You can repeat the EXCLUDING and INCLUDING commands as often as you need to get precisely the collection of words that you want.

9.6.6 Examples

- SELECT WORDS
  *PATTERN *ing
  All words that end with 'ing' are selected.

- SELECT WORDS
  *PATTERN *ing
  EXCLUDING
  *LIST OF WORDS running jumping
  All words ending with 'ing' apart from 'running' and 'jumping' are chosen.

- EVERY WORD
  EXCLUDING
  *PATTERN *ing
  INCLUDING
  *LIST OF WORDS running jumping
  Here we choose the whole vocabulary less the words ending with 'ing', but with 'running' and 'jumping' included.

10 TASK SELECTION COMMANDS

10.1 sorting criteria

Some commands such as `WORDLIST` will print your selected vocabulary in ascending alphabetic order unless you specify otherwise. These commands allow you to change the sorting criteria by means of a keyword described below:-

ALPHA This causes the package to sort the words into ascending alphabetic order. The collating order for letters is taken from the word definition commands.

DALPHA This causes the package to sort the words into descending alphabetic order. The collating order for letters is taken from the word definition commands.

REVALPHA This causes the package to sort the selected words into reverse alphabetic order, in which words with similar endings sort together. The collating order for letters is taken from the word definition commands.

AFREQ This causes the package to sort the words into ascending frequency order. Words having the same frequency of occurrence will be sorted in ascending alphabetic order.

DFREQ This causes the package to sort the selected words into descending frequency order. Words having the same frequency of occurrence will be sorted in ascending alphabetic order.

FIRST This causes the package to sort the selected words in the order in which they *first occur* in the text. Note that words are printed across the page.

LAST This causes the package to sort the selected words in the order in which they *last occur* in the text. Note that words are printed across the page.

ALENGTH This causes the package to sort the selected words in ascending length order, which is in order of their length in characters ignoring any deferred (or padding) letters. Words of equal length will be sorted in ascending alphabetic order.

DLENGTH This causes the package to sort the selected words in descending length order, which is the descending order of their length in characters ignoring any deferred (or padding) letters. Words of equal length will be sorted in ascending alphabetic order.

AXLENGTH This causes the package to sort the selected words in ascending extended length order, which is the order of their length in characters including any deferred (or padding) letters. Words of equal length will be sorted in ascending alphabetic order.

DXLENGTH This causes the package to sort the selected words in descending extended length order, which is the descending order of their length in characters including any deferred (or padding) letters. Words of equal length will be sorted in ascending alphabetic order.

10.2 style

style is the type of citations that are required.

- KWIC — key word on context, in which the word of interest is centralised on the print line. CENT can be used as a synonym for KWIC.
- LEFT — in which the line of context is printed as far to the left as possible.
- NULL — no citations of any kind will be printed, but the frequency counts will be printed.

10.3 citation width

citation width indicates the amount of context to be printed. This takes the form:

- CITE *integer1* BY *integer2*
  in which *integer1* words are printed before the keyword, and *integer2* words are printed after the keyword.
- CITE FROM*char1*TO*char2*INCLUSIVE
  This option causes the package to print the citations between two given characters *char1* and *char2*. The left context begins with character *char1*, and the right context ends with *char2*. When INCLUSIVE is present the characters *char1* and *char2* are removed from the printed line.
- CITE FROM*char1*TO*char2*EXCLUSIVE
  This option causes the package to print the citations between two given characters *char1* and *char2*. The left context begins with character *char1*, and the right context ends with *char2*. When EXCLUSIVE is present the characters *char1* and *char2* are contained in the printed line.

10.4 offset

offset is optional and when present takes the form:

- ABOUT NODE+*integer*
  This option allows citations to be printed about words near to the keyword. For example, ABOUT NODE+1 causes all citations to be printed as if the word to the right of the keyword was being used for citations.
- ABOUT NODE-*integer*
  This option allows citations to be printed about words near to the keyword. For example, ABOUT NODE-1 causes all citations to be printed as if the word to the left of the keyword was being used for citations.

- ABOUT COLLOCATE+*integer*
  This option is only relevant when using the COLLOCATIONS command. It allows citations to be printed about words near to the collocate. For example, ABOUT COLLOCATE+1 causes all citations to be printed as if the word to the right of the collocate was being used for citations.

- ABOUT COLLOCATE-*integer*
  This option is only relevant when using the COLLOCATIONS command. It allows citations to be printed about words near to the collocate. For example, ABOUT COLLOCATE-1 causes all citations to be printed as if the word to the left of the collocate was being used for citations.

10.5 references

reference allows each citation to be prefixed with portions of references which are embedded in the text data. The instruction takes the form:

10.5.1 Example

REFS A4P2L6

10.5.2 General Form

- REFS *letter number letter number ... letter number*
  The *letter* must be from A to Z, and identifies an embedded text reference. The number of characters printed for the reference is given by *number*. When printed, each reference will be separated from the next by one space.

- NOREFS
  When this keyword is present no text references of any kind will be printed.

- LINEREFS0
  The first line of your text known as 'line 0'.

- LINEREFS1
  The first line of your text known as 'line 1'. This is the default.

- WORDREFS0
  The first word of your text known as 'word 0'.

- WORDREFS1
  The first word of your text known as 'word 1'. This is the default.

10.5.3 Defaults

- When *sorting criterion* is absent, ALPHA is assumed
- When *style* is absent, KWIC is assumed
- When *citation width* is absent, CITE 4 BY 4 is assumed
- When *offset* is absent, ABOUT NODE+0 is assumed
- When *reference* is absent, an absolute record number is used and LINEREFS1,WORDREFS1 is assumed.

10.6 Task Summary

The following commands operate on the previously selected collection of words. Each command specifies an action to be performed. The following tasks can be selected.

WORDLIST sorting the chosen words into alphabetic and/or frequency order

INDEX printing a word-index

CONCORDANCE producing a concordance of the chosen words

CO-OCCURRENCE + finding co-occurrences of words

COLLOCATIONS discovering the collocations within the context of the chosen words.

STATISTICS count aspects of the text

WRITETEXT + write out the itemized text

NEWLINE + output a newline

NEWPAGE + output a newpage

MESSAGE + output a message

NOTE + include a comment

FINISH + terminate the run of the package

Each command can be included as often as required. Those marked + above do not need to be preceded by any word selection statements.

10.7 The WORDLIST Command

This command causes the package to sort the previously chosen collection of words into order, and print them. The type of sorted list that is produced is determined by the keyword in the specification field. This allows one to produce word counts in alphabetic order, reverse alphabetic order, etc. These are printed across the page, the number per line is determined by the maximum word length and output line width. In all word lists each word is preceded by its frequency of occurrence.

10.7.1 Examples

- `WORDLIST ALPHA`
  An alphabetic wordlist, is produced.
- `WORDLIST REVALPHA`
  A reverse alphabetic wordlist, i.e. one in rhyming order, is produced.
- `WORDLIST AFREQ`
  A wordlist in ascending frequency order is printed.

10.7.2 General form

WORDLIST *sorting criterion*

where the *sorting criterion* is one of the options described in Section 10.1.

In each of the above cases, each word printed is preceded by its frequency of occurrence in the text. By default, the words and their frequencies are printed across the page, rather than in columns. This is done so that you can easily write a program (say in SNOBOL or BASIC) to reformat the output from the CLOC package as suitable data for another package, say for statistical analysis or graph plotting.

10.8 The INDEX Command

This command instructs the package to print a word index of the selected words. The parameters in the specification field allow you to presort the keywords, and optionally supply the form of text reference that will be used.

10.8.1 Examples

- `INDEX`
  Produces a word index in ascending alphabetic order. Each reference to a line is specified by a simple line number.

- `INDEX  ALPHA`
  Produces a word index in ascending alphabetic order. Each reference to a line is specified by a simple line number.

- `INDEX  REVALPHA,REFS P2 L2`
  A reverse alphabetic word index is produced. Each reference gives information about 'Page number' and 'Line within page' assuming that `P` and `L` references are included in the text for each page.

- `INDEX  NOREFS`
  An alphabetical word index is produced. No text references of any kind are printed.

10.8.2 General Form

INDEX *sorting criterion , reference*

where the *sorting criterion* is one of the options described in Section 10.1. The chosen set of keywords are sorted into this order before the word index is produced.

reference allows each word to be referenced with portions of references which are embedded in the text data. See Section 10.5 for details.

10.9 The CONCORDANCE Command

This command instructs the package to print a concordance of the selected words. The parameters in the specification field allow the user to presort the keywords; to choose the citation style; to select a citation width; and optionally supply a text reference.

10.9.1 Examples

- `CONCORDANCE`
  The keywords are alphabetically sorted. Keyword in context citations are printed which have four words on either side of the word of interest. Each line is identified by a simple record number.

- `CONCORDANCE  ALPHA,KWIC,CITE 4 BY 4`
  The keywords are alphabetically sorted. Keyword in context citations are printed which have four words on either side of the word of interest. Each line is identified by a simple record number.

- `CONCORDANCE  REVALPHA,CITE 6 BY 6,REFS P2 L2`
  A reverse alphabetic KWIC concordance is produced with six words of context on either side of the keyword. Each line printed is prefixed by text reference information giving 'Page number' and 'Line within page' assuming that `P` and `L` references are included in the text for each page.

- `CONCORDANCE  CITE FROM.TO.INCLUSIVE`
  An alphabetical KWIC concordance is printed. The keyword is surrounded by as many words as possible up to and including a '.' character. By this means a sentence of context is printed.

- `CONCORDANCE  REVALPHA,LEFT,CITE FROM/TO/EXCLUSIVE,REFS S2 P3 L2`
  Assuming that '/' has been declared as a 'newline' character (see INPUT DETAILS command), this example will print a reverse alphabetic concordance. Each citation will consist of a full line of context, left justified and prefixed with 'section', 'page' and 'line number' information, assuming that `S`, `P` and `L` references have been included in the text.

- `CONCORDANCE  REVALPHA,CITE 6 BY 6,NOREFS`
  A reverse alphabetic KWIC concordance is produced with six words of context on either side of the keyword. No text references of any kind will precede the citations.

- `CONCORDANCE  CITE 4 BY 4, ABOUT NODE-1`
  The keywords are alphabetically sorted. Keyword in context citations are printed which have four words on either side of the word of interest. Each line is identified by a simple record number. The citations will be printed with the word before the keyword centralised on the line.

10.9.2 General Form

CONCORDANCE *sorting criterion, style, citationwidth, offset, reference*

where the *sorting criterion* is one of the options described in Section 10.1. The chosen set of keywords are sorted into this order before the concordance is produced.

style is the type of concordance required. See Section 10.2 for details.

citationwidth indicates the amount of context to be printed. This is described in Section 10.3.

offset allows you to choose which node will be centred. See Section 10.4 for details.

reference allows each citation to be prefixed with portions of references which are embedded in the text data. This is described in Section 10.5 .

10.10 The CO-OCCURRENCE Command

This command is used when you know two or more words and need to study how they occur in a text. The parameters in the specification field allow you to choose a style for presenting the results; to select a citation width and to optionally supply a text reference. Subsidiary commands enable you to choose word pairs or phrases of interest and also to choose a series of words separated by an arbitrary word distance. *Note*: The CO-OCCURRENCE command need not be preceded by any word selection commands.

10.10.1 Examples

- ```
  CO-OCCURRENCE
  *PHRASE         you are
  *PHRASE         now is the winter
  ```
  This produces 4 BY 4 KWIC citations of positions in a text in which the phrase 'you are' occurs. After these have been found, all occurrences of the phrase 'now is the winter' are found. In both cases punctuation is ignored, hence all examples are discovered.

- ```
  CO-OCCURRENCE   CITE 8 BY 8, REFS P2 L2
  *PHRASE         just man
  *SERIES         a UPTO6 goat
  *SERIES         how GAP2 see
  *SERIES         he UPTO3 miner GAP1 and
  ```
  This gives 8 BY 8 citations are printed centralised on the page, each prefixed with a 2 figure page number and 2 figure line number. After printing all occurrences of the phrase 'just man', the three *SERIES commands are obeyed in order. The first command finds all occurrences of 'a ... goat' where 'a' and 'goat' are separated from each other by 0, 1, 2, 3, 4, 5, or 6 words of context. The second command finds all occurrences of 'how' and 'see' separated from each other by *precisely* two arbitrary words. The third *SERIES command shows how you can mix the UPTO*number* and GAP*number* options within one specification. This example will find all occurrences of 'he' 'miner' 'and' where 'he' and 'miner' are separated by 0, 1, 2 or 3 arbitrary words and with 'miner' and 'and' separated by exactly 1 arbitrary word.

- ```
  CO-OCCURRENCE
  *PATTERN        *ing *ly
  ```
  This shows how you can also put a list of patterns which will be searched in the order they are given. This represents a skeletal form of a phrase.

- ```
  CO-OCCURRENCE
  *CHOICE         be is as was were
  ```
  This example shows how you can print the citations of the words BE IS AS WAS WERE as they are found in the text.

10.10.2 General Form

- CO-OCCURRENCE *style, citation width, offset, reference, counting*
- *PHRASE *list*
- *SERIES *word type number word ... type number word*
- *PATTERN *pattern1 pattern2 pattern3* etc.
- *PATTERN DUMMYaVARIABLEb *pattern1 pattern2 pattern3* etc.
- *CHOICE *choices*

style is the type of citation required. This is described in Section 10.2 .

citation width specifies the amount of context to be printed. This is described in Section 10.3

offset allows you to choose which node will be centred. See Section 10.4 for details. When citations are printed the node for offset purposes is the first word of a *PHRASE or a *SERIES or a *PATTERN. This option allows you to shift the citation left or right as required.

reference allows each citation to be prefixed with portions of references which are embedded in the text data. This is described in Section 10.5 .

counting allows you to produce frequency counts of the citations selected by the *PHRASE *SERIES *PATTERN or *CHOICE commands. You choose from:-

- COUNT — a citation frequency count precedes the citations.
- NOCOUNT — no frequency count is printed. This option is the default.

NOTE: You can use CO-OCCURRENCE NULL,COUNT if you just want the frequency counts and not the actual citations.

type number is either UPTO*number* or GAP*number*, and where

UPTO*number* means 0, 1, 2, 3.. *number* of arbitrary words may occur at this position.

GAP*number* means exactly *number* arbitrary words occur at this position.

list is one or more words separated from each other by spaces. These will be treated as consecutive words.

choices is one or more words separated from each other by spaces. These will be treated as alternative possibilities at a single position in the text rather than as a phrase.

Notes

- Each word must be separated from the *type* and *number* by at least one space.
- The *number* can be 0, in which case UPTO0 and GAP0 are equivalent and indicate that the two words are to be adjacent. *PHRASE is the degenerative case of a *SERIES in which all *numbers* are zero.
- *PHRASE and *SERIES and *PATTERN and *CHOICE commands can be repeated as often as required.

10.11 The COLLOCATIONS Command

This command instructs the package to examine the collocates in the context surrounding each selected word. Those collocates which are found to have a significant affinity to the selected word will have their context printed. This option allows closely associated pairs of words to have their context printed. The occurrence of a collocate will be counted whenever it occurs in a range several words to the left or to the right of the selected word. This region is termed a *span*, the size of which can be chosen using a subsidiary command.

10.11.1 Examples

- `COLLOCATIONS`
  The keywords are alphabetically sorted. The collocates are printed as keyword in context citations which have four words on either side of the word of interest. Each line is identified by a simple record number.

- `COLLOCATIONS  ALPHA,KWIC,CITE 4 BY 4`
  The keywords are alphabetically sorted. The collocates are printed as keyword in context citations which have four words on either side of the word of interest. Each line is identified by a simple record number.

- `COLLOCATIONS  REVALPHA,CITE 6 BY 6,REFS P2 L2`
  Reverse alphabetic KWIC citations are produced with six words of context on either side of the keyword. Each line printed is prefixed by text reference information giving 'Page number' and 'Line within page' assuming that `P` and `L` references are included in the text for each page.

- `COLLOCATIONS  CITE FROM.TO.EXCLUSIVE, ABOUT COLLOCATE-1`
  Alphabetical KWIC citations are printed with the word before the collocate as the central keyword. The keyword is surrounded by as many words as possible up to and including a '.' character. By this means a sentence of context is printed.

- `COLLOCATIONS  FIRST,CITE 6 BY 6, ABOUT COLLOCATE`
  The nodes are sorted in order of their first occurrence in the text. Each citation will be centred on the collocate and have 6 words of context on either side.

- `COLLOCATIONS  FIRST,CONDENSED`
  The nodes are sorted in order of their first occurrence in the text. The collocated will be printed in a simple table which will give the frequency and word of the node, it's collocate and the pair.

10.11.2 General forms

- COLLOCATIONS *sorting criterion, style, citation width, offset, reference*
- COLLOCATIONS *sorting criterion*, CONDENSED
- COLLOCATIONS *sorting criterion*, FULLCONDENSED

where *sorting criterion* is described in Section 10.1 The chosen set of keywords are sorted into this order before the collocations are produced.

style is the type of citation required. This is described in Section 10.2.

citation width indicates the amount of context to be printed. This is described in Section 10.3.

offset allows you to choose which node or collocate will be centred. See Section 10.4 for details.

reference allows each citation to be prefixed with portions of references which are embedded in the text data. This is described in Section 10.5.

The CONDENSED option causes the discovered collocates to be listed in a simple tabular form. This gives on quick look at an author's word associations, allowing one to choose accurately which nodes to select and which collocates to reject. The following example illustrates the format of the table produced by this command.

```
condensed collocation analysis of 71 nodes
==========================================
    node               collocate          pair
    ----               ---------          ----
    6 and                6 of               4
    6 and                9 the              4
    6 and                9 a                4
    4 in                 9 a                4
    6 of                 9 the              4
    9 the                6 university       5
    9 the                6 of               4
    9 the                6 and              4
```

```
    6 university         9 the               5
    6 university         9 a                 4
```

The FULLCONDENSED option produces a table which also contains the totalspan and the meanspan as follows:-

```
full condensed collocation analysis of 71 nodes
===============================================
* text has 113 running words, 71 distinct words
* span 4 by 4 restricted
* meanspan = totalspan/ (nodefrequency * (leftspan + rightspan))
    node               collocate           pair  totalspan  meanspan
    ----               ---------           ----  ---------  --------
    6 and                6 of                4          47  0.979167
    6 and                9 the               4          47  0.979167
    6 and                9 a                 4          47  0.979167
    4 in                 9 a                 4          28  0.875000
    6 of                 9 the               4          31  0.645833
    9 the                6 university        5          54  0.750000
    9 the                6 of                4          54  0.750000
    9 the                6 and               4          54  0.750000
    6 university         9 the               5          39  0.812500
    6 university         9 a                 4          39  0.812500
```

Subsidiary commands to the COLLOCATIONS command

10.12 The *SPAN command

This command specifies the range of searching that is done when the package performs a collocation analysis. The specification field of this command defines the range which will be searched in terms of the number of words to the left and right of the word of interest.

10.12.1 Examples

- *SPAN 4 BY 4
- *SPAN 4 BY 4 RESTRICTED
- *SPAN 4 BY 6 UNRESTRICTED

10.12.2 General form

*SPAN *integer1* BY *integer2 qualifier*

where *integer1* indicates the number of words to be searched before the word of interest and *integer2* indicates the number of words after the word of interest. Either of these integers may be zero.

qualifier is either UNRESTRICTED or RESTRICTED.

UNRESTRICTED means that all words in the left and right span will be counted as collocates.

RESTRICTED means that when a pair of nodes are closer that leftspan+rightspan, overlapping collocates will be counted once only, and the node will not be counted as a collocate.

Note that the *citation width* could be narrower than the *span*. This will cause some collocates to appear to be absent from the context, they will however be found in the text. Normally the *citation width* should be greater than the *span*.

10.12.3 Defaults

- When *SPAN is absent *SPAN 4 BY 4 UNRESTRICTED is assumed
- When *qualifier* is absent UNRESTRICTED is assumed.

10.13 The *FREQUENCY command

This command is used to select significant collocates according to their frequency of occurrence. Only those collocates whose *collocation frequency* is within the specified limits will have their citations printed. Note that the frequency of occurrence of a collocate is different from the frequency of occurrence of the same object treated as a *word*.

10.13.1 Example

*FREQUENCY (100 TO 500)

10.13.2 General Form

*FREQUENCY *expression*

Where *expression* is one or more *term*s connected by OR symbols. A *term* is one of the following:

- *integer* for example 10
  only collocates occurring exactly *integer* times will be selected.
- >*integer* for example >10
  only collocates occurring more than *integer* times will be selected.
- <*integer* for example <10
  only collocates occurring less than *integer* times will be selected.
- (*integer1* TO *integer2*) for example (100 TO 500)
  only collocates lying in the range *integer1* to *integer2* inclusive will be selected. Note that *integer1* must be smaller than *integer2*.

10.13.3 Defaults

When the *FREQUENCY command is absent `*FREQUENCY  >1` is assumed.

10.14 The EVERY COLLOCATE Command

This command ensures that every collocate chosen by the *SPAN and *FREQUENCY commands will be considered for selection.

10.14.1 Example and general form

EVERY COLLOCATE

10.15 The SELECTCOLLOCATE Command

This must be followed be a series of *set description*s and ensures only those words in the *set description*s will be considered as collocates.

10.15.1 Example

```
SELECTCOLLOCATE
*LIST OF WORDS   father mother
*PATTERN         *ing
```

This example only selects the collocates 'father' and 'mother' and all collocates which end with 'ing'.

10.16 The REJECTING Command

This must be followed be a series of *set description*s and ensures only those words in the *set description*s will be removed from the collocates that have been previously chosen. This command allows you to specify an exclusion list for collocates. Its function is to remove insignificant collocates.

10.16.1 Example

```
REJECTING
*PATTERN          run*
```

10.17 The ACCEPTING Command

This must be followed be a series of *set description*s and ensures those words in the *set description*s will be added to the collocates that have been previously chosen. It is most often used to supply words that were excluded because the REJECTING command was too restrictive.

The above commands can only be supplied in a fixed order. The order is similar to that for *word selection* described earlier but this time we use the collocate selection commands. The commands EVERY COLLOCATE or SELECTCOLLOCATE are alternatives, only one can be chosen, but if both are absent the command EVERY COLLOCATE is assumed. Thus we can say:-

- EVERY COLLOCATE
- SELECTCOLLOCATE
  set description

The commands REJECTING or ACCEPTING can be chosen as often as necessary so that you can get just those collocates that you want.

10.17.1 Examples

- `REJECTING`
  `*LIST OF WORDS          the but and`

  The specific collocates 'the' 'but' 'and' will not be printed.

- `REJECTING`
  `*PATTERN                *ing *ed`

  All collocates which end in 'ing' or 'ed' will not be printed.

- `REJECTING`
  `*FREQUENCY              (100 TO 700)`

  All collocates whose vocabulary frequency of occurrence is between 100 and 700 inclusive will not be printed.

- `SELECTCOLLOCATE`
  `*PATTERN                run*`

  Only those collocates which start with 'run' will be chosen.

- `SELECTCOLLOCATE`
  `*PATTERN                run*`
  `REJECTING`
  `*LIST OF WORDS          running`

  All collocates which start with 'run' but excluding the word 'running' will be chosen.

10.18 The STATISTICS Command

The subcommands of this command will count aspects of the text and produce tables of data.

10.19 The *PROFILE Command

This command will produce a table containing the number of words which fall into frequency bands and a cumulative total of those words.

10.19.1 Examples

- *PROFILE
- *PROFILE BINSIZE 50

10.19.2 General form

*PROFILE BINSIZE *integer*

The command will cause frequencies to be grouped into *integer* ranges.

10.19.3 Default

When *integer* is absent 100 is assumed.

10.20 The WRITETEXT Command

This command will cause CLOC to print out the itemized text. Each line can be prefixed with a text reference to the first word in each line.

10.20.1 Examples

- WRITETEXT
  Each line of the text will contain a reference which will be a simple record number.
- WRITETEXT REFS P3 L4
  Each line of the text will contain a reference which will be a 3 character page number P and a 4 figure line number L .
- WRITETEXT NOREFS
  No references of any kind will precede each line of text.

10.20.2 General form

WRITETEXT *reference*

where *reference* is optional, and is described in Section 10.5.

10.21 The NEWLINE Command

This command will insert one or more newlines in the CLOC results file. You can use this feature to widen the gap between the results from successive tasks.

10.21.1 Examples

- NEWLINE
- NEWLINE 1
- NEWLINE 5

10.21.2 General form

NEWLINE *integer*

The command will cause *integer* newlines to be sent to the CLOC results file.

10.21.3 Default

When *integer* is absent, a value of 1 is assumed.

10.22 The NEWPAGE Command

This command will cause a newpage to be thrown on the CLOC results file.

10.22.1 Example and general form

NEWPAGE

10.23 The MESSAGE Command

This command will send the contents of the specification field, and any continuation, to the CLOC results file.

10.23.1 Example

MESSAGE Henry the Fifth (part 1)

10.23.2 General form

MESSAGE *character sequence*

The *character sequence* will be sent to the CLOC results file.

10.24 The NOTE Command

This command can be used at your convenience to insert a commentary about the following or preceding task. All characters in the specification field of this command are ignored.

10.24.1 Example

NOTE THIS TEXT IS TAKEN FROM WORDSWORTH

10.24.2 General Form

NOTE *character sequence*

Where *character sequence* is totally ignored. This information will be printed on the CLOC diagnostic and information channel along with the other control statements.

10.25 The FINISH Command

This command must be the final one in the sequence. It informs the package that no further commands are to follow.

10.25.1 Example and general form

FINISH

A EXAMPLES

Before preparing a large volume of text, or before trying out the CLOC package on some prepared text, you should run the example programs given below. To do this you will need to read the documentation on using the package on your local computer. This will provide the basic information you need to know to run any CLOC job. You should compare the results produced by the computer with those given in the example programs to check your understanding of the command language. You are also recommended to vary the given commands in order to gain some feel for their effect. Often the examples will contain all the commands you need to solve your given problem, in which case all you need do is supply your own text. All the examples in this section use the following extract from 'CAUTION: LOW FLYING DUCKS' by the author.

```
     The University ; "A society of individuals living and working
together for the advancement of learning and the dissemination of
knowledge".  (University of York Development Plan).
     In 1617 James I received a petition requesting a University
for York.  This was followed by a petition to Parliament in 1652,
and a deputation to the University Grants Committee in 1947.  The
University officially opened in 1963 with a student population
comprising 216 undergraduates and 12 postgraduates.
     The site consisted of 190 acres of marshy land and a large
decrepit Elizabethan mansion, Heslington Hall, destined to become
the administration building.  Draining the saturated ground was
accomplished by widening a natural stream and creating a fourteen
acre artificial lake around which the University was constructed.
```

If the above were coded on punched cards using the recommendations in section 2 of this guide, it would look like this, the following:-

```
     =THE =UNIVERSITY : "=A SOCIETY OF INDIVIDUALS LIVING AND WORKING
TOGETHER FOR THE ADVANCEMENT OF LEARNING AND THE DISSEMINATION OF
=KNOWLEDGE".   (=UNIVERSITY OF =YORK =DEVELOPMENT =PLAN)
     =IN 1617 =JAMES =I RECEIVED A PETITION REQUESTING A =UNIVERSITY
FOR =YORK,  =THIS WAS FOLLOWED BY A PETITION TO =PARLIAMENT IN 1652,
AND A DEPUTATION TO THE =UNIVERSITY =GRANTS =COMMITTEE IN 1947.  =THE
=UNIVERSITY OFFICIALLY OPENED IN 1963 WITH A STUDENT POPULATION
COMPRISING 216 UNDERGRADUATES AND 12 POSTGRADUATES.
     =THE SITE CONSISTED OF 190 ACRES OF MARSHY LAND AND A LARGE
DECREPIT =ELIZABETHAN MANSION, =HESLINGTON =HALL, DESTINED TO BECOME
THE ADMINISTRATION BUILDING.  =DRAINING THE SATURATED GROUND WAS
ACCOMPLISHED BY WIDENING A NATURAL STREAM AND CREATING A FOURTEEN
ACRE ARTIFICIAL LAKE AROUND WHICH THE =UNIVERSITY WAS CONSTRUCTED.
```

We will assume that you are using a computer that has both upper and lower case and that you stored the text in the form that it was first written. The following examples show how CLOC commands are put together to perform the following tasks:-

- Alphabetic sorting
- The pattern feature
- The exclusion list
- Producing a concordance
- Finding collocations

Example number 1 *Alphabetic Sorting*
The following commands cause CLOC to read the above text and sort the vocabulary into ascending alphabetic order.

```
ITEMIZEUSING)cloc
*LETTERS)abcdefghijklmnopqrstuvwxyz
OUTPUTDETAILS)width80
EVERYWORD
WORDLIST)alpha
FINISH
```

The output produced by the computer is a listing of the commands, including the defaults and comments, and the results of the sorting process.

a) Control statement listing

```
default                input details  width80
                       ITEMIZEUSING)cloc
                       *LETTERS)abcdefghijklmnopqrstuvwxyz
default                *separators    !"#$%&'()*+,-./0123456789:;<=>?@[\]^_`{|}~
the text contains :
     113  running words
      71  distinct words
and the maximum word length is 14 characters

                       OUTPUTDETAILS)width80
                       EVERYWORD
                       WORDLIST)alpha
                       FINISH
```

b) The Results.

```
table of 71 words in ascending alphabetic order
================================================
    9 a                 1 accomplished      1 acre              1 acres
    1 administration    1 advancement       6 and               1 around
    1 artificial        1 become            1 building          2 by
    1 committee         1 comprising        1 consisted         1 constructed
    1 creating          1 decrepit          1 deputation        1 destined
    1 development       1 dissemination     1 draining          1 elizabethan
    1 followed          2 for               1 fourteen          1 grants
    1 ground            1 hall              1 heslington        1 i
    4 in                1 individuals       1 james             1 knowledge
    1 lake              1 land              1 large             1 learning
    1 living            1 mansion           1 marshy            1 natural
    6 of                1 officially        1 opened            1 parliament
    2 petition          1 plan              1 population        1 postgraduates
    1 received          1 requesting        1 saturated         1 site
    1 society           1 stream            1 student           9 the
    1 this              3 to                1 together          1 undergraduates
    6 university        3 was               1 which             1 widening
    1 with              1 working           2 york
```

Example number 2 *The PATTERN Feature*
The example illustrates how the pattern feature can be used to select, from the above text, words which end in a standard way. The selected words are then listed in alphabetic order.

```
ITEMIZEUSING)cloc
*LETTERS)abcdefghijklmnopqrstuvwxyz
OUTPUTDETAILS)width80
SELECTWORDS
*PATTERN) *ing
WORDLIST)alpha
FINISH
```

The output produced is a listing of the commands and the results.
a) Control statement listing

```
default                input details  width80
                       ITEMIZEUSING)cloc
                       *LETTERS)abcdefghijklmnopqrstuvwxyz
default                *separators    !"#$%&'()*+,-./0123456789:;<=>?@[\]^_‘{|}~
the text contains :
     113  running words
      71  distinct words
and the maximum word length is 14 characters

                       OUTPUTDETAILS)width80
                       SELECTWORDS
                       *PATTERN) *ing
                       WORDLIST)alpha
                       FINISH
```

b) The Results.

```
table of 9 words in ascending alphabetic order
==============================================
     1 building          1 comprising        1 creating          1 draining
     1 learning          1 living            1 requesting        1 widening
     1 working
```

Example Number 3 *The Exclusion List*

This example shows how one can exclude a set of words from a previously selected set. The resultant collection is listed in ascending alphabetic order.

```
ITEMIZEUSING)cloc
*LETTERS)abcdefghijklmnopqrstuvwxyz
OUTPUTDETAILS)width80
SELECTWORDS
*FREQUENCY) >1
EXCLUDING
*LISTOFWORDS) a the of and
WORDLIST)alpha
FINISH
```

The output produced is a listing of the commands and the alphabetically ordered list.

a) Control statement listing

```
default                input details  width80
                       ITEMIZEUSING)cloc
                       *LETTERS)abcdefghijklmnopqrstuvwxyz
default                *separators    !"#$%&'()*+,-./0123456789:;<=>?@[\]^_`{|}~
the text contains :
     113  running words
      71  distinct words
and the maximum word length is 14 characters

                       OUTPUTDETAILS)width80
                       SELECTWORDS
                       *FREQUENCY) >1
                       EXCLUDING
                       *LISTOFWORDS) a the of and
                       WORDLIST)alpha
                       FINISH
```

b) The Results

```
table of 8 words in ascending alphabetic order
==============================================
    2 by               2 for              4 in               2 petition
    3 to               6 university       3 was              2 york
```

Example number 4 *Producing a CONCORDANCE*

The following commands will produce a concordance of the words selected. The output is centralized on the page and sorted in ascending alphabetic order.

```
ITEMIZEUSING)cloc
*LETTERS)abcdefghijklmnopqrstuvwxyz
OUTPUTDETAILS)width80
SELECTWORDS
*PATTERN) *ed
CONCORDANCE) KWIC,CITE 5 BY 5
FINISH
```

a) Control statement listing

```
default                input details  width80
                       ITEMIZEUSING)cloc
                       *LETTERS)abcdefghijklmnopqrstuvwxyz
default                *separators    !"#$%&'()*+,-./0123456789:;<=>?@[\]^_`{|}~
the text contains :
     113  running words
      71  distinct words
and the maximum word length is 14 characters

                       OUTPUTDETAILS)width80
                       SELECTWORDS
                       *PATTERN) *ed
                       CONCORDANCE) KWIC,CITE 5 BY 5
                       FINISH
```

b) The results

```
concordance of 8 nodes
======================
node  accomplished  occurs 1 times
12       Draining the saturated ground was accomplished by widening a natural

node  consisted  occurs 1 times
9          12 postgraduates.      The site consisted of 190 acres of marshy

node  constructed  occurs 1 times
13         around which the University was constructed.

node  destined  occurs 1 times
10              mansion, Heslington Hall, destined to become the

node  followed  occurs 1 times
5           University for York.  This was followed by a petition to Parliament

node  opened  occurs 1 times
7         1947.  The University officially opened in 1963 with a student

node  received  occurs 1 times
4              Plan).      In 1617 James I received a petition requesting a

node  saturated  occurs 1 times
11                building.  Draining the saturated ground was accomplished by
```

Example number 5 *Finding COLLOCATIONS*

The following commands cause the package to scan the context of the selected words, and to print the examples of their collocations. The output is centralised on the page and sorted in ascending alphabetic order.

```
                    ITEMIZEUSING)cloc
                    *LETTERS)abcdefghijklmnopqrstuvwxyz
                    OUTPUTDETAILS)width80
                    SELECTWORDS
                    *LISTOFWORDS) university
                    COLLOCATIONS) KWIC, CITE 5 BY 5
                    *FREQUENCY)>2
                    FINISH
```

a) Control statement Listing

```
default               input details  width80
                      ITEMIZEUSING)cloc
                      *LETTERS)abcdefghijklmnopqrstuvwxyz
default               *separators    !"#$%&'()*+,-./0123456789:;<=>?@[\]^_`{|}~
the text contains :
     113  running words
      71  distinct words
and the maximum word length is 14 characters

                      OUTPUTDETAILS)width80
                      SELECTWORDS
                      *LISTOFWORDS) university
                      COLLOCATIONS) KWIC, CITE 5 BY 5
default               *span          4 by 4
                      *FREQUENCY)>2
                      FINISH
```

b) The Results

```
collocation analysis of 1 nodes (cited about node)
==================================================
node  university  occurs 6 times
collocate  the  occurs 9 times
node-collocate pair occurs 6 times
1                                     The University ; "A society of
3       the dissemination of knowledge".  (University of York Development Plan
6                  and a deputation to the University Grants Committee in
6                  and a deputation to the University Grants Committee in
7           Grants Committee in 1947.  The University officially opened in
13        artificial lake around which the University was constructed.

node  university  occurs 6 times
collocate  a  occurs 9 times
node-collocate pair occurs 4 times
1                                     The University ; "A society of
4         received a petition requesting a University for York.  This was
4         received a petition requesting a University for York.  This was
6                  and a deputation to the University Grants Committee in

node  university  occurs 6 times
collocate  of  occurs 6 times
node-collocate pair occurs 3 times
1                                     The University ; "A society of
3       the dissemination of knowledge".  (University of York Development Plan
```

```
3       the dissemination of knowledge".  (University of York Development Plan

node  university  occurs 6 times
collocate  in  occurs 4 times
node-collocate pair occurs 3 times
6                   and a deputation to the University Grants Committee in
7           Grants Committee in 1947.  The University officially opened in
7           Grants Committee in 1947.  The University officially opened in
```

B Messages Produced by the CLOC package

Three categories of message are printed by the package, these are *errors*, *warnings*, and *comments*.

B.1 Error Messages

These cause the run of the package to be abandoned. Where the error is caused by a mistake in a command the figure 1 is printed under the faulty position on the command, 2 for the second error on the line, and so on up to 9 . Error messages take the form:-

ERROR - *text of message*

where the text is one of the following

- MISSING MANDATORY STATEMENT
  You have forgotten to include or have mis-spelt an essential command.
- CONTROL STATEMENT ENDS PREMATURELY
  The continuation field of 15 spaces was expected but not found.
- INCORRECT CONTROL STATEMENT A mistake has been found on the line, the symbol 1 points to it.
- UNKNOWN SYMBOL
  The item in the specification field has not been recognised.
- CHARACTER ALREADY DEFINED
  The indicated character has occurred on this or an earlier line.
- NO LETTERS PROVIDED
  The *LETTERS command exists, but no letters have been put on it.
- NUMBER IS TOO LARGE
  The indicated number is too large for the CLOC package to use.
- UPPER VALUE DOES NOT EXCEED LOWER
  The upper value in a frequency range is smaller than the lower value.
- NO WORDS FOUND
  The combination of word selection commands has chosen no words.
- FILE NOT PRODUCED BY CLOC MARK *mark*
  The file used by the GET TEXT command was not produced by an earlier run of the package.
- ABOVE STATEMENT NOT EXPECTED
  This line may be a mis-spelt or spurious command.
- SYMBOL NOT ALLOWED IN THIS CONTEXT
  The indicated symbol is not permitted there.
- CAPACITY EXCEEDED
  The text to be processed contains more words than the package is able to handle.
- NUMBER OF REFERENCES EXCEEDS *number*
  The text contains more text references than the package can accept.
- NO WORD SELECTION COMMANDS PROVIDED
  The commands EVERY WORD or SELECT WORDS are absent or mis-spelt.
- NUMBER EXPECTED AT THIS POSITION
  The previous CLOC keyword must be followed by a number.
- A NUMBER CANNOT BE PLACED HERE
  The previous CLOC keyword must *not* be followed by a number.
- ZERO NUMBER NOT PERMITTED

- SPACE NOT ALLOWED HERE

The package makes several checks on its operation and in certain instances may fail with the message SYSTEM ERROR *number*. Such an occurrence should be reported to your local advisory service.

B.2 Warning Messages

These are produced when the package finds a simple mistake in a command not important enough to cause a fatal error. The mistake will be ignored and the next command will be examined. The figure 1 is printed under the faulty position on the line, 2 under the second position (and so on up to 9). Warning messages take the form:-

WARNING - *text of message*

where the text is one of the following.

- CHARACTER ALREADY DEFINED
  The *SEPARATORS, *PADDING commands etc. contain repeated characters.
- SPURIOUS CHARACTERS FOUND AND IGNORED
  The specification field contains characters which should not be present, they will be ignored.
- SET DESCRIPTION SPECIFIES NO WORDS
  The combination of word selection commands resulted in no words selected.
- WORD(S) NOT FOUND
  The indicated words on the *LIST OF WORDS command are not present in the vocabulary of the text.
- ABOVE ITEM TOO LONG
  The word printed is longer than the system can cope with, trailing letters have been removed.
- NO REFERENCE INFORMATION IN TEXT
  The citation option REFS was chosen but no text references were placed in the text.
- NO WORDS MATCH THIS PATTERN
  The current vocabulary does not contain words of this form.

The package makes several checks on its operation and in certain instances may produce the message SYSTEM WARNING *number*. Such an occurrence should be reported to your local computer advisory service.

B.3 Comment messages

These are produced when the system has read the text and is about to read the task selection commands.

- TEXT FILE *name* SAVED ON *date* AT *time*
  The text has been read and stored in a permanent file to be later used by the GET TEXT command.
- TEXT FILE *name* ACCESSED. (SAVED ON *date* AT *time*)
  The GET TEXT command has accessed this file.
- The TEXT CONTAINS:
  *integer*1 RUNNING WORDS
  *integer*2 DISTINCT WORDS
  AND THE MAXIMUM WORD LENGTH IS *integer*3 CHARACTERS.

The text under analysis is *integer*1 words in length, and the vocabulary used contains *integer*2 different words. The longest word or words contain *integer*3 characters.

C References

- 'The RUNCLOC macro', Computer Centre Users Manual, University of Birmingham
- 'CLOC - An Applications Package in ALGOL 68R' Presented to 'Applications of ALGOL 68' Conference, April 1975, University of Liverpool.
- 'English Lexical Studies', J. McH. Sinclair, S. Jones and R. Daley.
- 'The COCOA Manual',D.B. Russell,ATLAS Computer Laboratory.
- 'Statistical Package for the Social Sciences (SPSS)', N.H. Nie, D.H. Bent, and C.H. Hull, Publ. McGraw-Hill, New York, 1970.
- 'CLOC: A Collocation Package', ALLC Bulletin, Vol. 5, No. 2, 1977.
- 'RATS: A Middle-level Text Utility System': Smith; Computers and the Humanities, Vol. 6, P.277.
- 'JEUDEMO: A Text-Handling System': Bratley, Lusigaan, and Ouellette, Computers in the Humanities. Pub. Edinburgh University Press.
- 'Computer Analysis of Natural Language': Reed 1973, Birmingham University Computer Centre, internal report 1973.
- 'CLOC User Guide', A.Reed,1975,Computer Centre,University of Birmingham.
- 'OXEYE: A Text Processing Package for the 1906A', L. Burnard, 1976, Oxford University Computing Service.
- 'CLOC: A General-purpose concordance and collocations generator'; A. Reed, J. L. Schonfelder, 1979, Aston University.
- 'OCP: Oxford Concordance Program', S. Hockey and I Marriott, October 1980, Oxford University Computing Service.
- 'Anatomy of a Text Analysis Package', A. Reed, Computer Lang., Vol. 9, No. 2, pp 89-96, 1984.

D Glossary

LETTER One of an arbitrary collection of graphic signs, used to construct words.

SEPARATOR This is composed of:-

1. A graphic sign which is not a LETTER.
2. An arbitrary sequence of 1 above.

WORD An arbitrary sequence of LETTERs, generally contiguous,but may contain graphic signs which are totally ignored during reading.

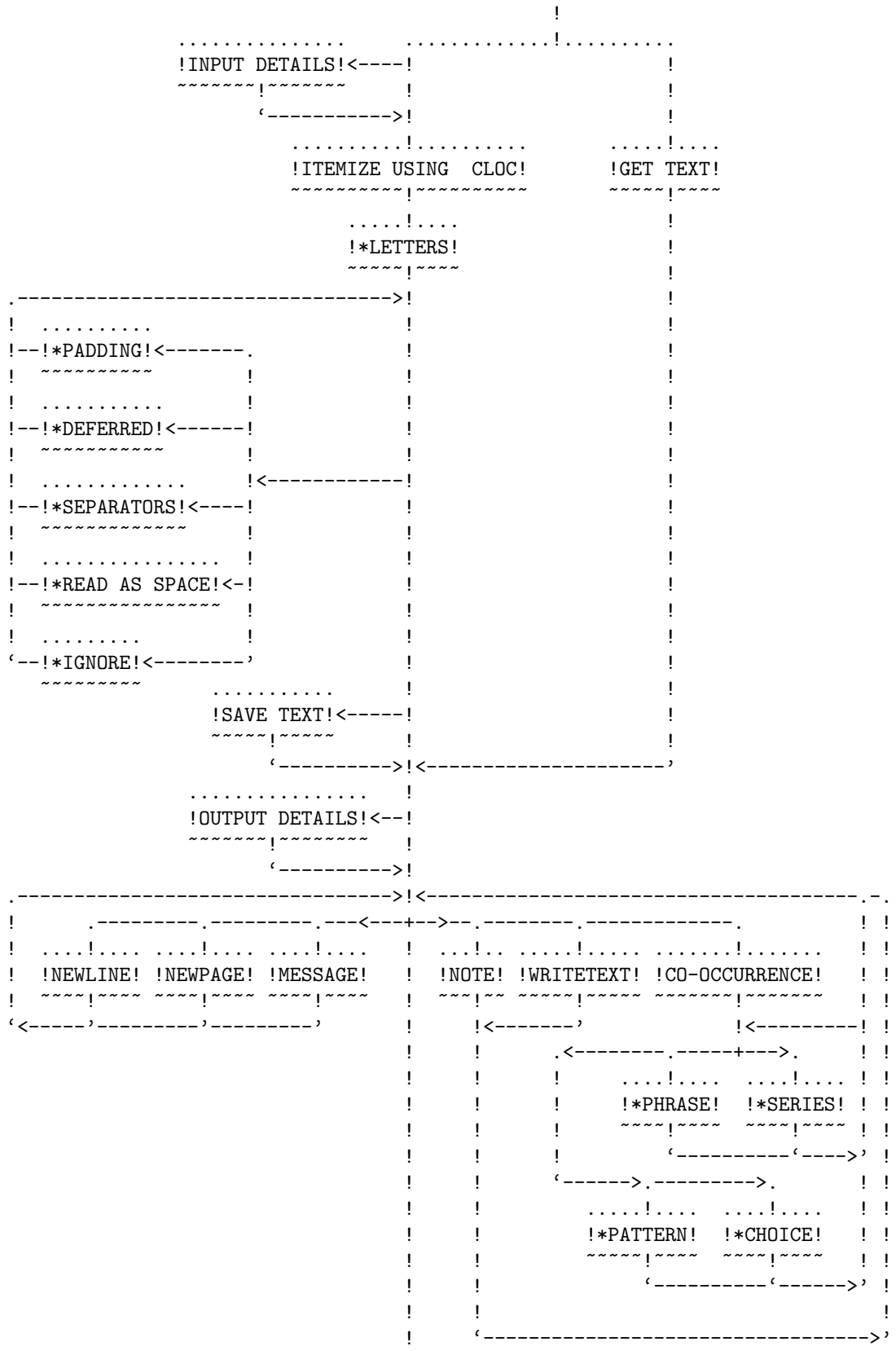
NODE A particular word about which a concordance can be printed or a collocation analysis performed.

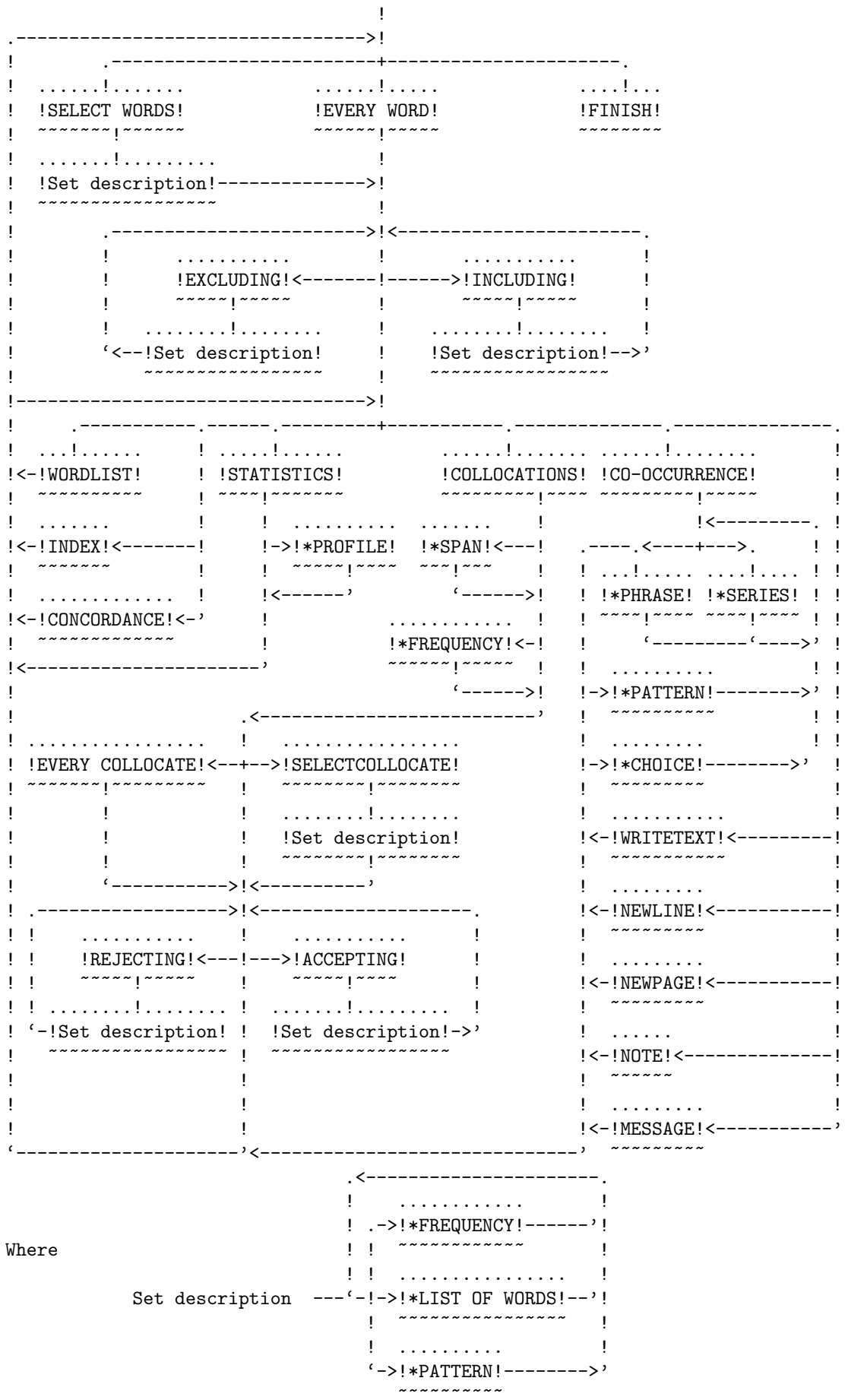
SPAN The context of words, surrounding a NODE, which is used during collocation analysis.

COLLOCATE one of the words of context in a SPAN.

E CLOC Global Syntax Rules

The following 'railroad' diagram describes the syntax rules for CLOC control statements. Follow the arrows from top to bottom and you will pass through all compulsory commands. A diversion of route indicates optional commands. A choice of route indicates a choice of commands at that position.


!INPUT DETAILS!
!ITEMIZE USING CLOC!
!GET TEXT!
!*LETTERS!
!*PADDING!
!*DEFERRED!
!*SEPARATORS!
!READ AS SPACE!
!IGNORE!
!SAVE TEXT!
!OUTPUT DETAILS!
!NEWLINE!
!NEWPAGE!
!MESSAGE!
!NOTE!
!WRITETEXT!
!CO-OCCURRENCE!
!*PHRASE!
!*SERIES!
!*PATTERN!
!*CHOICE!


!SELECT WORDS!
!EVERY WORD!
!FINISH!
!Set description!
!EXCLUDING!
!INCLUDING!
!Set description!
!Set description!
!WORDLIST!
!STATISTICS!
!COLLOCATIONS!
!CO-OCCURRENCE!
!INDEX!
!*PROFILE!
!*SPAN!
!CONCORDANCE!
!*FREQUENCY!
!*PHRASE!
!*SERIES!
!*PATTERN!
!EVERY COLLOCATE!
!SELECTCOLLOCATE!
!*CHOICE!
!Set description!
!WRITETEXT!
!REJECTING!
!ACCEPTING!
!NEWLINE!
!Set description!
!Set description!
!NEWPAGE!
!NOTE!
!MESSAGE!
Where
Set description
!*FREQUENCY!
!*LIST OF WORDS!
!*PATTERN!