CGTM # 91
Charles Holbrow
June 1970

CONC: A SNOBOL4 Program for Generating a Selective Concordance from Short Texts

Introduction

The program CONC will generate easily and flexibly a selective concordance on up to 50 words using relatively short texts. Because the program is written in SNOBOL4, it takes almost 60 seconds of IBM 360/91 CPU time to process a sample of about 15,600 words. Although a more efficient program could be written in machine code, the SNOBOL4 program has the important advantages of being brief, clear, and capable of being run on any of the several types of computers possessing a SNOBOL4 compiler.

A concordance is a dictionary of principal words and immediate context as they appear in a given document. A selective concordance is limited to words selected by the user. To generate a concordance two main activities are necessary:

1. The text must be scanned and compared to the list of selected words;
2. Each successful match must trigger storage of the matched word and the words surrounding it into the right place in an appropriate data structure.

Algorithm

Scanning and comparison with CONC can best be visualized as done with a template and an attached list. The list contains the words chosen by the user for concordancing. These are called select terms. The template is fifteen words long and slides along the string of input text. Over the eighth word of the template is a window. The select terms in the attached list are compared with the word appearing in the window of the template. Figure 1 schematically shows the template. Viewed in these terms the algorithm for generating a concordance is exceedingly simple:

1. Compare the select terms with the word in the window of the template. If there is a match, the 15-word string under the template is stored in a list corresponding to the select term matched. If the end of the text string is detected go to 3.

2. Advance the template one word along the input string and go to 1.

3. Print the lists of concordance strings and halt.

Thus, scanning is done by sliding the template word-by-word along the input string and successively comparing select terms with the word in the template window.

Input

Implementation of this algorithm in SNOBOL4 uses its pattern-matching, concatenation, and data features. The select terms are read in and combined into a pattern used for comparisons made by the SNOBOL4 pattern-matching facility. A successful match occurs when the select term is the same as the leading characters of the word in the window of the template. Thus, the select term "ASTRO" will match "ASTRONOMY" or "ASTROLABE". If upper and lower case are both used, "ASTRO" will not match "astronomy" nor will "astro" match "Astronomy". To account for differences in capitalization, it is necessary to use both "astro" and "Astro" as select terms. Fortunately, the way the lists are named facilitates the use of different forms of select terms, because the lists, defined by a DATA statement, are named by either a select term or a string of select terms.

This last point needs explanation. It may be desirable to generate a single concordance list for a number of closely related select terms, e.g. sun, star, solar, stellar, astro, rather than a single list for each select term. CONC permits the user to do this by the manner in which he prepares the cards from which select terms are read into the computer. All select terms appearing on the same card will be concordanced into a single combined list. The number of cards containing select terms determines the number of lists. To distinguish cards containing select terms from cards containing text, the first card of input data must have punched on it, starting in column 1, the number of cards containing select terms. If we choose, as in our example, to produce a single list then the first card would be

```
1
```

and the second card would be:

```
SUN STAR SOLAR STELLAR ASTRO
```

Alternatively, we might like a concordance which distinguishes the sun from the stars. We would then use as our first input cards

STAR STELLAR ASTRO

SUN SOLAR

2

The ability to build combined lists also means we can easily scan for closely related forms of a single term. For example, we might wish to search on

astro Astro "Astro "astro

1

to be sure of including in our concordance all the equivalent forms of words beginning with the letters a s t r o. (Actually CONC ignores leading double quotation marks, hyphens, and left parentheses.)

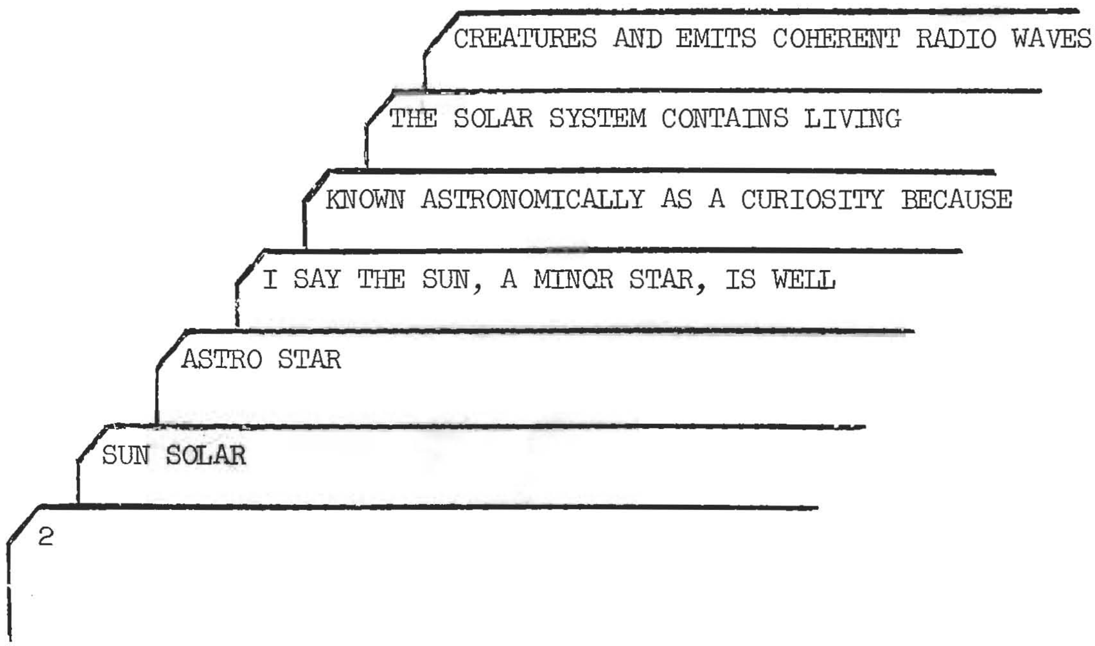
The number of select terms is limited to 50 by an array size specified within the program. This dimension could easily be changed. A more practical limitation is that if a large number of select terms is used, the pattern matching will take unduly long. It would then be good to change the program. After reading in the select terms and naming the corresponding lists, the select terms should be put in alphabetical order so they could be located by a binary search thereby eliminating the need for time consuming sequential searches. Alternatively, if the user has foreknowledge of the frequency with which select terms occur in the test, the matching time can be shortened by putting the most frequently occurring select terms first in the input so they appear early in the sequential search.

The text of the document to be concordanced is read in on cards following the cards listing the select terms. As noted before, the text can be in upper or lower case with any punctuation marks. Each word must be preceded by at least one blank or a double quote. These symbols are used for breaking out individual words from the input string. The only limitation on the length of the text is the running time, which is proportional to the number of words in the text.

Upon completion of the scan, compare, and sort portion of the program, control is shifted to the output portion. The complete contents of the individual lists are printed in the order in which the lists are defined, i.e. in the order in which the select terms are read in. Each list is titled by the select term or combination of select terms used to generate the list. The output is arranged so that the matched word is centered in the page; then the river of white running the length of the page makes it easy to see the principal word.

Example

The conceptual structure and function of CONC are summarized in the following diagram and hand-made example.



CONC generates two lists, called SUN SOLAR and ASTRO STAR. Repeated applications of the template produce four matches as shown in Figure 1. The resulting concordance strings are stored in their corresponding lists to produce the final list structures shown schematically in Fig. 2.

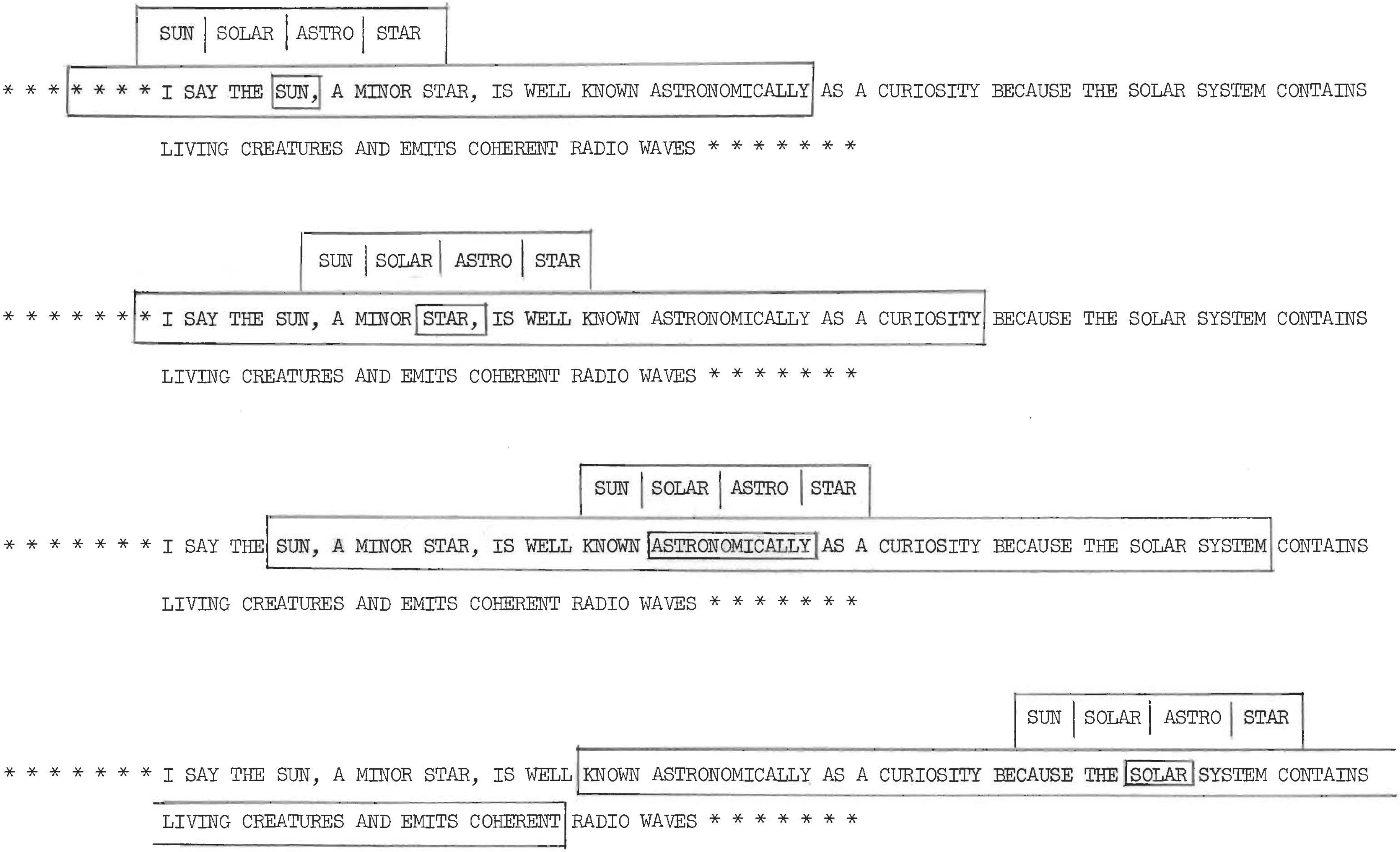


Figure 1. Diagram of four template positions at which match succeeds and the string under the template is stored.

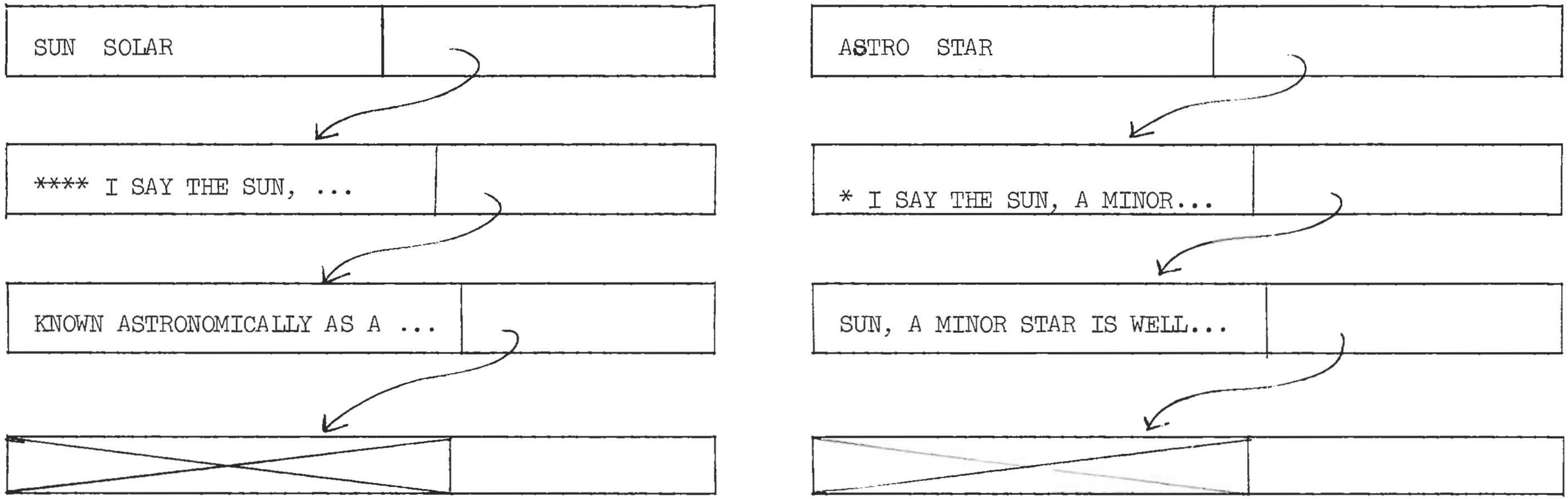


Figure 2. The final list structures generated by CONC from the text and select terms shown in Figure 1

The Program and the Template

With perhaps two exceptions the program is adequately documented by the comments distributed through it. These are given in the appendix. Further explanation of the template and the buffered input is necessary.

The template is constructed as part of the pattern P1. Because P1 is complicated, the pattern and its bead diagram are shown in Figs. 3 and 4 in order to illuminate the following discussion.

```
     P1 = SUCCEED TAB(*STRT)   (@S1  (B | (*GT(S2,SS)  REM) $ STR1
ABORT) @STRT  B B B B B B2 @S2  (PAT $ T2) B B B B B B B B
$ STR  *INSERT(STR,T2,S1,S2)
```

Figure 3. The Pattern P1 which contains the concordancing template

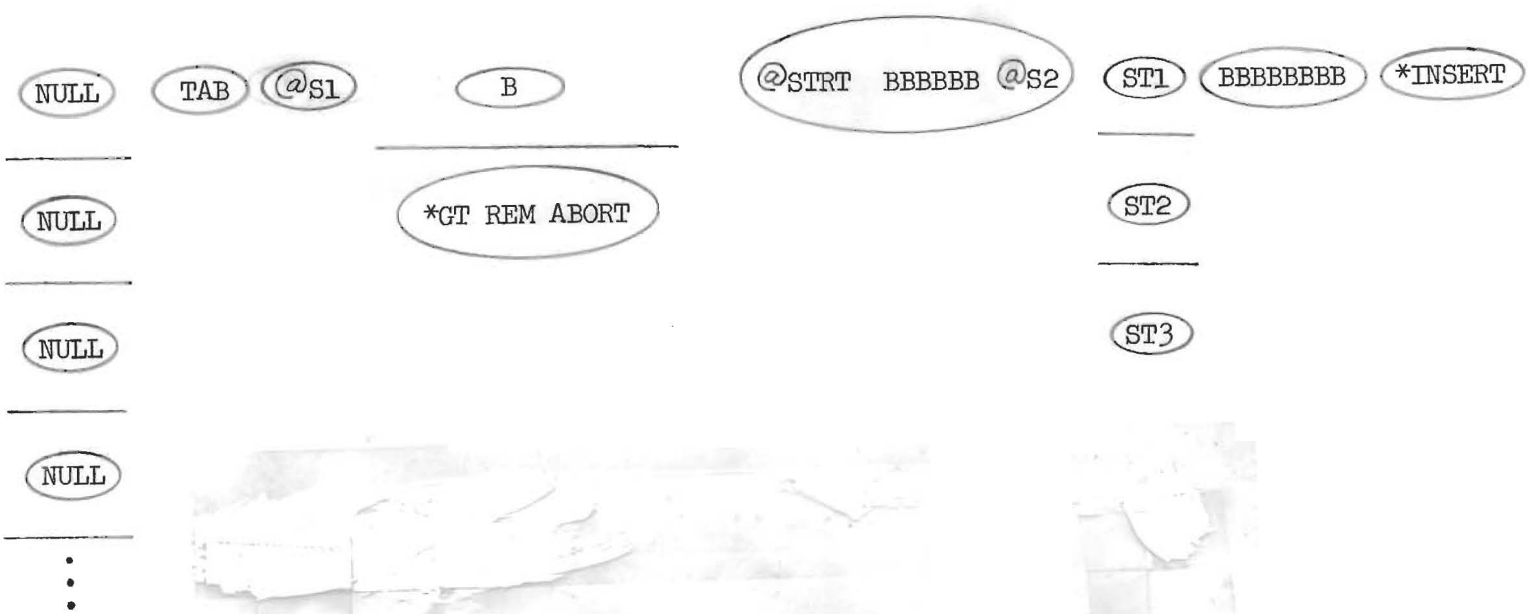


Figure 4. Bead diagram of the template pattern given in Figure 3. Name assignments and Function arguments are not shown.

Let us discuss P1 by reading it from left to right assuming initially that no element on the top row of the bead diagram fails. We will discuss the alternatives later. SUCCEED is null and always matches; we will see later why it is needed. TAB(*STRT) tabs to the position given by STRT. Since STRT initially is null, the template first lines up its leftmost edge at the first word in the input string with the window over the eighth word. To compensate for this, the program adds seven dummy words '* ' to the front of the input string. Thus, the template begins with the window over the first real word of the string.

The pattern B2 uses BREAK(' -') to skip a word and SPAN(' "-(') to ignore blanks, quotes, hyphens and left parentheses. The patterns B = BREAK(' ') SPAN(' ') simply skip from blank to blank; consequently each symbol or string surrounded by blanks is treated as a word. The quantities @S1 and @S2 store the values of the cursor position at the corresponding two points under the template. The values are used to test for the end of the input string and to pad out the concordance string so that it will look nice when printed. @STRT stores in STRT the position of the second word under the template. This is a crucial step, for it is to this point that TAB will move the template the next time it is used. STRT is the parameter by which the template is advanced one word further along the input string.

The pattern PAT contains the select terms which are compared with the word in the window. Assuming the match succeeds, the select term is given the name T2, the eighth word plus 7 more are stepped off by the string of B's, the entire 15 word string under the template is called STR and then STR, T2, S1 and S2 are passed to the function INSERT.

INSERT pads STR with sufficient blanks to center the eighth word in the output string and then stores the string STR in the output list to which T2 points.

Upon return from INSERT, P1 is successfully completed. Success causes reentry into the statement using P1. By means of the value of STRT assigned in the last pass through P1, TAB moves the template one word ahead and proceeds as before.

If, as is likely, PAT fails, the pattern scanner backs up to look for alternatives. The first alternative to be inspected is the pattern (*GT(S2,SS) REM) $ STR1 ABORT. If there is more input to be read, SS is the size of the present input string STR1 less 104 characters. If S2 is greater than SS, i.e. if the template is within 104 characters of the end of STR1, this pattern flushes all of STR1 up to the position given by STRT which leaves seven words plus about S2 characters, and then ABORTS and reads in more input adding it to STR1.

If there is no more input, SS is the size of STR1 less 18 characters. Because the last 17 characters are dummy words to enable the template window to cover the last actual word of the string, SS represents a point on the string past the first letter of this last word. When S2 is greater than SS, all words in the input string have been scanned and the pattern aborts. This time, since there is no more input, the program switches to OUT, where it prints the output lists and halts.

Usually *GT(S2,SS) will fail, thereby driving the pattern scanner back to look for further alternatives. It is important for the functioning of the template that P1's needle back up and come through TAB again. If there were no alternatives this would not happen because P1 would simply fail. SUCCEED provides dummy alternatives to force backing up. SUCCEED acts as an infinite set of NULL alternatives. Consequently P1's needle backs up to SUCCEED and reenters the pattern through the next null alternative. As the needle goes through TAB, the value of STRT assigned in the last pass is used. The effect is to advance the template one word along STR1. Thus PAT compares the select terms with the next word which now appears in the window.

With the template driver used in this program, failure can only occur at or near the end of an input string STR1. Because the IBM/360 implementation of SNOBOL4 limits maximum string length, the input has to be read in as a succession of input strings.

This is done by building each input string successively from the contents of 68 cards plus 104 characters and 7 words from the preceding input string. Assuming 72 characters per card, the resulting string length could in principle exceed 5,000 characters by the length of the 7 words. However, all input is trimmed of trailing blanks as it is read, and there are usually enough of these so that it is extremely unlikely that string overflow will occur.

After a string is built, it is processed by the template. When it has been scanned, a new string is built if there is any more input to be read. If no input remains, the program prints the output lists and halts.

Timing

A succession of runs in which the number of select terms was varied from 1 to 20 shows that on the 360/91, run time increases linearly by about 19 msec for each additional select term.

The scanning algorithm produces a linear search along the string. Comparing the length of time to process a sample of 4900 words with the time to process 15600 words, it was found that run time increases linearly at about 4 msec per word.

The amount of CPU time consumed also depends linearly on the number of successful matches. No measurements were made of this dependence. Under normal circumstances storage time should be only a small portion of the total duration of the run.

Changes and Improvements

The user might wish to increase or decrease the template size. Insertion of removal of a B on the left or the right of PAT will lengthen or shorten the corresponding side of the template.

The present program is easily adapted to tape input. The statement

```
INPUT(.IN,5,72)
```

need only be replaced with

```
INPUT(.IN,8,72)
```

and the appropriate control card added. In OS/360 this would be

```
//FT08F001    DD    DSN= TEXT, DCB=
(RECFM=FB, LRECL=72, BLKSIZE=7272,
BUFNO=2), VOL=SER= * * * * *
```

If more select terms are wanted, one need only change the size of the array T(50). The program has been tested and debugged for 28 select terms. When many of these are used, it is to be expected that not only will the search time in PAT become long, but the storage overhead for the output lists will become large. Probably many select terms and a large text would require buffering of the output to avoid exceeding maximum allowable storage.

The present buffered input is inelegant. The overlap between successive input strings is established assuming that the input is never going to be closer than 7 words to the maximum string length of 4896 characters. Because input is trimmed of trailing blanks and because 7 words represent only about 50 characters, overflow is unlikely. However, it is poor practice to use an algorithm which depends upon the probable form of the input. This portion of the program needs improvement.

For many purposes it would be convenient to print the number of the page from which the concordance string is taken. This could be done at the cost of making P1 more complex and increasing processing time somewhat. Alternatively, the sequence number of the card from which the input is read could be used to identify the text. This would be simpler and faster than page number, but would make the principal document the listing of the cards rather than the text itself.

Summary

1. To run on IBM/360, use customary SNOBOL4 job control language. CONC has been debugged and tested using version 3 of SNOBOL4, but it uses only features also available in version 2. (A sample of JCL suitable for running on the SLAC system is given in the appendix.)

2. Place CONC immediately after the JCL.

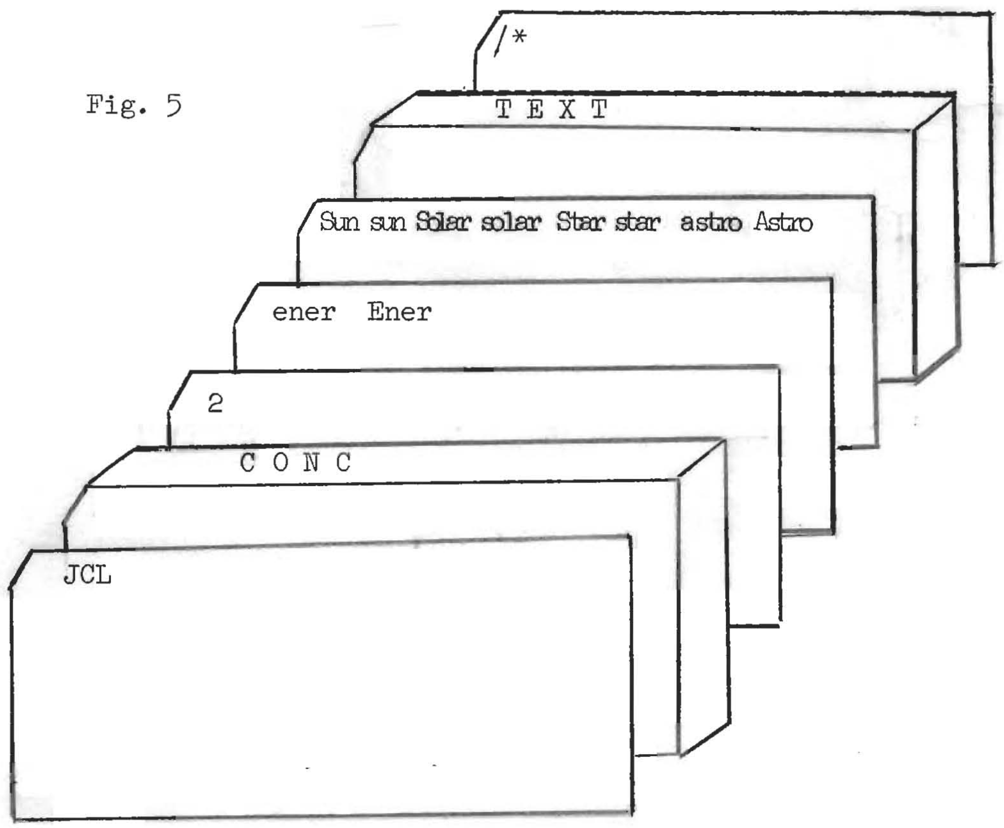
3. The data cards go in as part of the program deck. Do not separate them out with a /*.

a) First data card should be a single integer which is the number of data cards to follow which contain select terms.

b) The next cards contain the select terms. Concordance strings corresponding to the terms on a given card will appear in the same output list. They will be in the order in which they are met in the scan through the text.

c) The rest of the cards are the input text. It may be in any form with any punctuation. Words must be separated by blanks.

Fig. 5



The result is an assemblage of cards as shown above. Using as a text $\sim$1500 words from Henry Adams' essay "A Letter to American Teachers of History" and the depicted input cards, we obtain the sample results shown in the appendix.

Appendix

CONC was applied to $\sim$ 1500 words of Henry Adams' essay "A Letter to American Teachers of History." This material was in upper and lower case so the select terms were chosen to get capitalized as well as uncapitalized words.

There follows a list of the JCL used, a listing of CONC, the input cards and a sample of output. SYSOUT=D was used to print upper and lower case. Consequently output is delayed a day. SYSOUT=A is much faster and entirely satisfactory in most cases.

```
//JOBLIB DD DSN=SYS2.PROGLIB,DISP=(SHR,PASS),UNIT=2314
// EXEC PGM=SNOBOL4
//FT06F001 DD SYSOUT=A,DCB=(LRECL=133,BLKSIZE=3458,BUFNO=2,RECFM=FBA),
//       SPACE=(TRK,(100,50))
//FT05F001 DD *
-LIST LEFT
 INPUT('INPUT',5,72)
* FUNCTION DEFINITIONS, KEYWORD PARAMETERS, CONSTANTS AND ARRAYS
*
        DEFINE('INSERT(STR,T2,S1,S2)')
*            INSERT IS USED IN THE TEMPLATE PATTERN TO PAD OUT A
*            CONCORDANCE STRING WITH BLANKS,PLACE IT IN THE
*            APPROPRIATE OUTPUT LIST, AND CHECK WHETHER ADDITIONAL
*            CARDS NEED TO BE READ.
*
        INPUT(.IN,5,72)
*            THIS STATEMENT FACILITATES ESTABLISHING A LOGICAL INPUT
*            UNIT FOR THE TEXT, DISTINCT FROM THE INPUT UNIT ONTO WHICH
*            THE SELECT TERMS ARE READ.  THEN IF THE USER WANTS TO
*            READ HIS TEXT FROM A TAPE, HE SIMPLY REPLACES THIS
*            STATEMENT WITH
*                                 INPUT(.IN,8,72)
*            AND INSERTS IN THE JCL (FOR OS/360) THE FOLLOWING:
*            //FT08F001 DD DSN=***,DCB=(RECFM=FB,LRECL=72,BLKSIZE=7272
*            BUFNO=2),VOL=SER=********
*
        OUTPUT('PAGE',6,'(A1)')
*            USED IN PRINTING OUTPUT LISTS TO START EACH LIST ON
*            A NEW PAGE.
*
        &ANCHOR = 1
        BLANKS = '                                                          '
*            USED AS A SUPPLY OF BLANKS FOR PADDING CONCORDANCE
*            STRINGS
*
        Q = 'RT5'
*            SETS VALUE OF LABEL SO THAT WHEN ONE INPUT STRING HAS
*            BEEN PROCESSED, MORE INPUT WILL BE READ AND ANOTHER
*            STRING CONSTRUCTED.  ONLY WHEN INPUT FAILS WILL Q BE
*            CHANGED.
*
        T = ARRAY(50)
*            DEFINES ARRAY USED FOR CONSTRUCTING PATTERN CONTAINING
*            SELECT TERMS AND FOR INITIALLY SETTING UP OUTPUT LIST
*            STRUCTURES.
*
        DATA('LIST(NODE,LINK)')
*            DEFINES DATA TYPE FOR CONSTRUCTION OF OUTPUT LIST
*            STRUCTURES.
*
        B2 = BREAK(' -') SPAN(' "-(')
*            B2 IS A PATTERN USING BLANKS OR HYPHENS TO BREAK OUT
*            INDIVIDUAL WORDS FROM ANY GIVEN STRING.  THE CONCORDANCE
*            GENERATOR WILL DETECT PARTS OF HYPHENATED WORDS AS WELL
*            AS WHOLE WORDS.  SPAN CAUSES  CONC TO IGNORE LEADING
*            DOUBLE QUOTATION MARKS OR A LEADING PARENTHESIS.
*
        B = BREAK(' ') SPAN(' ')
*            B SKIPS FROM BLANK TO BLANK. CONSEQUENTLY, ANY PUNCTUATION
```

```
*             MARKS SURROUNDED BY BLANKS WILL BE COUNTED AS SINGLE
*             WORDS. IF B2 WERE USED THROUGHOUT P1, THEN PUNCTUATION
*             MARKS WOULD BE IGNORED.  IF SPAN(' ') WERE REPLACED BY
*             SIMPLY ' ', RUNNING TIME WOULD BE REDUCED BY 10% BUT THEN
*             N MULTIPLE BLANKS WOULD BE READ AS N-1 WORDS OF 1 BLANK
*             EACH.
*
         STR1 = '* * * * * * * '
*             PLACES 7 DUMMY WORDS AT THE FRONT OF THE TEXT INPUT
*             STRING
*
*
* READS IN GROUPS OF SELECT TERMS FROM WHICH PATTERN IS FORMED
*
         P = TRIM(INPUT)
*             P IS THE # OF INPUT CARDS WITH SELECT TERMS ON THEM;
*             TERMS MAY BE GROUPED TOPICALLY ON EACH CARD.
*
         N = 1
*             INITIALIZES N WHICH COUNTS THE NUMBER OF SELECT TERMS
*             AND INDEXES THE ELEMENTS OF THE ARRAY INTO WHICH THEY
*             ARE PLACED.
*
INP1     L1 = LT(L1,P) L1 + 1                                  :F(PAT)
*             L1 COUNTS THE CARDS CONTAINING SELECT TERMS
*
         STR = TRIM(INPUT) ' '
         B1 = STR
*             SAVES STRING OF SELECT TERMS FROM CARD JUST READ TO
*             PLACE IN FIRST CELL OF OUTPUT LIST STRUCTURE TO SERVE
*             AS TITLE IN OUTPUT.
*
*
* BREAKS OUT SELECT TERMS TO BE USED IN MATCHING.
*
RPT1     STR (BREAK(' ') . T<N> SPAN(' ')) =                   :F(RPT4)
*             BREAKS OUT INDIVIDUAL SELECT TERMS FROM THE LAST
*             CARD READ AND PLACES THEM IN THE ARRAY T
*
*
* SETS UP COMMON LIST NAME FOR GROUP OF SELECT TERMS ON EACH CARD
*
         $T<N> = B1
         N = N + 1                                             :(RPT1)
RPT4     $('N' $T<N - 1>) = LIST($T<N - 1>)
         $('H' $T<N - 1>) = $('N' $T<N - 1>)                   :(INP1)
*             MAKES EACH SELECT TERM IN A GIVEN GROUP, AN INDIRECT
*             REFERENCE TO THE SAME LIST STRUCTURE.
*
*
* ITERATIVELY CONSTRUCTS PATTERN TO BE USED IN MATCHING
*
PAT      M = N - 1
         PAT = T<1>
         N = 1
RPT2     N = LT(N,M) N + 1                                     :F(RTO)
         PAT = PAT | T<N>                                      :(RPT2)
*
*
```

```
* CONSTRUCTION OF THE PATTERN WHICH SERVES AS THE TEMPLATE
*
RT0       P1 = SUCCEED TAB(*STRT) (@S1 (B | (*GT(S2,SS) REM) $ STR1
.    ABORT) @STRT B B B B B B2 @S2 (PAT $ T2)   B B B B B B B B) $ STR
.    *INSERT(STR,T2,S1,S2)
*               FOR A FULL EXPLANATION SEE CGTM # 91.  TO INCREASE
*               THE NUMBER OF WORDS INCLUDED IN THE CONCORDANCE STRING,
*               SIMPLY ADD MORE B'S TO THE PATTERN.  A 'B' TO THE RIGHT
*               OF STRT WILL ADD A WORD TO THE LEFT HALF OF THE
*               CONCORDANCE STRING; A 'B' ADDED TO THE SERIES OF 'B'S
*               TO THE RIGHT OF PAT ADDS A WORD TO THE RIGHT HALF OF THE
*               CONCORDANCE STRING.
*
*
* INPUT OF STRING OF TEXT TO BE CONCORDANCED.
*
*               LIMITATION OF STRING SIZE IN IBM /360 IMPLEMENTATION
*               OF SNOBOL4 TO 5000 CHARACTERS, REQUIRES THAT INPUT BE
*               PUT INTO A SEQUENCE OF INPUT STRINGS.
*
RT5       STRT =
*               INITIALIZES CURSOR POSITION TO 0 FOR EACH SUCCESSIVE
*               STRING OF INPUT.
*
*
RT1       STR1 = STR1 TRIM(IN) ' '                        :F(RT4)
          LL = LT(LL,68) LL + 1                           :S(RT1)
*               COUNTS CARDS OF INPUT TEXT.  READS 69 CARDS FIRST CYCLE.
*
          LL = 1
*               RESETS LL TO 1 SO LATER PASSES THROUGH THIS LOOP READ
*               ONLY 68 CARDS.
*
          SS = SIZE(STR1) - 104                           :(RT2)
*               SS IS THE STRING SIZE USED TO TEST WHEN MORE INPUT SHOULD
*               BE READ.  THE 104 ASSURES PROPER OVERLAP BETWEEN THE END
*               OF THIS INPUT STRING AND THE NEXT.  A NEW INPUT STRING
*               WILL BE CONSTRUCTED WHEN THE OLD ONE HAS BEEN SCANNED TO
*               WITHIN APPROXIMATELY 104 CHARACTERS OF ITS END.
*
RT4       STR1 = STR1 '* * * * * * * * * '
*               ADDS DUMMY WORDS TO END OF TEXT SO THE WINDOW OF THE
*               TEMPLATE CAN REACH THE LAST REAL WORD IN THE STRING.
*
*
            Q = 'OUT'
*               WHEN THE INPUT FAILS, Q IS SET TO 'OUT' SO THAT ON THE
*               NEXT FAILURE OF THE TEMPLATE DRIVER, OUTPUT WILL BEGIN.
*
            SS = SIZE(STR1) - 18
*               SS IS THE STRING SIZE LESS 18 TO IGNORE THE 8 DUMMY WORDS
*               PUT ON THE END OF TEXT.  WHEN S2 IS GT SS, THE LAST WORD
*               OF THE TEXT HAS BEEN SCANNED.
*
* SCANNER AND TEMPLATE DRIVER
*
RT2       &FULLSCAN = 1
RT7       STR1 P1                                     :F($Q)S(RT7)
*               DRIVES TEMPLATE ALONG STRING; DOES COMPARISONS; AND BY
```

```
*               THE FUNCTION INSERT, FILLS OUTPUT LISTS, TESTS FOR
*               FURTHER INPUT AND IF NECESSARY PREPARES FOR HALT.
*
*
* PRINTS ALL OUTPUT LISTS IN ORDER
*
OUT         N = 1
            &FULLSCAN = 0
RPT6        N = N + R
*               N IS USED TO INDEX THE ARRAY ELEMENTS WHICH INDIRECTLY
*               REFER TO THE NAMES OF THE INDIVIDUAL LISTS.  BECAUSE
*               SEVERAL SUCCESSIVE ARRAY ELEMENTS MAY REFER TO THE SAME
*               LIST, WE COUNT THIS NUMBER, CALLED R, AFTER EACH LIST
*               IS PRINTED, AND THEN WE INCREMENT N BY R TO FIND THE
*               NEXT ARRAY ELEMENT WHICH POINTS TO ANOTHER LIST.
*
            R = LE(N,M)          :F(END)
*               M IS THE TOTAL NUMBER OF SELECT TERMS REGARDLESS OF
*               GROUPING.  WHEN N IS INCREMENTED BEYOND M, OUTPUT IS
*               COMPLETE AND THE PROGRAM ENDS.  THIS STATEMENT ALSO
*               SETS R TO 0 PREPARATORY TO COUNTING THE NUMBER OF SELECT
*               TERMS IN THE LIST
*
            PAGE = '1'
*               TURNS A PAGE BEFORE STARTING TO PRINT A NEW LIST
*
            $('N' $T<N>) = $('H' $T<N>)
RPT5        OUTPUT = NODE($('N' $T<N>))
            OUTPUT =
            $('N' $T<N>) = LINK($('N' $T<N>))
            IDENT($('N' $T<N>))                              :F(RPT5)
*               THE LIST IS CALLED AND ITS CONTENTS PRINTED SKIPPING
*               EVERY OTHER LINE, UNTIL AN EMPTY CELL IS FOUND IN THE
*               LIST.
*
*
* COUNTS # OF SELECT TERMS IN TOPICAL GROUP
*
RPT7        $T<N> B =                                       :F(RPT6)
            R = R + 1                                   :(RPT7)
*               EVALUATES R FOR THE LAST LIST PRINTED
*
*
* THIS FUNCTION IS USED BY THE TEMPLATE TO PAD CONCORDANCE STRINGS WITH
* BLANKS SO THE OUTPUT LOOKS NICE, TO ASSIGN THESE STRINGS TO THE
* PROPER OUTPUT LIST, AND TO TEST IF MORE TEXT NEEDS TO BE READ, AND
* WHETHER ALL THE TEXT HAS BEEN READ.
*
INSERT      RR = 60 + S1 - S2
*               CALCULATES THE NUMBER OF BLANKS FOR PADDING
*
*
RT6         RR = LT(RR,0) 0
            BLANKS LEN(RR) . LPAD
            STR = LPAD STR
*               CHECKS TO BE SURE NUMBER OF PADDING BLANKS IS NOT LESS
*               THAN 0, THEN BREAKS OUT RR BLANKS AND ADDS THEM TO THE
*               CONCORDANCE STRING.
*
```

```
        LINK($('N' $T2)) = LIST(STR)
        $('N' $T2) = LINK($('N' $T2))                          :(RETURN)
*            THE PADDED CONCORDANCE STRING IS THEN STORED IN THE
*            APPROPRIATE LIST.
*
*
END
```

2
ener Ener
sun Sun solar Solar space planet eart Eart astro Astro
)U The Problem)ULL

)P The mechanical theory of the universe governed physical science for three hundred years. Directly succeeding the theological scheme of a universe existing as a unity by the will of an infinite and eternal Creator, it affirmed or assumed the unity and indestructibility of Force or Energy, as a scientific dogma or Law, which was called the Law of the Conservation of Energy. Under this Law the quantity of matter in the universe remained invariable; the sum of movement remained constant; energy was indestructible; "nothing was added; nothing was lost;" nothing was created, nothing was destroyed.

)P Towards the middle of the nineteenth century, - that is, about 1850, - a new school of physicists appeared in Europe, dating from an Essay on the Motive Power of Heat, published by Sadi Carnot in 1824, and made famous by the names of William Thomson, Lord Kelvin, in England, and of Clausius and Helmholz in Germany, who announced a second law of dynamics. The first law said that Energy was never lost; the second said that it was never saved; that, while the sum of energy in the universe might remain constant, - granting that the universe was a closed box from which nothing could escape, - the higher powers of energy tended always to fall lower, and that this process had no known limit.

)P The second law was briefly stated by Thomson in a paper "On a Universal Tendency in Nature to the Dissipation of Mechanical Energy," published in October, 1852, which is now as classic as Kepler's or Newton's Laws, and quite as necessary to a scientific education. Quoted exactly from Thomson's "Mathematical and Physical Papers" (Cambridge, 1882, Vol. I, p. 514), the Law of Dissipation runs thus: -

)P "1. There is at present in the material world a universal tendency to the dissipation of mechanical energy.

)P "2. Any restoration of mechanical energy, without more than an equivalent of dissipation, is impossible in inanimate material processes, and is probably never effected by means of organized matter, either endowed with vegetable life or subjected to the will of an animated creature.

)P "3. Within a finite period of time past, the earth must have been, and within a finite period of time to come, the earth must again be, unfit for the habitation of man as at present constituted, unless operations have been, or are to be performed, which are impossible under the laws to which the known operations going on at present in the material world, are subject."

)P When this young man of twenty-eight thus tossed the universe into the ash-heap, few scientific authorities took him seriously; but after the first gasp of surprise physicists began to give him qualified support which soon became absolute. "This conclusion made much noise," says Ostwald ("L'Energie," Paris, 1910); "the more because Helmholz and Clausius gave in their adherence to it. We owe to the latter the following formula: 'The Entropy of the Universe tends toward a maximum.'" To physicists, this law of Entropy became "a prodigiously abstract conception, according to the familiar phrase of M. Poincare; but to the vulgar and ignorant historian it meant only that the ash-heap was constantly increasing in size; while the public understood little and cared less about Entropy, and the literary class knew only that the Newtonian universe, in which they had been cradled, admitted no loss of energy in the solar system, where the planets, at the end of their planetary years, returned exactly to their positions at the beginning. Gravitation showed no waste of energy whatever, except where friction occurred, but had planets gone off like comets, and never returned, the scholar of 1860 would still have feared to question the scientific dogma which asserted resolutely, without qualification, the fact that nothing in nature was lost. If no other assurance had satisfied him, all doubts were silenced by the famous outburst of eloquence with which Tyndall concluded his Lecture in 1862, on "Heat as a Mode of Motion." Old men can still recall how, after explaining that "the quantity of the solar heat intercepted by the earth is only 1/2,300,000,000 of the total radiation," Tyndall refrained from telling what became of the heat not intercepted by the earth, and went on to expatiate with enthusiasm on the unity of the universe and its energy: -

)P "Look at the integrated energies of our world, - the stored power of our coalfields; - our winds and rivers; - our fleets, armies and guns! What are they? They are all generated by a portion of the sun's energy which does not amount to 1/2,300,000,000 of the whole. This , in fact is the entire fraction of the sun's force intercepted by the earth, and in reality we convert but a small fraction of this fraction into mechanical energy. Multiplying all our powers by millions of millions, we do not reach the sun's expenditure. And, still, notwithstanding this enormous drain, in the lapse of human history we are unable to detect a diminution of his store. Measured by our largest terrestrial standards, such a reservoir of power is infinite; but it is our privilege to rise above these standards, and to regard the sun himself as a speck in infinite extension, - a mere drop in the universal sea. We analyse the space in which he is immersed, and which is the vehicle of his power. We pass to other systems and other suns, each pouring forth energy like our own, but still without infringement of the law which reveals immutability in the midst of change, which recognises incessant transference and conversion, but neither final gain nor loss. This law generalises the aphorism of Solomon, that there is nothing new under the sun, by teaching us to detect everywhere, under its infinite variety of appearances, the same primeval force. To nature nothing can be added; from nature nothing can be taken away; the sum of her energies is constant, and the utmost man can do in the pursuit of physical truth, or in the application of physical knowledge, is to shift the constituents of the never-varying total, and out of one of them to form another. The law of conservation rigidly excludes both creation and annihilation. Waves may change to ripples and ripples to waves, - magnitude may be substituted for number, and number for magnitude, - asteroids may aggregate to suns, suns may resolve themselves into florae and faunae, and florae and faunae melt in air, - the flux of power is eternally the same. It rolls in music through the ages, and all terrestrial energy, - the manifestations of life as well as the display of phenomena, are but the modulations of its rhythm."

)P This magisterial tone irritated some of the new physicists to the point of hinting that Tyndall deliberately misstated the facts of physics, for fear lest some one should drive him into a logical snare, ending in the necessity of admitting a Creation. In flat contradiction to Tyndall, Kelvin and Tait affirmed that "the same primeval force" could never be detected, - much less recovered; that all nature's energies were slowly converting themselves into heat and vanishing in space, until, at the last, nothing would be left except a dead ocean of energy at its lowest possible level, - say of heat at 1 degree Centigrade, or -272 degrees C. below the freezing point of water, - and incapable of doing any work whatever, since work could be done only by a fall of tension, as water does work in falling to sea-level.

)P Between such authorities the unscientific student could not interfere. Naturally, all his sympathies were with Tyndall. The idea that the entire sidereal universe could have gone on for eternity dissipating energy, and never restoring it, seemed, at the least, unreasonable; while the astronomers drew up lists of nebulae by hundreds in the very act of generating universes, and the geologists showered the theory with rocks in order to show that the sun had already reached an age many times greater than Thomson was willing to allow it.

)P No one knew, although every one explained what had caused the inequalities of energy; least of all could the historian of human society assert or deny that energy could be created or could not be destroyed. The subject was beyond his province. Since the Church had lost its authority, the historian's field had shrunk into narrow limits of rigorously human action; but, strictly within those limits, he was clear that the energy with which history had to deal could not be reduced directly to a mechanical or physico-chemical process. He was therefore obliged either to deny that social energy was an energy at all; or to assert that it was an energy independent of physical laws. Yet how could he deny that social energy was a true ener Ener

the unity and indestructibility of Force or Energy, as a scientific dogma or Law, which
called the Law of the Conservation of Energy. Under this Law the quantity of matter
invariable; the sum of movement remained constant; energy was indestructible; "nothing was added; nothing was
of dynamics. The first law said that Energy was never lost; the second said that
never saved; that, while the sum of energy in the universe might remain constant, -
could escape, - the higher powers of energy tended always to fall lower, and that
in Nature to the Dissipation of Mechanical Energy," published in October, 1852, which is now
universal tendency to the dissipation of mechanical energy.)P "2. Any restoration of mechanical energy,
energy.)P "2. Any restoration of mechanical energy, without more than an equivalent of dissipation,
had been cradled, admitted no loss of energy in the solar system, where the planets,
the beginning. Gravitation showed no waste of energy whatever, except where friction occurred, but had
the unity of the universe and its energy: -)P "Look at the integrated energies
energy: -)P "Look at the integrated energies of our world, - the stored power
generated by a portion of the sun's energy which does not amount to 1/2,300,000,000 of
small fraction of this fraction into mechanical energy. Multiplying all our powers by millions of
systems and other suns, each pouring forth energy like our own, but still without infringement
be taken away; the sum of her energies is constant, and the utmost man can
music through the ages, and all terrestrial energy, - the manifestations of life as well
- much less recovered; that all nature's energies were slowly converting themselves into heat and
be left except a dead ocean of energy at its lowest possible level, - say
could have gone on for eternity dissipating energy, and never restoring it, seemed, at the
explained what had caused the inequalities of energy; least of all could the historian of
of human society assert or deny that energy could be created or could not be
those limits, he was clear that the energy with which history had to deal could
therefore obliged either to deny that social energy was an energy at all; or to
to deny that social energy was an energy at all; or to assert that it
or to assert that it was an energy independent of physical laws. Yet how could
Yet how could he deny that social energy was a true * * * *

sun Sun solar Solar space planet eart Eart astro Astro

a finite period of time past, the earth must have been, and within a finite

finite period of time to come, the earth must again be, unfit for the habitation

admitted no loss of energy in the solar system, where the planets, at the end

energy in the solar system, where the planets, at the end of their planetary years,

the planets, at the end of their planetary years, returned exactly to their positions at

whatever, except where friction occurred, but had planets gone off like comets, and never returned,

after explaining that "the quantity of the solar heat intercepted by the earth is only

of the solar heat intercepted by the earth is only 1/2,300,000,000 of the total radiation,"

of the heat not intercepted by the earth, and went on to expatiate with enthusiasm

all generated by a portion of the sun's energy which does not amount to 1/2,300,000,000

fact is the entire fraction of the sun's force intercepted by the earth, and in

of the sun's force intercepted by the earth, and in reality we convert but a

of millions, we do not reach the sun's expenditure. And, still, notwithstanding this enormous drain,

above these standards, and to regard the sun himself as a speck in infinite extension,

in the universal sea. We analyse the space in which he is immersed, and which

We pass to other systems and other suns, each pouring forth energy like our own,

that there is nothing new under the sun, by teaching us to detect everywhere, under

for magnitude, - asteroids may aggregate to suns, suns may resolve themselves into florae and

magnitude, - asteroids may aggregate to suns, suns may resolve themselves into florae and faunae,

converting themselves into heat and vanishing in space, until, at the last, nothing would be

seemed, at the least, unreasonable; while the astronomers drew up lists of nebulae by hundreds

rocks in order to show that the sun had already reached an age many times

SNOBOL4 STATISTICS SUMMARY-

```
     1048 MS. COMPILATION TIME
     4077 MS. EXECUTION TIME
      966 STATEMENTS EXECUTED,     111 FAILED
      284 ARITHMETIC OPERATIONS PERFORMED
      130 PATTERN MATCHES PERFORMED
        4 REGENERATIONS OF DYNAMIC STORAGE
      138 READS PERFORMED
      106 WRITES PERFORMED
     4.22 MS. AVERAGE PER STATEMENT EXECUTED
```