RECOMP II USERS' PROGRAM NO. 1090

PROGRAM TITLE: SEQUENTIAL ANALYSIS PROGRAM

PROGRAM CLASSIFICATION: Service Routine

AUTHOR: R. C. Haygood
                                    Autonetics

PURPOSE: To tally pairs of events
                                    occurring in sequence.

DATE: 28 June 1961

Published by

RECOMP Users' Library

at

AUTONETICS INDUSTRIAL PRODUCTS
A DIVISION OF NORTH AMERICAN AVIATION, INC.
3400 E. 70th Street, Long Beach 5, Calif.

PROGRAM TITLE: SEQUENTIAL ANALYSIS PROGRAM

- -

1.   INTRODUCTION

When dealing with a limited population of events, such as the
sequence of letters in written English, or the sequence of use
of controls on equipment, it is often desirable to find out how
often a specific event follows another specific event. In
written language, the occurrence of a letter typically imposes
a constraint on the letters which follow. For example, "Q"
must be followed by "U", and the second letter following must
be "A", "E", "I", or "O". This constraint is called sequential
redundancy. A precise quantification of sequential redundancy
is useful in communication studies, particularly those which deal
with transmission under high noise levels.

Analysis of the sequence of use of controls and displays is useful
in design of a control panel, especially where the components
must be scattered over a large area. If the use of control $\underline{A}$
typically is followed by the use of control $\underline{B}$ (a sequential link),
usually it is desirable to have $\underline{A}$ and $\underline{B}$ close together. On the
other hand, if $\underline{A}$ is never followed by $\underline{B}$, then their relative
positions may be unimportant. The evaluation of sequential
links between two components is part of what is called a link
analysis. When the operating sequence is short, say one or two
hundred operations, the tally of links may me made with paper
and pencil without difficulty. However, when there are several
thousand operations in the sequence, a manual tally may not be
feasible. Therefore, a RECOMP II program was developed.

The program has been used extensively, both in determining
sequential redundancy in various types of English writing, and
in a link analysis of a major item of Autonetics equipment.

2.   RESTRICTIONS

The total number of different events may not exceed 28. The
sequence is entered manually through the typewriter, and the rate
of typing may not exceed 5 characters/sec. A faster rate will
give an overflow stop. The length of sequence is virtually un-
limited (10^4 for the most common pairing, which may be increased
to 10^{10} if desired).

3.   METHOD

The program is entered in the loops and control is transferred
to the typewriter to accept one character. When the character
is typed, it is entered at b 39, shifted to b 38, and the
preceding character is added back in at b 32. The pair is used

PROGRAM TITLE: SEQUENTIAL ANALYSIS PROGRAM

- -

to create a retrieval and storage command, and the pair itself
is stored for later use. The storage command sequence adds a
1 in the location representing the pair. The pair then is
brought back in, the character at b 32 erased, and the character
just typed (at b 38) is shifted left and stored at b 32 scale.
Then control is shifted back to the typewriter for another
character.

4.   USAGE

4.1   Initial conditions.

          Main memory cleared to negative zeros, typewriter set to letter
          shift, output in computer/punch, sense switches and transfer
          stop off, mode continuous, program tape filled and verified,
          location set to 6675.0, and start.

4.2   Type the sequence of letters.

4.3   When the entire sequence has been typed in, the operator
          restores control to the console, sets location to the beginning
          of the print-out routine (6716.0) and starts. The computer
          then examines each location up through 4141.0 for non-zero
          (positive) contents. When non-zero contents are found, they
          are converted to BCD format and printed. Then the location of
          the number is printed out as an alpha pair.

5.   CODING INFORMATION

5.1   Timing

          Input proceeds at operator's own rate. The writer's rate in
          typing has been about 100 characters/min. Output takes between
          5 and 28 minutes, depending on how many locations are filled.
          In adult English, approximately 400-500 locations are used,
          depending on length of sample, and print-out takes about 15 min.

5.2   Locations Used

          $6675_8 - 6754_8$

PROGRAM TITLE: SEQUENTIAL ANALYSIS PROGRAM
- -

TEST LINK ANALYSIS ROUTINE
<u>---</u>

ACRVFVWKUWXYZJVYZQSDHVIVQSGBEF   CRVFVWKUWXYZJVYZSHVSIVS

0001.A
0001.EF
0001.AC
0002.
0001. C
0001.S
0001.SI
0001.SD
0001.SH
0001.SG
0002.IV
0002.UW
0001.DH
0002.RV
0002.JV
0001.F
0002.FV
0002.CR
0002.KU
0001.ZS
0002.ZJ
0001.ZQ
0002.WK
0002.WX
0002.HV
0004.YZ
0002.QS
0001.BE
0001.GB
0002.XY
0002.VS
0001.VI
0002.VF
0002.VW
0002.VY
0001.VQ

Each letter represents one component (control or display) on the control
panel under study. This is a relatively short operating sequence, in-
volving only 53 steps.

Print-out time requirements are 140 seconds plus 2 seconds for each
location which contains a non-zero entry. The BLANK/A entry at the be-
ginning is discarded.

PROGRAM TITLE: SEQUENTIAL ANALYSIS PROGRAM

- -

```
┌─────────────┐                              ┌ ─ ─ ─ ─ ─ ─ ─ ┐      ┌─────────────┐
│Put program  │                              │Change control │      │Halt         │
│in computer  │                              │to print-out   │      │computer     │
│and start    │                              │routine        │      │             │
└─────────────┘                              └ ─ ─ ─ ─ ─ ─ ─ ┘      └─────────────┘
       │                                             │
       ▼                                             ▼                  Program finished
┌─────────────┐         Operator interrupts  ┌─────────────┐     Program
│Put input    │         program when all     │Examine each │◄──  not finished
│routine in   │         letters are typed in │data location│◄─────────
│high-speed   │                              │for non-zero │
│loops        │                              │contents     │
└─────────────┘                              └─────────────┘
       │                                             │
       ▼                                      Location is      Location is
┌─────────────┐                              positive (non-   negative
│Change control│◄─────────                   zero             (zero)
│to typewriter │                                    │
│and IDLE      │                                    ▼
└─────────────┘                              ┌─────────────┐
       │                    ┌─────────────┐  │Store location│
       ▼                    │Shift present│  │number        │
┌ ─ ─ ─ ─ ─ ┐              │letter to    │  └─────────────┘
│Type one    │             │become previous│       │
│letter      │             │letter & store │       ▼
│(present    │             └─────────────┘  ┌─────────────┐
│    letter) │                    ▲          │Convert data │
└ ─ ─ ─ ─ ─ ┘              ┌─────────────┐  │location     │
   Automatic               │Add letter   │  │contents to  │
   conversion to           │pair back in,│  │output format│
   number                  │erase previous│  └─────────────┘
       │                   │letter        │         │
       ▼                   └─────────────┘         ▼
┌─────────────┐                   ▲          ┌─────────────┐
│Retrieve     │            ┌─────────────┐  │Print total  │
│previous     │            │Store new sum│  │for this     │
│letter       │            │for this     │  │location     │
└─────────────┘            │letter pair  │  └─────────────┘
       │                   └─────────────┘         │
       ▼                          ▲                ▼
┌─────────────┐            ┌─────────────┐  ┌─────────────┐
│Use both     │            │Add one to   │  │Add in locat-│
│letters to   │            │old sum      │  │ion number and│
│create storage│           └─────────────┘  │print-out as │
│command      │                   ▲          │letter pair  │
└─────────────┘                              └─────────────┘
       │                                             │
       ▼                                             ▼
┌─────────────┐            ┌─────────────┐  ┌─────────────┐  ┌─────────────┐
│Use both     │───────────►│Retrieve old │  │Add one to   │─►│Check for    │
│letters for  │            │sum for this │  │location     │  │end of data  │
│retrieval    │            │letter pair  │  │number and   │  │locations    │
│command      │            └─────────────┘  │store        │  └─────────────┘
└─────────────┘                             └─────────────┘
```