Technical Note

Fuzzy Query Language

Robert E. La Quey
San Diego, California

Introduction

The purpose of this technical note is to provide a simple model demonstrating one use of fuzzy sets in FORTH.

Fuzzy sets have become an academic industry. My intent is not to try to review the products of this industry, but to point to their existence and demonstrate that they are easily imported into FORTH. The seminal work is by Zadeh [1978].

Modern mathematics is built up from set theory. Thus, if one redefines the fundamental notion of a set, then mathematics in general needs reconstruction. The proposition has the same profound implications for mathematics as the redefinition of `@` and `!` has for FORTH.

Zadeh pointed out that classical set theory was rather fascist in its insistence that an element must either belong (1) or not belong (0) to a set. He introduced the idea of a "fuzzy" set which was characterized by allowing elements to have a degree of membership, i.e. a value between 0 and 1, in the set.

A fuzzy set is described by a characteristic function. I chose arbitrarily to let this value vary from 0 to 100. An element that is certainly in the set has a characteristic value 100. An element that is certainly not in the set has a value 0. In between, the value is between 0 and 100. Scaling serves to map the real world into this range.

An Implementation

Screen 9 provides a few words for dealing with this scaling. The fuzzy set itself is represented by a table which allows one to determine the degree of membership given any element. Screen 10 implements this table. Screen 11 associates a set of conventional flags with ranges of values in the table.

Much of this is arbitrary and should be designed to fit a specific problem. For many problems, two or three levels between the limits of **`ALWAYS CERTAIN`** and **`NEVER`** are probably adequate. On the other hand artists probably discriminate a much wider range of REDs than do most accountants.

Entire books have been written on the first few lines of Screen 12. The interesting fact seems to be that there are many different ways of choosing **`AND OR NOT`** which preserve De Morgan's Theorem. (See Charles Moore's digital simulator for instance [MOO84].) The choice presented here is perhaps the simplest and is rather intuitive.

Note that this text is sprinkled with words like much, many, few, probably, most, rather, etc. These are the words that fuzzy sets are aimed at. In Screen 13 we provide a word `.F` which attempts to capture some of this flavor.

A toy application has been included to give the reader a feel for fuzzy sets. One has:

Application = Database + Fuzzy Sets + Query Language.

Screens 18 to 20 lay down a simple database in RAM. Screen 21 defines some fuzzy sets and illustrates how easily they may be combined.

Screens 22 through 25 define a Query Language. The syntax is more like FORTH than English, but the semantics is like English. If you want English-like syntax, then use a syntactical parser [MOR85] on the input stream. One might consider this use of fuzzy sets as an example of how to implement one aspect of a semantic parser (PAR85).

Finally Screens 26 and 27 contain examples of queries that are supported by the fuzzy sets and the use of the query language.

Bibliography

1. Zadeh, L. A. "Fuzzy Sets as a Basis for a Theory of Possibility," *Fuzzy Sets and Systems,* **1**:3-28, 1978.

2. Moore, Charles. "A Language for Digital Design," paper presented at the 1984 FORML Conference, Asilomar, California, November 23-25, 1984.

3. Morgenstern, Leonard. "BNF: A Parser Written in Forth," paper presented at the 1985 FORML Conference, Asilomar, California, November 29-December 1, 1985.

4. Park, Jack. "An Approach to Natural Language Parsing," paper presented at the 1985 FORML Conference, Asilomar, California, November 29-December 1, 1985.

In memoriam: Michael Reagan 1961-1985

Few are wholly dead
Blow on a dead man's embers
And a live flame will start.
Robert Graves

Mike introduced me to fuzzy logic. He was a good FORTH programmer, and he was human.

Manuscript received June 1986.

Appendix

The following source code is written in a variant of Laxen-Perry F83. It is placed in the public domain and may be modified as needed. I would appreciate hearing about interesting improvements users may implement.

```
Scr # 1
( load screen )

  2 LOAD  ( FORTH Extensions     )
  9 LOAD  ( Fuzzy set Primitives )
 18 LOAD  ( Application = Database + Fuzzy Sets + Query Lang)
 26 LOAD  ( Queries )
```

```
Scr # 2
( FORTH EXTEN   6/22/85 rel )

 FORTH DEFINITIONS

 : KQ KEY 27 = IF QUIT THEN ;
 : PRINTER [ HIDDEN ] PR-START ; FORTH
 : CONSOLE [ HIDDEN ] PR-STOP  ; FORTH
 \ HIDDEN is a vocabulary - for some implementations, PRINTER may
 be defined as PRINTING ON.
 VARIABLE REPEAT?
 : Begin 1 REPEAT? ! >IN @ ;

 : Until ( >IN,Flag --- >IN )
      IF DUP >IN ! ELSE DROP Ø REPEAT? ! THEN ;     EXIT

 Latter constructs allow interpretive looping
```

```
Scr # 9
( Fuzzy Sets   6/22/85 rel )

 VARIABLE SET^

 : MINX    SET^  @ 2+  @;                         ( --- Minvalue      )
 : MAXX    SET^  @     @;                         ( --- Maxvalue      )
 : DX      MAXX  MINX - ;                         ( --- Delta val     )
 : POINT   SET^  @ 4 +  ;                         ( --- Addr Deg Table)
 : SCALE   10  DX */ 2* ;                         ( Value --- Index   )
 : MINDEG  POINT       @;                         ( --- Deg at Minval )
 : MAXDEG  POINT 20 + @ ;                         ( --- Deg at Maxval )
 : DEGREE  MINX -   SCALE  POINT + @ ;            ( Value --- Degree  )
   -->
  These words provide support for computing the Degree of
  Membership in a fuzzy set associated with the value of an
  element, i.e. the characteristic function of the fuzzy set.
```

```
Scr # 10
( Fuzzy Sets    6/22/85 rel )

: FUZZY.SET    ( Table, Minvalue, Maxvalue --- )
    CREATE 13 0 DO   , LOOP
    DOES> SET^ !             ( Value --- Degree of Membership)
          DUP MINX <
          IF DROP MINDEG
          ELSE DUP MAXX >
               IF  DROP MAXDEG ELSE DEGREE THEN
          THEN ;                                   -->

At compile time a Table containing 11 elements representing the
degrees of membership in the fuzzy set is placed on the stack
followed by the minimum and maximum values of the input value
of a candidate for membership.  At run time a value is input on
the stack and a degree of membership is returned.
```

```
Scr # 11
( Fuzzy Sets    6/22/85 rel )

: ALWAYS?         98 100 BETWEEN ;     ( --- Flag )
: ALMOST.ALWAYS?  85  97 BETWEEN ;     ( --- Flag )
: VERY.OFTEN?     69  84 BETWEEN ;     ( --- Flag )
: OFTEN?          60  68 BETWEEN ;     ( --- Flag )
: UNSPECIFIC+?    50  59 BETWEEN ;     ( --- Flag )
: UNSPECIFIC-?    40  49 BETWEEN ;     ( --- Flag )
: SELDOM?         31  39 BETWEEN ;     ( --- Flag )
: VERY.SELDOM?    15  30 BETWEEN ;     ( --- Flag )
: ALMOST.NEVER?    2  14 BETWEEN ;     ( --- Flag )
: NEVER?           0   1 BETWEEN ;     ( --- Flag )

 -->
 Membership tests.  If Degree of Membership is between 85 and 97
 then we say something is " almost always " true.
```

```
Scr # 12
( Fuzzy Sets   6/22/85 rel )

 100  CONSTANT CERTAIN

: Or  MAX ;    ( Think about it!  Many other choices are      )
: And MIN ;    ( possible and may be desirable on occasion.   )
: Not CERTAIN SWAP - ;

( Some useful tables )
: SAME         100 90 80 70 60 50 40 30 20 10   0 ;
: MORE         100 81 64 49 36 25 16  9  4  1   0 ;
: LESS         100 95 89 85 78 71 65 55 45 32   0 ;

: INCREASING   100 92 77 65 55 45 36 23  8  2   0 ;
: DECREASING     0  1  8 23 36 45 55 65 77 92 100 ;
: LOCAL          0  2 16 46 72 98 98 72 46 16   0 ;  -->
```

```
Scr # 13
( Fuzzy Sets    6/22/85 rel )

 : DE DROP EXIT ;

 : .F ( Degree ---- )
     DUP ALWAYS?         IF ." always "              DE   THEN
     DUP ALMOST.ALWAYS?  IF ." almost always "       DE   THEN
     DUP VERY.OFTEN?     IF ." very often "          DE   THEN
     DUP OFTEN?          IF ." often "               DE   THEN
     DUP UNSPECIFIC+?    IF ." occasionally "        DE   THEN
     DUP UNSPECIFIC-?    IF ." probably would not "  DE   THEN
     DUP SELDOM?         IF ." seldom "              DE   THEN
     DUP VERY.SELDOM?    IF ." very seldom "         DE   THEN
     DUP ALMOST.NEVER?   IF ." almost never "        DE   THEN
     DUP NEVER?          IF ." never "               DE   THEN ;
     -->
```

```
Scr # 14
( Fuzzy Sets    6/22/85 rel )
 VARIABLE 'FUZZY
 : SET_FUZZY ' DUP ' FUZZY ! >BODY SET^ ! ;
 : FUZZY 'FUZZY @ EXECUTE ;
 : .FUZZY_NAME 'FUZZY @ >NAME .ID ;
 : .FUZZY FUZZY DUP 5 .R SPACE .F SPACE .FUZZY_NAME ;
 : SHOW
     SET_FUZZY
     CR ." MAX " MAXX 3 .R 5 SPACES ." MIN " 3 .R
     MAXX DX 2/ + MINX DX 2/ -
     DO   CR I 3 .R I .FUZZY   LOOP  ;

 EXIT
 Miscellaneous tools.  SHOW is used to display a fuzzy set.
 SHOW <fuzzy.set> <CR>    e.g.   SHOW TALL
 A good fuzzy.set editor would be useful.
```

```
Scr # 18
( Database      6/22/85 rel )

 VARIABLE DATA^  VARIABLE #RECS  VARIABLE REC#  VARIABLE FIELD#

 : DATA
    HERE DATA^ !     0 #RECS !
 0  BEGIN
      BL WORD DROP     DUP 4 MOD 0=
      IF 1 #RECS +!  5 ALLOT ELSE HERE NUMBER DROP , THEN    1+
      >IN @ 1000 >
    UNTIL DROP  ;

 -->
DATA simply lays the data down in the dictionary


```

```
Scr # 19
 ( Database      6/22/85 rel )

 DATA   JOHN      18          72          160
        JIM       18          75          210
        JACK      19          60          180
        TOM       25          66          150
        DICK      30          69          175
        HARRY     35          68          170
        GUY       40          70          170
        BOB       45          64          205
        RICH      50          58          100
        RICK      55          69          180
        BILL      60          71          150
        PHIL      65          66          150
        AL        70          68          155
        MIKE      71          75          180       -->
```

```
Scr # 20
 ( Database      6/22/85 rel )

  : NAME^ REC# @ 11 * DATA^ @ + ;

  : NAME   0 FIELD# ! NAME^       ;
  : AGE    1 FIELD# ! NAME^ 5 + @ ;
  : HEIGHT 2 FIELD# ! NAME^ 7 + @ ;
  : WEIGHT 3 FIELD# ! NAME^ 9 + @ ;         -->

    Field definitions






```

```
Scr # 21
 ( Fuzzy Sets    6/22/85 rel )
  INCREASING  64 74 FUZZY.SET TALL
  DECREASING  60 66 FUZZY.SET SHORT
  LOCAL       64 70 FUZZY.SET AVG.HT
  INCREASING  50 65 FUZZY.SET OLD
  DECREASING  18 30 FUZZY.SET YOUNG
  MORE        0 100 FUZZY.SET VERY
  DECREASING  0   0 FUZZY.SET SMALL

  : Near 2DUP < IF SWAP THEN 2DUP + >R - 100 R> */ SMALL ;
  : MIDDLE     DUP OLD Not SWAP YOUNG Not And ;
  : OLDTALL    HEIGHT TALL  AGE OLD   And ;
  : YOUNGSHORT HEIGHT SHORT AGE YOUNG And ;
  -->
   Fuzzy set definitions.  People may differ on the degrees of
   membership.
```

```
Scr # 22
( Query Lang   6/22/85 rel )

 VOCABULARY PRONOUN

 PRONOUN DEFINITIONS

 : HE ;  : HIS ;   : SHE ;   : HER ;
 : ANYONE 0 REC# ! FORTH Begin ;

  -->
```

```
Scr # 23
( Query Lang   6/22/85 rel )
 FORTH DEFINITIONS
 : >LINE >IN @ C/L / C/L * ;
 : .Q BLK @ NOT
      IF CR  BLK @ BLOCK >LINE + C/L 10 - CR TYPE THEN ;

 : SEARCH ( stringaddr --- Flag    Searches database )
      0 REC# <
      BEGIN
         REC# @ #RECS @ <
      WHILE
         DUP NAME^ COUNT COMPARE 0=
         IF  DROP TRUE EXIT   ELSE   1 REC# +!   THEN
      REPEAT DROP FALSE ;
 -->
    .Q displays query when LOADing
```

```
Scr # 24
( Query Lang   6/22/5 rel )

 : Is .Q
      ONLY PRONOUN DEFINED
      IF   EXECUTE         ( is a pronoun so execute it      )
      ELSE 1+ SEARCH 0=    ( Not a pronoun so search database)
           IF CR ." I do not know " HERE COUNT TYPE CR
              ASCII ? WORD DROP     ( Skip over rest of query)
              ONLY FORTH EXIT       ( Continue interpreting  )
           THEN
      THEN ONLY FORTH ;
 -->
 Is parses a word.  If it is in the PRONOUN VOCABULARY then
 it is executed.  If not then Is SEARCHs the database to see
 if it is a name that it recognizes.
```

```
Scr # 25
( Query Lang   6/22/85 rel )

 : Tell
     Is CR NAME^ COUNT TYPE ."  is " AGE . ."  years old. "
        ." He is " HEIGHT . ."  inches tall and weighs "
        WEIGHT . ."  pounds." CR CR ;
 : About Tell 28 LOAD ;
 : ? ( Degree ---- )
     REPEAT? @
     IF    85 > IF CR NAME COUNT TYPE THEN
           1 REC# +! REC# @ #RECS @  < Until
     ELSE  CR 5 SPACES ." People "  .F ." say so." CR KQ THEN ;
 EXIT
 Tell is used in the sense of Tell about. e.g. Tell JOHN <CR>
 About is Tell followed by the batch of queries on Screen 28.
 Change About if you enter queries on other screens.
```

```
Scr # 26
( Queries     6/22/85 rel )
CR
Tell BILL
Is BILL HEIGHT TALL ?
Is HIS  HEIGHT SHORT ?
Is BILL HEIGHT AVG.HT Not ?
Is HER  HEIGHT TALL VERY Not ?
Is BILL HEIGHT TALL Not VERY ?
Is HIS  HEIGHT TALL HEIGHT SHORT Or ?
Is HIS  AGE OLD ?
Is BILL AGE YOUNG ?
Is BILL AGE MIDDLE ?
Is BILL AGE OLD Not HEIGHT TALL And ?
Is ANYONE HEIGHT TALL ? KQ
About DICK  About JACK
About MIKE                                          QUIT
```

```
Scr # 28
( Queries About 6/22/85 rel )
Is HIS  HEIGHT TALL ?
Is HIS  HEIGHT 70 Near ?
Is HIS  HEIGHT SHORT ?
Is HIS  HEIGHT AVG.HT   ?
Is HIS  HEIGHT TALL VERY ?
Is HIS  HEIGHT TALL Not VERY ?
Is HIS  HEIGHT TALL AGE OLD Or ?
Is HIS  AGE OLD ?
Is HIS  AGE YOUNG ?
Is HIS  AGE MIDDLE ?
Is HIS  AGE OLD Not HEIGHT SHORT And ?
Is HIS  AGE YOUNG HEIGHT TALL And ?
Is HE   YOUNGSHORT ?
Is HE   OLDTALL ?
EXIT
```