Information Processing in Japan Vol. 17, 1977

Implementation of a Fuzzy Sets Manipulation System

Motohide UMANO*, Masaharu MIZUMOTO* and Kokichi TANAKA*

Abstract

This paper describes an implementation of a system for fuzzy sets manipulation which is based on FSTDS (Fuzzy-Set-Theoretic Data Structure), an extended version of Childs' STDS (Set-Theoretic Data Structure). FSTDS language is considered as a fuzzy-set-theoretically oriented language which can deal with various kinds of fuzzy sets. FSTDS system, in which 52 fuzzy set operations are available, is implemented in FORTRAN, and currently running on a FACOM 230-45S computer.

1. Introduction

In the real world, there exist many fuzzy things which can not or need not be precisely defined. However, since Zadeh proposed the concept of fuzzy sets in 1965 [1], it has been studied vigorously and applied to various fields such as automata theory, formal languages, natural languages, logic, pattern recognition, learning theory, decision making and the mathematical theory of computation.[2]

It is well-known that ordinary set theory is very useful. Some systems can deal with ordinary sets; for example, STDS developed by Childs,[3] SETL by Schwartz and LOREL by Katayama.

Since fuzzy sets are considered a generalization of ordinary sets, a system which can deal with fuzzy sets is much more useful because of the wide applicability of fuzzy set theory.[4]

In this paper, we describe an implementation of a system for fuzzy sets manipulation which is based on FSTDS (Fuzzy-Set-Theoretic Data Structure), an extended version of Childs' STDS (Set-Theoretic Data Structure). FSTDS language is considered as a fuzzy-set-theoretically oriented language which can, for example, deal with ordinary sets, ordinary relations, fuzzy sets, fuzzy relations, L-fuzzy sets, level-m fuzzy sets, type-n fuzzy sets and generalized fuzzy sets.

FSTDS system, in which 52 fuzzy set operations are available, is implemented in FORTRAN, and currently running on a FACOM 230-45S computer.

2. Fuzzy-Set-Theoretic Data Structure

FSTDS is a data structure for representing fuzzy sets so that they can be manipulated conveniently and efficiently by fuzzy set operators. FSTDS is composed of eight areas as follows (see Figs. 1 - 3):

(1) Fuzzy Set Area (FSA)

(2) Fuzzy Set Representation Area (FSRA)

This paper first appeared in Japanese in Joho－Shori (Journal of the Information Processing Society of Japan), Vol. 18, No. 9 (1977), pp. 884～892.

* Faculty of Engineering Science, Osaka University

(3) Grade Area (GA)
(4) Grade Tuple Area (GTA)
(5) Element Area (EA)
(6) Element Tuple Area (ETA)
(7) Fuzzy Set Name Area (FSNA)
(8) Fuzzy Set Operator Name Area (FSONA).

<u>Example 1.</u> The fuzzy Set F and fuzzy relation R defined by

F = {0.1/a, 0.8/b, 0.9/d}

and

R = {0.3/<a,b>, 0.9/<b,d>},

respectively, are represented by FSTDS in Fig.1.

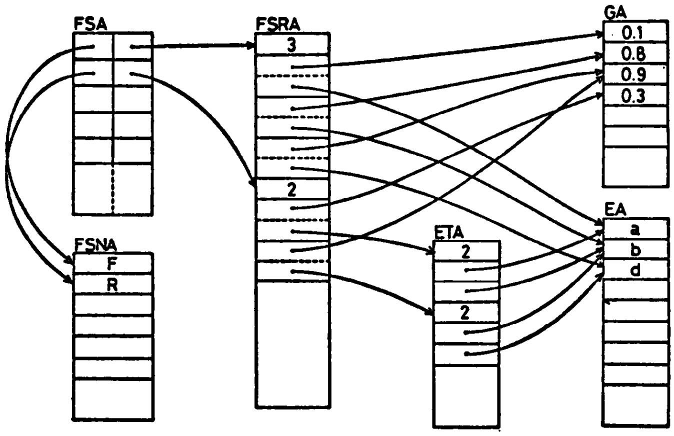


Fig.1 The representation of a fuzzy set F and a fuzzy relation R by FSTDS

<u>Example 2.</u> The L-fuzzy set X:

X = {<0.1,0.9>/a, <0.8,1>/b, <0.9,0>/c}

and level-2 fuzzy set Y:

Y = {0.6/Y1, 0.1/Y2}

where Y1 and Y2 are defined by

Y1 = {0.3/a, 0.2/b, 0.9/d},

Y2 = {0.6/a, 0.1/b},

respectively, are represented by FSTDS in Fig.2 and Fig.3, respectively.

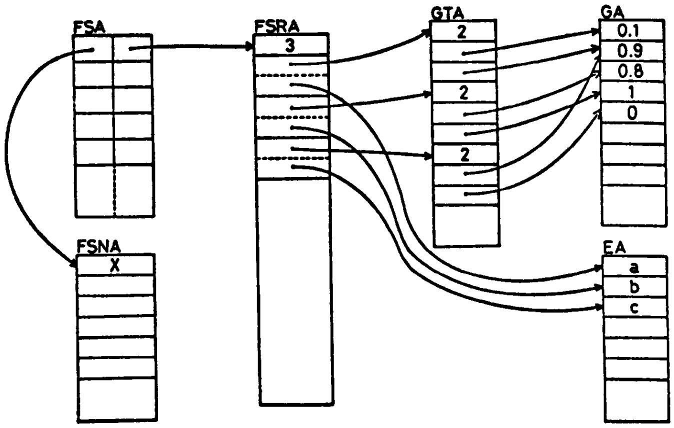


Fig.2 The representation of an L-fuzzy set X by FSTDS

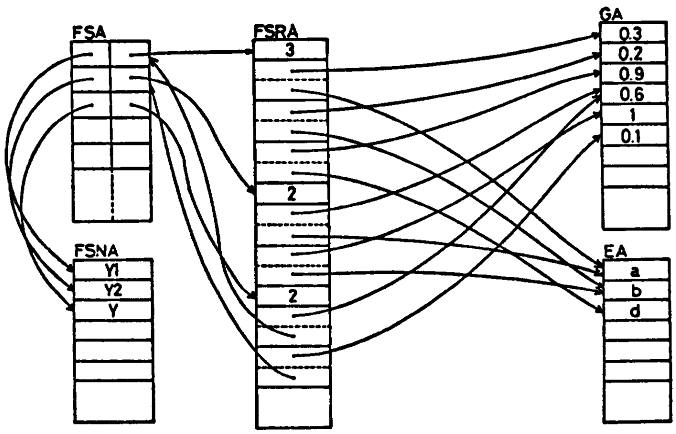


Fig.3 The representation of a level-2 fuzzy set Y by FSTDS

With FSTDS it is possible to represent not only ordinary fuzzy sets, L-fuzzy sets, level-m fuzzy sets and type-n fuzzy sets, but also more complex fuzzy sets which we call "generalized fuzzy sets", for example, an L-type-3 fuzzy set, a level-5 type-2 fuzzy relation.

We can use FSTDS as a data structure for representing fuzzy sets. But it is somewhat troublesome and a source of error if a user has to manage and manipulate many sorts of pointers in FSTDS. We, therefore, give a method by which a user can define and manipulate fuzzy sets without worrying about the pointers in FSTDS. This may be considered as a programming language for the manipulation of fuzzy sets and fuzzy relations.

3. Fuzzy-Set-Theoretic Data Structure System

In this section, we shall describe FSTDS language, FSTDSL for short, which is a programming language that enables a user to make use of FSTDS, and FSTDS system which interpretes and executes a program written in FSTDSL.

FSTDS system has 52 fuzzy set operators currently available. (See Table 1.) We present some simple examples to illustrate FSTDSL.

Table 1 Fuzzy set operators available in FSTDS system

fuzzy set operators	#opd	remarks
SET($u_1,u_2,\ldots,u_n$)	n≥0	construct ordinary set
FSET($\mu_1/u_1,\mu_2/u_2,\ldots,\mu_n/u_n$)	n≥0	construct fuzzy set
ASSIGN(Y,X)	2	assign X to Y (same as Y:=X)
UNION($X_1,X_2,\ldots,X_n$)	n≥2	union of $X_1,X_2,\ldots,X_n$
INTERSECTION($X_1,X_2,\ldots,X_n$)	n≥2	intersection of $X_1,X_2,\ldots,X_n$
PROD($X_1,X_2,\ldots,X_n$)	n≥2	product of $X_1,X_2,\ldots,X_n$
ASUM($X_1,X_2,\ldots,X_n$)	n≥2	algebraic sum of $X_1,X_2,\ldots,X_n$
ADIF($X_1,X_2,\ldots,X_n$)	n≥2	algebraic difference of $X_1,X_2,\ldots,X_n$
BSUM($X_1,X_2,\ldots,X_n$)	n≥2	bounded sum of $X_1,X_2,\ldots,X_n$
BDIF($X_1,X_2,\ldots,X_n$)	n≥2	bounded difference of $X_1,X_2,\ldots,X_n$
UNIONA(χ)	1	
INTERSECTIONA(χ)	1	
PRODA(χ)	1	operate on all fuzzy sets over the domain of the operand set χ
ASUMA(χ)	1	
ADIFA(χ)	1	
BSUMA(χ)	1	
BDIFA(χ)	1	
COMPOSE($R_1,R_2,\ldots,R_n$)	n≥2	composition of $R_1,R_2,\ldots,R_n$
CONVERSE(R)	1	converse relation of R
IMAGE(R,X)	2	image of X under R
CIMAGE(R,X)	2	converse image of X under R
DOMAIN(R)	1	domain of R
RANGE(R)	1	range of R
CP($X_1,X_2,\ldots X_n$)	n≥2	Cartesian product of $X_1,X_2,\ldots X_n$
RS(R,X)	2	restriction of R to X
RELATION(X)	1	translate level m fuzzy set X to fuzzy relation

fuzzy set operators	#opd	remarks
EQ(X_1,X_2)	2	Is X_1 equal to X_2?
SUBSET(X_1,X_2)	2	Is X_1 a subset of X_2?
DISJOINT($X_1,X_2,\ldots,X_n$)	n≥2	Are $X_1,X_2,\ldots,X_n$ disjoint from each other?
ELEMENT(μ/u,X)	2	Is μ/u an element of X?
CUT(μ_1/μ_2,X)	2	$A_{\mu_2}^{\Lambda}\ \mu_1$
SOP(μ/n,X)	2	scalar operation of μ and X
EXP(μ/x,X)	2	$X^x \Lambda\ \mu$
DIL(X)	1	dilation
CON(X)	1	concentration
CINT(X)	1	contrast intensification
NORM(X)	1	normalization of X
CD(X)	1	cardinality of X
#(X)	1	the number of elements of X
MAXG(X)	1	the maximum grade of X
SF(X,K)	2	support fuzzification of X by K
GF(X,K)	2	grade fuzzification of X by K
DLT($X_1,X_2,\ldots,X_n$)	n≥1	delete $X_1,X_2,\ldots,X_n$ from system
PRINT($X_1,X_2,\ldots,X_n$)	n≥1	print out $X_1,X_2,\ldots,X_n$
PRINTB($X_1,X_2,\ldots,X_n$)	n≥1	print out $X_1,X_2,\ldots,X_n$ in Boolean type
PRINTS($X_1,X_2,\ldots,X_n$)	n≥1	print out $X_1,X_2,\ldots,X_n$ in set type
PRINTN($X_1,X_2,\ldots,X_n$)	n≥1	print out $X_1,X_2,\ldots,X_n$ with names
PRINTL(character string)	1	print out character string
DUMP($\alpha_1,\alpha_2,\ldots,\alpha_n$)	n≥1	dump areas in FSTDS
SNAP(α)	1	print out all fuzzy sets
PARA($\alpha_1=\beta_1,\alpha_2=\beta_2,\ldots,\alpha_n=\beta_n$)	n≥1	specify the options
END(X)	1or0	evaluate X and halt

1. The symbols (with subscripts) in Table 1 represent the following meaning:

   u: an element; that is, a real number, a character string or a fuzzy set, or an n-tuple of them
   μ: a grade; that is, a number in the interval [0,1] or a fuzzy set, or an n-tuple of them
   X: an expression or a fuzzy set
   Y: a fuzzy set or a fuzzy set to be defined
   R: a fuzzy relation
   χ: a set of fuzzy sets
   n: an integer
   x: a real number
   α: an alphabetical character
   K: a kernel set
   β: an option of PARA operator.

2. For SET and FSET operator, SET() and FSET(), i.e. n=0, mean the empty set.
3. For END operator, END can be used for END().

Example 3. In FSTDSL, we can write

```
F:=FSET(0.1/A, 0.8/B, 0.9/D);
R:=FSET(0.3/<A,B>, 0.9/<B,D>);
```

to define the fuzzy set F and the fuzzy relation R in Example 1. FSTDS system interprets above statements and sets up FSTDS shown in Fig.1.

Example 4. We can represent a fuzzy directed graph G shown in Fig.4 by FSTDSL statements:

```
V:=SET(X,Y,Z,W);
A:=FSET(0.1/<X,Y>, 0.7/<Y,Z>, 0.4/<W,Z>,
   1/<W,Y>, 0.3/<X,W>, 0.9/<W,X>);
G:=SET(<V,A>);
```

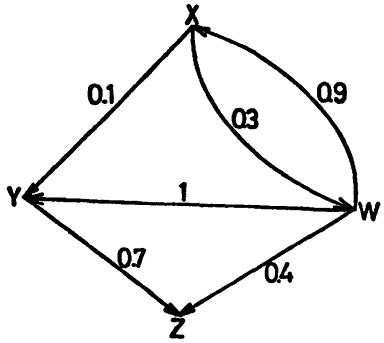


Fig.4 A fuzzy directed graph G

Example 5. Let X and Y be fuzzy sets in U, and R a fuzzy relation in U × V. Then, we have the following identity:

$$(X \bigcup Y) \circ R = (X \circ R) \bigcup (Y \circ R)$$

where $\bigcup$ denotes the union of fuzzy sets and $\circ$ the composition of fuzzy relations. But in this case, X and Y are unary fuzzy relations (i.e., ordinary fuzzy sets), so X$\circ$R reduces to the image of X under R. Suppose that X and Y are defined by

X = {1/a, 0.9/b, 0.3/c}

Y = {0.1/a, 0.7/b, 0.9/c}

and R is defined in terms of relation matrix:

$$R = \begin{array}{c} \\ a \\ b \\ c \end{array} \begin{array}{c} \begin{array}{ccc} a & b & c \end{array} \\ \left[\begin{array}{ccc} 1 & 0.8 & 0 \\ 0.7 & 1 & 0.2 \\ 0 & 0.5 & 0.1 \end{array}\right] \end{array}$$

Then we can write the program in Fig.5(a) and the execution results are shown in Fig.5(b).

```
X:=FSET(1/A, 0.9/B, 0.3/C);
Y:=FSET(0.1/A, 0.7/B, 0.9/C);
R:=FSET(1/<A,A>, 0.8/<A,B>, 0.7/<B,A>, 1/<B,B>,
        0.2/<B,C>, 0.5/<C,B>, 0.1/<C,C>);
PRINT(ASSIGN(Z,UNION(X,Y)));
PRINT(IMAGE(R,Z));
V:=IMAGE(R,X);   W:=IMAGE(R,Y);
PRINT(UNION(V,W));
END;
```

(a) Program in FSTDSL

```
FSET(1/A, 0.9/B, 0.9/C);   ... X⋃Y
FSET(1/A, 0.9/B, 0.2/C);   ... (X⋃Y)oR
FSET(1/A, 0.9/B, 0.2/C);   ... (XoR) ⋃ (YoR)
```

(b) Output

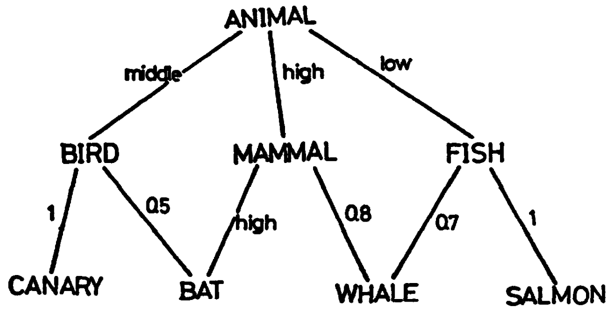
Fig.5 Program and output of $(X \bigcup Y) \circ R = (X \circ R) \bigcup (Y \circ R)$

Example 6. The fuzzy knowledge shown in Fig.6(a) is represented in FSTDSL by the level-2 type-2 fuzzy set ANIMAL shown in Fig.6(b).

The question "What does a BAT belong to?" will be translated into FSTDSL statements as

```
ISA:=CONVERSE(
     RELATION(ANIMAL));
X:=IMAGE(ISA,SET(BAT));
```

where RELATION is a fuzzy set operator by which a level-m fuzzy set is translated into



(a) An example of fuzzy knowledge

```
LOW:=FSET(1/0, 0.8/0.1, 0.4/0.2);
MIDDLE:=FSET(1/0.5, 0.5/0.6, 0.5/0.4);
HIGH:=FSET(1/1, 0.8/0.9, 0.4/0.8);
BIRD:=FSET(1/CANARY, 0.5/BAT);
MAMMAL:=FSET(HIGH/BAT, 0.8/WHALE);
FISH:=FSET(0.7/WHALE, 1/SALMON);
ANIMAL:=FSET(MIDDLE/BIRD, HIGH/MAMMAL, LOW/FISH);
```

(b) FSTDSL statements for the fuzzy knowledge

Fig.6 Fuzzy knowledge and its representation by FSTDSL a fuzzy relation. Then the output of above X (i.e., PRINT(X);) is

```
FSET(0.5/BIRD, HIGH/MAMMAL);
```

Thus, we have the answer "A BAT belongs to BIRDs with the compatibility 0.5 and to MAMMALs with high compatibility".

As shown in Examples 3 - 6, FSTDSL has a simple syntax and is designed to have no labels and no control structures. This is because FSTDS system has another user interface, that is, the connection of FSTDSL and FORTRAN. If a user wants to use a control structure, he may make use of that of FORTRAN.

Example 7. A program in FSTDSL and FORTRAN shown in Fig.7(a) causes the result output in Fig.7(b). FSTDSL can be embedded in FORTARN like this. One must put the character F at the head of an FSTDSL statement to differentiate an FSTDSL statement from a FORTARN statement and put one exclamation mark ! and two !! followed by FORTRAN integer and real variables, respectively, indicating its value inside an FSTDSL statement. This program shows how to define a universe of discourse U and a fuzzy set LARGE, and compute the complement of LARGE and output them.

```
F       PARA(G=1)
        N=5
F       LARGE=U=EMPTY
        DO 10 I=1,N
        G=FLOAT(I)/FLOAT(N)
F       LARGE=UNION(LARGE, FSET(!!G/!I))
F       U=UNION(U,SET(!I))
     10 CONTINUE
F       PRINTN(U,LARGE)
F       NOT_LARGE=ADIF(U,LARGE); PRINTN(NOT_LARGE)
        STOP
        END
```

(a) FORTRAN program with FSTDSL

```
U=FSET(1.0/1, 1.0/2, 1.0/3, 1.0/4, 1.0/5);
LARGE=FSET(0.2/1, 0.4/2, 0.6/3, 0.8/4, 1.0/5);
NOT_LARGE=FSET(0.8/1, 0.6/2, 0.4/3, 0.2/4);
```

(b) Output

Fig.7 FSTDSL embedded in FORTRAN

The connection of FSTDSL and FORTRAN greatly extends the capability and the applicability of FSTDSL. From an opposite point of view, this allows the provision in FORTRAN of facilities to define and manipulate fuzzy sets and fuzzy relations.

4. Conclusion

To solve a given problem, we can write a program in FSTDSL using the concept of fuzzy sets. We can use FSTDS system to construct a fairly large scale system, for we need not pay attention to the representation of fuzzy sets and the computations of fuzzy set operations, and we can describe complex and detailed processing in FORTRAN.

As applications of FSTDS system, we are now implementing an Approximate Reasoning System[4] and a Fuzzy Graph Manipulation System.

FSTDS system will find various applications in the fields in which we have to deal with fuzzy information and fuzzy knowledge in nature.

[References]

1. L.A.Zadeh, Fuzzy Sets, *Information and Control*, Vol.8, pp.338-353 (1965).
2. L.A.Zadeh, K.Tanaka et al. (eds.), *Fuzzy Sets and Their Applications to Cognitive and Decision Processes*, Academic Press Inc., New York (1975).
3. D.L.Childs, Description of a Set-Theoretic Data Structure, *FJCC*, pp.557-564 (1968).
4. L.A.Zadeh, The Concept of a Linguistic Variable and Its Application to Approximate Reasoning, *Information Sci.*, Vol.8, pp.199-249; pp.301-357; Vol.9, pp.43-80 (1975).