Reprinted from:
FUZZY SETS AND THEIR APPLICATIONS TO
COGNITIVE AND DECISION PROCESSES

New York San Francisco London

FUZZY PROGRAMS AND THEIR EXECUTION

K. Tanaka and M. Mizumoto
Department of Information
and Computer Sciences
Faculty of Engineering Science
Osaka University
Toyonaka, Osaka 560, Japan

1. INTRODUCTION

In our daily life, we often encounter situations where we shall not always need the exact and detailed information to execute the intended behavior. For instance, let us suppose the case where a person asks the way in a strange place. For example, he will receive such an instruction as: "go straight on this way and turn right at the signal, then you could find the spot after about a few minutes walk." Then he could get to the spot without trouble, if the instruction is true. However, if we want to make a machine execute such an instruction as mentioned above, just then we shall find it difficult to do.

In the real world, as a matter of fact, many ill-defined and inexact instructions, that is, the so-called fuzzy instructions exist which we want to translate and execute by a machine. Therefore, the execution of fuzzy instructions using a machine is of much interest and very useful in a wide variety of problems relating to pattern recognition, control, artificial intelligence, linguistics, information retrieval and decision processes involved in psychological, economical and social fields.

In this paper, a generalized automaton is proposed as an abstract model for a fuzzy machine which can translate and execute fuzzy programs and several methods which translate a given sequence of fuzzy instructions into another

sequence of precise instructions called a machine program are also discussed.

In addition, the practical application is presented in a few interesting examples to demonstrate the usefulness of the foregoing proposal.

2. GENERALIZED FUZZY MACHINES

A finite-state automaton has been taken up as a fuzzy machine model which executes a fuzzy program by S. K. Chang [1].

Here formulated is an extended fuzzy machine based on a generalized automaton and a few procedures for execution of fuzzy programs are also presented.

Definition 1. A generalized machine M is a system given by

$$M = (k, X, \psi, x_0, T, V) \tag{1}$$

where (i) K is a finite set of machine instructions.

(ii) X is a finite set of internal states.

(iii) ψ is a function such that

$$\psi: \quad X \times K \times X \to V \tag{2}$$

and is called a state transition function. The value of ψ, $\psi(x, \mu, x') \in V$, designates a weight value controlling the transition from a state x to a new state x' for a given machine instruction μ.

(iv) x_0 is an initial state in X.

(v) T is a finite set of final states and is a subset of X.

(vi) V is a space of weight (or grade) controlling the state transition.

In the present paper an "L-fuzzy automaton" with the weight space defined in the lattice ordered semigroup is considered as a general machine. Several machines are also derived from L-fuzzy automata as their specific examples [2], [6].

Let us now define $V = (L, v, *, 0, I)$ in a lattice ordered semigroup L. where v and * denote a least upper bound in L and an operation of semigroup, respectively; and 0 and I denote zero (least element) and identity (greatest element), respectively. Then the state transition of L-fuzzy automata can be formulated as follows. For a given string of machine instructions $\bar{\mu} = \mu_1\mu_2\cdots\mu_n$ in K* where K* denotes a set of all finite strings over K, the state transition function at each step of the machine instruction will be $\psi(x_0, \mu_1, x_1)$, $\psi(x_1, \mu_2, x_2), \ldots, \psi(x_{n-1}, \mu_n, x_n)$. Then the state of the L-fuzzy automaton is said to transit from x_0 through x_n one by one by the string of machine instruction $\bar{\mu}$ and the weight (or grade) corresponding to this state transition is simply given by

$$\psi(x_0, \psi_1, x_1)*\psi(x_1, \mu_2, x_2)*\cdots*\psi(x_{n-1}, \mu_n, x_n) \quad (3)$$

Thus the domain X x K x X of the state transition function ψ will be extended to X x K* x X and the weight (or grade) of the state transition for any input string $\bar{\mu} = \mu_1\mu_2\cdots\mu_n \in K^*$ can be given recursively as

$$\psi(x, e, x') = \begin{cases} I & \text{for} \quad x = x' \\ 0 & \text{for} \quad x \neq x' \end{cases} \quad (4)$$

$$\psi(x, \bar{\mu}, x') = \bigvee_{x_1, x_2, \ldots, x_{n-1}} [\psi(x, \mu_1, x_1)*\psi(x_1, \mu_2, x_2)* \cdots *\psi(x_{n-1}, \mu_n, x')] \quad (5)$$

where e denotes a null string.

Note: For the following algebraic structure of V, various types of automata can be derived as specific cases of L-fuzzy automata.

(i) For V = ([0,1], max, min, 0, 1), fuzzy automata can be obtained.

(ii) For V = ({0,1}, max, min, 0, 1), nondeterministic automata can be obtained.

(iii) For V = ({0,1}, max, min, 0, 1), deterministic automata can be obtained under the constraint as follows: there exists x' uniquely such that $\psi(x, \mu, x') = 1$ for each pair of x and μ.

(iv) For V = ([0,1], +, x, 0, 1), probabilistic automata can be obtained under the constraint such that $\sum_{x' \in X} (x, \mu, x') = 1$.

Definition 2. A generalized fuzzy machine is a system

$$\hat{M} = (\Sigma, M, W) \tag{6}$$

where (i) Σ is a finite set of fuzzy instructions and each fuzzy instruction σ_i is a function such that

$$\sigma_i : X \times K \to W \tag{7}$$

(ii) M is a generalized automaton defined by Definition 1.

(iii) W is a space of weight (or grade) with respect to the selection of a machine instruction μ_i. The value of $\sigma_i(x_i, \mu_i) \in W$ designates the weight (or grade) of selecting the machine instruction μ_i when a generalized fuzzy machine $\hat{M}$ associated with a generalized automaton M in the state of x_i has received a fuzzy instruction σ_i.

2.1. Fuzzy Machines Derived From Deterministic Automata

This is the case where a deterministic automaton is chosen as an example of generalized machine M.

(a) For W = [0,1], a fuzzy-deterministic machine similar to that of S. K. Chang can be derived, where

$$\sigma_i(x_i, \mu_i) = \min[f(x_i, \sigma_i, \mu_i), \lambda(x_i, \sigma_i, \mu_i)]$$

shows the grade of selecting the machine instruction μ_i, when the machine M is in the state of x_i and receives the fuzzy instruction σ_i. Here note that $f(\cdot)$ and $\lambda(\cdot)$ in the above equation represent the feasibility function and the performance function, respectively [1].

(b) For W = [0,1], a probabilistic-deterministic machine can be derived under the condition that

$$\sum_{\mu_i} \sigma_i(x_i, \mu_i) = 1 \text{ for every } \sigma_i \in \Sigma \text{ and } x_i \in X.$$

This condition shows that a machine instruction μ_i is selected in a probabilistic way when the machine is in the state of x_i and receives a fuzzy instruction of σ_i.

(c) For W = {0,1} can be obtained a nondeterministic-deterministic machine, where $\sigma_i(x_i, \mu_i) = 1$ or $\sigma_i(x_i, \mu_i) = 0$.

The equation of $\sigma_i(x_i, \mu_i) = 1$ shows that a machine instruction μ_i is selected in a nondeterministic way when the machine is in the state x_i and receives a fuzzy instruction σ_i.

2.2. Another Type of Fuzzy Machines Derived From Various Classes of Automata

A variety of generalized fuzzy machines will be derived from various classes of automata.

For W = [0,1] and V = [0,1] can be exemplified the generalized fuzzy machines enumerated as below.

Fuzzy-fuzzy machine; Fuzzy-probabilistic machine; Fuzzy-nondeterministic machine; Fuzzy-deterministic machine; Probabilistic-fuzzy machine; Probabilistic-probabilistic machine; Probabilistic-nondeterministic machine; Probabilistic-deterministic machine; Nondeterministic-fuzzy machine; Nondeterministic-probabilistic machine; Nondeterministic-nondeterministic machine; Nondeterministic-deterministic machine

Note: The fuzzy machine defined by S. K. Chang is equivalent to a fuzzy-deterministic machine and that of Jakubowski [3] is equivalent to a fuzzy-nondeterministic machine.

Definition 3. A sequence of fuzzy instructions of $\sigma_1, \sigma_2, \ldots, \sigma_n \in \Sigma$ is called an elementary fuzzy program $\bar{\sigma} = \sigma_1\sigma_2 \ldots \sigma_n \in \Sigma^*$ and a fuzzy program is a regular expression over Σ. If every machine instruction μ_i is feasible with respect to the every situation of the machine M and the state of the machine will transit successively from x_0 through x_n, that is, if

$$\begin{aligned} &\sigma_1(x_0, \mu_1) = w_1 \qquad \psi(x_0, \mu_1, x_1) = v_1 \\ &\sigma_2(x_1, \mu_2) = w_2 \qquad \psi(x_1, \mu_2, x_2) = v_2 \\ &\qquad \cdots\cdots\cdots\cdots\cdots\cdots \\ &\sigma_n(x_{n-1}, \mu_n) = w_n \qquad \psi(x_{n-1}, \mu_n, x_n) = v_n, \end{aligned} \tag{8}$$

then the fuzzy program $\bar{\sigma}$ is said to be executable with respect to the fuzzy machine $\hat{M}$ and the machine program $\bar{\mu}$ is said to be an execution of the fuzzy program $\bar{\sigma}$. This statement will be represented in the form as

$$x_0 \frac{(\sigma_1, \mu_1)}{w_1} x_1 \frac{(\sigma_2, \mu_2)}{w_2} x_2 \cdots x_{n-1} \frac{(\sigma_n, \mu_n)}{w_n} x_n. \tag{9}$$

Definition 4. Let the weight (w,v) be defined by

$$(w,v) = (w_1 \cdot w_2 \cdot \ldots \cdot w_n,\ v_1 * v_2 * \ldots * v_n) \quad (10)$$

where the marks $\cdot$ and $*$ denote the operation on the weight space W and V, respectively. Then the fuzzy program $\bar{\sigma}$ is said to be executable with the weight (w,v) if $(w,v) > (0,0)$.

Example 1: Let the operation $\cdot$ on $W = [0,1]$ be min ($\wedge$) and the operation $*$ on $V = \{0,1\}$ be min or x(product). Then the generalized fuzzy machine will be a fuzzy-nondeterministic machine described previously and the corresponding weight (w,v) is given by

$$(w,v) = (w_1 \wedge w_2 \wedge \ldots \wedge w_n,\ 1),$$

when the fuzzy program $\bar{\sigma}$ is feasible. The 1 on the right side of the above expression means that $\psi(x_0, \mu_1, x_1) = \psi(x_1, \mu_2, x_2) = \ldots = \psi(x_{n-1}, \mu_n, x_n) = 1$.

Example 2: Let the operation $\cdot$ on $W = [0,1]$ be min and the operation $*$ on $V = [0,1]$ be x(product). Then the generalized fuzzy machine will be a fuzzy-probabilistic machine and the corresponding weight (w,v) is given by

$$(w,v) = (w_1 \wedge w_2 \wedge \ldots \wedge w_n,\ v_1 \times v_2 \times \ldots \times v_n).$$

3. EXECUTION PROCEDURE OF FUZZY PROGRAMS

As one particular way of executing fuzzy programs using a finite state machine, Chang has given the way called simple execution procedure which selects the machine instruction μ_i with the highest grade $w_i = \sigma(x_{i-1}, \mu_i)$ with respect to the fuzzy instruction σ_i at each step i.

In this paper discussed is a more general way of executing fuzzy programs by making use of the generalized fuzzy machine described previously. The reason why the machine is to be given with such a generality as mentioned above with

regard to not only the selection of the machine instruction, but also the mode of state transition will be easily approved in the following example. Suppose that a person has received such a fuzzy instruction as, for instance, "come to the school at about 9:30 a.m.". Then he will make up his mind to come to the school just on 9:20 a.m. This corresponds to the selection of a machine instruction. In fact, however, he will not be able to come to the school just on 9:20 a.m. as usual, but his arrival will shift slightly from 9:20 a.m. This may be interpreted as corresponding to the state transition. Thus, the generality of the state transition given to the machine will enlarge the executability of the fuzzy programs.

Furthermore, in such a case where the state aimed at (for instance, arrival to the destination, in the example of simulation of human drivers) can not be attained, there will be needed to alter the way of selecting either the machine instruction or the state transition. That is to say, if a fuzzy instruction is received by the machine in any state, the machine instruction will be selected in a certain way and the state of the machine will change according to a certain manner. Then, if the state after the transition is not equivalent to the state aimed at, the successive transition will occur step by step according to the same manner as above until the attainment of the state aimed at. In this case, if the successive state will not be available, the machine instruction is updated and the same procedure is repeated for the same fuzzy instruction. If there is no machine instruction available for the given fuzzy instruction, a backtracking procedure will be introduced. That is to say, the available machine instruction must be selected to the fuzzy instruction of one step prior to the last one and the desired

state must be searched according to the same procedure described above.

The following Figure 1 shows the flow chart of the above mentioned procedure for execution of an elementary fuzzy program $\bar{\sigma} = \sigma_1\sigma_2 \ldots \sigma_n$ by making use of a generalized fuzzy machine $\hat{M}$ given by Equation (6). The way of selecting the machine instruction and that of transition of the state, which is labeled as (1) and (2) in Figure 1, respectively, will differ depending on the class of a generalized machine chosen. Then let us now explain that in detail.

3.1 Selection of Machine Instruction

(a) In case of fuzzy selection, the machine $\hat{M}$ selects the machine instruction $\mu \in K(i, x(i-1))$ with the highest grade at each step of the fuzzy instruction σ_i such that

$$\sigma_i(x(i-1),\mu) \geq \sigma_i(x(i-1), \mu') \tag{11}$$

for all other μ' in K.

(b) In case of probabilistic selection, the machine $\hat{M}$ selects the machine instruction $\mu \in K(i, x(i-1))$ with the probability p at each step of the fuzzy instruction σ_i in proportion to the fuzzy grade $\sigma_i(x(i-1),\mu)$ such that

$$p = \frac{\sigma_i(x(i-1),\mu)}{\sum_{\mu' \in K(i,\, x(i-1))} \sigma_i(x(i-1),\mu')} \tag{12}$$

(c) In case of nondeterministic selection, the machine instruction μ is chosen in $K(i, x(i-1))$ in a nondeterministic manner.

3.2 State Transition

(a) In case of fuzzy transition, the state of the machine transits from $x(i-1)$ to x such that

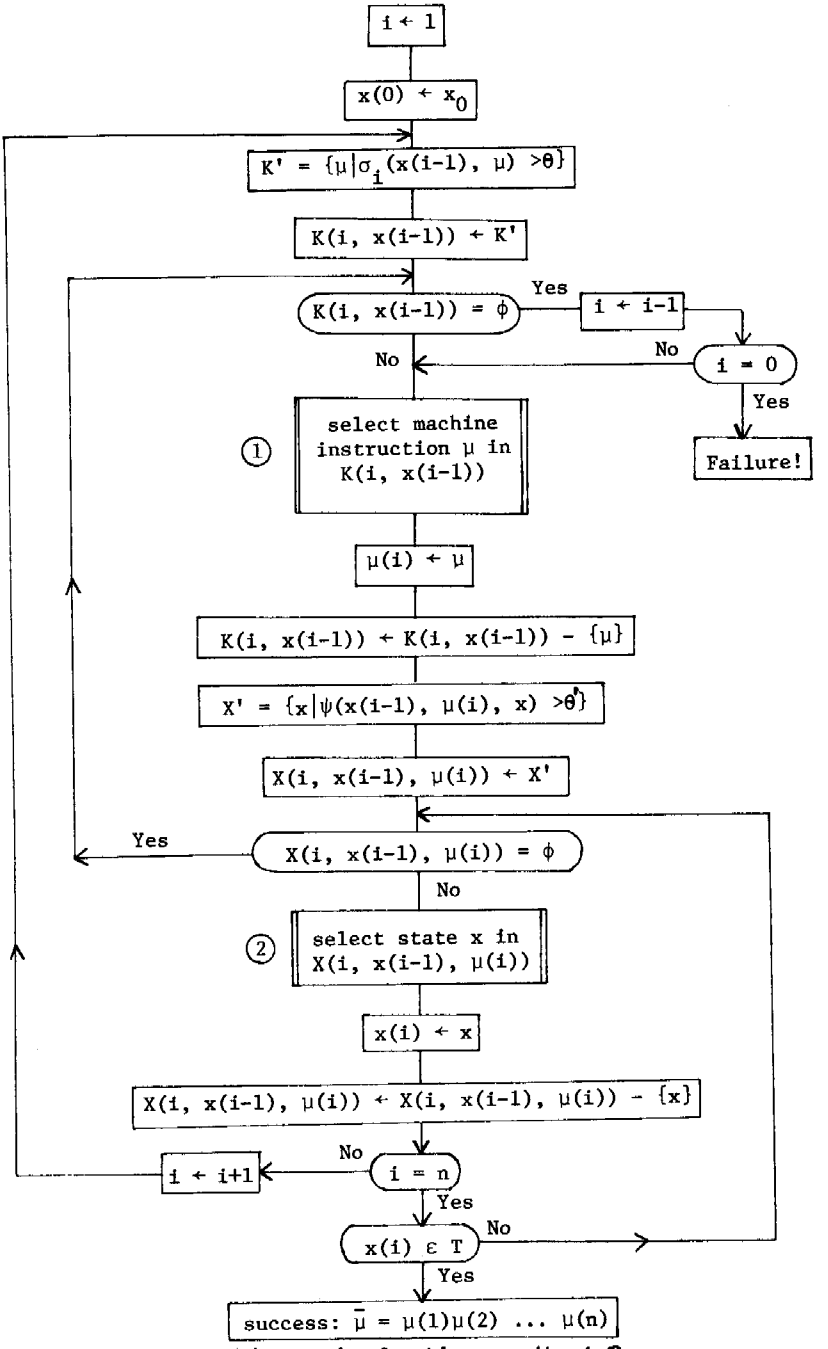


Figure 1 Caption on Next Page

$$\psi(x(i-1), \mu(i), x) \geq \psi(x(i-1), \mu(i), x') \qquad (13)$$

for all other $x' \in X(i, x(i-1), \mu(i))$.

(b) In case of probabilistic transition, the state of the machine transits to x from x(i-1) with the probability p, where

$$p = \frac{\psi(x(i-1), \mu(i), x)}{\sum_{x' \in X(i, x(i-1), \mu(i))} \psi(x(i-1), \mu(i), x')}. \qquad (14)$$

(c) In case of nondeterministic transition, the state of the machine transits from x(i-1) to x(i) in $X(i, x(i-1), \mu(i))$ in a nondeterministic way.

(d) In case of deterministic transition, the state available for the machine is uniquely determined depending on the property of its state transition function ψ.

4. SIMULATION OF HUMAN DRIVER'S BEHAVIOR

A simulation was conducted so as to experiment the procedures for executing a fuzzy program by the use of either a fuzzy-deterministic machine or a probabilistic-deterministic machine.

Let the fuzzy instructions for a driver by the following five kinds:

(i) $\sigma_{go}(L)$: Go about L meters, (ii) σ_R: Turn right,
(iii) σ_L: Turn left, (iv) σ^*_{go}: Go straight, (15)
(v) $\sigma_{\{\sim\}}$: Until $\sim$,

Preceding to the execution of these fuzzy instructions, each of them has to be rewritten as a sequence of the following three kinds of quasi- fuzzy instructions, i.e.,

Figure 1 Flow Chart Illustrating Execution Procedure of Fuzzy Program

(i) σ_{go}: Go ahead one step, (ii) σ_R: Turn right, (iii) σ_L: Turn left, (16)

by making use of the three methods of MAX-method, PROB-method and *-method for rewriting. The quasi-fuzzy instructions thus obtained are then interpreted by the MAX-method into the eight kinds of machine instructions which are composed of the elementary movements given by the eight directions.

4.1. Fuzzy Instructions, Quasi-fuzzy Instructions, and Quasi-internal States

A computer experiment was made to simulate the behavior of a driver who is directed the way by a sequence of fuzzy instructions. The initial position (the coordinates and the direction) is given and a typical set of fuzzy driving instructions and a set of quasi-fuzzy instructions are shown respectively by (15) and (16) mentioned previously.

Assuming the quasi-fuzzy instructions just as the machine instructions, the execution procedure discussed in the Section 3 can now be available to rewrite the fuzzy instructions into a sequence of the quasi-fuzzy instructions.

The internal state of a fuzzy machine is given as a pair of the coordinate and the direction with respect to each position of the driver on a digitized map shown in Figure 7. However, let us now introduce the notion of a quasi-internal state so as to reduce the number of the internal states. The roads on the map are classified according to the shape of the node and the branch, and the quasi-internal state of the fuzzy machine is designated as a pair of the shape of the node or the branch on the map and the direction of the driver as shown in Figure 4.

Then let us compose an evaluation table for selecting a machine instruction at each step of the quasi-fuzzy

instruction and the quasi-internal state as follows. Let the machine instructions be assigned by the following eight instructions of μ_j's(j=1,2,...,8) where μ_j denotes the instruction with respect to the j-th direction in Figure 7. If the evaluation value $\Phi(\beta,s)$ defined below is given for selecting a machine instruction, the evaluation table can be made for a given pair (β,s) of a quasi-fuzzy instruction β and a quasi internal-state s as shown in Figure 4.

$$\Phi(\beta,s) = \begin{cases} 0, \text{ if there is no machine instruction available.} \\ i, \text{ if } \mu_i \text{ is available.} \\ ix10+j, \text{ if } \mu_i \text{ is more available than } \mu_j. \end{cases} \quad (17)$$

where the last equation means that the grade of selecting the machine instruction μ_i is higher than that of selecting μ_j when the fuzzy machine is in the quasi-internal state s, i.e., $\beta(s, \mu_i) > \beta(s, \mu_j)$.

4.2. Execution Procedures of Fuzzy Instructions

There are two cases of giving fuzzy instructions, that is, (a) the case where fuzzy instructions are given step by step, and (b) the case where a sequence of fuzzy instructions is given a priori. A practical example of (a) is supposed to be the case where a fellow passenger gives the fuzzy instruction step by step to the driver who has to memorize all the past fuzzy instructions given at each step as well as the past fuzzy instructions given at each step as well as the present one and has to judge and behave by himself. On the other hand, the case of (b) will be illustrated by such an example that the driver is given a note showing a route and he can know the state of the route beforehand.

Let us consider the execution procedure of, for instance, a fuzzy instruction as "Go about L meters" given in the

expression (15). The idea of "about L meters" may be dealt with the concept of a fuzzy set and it will be characterized by a membership function as shown in Figure 2. The mode of selection of the distance for a fuzzy instruction named "Go about L meters" willbe specified by the following three types.

① Type 1 where a threshold α is set and the distance with the highest grade of membership among all the distances whose grades of membership are larger than α is selected.

② Type 2 where the distance is selected with the probability proportional to the grade of membership which is larger than a specified threshold α.

③ Type 3 where any distance whose grade of membership is larger than a specific threshold α is permissible.

Here let the fuzzy instruction be given in the same way as in the case of (a) mentioned previously, and let us discuss in more detail on the execution procedures for this case.

[a-1] MAX-Method

A set of fuzzy instructions named "Go about L meters" can be specified by the membership function. Here, setting a threshold $\alpha(1 \geq \alpha \geq 0)$, the membership function $w(\ell)$ is truncated at a point η where $w(\eta) = \alpha$ and at another point η' where $w(\eta') = \alpha$ as shown in Figure 2.

The driver goes ahead η meters without condition. Upon arriving at $\ell = \eta$ he selects the distance with the highest grade of membership in the interval $[\eta,\eta']$. If it is not available, the distance with the second highest grade is selected. Such a method is designated MAX-Method and η is named a "Lowest Bound."

[a-2] PROB-Method

The fuzzy instruction is chosen with a probability $p(\ell)$

which is proportional to the grade of membership $w(\ell)$ in the interval $[\eta,\eta']$. This method is called PROB-Method.

[a-3] *-Method

The drivers goes ahead step by step to the point where the next instruction will be executable. This is named *-Method and is a specific type of 3 mentioned previously.

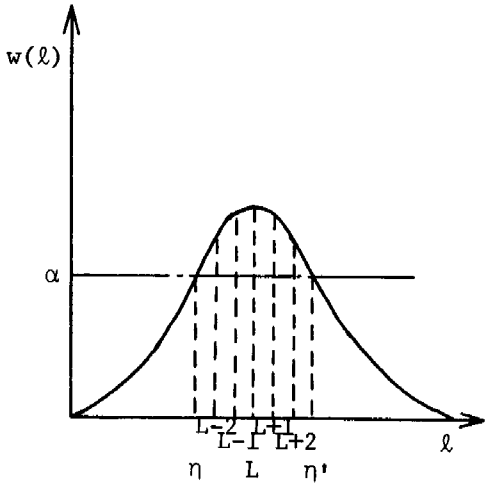


Figure 2 Membership Function w(ℓ) for a Set of Fuzzy Instructions named "Go about L Meters"

In the next section, we shall present an example of simulation of human drivers to whom the driving instructions are given step by step as in the case of (a).

4.3 Simulation of Human Driver's Behavior Directed by Fuzzy Instructions

In our example, the map is digitized as shown in Fig. 7, where the unit scale of the coordinates is equal to 10 meters, the direction allowable to the driver is quantized in the eight directions and the symbols on the map are illustrated in Fig. 3. The initial position is given by a triplet (x, y; d), where (x, y) is the coordinates of the initial position of the driver and d indicates the direction of the driver.

The procedure to rewrite a fuzzy instruction into a quasi-fuzzy instruction is as follows.

(i) "Go about L meters" $\sigma_{go}(L)$

The fuzzy instruction $\sigma_{go}(L)$ is divided into two instructions such as $\sigma_{go}^{\eta}(L)$ and $\sigma_{go}^{Y-\eta}(L)$, where $\sigma_{go}^{\eta}(L)$ is the instruction such that the driver goes on until the lowest bound η meters and $\sigma_{go}^{Y-\eta}(L)$ is the instruction such that the driver goes $Y - \eta$ meters.* Here note that $\sigma_{go}^{\eta}(L)$ is executed by the η times of quasi-fuzzy instruction σ_{go} named "Go ahead one step", $\sigma_{go}^{Y-\eta}(L)$ is executed by the $Y - \eta$ times of σ_{go}, and Y is determined by making use of one of the MAX-Method, PROB-Method and *-Method.

Hereupon the membership function for the set named "Go about L meters" is given by the following equation (18),

$$w(\ell) = \frac{1}{1 + \left(\frac{\ell - L}{a}\right)^2} \tag{18}$$

where $a = kL \quad (1 > k > 0)$, (19)

k may seem to denote a "Parameter Representing Distance-Sense" in the meaning that the driver is said to be sensitive to distance when k is small.

(ii) "Turn right" σ_R

(iii) "Turn left" σ_L

σ_R and σ_L are both found also in the quasi-fuzzy instructions defined already.

(iv) "Go straight" σ_{go}*

σ_{go}* can be assumed to be the succession of the quasi-fuzzy instruction σ_{go} named "Go ahead one step" until the next State-testing Fuzzy Instruction such as σ_R, σ_L and $\sigma\{\sim\}$ in our case will be executable.

(v) "Until $\sim$" $\sigma\{\sim\}$

This fuzzy instruction is rather regarded as a state-

---

* In our experiment, one step is made equal to one meter.

testing instruction and does not need to be converted into a quasi-fuzzy instruction. The interpretation of this instruction is to examine whether the present location of the driver is coincident with the destination or not by comparing the present coordinate with that of the destination stored in the machine. Fig. 3 shows a table of the destination to be stored in the machine.

```
+--------- *** SYMBOLS ON THE MAP *** ---------+
I                                                !
I  ( H ) HALT POINT          ( + ) CROSSING      I
I  ( S ) STARTING POINT      ( P ) PASSED POINT  I
I  ( * ) OBJECT POINT                            I
I    SG / TRAFFIC SIGNAL       RE / RESTAURANT   I
I    BK / BANK                 SC / SCHOOL       I
I    GS / GAS STATION          MP / PARKING LOT  I
I                                                I
+------------------------------------------------+
```

Destination	Coordinates
Traffic Signal	(20,05),(40,12), (20,35),(50,35)
Restaurant	(50,07),(40,09), (40,15)
Gas Station	(31,12),(50,29)
Parking Lot	(20,20),(50,25)
Bank	(43,24),(35,40), (45,40)
School	(10,05),(50,55)

Figure 3 The Symbols and the Coordinates of Destinations on the Map

Thus the five kinds of fuzzy instructions are converted into a sequence of the three kinds of quasi-fuzzy instructions in (16).

Then the machine instruction μ_i (i=1,2,...,8) which is really executable in the machine is selected by reference to the evaluation value $\Phi(\beta, s)$ accompanied with the pair of the

present quasi-fuzzy instruction β and the quasi-internal state s depending on the present position of the driver. The following Fig. 4 gives such an evaluation table where the integer of two figures shows that the machine instruction indicated by the second order figure is more preferable than that in the first order, and the integer of only one figure shows that the machine instruction indicated by that figure can be executable.

After all, summarizing the above argument, the execution procedure for a simulation of human driver's behavior directed by fuzzy instruction is illustrated by the flow chart shown in Fig. 5.

4.4 Computer Simulation Example for Human Driver's Behavior

Let such a sequence of fuzzy instructions as shown in Fig. 6 be given to a driver. This instruction means that (i) the driver starts from the point (50, 55), (ii) turns left at the branch point of y = 31, (iii) stops in the bank at (43, 24), (iv) turns right at the crossing of (40, 21), (v) drops in the restaurant at (40,15), (vi) and then goes on until the school at (10, 5) after turning left at the three-fork of (40, 5).

The computer simulation executed by the respective method of MAX-, PROB- and *-Method for the sequence of fuzzy instructions given above is exemplified in Figures 7, 8, and 9.

As can be seen from Figures 7, 8, and 9, the MAX-Method is most efficient to get to the destination, while the PROB-Method causes the driver to loiter around the same point and the *-Method lets him try such points as seem not to be concerned.

Such relative qualities of the three methods as mentioned above are supposed to be true from some results of simulation conducted with respect to some different kind of sequences of

Quasi-interval State s		Quasi-fuzzy Instruction β		
		σ_{go}	σ_R	σ_L
1		01	0	0
		05	0	0
2		02	0	0
		06	0	0
3		03	0	0
		07	0	0
4		04	0	0
		08	0	0
5		01	0	0
		0	0	0
6		03	0	0
		0	0	0
7		05	0	0
		0	0	0
8		0	0	0
		07	0	0
9		0	07	0
		0	0	05
		05	0	07
		07	05	0
10		01	0	03
		03	01	0
		0	03	0
		0	0	01
11		02	07	0
		02	0	0
		07	0	0
		07	0	0
12		01	0	03
		03	01	05
		05	03	0
		0	05	01

Ouasi-interval State s		Quasi-fuzzy Instruction β		
		σ_{go}	σ_R	σ_L
13		01	07	0
		0	01	05
		05	0	07
		07	05	01
14		0	07	03
		03	0	05
		05	03	07
		07	05	0
15		34	0	04
		43	03	07
		07	04	0
		07	07	03
16		18	08	0
		05	01	05
		05	0	08
		81	05	01
17		01	07	03
		03	01	05
		05	03	07
		07	05	01
18		34	08	04
		43	03	07
		78	04	08
		87	07	03
19		01	76	03
		13	71	03
		03	01	06
		06	03	76
		67	03	71
		76	06	01

Mark ↑ shows the direction of the driver.

Figure 4 Example of Evaluation Table in the Simulation

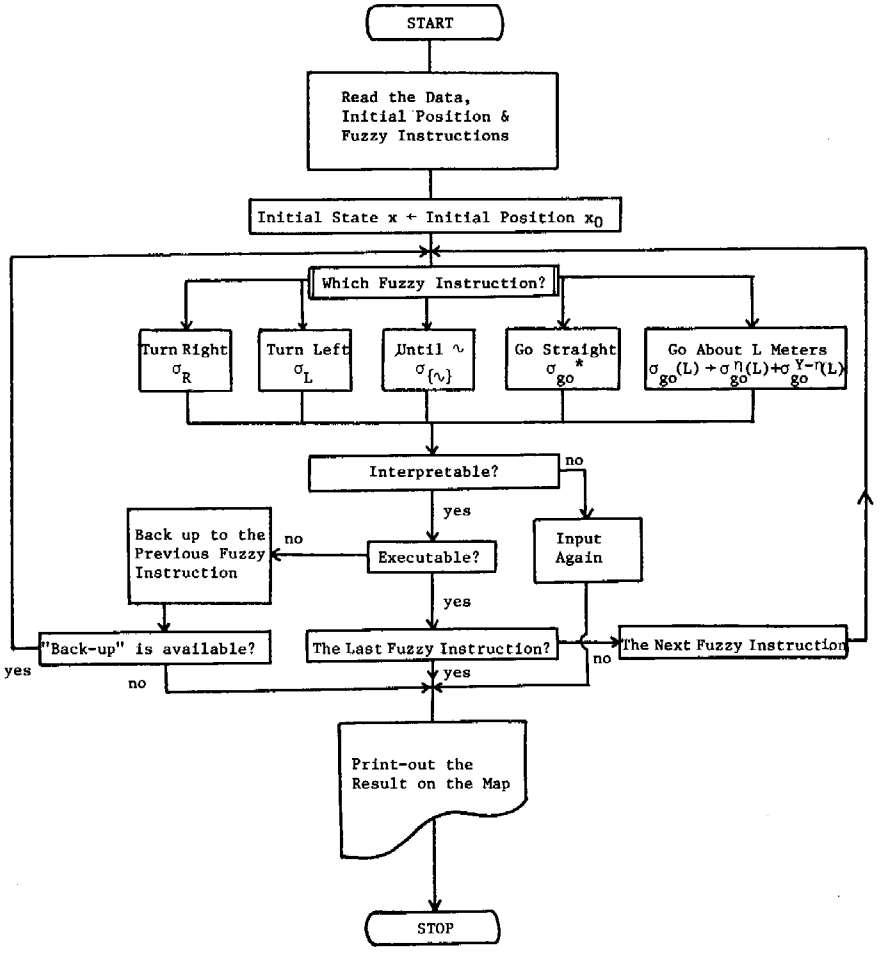


Figure 5 *Flow Chart for Simulation of Human Drivers's Behavior (Remark: This chart can be available to MAX-, PROB-, and *-Method, but here note that* $\sigma_{go}(L) \to \sigma_{go}^{\eta}(L) + \sigma_{go}^{*}(L)$ *is used for the *-Method.)*

```
*** INITIAL POSITION ***

      ( 50, 55, 3 )

** THE FUZZY INSTRUCTIONS **

      ( 1)  GO ABOUT 200 METERS
      ( 2)  TURN LEFT
      ( 3)  GO ABOUT 100 METERS
      ( 4)  UNTIL BANK
      ( 5)  GO ABOUT  50 METERS
      ( 6)  TURN RIGHT
      ( 7)  GO ABOUT  50 METERS
      ( 8)  UNTIL RESTAURANT
      ( 9)  GO ABOUT 100 METERS
      (10)  TURN LEFT
      (11)  GO STRAIGHT
      (12)  UNTIL SCHOOL
```

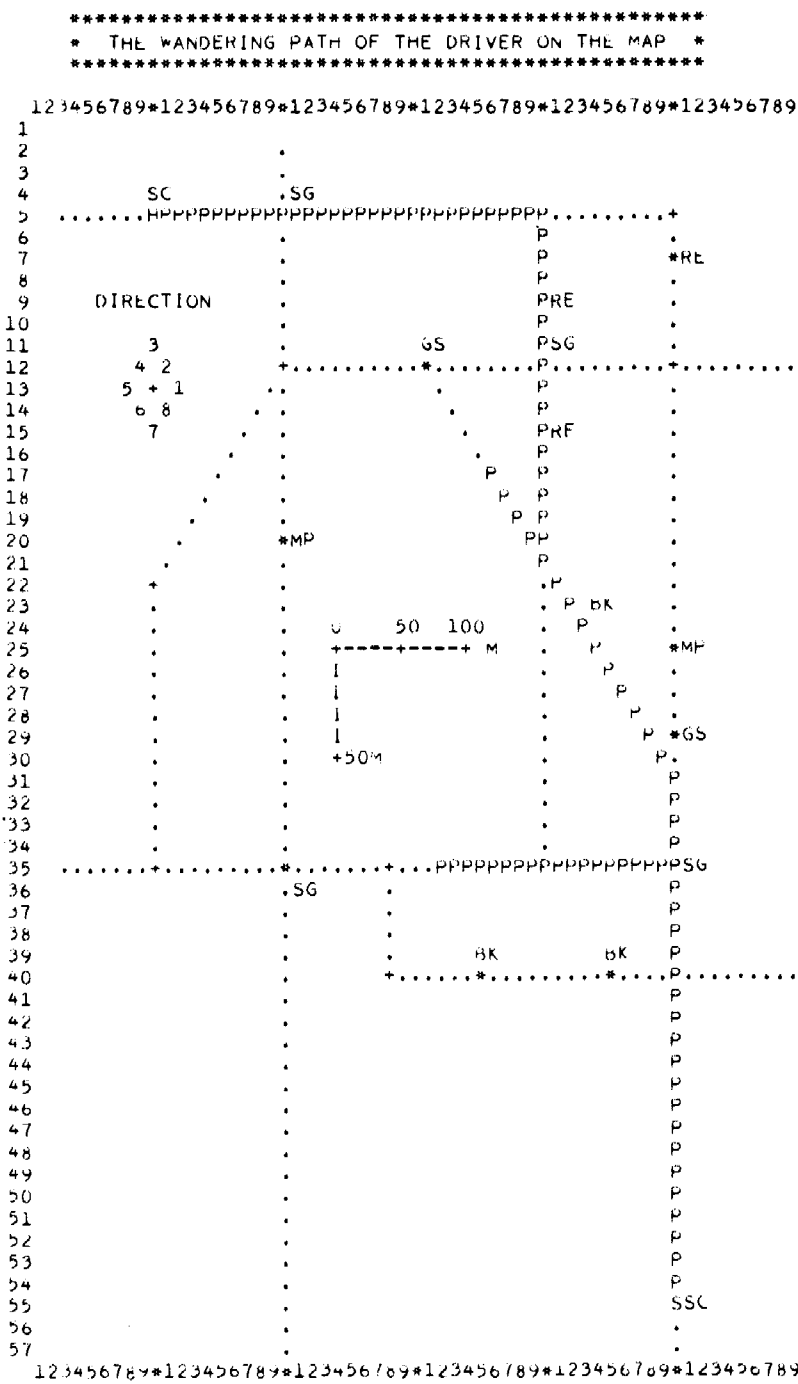
Figure 6 Initial Position and Fuzzy Instructions given in the Simulation

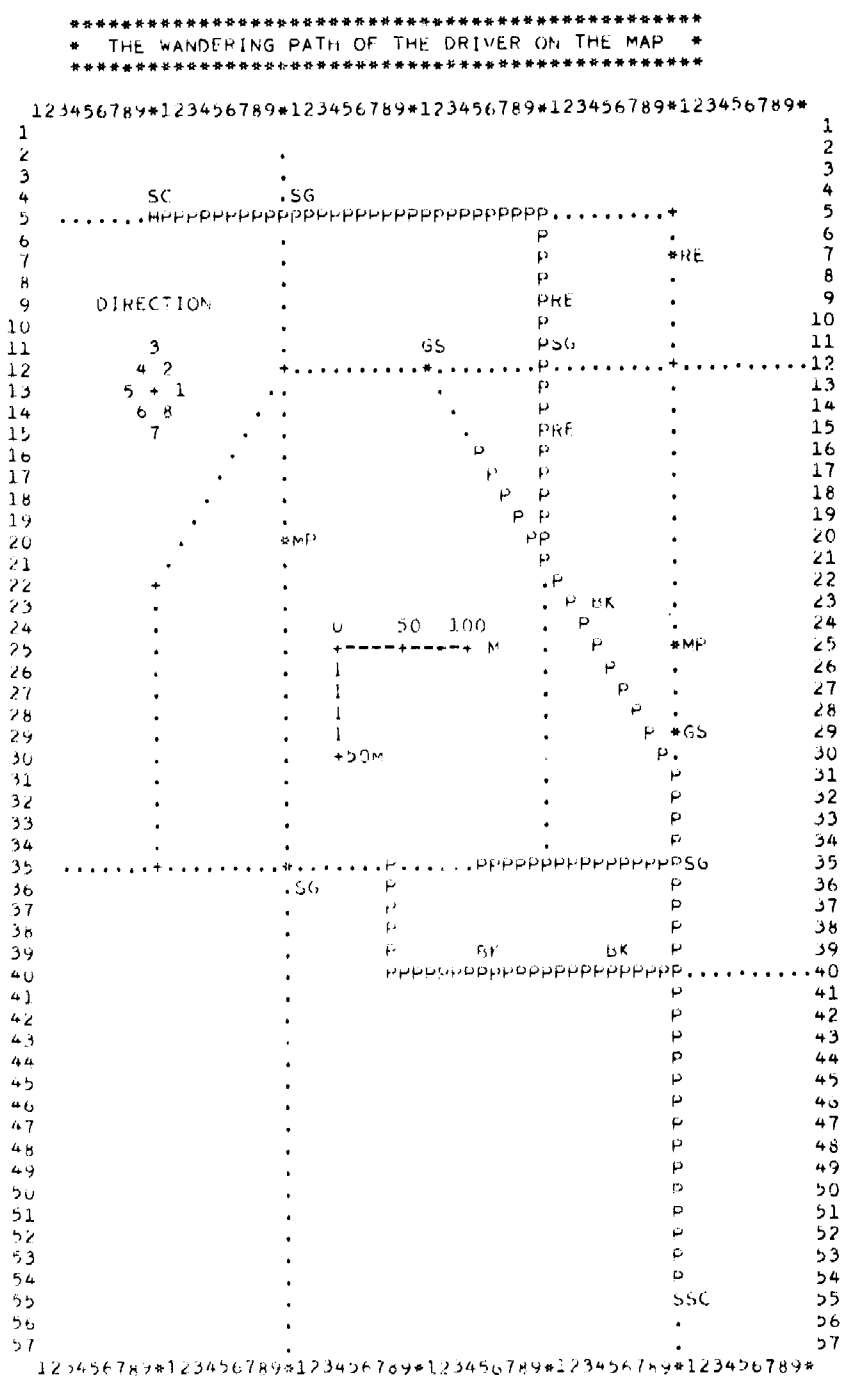
fuzzy instructions and various values of parameter representing distance-sense.

5. SIMULATION OF CHARACTER GENERATION

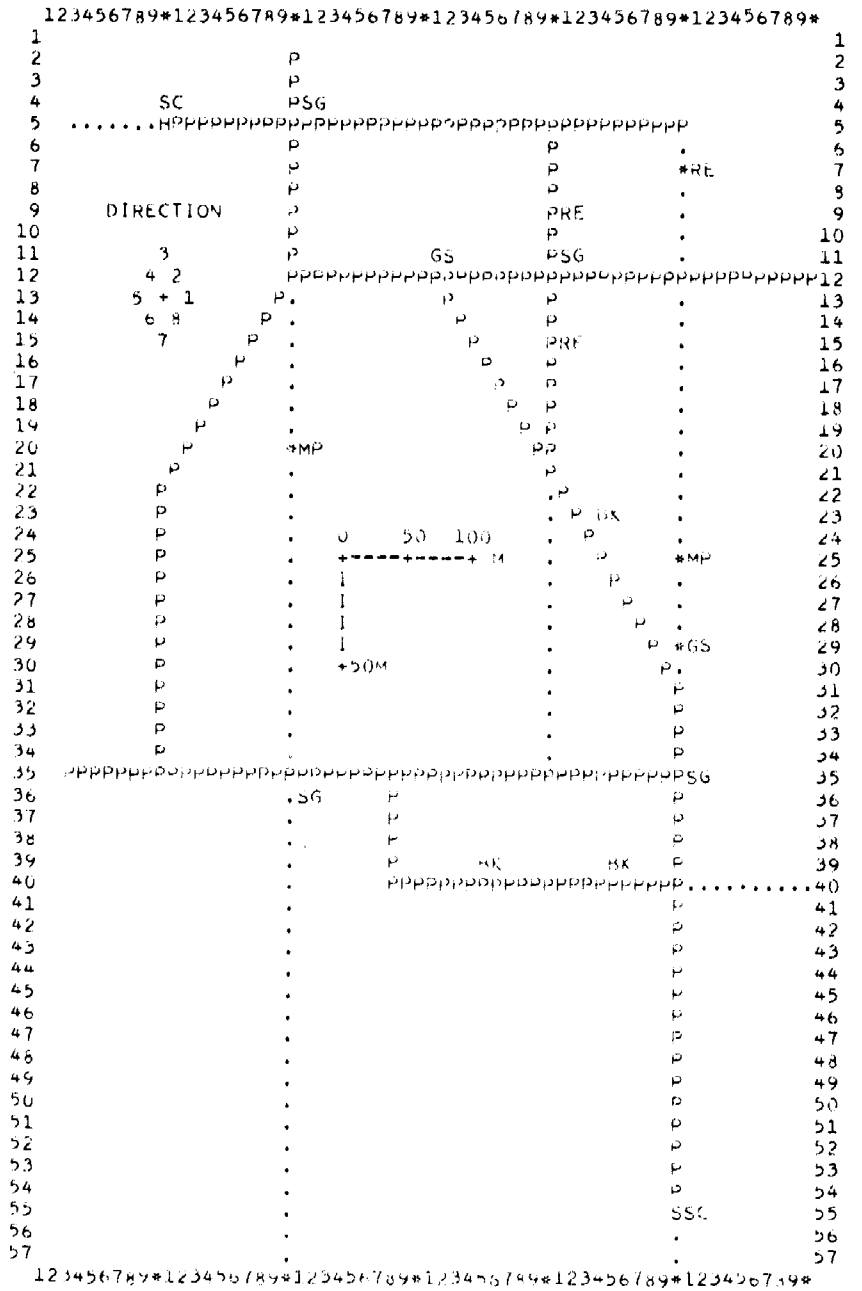
Let us now consider the process that a child starts to learn how to write characters and he will be good at writing. At the beginning, he is taught to write characters by his parents or his teachers. As is usual with this case, the teacher will teach him how to write a character in a rough way without measuring length, inclination and other features of strokes in the character or will set him a copy of a correct character. The above statement may seem to show that a child learns how to write a character based on a kind of rough rather than complete and correct informations about the character.

Thus a child writes a character following the instructions of his teacher, and then his teacher lets him correct the character by giving such ambiguous instructions as "make here a little shorter", "write round a little" and so on.


THE WANDERING PATH OF THE DRIVER ON THE MAP
SC
SG
DIRECTION
3
4 2
5 + 1
6 8
7
GS
MP
RE
SG
RF
BK
BK
50
100
M
50M
MF
GS
SG
SSC


THE WANDERING PATH OF THE DRIVER ON THE MAP
SC
SG
DIRECTION
3
4 2
5 + 1
6 8
7
GS
PSG
PRE
*RE
*MP
BK
0 50 100
M
+50M
*GS
SSC

```
**************************************************
*  THE WANDERING PATH OF THE DRIVER ON THE MAP   *
**************************************************
```



Through repeated practice in this manner, the child will gradually become to be able to write the characters in a correct way.

In this chapter, we shall conduct a computer simulation of the process of human learning stated above by making use of the concept of fuzzy program and learning algorithm. Let us provide the four kinds of fuzzy instruction as follows.

(i) Start (to write) from a point nearby (x, y).

(ii) Turn by about ρ degrees.

(iii) Draw about κ steps.

(iv) If the end point is not close to (x', y')
then do "Back-up".

Then let us adopt the following three types of procedure for executing a fuzzy program composed of the fuzzy instructions given above.

(I) MAX-Method,

(II) PROB-Method with Simple Modification, (21)

(III) PROB-Method with Reinforced Modification.

5.1 Execution Procedures of Fuzzy Instructions

The execution procedure of a fuzzy program for generating a character will be similar in general to that in a simulation of human driver. However, there exists a slight discrepancy that fuzzy instructions in this case are converted directly into machine instructions while those in a simulation of human drivers are translated into machine instructions after converting once into quasi-fuzzy instructions.

As a matter of fact, in the latter case there exist some constraints that the driver has to go ahead step by step

---

Figure 7 Trail of Driver on Map Simulated by MAX-Method
Figure 8 Trail of Driver on Map Simulated by PROB-Method
*Figure 9 Trail of Driver on Map Simulated by *-Method*

looking up the destination on the map and that he can not start to walk from quite a different point from the present location, but he must go on the road consequtively without skip. On the other hand, in the former case there is no constraint excepting the space limit for writing a character and also there is no necessity for drawing a line step by step. Therefore, in a simulation of human driver, if there is a fuzzy instruction which is not executable, the driver is forced to turn back to that previous to the present fuzzy instruction, while in a generation of a character there is no such a constraint but the interpretation may proceed from any instruction which is not always previous to the present instruction. This matter may seem to correspond to the fact that if there is an incorrect portion in a character, we can erase that portion and rewrite it in a correct way.

The mode to select the machine instruction is specified as follows in the same way as in the simulation of human drivers, that is,

(a) MAX-Method

(b) PROB-Method

(c) Non-deterministic Method.

In case where a fuzzy instruction is not executable, the following Back-up procedures are provided.

(1) Turn back to the fuzzy instruction previous to the present one.

(2) Turn back to the fuzzy instruction corresponding to the machine instruction with the lowest grade of membership in a series of machine instructions selected consequtively up to now.

(3) The "Back-up" procedure is the same as (2). However, as for the selection of the machine instruction corresponding to the fuzzy instruction to which

"Back-up" was done, the machine instruction with the higher grade of membership than before is to be selected, though in (2) the selection is made without constraint as above.

Procedure (1) is the same as used in a simulation of human driver. Procedure (2) is available to the case where correction is made from the worst portion in a character written. And procedure (3) is used in the case where we try to rewrite better than before.

By combining the selection mode of machine instructions with the "Back-up" methods, there can be obtained a variety of execution procedures of a fuzzy program. For instance, combining the MAX-Method (a) with the Back-up Method (1), we can obtain the MAX-Method in Equation (22) and also, combining the PROB-Method (b) with the Back-up Method (1), we can obtain the PROB-Method as discussed in the previous chapter.

In this chapter, as is shown in Equation (21), let us use the MAX-Method, PROB-Method with Simple Modification which is a combination of (b) and (2), and PROB-Method with Reinforced Modification which is a combination of (b) and (3).

5.2 Simulation of Character Generation

A sypical set of fuzzy instructions for generating, for example, a character "A" is given in Fig. 10. The meaning of this sequence is illustrated in Fig. 11, where the mark of wavy underline "$\underset{\sim}{x}$" shows that $\underset{\sim}{x}$ is approximate to x and the degree of an angle measured anti-clockwise is positive and that measured clockwise is negative. And also the instruction named "TURN BY" indicates to turn by ρ degrees from the present direction.

```
NO.     FUZZY INSTRUCTIONS

 1      START FROM A POINT NEARBY (0,75).
 2      TURN BY ABOUT (150) DEGREES.
 3      DRAW ABOUT (200) STEPS.
 4      START FROM A POINT NEARBY (-5,75).
 5      TURN BY ABOUT ( 60) DEGREES.
 6      DRAW ABOUT (190) STEPS.
 7      START FROM A POINT NEARBY (-50,-10).
 8      TURN BY ABOUT ( 60) DEGREES.
 9      DRAW ABOUT ( 90) STEPS.
10      IF THE POINT IS NOT CLOSE TO (45,-10) THEN BACK UP.
11      END.
```

Figure 10 Fuzzy Instructions for Generation of "A"

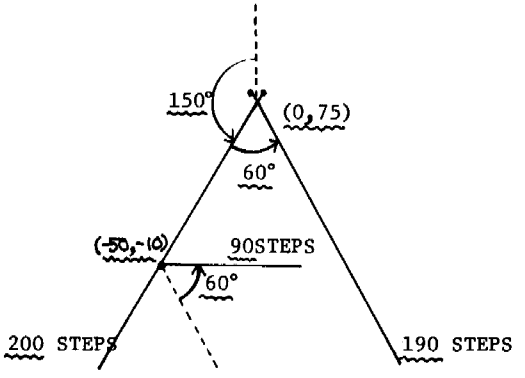


Figure 11 Example Illustrating the Meaning of Fuzzy Instructions

Plotted is in Figs. 12, 13, and 14, respectively, an example of simulation for generating a character, say, "A" by making use of the three types of execution procedures of fuzzy instructions shown in Equation (21). The only one conditional statement among the fuzzy instructions for generation of a character "A" is No. 10 in Fig. 10. If this condition is not satisfied, then "Back-up" is to be conducted. From Figs. 12, 13, and 14 we can see how "Back-up" is conducted in the respective method.

As is suggested from those figures, the execution of fuzzy programs employing the MAX-Method does not consume much

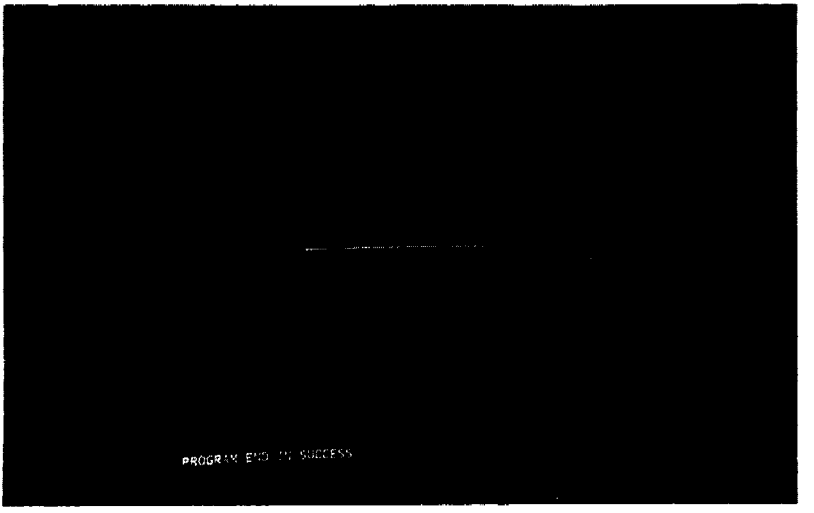


Figure 12 MAX-Method

Figure 13 PROB-Method with Simple Modification

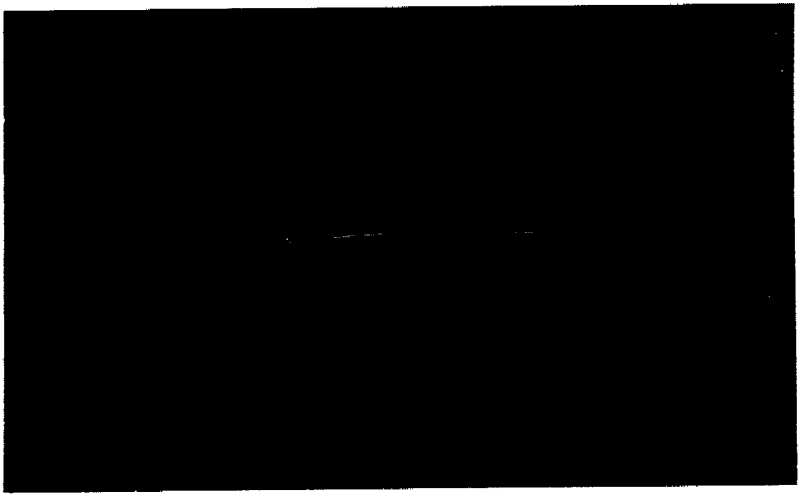


Figure 14 PROB-Method with Reinforced Modification

time in general if the parameters in the fuzzy instructions are correct. The reason is as follows. There is no contraint in the case of character generation unlike the case of simulation of driver's behavior where the driver is subject to some restriction depending on the road and others. Therefore the machine instructions selected in this case are equivalent to such parameters as coordinate, angle, number of steps and so on in the fuzzy instruction. On the other hand, the MAX-Method is not adequate to generation of characters in a free way. Form the fact that human does not always write the completely same character, this method may be said to be not so much suitable for character generation.

The PROB-Method with Simple Modification will be able to generate characters most freely. However, it consumes much time to execute fuzzy programs, because there is a possibility to select the machine instruction with a lower grade of membership than the previous one when the "Back-up" is conducted.

Finally, the PROB-Method with Reinforced Modification may seem to be the best one comparing with the other two methods

stated above. In fact, this method does not consume so much time to execute fuzzy programs and besides can generate characters fairly freely, because the machine instruction with a higher grade of membership than the previous one is to be selected compulsory when a "Back-up" occurred.

5.3 Character Generation with Learning Process

Incorrect portions of a character generated by a fuzzy program will usually be corrected also by "fuzzy correcting instructions." If there remain still incorrect portions in the character thus corrected, the same procedure will be repeated again and again. In this case, however, it should be suggested that the fuzzy program will be able to generate the more correct character faster than usual through a learning process which is encountered always in human study of writing.

5.3.1 Fuzzy Instruction for Correction & Learning Algorithm

Let the fuzzy instructions for correcting bad portions of a character be composed of the following instructions (a) - (h) and the following adjectives (i) - (k).

(a) LONG, (b) SHORT, (c) ANTICLOCKWISE, (d) CLOCKWISE,
(e) RIGHT, (f) LEFT, (g) UP, (h) DOWN
(i) VERY, (j) LITTLE, (k) MUCH

The learning algorithm used in this simulation is a linear reinforcement rule given by Equation (22).

$$w_{n+1}(x) = \lambda w_n(x) + (1 - \lambda)\chi_n \qquad (22)$$

where w_n denotes the grade of a parameter x at the n-th learning stage which is involved in the membership function specifying a fuzzy instruction and $0 \leq \lambda \leq 1$.

$$\chi_n = \begin{cases} 1, & \text{if x is adequate,} \\ 0, & \text{if x is not adequate,} \end{cases}$$

that is, if the parameter x (such as, say, a stroke length or

a stroke inclination, etc) modified by fuzzy correcting instructions is assumed to be adequate by the teacher, $\chi_n = 1$ and if it is assumed to be not adequate, $\chi_n = 0$.

In practice, bad portions of a character displayed on a graphic unit are corrected by fuzzy correcting instructions through a light pen as shown in Fig. 16. After cleaning out all of bad portions, the character thus corrected is displayed again and the instruction named "GOOD" is pointed out by a light pen if there is no bad portion.

It should be noticed that, in the present case, the initial fuzzy instructions for character generation are modified through learning process under supervision of the fuzzy instructions which is given to correct bad portions of a character.

5.3.2 Simulation of Learning Process

The fuzzy program to generate a character "B" is as shown in Fig. 15. Fig. 16 exemplifies the character "B" generated by the execution of this program by making use of the PROB-Method with Reinforced Modification.

Let us now correct bad portions of the character "B" thus displayed. This is simply performed as follows. By applying the fuzzy correcting instructions, the grade of membership of the fuzzy instructions given originally can be updated so as to generate the more correct character. This updating procedure is just a learning process and its algorithm is based on a linear reinforcement rule. On the lefthand side of Fig. 16 are shown the fuzzy correcting instructions used and Fig. 17 demonstrates the corrected character "B" by learning correction mentioned above.

Of course, the simulation has been performed also in the two cases of MAX-Method and PROB-Method with Simple Modification other than PROB-Method with Reinforced Modification

mentioned above.

```
NO.     FUZZY INSTRUCTIONS

 1      START FROM A POINT NEARBY (-20,70).
 2      TURN BY ABOUT (-180) DEGREES.
 3      DRAW ABOUT (150) STEPS.
 4      START FROM A POINT NEARBY ((-20,70).
 5      TURN BY ABOUT ( 90) DEGREES.
 6      DRAW ABOUT ( 60) STEPS.
 7      TURN BY ABOUT (-45) DEGREES.
 8      DRAW ABOUT ( 30) STEPS.
 9      TURN BY ABOUT (-45) DEGREES.
10      DRAW ABOUT ( 30) STEPS.
11      TURN BY ABOUT (-45) DEGREES.
12      DRAW ABOUT ( 30) STEPS.
13      TURN BY ABOUT (-45) DEGREES.
14      DRAW ABOUT ( 60) STEPS.
15      IF THE POINT IS NOT CLOSE TO (-20,-2) THEN BACK UP.
16      TURN BY ABOUT (-180) DEGREES.
17      DRAW ABOUT ( 60) STEPS.
18      TURN BY ABOUT (-45) DEGREES.
19      DRAW ABOUT ( 30) STEPS.
20      TURN BY ABOUT (-45) DEGREES.
21      DRAW ABOUT ( 40) STEPS.
22      TURN BY ABOUT (-45) DEGREES.
23      DRAW ABOUT ( 30) STEPS.
24      TURN BY ABOUT (-45) DEGREES.
25      DRAW ABOUT ( 60) STEPS.
26      IF THE POINT IS NOT CLOSE TO (-20,-80) THEN BACK UP.
27      END.
```

Figure 15 Fuzzy Instructions for Generation of "B"

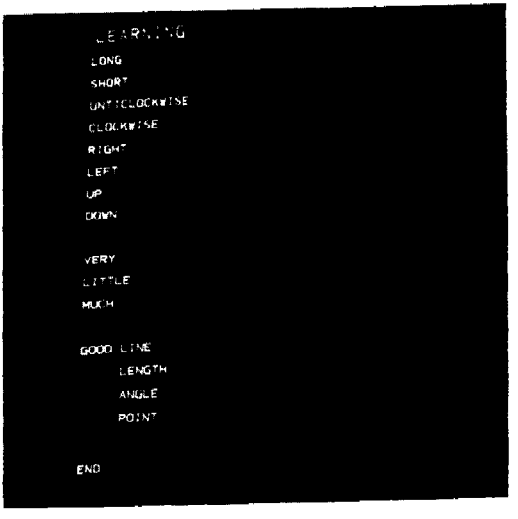


Figure 16 Character "B" Displayed Originally

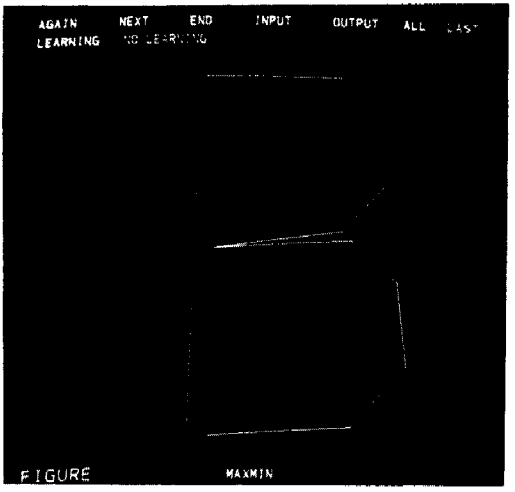


Figure 17 Character "B" Displayed after Learning Correction

Summarizing those simulation results, the following comparison can be obtained. The MAX-Method is not adequate to generate a character written freely while the number of correction procedures is small and it can generate a character as directed by the program. The PROB-Method with Simple Modification has a merit to be able to generate a freely-written character excepting that it requires a large number of correction procedures and not so enough learning effect can be expected. Contrary to the above two methods, the PROB-Method with Reinforced Modification may seem to be the best way because it can generate a fairly free character and none the less enough learning effect can be expected by a fairly few correction procedures.

6. CONCLUSION

In this paper, as an abstract model of a machine for the execution of fuzzy programs, a generalized fuzzy machine has been formulated from which a variety of fuzzy machines have been introduced.

Several methods of execution of fuzzy programs have been investigated by making use of the fuzzy machines introduced above. It has been pointed out that there exist three ways such as fuzzy- , probabilistic- and nondeterministic way depending on the specific character of the respective fuzzy machine with repsect to the way of selecting a machine instruction to a given fuzzy instruction and the way of state transition. Thereby, a unified survey for various execution methods of fuzzy programs can be obtained.

In addition, as some application examples using the presented methods for execution of fuzzy programs, the two simulation experiments such as human driver's behavior and character generation with learning process have been discussed. As the conclusion of this simulation, it has been found that in case of human driver's behavior the MAX-Method is best and in case of character generation the PROB-Method with Reinforced Modification is most favorable.

As a topic for further discussion, there remains an investigation of interpretation and execution methods of the more complicated fuzzy programs by making use of the concept of fuzzy semantics [4] as well as that of fuzzy algorithm [5].

REFERENCES

[1] S. K. Chang: "On the Execution of Fuzzy Programs Using Finite-State Machines," IEEE Trans. on Comp. Vol. C-21, No. 3, March 1972.

[2] E. S. Santos & W. G. Wee: "General Formulation of Sequential Machines," Inf. & Cont., Vol. 12, pp. 5-10, 1968.
E. S. Santos: "Maximin, Minimax and Composite Sequential Machines," J. Math. Annals & Appl. Vol. 24, pp. 246-259, 1968.

[3] R. Jakubowski & A. Kasprzak: "Application of Fuzzy Programs to the Design of Matching Technology," Bulletin de L'Academie, Polonaise des Sciences, Vol. XXI, No. 1,1973.

[4] L. A. Zadeh: "Quantitative Fuzzy Semantics," Inf. Sci. Vol. 3, pp. 159-176, 1971.
[5] L. A. Zadeh: "Fuzzy Algorithms," Inf. & Cont., Vol. 12, pp. 94-102, 1968.
[6] M. Mizumoto & K. Tanaka: "Various Kinds of Automata with Weights," J.CSS (to appear).