FUZZY ALGORITHMS

by

L. A. Zadeh

Memorandum No. ERL-M 236

16 January 1968

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

FUZZY ALGORITHMS

L. A. Zadeh

Department of Electrical Engineering and Computer Sciences
Electronics Research Laboratory
University of California, Berkeley 94720

I. INTRODUCTION

Unlike most papers in Information and Control, our note contains no theorems and no proofs. Essentially, its purpose is to introduce a basic concept which, though fuzzy rather than precise in nature, may eventually prove to be of use in a wide variety of problems relating to information processing, control, pattern recognition, system identification, artificial intelligence and, more generally, decision processes involving incomplete or uncertain data.

The concept in question will be called a fuzzy algorithm because it may be viewed as a generalization -- through the process of fuzzification -- of the conventional (non-fuzzy) conception of an algorithm.

More specifically, unlike a non-fuzzy deterministic or non-deterministic algorithm (Floyd, 1967), a fuzzy algorithm may contain fuzzy statements, that is, statements containing names of fuzzy sets

---

Research sponsored by the National Aeronautics and Space Admin. under Grant NsG-354, Suppl. 4.

Prof. Zadeh is on leave with the Department of Electrical Engineering and Project MAC, MIT, Cambridge, Mass.

(Zadeh, 1965), by which we mean classes in which there may be grades of membership intermediate between full membership and non-membership.

To illustrate, fuzzy algorithms may contain fuzzy instructions such as: (a) "Set y approximately equal to 10 if x is approximately equal to 5." Or (b) "If x is large, increase y by several units." Or (c) "If x is large, increase y by several units; if x is small, decrease y by several units; otherwise keep y unchanged." The sources of fuzziness in these instructions are fuzzy sets which are identified by their underlined names.

Familiar examples of fuzzy algorithms drawn from everyday experience are cooking recipes, directions for repairing a TV set, instructions on how to treat a disease, instructions for parking a car, etc. Generally, such instructions are not dignified with the name "algorithm." From our point of view, however, they may be regarded as very crude forms of fuzzy algorithms.

A fuzzy instruction which is a part of a fuzzy algorithm can be assigned a precise meaning by making use of the concept of the membership function of a fuzzy set. For example, in (a) the class of numbers which are approximately equal to 5 is a fuzzy set, say A, in the space of real numbers, R'. Similarly, the class of numbers which

---

* We use the symbol μ rather than f, as in (Zadeh, 1965), to denote a membership function. With this exception the notation and terminology used in this note follow that of (Zadeh, 1965).

are approximately equal to 10 is a fuzzy set, say B, in R'. These sets can be defined precisely by their respective membership functions $\mu_A(x)$* and $\mu_B(y)$ which associate with each x and y in R' their grades of membership in the fuzzy sets A and B. The grades of membership may be numbers in the interval [0, 1] or, more generally, points in a lattice (Goguen, 1966) or even a more general type of space. Clearly, such specifications are subjective in nature and, in general, reflect the context in which the problem is viewed.

Thus, the meaning of (a) can be made precise by specifying the membership functions of A and B. More generally, a statement such as (a) may be regarded as a binary fuzzy relation, say C, in R^2, which is characterized by a bivariate membership function $\mu_C(x, y)$. From this point of view, the fuzzy sets A and B are the shadows (Zadeh, 1965, 1966) of C on the coordinate axes 0x and 0y, respectively.

To gain a better understanding of the significance of C, it is helpful to visualize (a) as a fuzzified version of a statement such as (a') "Choose y in the interval [9.9, 10.1] if x is in the interval [4.9, 5.1]." In this case, A is the non-fuzzy set [4.9, 5.1], B is the non-fuzzy set [9.9, 10.1] and C is the two-dimensional interval [4.9, 5.1] × [9.9, 10.1] in R^2 whose projections on 0x and 0y are the intervals A and B, respectively.

Clearly, non-fuzzy instructions such as that cited above and its simpler version: (a") "Set y equal to 10 if x is equal to 5," convey no information on what should be done if x is not in [4.9, 5.1] or $x \neq 5$.

To provide such information in the latter case, for example, we must have a set of instructions like (a'') covering all possible values of x. Thus, if the domain of x is an interval, say Γ, then we must specify the pairs (x, y) for all $x \in \Gamma$. The result will be a graph, G, in R^2 which is the union of the ordered pairs (x, y), with x ranging over Γ.

Similarly, in the fuzzy case an instruction such as (a) may be regarded as a member, say C_x, of an indexed family of fuzzy sets $\{C_x\}$, with x ranging over a non-fuzzy set Γ. Then, the analog of the non-fuzzy graph G will be a fuzzy graph (relation), G, expressed by

$$G = \bigcup_x C_x \tag{1}$$

where the union * is taken over $x \in \Gamma$. This fuzzy graph may be visualized as a fuzzified version of the curve which depicts y as a function of x in the non-fuzzy case.

II. EXECUTION OF FUZZY INSTRUCTIONS

The above examples illustrate how a precise meaning may be assigned to a fuzzy instruction or to a family of such instructions by the use of the membership functions of the fuzzy sets which enter into such instructions. However, the assignment of a precise meaning to a fuzzy instruction does not in itself resolve the ambiguity of how it

---

* The union of two fuzzy sets A and B is the smallest fuzzy set which contains both A and B. The membership function of $A \cup B$ is given by $\mu_{A \cup B}(x) = \text{Max}[\mu_A(x), \mu_B(x)]$.

should be executed. The same is true, of course, of non-fuzzy instructions in a non-deterministic algorithm, in which the sources of ambiguity are statements of the form "Choose any x in a set A."

To illustrate the nature of the ambiguity of execution of a fuzzy instruction, consider the simple example of an unconditional instruction "Move several steps forward." Suppose that the membership function of A, the fuzzy set named "several steps," is specified as follows: $\mu_A(0) = \mu_A(1) = \mu_A(2) = \mu_A(3) = 0$; $\mu_A(4) = 0.8$; $\mu_A(5) = \mu_A(6) = \mu_A(7) = 1$; $\mu_A(8) = 0.7$; $\mu_A(x) = 0$ for $x \geq 9$. What would a human being do given $\mu_A(x)$ and instructed to move several steps forward, assuming that his actions are not influenced by any external factors such as the expenditure of energy involved in taking n steps, etc. ?

Needless to say, our consideration of the response of a human being to a fuzzy instruction is intended to provide an intuitive basis for formulating as a convention the way or ways in which such instructions should be executed. With this understanding, we are led to considering the following modes of execution.

I. Probabilistic execution. In this case, we assume that if a possible outcome x is assigned a grade of membership $\mu_A(x)$, then it should be chosen with a probability proportional to $\mu_A(x)$. Thus, in the above example, 0, 1, 2, and 3 would be chosen with probability 0; 4 with probability 0.8/4.5; 5, 6 and 7 with probability 1/4.5; 8 with probability 0.7/4.5; and 9, 10, ... with probability 0.

A variant of this mode of execution is one where a threshold α is set and all x's whose grade of membership is less than α are executed with probability 0. For example, if $\alpha = 0.8$, then 4 would be chosen with probability 0.8/3.8; 5, 6, and 7 with probability 1/3.8; and the rest with probability 0.

II. Non-deterministic execution with threshold. In this case, let A_α be the set of all x's such that $\mu_A(x) \geq \alpha$, where α denotes a specified threshold. Then, as in a non-deterministic algorithm, any x in A_α would constitute a permissible choice.

A special case of this mode of execution is one where α has the largest possible value, with the constraint that A_α be non-empty. In this case, A_α is the core of A, that is, the set of points which have maximal grade of membership in A.

The above modes of execution apply also to conditonal fuzzy instructions such as (b) "If x is large, increase y by several units." In this case, assume that x and y range over non-negative integers and that the fuzzy set, B, of "large x" is characterized by the membership function $\mu_B(x) = (1 + (\frac{x}{100})^{-10})^{-1}$. Let α be a specified threshold, say $\alpha = 0.9$, and let B_α be the set of all x such that $\mu_B(x) \geq 0.9$. Then, for each x in B_α, the instruction "Increase y by several units" will be executed as an unconditional instruction in the manner of (I) or (II). For x not in B_α, no execution will take place since the instruction does not cover this contingency.

Now suppose that instead of having a single instruction like (b) which is conditioned on a fuzzy set, we have a family of such instructions

$U = \{u_0, u_1, \ldots, u_m\}$ be its set of tape symbols. Now, in the case of a non-fuzzy deterministic Turing machine, the state at time $n+1$ is a function of the state at time n and the tape symbol at time n, i.e.,

$$q^{n+1} = f(q^n, u^n) \tag{2}$$

where f is a function from $Q \times U$ to Q and q^n and u^n are variables ranging over Q and U, respectively.

In the case of a non-fuzzy non-deterministic Turing machine, f in (2) is a multi-valued rather than a single-valued function. This is equivalent to saying that the dependence of q^{n+1} on q^n and u^n is described by a relation

$$R = \{(q^{n+1}, q^n, u^n)\} \tag{3}$$

where R is a subset of the product space $Q \times Q \times U$, rather than by a function, as in (2). It should be noted that a non-deterministic algorithm (Floyd, 1967), corresponds to a non-fuzzy non-deterministic Turing machine.

Now in the case of a fuzzy Turing machine, the relation (3) is a fuzzy rather than a non-fuzzy subset of the product space $Q \times Q \times U$. Such a fuzzy subset would be characterized by a membership function $\mu_R(q^{n+1}, q^n, u^n)$ which associates with each triplet (q^{n+1}, q^n, u^n) a grade of membership in the fuzzy relation R. Thus, a non-fuzzy non-deterministic Turing machine may be regarded as a special case of a fuzzy Turing machine in which $\mu_R(q^{n+1}, q^n, u^n)$ can take only two values,

1 or 0, according as (q^{n+1}, q^n, u^n) does or does not belong to R. From a more general point of view, a fuzzy Turing machine may be regarded as a special case of a fuzzy system (Zadeh, 1965).

In short, an algorithm corresponds to a Turing machine, a non-deterministic algorithm corresponds to a non-deterministic Turing machine, and a fuzzy algorithm corresponds to a fuzzy Turing machine. It should be noted, however, that the identification of an algorithm with a Turing machine restricts the applicability of the notion of an algorithm --whether fuzzy or not --to those situations in which the variables entering into the algorithm range over finite, or, at most, countable sets. Actually, it is common practice to use the term "algorithm" in a broader and more loose sense to describe recursive procedures in which the variables may range over continua, e.g., the simplex algorithm of linear programming, rather than just finite or countable sets. In this sense, an algorithm is a fuzzy algorithm when its variables range over fuzzy sets, regardless of whether they are finite sets or continua.

IV. RATIONALE FOR FUZZY ALGORITHMS

It is a truism that precision is respectable and fuzziness is not. However, in our quest for ever greater degree of precision in pure and applied science, we have perhaps tended to lose sight of one basic fact, namely, that the class of non-trivial problems for which one can find

precise algorithmic solutions is quite limited. Unfortunately, most realistic problems tend to be complex, and many complex problems are either algorithically unsolvable or, if solvable in principle, are computationally infeasible. For example, it is well known that, in principle, there is an optimal strategy for playing chess. In reality, however, it is completely beyond the capability of any conceivable computer to trace the decision tree of all possible moves and, using backward or forward iteration, decide on the best move at each stage of the game. Thus in chess, as in many other complex situations, fuzzy local goals must be substituted for the precisely specified terminal objective, and what on the surface appears to be a precise problem turns out to be a very fuzzy one.

Another illustration of a situation in which complexity rules out a precise algorithmic solution is provided by a problem drawn from everyday experience, namely, the problem of parking a car C in a space available between two cars parked at a curb.

A control theorist would formulate this problem as follows: Let w denote the position of a fixed reference point in C (e.g., the center of a rectangle which approximates to C) and let θ denote the orientation of C. Then the state of C can be identified with the vector $x = (w, \theta)$ and the differential equation of motion of C can be expressed in the form

$$\dot{x} = f(x, u) \tag{4}$$

where the control vector u is assumed to have two components, u_1 and u_2, representing respectively the angle of the front wheels and the speed of C. u_1 is a bounded variable and u_2 is assumed for simplicity to be capable of taking only three values α (in the forward direction), β (in the reverse direction), and zero.

The space available between the two cars defines a set of allowable terminal states Γ, and the two cars define a constraint set Ω from which x is excluded. The problem of parking the car, then, may be regarded as that of finding a strategy u(x) for transferring a specified initial state x^0 into Γ subject to the prescribed constraints on u and x.

When formulated in this precise form, the problem in question is too complex for solution even with the aid of large scale computers. Thus, for all practical purposes, the stated problem does not have a precise algorithmic solution. On the other hand, we know that by following a set of instructions of the type one gets in a driving school -- instructions which may be regarded as a crude form of a fuzzy algorithm -- an inexperienced driver can park his car in the available space without having precise information concerning the differential equation characterizing C, the constraint set Ω or the set of allowable terminal states Γ. Thus, by treating the parking problem as a fuzzy rather than a precise problem, one can formulate a fuzzy algorithm for solving it.

The crux of such an algorithm is the observation that the reference point w in C can be transferred in a lateral direction by performing the

following maneuver: First, while the car is moving forward, the wheels are turned to the right and then to the left; and second, the direction of motion is reversed and the wheels are turned first to the left and then to the right. By repeating this maneuver as many times as necessary, C can be moved in a lateral direction by any desired amount.

If we were not familiar with how to park a car, the finding of a fuzzy algorithm for solving the problem would not be entirely trivial. The development of systematic procedures for finding fuzzy algorithms and the recognition that in many realistic problems it is not practicable to search for non-fuzzy algorithms may in time confer upon fuzzy algorithms the respectability which they lack at present.

It should be noted that, in order to lead to a fuzzy solution of a problem such as that of parking a car, a fuzzy algorithm must be robust in the sense that its success should not depend on the knowledge of the precise meaning of its instructions, that is, on the precise specification of the membership functions of the fuzzy sets entering into such instructions. Indeed, the property of robustness in the above sense constitutes an essential characteristic of a fuzzy algorithm.

It is of interest to note that, in addition to providing a possible way of approaching complex control problems, fuzzy algorithms might be useful in defining fuzzy sets of objects such as, for example, the class of handwritten versions of the letter script A or the cardiograms associated with a particular disease of the heart. In such cases, the

algorithm would serve as a fuzzy algorithm definition of a fuzzy set of objects, just as the differential equation $\ddot{x} + w^2 x = 0$ serves as a non-fuzzy algorithmic definition of the non-fuzzy class of sine waves of the form $a\cos(\omega t + \theta)$, where a and θ range over scalars.

What is the relationship between a fuzzy algorithm and a heuristic program? In effect, a heuristic program is a non-fuzzy approximation -- expressed in a computer language -- to a fuzzy algorithm. We have to employ such programs in implementing a fuzzy algorithm on a computer because present day computers cannot operate on fuzzy sets. It would be an advance of vast importance when we learn how to design machines that can understand fuzzy concepts in much the same way as human beings are capable of doing.

What we have said here about fuzzy algorithms is very preliminary in nature. Although it may be premature to say so at this juncture, this writer believes that the domain of applicability of systematic reasoning might be enlarged by the acceptance of fuzzy algorithmic solutions to both precise and imprecise problems. Clearly, there are many obvious questions about fuzzy algorithms that we have not posed, much less attempted to answer in this note. It is possible to prove theorems about fuzzy algorithms. But the real challenge is to discover those, possibly fuzzy, properties of such algorithms which do not depend on the precise specification of the fuzzy sets which enter into their fuzzy instructions, nor on the precise manner in which such instructions should be executed.

REFERENCES

Floyd, R. W., (1967), Nondeterministic Algorithms, J. Assoc. Comp. Mach. 14, 636-644.

Zadeh, L. A., (1965), Fuzzy Sets, Inform. Control 8, 338-353.

Goguen, J. A., (1967), L-Fuzzy Sets, J. Math. Anal. and Appl. 18, 145-174.

Aizerman, M. A., Gusev, L. A., Rozonoer, L. I., Smirnova, I. M., and Tal, A. A., (1963), "Logic, Automata and Algorithms," Fizmatgiz, Moscow.

Korfhage, R. R., (1966), "Logic and Algorithms," J. Wiley and Sons, New York.

Zadeh, L. A., (1966), Shadows of Fuzzy Sets, Problems in Trans. of Information, 3, (37-44).

Zadeh, L. A., (1965), Fuzzy Sets and Systems, Proc. Symp. on System Theory, Polytechnic Inst. of Brooklyn, 29-37.