7. Given the following rules an
approach to determine the le

You are definitely smart.
You may have some talen
You will probably accept a pay cut.
You will definitely not move.

IF You are smart THEN You get the job CF 0.8
IF You have talent THEN You get the job CF 0.7
IF You will take a pay cut THEN You get the job CF 0.6
IF You will move THEN You get the job CF 0.5

8. Using the following rules and information, determine investment recommendations.

CF(Interest rate low) = 0.7
CF(Inflation low) = 0.6
CF(Employment good) = 0.6
CF(Stock prices high) = 0.6
CF(Interest rate high) = 0.2
CF(Inflation high) = 0.5
CF(Stock prices low) = 0.4

R1 IF Interest rate low
AND Inflation low
THEN Market steady CF 0.9

R2 IF Interest rate high
AND Inflation high
THEN Market unsteady CF 0.8

R3 IF Market steady
OR Employment good
THEN Buy stocks CF 0.8

R4 IF Stock prices low
THEN Buy stocks CF 0.7

R5 IF Market unsteady
AND Employment good
THEN Buy bonds CF 0.9

R6 IF Bond prices low
THEN Buy bonds CF 0.6

9. Discuss the advantages and disadvantages of the CF approach to managing a situation where many sources confirm a hypothesis while only one disconfirms it.
10. Discuss several ways the CF value can be used to control the search.
11. Describe a problem that has both certain and uncertain issues.
12. Describe techniques for acquiring the CF values from the expert.
13. Why should deep inference chains be avoided?

CHAPTER 13

Fuzzy Logic

INTRODUCTION

Experts often rely on *common sense* to solve problems. We see this type of knowledge exposed when an expert describes a problem using vague or ambiguous terms. For example, the expert might state, "When the motor is running *really hot* I decrease the speed *a little*." We are accustomed to hearing a problem described in this manner, and usually have little difficulty with interpreting the use of the vague terms. However, providing a computer with the same understanding is a challenge—How can we represent and reason with vague terms in a computer?

This chapter answers the question by exploring the subject of **fuzzy logic.** We first review the basic ideas behind fuzzy logic and then look at its mathematical formalism. We then see how fuzzy logic can be employed in an expert system. The chapter also provides a structured approach for designing a fuzzy expert system and illustrates the process through the design of an example system.

OVERVIEW OF FUZZY LOGIC

Fuzzy systems have been around since the 1920s, when they were first proposed by Lukasiewicz (the inventor of reverse Polish notation) (Rescher 1969). Lukasiewicz studied the mathematical representation of "fuzzy" terms such as *tall*, *old*, or *hot*. His motivation for the work came from an understanding that these types of terms defied a truth representation in the two-valued [0,1] Aristotelian logic: true or false.

He developed a system of logic that extended the range of truth values to all real numbers in the range of 0 to 1. He used a number in this set to represent the *possibility* that a given statement was true or false. For example, the possibility that a person 6 feet tall is really tall might be set to a value of 0.9; it is *very likely* that the person is tall. This research led to a formal inexact reasoning technique aptly named **possibility theory.**

In 1965, Zadeh (1965) extended the work on the possibility theory into a formal system of mathematical logic. Possibly more important, Zadeh brought to the attention of scientists and engineers a collection of valuable concepts for working with *fuzzy* natural language terms that were almost exclusively in the hands of academic philosophers for the past forty years. This new logic tool for representing and manipulating fuzzy terms was called **fuzzy logic.**

DEFINITION 13.1: Fuzzy Logic
A branch of logic that uses degrees of membership in sets rather than a strict true/false membership.

Linguistic Variables

Fuzzy logic is primarily concerned with quantifying and reasoning about vague or fuzzy terms that appear in our natural language. In fuzzy logic, these fuzzy terms are referred to as **linguistic variables** (also called fuzzy variables).

TABLE 13.1 Examples of Linguistic Variables With Typical Values

Linguistic Variable	*Typical Values*
temperature	hot, cold
height	short, medium, tall
speed	slow, creeping, fast

DEFINITION 13.2: Linguistic Variable
Term used in our natural language to describe some concept that usually has vague or fuzzy values.

For example, in the statement "Jack is young," we are saying that the implied linguistic variable **age** has the linguistic value of **young.** Table 13.1 shows other examples of linguistic variables and typical values that we might assign to them.

In fuzzy expert systems, we use linguistic variables in *fuzzy rules*. A fuzzy rule infers information about a linguistic variable contained in its conclusion from information about another variable contained in its premise. For example:

Rule 1
IF Speed is slow
THEN Make the acceleration high

Rule 2
IF Temperature is low
AND Pressure is medium
THEN Make the speed very slow

We call the range of possible values of a linguistic variable the variable's **universe of discourse.** For example, we might give the variable "speed" used in RULE 1 the range between 0 and 100 mph. The phrase "speed is slow" occupies a section of the variable's universe of discourse—it is a fuzzy set.

Fuzzy Sets

Traditional set theory views the world as black or white. That is, an object is either in or not in a given set. Consider for example a set consisting of *young* people, i.e., children. Traditional set theory would set a sharp boundary on this set and give each set member the value of 1, and all members not within the set a value of 0; this is called a crisp set. For instance, set members might consist of only those people whose age is less than 10. Using this strict interpretation, on a person's eleventh birthday their childhood suddenly vanishes.

Fuzzy logic provides a more reasonable interpretation of *young* people using a **fuzzy set.** A fuzzy set assigns membership values between 0 and 1 that reflect more naturally a member's association with the set. For example, if a person's age is 5, we might assign a membership value of 0.9, or if the age is 13, a value

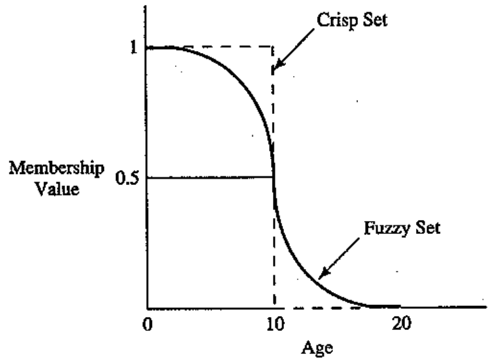


FIGURE 13.1 Fuzzy and crisp sets of "young" people.

of 0.1. In this example "age" is the linguistic variable and "young" one of its fuzzy sets. Other sets that we might consider are "old," "middle-age," etc. Each of these sets represent an *adjective* defined on the linguistic variable.

In general, a fuzzy set provides a graceful transition across a boundary as illustrated in Figure 13.1. The x-axis, or universe of discourse, represents a person's age. The y-axis is the fuzzy set membership value. The fuzzy set of "young" people maps age values into corresponding membership values. You can see from the figure that our 11-year-old person is no longer suddenly not a child. The person is gradually removed from this classification as his age increases.

Formally, we can define a fuzzy set as:

DEFINITION 13.3: Fuzzy Set
Let X be the universe of discourse, with elements of X denoted as x. A fuzzy set A of X is characterized by a membership function $\mu_A(x)$ that associates each element x with a degree of membership value in A.

In contrast to probability theory, which relies on assigning probabilities to a given event on the basis of prior frequencies of the event, fuzzy logic assigns values to the event on the basis of a **membership function** defined as:

$$\mu_A(x): X \rightarrow [0, 1] \qquad (1)$$

In fuzzy logic, event or element x is assigned a membership value by a membership function μ. This value represents the degree to which element x belongs to fuzzy set A.

$$\mu_A(x) = \text{Degree}(x \in A) \qquad (2)$$

The membership value of x is bounded by the following relationship:

$$0 \leq \mu_A(x) \leq 1$$

A fuzzy set is an extension of the traditional set theory. It generalizes the membership concept by using the membership function μ that returns a value between 0 and 1 that represents the **degree of membership** (also called membership value) an object x has to set A.

FORMING FUZZY SETS

To represent a fuzzy set in a computer we need to define its membership function. One approach that we can use is to poll a group of people for their understanding of the term that we are attempting to represent by the fuzzy set. For example, consider the concept of a *tall* person. We could ask each of these individuals to what degree they believe a person of a given height is *tall*. After acquiring answers for a range of heights, we could perform simple averaging to produce a fuzzy set of *tall* people. We can now use this function to ascribe a belief (or membership value) to a given individual that they belong to the fuzzy set of *tall* people.

We could continue this polling to account for other height descriptions such as *short*, or *medium*. In this fashion we can obtain fuzzy sets that reflect the popular opinion of most people for each of these classifications. This point is illustrated in Figure 13.2 where fuzzy sets are shown in a piecewise linear form for the issues of three different categories of an individual's height. When we define multiple fuzzy sets on the same universe of discourse, the fuzzy literature often refers to them as *fuzzy subsets*.

By forming fuzzy subsets for various vague terms, we can ascribe a membership value of a given object to each set. Consider Figure 13.2 again. An individual

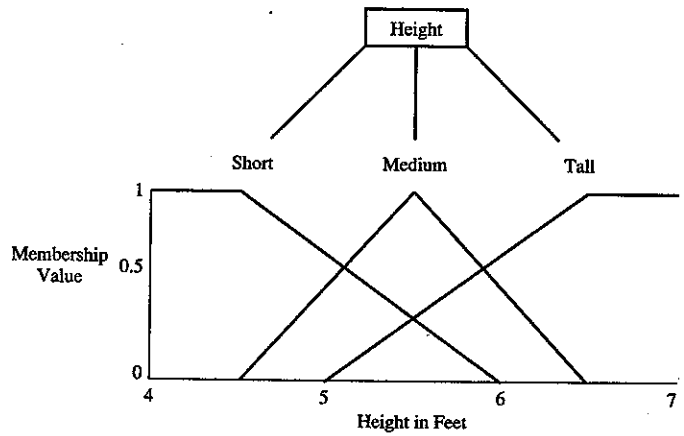


FIGURE 13.2 Fuzzy sets on height.

of a height of 5.5 feet is a member of *medium* persons with a membership value of 1, and at the same time a member of *short* and *tall* persons with a value of 0.25. This is an interesting result—a single object is considered a *partial* member of multiple sets. You will see the value of this point later when we review the operation of a fuzzy expert system.

Another approach often found in practice for forming a fuzzy set relies strictly on the interpretation of a single expert. Like the prior polling technique, you can ask the expert for his belief that various objects belong to a given set. Though this approach lacks the input from a wider audience, it does provide the expert's understanding of the concept.

A more recent approach described by Kosko (1992) relies on a neural network technique. This approach is usually seen in control type applications, where the neural network uses data on the system's operation to derive through a learning mode the form of the fuzzy set.

FUZZY SET REPRESENTATION

Prior sections introduced fuzzy sets and discussed their value in capturing quantitatively ambiguous terms used in our natural language. In this section, we introduce a formal representation of fuzzy sets.

Assume we have a universe of discourse X and a fuzzy set A defined on it. Further assume we have a discrete set of X elements $\{x_1, x_2, \ldots, x_n\}$. The fuzzy set A defines the membership function $\mu_A(x)$ that maps the elements x_i of X to the degree of memberships in [0,1] (see equation 1). The membership values indicate to what degree x_i belongs to A (see equation 2). For a discrete set of elements, a convenient way of representing a fuzzy set is through the use of a vector:

$$A = (a_1, a_2, \ldots, a_n) \tag{3}$$

where,

$$a_i = \mu_A(x_i) \tag{4}$$

For a clearer representation, the vector often includes the symbol "/" which associates the membership value a_i with its x_i coordinate:

$$A = (a_1/x_1, a_2/x_2, \ldots, a_n/x_n) \tag{5}$$

As an example, consider the fuzzy set of tall people shown previously in Figure 13.2:

$$\text{TALL} = (0/5,\ 0.25/5.5,\ 0.7/6,\ 1/6.5,\ 1/7)$$

Standard fuzzy set notation represents the union of the vector's dimensions as follows; where "+" represents the Boolean notation for union:

$$A = \mu_1/x_1 + \mu_2/x_2 + \ldots + \mu_n/x_n \tag{6}$$

or

$$A = \sum_{i=1}^{n} \mu_i/x_i$$

If X is a continuous function, then the set A can be represented as:

$$A = \int_x \mu_A(x_i)/x_i \tag{7}$$

For a continuous set of elements, we need some function to map the elements to their membership values. Typical functions used are sigmoid (see Figure 13.1), gaussian, and pi. These types of functions are smooth and can typically provide a close representation of the data that is the basis of the fuzzy set. However, these functions add to the computational load of the computer.

In practice, most applications rely on a piecewise linear fit function to represent the fuzzy set as illustrated in Figure 13.2. To capture a linear fit, we can code each fuzzy set into a **fit-vector.** For example, we can code the fuzzy set *tall* of Figure 13.2 in the vector (0/5, 0.3/5.5, 0.7/6, 1/6.5, 1/7). Mid-range fuzzy sets, such as *medium* of Figure 13.2, are usually represented by a triangular fit-vector function such as (0/4.5, 0.5/5, 1/5.5, 0.5/6, 0/6.5). Fuzzy logic development tools on the market often encode a triangular function using end-point and mid-point vector coding. For example, we could code the *medium* fuzzy set of Figure 13.2 using a vector of the form (0/4.5, 1/5.5, 0/6.5).

HEDGES

Prior sections described methods for both capturing and representing vague linguistic terms quantitatively through the use of a fuzzy set. In normal conversations, humans may add additional vagueness to a given statement by using adverbs such as *very*, *slightly*, or *somewhat*. An adverb is a word that modifies a verb, an adjective, another adverb, or a whole sentence. Consider for example an adverb modifying an adjective, "The person is *very* tall."

If we needed to represent this new fuzzy set, we could poll the same group of people used in obtaining our earlier fuzzy set on *tall persons*. However, techniques are available for working with an existing fuzzy set to capture the impact of an added adverb. This is the subject of **hedges.**

A hedge modifies mathematically an existing fuzzy set to account for some added adverb. For example, Figure 13.3 shows the three fuzzy sets on height shown previously in Figure 13.2, along with the sets adjusted by the introduction of the term *very* derived through an operation discussed later.

To illustrate the impact of a fuzzy set modified by a hedge operation, consider a person 6 ft tall. According to Figure 13.3 we consider this individual *tall* with a belief 0.7. However, as also shown in Figure 13.3, we consider the same

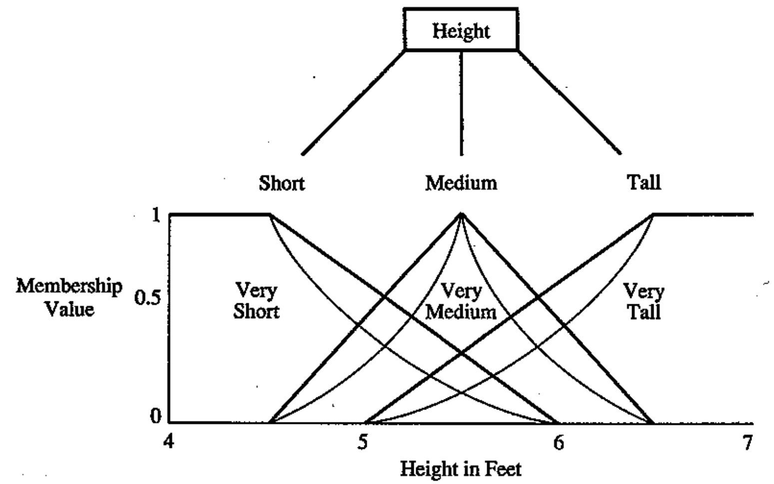


FIGURE 13.3 Fuzzy sets on height with "very" hedge.

individual *very tall* to a belief of 0.4—a reasonable result. The next sections discuss the hedges commonly used in practice.

Concentration (Very)

The concentration operation has the effect of further reducing the membership values of those elements that have smaller membership values. This operation is given as

$$\mu_{\mathrm{CON}(A)}(x) = (\mu_A(x))^2 \tag{8}$$

Given a fuzzy set of *tall persons*, we could use this operation to create the set of *very tall persons*.

Dilation (Somewhat)

The dilation operation dilates the fuzzy elements by increasing the membership value of those elements with small membership values more than those elements with high membership values. This operation is given as

$$\mu_{\mathrm{DIL}(A)}(x) = (\mu_A(x))^{0.5} \tag{9}$$

Given a fuzzy set of *medium persons*, we could use this operation to create the set of *more or less medium persons*.

Intensification (Indeed)

The intensification operation has the effect of intensifying the meaning of the phrase by increasing the membership values above 0.5 and decreasing those below 0.5. This operation is given as

$$\begin{aligned}\mu_{\mathrm{INT}(A)}(x) &= 2(\mu_A(x))^2 && \text{for } 0 \le \mu_A(x) \le 0.5 \\ &= 1 - 2(1 - \mu_A(x))^2 && \text{for } 0.5 < \mu_A(x) \le 1\end{aligned} \tag{10}$$

Given a fuzzy set of *medium persons*, we could use this operation to create the set of *indeed medium persons*.

Power (Very Very)

The power operation is an extension of the concentration operation.

$$\mu_{\mathrm{POW}(A)}(x) = (\mu_A(x))^n \tag{11}$$

Given a fuzzy set of *tall persons*, we could use this operation with $n = 3$ to generate a set of *very very tall persons*.

FUZZY SET OPERATIONS

Intersection

In classical set theory, the intersection of two sets contains those elements that are common to both. However, in fuzzy sets, an element may be partially in both of the sets. Therefore, when considering the **intersection** of these two sets, we can't say that an element is more likely to be in the intersection than in one of the original sets.

To account for this, the fuzzy operation for creating the intersection of two fuzzy sets A and B defined on X is given as

$$\begin{aligned}\mu_{A \wedge B}(X) &= \min(\mu_A(x), \mu_B(x)) \qquad \text{for all } x \in X \\ &= \mu_A(x) \wedge \mu_B(x) \\ &= \mu_A(x) \cap \mu_B(x)\end{aligned} \tag{12}$$

The symbol $\wedge$ (called the logical "AND") is used in fuzzy logic to represent the "min" operation. It simply takes the minimum of the values under consideration. To illustrate this operation, consider the fuzzy sets of tall and short persons:

$$\text{TALL} = (0/5, 0.2/5.5, 0.5/6, 0.8/6.5, 1/7)$$

$$\text{SHORT} = (1/5, 0.8/5.5, 0.5/6, 0.2/6.5, 0/7)$$

According to equation 12, the intersection of these two sets is:

$$\mu_{\text{TALL} \wedge \text{SHORT}}(x) = (0/5, 0.2/5.5, 0.5/6, 0.2/6.5, 0/7)$$

When we consider the term "tall and short" it is reasonable to interpret the term as meaning "medium." With this interpretation, we would expect the highest degree of membership to be in the middle of the set, and the lowest at the set limits. This is the result we observe when we form the intersection of the "tall" and "short" fuzzy sets. This example illustrates how two fuzzy sets can be combined to form a new set. The linguistic term we might apply to this new set might be *medium height persons*.

Union

A second way of combining fuzzy sets is through their **union.** The union of two sets is comprised of those elements that belong to one or both sets. In this situation, the members of the union cannot have a membership value that is less than the membership value of either of the original sets. Fuzzy logic uses the following equation to form the union of two sets A and B:

$$\begin{aligned} \mu_{A \vee B}(x) &= \max(\mu_A(x), \mu_B(x)) \quad \text{for all } x \in X \\ &= \mu_A(x) \vee \mu_B(x) \qquad (13) \\ &= \mu_A(x) \cup \mu_B(x) \end{aligned}$$

The symbol $\vee$ (called the logical "OR") is used in fuzzy logic to represent the "max" operation. It takes the maximum of the values under consideration. To illustrate this operation, consider again the two fuzzy sets of tall and short persons.

$$\text{TALL} = (0/5, 0.2/5.5, 0.5/6, 0.8/6.5, 1/7)$$

$$\text{SHORT} = (1/5, 0.8/5.5, 0.5/6, 0.2/6.5, 0/7)$$

According to equation 13, the intersection of these two sets is

$$\mu_{\text{TALL} \vee \text{SHORT}}(x) = (1/5, 0.8/5.5, 0.5/6, 0.8/6.5, 1/7)$$

The results indicate the union membership attains its highest values at the limits and its lowest at the middle of the set. A linguistic interpretation of this new set might be *not medium*.

Complementation (Not)

Given the fuzzy set A, we can find its complement $\sim A$ by the following operation:

$$\mu_{\sim A}(x) = 1 - \mu_A(x) \qquad (14)$$

Given a fuzzy set of *tall persons*, this operation could be used to create the set of *not tall persons* or *medium or short persons*.

$$\mu_A(x) = \text{TALL} = (0/5, 0.2/5.5, 0.5/6, 0.8/6.5, 1/7)$$

$$\mu_{\sim A}(x) = \text{NOT TALL} = (1/5, 0.8/5.5, 0.5/6, 0.2/6.5, 0/7)$$

DERIVING ADDITIONAL FUZZY SETS

Using hedges and fuzzy set operations we can derive a variety of other fuzzy sets from existing ones. Assume for example we have a fuzzy set A of *tall persons*. We could derive a set B of *not very tall persons* from the following operation:

$$\mu_B(x) = 1 - (\mu_A(x))^2$$

To extend this idea, assume we have the fuzzy sets A of *tall persons* and B of *short persons*. We could derive a set C of *not very tall persons* and *not very short persons* from the following operation:

$$\mu_C(x) = [1 - (\mu_A(x))^2] \wedge [1 - (\mu_B(x))^2]$$

In general, we can use fuzzy operators and hedges to derive fuzzy sets that represent various linguistic descriptions and combinations of statements found in our natural language.

FUZZY INFERENCE

Fuzzy logic treats a fuzzy set as a *fuzzy proposition*. A fuzzy proposition is a statement that asserts a value for some given linguistic variable such as, "height is tall." In general, we can represent a fuzzy proposition as

Proposition: X is A

where A is a fuzzy set on the universe of discourse X. A fuzzy rule relates two fuzzy propositions in the form:

IF X is A THEN Y is B

This rule establishes a relationship or association between the two propositions.

Fuzzy expert systems store rules as fuzzy associations. That is, for the rule IF A THEN B, where A and B are fuzzy sets, a fuzzy expert system stores the association (A, B) in a matrix M (often labeled in the literature as R for *relationship*). The fuzzy associative matrix M maps fuzzy set A to fuzzy set B. Kosko (1992) calls this fuzzy association or fuzzy rule a **Fuzzy Associative Memory** (FAM). A FAM maps a fuzzy set to a fuzzy set—the fuzzy inference process.

Like other inexact reasoning techniques used in the design of an expert system, fuzzy inference attempts to establish a belief in a rule's conclusion given available evidence on the rule's premise. However, since the propositions contained in a fuzzy rule are fuzzy sets, fuzzy logic must map premise set information to conclusion set information. To accomplish this, fuzzy inference establishes an induced fuzzy set from information about a related fuzzy set.

Consider for example the fuzzy sets A—"Height is tall" and B—"Weight is heavy" related by the rule IF A THEN B.

IF Height is tall THEN Weight is heavy

We can represent both A and B as fit-vectors and capture their relationship in the fuzzy associative matrix M. You will see how to form this matrix in later sections.

In practice, we would could use this rule to form a degree of belief that some person of a given height is heavy. Fuzzy inference does this by taking the available height information encoded in A' (a subset of A) and inducing a fuzzy set B' on B that quantitatively captures this belief. To derive the induced fuzzy set, fuzzy inference relies on **fuzzy vector-matrix multiplication.**

The two most popular fuzzy inference techniques used in practice are **max-min inference** and **max-product inference.** Before looking at these two inference techniques, you must first understand fuzzy vector-matrix multiplication.

Fuzzy Vector-Matrix Multiplication

In classical vector-matrix multiplication, we can derive a vector y given a vector x and a matrix A by:

$$\underset{1\times n}{x} \cdot \underset{n\times p}{A} = \underset{1\times p}{y}$$

$$y_j = \sum_{i=1}^{n} x_i a_{ij}$$

Fuzzy vector-matrix multiplication uses a technique known as **max-min composition** (Klir and Foger 1988), defined by the composition operator "$\circ$". This operator performs a max-min operation on a given vector and matrix. The operation is similar to classical vector-matrix multiplication, however, we replace pairwise multiplications with pairwise minima and column (row) sums with column (row) maxima.

Consider this operation as applied to a fuzzy rule or FAM IF A THEN B, where A is a fuzzy set defined on X and B a fuzzy set defined on Y. For row fit vectors A and B represented as

$$A = (a_1, a_2, \ldots, a_n); \qquad a_i = \mu_A(x_i)$$

$$B = (b_1, b_2, \ldots, b_p); \qquad b_i = \mu_B(y_i)$$

we can define a fuzzy n by p matrix M such that

$$A \circ M = B \tag{15}$$

and compute component b_j by:

$$b_j = \max_{1\le i\le n}\{\min(a_i, m_{ij})\} \tag{16}$$

To illustrate, assume $A = (.2\ .4\ .6\ 1)$ and the fuzzy matrix M is

$$M = \begin{vmatrix} .1 & .6 & .8 \\ .6 & .8 & .6 \\ .8 & .6 & .5 \\ 0 & .5 & .5 \end{vmatrix}$$

From equation 16, we can compute B as

$$\begin{aligned} b_1 &= \max\{\min(.2, .1), \min(.4, .6), \min(.6, .8), \min(1, 0)\} \\ &= \max\{.1, .4, .6, 0\} \\ &= 0.6 \\ b_2 &= \max\{.2, .4, .6, .5\} \\ &= 0.6 \\ b_3 &= \max\{.2, .4, .5, .5\} \\ &= 0.5 \end{aligned}$$

Basic Ideas of Fuzzy Inference

To obtain a basic understanding of today's approaches to fuzzy inference, we need to go back and review some of the earlier ideas. Our best source for this information is the original work by Zadeh (1965).

Zadeh viewed a fuzzy set as a possibility distribution function. This function mapped elements of some universe of discourse into a number between 0 and 1 that reflected the degree of belief that some element belonged to the fuzzy set.

$$\begin{aligned} A &= \text{Possibility distribution} \\ &= \mu_A(x) \\ &= \Pi_A \end{aligned}$$

Zadeh also saw a need to be able to infer information on a fuzzy set B from information gained on another related one A. The approach taken to accomplish this was similar to classical conditional probability theory, where the compositional operator was used for the classical vector-matrix operation. Zadeh looked for a conditional possibility distribution matrix $\Pi_{B|A}$ such that if he composed it with the possibility distribution of A he would get back the possibility distribution of B

$$\Pi_A \circ \Pi_{B|A} = \Pi_B$$

where in general Π_A is a $1 \times n$ vector, $\Pi_{B|A}$ is a $n \times p$ matrix, and Π_B a $1 \times p$ vector. Using this approach, Zadeh could put in some information on A (labeled A'), and obtain information on B (labeled B'). He called this technique the **compositional rule of inference.**

The next question was how to form the $\Pi_{B|A}$ distribution matrix. Zadeh had an interesting approach to this question. He interpreted the components of the $\Pi_{B|A}$ matrix as pairwise implications between A and B. For example, given the fuzzy sets in fit-vector form, this matrix would appear as:

$$\Pi_{B|A} = \begin{vmatrix} a_1 \rightarrow b_1 & a_1 \rightarrow b_2 & \cdots \\ a_2 \rightarrow b_1 & & \\ \vdots & & \end{vmatrix}$$

This matrix is the same as our fuzzy associative matrix M described earlier. Over the years several implication operators have been proposed. Whalen and Schott (1983) provide an excellent review of the more popular ones. The next sections describe the two most common ones used today in the design of fuzzy expert systems.

Max–Min Inference

In max–min inference the implication operator used is *min*. That is:

$$m_{ij} = \text{truth}(a_i \rightarrow b_j) = \min(a_i, b_j) \tag{17}$$

Given two fuzzy sets A and B, we can use equation 17 to form the matrix M. We can then next use equation 16 to determine the induced vector B' from a subset of A designated as A'.

To illustrate, assume we have a universe of discourse defined on X that represents "temperature," and a fuzzy set A defined on X that represents "normal temperature." Also assume we have a universe of discourse defined on Y that represents "velocity," and a fuzzy set B defined on Y that represents "medium velocity." Finally assume we have the following fuzzy rule:

IF Temperature is normal THEN Velocity is medium

or

IF A THEN B

Further assume that the fuzzy sets are represented by the following vectors, where for clarity, the vector elements are shown with their corresponding domain

values:

$$\text{Normal temperature} = (0/100, .5/125, 1/150, .5/175, 0/200)$$

$$\text{Medium velocity} = (0/10, .6/20, 1/30, .6/40, 0/50)$$

We begin by forming the M matrix according to equation 17:

$$M = m_{ij} = \min(a_i, b_j)$$

$$= \begin{vmatrix} \min(0., 0.) & \min(0., .6) & \min(0., 1.) & \min(0., .6) & \min(0., 0.) \\ \min(.5, 0.) & \min(.5, .6) & \min(.5, 1.) & \min(.5, .6) & \min(.5, 0.) \\ \min(1., 0.) & \min(1., .6) & \min(1., 1.) & \min(1., .6) & \min(1., 0.) \\ \min(.5, 0.) & \min(.5, .6) & \min(.5, 1.) & \min(.5, .6) & \min(.5, 0.) \\ \min(0., 0.) & \min(0., .6) & \min(0., 1.) & \min(0., .6) & \min(0., 0.) \end{vmatrix}$$

$$= \begin{vmatrix} 0. & 0. & 0. & 0. & 0. \\ 0. & 0.5 & 0.5 & 0.5 & 0. \\ 0. & 0.6 & 1. & 0.6 & 0. \\ 0. & 0.5 & 0.5 & 0.5 & 0. \\ 0. & 0. & 0. & 0. & 0. \end{vmatrix}$$

Next, assume that the subset A' is given as

$$A' = (0/100, .5/125, 0/150, 0/175, 0/200)$$

This subset represents a crisp reading in temperature of 125 degrees. This measurement maps to a membership value of 0.5 for the fuzzy set "normal temperature." This induces a fuzzy set B' (i.e., a belief in B) that we next want to determine.

With $A' = (0/100, .5/125, 0/150, 0/175, 0/200)$, then through max–min composition (equation 16) we have

$$b_j = \max_{1 \le i \le n}\{\min(a_i', m_{ij})\}$$

$$b_1 = \max[\min(0., 0.), \min(.5, 0.), \min(0., 0.), \min(0., 0.), \min(0., 0.)]$$

$$b_2 = \max[\min(0., 0.), \min(.5, .5), \min(0., .6), \min(0., .5), \min(0., 0.)]$$

$$b_3 = \max[\min(0., 0.), \min(.5, .5), \min(0., 1.), \min(0., .5), \min(0., 0.)]$$

$$b_4 = \max[\min(0., 0.), \min(.5, .5), \min(0., .6), \min(0., .5), \min(0., 0.)]$$

$$b_5 = \max[\min(0., 0.), \min(.5, 0.), \min(0., 0.), \min(0., 0.), \min(0., 0.)]$$

$$B' = (0/10, .5/20, .5/30, .5/40, 0/50)$$

In effect, this induced fuzzy set is a clipped version of B, whose height is set by A'. This is the general effect of max–min inference as illustrated in Figure 13.4 for triangular shaped fuzzy sets. What we typically do with this induced set is discussed later under the subject of **defuzzification.**

A key point to notice from this example is the result we obtained by limiting A' to a single value. That is, we stated that our temperature reading was 125

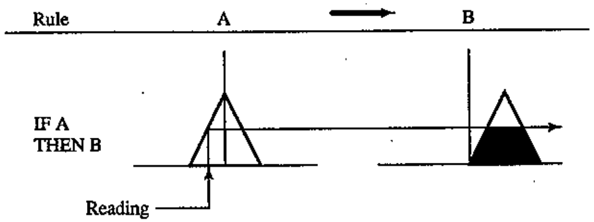


FIGURE 13.4 Max-min inference.

degrees which gave us a A' fit vector of (0 .5 0 0 0) that resulted in a B' of (0 .5 .5 .5 0).

In most real-world applications of fuzzy logic systems we have a crisp value on some measurement (e.g., $x_k = 125$ degrees). With a single measurement value x_k we can use $\mu_A(x_k)$ directly with the fuzzy set representation of B, namely $\mu_B(y)$, to obtain the induced fuzzy set on B':

$$B' = \mu_A(x_k) \wedge \mu_B(y) \tag{18}$$

For instance, in our example where we assumed the temperature was 125 degrees, it follows that $\mu_A = 0.5$ and

$$B' = [\min(.5, 0), \min(.5, .6), \min(.5, 1), \min(.5, .6)] \min(.5, 0)]$$
$$= (0, .5, .5, .5, 0)$$

This is the same result obtained earlier working with the fuzzy associative matrix. Therefore, when input information is in crisp form, we do not have to calculate and maintain fuzzy matrices, but can simply work with the less demanding (from a computer overhead viewpoint) fuzzy set information.

In the event the input to a rule represents a fuzzy reading, we can still take a simple approach. Consider the rule IF A THEN B, and a fuzzy reading of A designated as A'. We can simply take the intersection of the two as our input, $\min(a'_i, a_i)$, to induce the fuzzy set B'. This approach is illustrated in Figure 13.5.

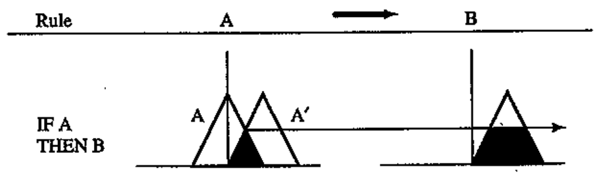


FIGURE 13.5 Max-min inference for fuzzy input.

Max–Product Inference

Max–product inference uses the standard product as the implication operator when forming the components of M:

$$m_{ij} = a_i b_j \tag{19}$$

Following the calculation of this matrix, max–min composition is used to determine the induced matrix B' from some subset vector A'. To illustrate this inference technique we consider the same fit vectors used in the previous section:

$$A = (0, .5, 1, .5, 0) \qquad B = (0, .6, 1, .6, 0)$$

$$M = \begin{vmatrix} (0.\cdot 0.) & (0.\cdot .6) & (0.\cdot 1.) & (0.\cdot .6) & (0.\cdot 0.) \\ (.5\cdot 0.) & (.5\cdot .6) & (.5\cdot 1.) & (.5\cdot .6) & (.5\cdot 0.) \\ (1.\cdot 0.) & (1.\cdot .6) & (1.\cdot 1.) & (1.\cdot .6) & (1.\cdot 0.) \\ (.5\cdot 0.) & (.5\cdot .6) & (.5\cdot 1.) & (.5\cdot .6) & (.5\cdot 0.) \\ (0.\cdot 0.) & (0.\cdot .6) & (0.\cdot 1.) & (0.\cdot .6) & (0.\cdot 0.) \end{vmatrix}$$

$$= \begin{vmatrix} 0. & 0. & 0. & 0. & 0. \\ 0. & 0.3 & 0.5 & 0.3 & 0. \\ 0. & 0.6 & 1. & 0.6 & 0. \\ 0. & 0.3 & 0.5 & 0.3 & 0. \\ 0. & 0. & 0. & 0. & 0. \end{vmatrix}$$

Assuming again $A' = (0, .5, 0, 0, 0)$, then through max–min composition we have

$$b_j = \max_{1\le i\le n}\{\min(a'_i, m_{ij})\}$$

$$b_1 = \max[\min(0., 0.), \min(.5, 0.), \min(0., 0.), \min(0., 0.), \min(0., 0.)]$$

$$b_2 = \max[\min(0., 0.), \min(.5, .3), \min(0., .6), \min(0., .5), \min(0., 0.)]$$

$$b_3 = \max[\min(0., 0.), \min(.5, .6), \min(0., 1.), \min(0., .5), \min(0., 0.)]$$

$$b_4 = \max[\min(0., 0.), \min(.5, .3), \min(0., .6), \min(0., .5), \min(0., 0.)]$$

$$b_5 = \max[\min(0., 0.), \min(.5, 0.), \min(0., 0.), \min(0., 0.), \min(0., 0.)]$$

$$B' = (0, .3, .5, .3, 0)$$

The max–product inference technique produces a scaled version of B. Figure 13.6 illustrates the general result of using this technique for triangular fuzzy sets.

Like the example used in the prior section, the A' fit-vector contained a single crisp value. Because of this limiting but typical situation found in practice, we can again make an observation that eases the computation of B'. Given that the measurement of A is x_k, we can obtain B' from

$$B' = \mu_A(x_k) \cdot \mu_B(y) \tag{20}$$

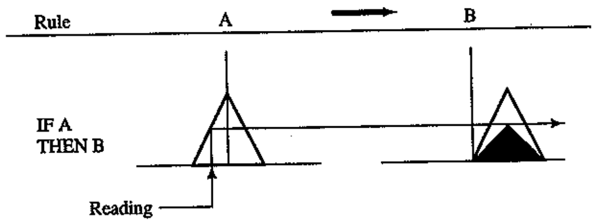


FIGURE 13.6 Max-product inference.

For our example

$$B' = 0.5 \cdot (0, .6, 1, .6, 0)$$
$$= (0, .3, .5, .3, 0)$$

As previously shown, max–min inference produces a clipped version of B. As just illustrated, the max–product inference technique produces a scaled version of B. In this sense, max–product inference preserves more information than max-min inference. When we combine induced fuzzy sets from multiple rules, as a precursor to defuzzification, this point becomes important.

MULTIPLE-PREMISE RULES

In the prior sections we have been looking at fuzzy rules that contain a single premise of the form IF A THEN B. However, in practice, we will often need to work with multiple premise rules (e.g., IF A AND B THEN C). The question we now face is how do we form the fuzzy associative matrix M for this rule?

Kosko (1992) suggests a simple answer to this question—one that is typically used in practice. Assume that fuzzy set A is defined on X, set B on Y, and C on Z. The approach relies on first defining for each premise a separate M matrix, that relates the premise to the conclusion (e.g., M_{AC} and M_{BC}). Then given some input information on the premises, A' and B', the induced fuzzy sets on C can be computed independently through composition

$$A' \circ M_{AC} = C_{A'} \tag{21}$$

$$B' \circ M_{BC} = C_{B'} \tag{22}$$

The next step is to recompose the fuzzy sets $C_{A'}$ and $C_{B'}$. The approach taken depends on whether the premises are joined conjunctively "AND" or disjunctively "OR". For premises joined in an AND fashion, the fuzzy logic intersection operator is used to join the induced fuzzy sets

$$C' = [A' \circ M_{AC}] \wedge [B' \circ M_{BC}]$$
$$= C_{A'} \wedge C_{B'} \tag{23}$$

TABLE 13.2 Computing Induced Fuzzy Sets for Multiple Premise Rules with Crisp Input Values

C'	*Premise Joining*	*Inference*
$\min(a_i, b_j) \wedge \mu_C(z)$	AND	Max–Min
$\max(a_i, b_j) \wedge \mu_C(z)$	OR	Max–Min
$\min(a_i, b_j) \cdot \mu_C(z)$	AND	Max–Product
$\max(a_i, b_j) \cdot \mu_C(z)$	OR	Max–Product

while the union operator is used for OR joined premises

$$C' = [A' \circ M_{AC}] \vee [B' \circ M_{BC}]$$
$$= C_{A'} \vee C_{B'} \tag{24}$$

This process can be simplified further when the inputs are crisp values. Consider the case of a single value of A defined at x_k and a single value of B defined at y_j. Given the fuzzy set membership values of $a_i = \mu_A(x_k)$ and $b_j = \mu_B(y_j)$, Table 13.2 shows simple approaches for computing C' for multiple premise rules.

Defuzzification

You have now seen two techniques for determining the induced fuzzy set B' from an estimate of A as defined by A'. In most applications, we need to take B' and obtain a crisp value. For example, earlier we worked with the fuzzy rule

IF Temperature is normal THEN Velocity is medium

and used a temperature measurement to induce a fuzzy set on "normal velocity." In a control application we would want to know what specific velocity value we should use. This requires us to take the induced fuzzy set and obtain a crisp value. This is the subject of **defuzzification.**

The most popular defuzzification technique used is the **fuzzy centroid** method. This technique provides a single value y_i from B' by way of the following:

$$y_i = \frac{\sum_{j=1}^{p} y_j m_{B'}(y_j)}{\sum_{j=1}^{p} m_{B'}(y_j)} \tag{25}$$

Using this technique on our example provides us with a single velocity value. However, using it with a single rule is not very interesting because the value will always be the same—the centroid of B. In general, defuzzification is important when multiple rules conclude the same event.

Multiple Fuzzy Rules

We next consider the situation where we have n fuzzy rules or associations $(A_1, B_1), \ldots, (A_n, B_n)$. This situation leads to n matrices $M_1, \ldots, M_n$ to encode the associations or relationships between A_i and B_i.

Our interest in this set of fuzzy rules is the resultant overall belief in B given a single measurement A'. We proceed by applying A' in parallel to the bank of rules, producing an induced fuzzy set B'_i for each rule. We then sum all of the B'_i sets to form the resultant composite induced fuzzy set B', using the following standard set union operation where B is defined on domain X:

$$B' = B'_1 \cup B'_2 \cup \ldots B'_{n-1} \cup B'_n$$

$$= \max(B'_1(x), B'_2(x), \ldots B'_{n-1}(x), B'_n(x)) \qquad \text{for all } x \in X \qquad (26)$$

Following this union operation, we can next defuzzify the resultant B' using the centroid method. This process produces a crisp output value y_i. This entire operation is illustrated in Figure 13.7. In effect, this operation takes an input measurement of A and produces an estimate of B.

To expand on this idea, assume we want to control the velocity of some vehicle. Further assume that our decision is based on the vehicle's temperature and pressure as captured in the following rules:

IF Temperature is normal
OR Pressure is low
THEN Velocity is medium

IF Temperature is normal
AND Pressure is normal
THEN Velocity is low

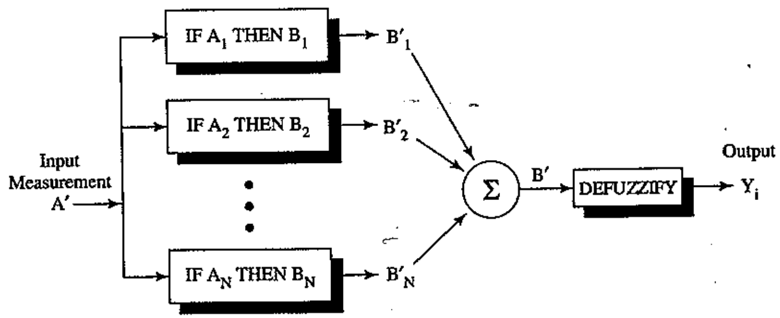


FIGURE 13.7 Fuzzy rule system architecture.

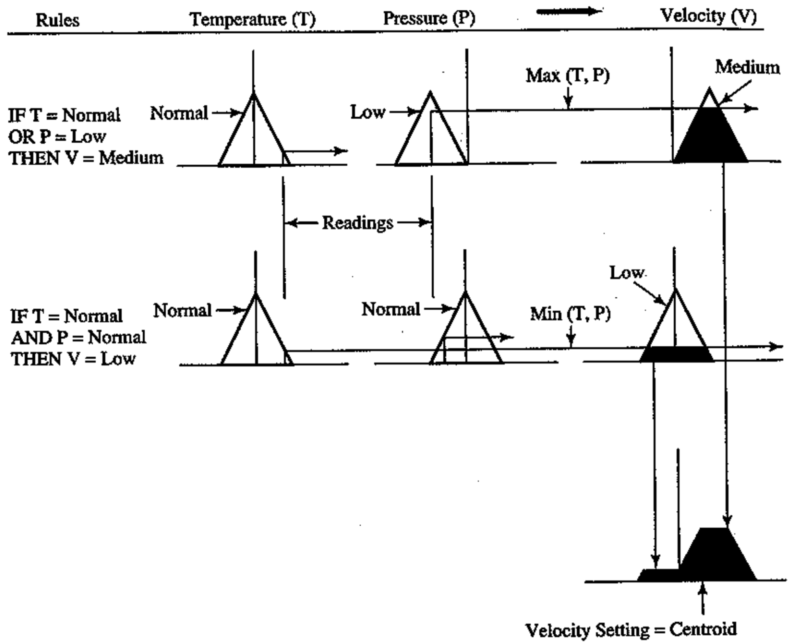


FIGURE 13.8 Max-min inference for multiple rules.

Figure 13.8 illustrates the max–min inference process for these two rules, while Figure 13.9 shows the max-product inference process. Readings of temperature and pressure are used to determine their corresponding membership values in the fuzzy sets shown. The largest value is then chosen for the disjunctive rule and the smallest for the conjunctive rule. These values are then used to either clip the inferred velocity fuzzy set (max–min inference) or scale it (max–product inference). The induced fuzzy sets on velocity are then summed (fuzzy union) and the centroid of the result found. The resultant centroid provides the desired velocity value given the input measurements.

In general, we can apply the process shown in Figure 13.7 to any number of fuzzy rules with any number of antecedent fuzzy-variable conditions.

BUILDING A FUZZY LOGIC EXPERT SYSTEM

To illustrate the design of a fuzzy logic expert system we consider a problem of navigating a golf cart. We want to design a fuzzy logic system that will automatically take Kathy—an avid weekend golfer—around the golf course.

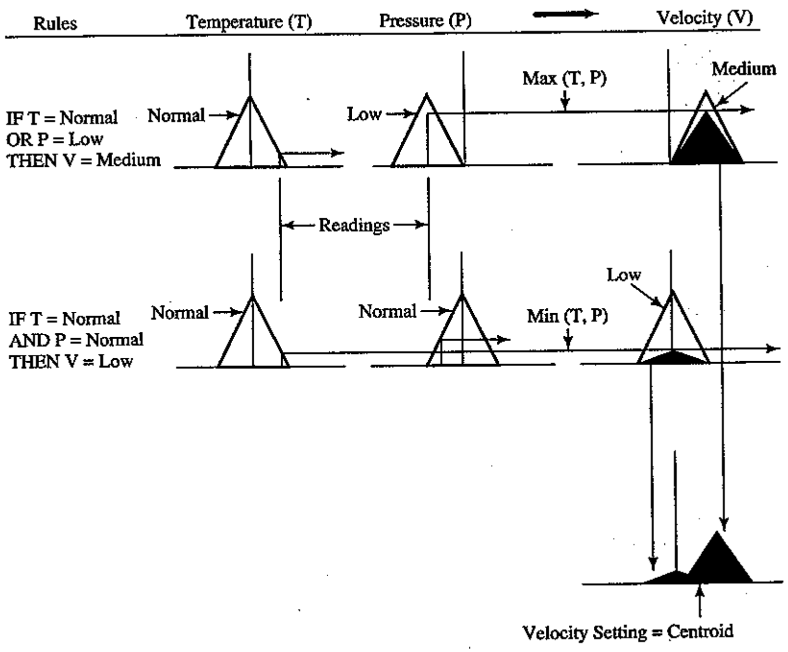


FIGURE 13.9 Max-product inference for multiple rules.

Kathy has had trouble controlling her cart which has led to an increase in both her anxiety and golf score. With an automatic golf cart she can relax, and hopefully better enjoy and improve her game.

There are seven major tasks you will typically perform when developing a fuzzy logic expert system:

Task 1: Define the problem
Task 2: Define the linguistic variables
Task 3: Define the fuzzy sets
Task 4: Define the fuzzy rules
Task 5: Build the system
Task 6: Test the system
Task 7: Tune the system

Task 1: Define the Problem

Like all expert system projects, we need to first obtain a source of knowledge. Usually this source is an expert on the problem. For our problem, we contact

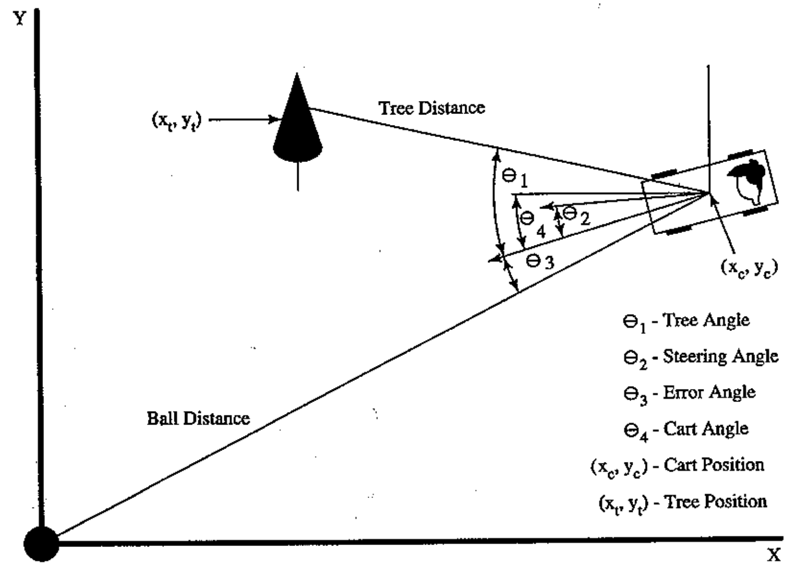


FIGURE 13.10 Cart navigation geometry.

Bob, a golf pro at a local course, who has years of experience in driving a golf cart.

Bob tells us that the basic problem is to navigate the cart in an efficient and safe fashion from some initial stationary position to the location of the golf ball. We obtain efficiency by minimizing both the distance traveled and travel time. We obtain safety by avoiding any trees along the way. To accomplish these tasks we will need to provide the fuzzy system with control over both the direction and speed of the cart. Figure 13.10 illustrates our navigation problem.

This is basically an error-nulling problem. The cart must initially steer toward the ball by nullifying the error between the angular direction of the cart and the direction toward the ball. The cart should also accelerate to some maximum allowable speed, then slow down and eventually stop when it is close to the ball; we will pick a stopping distance around 3 yards. This minimizes the error between the location of the cart and the ball's location in a quick fashion. In addition, when a tree is in the path of the cart, the cart must steer around it cautiously (i.e., slow down), then pick up speed and again steer toward the ball.

Task 2: Define Linguistic Variables

We next need to define the linguistic variables for our problem. We accomplish this task by listening to how Bob solves the problem. We want to uncover the variables that will represent our universes of discourse and the fuzzy sets that will be defined on each.

From task 1, we know that our fuzzy logic system must contend with three basic problems:

1. Control steering of cart to direct it toward the ball.
2. Control the cart's speed.
3. Control steering of cart to avoid any trees.

We next ask Bob to discuss in general how each of these problems are solved. Consider problem 1 and assume Bob provides us with the following common sense strategy for steering the cart toward the ball:

> "When the direction of the cart is away from the ball, make the cart's direction toward the ball."

In a similar fashion we obtain the expert's strategies for controlling the cart's speed:

> "When the cart is far from the ball, make the cart's speed fast."

> "When the cart is close to the ball, make the cart's speed slow."

and the cart's direction and speed:

> "When the cart is close to the tree and heading toward it, then slow the cart down and make the cart's direction away from the tree."

From this discussion we can now define our linguistic variables—the universe of discourses, and also ask our expert to define their ranges:

LINGUISTIC VARIABLE		RANGE	
Error angle	−180	to	180 degrees
Tree angle	−180	to	180 degrees
Steering angle	−45	to	45 degrees
Speed	0	to	5 yd/s
Acceleration	−2	to	1 yd/s/s
Ball distance	0	to	600 yards
Tree distance	0	to	1000 yards

Task 3: Define Fuzzy Sets

The next task involves defining the fuzzy sets on each universe. To accomplish this we again consult Bob and asked him for a list of typical adjectives used with each linguistic variable. Assume this effort resulted in the lists shown in Figure 13.11. This figure represents a vocabulary "dictionary" for the problem.

DESIGN SUGGESTION: Maintain a dictionary of terms used in the system that includes all linguistic variables and their associated adjectives.

Error Angle	Tree Angle	Steering Angle	Speed	Acceleration	Ball Distance	Tree Distance
Large Negative	Large Negative	Hard Right	Zero	Brake Hard	Zero	Close
Small Negative	Small Negative	Slight Right	Real Slow	Brake Light	Real Close	
Zero	Zero	Zero	Slow	Coast	Close	
Small Positive	Small Positive	Slight Left	Medium	Zero	Medium	
Large Positive	Large Positive	Hard Left	Fast	Slight Acceleration	Far	
				Floor It		

FIGURE 13.11 Fuzzy variables with adjectives.

We next ask Bob for information that will allow us to define the fuzzy sets for each adjective given in Figure 13.11. For example, we might first ask, "What speed do you consider slow?" Assume Bob provides a vague answer to this question—"around 1 to 3 yards per second." At this point, we can ask him to what degree he believes various speed values are "slow"—"To what degree do you believe that 1 yd/s is slow?" We can use similar questions for other speed values, and follow this effort with the selection of a function that reasonably maps the speed values into their corresponding belief values—we now have our first fuzzy set.

Fuzzy mapping or membership functions can have a variety of shapes depending on how the expert relates different domain values to belief values. In practice, a piecewise linear function, such as a triangular or trapezoidal shape, provides an adequate capture of the expert's belief and simplifies the computation.

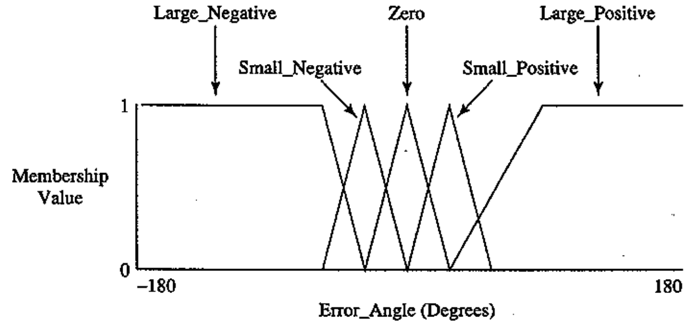
We could continue this process in order to define the other fuzzy sets for the "speed" linguistic variable. In practice, however, once the expert has seen how the first fuzzy set is formed, he is usually able to define the other sets simply by shifting the selected mapping function over the variable's universe. However, this approach may require a small amount of narrowing or widening of the shifted function.

DESIGN SUGGESTION: Allow the expert to shift an existing fuzzy set function along the universe of discourse to account for other adjectives.

Figures 13.12 through 13.18 show the fuzzy sets for all the terms shown in Figure 13.11. Each figure also includes fit vectors for its various fuzzy sets.

One key point when defining the fuzzy sets is to make sure you have sufficient overlap in the sets to assure that every possible value establishes some fuzzy set membership value. If this is not done, the possibility will always exist that your system will be unresponsive to some value.

DESIGN SUGGESTION: Maintain sufficient overlap in adjacent fuzzy sets.



Large_Positive

X	Y
30	0
90	1
180	1

Small_Positive

X	Y
0	0
30	1
60	0

Zero

X	Y
−30	0
0	1
30	0

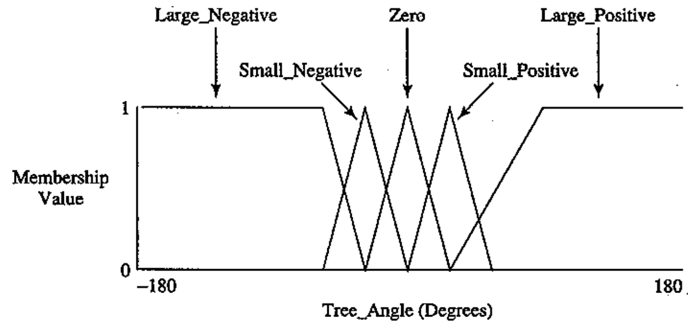
Small_Negative

X	Y
−60	0
−30	1
0	0

Large_Positive

X	Y
−180	1
−60	1
−30	0

FIGURE 13.12 Fuzzy sets on error angle.



Large_Positive

X	Y
30	0
90	1
180	1

Small_Positive

X	Y
0	0
30	1
60	0

Zero

X	Y
−30	0
0	1
30	0

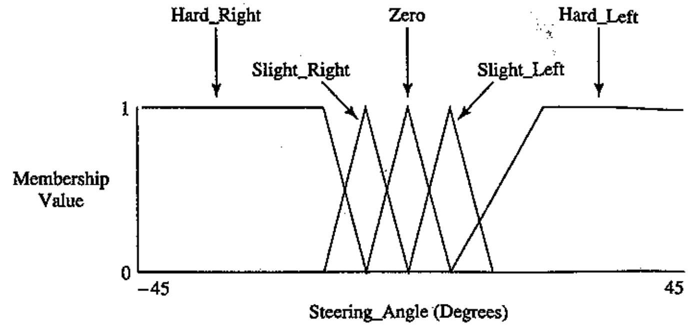
Small_Negative

X	Y
−60	0
−30	1
0	0

Large_Negative

X	Y
−180	1
−60	1
−30	0

FIGURE 13.13 Fuzzy sets on tree angle.



Hard_Left

X	Y
7.5	0
22.5	1
45	1

Slight_Left

X	Y
0	0
7.5	1
15	0

Zero

X	Y
−7.5	0
0	1
7.5	0

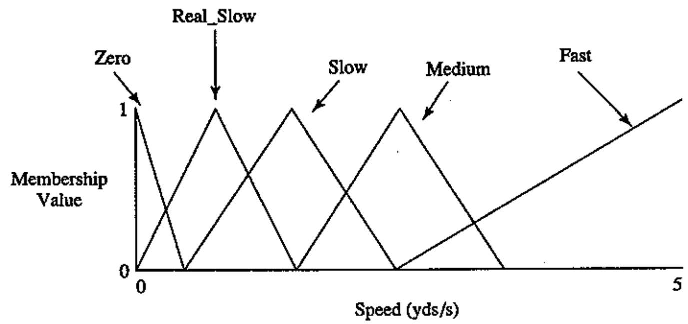
Slight_Right

X	Y
−15	0
−7.5	1
0	0

Hard_Right

X	Y
−45	1
−15	1
−7.5	0

FIGURE 13.14 Fuzzy sets on steering angle.



Zero

X	Y
0	1
0.5	0

Real_Slow

X	Y
0	0
0.7	1
1.5	0

Slow

X	Y
0.5	0
1.5	1
2.5	0

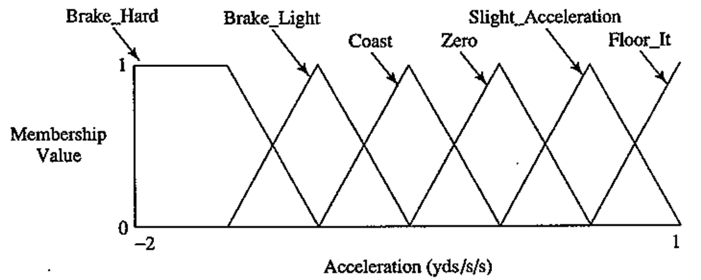
Medium

X	Y
1.5	0
2.5	1
3.5	0

Fast

X	Y
0	0
2.5	0
5.0	1

FIGURE 13.15 Fuzzy sets on speed.



Brake_Hard

X	Y
−2	1
−1.5	1
−1	0

Brake_Light

X	Y
−1.5	0
−1	1
−0.5	0

Coast

X	Y
−1	0
−0.5	1
0	0

Zero

X	Y
−0.5	0
0	1
0.5	0

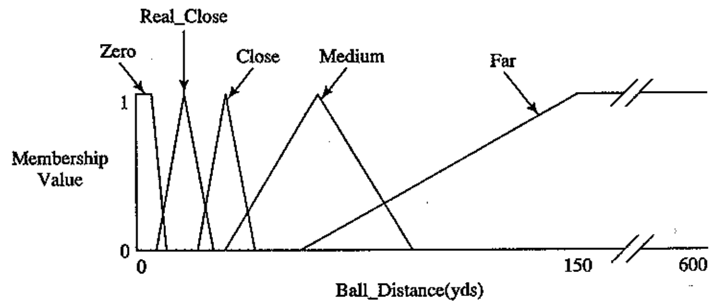
Slight_Acceleration

X	Y
0	0
0.5	1
1	0

Floor_It

X	Y
0.5	0
1	1

FIGURE 13.16 Fuzzy sets on acceleration.



Zero

X	Y
0	1
3	1
9	0

Real_Close

X	Y
3	0
12	1
18	0

Close

X	Y
12	0
24	1
36	0

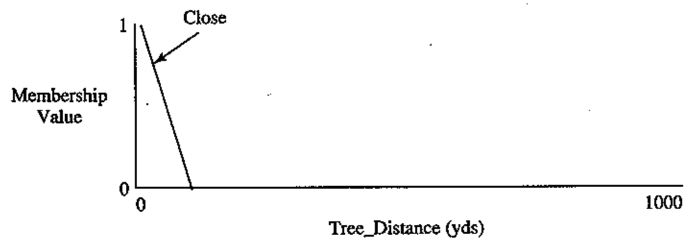
Medium

X	Y
24	0
60	1
96	0

Far

X	Y
60	0
150	1
600	1

FIGURE 13.17 Fuzzy sets on ball distance.



Close

X	Y
0	1
50	0
1000	0

FIGURE 13.18 Fuzzy sets on tree distance.

Task 4: Define Fuzzy Rules

Next we need to define the fuzzy rules. To accomplish this, we ask Bob to discuss how he addresses the three primary problems: steering the cart to the ball, controlling the cart's speed, and steering the cart to avoid a tree. We also ask him to consider the use of the fuzzy adjectives previously defined. However, if other adjectives surface during this discussion, we can simply add them to our prior list and define a fuzzy set for each. The results of this effort are shown in Figure 13.19, where several fuzzy hedges and the fuzzy operator NOT are used.

It should be noted that the use of hedges in fuzzy rules should only be considered later when the system is being tuned. The initial set of rules should

```
==========================================================
Rules for steering
==========================================================

RULE 1S - Maintain steering direction
IF   error_angle is zero
AND  tree_distance is NOT SOMEWHAT close
AND  tree_angle is NOT SOMEWHAT zero
THEN make steering_angle zero

RULE 2S - Change steering direction slightly right
IF   error_angle is small_positive
AND  tree_distance is NOT SOMEWHAT close
AND  tree_angle is NOT SOMEWHAT zero
THEN make steering_angle slight_right
```

FIGURE 13.19 Fuzzy rules for cart navigation system.

```
RULE 3S - Change steering direction slightly left
IF   error_angle is small_negative
AND  tree_distance is NOT SOMEWHAT close
AND  tree_angle is NOT SOMEWHAT zero
THEN make steering_angle slight_left

RULE 4S - Change steering direction slightly right
IF   error_angle is large_positive
AND  speed is fast
THEN make steering_angle slight_right

RULE 5S - Change steering direction hard right
IF   error_angle is large_positive
AND  speed is NOT fast
THEN make steering_angle hard_right

RULE 6S - Change steering direction slightly left
IF   error_angle is large-negative
AND  speed is fast
THEN make steering_angle slight_left

RULE 7S - Change steering direction hard left
IF   error_angle is large_negative
AND  speed is NOT fast
THEN make steering_angle hard_left

==========================================================
Rules for acceleration
==========================================================

RULE 1A - Brake lightly
IF   error_angle is large_positive
AND  speed is fast
THEN make acceleration brake_light

RULE 2A - Brake lightly
IF   error_angle is large_negative
AND  speed is fast
THEN make acceleration brake_light

RULE 3A - Floor it
IF   ball_distance is far
AND  speed is NOT VERY fast
THEN make acceleration floor_it

RULE 4A - Set acceleration to zero
IF   ball_distance is far
AND  speed is VERY fast
THEN make acceleration zero

RULE 5A - Slight acceleration
IF   ball_distance is medium
AND  speed is NOT fast
THEN make acceleration slight_acceleration
```

FIGURE 13.19 Continued


```
RULE 6A - Set acceleration to zero
IF   ball_distance is medium
AND  speed is fast
THEN make acceleration zero

RULE 7A - Brake lightly
IF   ball_distance is close
AND  speed is fast
THEN make acceleration brake_light

RULE 8A - Slight acceleration
IF   ball_distance is close
AND  Speed is zero
THEN make acceleration slight_acceleration

RULE 9A - Brake hard
IF   ball_distance is real_close
AND  speed is fast
THEN make acceleration brake_hard

RULE 10A - Brake lightly
IF   ball_distance is real_close
AND  speed is medium
THEN make acceleration brake_light

RULE 11A - Coast
IF   ball_distance is real_close
AND  speed is slow
THEN make acceleration coast

RULE 12A - Set acceleration to zero
IF   ball_distance is real_close
AND  speed is real_slow
THEN make acceleration zero

RULE 13A - Slight acceleration
IF   ball_distance is real_close
AND  speed is zero
THEN make acceleration slight_acceleration

RULE 14A - Brake hard
IF   ball_distance is zero
AND  speed is NOT zero
THEN make acceleration brake_hard

RULE 15A - Coast
IF   ball_distance is close
AND  speed is medium
THEN make acceleration coast

RULE 16A - Set acceleration to zero
IF   ball_distance is close
AND  speed is slow
THEN make acceleration zero
```

```
============================================================
Rules to avoid the tree
============================================================

RULE 1T - Turn slightly left to avoid tree
IF   tree_distance is SOMEWHAT close
AND  tree_angle is SOMEWHAT zero
AND  tree_angle is SOMEWHAT small_positive
THEN make steering_angle slight_left

RULE 2T - Turn slightly right to avoid tree
IF   tree_distance is SOMEWHAT close
AND  tree_angle is SOMEWHAT zero
AND  tree_angle is SOMEWHAT small_negative
THEN make steering_angle slight_right

RULE 3T - Brake hard to avoid tree
IF   tree_distance is VERY close
AND  tree_angle is zero
THEN make acceleration brake_hard
```

FIGURE 13.19 Continued

be small and only includes fuzzy sets for the adjectives in the dictionary without any added adverbs. As you test the system, you can add adverbs to see if the system's performance improves.

DESIGN SUGGESTION: Use adverbs in the rules to tune the system's performance.

Task 5: Build System

Now that we have the fuzzy sets and rules, our next task is to build the system. This task involves the coding of the fuzzy sets, and rules and procedures for performing fuzzy logic functions such as fuzzy inference. To accomplish this task you can go in one of two ways: build the system from scratch using a basic programming language, or rely on a fuzzy logic development shell.

The C programming language is the language of choice by most developers of fuzzy logic expert systems. It offers data structures that are conducive for implementing fuzzy logic procedures. However, building a fuzzy logic system using a basic programming language requires the developer not only to code the problem's knowledge (e.g., fuzzy sets and fuzzy rules); he is also responsible for coding the fuzzy logic procedures.

A fuzzy logic shell (like all shells) provides a complete environment for building a fuzzy logic expert system. The designer is only responsible for coding the problem's knowledge—a task often accomplished by using a natural language syntax for the rules and a graphical method for defining the fuzzy sets. The most popular fuzzy logic shells available today are listed in Appendix B.

We will assume that our example system was developed using the shell CubiCalc. CubiCalc is a windows-based development tool that permits rapid prototyping of a fuzzy logic expert system. It also offers a simulation utility that allows us to easily test our system.

Task 6: Test System

After you have built the system, you will want to test it to see if it meets the specifications defined during task 1. For our system, test cases were simulated using CubiCalc's simulation utility. All test runs used the max–product inference technique.

A number of test situations exist that depend on the location of the ball, tree, and cart, and on the orientation of the cart. We will only look at two situations: no nearby tree and one with a tree in the cart's path. For both test cases we assume the ball is at coordinates (0,0).

Test 1: No Tree in Path

For the first test case we consider a situation where the tree is not in the direct path of the cart. Also, the cart is directed away from the ball at a cart angle of −45 degrees. Simulated test results are shown in Figures 13.20 and 13.21.

Figure 13.20 shows the cart's path during the test and illustrates that it had little difficulty in reaching the ball. After an initial maneuver to obtain the target,

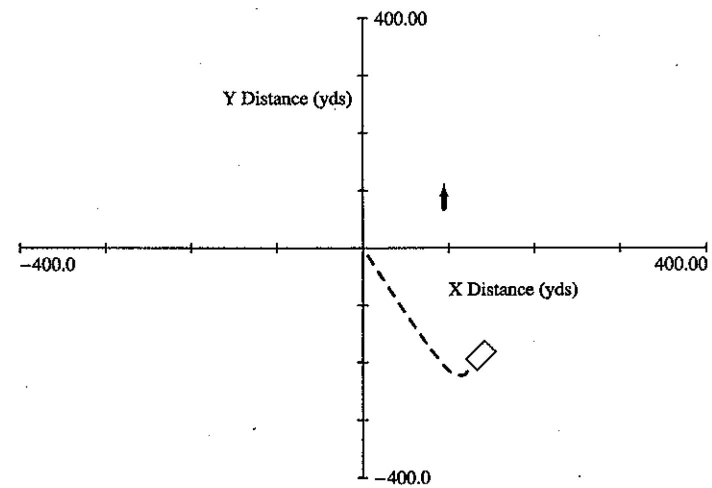


FIGURE 13.20 Cart navigation test case1.

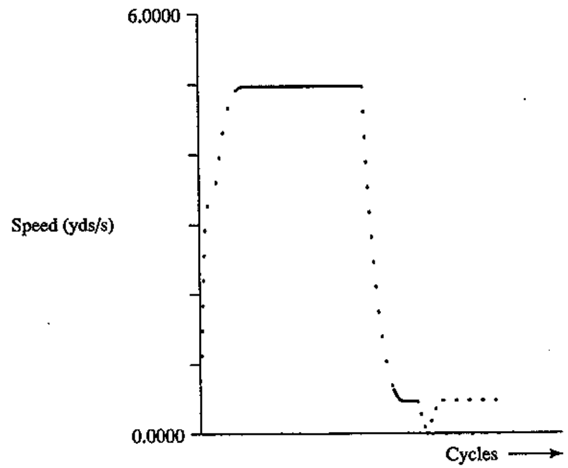


FIGURE 13.21 Cart speed test case1.

the cart followed a straight line toward the ball. No deviations from this straight line are noted due to the remoteness of the tree. The cart stops approximately 3 yards from the ball.

Figure 13.21 shows the cart's speed during the test. From a stationary position the cart quickly picks up speed to an allowable maximum value of 5 yards/second. As the cart approaches the ball it quickly decelerates. At this point the speed begins to oscillate—an undesirable situation that is discussed later.

To provide insight into the system's inference process, the simulation was paused to inspect the firing of the fuzzy rules. At the point where the simulation was paused, the following conditions existed:

—cart was 28.5 yards from ball
—cart was moving directly toward ball
—cart's speed was 4.5 yards/second
—cart was decelerating at a rate of 0.5 yards/s/s
—error angle was negligible
—tree was far away

Under these conditions only three rules apply: RULES 5A, 6A and 7A. To evaluate RULE 5A, the membership functions ''medium'' of ''ball_distance'' and ''fast'' of ''speed'' must be evaluated. A ''ball_distance'' of 28.5 yards belongs to the classification of ''medium'' to a degree of 0.125 (see Figure 13.17). A ''speed'' of 4.5 yards/sec is considered ''fast'' to a degree of 0.8 (see Figure 13.15); therefore it is ''NOT fast'' to a degree of 0.2. When ANDing these two values, the minimum operator produces 0.125 as the truth of the antecedents. Using the max–product inference technique, this value is used to

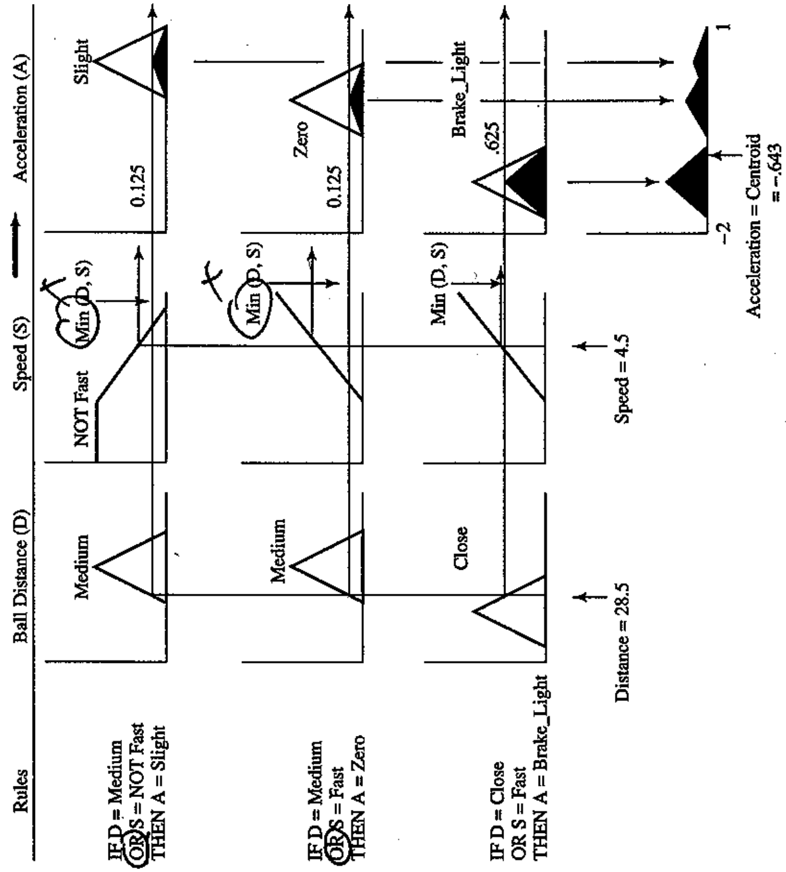


FIGURE 13.22 Fuzzy inference sample for test case1.

scale the membership function of the conclusion of RULE 5A "slight_acceleration."

When RULE 6A is evaluated, using the truth values for "medium" and "fast" found above, the antecedents again have a combined truth value of 0.125. The membership value of "zero" acceleration is scaled by this value.

When evaluating RULE 7A, the distance of 28.5 yards is considered "close" to a degree of 0.625 and the "speed" is "fast" to a degree of 0.8. Therefore, "brake_light" becomes scaled to a value of 0.625.

The parallel firing of these three rules is shown in Figure 13.22. The three induced fuzzy sets on "acceleration" are also shown combined using the union operator. A centroid of this result is then computed providing an update in the cart's acceleration. Since the final value is negative, the simulator decreases the cart's speed and the process continues.

Test 2: Tree in Path

In the next test case we consider a situation where the tree is in the direct path of the cart. Also, the cart is directed away from the ball at an angle of 60 degrees. Simulated test results are shown in Figures 13.23 and 13.24.

Figure 13.23 illustrates that the cart must first maneuver to obtain the target. The cart then follows a straight line towards the ball until it comes near the tree. At this point, it veers around the tree and then continues a straightline approach toward the ball. As in test 1, it stops approximately 3 yards from the ball.

Figure 13.24 shows that the cart once again quickly picks up speed and proceeds at the maximum allowable speed until it nears the tree. At this point, it slows down to cautiously move around the tree. After maneuvering around the tree it again picks up speed until it approaches the ball. Here it quickly reduces speed to prepare for stopping. Unfortunately, as was the case in test 1, the cart's speed oscillates as it nears the ball.

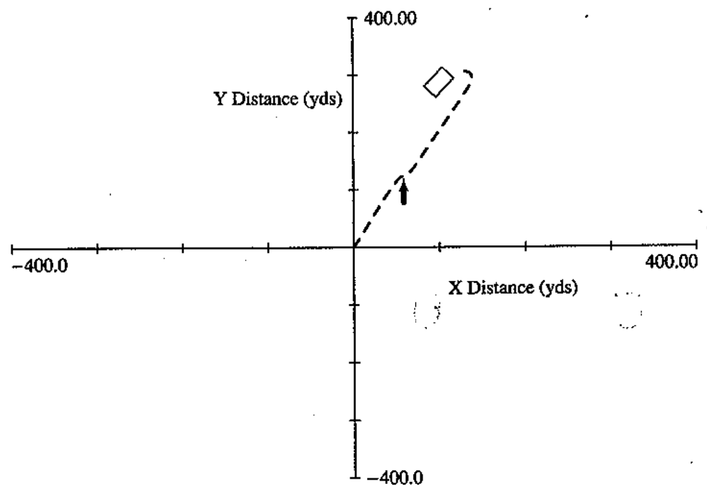


FIGURE 13.23 Cart navigation test case2.

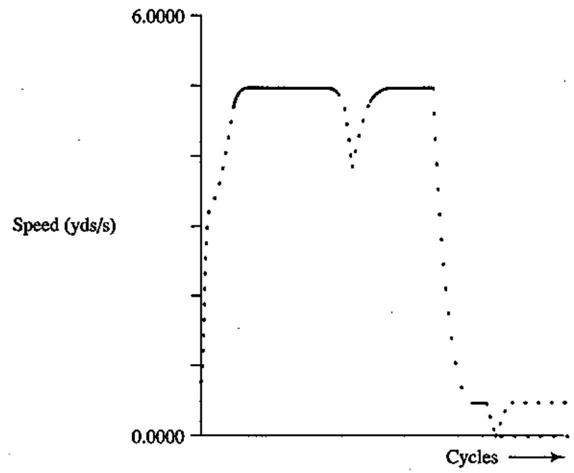


FIGURE 13.24 Cart speed test case2.

Task 7: Tune System

Tuning a fuzzy expert system begins with a comparison of the system's response with initial expectations. Deviations then become the focus of additional effort.

In most fuzzy expert system projects, the time spent developing the fuzzy sets and rules is small in comparison to the time spent tuning the system. Usually, the first series of fuzzy sets and rules provide a reasonable solution to the problem. This is perhaps one of the principal advantages of fuzzy logic. That is, common sense is used in forming the basis of the knowledge that tends to provide quick and reasonable results. However, obtaining more accurate results takes additional effort and becomes more of a art form.

A study of the results of our two tests shows that we meet most of the initial specifications. That is, the cart successfully reaches the ball and avoids the calamity of running into a tree. However, inspection of the speed plots indicate that we may have a problem.

Both Figures 13.21 and 13.24 show a problem when the cart is very close to the ball, because the speed continually oscillates between zero and some value. A programmer may overlook this point and be happy that the cart successfully reached the ball. However, the cart's rider Kathy may not share the programmer's delight when her teeth fall out due to the abrupt changes in speed.

To correct this problem, we need to inspect the speed control rules. Especially, we want to look at the rules that control the cart's speed as the cart comes "really close" to the ball. Rules 9A through 13A cover this situation. Rule 13A causes an increase in speed when the speed approaches zero, which initiates the oscillation. To prevent the oscillation we can eliminate this rule.

Figure 13.25 shows the cart's speed with RULE 13A eliminated for the test 1 situation. The elimination of this rule has prevented the speed oscillation. However, the simulation run now shows that the cart stops 7 yards from the ball versus the 3 yards found from the earlier simulation. In practice, we might now go back to make further adjustments to either the fuzzy sets or rules to correct this new problem.

In general, tuning a fuzzy logic system involves one or more of the following:

RULES
- Adding rules for special situations
- Adding premises for other linguistic variables
- Using Adverbs through hedge operators

FUZZY SETS
- Adding sets on a defined linguistic variable
- Broadening or narrowing existing sets
- Shifting laterally existing sets
- Shape adjustment of existing sets

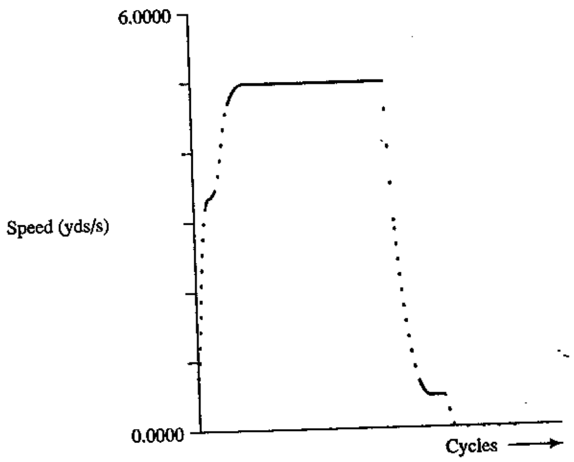


FIGURE 13.25 Cart speed adjusted rule set.

DESIGN SUGGESTION: When fine control is needed over some existing fuzzy variable, think of adding narrow fuzzy sets centered around a point of interest. In general, wide fuzzy sets provide rough control.

SUMMARY ON FUZZY LOGIC

Fuzzy logic provides the means to both represent and reason with common sense knowledge in a computer. This ability is extremely valuable to the knowledge engineer responsible for building an expert system who is confronted with an expert that explains the problem-solving tasks in common sense terms. Vague terms or rules can be represented and manipulated numerically to provide results that are consistent with the expert.

Fuzzy logic has particular value in those control applications where it is difficult or impossible to develop a traditional control system. To date, Japan has been the leader in developing fuzzy logic control systems for such diverse applications as washing machines, video camcorders, and railway systems (Self 1990) and (Waller 1989). Market Intelligence Research Corporation (MIRC) expects total revenues for fuzzy logic-based products and neural networks combined to grow to $10 billion by 1998 (Kandel 1993). Maiers and Sherif (1985) provide 450 references to fuzzy logic applications and theory. Gaines and Kohout (1977) provide an extensive discussion and reference to fuzzy logic applications.

Key issues discussed in this chapter were:

- Fuzzy logic provides the means to represent and reason with vague or ambiguous terms in a computer.
- A linguistic variable is a term used in our natural language that describes a concept that has vague or fuzzy values. In fuzzy logic, this variable is often referred to as a *fuzzy* variable.
- Values (adjectives) of fuzzy variables are represented using *fuzzy sets*, which map set elements to a degree of belief that the element belongs to the fuzzy set.
- Hedges are mathematical operations performed on an existing fuzzy set (e.g., *tall*) to produce a new fuzzy set to account for an added adverb (e.g., *very tall*).
- The two most popular fuzzy inference techniques are max–min and max–product.
- Building a fuzzy logic expert system is an iterative process, where an initial collection of fuzzy rules and fuzzy sets is formed and later tuned to meet the project's specifications.
- Tuning a fuzzy logic expert system involves adjustments to existing rules or fuzzy sets.

REFERENCES

Gaines, B.R. and L.J. Kohout, The Fuzzy Decade: A Bibliography of Fuzzy Systems and Related Topics, Int. J. Man-Machine Studies, vol. 9, pp. 1–68, 1977.

Kandel, A., Interview, PC AI, pp. 40–41, Mar./Apr., 1993.

Klir, G.J. and T.A. Foger, Fuzzy Sets, Uncertainty and Information, Prentice-Hall, Englewood Cliffs, N.J., 1988.

Kosko, B., Neural Networks and Fuzzy Systems, Prentice-Hall, Englewood Cliffs, N.J., 1992.

Maiers, J. and Y.S. Sherif, Applications of Fuzzy Set Theory, IEEE Trans. Systems, Man, and Cybernetics, vol. SMC-15, no. 1, pp. 175–189, Feb. 1985.

Rescher, N., Many-Valued Logic, McGraw-Hill, New York, 1969.

Self, K., Designing with Fuzzy Logic, IEEE Spectrum, pp. 42–44, Nov. 1990.

Waller, L., Fuzzy Logic: It's Comprehensible, It's Practical—And It's Commerical, Electronics, pp. 102–103, Mar. 1989.

Whalen, T. and B. Schott, Issues in Fuzzy Production Systems, Int. J. Man-Machine Studies, vol. 19, pp. 57–71, 1983.

Zadeh, L.A., Fuzzy Sets, Information and Control, vol. 8, 1965.

EXERCISES

1. Define some typical fuzzy variables.
2. Define typical fuzzy sets for the fuzzy variables:
   temperature
   inflation
   intelligence
3. Draw each fuzzy set defined in problem 2.
4. For the fuzzy sets defined for the variable "temperature" create the modified sets using the hedge VERY.
5. Given the following fit vectors, find their union, intersection and complements.

   (0, 0.1, 0.2, 0.8, 1)

   (0, 0.5, 0.5, 0.5, 0)
6. Given fuzzy sets A and B, where $A = (.3, .8)$ and $B = (.2, .6, .5)$, use max–min composition to show that $A' = (.4, .7)$ recalls B. Discuss the significance of this result.
7. For the fuzzy sets given in problem 6, show that $A' = (.1, .3)$ does not recall B. Discuss the significance of this result.
8. Given the rule IF A THEN B, with $A = (0, .5, 1, .5, 0)$ and $B = (0, .5, 1, .5, 0)$, and $A' = (0, .5, 0, 0, 0)$, find B' using max–min inference.
9. Repeat problem 8 using max–product inference.
10. Fuzzy rules contain premises that include fuzzy variables. Each fuzzy variable is represented by a number of fuzzy sets. A fuzzy logic expert system may include a set of rules that do not account for each established fuzzy set, but can still function properly. Explain why this is the case.

11. Using the software of your choice, build the golf cart navigation fuzzy logic system described in this chapter. Run several tests on the system where for each test several rules are deleted. Provide comment on each test result.
12. Using the software of your choice, build a fuzzy logic automobile cruise control system.
13. Using the software of your choice, build a fuzzy logic system that simulates a dog attempting to catch a cat. Assume the cat is mobile, having a speed and direction.
14. Using the software of your choice, build a fuzzy logic system for the rule-based investment example system given in Chapter 8.