A Fuzzy Logic Tutorial

Michael A. Goodrich

July 10, 2001

Background

As many of you know, there is a strong relationship between set theory and propositional logic. In this tutorial, we will exploit this relationship. More specifically, we will

- define a set membership function,
- discuss how set theory and logic are related,
- show how operations in logic can be translated into operations on crisp set membership functions,
- define the notions of a fuzzy set and a fuzzy set membership function,
- show how operations in logic can be translated into operations on fuzzy set membership functions, and
- give examples of how to use fuzzy logic in solving a problem.

A great reference, some of which is paraphrased in this tutorial, on fuzzy logic is [1].

Part 1: Conventional Logic and Set Theory

Concept #1: The Set Membership Function

As most of you know, a set is just a collection of elements. We use sets all of the time; there is the set of students who will get A's in this class, the set of NBA players who don't have tatoos, the set of cats that I like (the empty set), etc. For any object, call it x, and for any set, call it A, I can identify whether or not the object x is in the set A. The set membership function (often called the characteristic function in conventional set theory) returns one if $x \in A$ and zero otherwise. If we let μ_A represent this function then it is defined as

$$\mu_A(x) = \begin{cases} 1 & \text{If } x \in A \\ 0 & \text{Otherwise} \end{cases} .$$

For example, consider the set of integers between 3 and seven, $A = \{4, 5, 6\}$. The membership function for this set is

$$\mu_A(x) = \begin{cases} 1 & \text{If } x = 4,\ x = 5, \text{ or } x = 6 \\ 0 & \text{If } x \text{ equals any other integer.} \end{cases}.$$

The key concept is that the set membership function is unity if and only if the argument x is in the set A. Thus, rather than explicitly specifying the set by listing its elements, I can implicitly specify the set by defining it's set membership function.

Implicitly describing sets via a set membership function is sometimes helpful if I can translate operations on sets into corresponding operations on the set membership functions. In other words, rather than explicitly listing every element that satisfies a relation between two sets, I can instead implicitly specify this set by giving a function that translates the two set membership functions into a new set membership function. For example, the intersection of two sets A and B is defined as

$$A \cap B = \{x : x \in A \wedge x \in B\}.$$

The set membership function for such a set is defined as

$$\mu_{A \cap B}(x) = \begin{cases} 1 & \text{If } x \in A \cap B \\ 0 & \text{Otherwise} \end{cases}.$$

Rather than giving the definition of $\mu_{A \cap B}$ in this way, I can instead give a definition using μ_A and μ_B. One way to do this is to use the following operation

$$\mu_{A \cap B}(x) = \min\{\mu_A(x), \mu_B(x)\}.$$

Check this for yourself, but $\mu_{A \cap B}(x) = 1$ if and only if $\mu_A(x) = 1$ and $\mu_B(x) = 1$ which means that x is only in the intersection of A and B if x is in both A and B.

Interestingly, using the minimum operator is not the only way for μ_A and μ_B to be combined to form $\mu_{A \cap B}$. Alternatively, I could define $\mu_{A \cap B}$ as follows:

$$\mu_{A \cap B}(x) = \mu_A(x) \times \mu_B(x).$$

Check for yourself, but the product of μ_A and μ_B is one if and only if x is in A and x is in B.

Although we will, in general, use the minimum operator (rather than the product operator — there are often very good reasons for using the product operator), it turns out that there are a lot of ways that we can combine μ_A and μ_B such that they produce a correct set membership function for $\mu_{A \cap B}$. To represent any arbitrary combination of μ_A and μ_B, we use the (arbitrary) symbol $\star$ whence we can define the set membership function of $\mu_{A \cap B}$ as

$$\mu_{A \cap B}(x) = \mu_A(x) \star \mu_B(x).$$

If might help you remember that the $\star$ is associated with intersection if you note that a $\star$ kind of looks like an asterisk $*$, the symbol for multiplication in C.

Now that we have defined the set membership function for a crisp set and introduced a characterization of intersection using membership functions, we should also characterize union and complement. We'll start with union. The set membership function for the set obtained by taking union of two sets A and B is defined as

$$\mu_{A \cup B}(x) = \mu_A(x) \oplus \mu_B(x).$$

The most common form of the $\oplus$ operator is the maximum operator,

$$\mu_{A \cup B}(x) = \max\{\mu_A(x), \mu_B(x)\}$$

which you should verify is unity whenever x is in either A or B or both. Another common form of the $\oplus$ operator is bounded sum,

$$\mu_{A \cup B}(x) = \min\{1, \mu_A(x) + \mu_B(x)\}$$

which you should again verify produces the correct set membership function for union.

Finally, we can characterize taking the complement of a set $\bar{A} = \{x : x \notin A\}$. This set membership function is easily characterized as

$$\mu_{\bar{A}}(x) = 1 - \mu_A(x).$$

Concept #2: Set Theory and Logic

At this point in your CS career, you should be an expert at propositional logic. Most of you have also been exposed to predicate logic, and some of you have been exposed to how logic is related to set theory. In this section, I'll try to make the connection more explicit and directly applicable to fuzzy logic.

The fundamental building blocks of first order logic are objects and predicates, and the fundamental building blocks of set theory are sets. Each predicate is a mapping from the set of possible objects to the set $\{T, F\}$. In addition to describing a predicate as a function that returns a boolean value, I could just as easily describe a predicate as a set. For example, I can be dealing with the predicate *IsTall(people)* and describe this as a function that takes *people* and returns a true or a false depending on whether they are tall or not. Corresponding to this function is the set *IsTall* which is the collection of all people who are tall, *IsTall* = { *people* : *IsTall(people)=true*}.

Now that we've made the connection between predicates, objects, and sets, we are in a position to make the connection between logic and set theory. In general, we can view predicates in two ways: (a) as a function that returns a true or false, or (b) as the subset of the domain of discourse for which the predicate function is true. The most basic mathematical equivalences between logic and set theory can the be stated as

Logic	Set Theory
$A(\cdot) \wedge B(\cdot)$	$A \cap B$
$(\cdot) \vee A(\cdot)$	$A \cup B$
$\sim A(\cdot)$ (negation)	$\bar{A}$ (complement)

where $A(\cdot)$ represents a predicate (with $\cdot$ a placeholder for the variable name) with corresponding set A. That is, performing an AND between two predicate functions is equivalent to taking the intersection between two predicate sets.

Although these equivalences are important, there are two other very important equivalences from logic that we want to use. The first of these equivalences is implication, $\Rightarrow$. In logic, I often know whether predicate B is true/false when I know the truth/falsehood of predicate A; in other words, I know that there exists a relation between A and B of the form $A(\cdot) \Rightarrow B(\cdot)$. How does implication translate into a set theoretic equivalence? First, I note that $(A \Rightarrow B) \Leftrightarrow ((\sim A) \vee B)$. Second, I translate this latter equation into set theoretic notation

Logic	Set Theory
$A(\cdot) \Rightarrow B(\cdot)$	$\bar{A} \cup B$

The second equivalence of interest involves Modus Ponens. I think that it is best if we postpone how to do modus ponens until the next section, so you'll just have to read on.

Concept #3: Translating Conventional Logic Into Operations on Set Membership Functions

Now that we have mapped logic (done via predicates) into set theory, we can take the next step and map these set theoretic operations into corresponding operations on set membership function operations. We will do this mapping for AND, OR, NOT, IMPLIES, and MODUS PONENS. The first four are easy:

Logic	Set Theory	Operator
$A(\cdot) \wedge B(\cdot)$	$A \cap B$	$\mu_A \star \mu_B$
$A(\cdot) \vee B(\cdot)$	$A \cup B$	$\mu_A \oplus \mu_B$
$\sim A(\cdot)$	$\bar{A}$	$(1 - \mu_A)$
$A(\cdot) \Rightarrow B(\cdot)$	$\bar{A} \cap B$	$(1 - \mu_A) \star \mu_B$

Note in advance that one way that fuzzy logic differs from conventional logic is that the operator used for implication is different from that listed above.

To deal with modus ponens, it is necessary to talk about (1) *relations* and (2) *compositions* of relations.

Relations As discussed in Chapter 7 of the textbook, a relation is a property among various objects; it is a predicate with more than one argument. In terms of sets, a relation represents an association between elements of two or more sets. Implication can be viewed as a relation. For example, consider the statement $\forall x A(x) \Rightarrow B(x)$. We can construct a relation (a two-variable predicate) from this implication, call it $R_{A\Rightarrow B}(x, y)$. Since the implication tells us that $B(\cdot)$ is true whenever $A(\cdot)$ is, the implication relation is

$$R_{A\Rightarrow B}(x, y) = \begin{cases} 1 & \text{when } y = x \\ 0 & \text{otherwise} \end{cases} .$$

The statement $\exists x A(x) \Rightarrow B(x)$ can be thought of as a predicate that returns true whenever $y = x = G$ where $A(G) \Rightarrow B(G)$.

In addition to implications of the form $A(x) \Rightarrow B(x)$ there are other types of implications. One type of implication has the form $\forall x, y A(x) \Rightarrow B(x, y)$ which state that whenever the predicate

$A(X_i)$ is in the knowledge base then we can conclude $\forall y B(X_i, y)$. This latter statement is akin to concluding $B(X_i, Y_1) \wedge \ldots \wedge B(X_i, Y_n)$ where the variable y has a domain of $\{Y_1, \ldots, Y_n\}$. In essence, this type of implication says that whenever A holds then we can conclude something about x's relationship to the family y. A particularly useful type of implication in fuzzy logic has the form $\forall x, y A(x) \Rightarrow B(y)$. This implication encodes a relationship between two predicates on different domains.

Let's go over a fairly complete discussion that deals with relations in general rather than just dealing with implication. Let R_{AB} denote a relation between the sets $A = \{a_1, a_2, a_3\}$ and $B = \{b_1, b_2\}$. We can construct a relational matrix as

$$\begin{array}{cc} & \begin{array}{cc} b_1 & b_2 \end{array} \\ \begin{array}{c} a_1 \\ a_2 \\ a_3 \end{array} & \left(\begin{array}{cc} \text{true} & \text{false} \\ \text{true} & \text{true} \\ \text{false} & \text{false} \end{array}\right) \end{array}$$

which translates into a set membership function for this relation as

$$\mu_{R_{AB}}(a, b) = \begin{cases} 1 & \text{if and only if } (a, b) \in R_{AB} \\ 0 & \text{otherwise} \end{cases}$$

where $R_{AB} = \{(a, b) : R_{AB}(a, b) = \text{true}\}$. This yields an alternative form for the relational matrix

$$\begin{array}{cc} & \begin{array}{cc} b_1 & b_2 \end{array} \\ \begin{array}{c} a_1 \\ a_2 \\ a_3 \end{array} & \left(\begin{array}{cc} 1 & 0 \\ 1 & 1 \\ 0 & 0 \end{array}\right) \end{array}$$

Example:

Earlier, we said that implication was a type of relation. To illustrate this and to introduce an example that we will use throughout this section, consider the problem of trying to decide whether or not to turn on the heater in your apartment. Suppose that I have a thermometer that gives three readings, $A = \{\text{“}T < 30\text{”}, \text{“}30 \leq T \leq 60\text{”}, \text{“}T > 60\text{”}\}$ *where I have used quotation marks to indicate that these statements can be interpreted as predicates. If you prefer, you can think of these three predicates as* $A = \{\text{IsCold, IsCool, NotCold}\}$. *In addition to these three input predicates, I have two actions available* $B = \{\textit{HeatOn}, \textit{HeatOff}\}$. *Suppose further that I have a rule base that says*

$$\begin{array}{ccc} \textit{Reading } (a) & \Rightarrow & \textit{Action } (b) \\ T < 30 & \Rightarrow & \textit{HeatOn} \\ 30 \leq T \leq 60 & \Rightarrow & \textit{HeatOn} \\ T > 60 & \Rightarrow & \textit{HeatOff} \end{array}$$

In this case, the implies in the statement “$T < 30$” $\Rightarrow$ *HeatOn does not mean ”if* $T < 30$ *it follows logically that the heat is on” but rather ”if* $T < 30$ *it follows logically from* *what my goals are that the heat should be turned on." In this latter case, implication is nothing more than a relation between readings and actions*

$$\begin{array}{cc} & \begin{array}{cc} \textit{HeatOn} & \textit{Heatoff} \end{array} \\ \begin{array}{c} T < 30 \\ 30 \leq T \leq 60 \\ T > 60 \end{array} & \left(\begin{array}{cc} 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{array} \right) \end{array}$$

Compositions Now, let's explore what happens if I want to relate a relation on one set of domains to a relation on a different set. I'll give two examples. **Example:**

In the first example, suppose that I work for BYU's employment office and my job is to help students find jobs. To find a job, I must identify some potential employers. One way to do this is to identify which faculty members students might know and then identify which potential employers are known by which faculty members. Let Mentor(x,y) *encode the relationship between students x and faculty members y. Let* Friend(y,z) *encode the relationship between faculty members y and potential employers z. My job is to find out if there is a network between the students and the potential employers. Let* Network(x,z) *encode this relationship. The* Network *relationship is given by the composition of* Mentor *and* Friend,

$$\textit{Network(x,z)} = \textit{Mentor(x,y)} \circ \textit{Friend(y,z)}.$$

Suppose that there are three students, {Curtis, Jacob, Nancy}, three faculty members, {Mike, Todd, Tony}, and three employers, {Bill, George, Sarah}. The Mentor relation is given by

	Curtis	*Jacob*	*Nancy*
Mike	*1*	*1*	*0*
Todd	*0*	*0*	*1*
Tony	*0*	*1*	*1*

which says that Curtis has been mentored by Mike but no other. The Friend relation is given by

	Mike	*Todd*	*Tony*
Bill	*1*	*0*	*0*
George	*0*	*1*	*0*
Sarah	*0*	*0*	*1*

.

Which students have a network to which employers? We can do this by finding when mentor relations hold between a faculty member and a student, and when a friendship holds between that faculty member and an employer. For example, since $\mu_{\text{Mentor}}(Curtis, Mike) = 1$ and $\mu_{\text{Friend}}(Mike, Bill) = 1$ it is apparent that there is a network between Curtis and

Bill, $\mu_{\text{Mentor}\circ\text{Friend}}(Curtis, Bill) = 1$. *The membership functions for this network relation can be given by*

$$\begin{aligned}\mu_{\text{Network}}(x,z) &= \mu_{\text{Mentor}\circ\text{Friend}}(x,z) \\ &= \max_y \left[\mu_{\text{Mentor}}(x,y) \star \mu_{\text{Friend}}(y,z)\right].\end{aligned}$$

The $\star$ *indicates that for a particular* y *both predicates must be true for the Network to pass through faculty member* y*, and the* max *indicates that the Network predicate is true if it is true for any faculty member. The resulting Network predicate is given by*

	Curtis	*Jacob*	*Nancy*
Bill	*1 (through Mike)*	*1 (through Mike)*	*0*
George	*0*	*0*	*1 (through Todd)*
Sarah	*0*	*1 (through Tony)*	*1 (through Tony)*

Example:

As a second example, let's return to the temperature/heater example. Suppose that you bring a date to your (underheated) apartment and she or he has a thermometer that reads temperature in one degree increments. You don't want to change your reading/action rulebase (it was programmed in Fortran in 1978), so you instead write a new program that translates the temperature on your date's thermometer into one of the three classes known to your Fortran program. In other words, you create a new relation Q_{CA}*, where* $C = \{0, 1, \ldots, 120\}$ *is the range of the thermometer. The relation is defined in terms of the membership function* $\mu_{Q_{CA}}(c,a)$ *as*

	a		
$c = T$	"$T < 30$"	"$30 < T \leq 60$"	"$T > 60$"
$c < 30$	*1*	*0*	*0*
$30 \leq c \leq 60$	*0*	*1*	*0*
$c > 60$	*0*	*0*	*1*

Let P_{CB} *denote the new relation between the temperature reading from your date's thermometer and the decision to turn on your heater. How do I combine* Q_{CA} *with* R_{AB} *to find* P_{CB}*? We do this by the composition operator,*

$$P_{CB}(c,b) = Q_{CA}(c,a) \circ R_{AB}(a,b)$$

which is defined as a relation on $C \times B$ *such that* $(c,b) \in P_{CB}$ *if and only if there exists at least one* $a \in A$ *such that* $(a,b) \in R_{AB}$ *and* $(c,a) \in Q_{CA}$*. In other words, you will turn the HeatOn whenever your date reports a temperature for which the relation between this temperature and either one of the categories "*$T < 30$*" and "*$30 \leq T \leq 60$*" is true.*

The trick is to come up with a formula on the membership functions of $\mu_{R_{AB}}$ *and* $\mu_{Q_{CA}}$ *that correctly produces* $\mu_{P_{CB}}$*. The formula is given by*

$$\mu_{P_{CB}}(c,b) = \mu_{Q\circ P}(c,b) = \max_{a\in A} \mu_{Q_{CA}}(c,a) \star \mu_{R_{AB}}(a,b).$$

Basically, this formula says that the truth of the predicate P_{CB}*, which was created by combining the predicates* Q_{CA} *and* R_{AB}*, is obtained by seeing if both predicates* Q *and* R *are simultaneously true for any object* $a \in A$*. If I can find at least one object for which both predicates are true then the composition of these two predicates is also true. Otherwise, the composition is false.*

Let's check to see that this works for the case when $\star$ *is implemented as a minimum,*

$$\mu_{Q \circ P}(c,b) = \max_{a \in A} \min\{\mu_{Q_{CA}}(c,a), \mu_{R_{AB}}(a,b)\}.$$

Suppose that your date's thermometer reads 32. Then $c = 32$*. We want to find out if HeatOn is true. So, we calculate*

$$\begin{aligned}
\mu_{Q\circ P}(32, \mathrm{HeatOn}) &= \max_{a \in \{\text{“}T<30\text{”}, \text{“}30 \leq T \leq 60\text{”}, \text{“}T>60\text{”}\}} \min\{\mu_{Q_{CA}}(32, a), \mu_{R_{AB}}(a, \mathrm{HeatOn})\} \\
&= \max \left\{ \begin{array}{l} \min\{\mu_{Q_{CA}}(32, \text{“}T < 30\text{”}), \mu_{R_{AB}}(\text{“}T < 30\text{”}, \mathrm{HeatOn})\}, \\ \min\{\mu_{Q_{CA}}(32, \text{“}30 \leq T \leq 60\text{”}), \mu_{R_{AB}}(\text{“}30 \leq T \leq 60\text{”}, \mathrm{HeatOn})\}, \\ \min\{\mu_{Q_{CA}}(32, \text{“}T > 60\text{”}), \mu_{R_{AB}}(\text{“}T > 60\text{”}, \mathrm{HeatOn})\} \end{array} \right\} \\
&= \max \left\{ \begin{array}{l} \min\{0,1\}, \\ \min\{1,1\}, \\ \min\{0,0\} \end{array} \right\} \\
&= \max\{0,1,0\} \\
&= 1.
\end{aligned}$$

So, at least for this temperature reading you should turn the HeatOn.

Now, suppose that your date's thermometer reads 82. Then $c = 82$*. We want to find out if HeatOn is true. So, we calculate*

$$\begin{aligned}
\mu_{Q\circ P}(82, \mathrm{HeatOn}) &= \max_{a \in \{\text{“}T<30\text{”}, \text{“}30 \leq T \leq 60\text{”}, \text{“}T>60\text{”}\}} \min\{\mu_{Q_{CA}}(82, a), \mu_{R_{AB}}(a, \mathrm{HeatOn})\} \\
&= \max \left\{ \begin{array}{l} \min\{\mu_{Q_{CA}}(82, \text{“}T < 30\text{”}), \mu_{R_{AB}}(\text{“}T < 30\text{”}, \mathrm{HeatOn})\}, \\ \min\{\mu_{Q_{CA}}(82, \text{“}30 \leq T \leq 60\text{”}), \mu_{R_{AB}}(\text{“}30 \leq T \leq 60\text{”}, \mathrm{HeatOn})\}, \\ \min\{\mu_{Q_{CA}}(82, \text{“}T > 60\text{”}), \mu_{R_{AB}}(\text{“}T > 60\text{”}, \mathrm{HeatOn})\} \end{array} \right\} \\
&= \max \left\{ \begin{array}{l} \min\{0,1\}, \\ \min\{0,1\}, \\ \min\{1,0\} \end{array} \right\} \\
&= \max\{0,0,0\} \\
&= 1.
\end{aligned}$$

So, at least for this temperature reading you should not turn the HeatOn.

Whew! To define all of the relation, I have to do this procedure for every temperature and every action. When I am done, I have defined a set of temperatures for which the heat should be turned on. (We'll use a graphical technique to help us do this in the fuzzy domain.)

The rule for creating the membership function of a new predicate obtained by combining predicates $A(x, y)$ and $B_{y,z}$ is given by

$$\mu_{A \circ B}(x, z) = \max_y \left[\mu_A(x, y) \star \mu_B(y, z)\right]$$

Generalized Modus Ponens

- Review Generalized Modus Ponens
- Introduce the Substitution Predicate
- Review Implication Relation
- Map $A(x, y) \circ B(y, z)$ into Substitution and Implication Predicates

In chapter 9 of the book, the author defines generalized modus ponens (GMP) for first order logic. This inference rule requires three things: a rule (implication), a list of predicates that are part of the rule antecedant, and a substitution that binds all variables in such a way that the individual predicates match the predicates in the antecedant. Thus, there are three steps that must be done to apply this inference rule: find the substitution, combine this substitution to the implication, and then conclude the consequent with the substitution in place.

It would be nice if we could use composition to determine how to find the set-membership function implementation of GMP. To do this, we need to figure out how the three steps in GMP translate into predicates and composition. Let's summarize what we know:

GMP	Composition
SUBST$(\theta, P(\cdot))$	$\mu_A(x, y)$
$P(x) \Rightarrow Q(x)$	$\mu_B(y, z)$
SUBST$(\theta, Q(\cdot))$	$\mu_{A \circ B}(x, z)$

The most obvious step is to set $\mu_B(y, z) = \mu_{P \Rightarrow Q}(y, z)$ where this latter relation is the implication relation that we talked about earlier. What should we do with $\mu_A(x, y)$? Clearly, y must be treated as the same variable as the y in $\mu_B(y, z)$. The other variable that we have to play with is our binding list θ, so somehow we must associate x in $\mu_A(x, y)$ with θ in SUBST$(\theta, P(\cdot)$. We will define a substitution-based predicate as follows:

$$\mu_{P^*}(\theta, y) = \begin{cases} 1 & \text{if } y = Const \text{ where } \theta = \{y/Const\} \\ 0 & \text{otherwise} \end{cases}.$$

This predicate returns true whenever my free variable y corresponds to its bound value and false otherwise. When I compose these two predicates, I get

$$\mu_{Q^*}(\theta, z) = \mu_{P^* \circ [P \Rightarrow Q]}(\theta, z).$$

Let's summarize what we've derived:

GMP	Composition
SUBST$(\theta, P(\cdot))$	$\mu_{P^*}(\theta, y)$
$P(x) \Rightarrow Q(x)$	$\mu_{P \Rightarrow Q}(y, z)$
SUBST$(\theta, Q(\cdot))$	$\mu_{Q^*}(\theta, z)$

Example:

We should now look at an example to help us put these pieces together. Consider a world with two people in it, {Goodrich,Bradley} where Goodrich is about six feet tall and Bradley is about seven and a half feet tall. These are our objects. We also have a rule in our knowledge base that says IsTall(x) ⇒ IsntShort(x). Someone makes the observation that Bradley is tall, so add the following statement into our knowledge base: IsTall(Bradley). We want to apply GMP to deduce IsntShort(Bradley).

The first thing that we notice is that observing IsTall(Bradley) corresponds to creating the substitution list $\theta = \{x/\text{Bradley}\}$ *and the corresponding predicate IsTall**(θ, y). *Let's look at the relational matrices for* $\mu_{\text{IsTall}^*}(\theta, x)$ *and* $\mu_{\text{IsTall}\Rightarrow\text{IsntShort}}(y, z)$ *which are given by, respectively,*

y	$\theta = \{y/\text{Bradley}\}$
Bradley	*1*
Goodrich	*0*

and

z	y Bradley	y Goodrich
Bradley	*1*	*0*
Goodrich	*0*	*1*

When I take the composition of these two predicates, I get

$$\begin{aligned}
\mu_{\text{IsntShort}^*}(\theta, z) &= \max_y \left[\mu_{\text{IsTall}^*}(\theta, y) \star mu_{\text{IsTall}\Rightarrow\text{IsntShort}}(y, z)\right] \\
&= \mu_{\text{IsTall}^*}(\theta, \text{Bradley}) \star mu_{\text{IsTall}\Rightarrow\text{IsntShort}}(\text{Bradley}, z) \\
&= \begin{cases} 1 & z = \textit{Bradley} \\ 0 & z = \textit{Goodrich} \end{cases} .
\end{aligned}$$

This last equation says that we conclude a predicate that is unity when z *is Bradley and zero otherwise. In other words, we conclude that Bradley isn't short.*

One final comment is in order. Since θ *isn't used in the final predicate, we can implicitly include this binding list in both* $\mu_{\text{IsTall}^*}(\theta, y)$ *and* $\mu_{\text{IsntShort}^*}(\theta, z)$ *yielding, respectively,* $\mu_{\text{IsTall}^*}(y)$ *and* $\mu_{\text{IsntShort}^*}(z)$.

Further Simplifying Modus Ponens One more item deserves attention. Consider what happens when we use the minimum operator to implement the $\star$ function. We'll leave the world of concrete examples and use the generic predicates B^*, A^*, and $A \Rightarrow B$. When we apply the minimum operator, the max-star composition applied to GMP becomes

$$\begin{aligned}
\mu_{B^*}(b) &= \max_{a \in A} \left[\mu_{A^*}(a) \star \mu_{A\Rightarrow B}(a, b)\right] \\
&= \max_{a \in A} \min \left[\mu_{A^*}(a), \mu_{A\Rightarrow B}(a, b)\right] .
\end{aligned}$$

In general, it is convenient to use μ_A and μ_B to determine $\mu_{A\Rightarrow B}$ rather than figuring out what this is as we did in the previous example. Using the equivalence $(A \Rightarrow B) \Leftrightarrow (\not{A} \cup B)$, substituting $\mu_{A\Rightarrow B}(a,b) = (1-\mu_A(a)) \oplus \mu_B(b)$, and replacing $\oplus$ with *max* we get

$$\begin{aligned}
\mu_{B^*}(b) &= \max_{a\in A} \min\left[\mu_{A^*}(a), \max(1-\mu_A(a), \mu_B(b))\right] \\
&= \max_{a\in A} \max\Big[\min\left(\mu_{A^*}(a), (1-\mu_A(a))\right), \min\left(\mu_{A^*}(a), \mu_B(b)\right)\Big].
\end{aligned}$$

Since $\mu_{A^*}(a) = 0$ except when a equals the specified constant in the binding list (call this constant X), and since $(1-\mu_A(X)) = 0$ then $\min\left(\mu_{A^*}(a), (1-\mu_A(a))\right) = 0$. Thus, $\mu_{B^*}(b)$ becomes

$$\begin{aligned}
\mu_{B^*}(b) &= \max_{a\in A} \max\Big[\mu_{A^*}(a), (1-\mu_A(a))), \min\left(\mu_{A^*}(a), \mu_B(b)\right)\Big] \\
&= \max_{a\in A} \max\Big[\min\left(0, \min\left(\mu_{A^*}(a), \mu_B(b)\right)\right)\Big] \\
&= \max_{a\in A} \min\left(\mu_{A^*}(a), \mu_B(b)\right).
\end{aligned}$$

Look at this carefully. It should seem strange to you that we are defining $\mu_B(b)$ as a function of $\mu_{A^*}(a)$ and $\mu_B(b)$. This is a bit like defining the term *redundancy* as *the quality or state of having redundancy*. The solution to this is to assume that $B(\cdot)$ is true unless told otherwise. Thus, we will set $\mu_B(b) = 1$ for all b (we will change this in the fuzzy logic section). When we do this, we basically say that the truth of B^* equals the truth of A^*, but since each of these predicates is the bound-version of the predicate, we are essentially concluding $\text{SUBST}(\theta, B(\cdot))$.

One final note is in order. Note that we can save a lot of time if we take $\mu_{B^*}(b) = \max_{a\in A} \min\left(\mu_{A^*}(a), \mu_B(b)\right)$ and generalize it to

$$\mu_{B^*}(b) = \max_{a\in A} \mu_{A^*}(a) \star \mu_B(b).$$

This is the form of GMP that we will find most useful.

Now that we have introduced a bunch of terminology and some tools for manipulating the symbols, it is time to do a fairly complete example. Hopefully, the following example will help make things more clear.

An Example

We can now play some games with these set membership functions and predicates. I'm going to use the world of professional basketball and the decision of which players will be drafted. In this world, the objects of interest are players, and predicates are properties that these players do or do not possess. The predicates of interest for this world are

Predicate	Meaning
IsTall	The player is tall by NBA standards
CanDefend	The player plays defense well
CanShoot	The player can shoot well
CanRebound	The player gets lots of rebounds
BePopular	The player will be liked by advertisers
CanPass	The player is a good passer

Given these predicates, I want to construct a rule-base of which players will be drafted. Thus, I have one additional predicate, the *WillBeDrafted* consequent predicate. Based on the predicates above, I will construct a rule base that relates the antecedant predicates to the consequent predicates, each as a function of player p.

$$\begin{array}{ll} \text{Rule \#1} & \text{IsTall}(p) \wedge \text{CanRebound}(p) \wedge \text{CanDefend}(p) \Rightarrow \text{WillBeDrafted}(p) \\ \text{Rule \#2} & \text{IsTall}(p) \wedge \text{CanRebound}(p) \wedge \text{BePopular}(p) \Rightarrow \text{WillBeDrafted}(p) \\ \text{Rule \#3} & \text{CanDefend}(p) \wedge \text{CanShoot}(p) \wedge \text{BePopular}(p) \Rightarrow \text{WillBeDrafted}(p) \\ \text{Rule \#4} & \text{CanDefend}(p) \wedge \text{CanShoot}(p) \wedge \text{CanPass}(p) \Rightarrow \text{WillBeDrafted}(p) \end{array}$$

I have four players that I am considering drafting. These players have the following properties, presented in set-membership function form.

Membership	Mikeli Wesley	John Stockton	Greg Ostertag	Michael Goodrich
μ_{IsTall}	1	0	1	0
$\mu_{\text{CanDefend}}$	0	1	1	0
μ_{CanShoot}	1	1	0	0
μ_{CanPass}	0	1	0	0
$\mu_{\text{CanRebound}}$	0	0	1	0
$\mu_{\text{BePopular}}$	0	0	0	1

I have four rules in my rule base, all of which are made up of a series of antecedants connected by ANDs. We can apply logic in two steps. The first is to apply the $\star$ operator on the predicates in the antecedant to determine the truth of the antecedant. Doing so yields the truth of the antecedant for each rule which is denoted, for example, by $\mu_{\text{Ant 1}}(\cdot)$ as the antecedant for rule 1:

Membership	Mikeli Wesley	John Stockton	Greg Ostertag	Michael Goodrich
$\mu_{\text{Ant 1}}$	0	0	1	0
$\mu_{\text{Ant 2}}$	0	0	0	0
$\mu_{\text{Ant 3}}$	0	0	0	0
$\mu_{\text{Ant 4}}$	0	1	0	0

We can now apply GMP to determine the truth level of the consequent *WillBeDrafted*. We do this by using the formula

$$\begin{aligned} \mu_{\text{WillBeDrafted}^*}(q) &= \max_p \left\{ \mu_{\text{Ant i}^*}(p) \star \mu_{\text{Ant i} \Rightarrow \text{WillBeDrafted}}(p, q) \right\} \\ &= \max_p \mu_{\text{Ant i}^*}(p) \star \mu_{\text{WillBeDrafted}}(q) \end{aligned}$$

where this last step was obtained by using the simple form of GMP we derived in the previous section. We already know that Ant $1^*(\cdot)$ is the predicate *Ant* 1 bound to a particular player. Furthermore, we decided that, by convention, we were going to assume $\mu_B(\cdot)$ was true when used in the inference rule. Thus, my formula becomes

$$\begin{aligned} \mu_{\text{WillBeDrafted}}(q) &= \max_p \mu_{\text{Ant i}^*}(p) \star \mu_{\text{WillBeDrafted}}(q) \\ &= \mu_{\text{Ant i}^*}(p) \end{aligned}$$

which returns one if and only if q is the name of the player that was bound to Ant i to create the predicate Ant i*.

We now plug in the truth of each rule and make the following conclusions

Membership	Mikeli Wesley	John Stockton	Greg Ostertag	Michael Goodrich
$\mu_{WillBeDrafted}$	0	0	1	0
$\mu_{WillBeDrafted}$	0	0	0	0
$\mu_{WillBeDrafted}$	0	0	0	0
$\mu_{WillBeDrafted}$	0	1	0	0

Thus, only John Stockton and Greg Ostertag will be drafted.

Part 2: Fuzzy Logic

You may have hesitated a little bit when you saw that Mikeli Wesley was listed as tall. For those of you who don't know, Wesley was played for the BYU men's basketball team last year, and he is around six and a half fee tall. Clearly, Wesley is tall by CS department standards, but by NBA standards he has a pretty average height. What we need is some mechanism for encoding how much a person satisfies the *IsTall* predicate. This is where fuzzy logic applies; it gives a mechanism for assigning degree of membership other than 0 or 1.

Fuzzy Sets and Linguistic Variables

A linguistic variable is a predicate that can return something other than just true or false. For example, the predicate *IsTall* is probably false for someone who is four feet tall, is probably true for some who is seven and a half feet tall, and is somewhere in between for someone like me (at six feet and three-quarters inches). Associated with this linguistic variable is a set membership function that can take on values in the interval $[0, 1]$ rather than just from the set $\{0, 1\}$. A fuzzy set is a set with such a set membership function. For example, the fuzzy set for the *IsTall* linguistic variable might have a set membership function as shown in Figure 1.

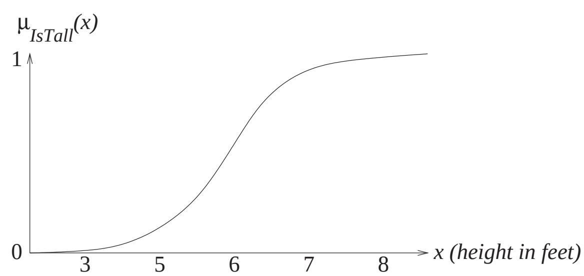

Figure 1: Degree of membership of height in the IsTall set.

Given the fuzzy characterization of *IsTall*, our task is to figure out how to infer some conclusion. To make this discussion complete, it is helpful to have a concrete example. To this end, let's return to the professional basketball players example again. The four players that I am considering drafting have the following properties, presented in fuzzy set-membership function form.

Membership	Mikeli Wesley	John Stockton	Greg Ostertag	Michael Goodrich
μ_{IsTall}	0.8	0.3	0.95	0.3
$\mu_{\text{CanDefend}}$	0.2	0.8	0.8	0.1
μ_{CanShoot}	0.8	0.9	0.2	0.3
μ_{CanPass}	0.2	1.0	0.1	0.2
$\mu_{\text{CanRebound}}$	0.3	0.2	0.8	0.2
$\mu_{\text{BePopular}}$	0.3	0.4	0.1	0.9

I pulled these numbers out of the air, but it might help to motivate them a little further. The *IsTall* membership function is something like that shown in Figure 1. The *CanDefend* membership function can be obtained by figuring out how many steals, taken-charges, or blocked-shots each player recorded and then rank ordering the results. *CanShoot* can be obtained from shooting percentage, *CanPass* from average assists per game, *CanRebound* from average rebounds per game, and *BePopular* by letting the students in Deseret Towers rank the players on a scale from one to ten (notice that Mike Goodrich is the most popular).

I can now find the truth value for the antecedants in each rule using the $\star$ operator; I'll use minimum yielding,

$$\mu_{\text{Ant1}} = \min\{\mu_{\text{IsTall}}, \mu_{\text{CanRebound}}, \mu_{\text{CanDefend}}\}$$

Plugging in the actual membership functions yields,

Membership	Mikeli Wesley	John Stockton	Greg Ostertag	Michael Goodrich
μ_{Ant1}	$\min(0.8, 0.3, 0.2)$	$\min(0.3, 0.2, 0.8)$	$\min(0.95, 0.8, 0.8)$	$\min(0.3, 0.2, 0.1)$
μ_{Ant2}	$\min(0.8, 0.3, 0.3)$	$\min(0.3, 0.2, 0.4)$	$\min(0.95, 0.8, 0.1)$	$\min(0.3, 0.2, 0.9)$
μ_{Ant3}	$\min(0.2, 0.8, 0.3)$	$\min(0.8, 0.9, 0.4)$	$\min(0.8, 0.2, 0.1)$	$\min(0.1, 0.3, 0.9)$
μ_{Ant4}	$\min(0.2, 0.8, 0.2)$	$\min(0.8, 1.0, 0.8)$	$\min(0.8, 0.1, 0.8)$	$\min(0.1, 0.3, 0.2)$

which simplifies to

Membership	Mikeli Wesley	John Stockton	Greg Ostertag	Michael Goodrich
μ_{Ant1}	0.2	0.2	0.8	0.1
μ_{Ant2}	0.3	0.2	0.1	0.2
μ_{Ant3}	0.2	0.4	0.1	0.1
μ_{Ant4}	0.2	0.8	0.1	0.1

Generalized Modus Ponens Revisited

I am now almost in position to use the max-star composition to translate the truth of the antecedant into the truth value of the consequent *WillBeDrafted*. First, I need to define how much each antecedant implies each consequent. Earlier, we computed the equivalences

Logic	Set Theory	Fuzzy Operator
$A \Rightarrow B$	$\bar{A} \cup B$	$(1 - \mu_A) \oplus \mu_B$

but this actually yields some counter-intuitive results when applied to fuzzy decision problems (mostly because of the False⇒True quirk of propositional logic). To fake our way around this problem, we will use the $\star$ operator for implication, most often either the minimum or the product operator. Thus, we use

Logic	Set Theory	Fuzzy Operator
$A \Rightarrow B$	$\bar{A} \cap B$	$\mu_A \star \mu_B$.

When we do this, we can simplify the fuzzy version of generalized modus ponens as follows:

$$\mu_{B^*}(b) = \max_{a \in A} \Big[\mu_{A^*}(a) \star \mu_A(a) \star \mu_B(b)\Big].$$

Fuzzification

In the previous discussion, $A^*(\cdot)$ was defined as a predicate that was true for one and only one value of the variable, the value that the variable was bound to. In general, we can think of this variable-binding process of making an observation (we are evaluating Mikeli Wesley) and doing inference with this observation (Wesley won't be drafted). When we are dealing with observations from sensors, it is often the case that the sensor reading is imprecise. Consequently, a particular value of a sensor should have a fuzzy set membership value of one, but values near this observation should also have some degree of truth. The process of taking an observation and creating a fuzzy set from it is called *fuzzification*.

There are some nice papers that use sophisticated forms of fuzzification, but in this class we will restrict attention to *singleton fuzzifiers*. Such fuzzifiers assign a set membership value of one to the observation and zero to everything else. The term *singleton* is suggestive of having only the variable bound to a single constant, hence the proposition is true of a set with only a single value.

Completing The Basketball Example

We now have the pieces in place to finish the basketball example. We computed the truth value of each antecedant, which I will include here again for readability

Membership	Mikeli Wesley	John Stockton	Greg Ostertag	Michael Goodrich
μ_{Ant1}	0.2	0.2	0.8	0.1
μ_{Ant2}	0.3	0.2	0.1	0.2
μ_{Ant3}	0.2	0.4	0.1	0.1
μ_{Ant4}	0.2	0.8	0.1	0.1

Each of these membership functions represent the fuzzy sets that were created when the free variable was bound to the players listed in the table. Applying modus ponens gives

$$\begin{aligned}
\mu_{\text{WillBeDrafted}}(q) &= \max_{q \in \text{Ant i}} \Big\{\mu_{\text{Ant i}^*}(q) \star [\mu_{\text{Ant i}}(q) \oplus \mu_{\text{WillBeDrafted}}(q)]\Big\} \\
&= \max_{q \in \text{Ant i}} \Big\{\mu_{\text{Ant i}^*}(q), \mu_{\text{WillBeDrafted}}(q)\Big\}
\end{aligned}$$

which yields the truth of the *WillBeDrafted* predicate for each player

Membership	Mikeli Wesley	John Stockton	Greg Ostertag	Michael Goodrich
$\mu_{WillBeDrafted}$	0.2	0.2	0.8	0.1
$\mu_{WillBeDrafted}$	0.3	0.2	0.1	0.2
$\mu_{WillBeDrafted}$	0.2	0.4	0.1	0.1
$\mu_{WillBeDrafted}$	0.3	0.8	0.1	0.1

To decide who will be drafted, we (arbitrarily) choose a threshold and determine whose truth value is above a threshold. Again, it looks like Stockton and Ostertag will probably be drafted.

An Example from Control

Consider the problem diagrammed in Figure 2. The task is to make sure that our telescope points

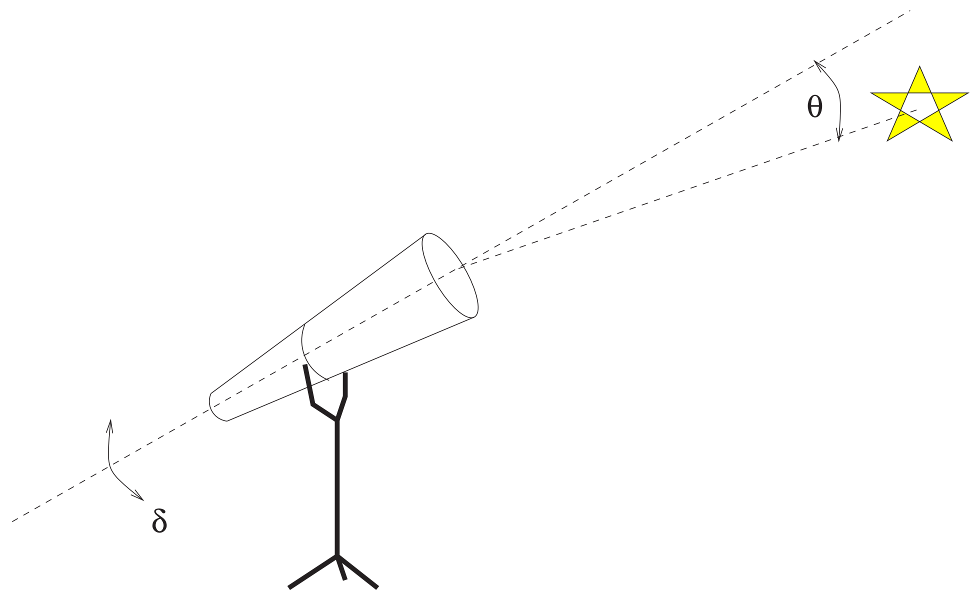

Figure 2: Controlling the direction the telescope points.

toward to a star and tracks the star's motion through the night. We want to design a system that will automatically tell us what δ, the change in elevation of the telescope, should be to make sure that the value of θ allows for correct viewing of the star (i.e., $\theta \approx 0$).

Setting Up The Problem To create such a controller, we first need to define some fuzzy sets. We will define such sets over the two variables that we have, δ and θ. In essence, we will discretize the domains of these variables. Let's use five discrete chunks, call them ZERO, SMALL POSITIVE, SMALL NEGATIVE, LARGE POSITIVE and LARGE NEGATIVE, and make up set membership functions for each chunk. Note that we have defined the domain of the output predicates prior to applying any form of reasoning. I like to think of this prior restriction as akin to saying that no matter what the truth value of the fuzzy antecedant, the fuzzy consequent can never exceed the values chosen *a priori* for the output consequent.

The input and output set membership functions are shown in Figures 3 and 4, respectively. membership functions

Our task is now to construct a rule base. We will follow the simple protocol that if the telescope is pointed too high then θ is negative. If we want to move the telescope so that it points to a lower elevation than δ must be positive. Our rules are of the form *If θ is large and negative (we are aiming too high) then δ should be large and positive (we should decrease the elevation of the telescope).* We formalize these as follows:

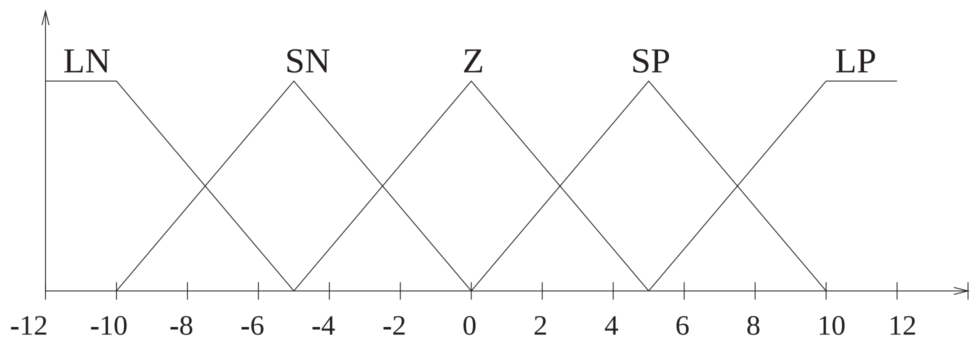

Figure 3: Input fuzzy sets

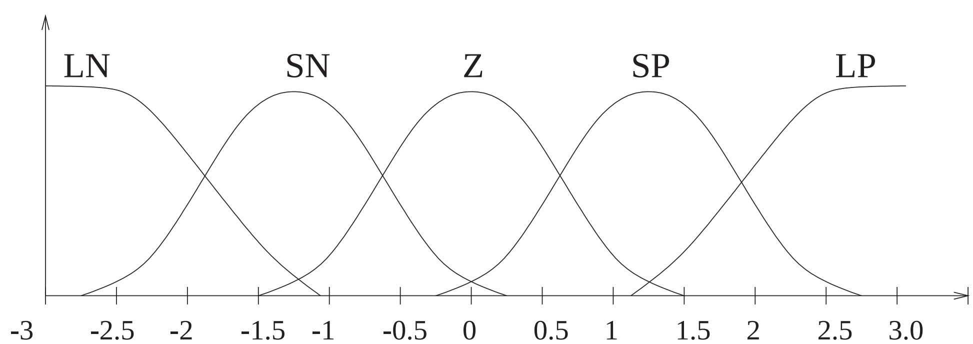

Figure 4: Output fuzzy sets

Antecedant	$\Rightarrow$	**Consequent**
θ is LN	$\Rightarrow$	δ is LP
θ is SN	$\Rightarrow$	δ is SP
θ is Z	$\Rightarrow$	δ is Z
θ is SP	$\Rightarrow$	δ is SN
θ is LP	$\Rightarrow$	δ is LN

Reasoning Now, suppose that we observe the $\theta = 0$. We look at each of my input fuzzy sets to determine the truth of each possible antecedant:

Membership	**Value**
Large Negative	$\mu_{\mathrm{LN}}(\theta = 0) = 0$
Small Negative	$\mu_{\mathrm{SN}}(\theta = 0) = 0$
Zero	$\mu_{\mathrm{Z}}(\theta = 0) = 1.0$
Small Positive	$\mu_{\mathrm{SP}}(\theta = 0) = 0$
Large Positive	$\mu_{\mathrm{LP}}(\theta = 0) = 0$

.

This is diagrammed on the left-hand side of Figure 5. In the diagram, each row on the left-hand

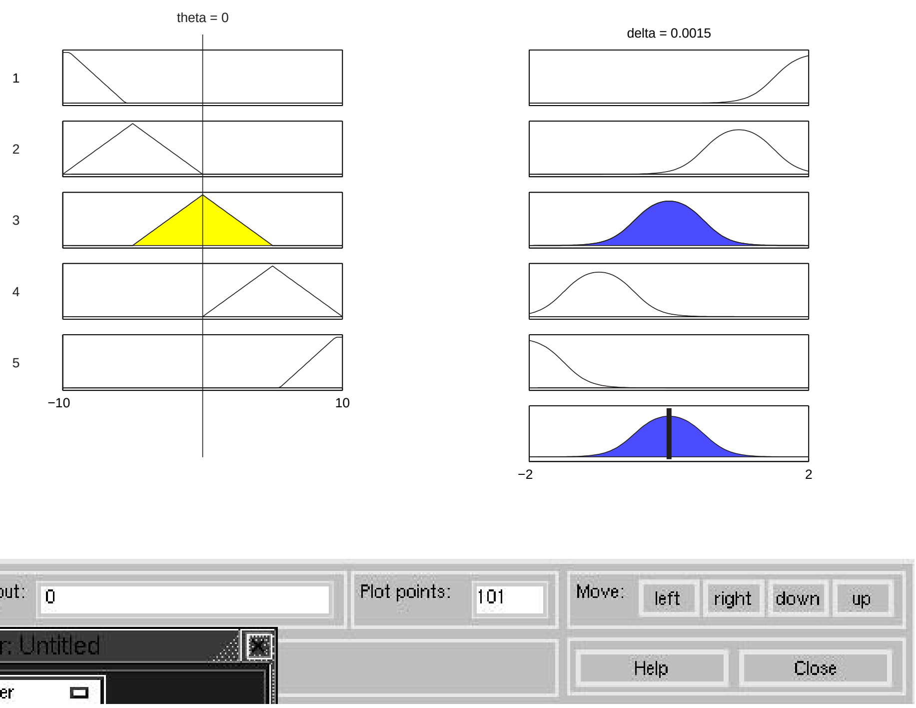

Figure 5: Output fuzzy sets for $\theta = 0$.

side represents one of my fuzzy inputs; e.g., the top triangle is a plot of $\mu_{\mathrm{LN}}(\theta)$ shown as a function of θ. The observation $\theta = 0$ is illustrated as a vertical line through each membership function with each antecedant triggered by this observation shaded.

To further illustrate the process of reasoning, consider $\theta = 2$ as diagrammed in Figure 6. Since

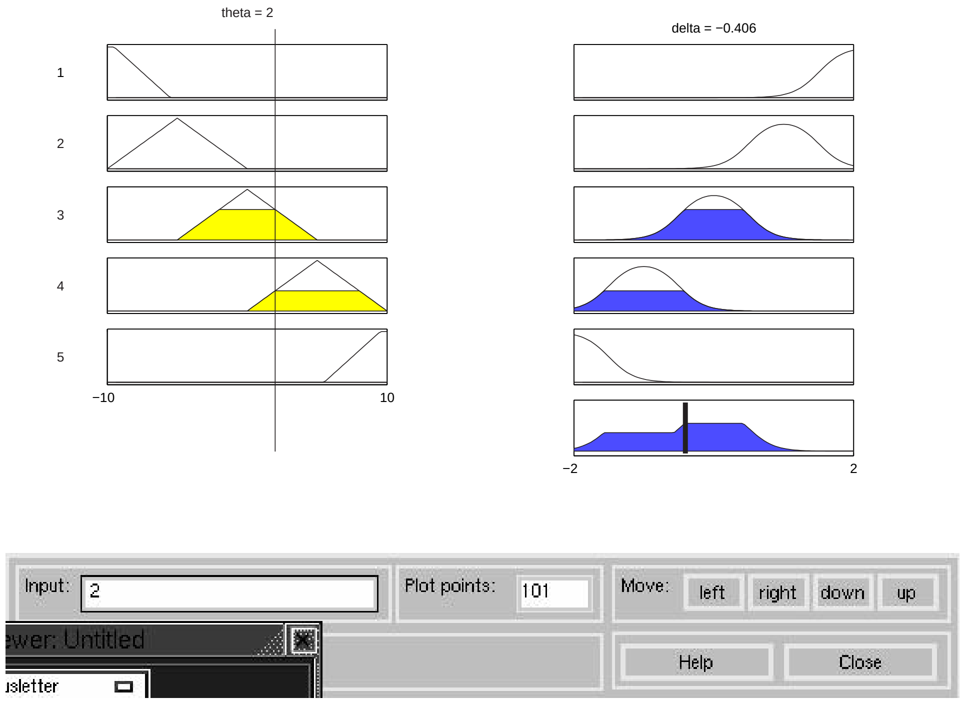

Figure 6: Output fuzzy sets for $\theta = 2$.

$\theta = 2$ corresponds to positive membership for both μ_{Z} and μ_{SP}, both membership functions are partially shaded. Note that unlike the case for $\theta = 0$, the triangles are not completely shaded. Rather, they are shaded only below the value of $\mu_{\mathrm{Z}}(2)$ and $\mu_{\mathrm{SP}}(2)$. Why is this done?

The answer to this question lies in the formula for doing generalized modus ponens:

$$\mu_{B^*}(b) = \max_{a \in A} \Big[\mu_{A^*}(a) \star \mu_A(a) \star \mu_B(b) \Big].$$

Consider the predicate Small Positive and the corresponding singleton fuzzy set specified by

$$\mu_{\mathrm{SP}^*}(\theta) = \begin{cases} 1 & \theta = 2 \\ 0 & \text{otherwise} \end{cases} .$$

Then $\max_{\theta \in [-15:15]} \Big[\mu_{\mathrm{SP}^*}(\theta) \star \mu_{\mathrm{SP}}(\theta) \star \mu_B(b) \Big]$ has to occur at $\theta = 2$. To see this, observe that $\mu_{\mathrm{SP}^*}(\theta) \star \mu_{\mathrm{SP}}(\theta) = 0$ except at $\theta = 2$. Since all membership functions are no less than zero, this means that the maximum over all θ must occur at $\theta = 2$. Thus, we can simplify $\max_{\theta \in [-15:15]} \Big[\mu_{\mathrm{SP}^*}(\theta) \star \mu_{\mathrm{SP}}(\theta) \star \mu_B(b) \Big]$ to $\mu_{\mathrm{SP}}(\theta = 2) \star \mu_B(b)$. The value, $\mu_{\mathrm{SP}}(\theta = 2)$, is illustrated by the horizontal line across the plot for μ_{SP}.

Now, unlike the case for crisp logic, the output predicate has varying degrees of truth as δ varies. How much merit do we place on concluding that δ should be used? We choose the minimum of the strength of the antecedent rule (e.g., $\mu_{\mathrm{SP}}(\theta = 2))$) and the degree to which δ is in the consequent set (e.g., $\mu_{\mathrm{SN}}(\delta)$). In words, we make an observation but conclude an entire fuzzy predicate, a function of δ, that differs from the original consequent predicate. In Figure 5 for $\theta = 0$ we conclude that

$$\begin{aligned}
\mu_{Z^*}(\delta) &= \max_{\theta \in [-15:15]} \left[\mu_{SP^*}(\theta) \star \mu_{SP}(\theta) \star \mu_{Z}(\delta) \right] \\
&= \mu_{SP}(\theta = 0) \star \mu_{Z}(\delta) \\
&= 1 \star \mu_{Z}(\delta) \\
&= \mu_{Z}(\delta).
\end{aligned}$$

In Figure 6 for $\theta = 2$ we conclude that

$$\begin{aligned}
\mu_{Z^*}(\delta) &= \max_{\theta \in [-15:15]} \left[\mu_{SP^*}(\theta) \star \mu_{SP}(\theta) \star \mu_{Z}(\delta) \right] \\
&= \mu_{SP}(\theta = 2) \star \mu_{Z}(\delta) \\
&= 0.6 \star \mu_{Z}(\delta)
\end{aligned}$$

which is illustrated on the right-hand side of Figure 6 as the shaded portion of $\mu_Z(\delta)$.

Aggregation Observe that when $\theta = 2$ two antecedants have non-zero membership functions and, consequently, two consequents can be inferred (the two shaded membership functions on the right-hand side of Figure 6. The process of combining two consequent membership functions to form a third, complete consequent is called *aggregation*. Typically, maximum or some other $\oplus$ operator is used to perform aggregation. The bottom membership function on the right-hand side of Figure 6 is the aggregation of the two consequents (SP and Z).

Deffuzification It's great to have a plot of the aggregated consequent. If I ask this function, "Given that $\theta = 2$, how much support to I have for setting δ to -1.2?" it can answer "About 0.4." Unfortunately, this does not tell me what I should set δ to. Selecting an output from a consequent membership function is called *defuzzification* because it is the process by which a fuzzy consequent is translated into a crisp decision. A standard way to defuzzify is to take the centroid (i.e., center of mass) of the output consequent membership function. Indeed, the resulting δ is shown by a solid line on the plot of the consequent membership function of Figure 6. Thus, this reasoning system observes $\theta = 2$ and concludes that I should set $\delta = -.41$.

One caution is in order to prevent you from blindly applying the centroid operator as a defuzzifier. Consider the aggregated consequent membership function shown in Figure 7. Where is the centroid of this figure? At an output value right between the two humps in the membership function. What does this mean? It means that I have evidence (from observations) that either one of two sets of outpus are justifiable, but I ignore that evidence and conclude something which has zero justification.

Testing the System Suppose that you have the job of running the program that makes sure the telescope is aligned with the star. Unfortunately, the stars come out at night, and you have fallen asleep. When you awake, you observe that the telescope is eight degrees off center. What does your fuzzy system say to do? It tells you to lower your end of the telescope by 1.18 milimeters every tenth of a second.

Let's check out some of the physics involved. The telescope is essentially a see-saw with one end shorter than the other. Suppose that the telescope is three feet long with your end being one

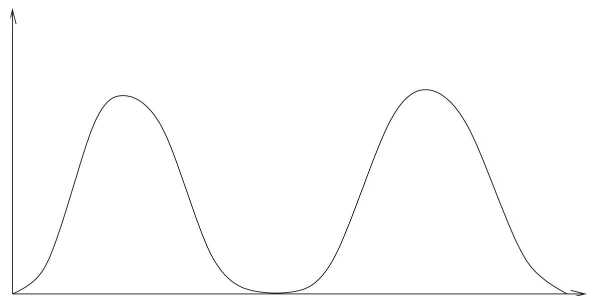

Figure 7: A consequent membership function that illustrates the problem with blindly applying a centroid defuzzifier.

foot long. The output of the fuzzy reasoning system we just constructed is called δ and it encodes the rate at which your end of the see-saw is lowered. If you lower your end at a rate of δ for ΔT seconds then the other end rises $2\delta\Delta T$; more precisely, the end of the telescope carves out an arc that is $2\delta\Delta T$ milimeters long. Since arclength is just the angle that is changed times the length of the lever, the change in angle that accompanies a $2\delta\Delta T$ milimeters long arc is

$$2\theta = 2\delta\Delta T$$

or $\Delta\theta = \delta$.

So, you read the current angle θ from one computer screen, type this into the terminal of a second computer, and this second computer spits out the δ that you should input into the first computer. You type this in, the telescope starts moving and, after a little while, a new observation θ appears. You then repeat this process. Suppose that you can do this process about ten times a second (your fuzzy system depends on this). Then, the angle of the telescope gradually approaches zero. Figure 8 shows how th nty seconds.

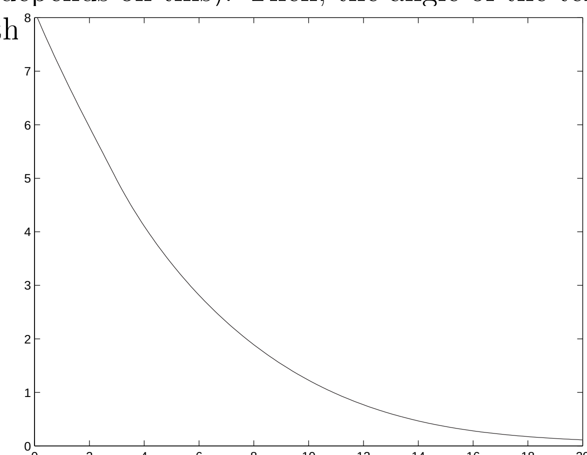

Figure 8: The history of angles as you apply your fuzzy reasoning system. The vertical axis is θ and the horizontal axis is time. Thus, as time progresses, the error of the telescope angle decreases.

One Final Note In the real world, things have momentum and they don't always respond immediately to our instructions. If the telescope was 3 meters long instead of 3 feet long, it is likely that a rapid movement by the telescope would build up so much momentum that we would overshoot our desired setting. One way to counteract this action is to control forces (instead of velocities) and let the action be a function not only of the current angle but also of the current angular velocity. This turns the system from one in which the only input is the angle to one which has two inputs: angle θ and angular velocity $\dot{\theta}$ (notice the $\cdot$ over the θ to indicate $\dot{\theta} = \frac{d\theta}{dt}$).

When a problem has two inputs, a convenient way to represent the rule base is via a fuzzy associative memory. This is just a two dimensional array with indices corresponding to the antecendants and entries corresponding to the consequents. For our telescope problem, the fuzzy associative memory might be

		θ				
		LN	SN	Z	SP	LP
$\dot{\theta}$	N	LP	SP	SP	Z	SN
	Z	SP	SP	Z	SN	SN
	P	SP	Z	SN	SN	LN

.

The cell two from the right and one down has a *SN* in it. The corresponding rule is read *if θ is SmallPositive and $\dot{\theta}$ is Zero then δ is SmallNegative.*

I made up some membership functions for $\dot{\theta}$ and created a corresponding rulebase. For $\theta = 0$ and $\dot{\theta} = 0$ the corresponding reasoning system can be visualized as in Figure 9.

Figure 10 shows the reasoning process for $\theta = 2$ and $\dot{\theta} = -1$.

References

[1] J. M. Mendel. Fuzzy logic systems for engineering: A tutorial. *Proceedings of the IEEE*, 83(3):345–377, March 1995.

[2] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach.* Prentice Hall, 1995.

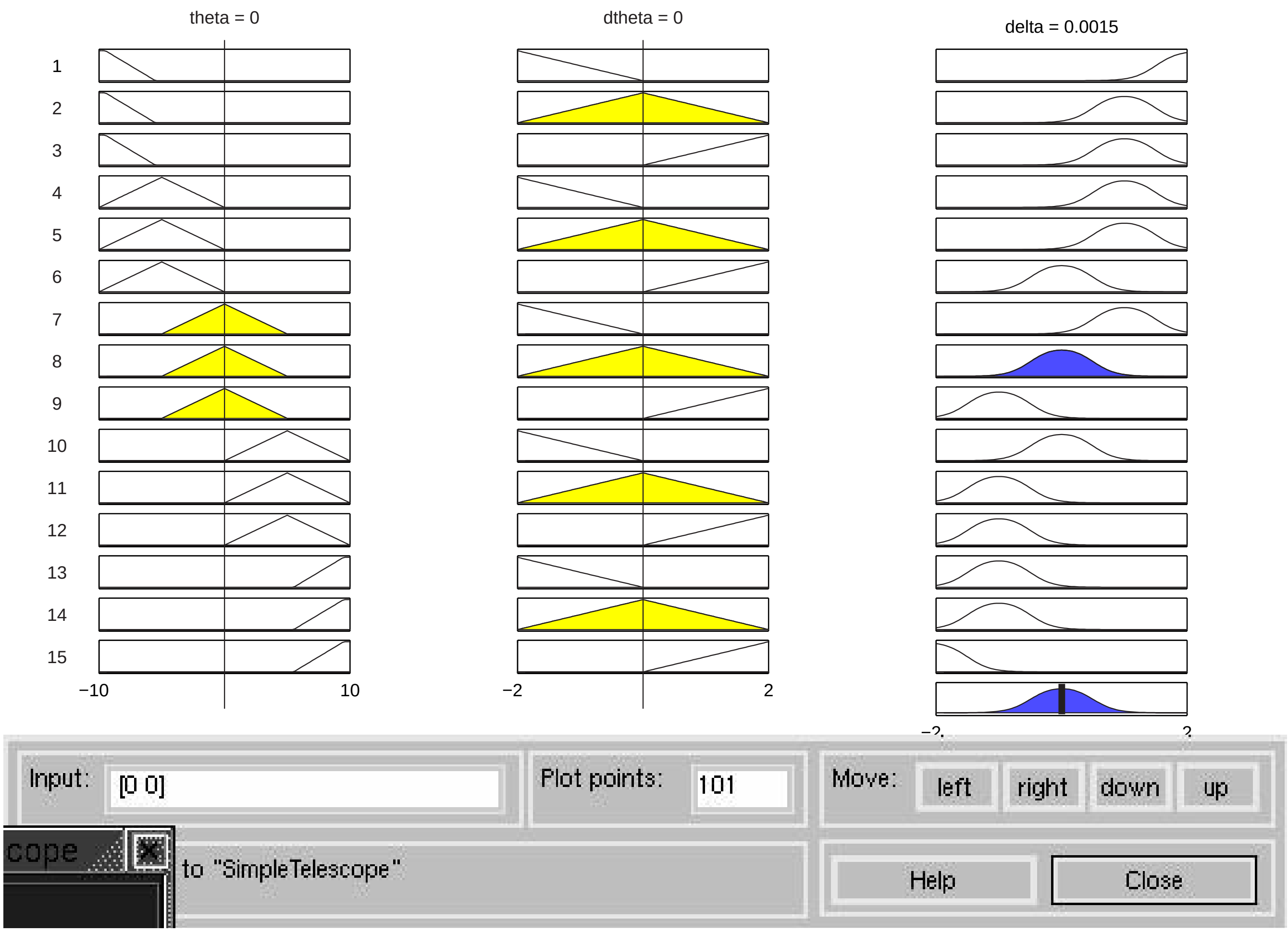

Figure 9: Output fuzzy sets for $\theta = 0$ and $\dot{\theta} = 0$. The top three figures on the left-hand side correspond to θ is LargeNegative. The top figure in the middle column corresponds to $\dot{\theta}$ is Negative, the second figurein the middle column corresponds to $\dot{\theta}$ is Zero, and so on.

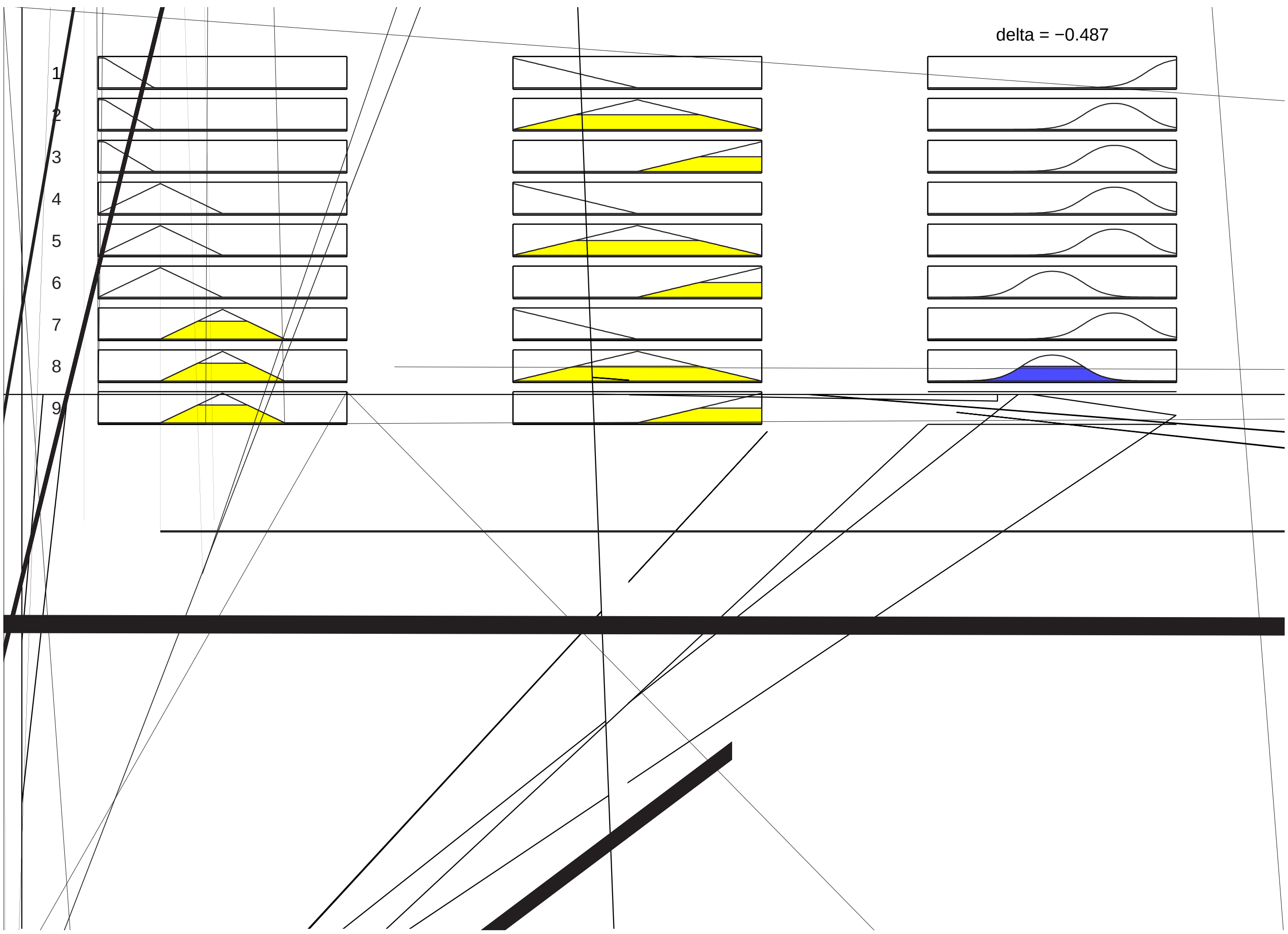

Figure 10: Output fuzzy sets for $\theta = 2$ and $\dot{\theta} = -1$.