

Fuzzy Logic

Part 1 – fuzzy essentials

By Owen Bishop

OandA.Bishop@bigpond.com

In this two-part article Owen Bishop looks at designing fuzzy control systems with the aid of software.

What is your size in shirts? There are two ways of answering this. Many shirtmakers quote one of several dimensions such as size of collar or circumference of chest. Run a tape measure around your neck, read off the size in centimetres, and select a shirt of the appropriate size. Wearer's necks are allocated to different size categories, often at 2 cm intervals. Thus, the category '41 cm' may refer to necks that are 40 cm or more in circumference but less than 42 cm. Note that the sizes of customer's necks, not the sizes of the necks of the shirts. The customers in a store could be measured and sorted out into groups on this basis. The manufacturers classify the **customers**, then make shirts intended to fit them. Each group of customers can be sold shirts that fit their necks with sufficient degree of comfort. Whether the shirts also fit the customers on the chest or waist in another matter! Putting this in logical terms, customers are classified into **sets** based on neck circumference.

For a given set (say, the '41 cm' set), a customer is either a member of that set or is not a member. This 'is'/'is not' feature is a **binary** one, a characteristic of Boolean logic. Because the neck size system leaves no room for doubt at the boundaries between one set and another, we say that its sets are **crisp** sets.

Fuzzy sets

Another way of defining size categories is by verbal description. The popular system has a range of values such as S, M, L, XL, and XXL.

A shirt marked 'L', for example, is intended to fit a 'typical' large man. Occasional shirt-makers produce shirts for really huge men in

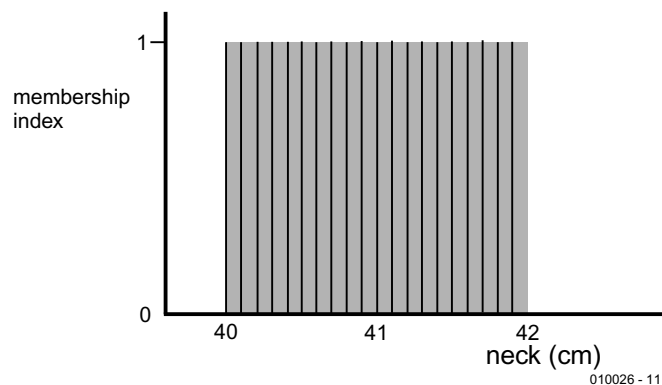


Figure 1. In a crisp (Boolean, binary) set all members have a membership index of 1.

sizes from 3XL up to 8XL. At the other end of the scale, there may be an XS size, and sometimes SM comes between S and M. In words, the most common categories are small, medium, large, extra-large, and extra-extra-large.

The essential feature of this system is that a customer may be classed as a member of more than one set. The customer may find that shirts of two adjacent categories fit reasonably well. For example, the author falls into both the L and the XL set and, when he buys a shirt, he usually takes an L shirt and an XL shirt along to the fitting room and buys the one that fits better. Much depends on the maker, the fabric and the style of the shirt and there are no clear-cut boundaries between the

sets. These are **fuzzy sets**. In general, fuzzy sets correspond to real-life situations and are a more satisfactory and practicable way of characterising shirt sizes. Humans come in a wide variety of shapes and sizes and it can not be expected for everyone to slot neatly into the crisp sets based on neck circumference.

Set membership

Figure 1 shows the **membership function** of the '41 cm' crisp set. Each of the various sizes of human neck included in this set are indicated by vertical lines spaced, for convenience, at 1-mm intervals. Neck sizes vary continuously over the range, to the actual membership function is the whole of the shaded

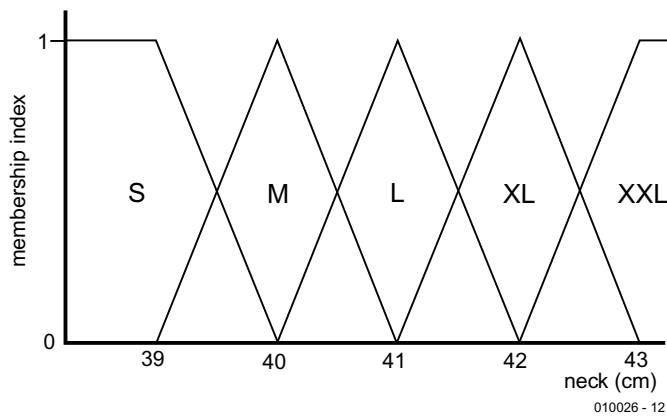


Figure 2. Fuzzy sets such as shirt sizes each cover a range of neck sizes.

area. Each of the sizes within the set has a **membership degree**, expressing the extent to which that size is considered to be a member of that set. In a crisp set, every member of the set has a membership degree of 1. They all take collars of one particular size, possibly made 42.5 cm in circumference to provide a comfortable fit and to allow for shrinking. The necks all belong fully to the '41-cm' set and to no other set. Conversely, all necks that do not belong to the set have a membership degree of zero. The 0 or 1 situation reflects the binary (two-valued) nature of a crisp set.

Figure 2 shows typical membership functions of fuzzy sets. Only necks toward the average of the set have membership degrees in the region of 1. Other members belong to the set with degrees between 0 and 1. Membership is not binary as in a

crisp set, but is **multi-valued**. The horizontal scale may be collar size as in Figure 1, but there is now a bigger spread of neck sizes in each set. In addition, a person with a neck of a given size may be a member of two (or even more) adjacent sets.

Because of their shape, the sets illustrated in Figure 2 are referred to as **triangular sets**. There may also be **trapezoidal sets**, in which members in the central region of the range all have a membership degree of 1. Trapezoidal sets are seen at the limits of the size ranges. In certain applications, there may be sets with Gaussian or other functions.

Linguistic variables

A very important feature of fuzzy logic is that it does not have to be quantitative. Sets are defined with fuzzy boundaries where the mem-

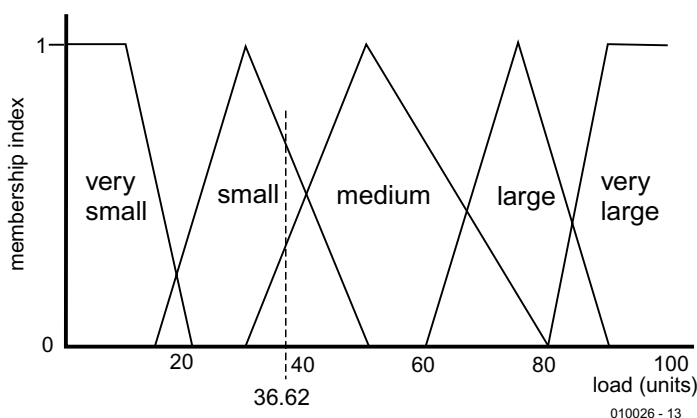


Figure 3. A washing machine load rated at 36.62 units is in both the 'small' and 'medium' fuzzy sets.

bers have low membership degrees and therefore do not contribute fully to the set. The exact cut-off point is not particularly important. Instead of defining a set in numerical terms, we describe it by using one or two appropriate words or abbreviations. We call these **linguistic variables**. The size variables S, M, L, XL, and XXL are examples. In the case of a dishwasher, the size of the load could be described by variables such as: 'very small', 'small', 'medium', 'large', and 'full'.

Fuzzy controller

Taking the example of a dishwasher in more detail, we will show how to build a fuzzy logic controller for a hypothetical dishwasher, which uses fuzzy logic to control the wash cycle. The input to this system is the size of the load, which may be categorised under the five linguistic variables listed in the previous paragraph. **Figure 3** defines the five fuzzy input sets on an arbitrary scale from 0 (empty) to 100 (full). The size of the load could be measured electronically by load cells on the wire baskets. The output variable is the wash time, which might range from 5 minutes to 40 minutes. **Figure 4** shows five fuzzy output sets and variable names. Note that the number of output sets does not have to be equal to the number of input sets.

Like any logic system, a fuzzy logic program processes a set of logical rules. The rules for this dishwasher might be:

1. IF load is 'very small',
THEN washtime is 'minimum'.
2. IF load is 'small',
THEN washtime is 'reduced'.
3. IF load is 'medium',
THEN washtime is 'medium'.
4. IF load is 'large',
THEN washtime is 'extended'.
5. IF load is 'full',
THEN washtime is 'maximum'.

These rules have the same format as those in crisp logic but there is an essential difference. In crisp logic, the rules are applied in sequence. As each rule is processed, the computer moves on to the next line the program. The rules are processed one at a time, in the order in which they appear in the program. In fuzzy logic, the rules are applied in parallel. A fuzzy logic processor chip is specially designed for parallel processing of the rules. However, this does not prevent us writing fuzzy logic programs for ordinary com-

puters that process rules sequentially. The results of applying each rule in turn can be kept on hold, until all are processed. This is not as efficient as parallel processing but makes it possible to write useful fuzzy programs for PCs and other computers as described below.

Fuzzy processing

We take the example of the fuzzy dishwasher. The first stage in operating the system is to measure the size of the load electronically. We might measure weight or volume, or perhaps both and combine the two by using a formula. Whatever kind of measurement we make, it will have a definite value. It will be **crisp**, even if it is measured on an arbitrary scale. For example, the load size may be 36.62, measured on an arbitrary scale from 0 to 100.

The next stage in processing is **fuzzification**. In **Figure 3**, the input value 36.62 is seen to belong to two input sets, 'small' and 'medium'. When the rules are applied, the conditions of both Rule 2 and Rule 3 apply. We say that rules 2 and 3 are **fired**. Rules 1, 4 and 5 are not fired.

Fuzzy inference

Next comes the stage known as **fuzzy inference**, estimating the result of the firings. Note that the two rules are not fired to the same extent. They have different **degrees of applicability**. The value 36.62 has a membership index of approximately 0.67 in the 'small' set, and an index of 0.33 in the 'medium' set. The corresponding outputs have correspondingly different degrees of applicability. Output 'reduced' has a DOA of 0.67 and output 'medium' has a DOA of 0.33. We need to combine the two outputs in some way to give a single output value.

Defuzzifying

There are various ways of obtaining crisp output, of which the following was developed by Mamdani and is one of the most popular. Its mathematical basis is beyond the scope of this description but, fortunately, the computer does the mathematics. We will look at some software in Part 2 of this article.

The first step is to truncate the two membership functions at levels equal to the two membership indices (**Figure 5**). The functions are not only clipped but also scaled, as the figure shows. The two membership functions are now regarded as a single function. Usually the area of overlap is ignored. Finally, we produce a single crisp value to represent this

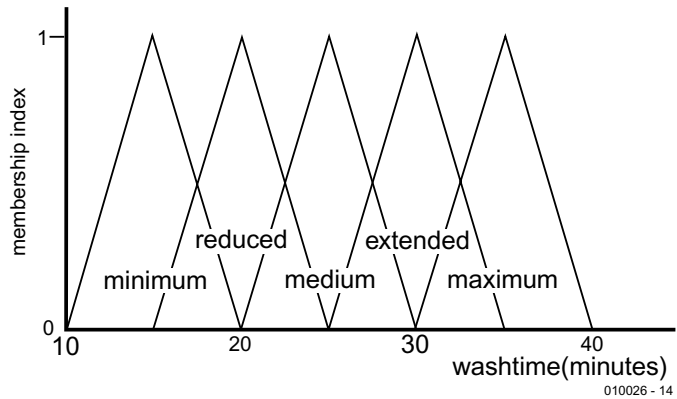


Figure 4. Washtimes are classified into five fuzzy sets covering washtimes from 10 to 40 minutes.

combined function. There are several ways of doing this. The most favoured way is known as the 'centre of area' or 'centre of gravity' technique. This produces a weighted mean of the wash times in proportion to their membership indices. Imagine the combined shape to be cut from thin card, then balanced on a pencil that lies parallel to the vertical (index) axis. The card balances when the pencil is placed under the card at 21.6550 on the horizontal (time) axis.

There are other ways of defuzzifying, which give better results in certain types of application, and are easier to calculate. For example, where two areas do not overlap we may take the 'centre of the largest area' as the defuzzified value.

Summing up: the action of a fuzzy controller (as realised on a fuzzy

computer or on a PC) is:

1. Crisp input
2. Fuzzification — fire all rules
3. Fuzzy inference — combining the results from those rules that fire.
4. Defuzzification.
5. Crisp output.

In Part 2 we'll look at the software and carry the development of the fuzzy controller a stage further.

(010026-1)

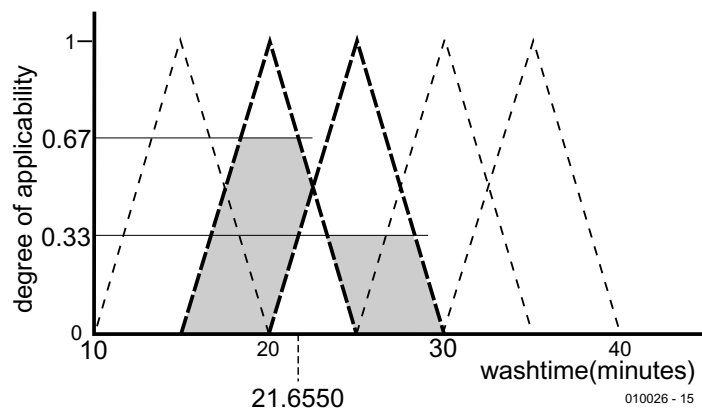


Figure 5. If Rules 2 and 3 fire with applicability 0.67 and 0.33, the resulting crisp output is the centre of the shaded areas, at 21.6550.