Analysis and Modelling of Hierarchical Fuzzy Logic Systems

Masoud Mohammadian
University of Canberra, Australia

INTRODUCTION

Computational intelligence techniques such as neural networks, fuzzy logic, and evolutionary algorithms have been applied successfully in the place of the complex mathematical systems (Cox, 1993; Kosko, 1992). Neural networks and fuzzy logic are active research area (Cox, 1993; Kosko, 1992; Lee, 1990; Mohammadian & Stonier, 1995; Welstead, 1994; Zadeh, 1965). It has been found useful when the process is either difficult to predict or difficult to model by conventional methods. Neural network modelling has numerous practical applications in control, prediction, and inference.

Time series (Ruelle, 1998) are a special form of data where past values in the series may influence future values, based on presence of some underlying deterministic forces. Predictive model use trends cycles in the time series data to make prediction about the future trends in the time series. Predictive models attempt to recognise patterns and trends. Application of liner models to time series found to be inaccurate, and there has been a great interest in nonlinear modelling techniques.

Recently, techniques from computational intelligence fields have been successfully used in the place of the complex mathematical systems for forecasting of time series. These new techniques are capable of responding quickly and efficiently to the uncertainty and ambiguity of the system.

Fuzzy logic and neural network systems (Welstead, 1994) can be trained in an adaptive manner to map past and future values of a time series and thereby, extract hidden structure and relationships governing the data. The systems have been successfully used in the place of the complex mathematical systems, and have numerous practical applications in control, prediction, and inference. They have been found useful when the system is either difficult to predict and/or difficult to model by conventional methods. Fuzzy set theory provides a means for representing uncertainties. The underlying power of fuzzy logic is its ability to represent imprecise values in an understandable form. The majority of fuzzy logic systems, to date, have been static and based upon knowledge derived from imprecise heuristic knowledge of experienced operators, and where applicable, also upon physical laws that governs the dynamics of the process.

Although its application to industrial problems has often produced results superior to classical control, the design procedures are limited by the heuristic rules of the system. It is simply assumed that the rules for the system are readily available or can be obtained. This implicit assumption limits the application of fuzzy logic to the cases of the system with a few parameters. The number of parameters of a system could be large.

Although the the number of fuzzy rules of a system is directly dependant on these parameters. As the number of parameters increase, the number of fuzzy rules of the system grows exponentially.

In fuzzy logic systems, there is a direct relationship between the number of fuzzy sets of input parameters of the system and the size of the fuzzy knowledge base (FKB). Kosko (1992) call this the “Curse of Dimensionallity.” The “curse” in this instance is that there is exponential growth in the size of the fuzzy knowledge base (FKB), where k is the number of rules in the FKB, m is the number of fuzzy sets for each input and n is the number of inputs into the fuzzy system.

As the number of fuzzy sets of input parameters increase, the number of rules increases exponentially. There are a number of ways that this exponential growth in the size of the FKB can be contained. The most obvious is to limit the number of inputs that the system is using. However, this may reduce the accuracy of the system, and in many cases, render the system being modelled unusable. Another approach is to reduce the number of fuzzy sets that each input has. Again, this may reduce the accuracy of the system. The number of rules in the FKB can also be trimmed if it is known that some rules are never used. This can be a time-consuming and tedious task, as every rule in the FKB may need to be looked at.

Raju and Zhou (1993), Mohammadian and Kingham (1997), and Mohammadian, Kingham, and Bignall (1998) suggested using a hierarchical fuzzy logic structure for such fuzzy logic systems to overcome this problem. By using hierarchical fuzzy logic systems, the number of fuzzy rules in the system are reduced, thereby, reducing the computational time while maintaining the systems robustness and efficiency. In this chapter, the design and development of a hierarchical fuzzy logic systems using genetic algorithms to model and predict interest rate in Australia is considered. Genetic algorithms are employed as an adaptive method for design and development of hierarchical fuzzy logic systems.

HIERARCHICAL FUZZY LOGIC SYSTEMS

The hierarchical fuzzy logic structure is formed by having the most influential inputs as the system variables in the first level of the hierarchy, the next important inputs in the second layer, and so on. If the hierarchical fuzzy logic structure contains n system input parameters and L number of hierarchical levels with n_i the number of variables contained in the ith level, the total number of rules k is then determined by:

$$k = \sum_{i-1}^{L} m^{n_i} \tag{1}$$

where m is the number of fuzzy sets. This equation means that by using a hierarchical fuzzy logic structure, the number of fuzzy rules for the system is reduced to a linear function of the number of system variables n, instead of an exponential function of n as is the conventional case. The first level of the hierarchy gives an approximate output, which is then modified by the second level rule set, and so on. This is repeated for all succeeding levels of the hierarchy. One problem occurs when it is not known which inputs to the system have more influence than the others. This is the case in many problems. In some case, statistical analysis could be performed on the inputs to determine which ones have more bearing on the system.

INTEGRATED HIERARCHICAL FUZZY LOGIC AND GENETIC ALGORITHMS

Genetic algorithms (GAs) (Goldberg, 1989; Goonatilake, Campbell, & Ahmad, 1995) are powerful search algorithms based on the mechanism of natural selection, and use operations of reproduction, crossover, and mutation on a population of strings. A set (population) of possible solutions, in this case, a coding of the fuzzy rules of a fuzzy logic system, represented as a string of numbers. New strings are produced every generation by the repetition of a two-step cycle. First, each individual string is decoded and its ability to solve the problem is assessed. Each string is assigned a fitness value, depending on how well it performed. In the second stage, the fittest strings are preferentially chosen for recombination to form the next generation. Recombination involves the selection of two strings, the choice of a crossover point in the string, and the switching of the segments to the right of this point, between the two strings (the cross-over operation). Figure 1 shows the combination of fuzzy logic and genetic algorithms for generating fuzzy rules.

For encoding and decoding of the fuzzy rule for a fuzzy logic system, first the input parameters of the fuzzy logic system is divided into fuzzy sets. Assume that the fuzzy logic system has two inputs α and β and a single output δ. Assume also that the inputs and output of the system is divided into five fuzzy sets. Therefore, a maximum of 25 fuzzy rules can be written for the fuzzy logic system.

The consequence for each fuzzy rule is determined by genetic evolution. In order to do so, the output fuzzy sets are encoded. It is not necessary to encode the input fuzzy sets because the input fuzzy sets are static and do not change.

The fuzzy rules relating the input variables (α and β) to the output variable (δ) have 25 possible combinations. The consequent of each fuzzy rule can be any one of the five output fuzzy sets. Assume that the output fuzzy sets are **NB** (Negative Big), **NS** (Negative Small), **ZE** (Zero), **PS** (Positive Small), and **PB** (Positive Big). Then the output fuzzy sets are encoded by assigning 1 = **NB** (Negative Big), 2 = **NS** (Negative Small), 3 = **ZE** (Zero), 4 = **PS** (Positive Small),

Figure 1. Combination of fuzzy logic and genetic algorithms for fuzzy rule generation