Hierarchical Fuzzy Rule Based Systems using an Information Theoretic Approach

ANTONY WALDOCK [1], BRIAN CARSE [2], CHRIS MELHUISH [2]

(1) Advanced Technology Centre, BAE SYSTEMS, Bristol, BS34 7QW, UK

(2) Intelligent Autonomous Systems Laboratory, University of the West of England, Bristol, BS16 1QY, UK

antony.waldock@baesystems.com
brian.carse@uwe.ac.uk
chris.Melhuish@uwe.ac.uk

Abstract

This paper proposes a novel anytime algorithm for the construction of a Hierarchical Fuzzy Rule Based System (HFRBS) using an information theoretic approach to specialise rules that do not effectively model the decision space. The amount of uncertainty tolerated within the decision provides a single tuneable parameter to control the trade off between accuracy and interpretability. The algorithm is empirically compared with existing methods of function approximation and is demonstrated on a mobile robot application in simulation.

Keywords

Information Theory, Hierarchical Fuzzy Rule Based Systems, Mobile Robot

1. Introduction

The construction of a mobile robot controller is a complex and time-consuming process. Classical Control Theory [1] allows the generation of control rules from mathematical models of the environment, platform and control laws. These models can be used to interpret the controller operation and provide stability and completeness analysis. Unfortunately the generation of models that accurately represent the dynamics of a complex environment can be a long and difficult, if not impossible, process. Soft Computing [2] provides a collection of methods that aim to exploit the tolerance for imprecision and approximation to achieve robustness and low cost solutions e.g. Neural Networks and Fuzzy Logic. Fuzzy Rule Based Systems have been demonstrated to cope well with uncertain and imprecise environments when used for mobile robot controllers [3]. The current

contribution proposes a new method for the construction a Hierarchical Fuzzy Rule Based System were the amount of uncertainty tolerated within a decision can be used to control the trade off between accuracy and interpretability.

Section 2 defines a Fuzzy Rule Based System and outlines existing solutions to the problem of generating accurate and interpretable function approximators. Information theoretic specialisation and the proposed algorithm are explained in sections 3 and 4 with comparative experimental results in section 5. The algorithm is demonstrated on a simulated mobile robot application in section 6 followed by conclusions and future work in section 7.

2. Fuzzy Rule Based Systems

Using Fuzzy Rule Based Systems (FRBS) to build mobile robot controllers [4, 5], has become increasingly popular due to their ability to automatically generate rules from large data sets while maintaining a degree of human interpretability [6]. A FRBS partitions the input space into a number of linguistic symbols, each associating a fuzzy set with a natural language meaning [7] e.g. small, medium, very large etc. A fuzzy set links an input variable with a membership function, to represent its applicability with regard to the current environmental state, and hence, determine its influence in the decision process. The decision space is partitioned using a series of IF...THEN... rules. The combination of meaningful linguistic symbols and well-structured rules provides a high degree of human interpretability. For example, a mobile robot that is required to avoid collisions might use a fuzzy rule such as:

IF (*wall is close*) AND (*wall_direction is infront-left*) THEN (*turn right*)

The way in which linguistic symbols are represented can directly affect the accuracy of the approximated function [8]. If a greater degree of accuracy is demanded then the structure (i.e. size, shape and position) of the membership functions which represent the linguistic symbols must be altered. Increasing the granularity of the linguistic symbols can facilitate improvements in accuracy, but as the complexity of a system increases, our ability to make precise and yet significant statements about its behaviour diminishes. Hence within this

contribution, the number of rules is used as a measure of interpretability. As the number of linguistic symbols increases human interpretability declines. This is referred to as the trade off between accuracy and interpretability [9].

The linguistic symbols in a FRBS facilitate the modelling of complex functions whilst retaining a degree of human interpretability. However, a model clearly interpretable by humans often leads to a reduction in accuracy. This reduction stems from the constrained structure of the linguistic symbols considered [8]. In the last few years, a new variant of FRBS has been developed in an attempt to improve the accuracy of the model and is referred to as an Approximate FRBS [10]. An approximate FRBS improves accuracy by working directly with the fuzzy sets within the rules instead of the linguistic symbols. By allowing the fuzzy variables to vary in number, size and position, they can be mapped directly to the data giving significant improvements [9, 11]. At the end of rule generation, the fuzzy variables are semantically free and tend to be excessively specific. This limits the use of an approximate FRBS as an interpretable mobile robot controller. However it must be noted that they remain more interpretable than models like neural networks.

Another possible method for improving accuracy is called a Hierarchical FRBS [12]. A HFRBS divides the input space into a fixed number of linguistic symbols each corresponding to a natural language meaning e.g. very small, small, large etc. Training data is used to automatically generate rules as in a standard FRBS. The HFRBS then employs an *expansion policy* to determine inaccurate areas of the decision space and the corresponding rules. When an inaccurate area is identified, the rule representing that portion of the decision space is specialised into a set of smaller specific rules. This process of specialisation continues until a desired level of accuracy is satisfied. Figure 1 shows an example of a partitioned decision space (i) and its corresponding hierarchical representation (ii). The concept of increasing the granularity to fit the underlying decision space has been applied to classifier systems for a number of years [13].

Figure 1

This type of linguistic symbol expansion increases the accuracy depending on the complexity of the data modelled and is commonly referred to as "ad-hoc data driven learning" [9]. The expansion can be controlled depending on the accuracy desired, human interpretability required and the complexity of the function to be approximated. Obviously some of these are mutually exclusive, for example, high human interpretability on a complex, accurate model may be impossible. Different methods of determining when expansion should occur are discussed below.

Holve [14] outlines a method for specialisation by carefully pre-processing training data such that, when a conflict is encountered, the linguistic symbol is expanded. Although demonstrated to approximate complex functions, the pre-processing limits its use to applications where all the training data is available and certain. Cordón, Herrera and Zwir [12] outline a HFRBS that uses expansion techniques to specialise linguistic symbols with a large degree of error. The error of each rule is calculated by the percentage of the Mean Square Error (MSE) associated with each rule over the MSE of the entire training set. A rule with bad performance is determined by comparing this error with a tuneable parameter α, which dictates the rate of expansion and hence the accuracy of the function approximated. This method relies on the complete training set and test set being available during the expansion phase. For an application, such as learning behaviour in a mobile robot, access to the entire training set is not possible thereby requiring an Anytime Algorithm [15]. Anytime is defined by Grefenstette and Ramsey as an algorithm that can be suspended and resumed without negligible overhead, terminated at anytime and provide an improvement over time. The next section outlines the proposed new anytime expansion policy based on Information Theory.

3. Information Theoretic Specialisation

The rule generation algorithm proposed in the current contribution uses a similar method of hierarchical specialisation, but the expansion rate is determined by the amount of uncertainty experienced by the rule. Information Theory, developed by Shannon [16], was initially concerned with modelling the efficiency of communication systems but has been applied to a multitude of other research areas including decision making (ID3 [17]) and fuzzy logic systems [18]. Shannon

demonstrates that the quality of information is defined as the amount of information conveyed in an event and depends on the probability of that event [16].

Suppose we have a set of n possible events whose probabilities of occurrence are $p_1,p_2...p_n$. These probabilities are known but that is all we know concerning the event. Shannon defines the amount of 'choice' involved or the uncertainty of the outcome as Entropy. The entropy, $H(p_1,p_2...p_n)$ of an event with n possible outcomes is defined by Shannon as

$$H(p_1, p_2..p_n) = \sum_{1}^{n}(p_i \times \log(p_i))$$

Equation 1 - Shannon Entropy

Entropy can be used to calculate the amount of uncertainty experienced by each rule in the FRBS [19]. If the FRBS is viewed as a Fuzzy Associative Memory (FAM) where the linguistic symbols M_i, for each input n, produce an n-dimensional decision space partitioned by an i-dimensional grid. Each cell can represent an IF-THEN rule with n linguistic inputs corresponding to a single output linguistic symbol [14]. A FAM is the ideal representation for a mobile robot controller because it offers completeness and can be implemented as a lookup table providing constant time access.

The proposed Information Theoretic Hierarchical Fuzzy Associative Memory (IT-HFAM) extends the traditional FAM approach by including an applicability distribution over all the output linguistic symbols. Figure 2 displays a FAM partitioned into four cells (rules), each with an applicability distribution over all possible output symbols.

Figure 2

Using Shannon's measure of uncertainty on the output applicability distribution gives the amount of choice (uncertainty) experienced by this cell. If the applicability distribution is flat i.e. A_i are equal, (Figure 2: Bottom Left Cell) then

the uncertainty of the cell is at a maximum. Using the uncertainty measure for each rule, the overall decision uncertainty can be calculated for a FRBS inference [18]. The decision entropy is found by multiplying the entropy of each active rule, $H(R_i)$, by the corresponding rule activation level, $activation(R_i)$ [19].

$$Decision\ Entropy = \sum_{i=1}^{n} (activation(R_i) \times H(R_i))$$

Equation 2 - Decision Entropy

This weights the uncertainty of each rule by the amount of influence within the inference. To control the amount of uncertainty within an inference, a single rule must not exceed the maximum uncertainty desired (E_{max}). The maximum uncertainty desired is used to identify rules that cannot effectively model the decision space with the desired uncertainty at this granularity. In order to model this area of the decision space more effectively it is necessary to specialise the cell into a number of smaller cells. Hence, the uncertainty tolerated within a decision can be used as an expansion policy. A detailed explanation of how this process of rule specialisation occurs is outlined in the following section.

4. IT-HFAM Algorithm

This section outlines how the Information Theoretic Hierarchical Fuzzy Associative Memory (IT-HFAM) is initialised, evaluated and trained. A method of compacting the hierarchical structure to produce a concise number of rules is then outlined.

Initialisation

The IT-HFAM algorithm can be initialised with and without prior knowledge of partitions in the decision space. When no prior knowledge is known about the decision space, the algorithm is initialised with a single linguistic symbol covering each input. Where prior knowledge of the problem is known, the algorithm can be initialised with known linguistic symbols. These initial symbols direct the algorithms specialisation of the decision space and are aimed at boosting the learning process. Obviously if the initial symbols are wrong, this could provide the opposite effect and delay the specialisation process.

Evaluation

The evaluation of an IT-HFAM is the same as for a traditional FRBS [20] except that inference is performed over all levels in the hierarchical rule base (even cells that have been specialised). Evaluation for a FRBS is performed in three stages: fuzzification, inference and defuzzification. The purpose of the fuzzification and inference stages is to calculate the activation level for every cell in the HFAM. This is performed in a recursive manner because of the hierarchical nature of the FAM. Each cell calculates its current activation level and the fuzzification of any specialised cells below. The activation level is calculated using a standard MAX-MIN inference strategy. The recursive process is demonstrated in Figure 3.

Figure 3

If a cell is not specialised (a leaf cell) and the activation level is greater than zero, it will be referred to as an *Active Cell*. The defuzzification process arbitrates over all currently active cells to produce a single output value. To facilitate defuzzification, all the active cells must be extracted from the hierarchical FAM. This is performed in a similar recursive manner to fuzzification and is detailed in Figure 4.

Figure 4

Once all active cells have been recovered from the FAM, a standard defuzzification strategy can be performed (*Centre of Sums* is used in this contribution). A detailed explanation on standard fuzzification and defuzzification methods can be found in [21].

Training

Training of an IT-HFAM consists of updating the applicability distribution when exposed to a new training pattern. A training set *T*, consists of a number of training patterns, T_p that consist of a *(n+1)* dimensional vector, $T_p = \langle x_1, x_2, x_3 ..... x_n, y \rangle$ where *n* = number of inputs; *x* = the current input values

and *y* the target value. The algorithm is trained on all patterns until the training set has been exhausted. For a mobile robot controller, a training pattern can arrive from the supervisor at any time and the end of the training set is never reached i.e. training is a continuous and anytime process.

Training is performed by first identifying all of the active cells when presented with the inputs ($x_1..x_n$) from the training pattern. This can be accomplished using the recursive fuzzification and getActiveCells procedures outlined in the previous section. The applicability distribution for each active cell, C_i, is updated depending on the target value *y*. For each output membership function, M_j^y, the applicability of the output symbol $a(M_j^y)$ is increased by the level of activation for the target value *y*. For example in Figure 5, the activation of membership function one and two is used to update the applicability distribution of an active cell.

Figure 5

Centre of Sums (Equation 3) is used on the applicability distribution to calculate the current output symbol for this cell in the FAM. As the rule can only have a single output, the round function is used to determine the nearest linguistic output symbol from the Centre of Sums equation.

$$output\ symbol = round\left(\frac{\sum_{j=1}^{n}\left(j \times a\left(M_j^y\right)\right)}{\sum_{j=1}^{n} a\left(M_j^y\right)}\right)$$

Equation 3 – Output Symbol calculation

After each training pattern, the entropy (uncertainty) of each cell is compared with the tuneable parameter $E_{\max}$, which represents the maximum amount of uncertainty tolerated within each cell. If the uncertainty of the cell exceeds $E_{\max}$ and the applicability distribution covers more than two consecutive output symbols then the cell is divided into four smaller specialised cells (assuming the

FAM consists of only two dimensions) as in Figure 6. The cell expansion is halted when only two consecutive output symbols are active in the applicability distribution.

Figure 6

Figure 6 shows the membership functions of a cell dividing into two smaller functions covering the same total width. The membership functions are then used to create four smaller cells covering the same decision space.

Hierarchical FAM Compaction

The structure of the hierarchical cells formed by the IT-FAM algorithm is determined by the order of the training patterns. Reorganisation of this structure could reduce the number of cells used within the HFAM and hence, model the training set using a smaller number of rules. The compaction process could be performed at various stages during construction or in a batch process before visualisation.

Figure 7

Figure 7 displays a HFAM after exposure to a training set with each colour representing the current output symbol used by the cell. The IT-HFAM algorithm repeatedly expands a single cell in an attempt to model the decision space with the specified uncertainty. Merging cells with the same output symbol together and rebalancing the structure can be used to minimise the number of cells used. The results after the compaction process are shown in Figure 8.

Figure 8

The compaction has reduced the number of cells in Figure 8 from 13 to 5 by merging neighbouring leaf cells with a common output symbol together. The compaction process can be applied to any level of the hierarchical structure

although if performed on the top-level cells (as in example Figure 8) any linguistic meaning is lost.

The IT-FAM algorithm proposed uses the maximum amount of uncertainty tolerated in an inference as an expansion policy in a HFRBS. Experiments exploring how this expansion policy impacts on key characteristics like accuracy are outlined in the following section.

5. Fuzzy Modelling of an Intermediate Complexity Function

This section describes the performance of the IT-HFAM algorithm, on an intermediate complexity function, compared with existing function approximators. Within the experiment below, the accuracy is determined using Root Mean Squared Error (RMSE) (Equation 4) over a test set of 100 evenly distributed points over the input space.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}((T_i - R_i)^2)}{n}}$$

Equation 4 - Root Mean Squared Error

Cordón, Herrera and Zwir [22] demonstrate the ability to train a HFRBS using a Mean Squared Error (MSE) expansion policy on an intermediate complexity function (Equation 5, Figure 9i).

$$f(x_1, x_2) = e^{x_1} \times \sin^2(x_2) + e^{x_2} \times \sin^2(x_1)$$

$$x_1, x_2 \in [-8,8],\ f(x_1, x_2) \in [0,5836]$$

Equation 5 - Intermediate Complexity Function

Although Cordón et al. produce accurate approximations, the training method dictates that the entire training set be available during rule generation which is undesirable for any anytime algorithm. The IT-HFAM algorithm proposed here

does not aim to improve on the accuracy of their results but demonstrate that an anytime solution based on information theory can achieve comparative results.

The algorithm was initialised in two ways: one with seven evenly distributed membership functions (IT-HFAM7) and the other with two user defined membership functions (IT-HFAM2) over the input space. IT-HFAM2 was initialised with 4 rules arranged such that a single rule covered the flat area of the decision space and 3 covering the raised areas on the edges (Figure 9i). Defining a set of initial input linguistic symbols allows the user to direct the function approximation using any prior knowledge to boost the learning process. These initial symbols are merely estimates to direct the algorithm because if the linguistic symbols were known at design time no expansion would be necessary.

The system was trained using a training set of 1156 training patterns, which were evenly distributed over the function, compared to Cordón's 1089 training set. The RMSE was calculated after exposure to the complete training set and then after a second iteration. Both had seven evenly distributed membership functions over the output. The results of the IT-HFAM algorithm using three different values for E_{max} are compared in Table 1 with a comparison to existing function approximators in Table 2. For each experiment, the number of cells after training (C_t), the number of cells after compaction (C_r) and the final accuracy on the test set before ($RMSE_t$) and after compaction ($RMSE_r$) are displayed.

Table 1

Table 2

The function approximated by the IT-HFAM7 – 45% algorithm is displayed in Figure 9ii on a normalised axis. Table 2 demonstrates that the IT-HFAM algorithm can achieve comparative accuracy with traditional non-hierarchical techniques such as the one proposed by Bardossy and Duckstein, Weighted Counting Algorithm (WCA) [23] and Fuzzy C-Means (FCM) methods [24]. The IT-HFAM algorithm generates more rules than Cordón et al's hierarchical

implementation due to the need for an anytime solution capable of learning rapidly. Cordón uses a Genetic Algorithm (GA) to optimise the rule base and approximate rules to increase accuracy. A GA provides a good search technique for optimisation but the computational complexity involved makes it time consuming. The use of approximate rules reduces the interpretability of the rules generated by allowing direct manipulation of the output fuzzy set membership function parameters. The following section gives detailed analysis of the characteristics of the IT-HFAM algorithm.

Figure 9

IT-HFAM Algorithm Discussion

The same function (Equation 5) is used to analyse key characteristics of the IT-HFAM algorithm (accuracy, number of rules and decision uncertainty). This analysis is based on the IT-HFAM algorithm initialised with seven input linguistic symbols with membership functions evenly distributed over the input space. The training set and test set are the same as the previous experiment.

The first aspect of the IT-HFAM algorithm to consider is how the uncertainty of each decision is affected when the maximum amount of uncertainty is decreased (E_{max}=[100,95,90...10,5]). Figure 10 shows the mean and maximum decision uncertainty, of the entire test set, as the maximum uncertainty tolerated within each cell of the FAM is decreased. The graph demonstrates that the uncertainty of each decision is decreased in line with the maximum uncertainty tolerated until it reaches 35% of the maximum entropy. This limit is due to the fact that cell expansion does not occur when the applicability distribution is divided between only two output symbols. Hence, if the number of output symbols is increased this limit could be reduced, for example 10 output symbols produced a minimum uncertainty of 30.1%. How is the accuracy of the algorithm affected as the uncertainty tolerated is decreased?

Figure 10

Figure 11

Figure 11 demonstrates that the accuracy of the algorithm increases as the maximum uncertainty is decreased until expansion is stopped. The graph demonstrates that accuracy is increased in bursts as different areas of the decision space are discovered. At 75% of the maximum uncertainty, the function discovers the large peak at (1,1) and then between 65 and 50 the other smaller peaks are gradually discovered and finessed (Figure 12). This demonstrates how reducing the uncertainty tolerated within a decision can increase the accuracy of the function approximation.

Figure 12

The compaction stage of the IT-HFAM algorithm optimises the number of cells used within the HFAM, but does this affect the accuracy of the approximation? Figure 13 shows the accuracy before and after compaction against a range of E_{max} values.

Figure 13

Figure 13 shows that the accuracy after cell optimisation is not identical to the original accuracy. This change is because the gradient of the membership functions are altered during a merge, which in turn affects the cooperation between rules. This does not have a significant effect with a deviation of only +/- 0.7%. In the next section the IT-HFAM algorithm is used to demonstrate application to learning behaviour on a simulated mobile robot.

6. Learning Behaviour on a Mobile Robot

As discussed in the introduction, fuzzy rule base systems have been widely used within the area of mobile robotics. They provide a means of coping with continuous probabilistic environments while retaining a degree of human interpretability. The majority of fuzzy mobile robot controllers rely on statically user defined linguistic symbols [4, 25, 26]. As previously discussed, this structure constrains the accuracy of possible solutions [8]. Allowing the linguistic symbols to be "semantically free" (as in a Neuro-Fuzzy controller [27]) increases accuracy but diminishes interpretability. Hence, a mobile robot controller can suffer from the same problem of balancing the trade off between performance (accuracy) and interpretable solutions. In this section the IT-HFAM algorithm is used to construct a set of fuzzy control rules for a mobile robot though interaction with a supervisor. The amount of uncertainty tolerated, E_{max}, within the decision is used to change the trade off between accuracy and interpretability.

Task

The task performed is inspired by a collision avoidance problem for ships demonstrated by Furuhashi, Nakaoka and Uchikawa [28]. This implementation of the task differs from that addressed by Furuhashi et al. by using simulated robot sensors to detect obstacles instead of the exact range and bearing. The performance and rules generated by the IT-HFAM algorithm over a number of E_{max} values are compared. The task involves navigating a mobile robot to reach a goal while avoiding two other robots (Figure 14). The controlled robot moves with a constant speed with the two crossing robots moving at half the speed of the controlled robot. Varying the angular velocity, ω, alters the direction of the controlled robot. The controlled robot starts from the bottom line and attempts to navigate towards the goal at the top. The two other robots appear from the left and right hand side and cross between the robot and the goal. The aim of the task is to reach the goal without hitting either of the two crossing robots.

Figure 14

Environment

The task is performed using the Pioneer simulation environment, Stage[1]. The controlled, and crossing robots, are simulated Pioneer robots, while a red box is used to represent the goal. The controlled robot can detect an obstacle using eight sonar sensors evenly distributed over the front of the robot. Each sonar sensor returns a value indicating the distance to the nearest obstacle. The sensors are divided into two sets, left and right, and the minimum distance for each set is normalised ([0,1]) before being used as an input to the controller (Figure 15).

A bearing to the goal is acquired by simulating a front mounted camera to identify the position of the red box representing the goal. The goal is detected in the camera's field of the view by extracting the red areas of the image (red blob detection). The pixel column with the highest concentration of red within the image is marked as the centre of the goal. A normalised bearing to the goal is found by dividing the pixel column with the highest concentration by the total pixel width of the image. The input to the controller is a continuous value in the range [0,1] indicating the bearing to the goal.

In summary, the inputs to the controller are a bearing to the centre of the goal and the distance to the nearest object on the front left and right of the robot.

Figure 15

Experimental Setup

The controlled robot is trained using a human supervisor to present examples of correct operation during three scenarios. The starting position of the controlled robot is varied in each scenario giving the controlled robot exposure to a large proportion of the decision space. The three scenarios used to train the robot are displayed in Figure 16. The supervisor uses a joystick to indicate the correct angular velocity (continuous value over the range [0,1] where 0.5 is forwards) by monitoring the inputs to the algorithm (left/right sonar and the position of the

[1] http://playerstage.sourceforge.net/

goal). The training examples are only provided when the supervisor deems the robot's behaviour is incorrect.

Figure 16

The IT-HFAM algorithm is initialised with three membership functions evenly distributed over each of the input variables and five membership functions over the output variable. The task performance and characteristics of the IT-HFAM algorithm are compared over a number of E_{max} values: 100, 90, 75, 50. Three test scenarios, where the position of the goal and crossing robots are varied, are used to determine the performance of the trained controller. Task performance is obtained by studying the amount of deviation from the shortest path to the goal. The three test scenarios are displayed in Figure 17. The shortest path for each scenario is a straight line between the start position and the goal. The amount of deviation is calculated by comparing the distance travelled by the controlled robot during the test scenario and the shortest path.

Figure 17

Results and Discussion

Table 3 outlines the average deviation for three runs on each test scenario (D_1, D_2, D_3 in millimetres), the total deviation (D_{total}) and IT-HFAM characteristics (C_t = Cells before compaction, C_r = Cells after compaction) for values of E_{max}.

Table 3

Table 3 demonstrates that E_{max} can be used to control the trade off between the number of rules generated (C_t) and the accuracy of the solution (D_{total}). With no specialisation (E_{max} = 100), the controlled robot was capable of navigating to the target in all test scenarios but deviated from the optimal path by 203mm on average. The algorithm produced only 15 rules that remain highly interpretable to the supervisor. Some of the rules are listed in Table 4.

Table 4

The selection of rules presented resembles control rules a human may have used to solve the problem and hence demonstrates that the algorithm can produce interpretable solutions. Only 6 out of the 15 rules are displayed helping to demonstrate that a human designer may not have designed a rule base that captures all situations and the arbitration between the two tasks.

As the amount of uncertainty tolerated (E_{max}) is decreased, the deviation from the optimal path is reduced with the side affect that the number of rules generated increases significantly. For example when E_{max} is 50, the deviation from the optimal path is 38.7% better than without specialisation while the number of rules generated has increased to 609. The growth in rules indicates that a more intelligent specialisation technique would be required for high dimensional problems. The hierarchical structure used to perform specialisation can now be used to increase the interpretability of a complex rule base containing 609 rules. As the hierarchical structure is maintained, a representation containing the initial 15 rules can be retrieved. For example when E_{max} = 50, a representation containing the initial rules (as in Table 4 when E_{max} = 100) can be extracted from the root of the hierarchical rule base. This indicates that the hierarchical structure could provide a means of increasing the interpretability of complex rule bases. Work is continuing on how this hierarchical structure could allow a supervisor to explore and understand complex behaviour.

These results demonstrate that adjusting the amount of uncertainty tolerated within the decision making process can be used as a trade off between the interpretability and performance of a solution. Figure 18 demonstrates how the deviation from the optimal path is reduced in test scenario 3 over E_{max} values 100, 75 and 50. The reduction in the deviation from the optimal path is achieved through specialisation of the decision space by more accurately modelling the behaviour demanded by the supervisor. For example, specialisation adjusts the distance at which the controlled robot performs obstacle avoidance (Figure 18).

Figure 18

7. Conclusions and Future Work

This paper has demonstrated a novel anytime expansion policy for a hierarchical Fuzzy Rule Based System that can achieve levels of accuracy comparative with existing function approximation methods. The algorithm uses an information theoretic approach to attach a level of uncertainty to each decision made and to determine if any rule does not effectively model the underlying decision space within a specified degree of uncertainty. The algorithm has been demonstrated to approximate an intermediate complex function and learn a simulated mobile robot task. The results demonstrate that the accuracy and interpretability trade off can be controlled using a single tuneable parameter (E_{max}) that represents the maximum uncertainty tolerated within a decision.

Future work will focus on investigating limitations of the algorithm when applied to a real mobile robot using a human supervisor. Another important aspect of constructing a human interpretable controller is the visualisation of complex controllers with large numbers of rules. When a high degree of certainty in the decision is required, the algorithm described within this paper produces a solution with a large number of rules. A possible method of interpreting a large number of rules could arise from the hierarchical structure of the rule base. This may provide a means of allowing the supervisor to explore different parts of the decision space (rules) at varying levels of complexity.

Acknowledgements

This work was funded by the BAE SYSTEMS Advanced Technology Centre, Bristol UK.

References

1. Ogata, K., *Morden Control Engineering*. International Third Edition ed. 1997: Tim Robbins. 997.
2. Zadeh, L.A., *Soft Computing and Fuzzy Logic.* IEEE Software, 1994: p. 48-56.

3. Hagras, H., V. Callaghan, and M. Colley, *Outdoor Mobile Robot Learning and Adapation.* IEEE Robotics and Automation Magazine, 2001. **8**(3): p. 53-69.
4. Tunstel, E., T. Lippincott, and M. Jamshidi. *Introduction to Fuzzy Logic with Application to Mobile Robotics*. in *First National Students Conference of the National Alliance of NASA University Research Centres*. 1996. NC A&T State Univ, Greensboro, NC.
5. Hoffman, F. and G. Pfister, *Evolutionary Design of a Fuzzy Knowledge Base for a Mobile Robot.* International Journal of Approximate Reasoning, 1997. **17**(4): p. 447-469.
6. Holve, R. and P. Protzel. *Generating Fuzzy Rules by Learning from Examples*. in *Biennial Conference of the North American Fuzzy Information Processing Society - NAFIPS*. 1996. Berkeley, CA, USA.
7. Zadeh, L.A., *Fuzzy Logic.* Computer, 1988. **21**(4): p. 83-93.
8. Bastian, A., *How to handle the flexibility of linguistic variables with applications.* Intl. Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 1994. **2**(4): p. 463-484.
9. Alcala, R., J. Casillas, O. Cordón, and F. Herrera, *Building Fuzzy Graphs: Features and Taxonomy of Learning for Non-grid-oriented Fuzzy Rule-based Systems.* Journal of Intelligent and Fuzzy Systems, 2001. **11**: p. 99-119.
10. Cordón, O., F. Herrera, F. Hoffmann, and L. Magdalena, *Genetic Fuzzy Systems: Evolutionary Tuning and Learning of Fuzzy Knowledge Bases*. 2001: World Scientific.
11. Carse, B., T.C. Fogarty, and A. Munro, *Evolutionary Learning of Fuzzy Rule Based Controllers using Genetic Algorithms.* Fuzzy Sets and Systems, 1996. **80**: p. 273-293.
12. Cordón, O., F. Herrera, and I. Zwir, *Fuzzy Modeling by Hierarchically built Fuzzy Rule Bases.* International Journal of Approximate Reasoning, 2001. **27**: p. 61-93.
13. Melhuish, C. and T. Fogarty. *Applying Restricted Mating Policy to Determine State Space Niches Using Immediate and Delay Reinforcement*. in *Evolutionary Computing, AISB Workshop*. 1994. Leeds, UK.
14. Holve, R. *Investigation of Automatic Rule Generation for Hierarchical Fuzzy Systems*. in *IEEE World Congress on Computational Intelligence, FUZZ IEEE*. 1998. Anchorage, Alaska.
15. Grefenstette, J.J. and C.L. Ramsey. *An Approach to Anytime Learning*. in *Ninth International Machine Learning Workshop*. 1982. San Mateo, CA: Morgan Kaufmann.
16. Shannon, C.E., *A mathematical theory of communication.* Bell System Technical Journal, 1948. **27**: p. 379-423 and 623-656.
17. Quinlan, J.R., *Induction of Decision Trees*. Readings in Machine Learning, ed. J.W. Shavlik and T.G. Dietterich. 1990: Morgan Kaufmann.
18. Al-sharhan, S., F. Karray, W. Gueaieb, and O. Basir. *Fuzzy Entropy: A Brief Survey*. in *The 10th IEEE International Fuzzy Systems Conference*. 2001.
19. Zadeh, L.A., *Probability measures of Fuzzy Events.* Journal Math. Anal. Appl., 1965. **23**: p. 421-427.
20. Holve, R. *Rule Generation for Hierarchical Fuzzy Systems*. in *North American Fuzzy Information Processing Society - NAFIPS*. 1997.

21. Hellendoorn, H. and C. Thomas, *Defuzzifcation in Fuzy Controllers.* Journal of Intelligent and Fuzzy Systems, 1993. **1**: p. 109-123.
22. Cordón, O., F. Herrera, and I. Zwir, *A Hierarchical Knowledge-Based Environment for Linguistic Modelling: Models and Iterative Methodology*. 2000, Department Of Computer Science and Artifical Intelligence, Universidad de Granada: Granada, Spain.
23. Bardossy, A. and L. Duckstein, *Fuzzy Rule-based Modelling with Application to Geophysical, Biological and Engineering Systems*. 1995: CRC Press.
24. Bezdek, J.C., *Fuzzy Models For Pattern Recognition: Methods that search for structures in Data*, ed. S. Pal. 1992: IEEE Press. 539.
25. Hagras, H., V. Callagham, and M. Colley, *Learning and adaption of an intelligent mobile robot navigator operating in unstructured environment based on a novel online Fuzzy-Genetic system.* Fuzzy Sets and Systems, 2004. **141**(1): p. 107-160.
26. Soffiotti, A., E.H. Ruspini, and K. Konolige, *Practical Applications of Fuzzy Technologies*, in *Handbooks of Fuzzy Sets*, H.-J. Zimmermann, Editor. 1999, Kluwer Academic. p. 185-205.
27. Marichal, G.N., L. Acosta, L. Moreno, J.A. Méndez, J.J. Rodrigo, and M. Sigut, *Obstacle avoidance for a mobile robot: A neuro-fuzzy approach.* Fuzzy Sets and Systems, 2001. **124**(2): p. 171-179.
28. Furuhashi, T., K. Nakaoka, and Y. Uchikawa, *A Study on Fuzzy Classifier Systems For Finding a Control Knowledge of Multi-input Systems.* Genetic Algorithms and Soft Computing, 1996: p. 489-502.

Table 1 – IT-HFAM Results

Experiment	1 Training Set Iteration				2 Training Set Iterations			
	C_t	C_r	$RMSE_t$	$RMSE_r$	C_t	C_r	$RMSE_t$	$RMSE_r$
Single Rule Outputting 0	1	1	0.1496	0.1496	1	1	0.1496	0.1496
IT-HFAM2 – 90%	7	4	0.0951	0.0941	16	**7**	0.0898	**0.0880**
IT-HFAM2– 70%	13	7	0.0981	0.0913	43	14	0.0778	0.0771
IT-HFAM2– 30 %	3181	233	0.0596	0.0768	11014	274	0.0375	0.0441
IT-HFAM7 – 90%	49	3	0.1014	0.1036	49	3	0.1014	0.1036
IT-HFAM7– 55%	247	55	0.0951	0.0857	1003	156	0.0545	0.0578
IT-HFAM7– 45 %	925	153	0.0717	0.0863	3952	**336**	0.0242	**0.0291**

Table 2 - Comparison with existing function approximators

Experiment	No. of Rules	RMSE
Static Weight Counting Algorithm (S-WCA)[23]	9	0.0868
Hierarchical S-WCA [12]	316	0.0134
Fuzzy C-Means (FCM) [24]	6	0.1124
Hierarchical FCM[12]	9	0.0403
IT-HFAM2 E_{max} = 90%	7	0.0880
IT-HFAM7 E_{max} = 45%	336	0.0291

Table 3 – IT-HFAM Simulated Robot Task Results

E_{max}	C_t	C_r	D_1	D_2	D_3	D_{total}
100	27	15	171	249	190.5	610.5
90	83	33	156	223.5	157	536.5
75	797	183	165.5	193.5	143	502
50	3066	609	141.5	170	62.5	374

Table 4 – Rule Generated for E_{max} = 100

Task	Sonar Left	Sonar Right	Vision	Action
Track	Far	Far	Left	Left
Track	Far	Far	Middle	Forward
Track	Far	Far	Right	Right
Avoid	Close	Far	Left OR Middle OR Right	Left
Avoid	Inrange	Far	Left OR Middle OR Right	Forward
Avoid	Close OR Inrange OR Far	Close	Left OR Middle OR Right	Right