Coevolutionary synthesis of fuzzy decision support systems

Huang, Haoming

2009

Huang, H. M. (2009). Coevolutionary synthesis of fuzzy decision support systems. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/19087>

<https://doi.org/10.32657/10356/19087>

NANYANG TECHNOLOGICAL UNIVERSITY

Coevolutionary Synthesis of Fuzzy Decision Support Systems

A thesis submitted to the Nanyang Technological University in partial fulfillment of
the requirement for the degree of Doctor of Philosophy

by

HUANG HAOMING

School of Computer Engineering

2009

Acknowledgements

I am indebted to my supervisor Associate Professor Michel Pasquier. His profound knowledge in computational intelligence has inspired and shaped my PhD research topic in this promising area. I thank him for his kind support and patient guidance through the most difficult periods of my PhD study. I would also like to express my great thank to Associate Professor Quek Hiok Chai for his valuable ideas and advices on my research work.

I am grateful to my lab mates at the Centre for Computational Intelligence, including Ang Kai Keng, Tung Whye Loon, Ting Chan Wai, Liu Feng, Ma Li, Wang Zhen, Wang Di, Tony Tan and Richard Oentaryo, for their helpful discussions and valuable feedback on my research.

I would like to thank the laboratory technicians, Tan Swee Huat and Lau Boon Chee, for their helpful supports. I would like to thank Nanyang Technological University for providing me the research scholarship for three years.

I am forever grateful to my parents, relatives and friends for their understanding, encouragement and support through the years.

Table of Contents

ACKNOWLEDGEMENTS	I
TABLE OF CONTENTS	II
LIST OF FIGURES.....	IV
LIST OF TABLES.....	VI
ABSTRACT	VII
CHAPTER 1 INTRODUCTION.....	1
1.1 MOTIVATIONS	1
1.2 OBJECTIVES.....	4
1.3 ORGANIZATION OF THE THESIS	5
CHAPTER 2 LITERATURE REVIEW.....	7
2.1 DECISION SUPPORT SYSTEMS.....	7
2.2 FUZZY RULE-BASED SYSTEMS	9
2.3 DESIGN ISSUES IN FRBS.....	13
2.4 EVOLUTIONARY FUZZY SYSTEMS	21
2.4.1 Evolutionary approaches for determining fuzzy membership functions	22
2.4.2 Evolutionary approaches for rule learning	24
2.4.3 Evolutionary approaches for selecting fuzzy inference operators	26
2.4.4 Evolutionary approaches for determining MFs and learning rules.....	27
2.4.5 Special evolutionary learning or optimizing methods for FRBS.....	29
CHAPTER 3 HIERARCHICAL COEVOLUTIONARY FUZZY SYSTEM SYNTHESIS.....	35
3.1 OVERVIEW	35
3.2 ARCHITECTURE OF THE FDSS SYNTHESIZED BY HiCEFS	37
3.3 IRREGULAR-SHAPED MEMBERSHIP FUNCTION	40
3.4 HIERARCHICAL COEVOLUTIONARY GENETIC ALGORITHM.....	43
CHAPTER 4 IRREGULAR-SHAPED MEMBERSHIP FUNCTIONS.....	46
4.1 OVERVIEW OF ISMFs.....	46
4.2 EVOLUTIONARY GENERATION OF ISMFs	48
4.2.1 The representation of ISMFs	48
4.2.2 The representation of ISMF partitions	49
4.2.3 The ISMF extraction process.....	50
4.2.4 Maintaining the interpretability of the ISMF partitions	53
4.2.5 Procedure and settings of the genetic algorithm	56
4.2.6 Fitness evaluation of the ISMFs	57
4.3 BENCHMARKING RESULTS AND ANALYSIS.....	58
4.3.1 Experimental settings.....	58
4.3.2 Pima diabetes diagnosis	59
4.3.3 Phoneme classification	59
4.3.4 Automobile MPG prediction.....	60
4.3.5 Solar activity prediction.....	61

Coevolutionary Synthesis of Fuzzy Decision Support Systems

4.3.6	<i>Analysis and discussions</i>	63
CHAPTER 5 HIERARCHICAL COEVOLUTIONARY GENETIC ALGORITHM		66
5.1	OVERVIEW OF THE HIERARCHICAL COEVOLUTIONARY MODEL	66
5.2	COEVOLUTION OF ISMF PARTITIONS	69
5.2.1	<i>Genetic representation</i>	69
5.2.2	<i>Extracting the ISMF partitions</i>	71
5.2.3	<i>Fitness evaluation and assignment</i>	74
5.2.4	<i>Settings of HCGA</i>	75
5.3	EXPERIMENTAL VALIDATION	76
5.3.1	<i>Experimental settings</i>	77
5.3.2	<i>Pima diabetes diagnosis</i>	77
5.3.3	<i>Phoneme classification</i>	78
5.3.4	<i>Ionosphere classification</i>	79
5.3.5	<i>Automobile MPG prediction</i>	80
5.3.6	<i>Traffic flow prediction</i>	81
5.3.7	<i>Analysis and discussions</i>	82
CHAPTER 6 SYNTHESIZING FUZZY DECISION SUPPORT SYSTEMS WITH THE HICEFS APPROACH		89
6.1	INTRODUCTION.....	89
6.2	REPRESENTING POSSIBLE FDSS IN HCGA	89
6.3	COEVOLUTIONARY SYNTHESIS PROCESS	92
6.4	EXTRACTING THE SOLUTION FROM HCGA.....	95
6.5	SETTINGS OF THE HCGA IN HICEFS	96
6.6	FUZZY RULE BASE DISCOVERY PROCESS.....	97
6.7	FITNESS EVALUATION OF THE SOLUTION	99
6.8	EXPERIMENTAL VALIDATION AND REAL-WORLD APPLICATIONS	101
6.8.1	<i>Experimental settings</i>	101
6.8.2	<i>Stock prediction and trading</i>	101
6.8.3	<i>Pima diabetes diagnosis</i>	109
6.8.4	<i>Gamma-ray signal detection</i>	111
6.8.5	<i>Concrete compressive strength prediction</i>	115
6.8.6	<i>Analysis and discussions</i>	119
CHAPTER 7 CONCLUSION AND FUTURE WORK		121
7.1	ACHIEVEMENTS.....	121
7.2	LIMITATIONS	123
7.3	FUTURE RESEARCH RECOMMENDATIONS.....	123
AUTHOR'S PUBLICATIONS		127
REFERENCES		128

List of Figures

Figure 1 Basic model of a fuzzy rule-based system	9
Figure 2 Example of a fuzzy partition on a variable.....	11
Figure 3 Commonly used membership functions (a) triangular, (b) trapezoidal, (c) Gaussian	11
Figure 4 The example of implementing a fuzzy rule-based system as a neuro-fuzzy system.....	15
Figure 5 Evolving the fuzzy partition on the input space by using binary coded chromosomes.....	23
Figure 6 Illustrations of different evolutionary approaches for learning both fuzzy membership functions and rule base in the FRBS.....	27
Figure 7 The relation between the components of the FDSS and hierarchical coevolutionary genetic algorithm in the HiCEFS approach.....	36
Figure 8 The architecture of FDSS derived by the HiCEFS approach.....	37
Figure 9 Examples of possible shapes represented by ISMFs.....	46
Figure 10 The genetic segment of an ISMF.....	49
Figure 11 Content of a chromosome representing the ISMF partitions for all input variables.....	50
Figure 12 Determining the position of the pivot points on an input variable	51
Figure 13 Determining the relative positions the sampling points in a ISMF according to the per unit offsets	52
Figure 14 Example of (a) a complete fuzzy partition and (b) an incomplete fuzzy partition.....	53
Figure 15 Example of (a) a distinguishable fuzzy partition and (b) a non-distinguishable fuzzy partition	54
Figure 16 Repair scheme (a) for reduce overlapping between two ISMFs and (b) for reposition sampling point with negative membership value	55
Figure 17 Repair scheme for ensuring two nearby ISMFs overlap at a proper membership value.....	55
Figure 18 (a) The predicted and the actual time series of the change in the monthly smoothed sunspot number (b) the errors of the predicted time series.....	62
Figure 19 Derived ISMFs and triangular MFs for (a) Pima diabetes diagnosis and (b) solar activity prediction	64
Figure 20 Flowchart of the evolutionary process of HCGA.....	67
Figure 21 Structure of the 2-Level HCGA for deriving ISMFs.....	70
Figure 22 The example of determining the positions of the pivots according to their offsets	73
Figure 23 Prediction of the traffic density on Lane 1 at 5 minutes later by M2.....	82
Figure 24 The ISMF fuzzy partitions in M1T2 and M2 in the Pima diabetes diagnosis	85
Figure 25 The structure of the 5-level HCGA for coevolving a FDSS	90
Figure 26 The evolutionary process of the 5-level HCGA for the synthesis of a FDSS93	
Figure 27 Flowchart of the fuzzy rule base discovery process in HiCEFS approach....	98

Coevolutionary Synthesis of Fuzzy Decision Support Systems

Figure 28 The proposed financial trading decision support system with a predictive model derived by HiCEFS approach	104
Figure 29 Price of HSI and trading signals given by FTDSS-HiCEFS	106
Figure 30 Portfolio values on HSI achieved by the financial trading decision support systems	107
Figure 31 Enlarged part of Figure 29 from time $t=1000$ to 1250	107
Figure 32 The derived membership functions in FTDSS-HiCEFS for HSI	108
Figure 33 The ISMFs in the FDSS derived by HiCEFS approach for Pima diabetes diagnosis	111
Figure 34 Illustration of some image parameters adapted from [197]	112
Figure 35 The relation between the age and the concrete compressive strength	118

List of Tables

Table I Fuzzy rules in the example fuzzy rule-based system for car braking.....	15
Table II The parameters in the popular clustering algorithms	17
Table III Result comparison on Pima diabetes diagnosis problem.....	59
Table IV Result comparison on Phoneme classification problem	60
Table V Result comparison on automobile MPG prediction.....	61
Table VI Result comparison on solar activity prediction	63
Table VII The settings of the parameters in the 2-level HCGA	76
Table VIII Results on Pima diabetes diagnosis	78
Table IX Results on phoneme classification.....	79
Table X Results on ionosphere classification	80
Table XI Results on automobile MPG prediction.....	80
Table XII Prediction results of the traffic flow density on Lane 1	81
Table XIII The average training time (in seconds) of the different fuzzy systems in the experiments.....	86
Table XIV The positive and negative effects of adopting ISMFs in a FRBS.....	88
Table XV Benchmarking results of the financial trading decision support systems on HSI.....	106
Table XVI The fuzzy rule base in the FTDSS-HiCEFS for predicting HSI.....	108
Table XVII The diagnosis results on the Pima diabetes	110
Table XVIII The fuzzy rules in the FDSS derived by HiCEFS approach for Pima diabetes diagnosis	111
Table XIX The features of the sample data in MAGIC telescope data set.....	112
Table XX Classification results on MAGIC gamma telescope dataset	113
Table XXI Ranges of the variables of the dataset of concrete compressive strength..	116
Table XXII The results of the concrete compressive strength prediction	116
Table XXIII Examples of the membership functions in the derived FDSS for concrete compressive strength prediction	118
Table XXIV The examples of the fuzzy rules in the derived FDSS for concrete compressive strength prediction	119

Coevolutionary Synthesis of Fuzzy Decision Support Systems

by
Huang Haoming

A thesis submitted to the Nanyang Technological University in fulfillment of the
requirement for the degree of Doctor of Philosophy
2008

Abstract

Many essential applications in finance, medicine, engineering, and science require increasingly complex decision-making capabilities. There is accordingly a growing demand for decision support systems (DSSs) to assist humans in their tasks. To provide accurate and reliable decision support, a DSS needs not only to be robust in the face of the uncertainty but also to model the decision-making logic in a form that is understandable. Compared with other machine learning methods, fuzzy rule-based systems possess the merits of providing strong approximate reasoning in the presence of imprecise data while representing domain knowledge as a set of interpretable semantic rules. Using them to realize DSSs is thus a most suitable approach yielding powerful fuzzy decision support systems (FDSSs).

However, the synthesis of an optimal FDSS with well-balanced accuracy and interpretability is an arduous task. Experience shows that it is very difficult for human experts to manually design its two most important components, the fuzzy membership functions and fuzzy rule base, which directly affect system performance. Ad-hoc architectures, which must be redesigned anew for every application, and improperly chosen parameters typically introduce unwanted biases and unavoidably result in suboptimal systems. Ideally, the decision-making logic should therefore be induced automatically from example and further optimized for the problem at hand. To achieve this goal, a generic approach is needed that can automatically synthesize an accurate and interpretable FDSS, while requiring minimal or no human effort.

Coevolutionary Synthesis of Fuzzy Decision Support Systems

Even though learning techniques for FDSS design are available, such as clustering and gradient learning algorithms, their performance relies heavily on intrinsic algorithm parameters, the optimal tuning of which requires a tedious trial-and-error process. In comparison, evolutionary algorithms, which realize the synthesis as a general search problem, are less dependent on a priori knowledge and less prone to be trapped in local optima. Furthermore, they are proven to be robust and efficient search techniques in various kinds of domain spaces. Evolutionary FDSS design constitutes therefore a promising approach, resulting into so-called evolutionary fuzzy systems.

This thesis describes a systematic approach, termed *Hierarchical Co-Evolutionary Fuzzy system Synthesis (HiCEFS)*, to automatically synthesize from data fuzzy decision support systems with a balanced level of accuracy and interpretability. A hierarchical coevolutionary genetic algorithm (HCGA) is devised as the main learning mechanism. HCGA efficiently coevolves the fuzzy membership functions and rules simultaneously in separate yet cooperating populations. A generic type of fuzzy membership function, termed *Irregular-Shaped Membership Function (ISMF)*, is proposed that is more flexible than commonly used fuzzy membership functions, thereby increasing accuracy and interpretability. Moreover, techniques for overcoming fitness stagnancy and facilitating rule learning are devised to further enhance the HiCEFS approach.

The HiCEFS approach can derive from sample data a near-optimal FDSS to solve any classification or regression problem while requiring minimal human intervention. Various FDSSs have been synthesized by HiCEFS for real-world applications such as stock trading, diabetes diagnosis, traffic flow prediction, and gamma signal detection. Promising results have been obtained and analyzed to provide a deeper understanding as well as a thorough experimental validation of the proposed approach.

Supervisor:

Assoc Prof Michel B. Pasquier
Division of Computer Science
School of Computer Engineering
Nanyang Technological University

Chapter 1

Introduction

1.1 Motivations

Many essential application domains such as finance, medicine, engineering, and science require increasingly complex decision-making capabilities. There is accordingly a growing demand for decision support systems (DSSs) to assist humans in their daily tasks. To meet the requirement of providing accurate and reliable decision support, a DSS needs not only to be robust enough to provide accurate decision support in the face of the uncertainty and vagueness present in real-world problems, but also to provide the decision-making logic in a form that can be understood and accepted by decision makers.

Various techniques, including machine learning and computational intelligence-based approaches, can be adopted in the decision support systems. For instance, Bayesian decision networks [1] apply the well-established theory of probability and statistics to complex decision problems. Artificial neural networks can model the complex relationships between the inputs (evidences) and outputs (conclusions) through emulating the structures and functions of biological neurons and synaptic connections in the human brain [2, 3]. The support vector machine (SVM) [4, 5] is a machine learning technique that computes the separating hyperplanes that maximize the margin between two sets of instances. The main advantages of SVM are that it is possible to give bounds on the generalization error, and to select the best system using the structural risk minimization [4].

Although the techniques discussed above have been successfully applied in several decision problems [6-8], their common shortcoming is that the coefficients in these numerical models have no meaning for human users and therefore they cannot provide interpretable decision-making logic. Compared to these techniques, decision trees [9] are able to break down a complex decision-making process into a collection of simpler decisions and thus easier to be interpreted. However, when dealing with complex

problems, the decision tree can grow to a very complex structure, which can also be difficult to interpret. Fuzzy rule-based systems (FRBSs) [10] possess the merits of providing strong approximate reasoning in the presence of imprecise data while representing domain knowledge as a set of interpretable semantic rules. One advantage of the FRBSs over decision trees is the greater flexibility of the decision boundaries [11] and hence greater robustness in decision making. Therefore, using FRBSs to realize DSSs has become a most suitable approach that yields powerful fuzzy decision support systems (FDSSs).

However, the synthesis of an optimal or near-optimal fuzzy decision support system with well-balanced accuracy and interpretability is an arduous task [12, 13]. The accuracy of a FDSS refers to correctness of the decisions made by the FDSS. The interpretability means that human users are able to understand the decision-making logic by inspecting the fuzzy linguistic rules in the FDSS. The fuzzy membership functions that model classes of input/output and the fuzzy rules that map input to output are two most important components that directly affect the performance of a FDSS in terms of both accuracy and interpretability. Experience shows that it is very difficult for a human expert to manually design these two components, since ad hoc architectures and improperly chosen parameters introduce unwanted biases and unavoidably result in suboptimal systems. Moreover, the design effort must be repeated anew for every application problem. Ideally, the decision-making logic should therefore be induced automatically from examples and further optimized for the problem at hand. To achieve this goal, a generic approach is needed that can automatically synthesize an accurate and interpretable FRBS, hence an efficient and dependable FDSS, while requiring minimal or no human effort.

Even though learning techniques for FRBS design are available, such as clustering algorithms [14] and gradient learning algorithms [15], they have their own drawbacks. The clustering algorithms are used for generating the fuzzy membership functions that partition the input-output spaces [16, 17]. They are also adopted to derive the fuzzy rule base, in which each fuzzy rule corresponds to a resultant cluster [18-20]. Many existing clustering techniques [16, 17, 21] rely heavily on the judicious selection of a set of

intrinsic algorithm parameters, such as the number of clusters, learning constants, neighborhood radius and thresholds, which need to be pre-specified by the designers. This means that even a slight difference in parameters may generate a resultant system that will exhibit a great discrepancy in performance.

The gradient learning methods [15, 22, 23] are used to improve the performance of the fuzzy rule-based systems through adapting their parameters. However, because of the complexity in the fuzzy rule-based system synthesis problem, they are prone to be trapped in local optima and consequently limit the performance of the resultant FDSS.

The rule learning algorithms also have their own limitations. The algorithms proposed in [24-26] consider all possible combination of fuzzy membership functions as fuzzy rules. The drawback of these techniques is that they are inefficient and the complexity of the rule base increases drastically as the number of the input variables increases. The other kind of rule learning techniques [16, 27-29] incrementally derive rules to cover the training samples. Although they are computationally efficient, these techniques require predefined fuzzy partitions, which can heavily affect the rule learning results. Furthermore, these rule learning techniques derive the rules with the best individual performance without considering the cooperation among the learned fuzzy rules, which is more important to the inference accuracy [30].

In comparison with the clustering and gradient learning techniques, evolutionary algorithms [31-34] are less dependent on a priori knowledge, less sensitive to their parametric settings and less prone to be trapped in local optima. Given a proper fitness function to evaluate the correctness of a candidate solution, they can robustly and efficiently search for optimal or near-optimal solutions in very different kinds of domain spaces [35]. Thus, evolutionary algorithms can be applied as robust and generic methods for FRBS synthesis that require minimal application domain knowledge [36]. One main target of evolutionary learning is the fuzzy membership functions [19, 37-39]. Furthermore, the evolutionary algorithms are suitable for generating the fuzzy rule base by considering it as a combinational problem. The cooperation among the fuzzy rules is

reflected in the fitness value and is thus improved through evolution [40-44]. However, the basic evolutionary techniques adopted in the early works [38, 40, 45] may be not efficient enough for solving the problem of learning and tuning a complete fuzzy rule-based system because of its high complexity.

With recent advances in evolutionary algorithms, more sophisticated evolutionary based techniques are proposed for learning FRBS. Some researchers optimized several components of a fuzzy rule-based system, such as membership functions and fuzzy rule bases, simultaneously in an evolutionary population [46, 47]. Different coding methods and genetic operators are tailored for different parts of the chromosomes [39]. Furthermore, some researchers started to utilize coevolutionary algorithms to learn the FRBS in a divide-and-conquer manner, which solves the complex learning task by dividing it into several easier subtasks [48-50]. The efficiency of the coevolutionary algorithms depends on a proper decomposition of the learning task, which in most cases relies on human expertise and domain analysis [51]. In the specific case of synthesizing a FRBS, the fuzzy membership functions and fuzzy rule base in a FRBS are simultaneously evolved in separate populations. These coevolutionary-base approaches encourage the cooperation between fuzzy membership functions and fuzzy rule base, which in consequence improve the accuracy and interpretability of the FDSS. With the advantages discussed above, such advanced evolutionary techniques constitute a promising approach to automate FDSS synthesis and improve the final FDSS.

Summarizing the discussion above, it is desirable to devise a generic approach based on evolutionary technique that is able to automatically synthesize a fuzzy decision support system from sample data, with the goal of achieving high system performance and minimizing the human effort in synthesis process.

1.2 Objectives

From the analysis in the previous section, the objective of this research is defined as follows. The primary objective of this project is to devise a generic coevolutionary-based approach for automatically synthesizing fuzzy decision support systems from sample data.

Two most important properties of the derived fuzzy decision support systems are as follows. First, a competent level of accuracy and interpretability should be achieved by the learned FDSS. Second, human efforts and interventions in the synthesis process should be minimized. Fulfilling these two requirements, the generic approach can be conveniently adopted to synthesize specific fuzzy decision support systems for solve complex decision problems from various fields.

For achieving these objectives, this thesis proposes a systematic approach, termed Hierarchical Co-Evolutionary Fuzzy system Synthesis (HiCEFS), to automatically construct from data fuzzy decision support systems with a balanced level of accuracy and interpretability. A hierarchical coevolutionary genetic algorithm (HCGA) is devised as the main learning mechanism, which efficiently searches solutions for the complex task of synthesizing a FDSS by dividing it into several easier subtasks of finding well cooperating components in the system. The salient components of the underlying FRBS, including fuzzy membership functions and fuzzy rules, are simultaneously coevolved by separate populations in HCGA. A generic type of fuzzy membership function, termed Irregular-Shaped Membership Function (ISMF), is proposed and adopted in the FRBS. Being flexible and able to approximate the commonly used types of fuzzy membership functions, the ISMFs help to ease the difficulties in deriving fuzzy membership functions and increase the accuracy and interpretability of the system. Moreover, computational techniques for overcoming fitness stagnancy and facilitating rule learning are devised to further enhance the HiCEFS approach.

1.3 Organization of the Thesis

This thesis is organized into seven chapters. Chapter 2 presents a brief survey of the background knowledge and research work of the related topics. Chapter 3 briefly introduces the overview of the Hierarchical Co-Evolutionary Fuzzy system Synthesis (HiCEFS) approach. The architecture of the fuzzy decision support system derived by HiCEFS approach and salient features of the HiCEFS approach are also introduced. Chapter 4 presents a generic type of membership functions called Irregular-Shaped Membership Functions (ISMF) as a crucial component in the HiCEFS approach. Its

capability for representing various data distributions is proven by experimental validations. Chapter 5 introduces a hierarchical coevolutionary genetic algorithm (HCGA) which is adopted as the main learning mechanism in the HiCEFS approach. Chapter 6 discusses the details of the HiCEFS approach of learning the whole fuzzy decision support system and applies the HiCEFS approach to synthesize fuzzy decision support systems for solving real-world applications. Chapter 7 draws the conclusion of the thesis and points out future research directions.

Chapter 2

Literature Review

2.1 Decision support systems

The goal of developing computer-based decision support systems (DSSs) is to help humans to make more accurate decisions with lowered risks [52]. For such systems, the underlying relationships between variables (such as symptoms and diagnosed disease) are organized in a form that is readily executable to infer outcomes (e.g., recommend decisions) when receiving new inputs. One primary requirement of decision support systems is accuracy, that is the correctness of the decisions given by the DSSs. The other is interpretability, which means the decision-making logic should be understandable by human users. This section briefly reviews several dominant DSSs, including Bayesian networks, artificial neural networks, support vector machines, decision trees, and fuzzy rule based systems.

Bayesian networks

Bayesian networks [1], also called belief networks, are probabilistic models that are often used as DSSs. Their structures connect up variables represented as nodes with the arcs which indicate the conditional independencies among the variables and their parameters represent the probability distribution for those conditional independencies [53]. Given certain input variables (e.g., symptoms), the network computes the probabilities of the presence of other variables (e.g., diseases), hence it can be used to solve decision problems. In terms of interpretability, although the arcs show the causality among the variables, the parameters in a Bayesian network remain very difficult to be analyzed by human users [54].

Artificial neural networks

Artificial neural networks (ANN) are inspired by the structure of human brain [2, 3]. They are networks where nodes and edges model the neurons and synapses, respectively. Learning algorithms progressively update the weights of the edges in a direction where

input-output pairs better approximate the historical data [55, 56]. ANNs are regarded as black-boxes as the networks do not have corresponding meanings in the real-world. Hence they are unable to be interpreted by human users.

Support vector machines

Support vector machines (SVM) are machine learning techniques that view the data in a multi-dimensional space [4, 5]. They classify data points by computing a separating hyperplane that maximizes the margin between two classes of data points. Similar techniques include principle component analysis (PCA) [57] and Linear Discriminant Analysis (LDA) [58], which look for a linear combination of features (which constitutes a hyperplane) that best separate the data from different classes. The parameters used in these techniques are purely mathematical, that is, they do not help in interpretability of the results.

Decision trees

Decision trees break a complex decision-making process into a collection of simpler decisions in a tree structure [9]. They are white box models which are easy to understand by humans. However, the crisp logic in tree structure is sensitive to the noise and may give very different decisions for similar data [59]. Furthermore, the decision tree induction methods [60] are unstable, which means small perturbations in the training dataset can generate a very different decision tree [61].

Fuzzy rule-based systems

Fuzzy rule-based systems (FRBSs) [10, 62] store their knowledge in the form of interpretable fuzzy rules, which form their main advantage over Bayesian networks and artificial neural networks. The fuzzy models of inputs and outputs allow for greater uncertainty and vagueness in data and greater flexibility of the decision boundaries [63], hence greater robustness in decision making than crisp logic-based decision trees. Therefore, using FRBSs to realize DSSs has become a most suitable approach that yields powerful fuzzy decision support systems (FDSSs). More detailed review on fuzzy rule-

based system and its design issues are given in the following Section 2.2 and 2.3, respectively.

2.2 Fuzzy rule-based systems

A fuzzy rule-based system (FRBS) is a rule-based system which infers its output based on fuzzy logic [64] instead of Boolean logic. Compared with Boolean logic, fuzzy logic provides a more robust approach for representing and processing information, which more closely resembles human reasoning process. Therefore, fuzzy rule-based systems provide greater power for representing knowledge, more tolerance to uncertainty and imprecision, hence enhanced modeling ability for real-world problems. They have been proven successful in many real-world applications where domain knowledge is imprecise or inexact [65-69].

A fuzzy rule-based system mainly consists of four components illustrated in Figure 1: fuzzifier, fuzzy rule base, fuzzy inference engine and defuzzifier, which are briefly introduced hereafter. For detailed introductions to the fuzzy logic and fuzzy rule-based systems, the readers are referred to [10, 68, 70, 71].

Figure 1 Basic model of a fuzzy rule-based system

Fuzzifier

The fuzzifier enables the fuzzy rule-based system to handle crisp input values, which are mapped to fuzzy sets defined on the input variables. The most widely used fuzzifier is the *singleton fuzzifier*, which converts the crisp input value x' into a *fuzzy singleton* X' , whose membership function is defined as per (1).

$$\mu_{x'}(x) = \begin{cases} 1, & \text{if } x = x' \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

Fuzzy rule base

The fuzzy rule base consists of a set of IF-THEN fuzzy rules, in which each term is associated with a certain fuzzy set in the fuzzy partitions. The two existing types of fuzzy rules that are widely used in real world applications are the Mamdani [72] and Takagi-Sugeno-Kang (TSK) [73]. Examples of these two types of fuzzy rules are:

Mamdani: IF *Speed* IS *fast* AND *Distance* IS *short* THEN *Braking force* IS *strong*

TSK: IF *Speed* IS *fast* AND *Distance* IS *short* THEN *Braking force* IS $f(\text{Speed}, \text{Distance})$

These example rules are for a car braking system, where the variables of *Speed* and *Distance* are input condition and the *Braking force* as consequent. Because the linguistic terms such as *fast* and *short* are much easier to comprehend than the meaningless weights and parameters in ANN [3] and SVM [4], it is one of the major advantages of fuzzy rule-based systems over these black-box models. The biggest difference between Mamdani type and TSK type is that the consequent part of Mamdani rule is a fuzzy set and the consequent part of TSK rule is a linear or non-linear function of the inputs. Generally, the existing TSK-type fuzzy systems provide more accuracy over Mamdani-type ones. However, the consequent parameters of TSK rules are difficult to comprehend, whereas rules in Mamdani fuzzy systems are more interpretable [71].

The linguistic terms in the fuzzy rules are taken from the fuzzy sets in the corresponding fuzzy partitions on the input and output variables. These fuzzy sets differ from crisp sets in that an element may belong to a fuzzy set to a certain degree within the range of [0, 1]. The fuzzy membership function $\mu_F(\cdot)$ computes the membership value $\mu_F(x)$ of a crisp input x , which is the degree to which x belongs to the fuzzy set F . As shown in the example fuzzy rules, there are linguistic terms assigned to these fuzzy membership functions. An example of a fuzzy partition on the variable *Speed* is given in Figure 2. It also shows how the speed of 50km/h is assigned different membership values for the fuzzy set of *slow* and *medium*.

Figure 2 Example of a fuzzy partition on a variable

The commonly adopted fuzzy membership functions, illustrated in Figure 3, are triangular, trapezoidal and Gaussian [68, 74].

Figure 3 Commonly used membership functions (a) triangular, (b) trapezoidal, (c) Gaussian

Fuzzy inference engine

The fuzzy inference engine gives conclusions based on the knowledge represented in fuzzy membership functions and fuzzy rule base. Different fuzzy inference methods can be formed with different composites of a S-norm and an T-norm [10, 75]. For instance, the S-norm MAX and the T-norm MIN are widely used to compose the max-min inference method [76]. The T-norm MIN is used to determine the degree of matching between a sample and the condition part of a fuzzy rule. The S-norm MAX is used to aggregate the fuzzy output of the fuzzy rules into a single fuzzy set. Other operator such as the PRODUCT operator can be used to compute the degree of matching instead of MIN operator, while the SUM operator can be used instead of MAX operator [43, 77, 78]. However, there is no well-established guideline for selecting the proper fuzzy inference operators for a certain problem. In most works, the fuzzy inference operators are predefined by the designers based on experience and preference.

Defuzzifier

After fuzzy inference, the system output is represented by the aggregated outputs of the fuzzy rules. The defuzzification process in the defuzzifier converts the fuzzy outputs into a crisp output. The common choices for defuzzification method include the center of area (COA) and the mean of maximum (MOM) method [10].

Accuracy and interpretability

The construction of a fuzzy rule-based system can be viewed as a task of determining the abovementioned components and their parameters. In particular, the accuracy and interpretability are two important criteria for measuring the performance of a fuzzy rule-based system. However, they are requirements to meet before the fuzzy decision support system can be successfully applied to complex real-world problems.

The accuracy of a FRBS refers to the conformity of the system outputs made by the FRBS to the actual or desired outputs. For classification problems, accuracy of a FRBS can be measured by the percentage of correctly classified instances among total instances. For regression problems, accuracy of a FRBS can be measured by either the mean squared error (MSE) or the square of the Pearson product-moment correlation coefficient (R^2).

In the past, researchers have mainly focused on increasing as much as possible the accuracy of fuzzy rule-based systems, while little attention was paid to their interpretability. However, it is a salient property of the fuzzy rules in a FRBS that they are comprehensible for human beings. Users can easily extract the knowledge represented by a FRBS, and conversely, they are able to incorporate their a priori knowledge or expertise into a FRBS.

Currently, there is no well-established definition for the interpretability of fuzzy rule-based systems because this concept remains subjective. Among various definitions and criteria, the most widely accepted facets of interpretability are as follows [13, 18, 79].

- The fuzzy partition should be formed by valid fuzzy membership functions

- The fuzzy membership functions in the same fuzzy partition should be distinguishable so that comprehensible linguistic terms can be assigned to them. The distinguishability of a fuzzy partition is further studied in Section 4.2.4.
- The fuzzy partition should be complete, which means that each data point on the variable domain should be covered by a least one fuzzy membership function. The completeness of a fuzzy partition is further studied in Section 4.2.4.
- The number of the fuzzy rules should be small enough to be comprehensible.
- The premises of the fuzzy rules should preferably contain only a few input variables so that they are easier to be analyzed.
- The fuzzy rules should not be contradictory, which means the fuzzy rules with the same premise should not lead to different consequents.

Nevertheless, accuracy and interpretability are two contradictory objectives, which means the increase in one criterion usually causes the decrease of the other. In recent years, researchers have shown great interests in designing fuzzy rule-based systems with well-balanced accuracy and interpretability [13, 39, 80, 81]. For synthesizing such a FRBS, the design issues and the possible techniques are reviewed in Section 2.3.

2.3 Design issues in FRBS

The synthesis of an optimal or near-optimal fuzzy rule-based system with well-balanced accuracy and interpretability is an arduous task [12, 13]. As mentioned in Section 2.2 and elaborated hereafter, synthesizing a fuzzy rule-based system amounts to determining its main components.

Until recently, fuzzy rule-based systems were constructed using a direct approach based on the knowledge from human experts [63, 82]. The task becomes difficult when available knowledge is incomplete or the problem is too complex. Furthermore, ad hoc architectures and improperly chosen parameters typically introduce unwanted biases and unavoidably result in suboptimal systems.

Therefore, approaches have been devised for deriving fuzzy partitions and fuzzy rule base from sample data. Moreover, the design effort must be repeated anew for every application problem. Ideally, the decision-making logic should therefore be induced automatically from examples and further optimized for the problem at hand. To achieve this goal, a generic approach is needed that can automatically synthesize an accurate and interpretable FRBS, hence an efficient and dependable FDSS, while requiring minimal or no human effort. Thus, it motivates the development of the automatic approaches that use learning techniques to derive some components in a fuzzy system from data samples. The techniques for facilitating the FRBS design including the neuro-fuzzy synergy [75], clustering algorithms [14], rule learning approaches [83] and evolutionary techniques [31] are briefly reviewed hereafter.

Neuro-fuzzy synergy

Although neural networks are black-box models as mentioned in Section 2.1, they possess the learning capability that is lacking in fuzzy rule-based systems. Furthermore, neural networks offer the potential to reduce the computational time in fuzzy inference by parallelizing the fuzzy operations. Therefore, combining these complementary techniques yields synergetic neuro-fuzzy systems that gain the benefits of both fuzzy systems and neural networks [75, 84].

In a neuro-fuzzy system, the fuzzy system components described in Section 2.2 are implemented in different layers of nodes in a global neural network architecture. The fuzzy operations of fuzzification, fuzzy inference and defuzzification are taken in these nodes. As a simple example, consider a fuzzy system for car braking takes the speed of the car and the distance to the obstacle ahead as inputs and the braking force as the output. The system comprises 2 fuzzy membership functions representing *slow* and *fast* on the input variable of *Speed*, 2 MFs representing *short* and *long* on input *Distance*, and 3 MFs representing *weak*, *medium* and *strong* on output *Braking force*. Its rule base consists of 4 rules as listed in Table I.

Table I Fuzzy rules in the example fuzzy rule-based system for car braking

Rule 1	IF <i>Speed</i> IS <i>slow</i> AND <i>Distance</i> IS <i>short</i> THEN <i>Braking force</i> IS <i>medium</i>
Rule 2	IF <i>Speed</i> IS <i>slow</i> AND <i>Distance</i> IS <i>long</i> THEN <i>Braking force</i> IS <i>weak</i>
Rule 3	IF <i>Speed</i> IS <i>fast</i> AND <i>Distance</i> IS <i>short</i> THEN <i>Braking force</i> IS <i>strong</i>
Rule 4	IF <i>Speed</i> IS <i>fast</i> AND <i>Distance</i> IS <i>long</i> THEN <i>Braking force</i> IS <i>medium</i>

Figure 4 The example of implementing a fuzzy rule-based system as a neuro-fuzzy system

The fuzzy system is implemented into a neuro-fuzzy system architecture and depicted in Figure 4. The nodes Input 1 and Input 2 receive the input *Speed* and *Distance*, respectively. They transmit the crisp input values to the nodes representing the fuzzy membership functions on the corresponding input variables, which serve as the fuzzifier in a fuzzy system. Through linking the fuzzy membership functions on the input and output variables, the nodes Rule 1 to Rule 4 are representing the rules in Table I, which together form the equivalent fuzzy rule base and inference engine. The inference results from the rule base are aggregated and defuzzified in the Output node, which serves as the defuzzifier and gives final system outputs. The fuzzy operations in these nodes are detailed in Section 3.2.

After the combination, the gradient learning and rule learning techniques that are suitable for neural networks can also be applied for synthesizing the neuro-fuzzy system, the details of which are reviewed in the remaining part of this section.

Clustering algorithms

Clustering is the unsupervised classification of patterns into clusters, so that the data points within the same cluster are more similar to each other than points from other cluster [14]. For FRBS design, clustering algorithms are mostly used for generating the fuzzy membership functions that partition the input-output spaces [16, 17, 85]. The clustering algorithms are also adopted to derive the fuzzy rule base, in which each fuzzy rule corresponds to a resultant cluster [19, 20].

One of the most widely used clustering techniques, the Fuzzy C-Means (FCM) clustering algorithm [21] proposed by Bezdek, is taken as a typical example and briefly analyzed. In FCM, the centers of the clusters are computed by iteratively minimizing the objective function (2), where NC is the predefined number of clusters, NS is the total number of sample data points, $sample_i$ is the i th data point, $center_j$ is the center of the j th cluster, μ_j is the membership function for the j th cluster, q is the weighting exponent for determining the amount of fuzziness and $\|sample_i - center_j\|$ is the Euclidian distance between $sample_i$ and $center_j$.

$$J_{FCM} = \sum_{i=1}^{NS} \sum_{j=1}^{NC} \mu_j (sample_i)^q \|sample_i - center_j\|^2 \quad (2)$$

There are three main drawbacks to FCM. First, the number of clusters NC is required to be manually defined by users. For classification problems, users usually directly apply the number of classes as the number of clusters. However, in some problems such as approximation problems, there are no explicit clusters and it is difficult for users to set the number of clusters. Second, there is no guideline for selecting the value of parameter q , yet fuzziness of the resultant fuzzy membership functions can vary when computed by FCM with different values of q . Last, poor clustering results will be obtained if the algorithm starts with inappropriate initial cluster centers.

Besides FCM, many existing clustering techniques [16, 17, 86] rely heavily on the judicious selection of a set of intrinsic algorithm parameters which need to be pre-specified by the designers. Even a slight difference in parameter settings may generate a resultant system that will exhibit a great discrepancy in performance. The parameters in the popular clustering techniques are listed in Table II.

Table II The parameters in the popular clustering algorithms

Clustering Algorithm	List of Parameters
Fuzzy C-Means (FCM)[21]	number of clusters, weighing exponent for fuzziness
Fuzzy Linear Vector Quantization (LVQ) [86]	number of clusters, termination criterion, choice of strategies
Fuzzy Kohonen Partition (FKP) [16]	number of clusters, termination criterion, learning constant, learning width
Pseudo Fuzzy Kohonen Partition (PFKP) [16]	number of clusters, termination criterion, learning constant, learning width
Discrete Incremental Clustering (DIC) [17]	elasticity parameter, tendency parameter, input threshold, output threshold, fuzzy set support parameter

Gradient Learning Methods

Gradient learning methods [15, 22, 23] are optimization algorithms used to improve the performance of the fuzzy rule-based systems through progressively adapting their parameters. Two most popular gradient learning methods are the least mean squares adaptive filter [55] and backpropagation [22]. Gradient learning methods are mostly applied to fine tune the parameters in neuro-fuzzy systems [87], and also to optimize consequent parameters of the fuzzy rules in TSK-type fuzzy rule-based systems [19, 88, 89].

However, because of the complexity in the fuzzy rule-based system synthesis problem, they are prone to be trapped in local optima and consequently limit the performance of the resultant FDSS. Furthermore, the success of gradient learning algorithms also depends on the user predefined parameters, such as learning rate and momentum constant.

Rule Learning Algorithms

Rule learning algorithms are popular methods for deriving the fuzzy rule base. The common trait of these algorithms is that they learn rules by observing the relation between the input variables and the output variables in the available training samples.

The rule learning algorithms proposed in [26] consider all possible combination of fuzzy membership functions as fuzzy rules. The weight of each rule is updated according to the degree of matching of the training samples. The drawback of such rule learning techniques is that they are computationally inefficient and the complexity of the rule base increases drastically as the number of the input variables increases.

Other rule learning techniques incrementally derive rules to cover the training samples [16, 27-29, 90]. First, the fuzzy membership functions that partition the input-output space need to be determined. Then, for each training sample, the rule learning algorithm will find a fuzzy rule in the existing rule base, the premise of which best matches the training sample. If a rule with the same premise cannot be found, the rule will be added. Although these incremental rule learning techniques are computationally efficient, one of their drawbacks is that the result depends heavily on the predetermined fuzzy membership functions. Another drawback is that they derive the rules with the best individual performance without considering the cooperation among the learned fuzzy rules, which is more important to the inference performance [30].

Evolutionary Algorithms

Evolutionary algorithms (EA) are a group of stochastic optimization techniques, the most famous of which include genetic algorithm (GA) [91], genetic programming [92] and evolution strategy [93]. These algorithms are inspired by the biological mechanisms found in natural evolution, including reproduction, selection, crossover and mutation [31]. Besides these canonical EA, other variations include the particle swarm optimization (PSO) [94], ant colony optimization (ACO) [95], differential evolution (DE) [96] and estimation of distribution algorithm (EDA) [97]. Evolutionary algorithms are practically

proven to provide efficient search capabilities for finding optimal or near-optimal solutions in various complex spaces [31-34].

The basic idea of EA is to maintain a population of *chromosomes*, which represent candidate solutions to the problem being solved. For encoding the candidate solutions, there are many different methods, which generally include *binary*, *integer* and *real coding*. The *binary coding* [98, 99] is easy to implement, but limited to a certain interval and resolution. The *integer coding* method treats genes as integer values taken [100, 101]. Both of the binary and integer coding are suitable for solving combinational problems. In the *real coding* method, each gene of the chromosome is a real number [102] and thus suitable for the problems in the continuous domain, such as tuning the parameters in fuzzy membership functions. Its advantage over *binary coding* is the increase in precision and a shorter chromosome length.

After determining the encoding method, the initial population is generated randomly or heuristically. During the iterations called *generations* in the evolutionary process, the chromosomes are evaluated for their *fitness values* as solutions to the search problem using a provided *fitness function*. A predefined selection scheme will be used to select the chromosome to participate in subsequent evolution according to their fitness values. The popular selection schemes include *fitness proportional*, *ranking* and *tournament selection*, a comparative analysis of which can be found in [103, 104]. With the fitness function and selection scheme, a selection pressure is achieved to progressively improve the overall fitness in the population. Those chromosomes with higher fitness values are more likely to be evolved into the next generation and those with low fitness are more likely to disappear.

The selected chromosomes act as parents and generate offspring through genetic operators including *reproduction*, *crossover* and *mutation*. Through reproduction, the good solutions represented by selected chromosomes are copied into the next generation. The crossover operator tries to generate better offspring by recombining the genetic material of the parents. The popular crossover operators include *one point crossover* [34],

multi-point crossover [105, 106], *uniform crossover* [107], and *arithmetic crossover* [107, 108]. Through changing the genes in the chromosomes, the mutation operators help to avoid local optima by preventing the chromosomes in the same population from becoming too similar to each other. Through crossover and mutation, evolution algorithms efficiently exploit the currently nearby space and explore the unseen space. Similar operators for achieving the exploitation and exploration are found in PSO [94], ACO [95] and EDA [97].

The solutions in the population are improved through generations and the evolution process continues until either the search goal or the number of maximum generation is reached.

Compared with non-evolutionary techniques such as clustering and gradient learning techniques, evolutionary algorithms exhibit various advantages. First, they usually require minimal domain knowledge and fewer assumptions about the characteristics of the search space. Second, they are less prone to be trapped in local optima. Third, they are less sensitive to their parametric settings. Finally, they can achieve balanced exploitation and exploration. Exploitation means obtaining even better solutions by searching the space nearby the good ones. Exploration means discovering better solutions in the unseen search space by introducing new genetic materials. With these advantages, evolutionary algorithms can be efficiently applied as robust and generic methods for synthesizing a fuzzy rule-based system regardless of the application domain [36]. The representative evolutionary-based methods adopted in the process of synthesizing a FRBS are reviewed in Section 2.4.

However, evolutionary algorithms also have their own limitations, including premature convergence and poor fine tuning [109]. Premature convergence happens when a sub-optimal individual is chosen to crossover many times and the similar individuals finally take over a large proportion of the population. To overcome this problem, approaches such as re-initialization [110] and niching [111] are adopted in EA to preserve information across a diverse population. Although genetic algorithms are proven to be

efficient in finding hills in the fitness landscape, it is not easy for them to reach the top of the hills because of their poor fine tuning ability. Adopting arithmetic crossover in real coded population [112] or gradually increasing the tuning resolution over generations [101] can help to increase the fine tuning ability. Another approach is to combine the EA with other techniques to form some hybrid methods. One example is the memetic algorithm (MA), which incorporates one or several heuristic local search procedures to improve the solutions generated by the EA [113, 114].

As a further enhancement of the canonical evolutionary algorithm, coevolutionary algorithms mimic the ecological model in nature and broaden the use of evolutionary algorithm. They provide suitable and efficient approaches for solving complex problems by dividing the difficult search task into several easier subtasks [50, 115, 116]. For instance, the task of learning a cascade network is divided into the subtasks of tuning the parameters in each neuron, which are coevolved in populations [50]. Similarly, the task of deriving a FRBS is divided into the subtasks of coevolving fuzzy membership functions and fuzzy rule base in two populations respectively [49]. Further study on coevolutionary algorithms is given in Chapter 5.

2.4 Evolutionary Fuzzy Systems

The learning and tuning of a fuzzy rule-based system can be formulated as a search and optimization problem in a high-dimensional hyperspace over the parameters in the FRBS. Such hyperspace is typically large since there are many parameters in a FRBS. It is also non-differentiable because the changes in the number of fuzzy membership functions and the combination of fuzzy rules are discrete. Lastly, the hyperspace is complex since similar membership functions and fuzzy rule base may have result in quite different performances.

Based on above characteristics, evolutionary algorithms are suitable search methods because they are not based on gradient search and are less prone to local minima entrapments. The synergy of fuzzy rule-based system and evolutionary algorithm, called evolutionary fuzzy systems, has thus yielded encouraging results [36, 117, 118].

Hereafter, the applications of evolutionary-based techniques in different steps of synthesizing a fuzzy rule-based system are briefly reviewed.

2.4.1 Evolutionary approaches for determining fuzzy membership functions

Because the fuzzy rules consist of the fuzzy sets, these fuzzy sets will greatly affect both of the inference result and interpretability of the whole system. In this section, evolutionary-based approaches for determining the number of membership functions (partition granularity) and learning the shapes and positions of membership functions are reviewed.

The pioneering work of using a GA to tune the fuzzy membership functions in a fuzzy system can be found in [119]. The parameters of the triangular fuzzy MFs in the fuzzy rules are encoded into the chromosomes. However, this method would result in a less interpretable rule base because each rule uses its individual, locally defined membership functions. Chang and Chen proposed a new encoding method, which is parameter-based, to diminish the redundant representation of fuzzy parameters [120]. They only encode the membership functions of each dimension. These membership functions would be used and shared in the fuzzy rules.

Murata and Ishibuchi proposed an evolutionary approach for determining the fuzzy partitions [38]. They evenly divided the input space into several intervals. If the peak of a triangular MF occurs at the i th point, the i th bit of the chromosome will be 1, otherwise 0. An example of this approach is shown in Figure 5. Similar methods to determine the fuzzy partitions could be found in [121, 122]. Wong and Chen [123, 124] adopted an approach similar to [38], in which asymmetric Gaussian MFs are used instead of triangular ones.

One main advantage of these methods is that they will automatically solve the problem of deciding the number of MFs with simple binary coded chromosomes, without explicitly encoding the number of MFs. However, one of the main drawbacks in these methods is that it is not easy for the user to set a proper interval on the input variables.

Figure 5 Evolving the fuzzy partition on the input space by using binary coded chromosomes

On the contrary, Furuhashi et al. learned the fuzzy partition on each variable by GA, with the numbers of MFs directly encoded as integers in the chromosomes [125, 126]. However, the evenly distributed fuzzy partition in this method may not be suitable for the data distribution. Therefore, fine tuning these initial fuzzy partitions is required for further increasing the performance of the final system.

The most common evolutionary-based method for fine tuning the fuzzy membership functions is encoding and evolving the parameters of MFs in chromosomes. A multi-resolutional dynamic genetic algorithm, the tuning resolution of which gradually decreases over time, is adopted for the task of fine tuning the center and width of Gaussian membership functions in [101]. Wong and Her utilized a GA to tune the triangular MFs on the input variables [127], with a fuzzy set elimination scheme as in [128]. In [129], each chromosome in the genetic population encodes centers and widths of the symmetric triangular membership functions on input and output variables.

In the various approaches discussed above, the GA is applied to decide the number of MFs and tune the parameters of MFs, while the other components are fixed. To increase the accuracy and maintain the interpretability of the final FRBS, some more sophisticated evolutionary approaches [39, 130-132] are devised to evolve the type, the nonlinear scaling factors and the linguistic hedges of the membership functions, the details of which are reviewed in Section 2.4.5.

2.4.2 Evolutionary approaches for rule learning

After determining the fuzzy partitions on each variable, fuzzy rule-based system designers will come to the vital stage of formulating the fuzzy rule base. Various methods are devised to generate fuzzy rules, as extensively discussed in the survey in [83]. In many recent researches, evolutionary algorithms have been successfully applied to learn fuzzy rule base during the evolution. In this section, three basic evolutionary approaches for learning the fuzzy rule base are reviewed, including Michigan [133], Pittsburgh [134] and Iterative Rule Learning (IRL) [41] approaches.

In the Michigan approach, the chromosomes are individual rules and a rule set is represented by the entire population [133]. Each fuzzy rule in the population functions as a classifier. The collection of rules is modified over time via interaction with the environment. However, one drawback of this approach is that it cannot directly optimize the fuzzy rule base because it only evaluates the fitness of each individual rule. Comparing to Pittsburgh and IRL approach, only a few works based on Michigan approach are proposed [43, 135].

The general outline of the fuzzy rule-based systems learned by Michigan approach is as follows: 1) generate an initial population of fuzzy rules; 2) evaluate each fuzzy rule in the current population; 3) generate new fuzzy classification rules by evolutionary algorithms; 4) replace a part of the current population with the newly generated rules; 5) terminate the algorithm if a stopping condition is satisfied, otherwise return to step 2.

In the Pittsburgh approach, the fuzzy rules in a whole rule base are encoded and concatenated in to a chromosome [134]. Crossover operator serves to provide a new combination of rules and mutation operator provides new rules. When fixed-length chromosomes are used to represent the fixed-length rule base, the mapping between the chromosome and the rule base is also fixed. In some research works, variable-length rule bases are used, employing modified genetic operators for dealing with these variable-length and position independent genes. Compared with the other two learning approaches, Pittsburgh approach is more popular [40, 106, 124, 128, 136]. That is because the Pittsburgh approach directly optimizes the whole rule base and it is thus more likely to obtain a fuzzy rule-based system presenting good cooperation among the fuzzy rules. But this approach has to deal with very large search spaces, which makes it difficult to find optimal solutions.

In Iterative Rule Learning (IRL) approach, each chromosome in the population represents a single rule as in the Michigan approach. But the difference with the Michigan approach is that only the best individual is selected to form part of the solution. In IRL, evolutionary algorithm provides a partial solution to the problem. Contrary to both previous approaches, evolutionary algorithm is run several times to obtain the complete rule base. In order to learn a rule base with IRL approach, the evolutionary algorithm is embedded into an iterative scheme similar to the following steps: 1) use evolutionary algorithm to obtain a rule for the system; 2) incorporate this rule into the final set of rules; 3) penalize this rule; 4) if the rule set generated till now is adequate to solve the problem, end and return it as the solution. Otherwise, return to step 1.

Research work adopting the IRL approach can be found in [41, 42, 44, 137-139]. However, in the IRL approach, the fitness of each chromosome is evaluated individually, without taking account its cooperation with the other chromosomes in the population. So, it is possible that some rules in the final rule set are very similar. To simplify the rule base, the process of finding and removing the redundant rules must be incorporated. The implementation of an IRL approach is more complicated than the previous two approaches.

Besides the basic evolutionary rule learning methods, a method hybridizing the Michigan and Pittsburgh approach was proposed in [140, 141]. It is mainly based on Pittsburgh approach to evolve the whole fuzzy rule base. The rule generation mechanism of Michigan approach is used as a mutation operation in this hybrid algorithm. With this mutation, the worst rule in a rule set is replaced with new rule that are generated from good rules in that rule set. Experimental results show that the hybrid method outperforms the non-hybrid methods.

Furthermore, evolutionary algorithm is also adopted in rule selection in order to address the complexity of fuzzy rule base [77, 142-147]. The rule selection is generally conducted after all possible fuzzy rules are learned. It is applied to select the best subset of fuzzy rules via incorporating the objective of maximizing the system accuracy and minimizing the number of rules in the fitness function.

2.4.3 Evolutionary approaches for selecting fuzzy inference operators

As mentioned in Section 2.2, the fuzzy inference operators are usually selected manually by the designer in most of the existing literature. However, few methods can be found that use evolutionary algorithms to determine the fuzzy inference operators.

Similar approaches were proposed to apply GA to select both of the composite operators and fuzzy relations under that fuzzy inference method at the same time from the given sample data [148-150]. Cordon et al. tested the possibility of adopting alternative fuzzy inference operators in a fuzzy rule-based system [151]. In this work, genetic learning process is applied to obtain the best parameters in the fuzzy inference operators for a certain problem. Results showed that the evolutionary determination of fuzzy inference operators helps to improve the accuracy of FRBS. However, because some complicated fuzzy inference operators such as the “Quasiarithmetic mean” in [151] are used, the interpretability of the FRBS may be undermined.

2.4.4 Evolutionary approaches for determining MFs and learning rules

In the Section 2.4.1 and 2.4.2, the basic evolutionary approaches of determining fuzzy membership functions alone and learning fuzzy rules only are reviewed, respectively. In this section, the evolutionary-based approaches for learning both of the fuzzy membership functions and the fuzzy rule base are categorized in three general approaches as shown in Figure 6, the details and examples of which are reviewed hereafter.

Figure 6 Illustrations of different evolutionary approaches for learning both fuzzy membership functions and rule base in the FRBS

Many researchers applied the process of determining membership functions and the process of fuzzy rule generation in different stages as shown in Figure 6 (a). In some works, EA is applied in both of the two learning processes. In [152-154], evolutionary algorithm is first applied to learn a initial fuzzy rule base, after which another evolutionary-based learning process is applied to fine tune the fuzzy membership functions to achieve a better performance.

Instead of applying an EA in both learning processes, other learning techniques can be applied in either learning process so as to further improve the performance of the

evolutionary-based learning approach. One example is the two-step fuzzy modeling approach proposed by Setnes and Roubos [18, 19]. First, a clustering algorithm is applied to obtain an initial rule base. The consequents of the TSK rules are then determined using the least-squares estimate. Then, the performance of the initial rule base is optimized by using a real-coded GA, allowing for simultaneous optimization of both the rule antecedents and consequents.

In other works, evolutionary algorithm is only applied for evolving the fuzzy membership functions or fuzzy rule base. According to the fuzzy membership functions represented in each chromosome, other learning techniques are applied to derive the corresponding fuzzy rule base, or vice versa, before the fitness evaluation. This group of approaches is depicted in Figure 6 (b). Cordon et al. applied a GA to determine number of labels for each variable and constrained triangular MFs at the same time [155]. An ad hoc data-driven method is adopted to derive the fuzzy rule base according to the fuzzy partitions extracted from each chromosome. Then the fitness values of the resultant systems can be evaluated. Based on their fitness, the evolutionary algorithm continues to evolve the fuzzy partitions. In another example, Nelles et al. learn the fuzzy rule base with a GA in learning process 1 of Figure 6 (b) and tune MFs with gradient optimization techniques in learning process 2 [156].

By handling the MFs generation and rule learning separately, the approaches shown in Figure 6 (a) and (b) deal with much smaller search space. Furthermore, the hybridization with other techniques, especially the fine-tuning techniques, helps to improve the search performance of GA. However, some researchers argued that the MFs and fuzzy rules should be determined simultaneously by evolutionary techniques, as shown in Figure 6 (c), because they are strongly interdependent and together comprise the major part of a fuzzy rule-based system [157].

Three design tasks of a TSK fuzzy system, including determining membership functions, number of rules and rule consequent parameters, are integrated into the evolutionary process in [128, 158]. The drawback of these approaches is that the length of the

chromosomes and the number of rules will increase dramatically with the number of inputs. Wong and Lin adopted a similar approach as in [128], except that they used Gaussian MFs instead of triangular ones [159]. Therefore, similar drawback can be found in this approach.

Wang et al. simultaneously learned the fuzzy rule base and membership functions by using a GA with variable-length chromosomes, in each of which two substrings are representing fuzzy rule base and MFs respectively [46]. The chromosomes are of varying length because they encode different number of fuzzy rules. To deal with these variable-length chromosomes, special crossover and mutation operators are devised to evolve these two substrings simultaneously. Similarly, Park et al. used two unrelated substructure in the chromosomes to learn the MFs and rules of a Mamdani-type fuzzy system [160].

Two main drawbacks can be found in the approaches shown in Figure 6 (c). One is that the longer chromosomes are needed to encode the possible solutions. Thus, the search space is enlarged and becomes much more complex. Another drawback is that, in some cases, different encoding method and special genetic operators must be applied to deal with the MFs and fuzzy rules encoded together in a chromosome. Therefore, efficient evolutionary algorithms with proper encoding methods are crucial to the success of the approaches in Figure 6 (c).

2.4.5 Special evolutionary learning or optimizing methods for FRBS

In the previous sections, most reviewed research works adopt classical approaches of applying evolutionary algorithms in fuzzy rule-based system synthesis. In this section, novel evolutionary approaches are discussed to outline the recent research trends.

Special evolutionary approaches for determining the fuzzy partitions

Although most works learn the fuzzy partitions solely by evolving the parameters of the membership functions, some works try to increase the accuracy and interpretability of the

final FRBS by selecting the type of the MFs or evolving the scaling factors and linguistic hedges of MFs.

Shi et al proposed an evolutionary method for synthesizing a fuzzy system in [161]. Each membership function in the fuzzy system has its own type, which can be chosen among 6 types (triangle, left triangle, right triangle, Gaussian, sigmoid, reverse sigmoid). The size of rule base, the type and parameters of each MF, and possible fuzzy rules are encoded into a chromosome and automatically determined by using GA. The advantage of this method is that it can reduce the chance of selecting an inappropriate type of MFs. The aggregation of several basic-shaped MFs results in more flexible fuzzy partitions which can capture the underlying data distribution exhibited by the numerical observations.

Casillas et al aimed at proposing a method to automatically obtain Mamdani-type fuzzy models with a balanced trade-off between interpretability and accuracy [39]. They defined the rules as “surface structure” and the rules with MFs as “deep structure”. The surface structure is adjusted with linguistic hedges while the rest of the deep structure is tuned by changing the membership functions and using non-linear scaling factors. The advantage of using linguistic hedges and nonlinear scaling factor is to avoid an excessive interpretability loss during the membership function tuning. However, they still increase the difficulties for understanding the rules. On the other hand, the membership functions are adjusted within the predefined intervals so as to avoid obtaining excessively deformed fuzzy partitions. However, one drawback of this method is that the predefined number of membership functions and intervals may not be suitable for the data distribution. In the chromosomes, the first two parts use real coding to represent the parameters of the MFs and the nonlinear scale factors. In the third part, the linguistic hedges are encoded as integers. Therefore, different crossover operators are used for these genetic segments. Two parents will generate eight offspring, among which the best two will be added to the next generation. However, this crossover operation needs higher computational cost than other crossover operators when generating same number of offspring because more fitness evaluations are required.

A genetic approach for learning accurate and compact fuzzy systems based on the 2-tuples linguistic representation is proposed in [132, 162]. In each chromosome, the first part is a string of integer numbers, which evolves the number of MF label for each input variable. Encoded as a string of real numbers, the second part evolves the lateral variation, which is horizontal offset within two nearby MFs. After the fuzzy partitions are decoded, the Wang-Mendel rule learning method [29] is adopted to generate the fuzzy rule base. In [131], the authors further extended their 2-tuples representation to 3-tuples. In the newly added part of the chromosomes, the amplitude variations of the support of the triangular MFs are tuned. While being fixed in [132, 162], the support amplitudes of the triangular MFs vary among different MFs in this approach. Therefore, this approach is able to generate more flexible and suitable fuzzy partitions according to the distribution of the sample data, which consequently increase the system accuracy. However, one main drawback is that no mechanism is adopted to prevent this method from generating incomplete or indistinguishable fuzzy partitions. In all these works, the authors adopted a restart scheme to help the evolutionary process escape from local optima.

Control genes in evolutionary algorithms

Besides evolving the parameters of the MFs and composition of the fuzzy rules, some evolutionary approaches also handle the membership function selection and rule selection by using control genes.

A fuzzy rule-based system is learned by using a Pittsburgh approach with clause selection and rule selection techniques integrated [47]. The MFs on each dimension and maximum of 50 fuzzy rules are encoded in a chromosome. For each fuzzy rule and each IF-part clause, there is a corresponding control gene indicating whether the rule or clause is activated. Thus, the approach learns the number, position and parameters of the input MFs together with the number and composition of the rules.

For evolving TSK fuzzy models, Kim et al. adopted an encoding scheme in which a chromosome consists of three parts [163]. By using control genes, the first and second part of the chromosome mainly deal with the rule selection and the membership function

selection, respectively. The rest part is used to tune the parameters of the Gaussian MFs. For maintaining the interpretability of the fuzzy model, the criteria of the membership functions distribution including completeness and distinguishability are considered in the fitness evaluation.

A tabular scheme is used in genetic algorithm for constructing a fuzzy rule-based system in [20]. In the genetic representation, each row represents a membership function, in which the first gene is the “inclusion-flag”, the second gene selects its type and the remaining genes encode its parameters. In the representation of fuzzy rule base, the first row directly encodes the number of selected fuzzy rules and each of the following rows encodes a fuzzy rule. For instance, if the number in the first row is 5, the first 5 rows of encoded fuzzy rules form the whole fuzzy rule base. When the number of selected fuzzy rules is changed, the fuzzy rules are inserted or deleted. One drawback of this approach is that the designer needs to predefine the number of candidate rules generated with the learning vector quantization clustering algorithm [164], which are used for insertion if needed.

As reviewed above, the evolutionary approaches adopting control genes can help achieve compact fuzzy systems through automatically determining the number of membership functions and fuzzy rules.

Advanced evolutionary approaches with multiple populations

Although most of these works handle the learning task in one genetic population, some researchers explored the feasibility of using advanced evolutionary algorithms with multiple populations for synthesizing the fuzzy rule-based systems in a more efficient way, as reviewed hereafter.

A parallel GA [165, 166] is applied to learn FRBSs by evolving multiple populations, in each of which fuzzy systems with different number of input variables, membership functions and fuzzy rules are simultaneously evolved and optimized [167]. The chromosomes are occasionally migrated to other populations, which improves the search

efficiency by exchanging genetic materials. In [100], the authors learned fuzzy systems by using a parallel GA with three populations, each of which encodes fuzzy partitions with different number of fuzzy membership functions and occasionally exchanges chromosomes.

The main advantage of adopting parallel GA is that the migration among the populations helps to overcome stagnation and increase the search efficiency [166]. Furthermore, the computational cost is distributed in different populations. However, the settings of the migration, such as how many migrants are to be selected and how often the migration takes place, need to be carefully devised to achieve a good performance [167].

While parallel GAs evolve the complete solutions in different populations, the cooperative coevolutionary genetic algorithms divide the complete solutions into several components and evolve them in corresponding populations. In this way, the difficult task of searching a complete solution is divided into easier subtasks of finding the components of the solution [50, 116]. As pointed out in [51], proper decomposition of a search task is important to the efficiency of the coevolutionary algorithm. The task of synthesizing a FRBS is divided into the subtasks of deriving the components in the FRBS, and the examples of which are reviewed hereafter. Detailed discussion on coevolutionary algorithms is given in Section 3.4.

A special cooperative coevolutionary fuzzy modeling approach called “Fuzzy CoCo” is proposed in [49]. The fuzzy membership functions and fuzzy rules are coevolved in two genetic species, respectively. Chromosomes are selected from both species to form a fuzzy rule-based system. The fitness value of the chromosomes depends on the performance of the fuzzy systems it participated in. In a similar work, a Hierarchical Genetic Algorithm (HGA) is adopted to optimize the membership functions and another population to coevolve the fuzzy rules [168]. In the HGA, a chromosome consists of two parts: the binary coded genes in the first part are control genes that determine whether a corresponding membership function is selected; the real coded genes in the second part represent the parameters of the membership functions.

Delgado et al. proposed an approach to learn fuzzy system by using a hierarchical coevolutionary genetic algorithm (HCGA) [48, 169, 170]. It is claimed that, when more complex design decisions must be made, the representation of the complete solution by only one chromosome can be inadequate. It would be more appropriate to optimize a large set of parameters encoded at different levels, producing a HCGA. In this method, all the parameters in a FRBS, including the membership functions, fuzzy rules, and antecedent aggregation operator are adjusted during evolution. The evolutionary process is hierarchically structured in modules of 4 levels: MFs at the first level, the population of individual rules at the second level, the population of fuzzy rule bases at the third level and population of fuzzy systems at the fourth level. In the first level, real coding is used to represent the trapezoidal, triangular, or Gaussian MFs. In the second and third level, integer coding is used for individual rules and rule bases. In the fourth level, real coding is used to encode the T-norm parameters and integer coding for inference operator, rule base and fuzzy partition set. The consequents of all the TSK rules are tuned by a least squares method.

In summary, reviewing the works above led to three key findings. First, the learning of optimal membership function is still one of the major focuses in FRBS synthesis, because the MFs affect both of the accuracy and interpretability. Furthermore, more flexible MFs are explored to reduce the limitation of predefining the MF type. Second, the use of control genes helps to construct a concise FRBS. Third, advanced evolutionary techniques such as coevolutionary algorithms provide efficient methods for synthesizing a FRBS because they can divide the complex optimization task into several subtasks which are evolved in corresponding populations. These three findings can be incorporated into a systematic approach for synthesizing a FRBS with accuracy and interpretability, which is the objective of our research.

Chapter 3

Hierarchical Coevolutionary Fuzzy System Synthesis

3.1 Overview

After reviewing the problems addressed and some closely related research works in Chapter 2, this chapter provides an overview of the newly proposed hierarchical coevolutionary approach called HiCEFS for the synthesis of a fuzzy decision support system. The main components of HiCEFS are elaborated in subsequent chapters.

In the proposed HiCEFS approach, the synthesis of a fuzzy decision support system is considered as a search and optimization problem, which is solved by a hierarchical coevolutionary genetic algorithm (HCGA). Similar to other approaches that utilize evolutionary algorithms as learning techniques, the most important issue is the transformation of the problem being solved into an appropriate genetic representation, which directly affects the effectiveness and performance of the genetic search. Therefore, an appropriate representation is devised to represent not only the components of a fuzzy rule-based system in chromosomes from different populations but also the proper relations between these components.

By using a special representation in the HiCEFS, the task of synthesizing a fuzzy decision support system is divided into the several subtasks at different levels. At the highest level, the synthesis of a FDSS is considered as the search of the good combination of two important components, the fuzzy partitions and the fuzzy rule base. For evolving the fuzzy partitions, this subtask can be further divided into the subtasks of optimizing the fuzzy partition on each input variable and searching the proper combination of the evolved fuzzy partitions. Similarly, the subtask of generating the fuzzy rule base can be divided into the subtask of evolving the possible individual fuzzy rules and searching the proper combination of these rules as a fuzzy rule base.

The coevolved populations in the HCGA handle these tasks according to this divide-and-conquer approach, an example of which is evolving the FDSS for car braking in Section 2.3, as illustrated in Figure 7. In the HCGA, there is a population for evolving a whole FDSS, which selects the proper combinations of the fuzzy partitions and the fuzzy rule base. Part (a) of Figure 7 shows a population for evolving the combinations of the fuzzy partitions and the populations for evolving the membership functions on each input variable. Part (b) shows a population for evolving the possible fuzzy rules and a population for evolving the fuzzy rule base through selecting the combination of the fuzzy rules in previous population.

Figure 7 The relation between the components of the FDSS and hierarchical coevolutionary genetic algorithm in the HiCEFS approach

Proper genetic operators are specifically selected for different type of chromosomes used in these populations. A fitness evaluation and assignment scheme is devised to correctly represent the users' objectives and assign proper fitness values to the chromosomes from different populations.

In the subsequent sections of this chapter, the architecture and the salient features of the HiCEFS are introduced. The details of the HiCEFS approach for synthesizing a fuzzy decision support system are discussed in Chapter 6.

3.2 Architecture of the FDSS synthesized by HiCEFS

As shown in Figure 8, the fuzzy decision support system synthesized by the HiCEFS approach is implemented as a neuro-fuzzy system with five layers of nodes, which is similar to the architectures adopted by POPFNN [16] and GenSoFNN [17]. In such architecture, the FDSS functions as a Mamdani-type fuzzy rule-based system, adopted because its rules are more interpretable than that of a TSK type. Furthermore, the advantage of adopting the neuro-fuzzy architecture is that all important components of the FDSS are separated and can be easily replaced no matter whether they are manually predefined or derived by learning techniques. This architecture thus forms a versatile platform for both research experiments and practical applications.

Figure 8 The architecture of FDSS derived by the HiCEFS approach

In the neuro-fuzzy architecture, the five layers from layer 1 to layer 5 are also named *input layer*, *condition layer*, *rule layer*, *consequent layer* and *output layer*. Each layer consists of a set of nodes which performs a specific fuzzy operation. The number of the nodes in each layer is determined as follows. Assume that $n1$ variables are preselected as the input variables and $n5$ is the number of output variables, the inputs of the FDSS can be represented by the vector $X = [x_1, x_2, \dots, x_i, \dots, x_{n1}]$ and the vector $Y = [y_1, y_2, \dots, y_z, \dots, y_{n5}]$ denotes the outputs of the FDSS with respect to the input vector X . Thus, the number

of nodes in layer 1 and layer 5 are $n1$ and $n5$, respectively. Because each input variable may have different number of fuzzy membership functions, the total number of nodes in layer 2, denoted as $n2$, can be computed using (3), where J_i is the number of fuzzy membership functions on the i th input variable. Similarly, the number of nodes in the layer 4, denoted as $n4$, can be computed using (4), where K_z is the number of fuzzy membership functions on the z th output variable. The $n3$ denotes the number of nodes in the *rule layer*, which varies according to the dataset and the rule learning method adopted.

$$n2 = \sum_{i=1}^{n1} J_i \quad (3)$$

$$n4 = \sum_{z=1}^{n5} K_z \quad (4)$$

The fuzzy operations in each layer of HiCEFS are described and formulated as below.

1) Input layer

Layer 1 is the *input layer* of the FDSS and serves as the *fuzzifier*. It consists of the *input linguistic* nodes denoted as I_i , each of which represent represents a certain input linguistic variable, such as “*Speed*” or “*Distance*”. Each I_i receives the crisp input value and directly transmits the equivalent singleton fuzzy set to the nodes that it links to in the *condition layer*. The net input Net_i^1 and net output Z_i^1 of the node I_i are given in (5) and (6), where x_i is the i th element of the input vector X .

$$Net_i^1 = x_i \quad (5)$$

$$Z_i^1 = Net_i^1 \quad (6)$$

2) Condition layer

Layer 2 represents the fuzzy partitions on the input variables and consists of a set of *condition* nodes. Each of them contains a fuzzy membership function which is related to a certain linguistic term, such as “*fast*” or “*short*”, and forms the condition part of the fuzzy rules. The node representing the j th linguistic term on the i th input variable is denoted as $IL_{i,j}$. The net input $Net_{i,j}^2$ and net output $Z_{i,j}^2$ of the node $IL_{i,j}$ are given in (7) and (8), where $\mu_{i,j}(\cdot)$ is the j th fuzzy membership function on the i th input variable.

$$Net_{i,j}^2 = Z_i^1 \quad (7)$$

$$Z_{i,j}^2 = \mu_{i,j}(Net_{i,j}^2) \quad (8)$$

3) Rule layer

Layer 3 encodes the fuzzy rule base of the FDSS. Each *rule* node R_m in the *rule layer* represents a rule by linking the *condition* nodes as the IF part to the *consequent* nodes in *consequent layer* as the THEN part. For instance, each link from the condition node $IL_{i,j}$ to the rule node R_m represents that the j th linguistic term of i th input variable exists in the IF-part of the m th Mamdani rule in the rule base. No more than one IL condition node of the same input variable will be linked to the same rule node. The links from R_m to consequent nodes OL represent the consequent part of the m th rule. Therefore, each rule node R_m models a Mamdani fuzzy rule in the rule base.

The net input Net_m^3 and net output Z_m^3 are given in (9) and (10), where $Z_{i,j}^2$ is the output of the *input-label* node $IL_{i,j}$ that forms the premise part of the m th fuzzy rule, OP_T is the T-norm operator. Because MIN operator is used in this work, (9) can be converted to $\min(Z_{i,j}^2)$, which calculates the minimum of all the membership values on the i th input variable. The output Z_m^3 represents the degree of matching between the current input instance and the m th rule.

$$Net_m^3 = OP_T [Z_{1,j}^2, \dots, Z_{i,j}^2, \dots, Z_{n1,j}^2] \quad (9)$$

$$Z_m^3 = Net_m^3 \quad (10)$$

4) Consequent layer

Layer 4 is used to represent the consequent part of the fuzzy rules. A *consequent* node $OL_{z,k}$ represents the k th linguistic term of the z th output variable. The net inputs $Net_{z,k}^4$ and net output $Z_{z,k}^4$ of the node $OL_{z,k}$ are given in (11) and (12), where OP_S is the S-norm operator, $\phi_{m,z,k}$ is the factor representing the link between the nodes R_m and $OL_{z,k}$.

Because MAX operator is used in this work, (11) is equivalent to $\max_m(\varphi_{m,z,k} Z_m^3)$ and computes the maximum of the degree of matching among all the rules whose consequent of the z th output variable is the k th label.

$$Net_{z,k}^4 = OP_S[\varphi_{1,z,k} Z_1^3, \dots, \varphi_{m,z,k} Z_m^3, \dots, \varphi_{n3,z,k} Z_{n3}^3],$$

$$\text{where } \varphi_{m,z,k} = \begin{cases} 1 & \text{if OL}_{z,k} \text{ is one of the consequent of } R_m \\ 0 & \text{otherwise} \end{cases} \quad (11)$$

$$Z_{z,k}^4 = Net_{z,k}^4 \quad (12)$$

5) Output layer

Layer 5 gives the final system outputs through the defuzzification of the fuzzy inference results. Each *defuzzification* node O_z computes the final output of a certain output variable from the outputs of the fired rules. The net input Net_z^5 and net output Z_z^5 are given in (13) and (14), where $v_{z,k}^4$ is the centroid of the k th linguistic term and K_z is the total number of linguistic terms on the z th output variable.

$$Net_z^5 = \frac{\sum_{k=1}^{K_z} v_{z,k}^4 \times Z_{z,k}^4}{\sum_{k=1}^{K_z} Z_{z,k}^4} \quad (13)$$

$$Z_z^5 = Net_z^5 \quad (14)$$

3.3 Irregular-Shaped Membership Function

The fuzzy membership functions (MF) on the input-output spaces together with the fuzzy rule base (FRB) encapsulates the expert knowledge in a fuzzy rule-based system. Thus, they directly affect the fuzzy rule-based system's modeling accuracy and interpretability.

The fuzzy membership functions on the input variables provide a mapping of actual input values into the fuzzy membership values. The accuracy of this mapping is vital to the subsequent fuzzy inference process and the final system outputs. The construction of the best descriptive membership functions is the most time-consuming phase in FRBS design, where characteristics including the type of MF to be used, the number of MFs on each input variable, the shape of the MFs have to be decided.

Fuzzy membership functions can be derived either from expert knowledge or directly from data. Deriving fuzzy membership functions from expert knowledge is the traditional top-down way of designing expert rule based system in artificial intelligence. However, these manually derived MFs remain subjective and may not adequately cover the entire input and output decision space, causing a loss of accuracy in the resultant fuzzy rule-based systems. Taking the example of a fuzzy rule-based system for determining the car braking force mentioned in Section 2.3, one designer may allocate 3 fuzzy membership functions including *slow*, *medium*, *fast* on the variable of *Speed*, while another may use 5, namely *very slow*, *slow*, *medium*, *fast* and *very fast*. Even if both of the designers adopt a same number of MFs on a certain variable, it is very likely that the shapes and positions of these MFs differ greatly. Such MFs usually need to go through a tedious trial-and-error tuning process before the optimal or near-optimal MFs are found. Addressing this issue is thus the main motivation for automatically and objectively inducing MFs from data and reducing the dependence on subjective interpretation.

Clustering algorithms are the most popular methods to derive fuzzy membership functions from data. The various clustering algorithms currently in use as reviewed in Section 2.3. While useful, the main drawback of these approaches is that they require much prior knowledge. Many crucial parameters such as the number of clusters and the learning constant listed in Table II in Section 2.3 need to be predefined. The MFs derived by these algorithms are sensitive to the choice of these parameters. Furthermore, clustering algorithms usually generate the MFs using a certain kind of metric, such as the J_{FCM} in (15) for Fuzzy C-Means, which is not directly related to the accuracy of the resultant FRBS. These deficiencies greatly limit the performance and adaptability of clustering algorithms.

Evolutionary techniques such as genetic algorithms have also been utilized to derive the fuzzy membership functions. As reviewed in Section 2.4.1, approaches can be found that use genetic algorithms to generate a priori fuzzy partitions with triangular MFs [38, 122], to fine tune the initial fuzzy partitions [45] [19, 171], to simultaneously learn MFs and fuzzy rules [46, 128, 160], or else to learn MFs and fuzzy rules in different stages [105,

152]. In comparison to clustering algorithms, evolutionary algorithms generate and tune the MFs directly according to the performance of the resultant fuzzy rule-based system.

Besides the approach for deriving the MFs, the type of MF is another factor that needs to be determined by the designer. Being simple and computationally efficient, triangular, trapezoidal, or Gaussian-shaped MFs have been widely adopted by fuzzy system designers to cover the input-output space. In early works, these MFs were symmetrical [45, 158, 160], while in later works, fuzzy systems designers started to utilize asymmetric MFs [78, 171, 172] because they are more flexible and can better represent the data distribution. However, in most existing approaches, the type of the MFs is manually chosen, therefore constraining their modeling capability. The predefined shape of the MF may not be adequate or sufficiently flexible to represent various types of data distribution. In [173] for instance, the authors pointed out that although the triangular MFs are used in most control problems, they may not be suitable for other applications.

In certain cases, an aggregation of several common type MFs may be required to capture the underlying data distribution exhibited by the numerical observations. To address this problem, GA is applied to randomly select the type of each MF [161], as reviewed in Section 2.4.5. However, the number of rules, the types and shapes of MFs for the input, output dimension, and fuzzy rules are encoded into a long integer string, which introduces great difficulty for genetic search. Moreover, the types of MFs used in [161] are still limited in the common types.

Because the representation capability of the common type MFs is limited by basic shape constraint, more generic and flexible forms of MFs should be further investigated. Some researchers pointed out that MFs must be flexible enough to develop an accurate fuzzy system [174, 175]. The piecewise linear MF is an example of flexible MF, which is determined by several sampling points and can be found in [10]. Given enough sampling points are used, they are able to either represent exactly or approximate well enough the common type MFs. Furthermore, these piecewise linear MFs can even represent shapes that are impossible for common type MFs. Although they possess great representation

capability, it is nearly impossible to manually specify such piecewise linear MFs and there is no well-established approach for deriving them from training data. Another approach for enhancing the modeling ability of MFs is that of type-2 membership functions [176]. However, type-2 MFs are difficult to understand and use because the membership value of an element belong to the type-2 MF is expressed as a type-1 fuzzy number [177].

To lessen the basic shape constraint of MFs, a type of highly-generic membership function and the learning approach for deriving these MFs are proposed in Chapter 4. While evenly distributed sampling points are adopted in the piecewise linear MFs in [10], the generic membership function is composed of unevenly spaced sampling point and straight line segments, and is capable of approximating all commonly used types of MFs. Because the shape of this newly proposed MF is not restricted to any regular shape, it is termed as “Irregular-Shaped Membership Function” (ISMF). The generality and improved accuracy of the ISMF model makes it a crucial component of the HiCEFS approach for synthesizing fuzzy decision support systems [178].

3.4 Hierarchical Coevolutionary Genetic Algorithm

Coevolution is a common scenario in nature, as many species have developed interactions with other species to improve their survival [115]. Coevolution in nature has inspired the derivation of the coevolutionary algorithms [50], which are able to search for the optimal or near-optimal solutions more efficiently than conventional evolutionary computation. These coevolutionary algorithms are more suitable for handling tasks with increasing requirements of complexity and modularity while at the same time keeping computational cost bounded [49, 179]. In coevolution, the species either interact in a competitive way (such as predators and preys) or cooperative manner (such as bees and flowers). Similarly, coevolutionary algorithms can be categorized into competitive coevolutionary algorithms and cooperative coevolutionary algorithms. The former tries to make individuals from different species more competitive through evolution and the latter aims to find individuals which cooperative well with each other and form better systems [116]. This thesis mainly focuses on the cooperative coevolutionary algorithm because it is a suitable

mechanism for synthesizing a FRBS with cooperative fuzzy membership functions and rule base. Therefore, in the rest part of this thesis, coevolutionary algorithm refers to cooperative coevolutionary algorithm for conciseness. Readers who are interested in competitive coevolutionary algorithms can refer to [180].

Conventional single-population genetic algorithms sometimes manifest convergence towards local optima and involve high computational cost. The problem is more obvious when one deals with tasks that are complex and have strong interdependencies among the components of the solution, an example of which is evolving a fuzzy decision support system. The conventional single-population genetic algorithms need to encode the parameters of the components in a candidate fuzzy decision support system and concatenate them into a single chromosome. Considering the number of parameters of the fuzzy membership functions and fuzzy rules in a complex FDSS, the corresponding chromosome can be very long [128]. In some cases, different encoding methods are adopted in the same chromosome, resulting in a complicated structure. For instance in [158], the genes in the first substring of a chromosome are real numbers while the genes in the following substring are integers. These long and hybrid structure of chromosomes can undermine the efficiency of the genetic search.

Coevolution algorithms effectively address these issues and widen the range of applications suitable for evolutionary computation. The coevolution of several species facilitates the existence of specialized populations dealing with subtasks of a complex problem. Coevolution algorithms can be robust in the face of the strong effects that interdependencies might dictate on fitness. In fact, as each individual has the possibility to participate in the cooperation several times, the chance that a good component disappears due to a destructive genetic operation on another component is lower than in conventional evolution with a single population. In coevolutionary algorithms, populations representing interdependent components are coevolved within a single ecosystem, where they can interact with one another and adapt themselves to cooperate better with other populations. If a population does not make a significant contribution to the problem-solving effort, it will eventually be eliminated or excluded from forming the full solution.

As a promising alternative to the single-population GA, coevolutionary genetic algorithms offer suitable techniques for solving complex and dynamic problems [50]. Several recent projects have proposed to use coevolutionary genetic algorithms in the design process of fuzzy systems or neural network ensembles for pattern recognition [49, 181-183]. In [181], a hierarchical coevolutionary genetic algorithm (HCGA) is used for representing different components in a fuzzy system. It efficiently coevolves the membership functions and the fuzzy rule base in populations from different hierarchical levels. However, one drawback of this approach is that predefined number of membership functions on all the input variables, which are encoded with integer genes for representing the types and real genes for the parameters of the MFs, are concatenated into a single chromosome. For high-dimension problems, the chromosome will be very long, which will in turn increase the chance of destroying the good genetic segments.

In HiCEFS, a specifically designed HCGA is adopted to automatically derive the fuzzy membership functions and fuzzy rule base from sample data. When compared with [181], the proposed HCGA in HiCEFS coevolves the ISMFs, which are represented with the per-unit offsets of the sampling points [178], for each input in separate populations. This characteristic avoids the long chromosomes found in [181], which usually result in low efficiency in evolution when dealing with high-dimension problems.

Chapter 5 provides the details of HCGA and introduces the approach of applying HCGA to coevolve the fuzzy partitions formed by ISMFs through automatically selecting most informative input variables and determine the number of ISMFs. Chapter 6 gives the details of adopting the HCGA in the HiCEFS approach which synthesizes a complete fuzzy rule-based system.

Chapter 4

Irregular-Shaped Membership Functions

4.1 Overview of ISMFs

The Irregular Shaped Membership Function [178] can be perceived as a generic form of membership function with the shape constraint significantly relaxed as compared to commonly used MFs, such as triangular, trapezoidal and Gaussian MFs. The shape of ISMF is determined by unevenly spaced sampling points and straight line segments. Figure 9 depicts some possible shapes of the ISMFs. The most important point that roughly determines the position of the ISMF is called *pivot* point. Each ISMF contains exactly one pivot point, at which the membership value is ensured to reach 1. The pivot point of the n th ISMF on an input variable is denoted as p_n . Other points in an ISMF are called *shoulder* points and positioned on either side of the pivot point. The allocation of these sampling points is a very important factor that affects the resultant ISMFs. The i th left shoulder point and the j th right shoulder point in the n th ISMF on an input variable are denoted as $s_{n,li}$ and $s_{n,rj}$, respectively.

Figure 9 Examples of possible shapes represented by ISMFs

The most important property of ISMF is that the sampling points are unevenly spaced. While the representation of evenly spaced sampling points is much simpler, because their horizontal positions are fixed after the interval is set and only parameters for membership values are required, it is difficult for the users to define a proper interval between

sampling points. As it turns out, inappropriately chosen intervals can undermine the performance of the ISMFs. If the interval is too large, the representation capability of the derived membership functions will be greatly degraded and accuracy will be reduced. If the interval is set to be too small, more than the necessary number of sampling points will be used for representing ISMFs and the computational cost will increase. Furthermore, in most cases, the data may not be evenly distributed. With unevenly spaced sampling points, the derived membership function has better capability to represent various types of data distribution.

Another important property of ISMFs is that the number of sampling points is automatically determined through evolution. According to the position of the sampling points, the repair scheme serves as the mechanism to automatically determine the number of sampling points in each derived ISMF, the detail of which will be described in Section 4.2.4. It means that an ISMF uses 3 points where a triangular MF is sufficient, while it uses more points in other places where more a complicated shape is required. Although the complexity of the search space is increased, the genetic learning approach tends to derive ISMFs that are as simple as possible for representing the data distribution, as shown in the experimental results in Section 4.3.

When the positions of the sampling points are determined, nearby sampling points are connected using straight lines. Yielding a highly customized piecewise linear representation, the ISMFs are thus not only capable of approximating the commonly used types of parameterized MFs, but also much more flexible than these in describing the data. Therefore, the ISMFs are able to better model data distributions that can not be properly represented by common type MFs, resulting in better performance of the FDSS. Despite flexibility being the main property of the genetically evolved ISMFs, decoding schemes are carefully devised to ensure the extracted ISMFs are kept valid and linguistic meaningful. The generic and accurate ISMF constitutes a crucial component in the automation of FDSS synthesis. The details of an evolutionary-based approach for deriving the ISMFs are elaborated in the following section.

4.2 Evolutionary Generation of ISMFs

4.2.1 The representation of ISMFs

For an ISMF, the maximum number of shoulder points per side, denoted as S , is an important parameter because it is directly related to the flexibility of the ISMFs and the complexity of the GA evolution. If S is too small, the data representation performance of the ISMF will be limited. However, if S is too large, the length of the chromosome representing the ISMFs will greatly increase, the search space will be enlarged and the efficiency of GA tuning will be reduced. For the leftmost and rightmost ISMFs on a variable, each requires one pivot point and maximally S shoulder points, a total of $1+S$ points. For other ISMFs, each requires one pivot point and maximally S shoulder points on each side, a total of $1+2S$ points.

Figure 10 shows the genetic segment for an ISMF with left and right shoulder points. The first gene po_n represents the x -axis per unit offset of the pivot point in the n th ISMF. The membership value of the pivot point is always unity. Similarly, the membership function of the last shoulder point on each side is always zero. Therefore, only the horizontal offsets of these points are encoded in order to reduce the length of the chromosomes. Other shoulder point requires two real values to define the horizontal and vertical offsets.

As a result, following the first gene for the pivot point, there are a total of $4S-2$ genes for S left and S right shoulder points of the ISMF. Each pair of nearby genes represents the horizontal and membership value offsets. For instance, the $xo_{n,li}$ is the horizontal offset and the $yo_{n,li}$ is the membership value offset of the i th left shoulder point $s_{n,li}$ from the $(i-1)$ th left shoulder point $s_{n,l(i-1)}$ in the n th ISMF. For the first left (right) shoulder point $s_{n,l1}$ ($s_{n,r1}$), the $xo_{n,l1}$ and $yo_{n,l1}$ ($xo_{n,r1}$ and $yo_{n,r1}$) are its horizontal and membership value offsets from the pivot point p_n .

Furthermore, for conciseness of the gene representation, the leftmost (rightmost) ISMF in a fuzzy partition only contains the pivot point and the right (left) shoulder points.

Figure 10 The genetic segment of an ISMF

In the chromosomes, the encoded offsets can be the absolute offsets or the per unit offsets. The absolute offset is the exact offset on the domain. The per unit offset on the other hand is the offset of a percent of the domain. Given any domain of range $[d_{\min}, d_{\max}]$, any absolute offset can be easily mapped to a per unit offset within $[0, 1]$ as per (16), where abo is the absolute offset, puo is the per unit offset. For instance, an absolute offset of 5 on the range $[0, 10]$ can be mapped to a per unit offset of 0.5. It is difficult to encode the absolute offsets in the same chromosome, as the ranges of different genes can vary much. Compared with absolute offsets, the per unit offset has advantage to be independent of the range of the domain. The unified content of the genes makes the representation and the implementation much easier. Therefore, per unit offsets are adopted in this work [178].

$$puo = \frac{abo}{d_{\max} - d_{\min}} \quad (16)$$

As a result, a total of $4S-1$ real numbers, which are within the range of $[0, 1]$, are required for representing the ISMF. Because the leftmost and rightmost ISMF only comprise 1 pivot and S shoulder points, they are represented by $2S$ real numbers.

Compared with representing widely used MFs (triangular, trapezoidal, Gaussian), which normally requires 2 to 4 real numbers, the cost for representing an ISMF is higher. However, it is not mandatory to include all encoded sampling points to represent the final ISMF extracted from the chromosome. In most ISMFs, not all sampling points are included in the final representation of the membership functions. In many cases, an ISMF is actually approximating a triangular, trapezoidal or Gaussian MF.

4.2.2 The representation of ISMF partitions

The fuzzy partitions formed by ISMF, called ISMF partition in short, on each input

variable will be represented in genetic segments and concatenated into a single chromosome. The length of chromosomes is denoted as Len and computed as in (17), where I is the number of input variables, N_i is the predefined number of ISMFs on the i th input variable, and S the number of shoulder points as before. The number of genes for encoding a normal ISMF is $4S-1$. The subtraction of $4S-2$ is due to the fact that the smallest (largest) linguistic value has no left (right) shoulder. The structure and content of a chromosome are illustrated in Figure 11, in which the $ISMF_n$ represents a genetic segment of ISMF.

$$Len = \sum_{i=1}^I [N_i(4S-1) - 4S - 2] \quad (17)$$

Figure 11 Content of a chromosome representing the ISMF partitions for all input variables

4.2.3 The ISMF extraction process

In the extraction process, the ISMF gene segments in the chromosome are first located according to the length of ISMF. The position and shape of an ISMF are well defined after all sampling points are determined. The position of a sampling point is decided by its x -axis and membership values. Therefore, the positions of all sampling points can be extracted by mapping the horizontal and membership value offsets to x -axis values and membership values respectively.

For N ISMFs, the default locations of their pivots are shown as the black dots and are evenly distributed on the input variable as shown in Figure 12. In terms of per unit offset on the input variable, the default position of the n th pivot is $\frac{n-1}{N-1}$. By using the evolutionary algorithm, the actual position of a pivot is tuned within the interval

$\left(\frac{n-1}{N-1} - \frac{1}{2(N-1)}, \frac{n-1}{N-1} + \frac{1}{2(N-1)} \right)$. According to po_n , which is the pivot offset in the

interval, the per unit offset of pivot p_n from $\frac{n-1}{N-1} - \frac{1}{2(N-1)}$ is $\frac{po_n}{N-1}$. Therefore, on the range R_D , the horizontal per unit offset pd_n of pivot point of the n th ISMF is computed in (19), where po_n is the pivot point offset within the interval, N is the number of ISMFs on the variable domain. However, if $po_1 < 0.5$ (or $po_N > 0.5$), the first pivot (or the last pivot) will be placed out of the range $[d_{\min}, d_{\max}]$. In this case, we need to shift the pivot to the closest point in the input dimension. Therefore, if $\lambda < 0$, let $pd_n = 0$; if $\lambda > 1$, let $pd_n = 1$.

Figure 12 Determining the position of the pivot points on an input variable

$$R_D = |d_{\max} - d_{\min}| \quad (18)$$

$$\begin{cases} \lambda = \frac{n-1}{N-1} - \frac{1}{2(N-1)} + \frac{po_n}{N-1} = \frac{2n-3}{2(N-1)} + \frac{po_n}{N-1} \\ pd_n = \lambda, \text{ if } 0 \leq \lambda \leq 1 \\ pd_n = 0, \text{ if } \lambda < 0 \\ pd_n = 1, \text{ if } \lambda > 1 \end{cases} \quad (19)$$

The horizontal per unit offset of a shoulder point is the percentage of the range between two nearby pivot points. The range $R_{PL,n}$ and $R_{PR,n}$ are for the left and right shoulder points, respectively, as defined in (20).

$$\begin{cases} R_{PL,n} = pd_n - pd_{n-1}, \text{ for left shoulder points of } n\text{th ISMF} \\ R_{PR,n} = pd_{n+1} - pd_n, \text{ for right shoulder points of } n\text{th ISMF} \end{cases} \quad (20)$$

For shoulder points, the per unit offset on the range R_D can be computed in (21), where $xd_{n,lu}$ is the horizontal per unit offset of the u th left shoulder point of the n th ISMF from

the range minimum $d_{\min}$ on R_D . xd_{n,r_v} is horizontal per unit offset of the v th right shoulder point of the n th ISMF from the range minimum $d_{\min}$ on R_D . $xo_{n,li}$ and $xo_{n,rj}$ are the offset of the i th left shoulder point and the offset of the j th right shoulder point of the n th ISMF encoded in the chromosomes.

$$\begin{cases} xd_{n,lu} = pd_n - R_{PL,n} \cdot \sum_{i=1}^u xo_{n,li} \\ xd_{n,r_v} = pd_n + R_{PR,n} \cdot \sum_{j=1}^v xo_{n,rj} \end{cases} \quad (21)$$

Based on these offsets, the x -axis value and membership value of each sampling point of an ISMF can be determined as given in (22)–(24), where px_n and py_n are the x -axis and membership values respectively of the pivot point of the n th ISMF. $x_{n,lu}$ and $y_{n,lu}$ are the x -axis and membership values respectively of the u th left shoulder point of the n th ISMF. x_{n,r_v} and y_{n,r_v} are the x -axis and membership values respectively of the v th right shoulder point of the n th ISMF.

$$\begin{cases} px_n = d_{\min} + R_D \cdot pd_n \\ py_n = 1 \end{cases} \quad (22)$$

$$\begin{cases} x_{n,lu} = d_{\min} + R_D \cdot xd_{n,lu} \\ y_{n,lu} = 1 - \sum_{i=1}^u yo_{n,li} \end{cases} \quad (23)$$

$$\begin{cases} x_{n,r_v} = d_{\min} + R_D \cdot xd_{n,r_v} \\ y_{n,r_v} = 1 - \sum_{j=1}^v yo_{n,rj} \end{cases} \quad (24)$$

Figure 13 Determining the relative positions the sampling points in a ISMF according to the per unit offsets

At the end of this subsection, Figure 13 provides a clearer view of how the positions of the sampling points in an ISMF are determined according to their per unit offsets.

4.2.4 Maintaining the interpretability of the ISMF partitions

As mentioned in Section 2.2, although accuracy is the primary focus of fuzzy modeling, researchers are starting to pay more attention to the issue of interpretability [13, 184, 185]. According to the excellent discussions in [71, 130, 186], the distribution of the fuzzy membership functions is an important factor that affects the interpretability of a fuzzy rule-based system. There are two important criteria for interpretable fuzzy partitions, including completeness and distinguishability.

The measure of completeness for a fuzzy partition is defined as per (25), where N is the number of membership functions and $\mu_j(x)$ is the j th membership function on the fuzzy partition. If $CM(x) = 0$ for a certain x , then the fuzzy partition is incomplete. This means that there is no fuzzy membership function covering the point of x . The examples of a complete and an incomplete fuzzy partition are shown in Figure 14.

Figure 14 Example of (a) a complete fuzzy partition and (b) an incomplete fuzzy partition

$$CM(x) = \sum_{j=1}^N \mu_j(x) \quad (25)$$

Distinguishability is one of the important aspects for interpretability of fuzzy partition. The examples of a distinguishable and a non-distinguishable fuzzy partition are given in Figure 15. In part (a) of Figure 15, the linguistic terms such as low, medium, high can be assigned to the fuzzy membership functions. However, as shown in part (b) of Figure 15, if two nearby fuzzy sets severely overlap each other, these membership functions are too

similar and not easy to distinguish, which increases the difficulty to assign a proper semantic meaning for each of them. Besides similarity, the number of membership functions within a fuzzy partition is another important factor for distinguishability. Namely, the fewer membership functions in a fuzzy partition, the more distinguishable the fuzzy partition is. Generally, the number of membership functions in a fuzzy partition is preferred to be smaller than ten [130, 161].

Figure 15 Example of (a) a distinguishable fuzzy partition and (b) a non-distinguishable fuzzy partition

In the proposed approach, the position of the sampling points is not constrained in the genetic representation so as to provide large degree of freedom in the evolution process for formulating flexible ISMFs. However, this can result in obtaining incomplete fuzzy partitions. Another possible result is to derive severely overlapping and indistinguishable fuzzy partitions, the examples of which can also be found in [19]. Thus, distinguishability and completeness of the fuzzy partitions are ensured by adopting the repair schemes in the extraction process. Besides this functionality, the repair schemes also serve as the mechanism to automatically determine the number of sampling points in each derived ISMF.

In Figure 16(a), a repair is adopted to reduce overlapping between two ISMFs by ensuring that, at the horizontal position of the pivot of an ISMF, the membership value of any other ISMF must be zero. In Figure 16 (b), if the membership value of a sampling point is a negative value, the membership value will be amended to zero, while the horizontal position of the sampling point remains unchanged. Note that if a left (right) shoulder point is repaired as in Figure 16(a) or (b), the remaining left (right) shoulder points will not be included in the final representation of the ISMF. This mechanism is used to automatically determine the actual number of sampling points in each ISMF.

Figure 16 Repair scheme (a) for reduce overlapping between two ISMFs and (b) for reposition sampling point with negative membership value

In the decoding process, it is possible that two nearby ISMFs overlap at a very high (e.g. above 0.8) or a very low (e.g. below 0.2) membership value. If they overlap at a very high membership value, these two ISMFs will be similar to each other and the distinguishability of the fuzzy partition will be diminished. On the other hand, it is possible that they overlap at a very low membership value, or worse that two nearby ISMFs do not overlap at all, so that some space in the variable domain is not covered by any membership function. In this case, the fuzzy partition is incomplete. In order to maintain the completeness and distinguishability of the fuzzy partition formed by ISMFs, proper repair schemes are devised. In Figure 17(a), (b) and (c), it shows the repair for ensuring that every two nearby ISMFs overlap at a proper membership value.

Figure 17 Repair scheme for ensuring two nearby ISMFs overlap at a proper membership value

The repairs in Figure 17 are achieved by re-computing the offsets as per (26)-(28), where $xo'_{n,rv}$ and $yo'_{n,rv}$ are the new offsets of the repositioned sampling point $s'_{n,rv}$; similarly $xo'_{(n+1),lu}$ and $yo'_{(n+1),lu}$ are for $s'_{(n+1),lu}$; $bound=b_U$ when two ISMFs cross above upper bound b_U , while $bound=b_L$ when two ISMFs cross below lower bound b_L or they never cross.

$$xo'_{n,rv} = xo'_{(n+1),lu} = (1 - \sum_{j=1}^{v-1} xo_{n,rj} - \sum_{i=1}^{u-1} xo_{(n+1),li}) / 2 \quad (26)$$

$$yo'_{n,rv} = 1 - bound - \sum_{j=1}^{v-1} yo_{n,rj} \quad (27)$$

$$yo'_{(n+1),lu} = 1 - bound - \sum_{i=1}^{u-1} yo_{(n+1),li} \quad (28)$$

Because these repair schemes only require minimal computational cost, their effect on the total computational complexity of deriving ISMF from chromosomes is insignificant.

4.2.5 Procedure and settings of the genetic algorithm

In this section, a canonical single-population GA is used for synthesizing the proposed ISMFs in fuzzy rule-based systems. The procedure of the GA is briefly described hereafter.

In step 1, the parameters in the GA are initialized, including the number of maximum generation $G_{\max}$, the size of population SP , the crossover rate P_c and mutation rate P_m . The chromosomes represent the ISMFs as described in Section 4.2.1 and 4.2.2. The contents of these chromosomes are randomly initialized. The number of generation G is set to 1.

In step 2, the ISMFs in each chromosome are extracted as described in Section 4.2.3. The extracted ISMFs are used to form the fuzzy partitions on the input variables in a FRBS. The resultant FRBS is evaluated with the training data and the fitness value is assigned to the corresponding chromosome.

In step 3, the population is evolved with a simple two-point crossover operator and a random mutation operator. By adopting elitism, the best 5 of the chromosomes in current generation are kept intact.

In step 4, if G reaches $G_{\max}$, the evolution terminates. Otherwise, G is set to $G+1$ and the evolutionary process is branched to step 2 to start the next generation.

In addition, the settings of some important parameters in GA are explained as follows. The $G_{\max}$ is heuristically selected according to the complexity of the problem being dealt with. In the experiments in Section 4.3, the value of $G_{\max}$ varies within the range from 500 to 5000.

For the size of population SP , a moderate value of 100 is selected to maintain the genetic diversity as well as search efficiency.

The crossover rate P_c and mutation rate P_m are selected as 80% and 1% respectively, after different combinations within the range of [50%-90%] and [0.1%-10%] are attempted. The results of experimental attempts show that the search performance of GA varies according to different selections of the P_c and P_m . However, the GA is not very sensitive to the P_c and P_m because the GA can yield satisfactory results when the P_c and P_m are selected in a suitable range.

4.2.6 Fitness evaluation of the ISMFs

The fuzzy decision support systems can be used for classification or prediction problems. The fitness value f_{Acc} of the decoded ISMF partition is directly related to the accuracy of the resultant FDSS in solving these problems. For classification, f_{Acc} is measured as per (29), where N_{CC} is the number of correctly classified instances and N_D is the total number of instances. For regression, the f_{Acc} can be evaluated either in terms of the mean squared error (MSE) or the square of the Pearson product-moment correlation coefficient (R^2) as per (30), where $std(Y')$ is the standard deviation of the target outputs.

$$f_{Acc} = 100 \times N_{CC} / N_D \quad (29)$$

$$f_{Acc} = \begin{cases} 100 / (1 + MSE / std(Y')), & \text{for minimizing MSE} \\ 100 \times R^2, & \text{for trend following} \end{cases} \quad (30)$$

4.3 Benchmarking Results and Analysis

4.3.1 Experimental settings

Two classification problems (Pima diabetes diagnosis and phoneme classification) and two approximation problems (automobile MPG prediction and solar activity prediction) are studied in this section to benchmark the ISMFs against common type MFs.

Because there are no established measures for direct assessment of which type is better than the others, the performance of MFs is measured according to the performance of the resultant fuzzy rule-based system. Except for the type of MFs, all major settings in the fuzzy rule-based systems, including the number of MFs for each variables and the fuzzy rule base are kept identical in order to make the comparison objective. The benchmarked systems thus include fuzzy rule-based systems employed with triangular MFs (FS-TRIA), with trapezoidal MFs (FS-TRAP), with Gaussian MFs (FS-GAUS), and lastly with the proposed ISMFs (FS-ISMF).

In all problems, fuzzy rule-base systems with different type of MFs will employ the same fuzzy rule base that is derived with Wang-Mendel (W-M) rule learning method [29]. The MFs for the output variable are either manually predefined or generated with the Fuzzy Kohonen Partition algorithm [16]. The approach described in this chapter is used to generate the ISMFs. The parameter S is set to 8 in the experiment to provide a large degree of freedom for the ISMFs. However, experiment results show that a smaller S is enough for most problems. Therefore, S can be predefined according to the complexity of the problem. For generating the common type MFs, the GA is slightly modified because they are defined by fewer parameters. That is the left, right width and the central point of triangular and Gaussian MFs; the left, right and two central points of trapezoidal MFs are encoded instead of the offsets of the sampling points in ISMF. For each experimental case, the learning and testing process is performed 30 times on a PC with a Pentium-4 2.4 GHz CPU.

4.3.2 Pima diabetes diagnosis

The Pima Indian diabetes dataset is a dataset with moderate number of inputs and instances, which is available in UCI machine learning repository [187]. It is adopted to benchmark the systems with different type of MFs on this unbalanced classification dataset with moderate number of instances, 500 benign and 268 malign instances. The fuzzy systems classify the instance to be benign or malign based on eight attributes including age, number of pregnant times, body mass index, plasma glucose concentration, diastolic blood pressure, diabetes pedigree function, and other medical tests. The whole dataset is randomly partitioned into 3 different train-test groups (50% and 50%), which are denoted as Div1-3 respectively. For this problem, $G_{\max}$ is set to 2000.

Table III Result comparison on Pima diabetes diagnosis problem

Pima	Div1 accuracy	Div2 accuracy	Div3 accuracy	Avg accuracy	Avg training time (sec)
FS-TRIA	70.03% $\pm$ 1.42%	68.92% $\pm$ 1.29%	70.84% $\pm$ 1.24%	69.93%	481.6
FS-GAUS	72.27% $\pm$ 1.35%	69.03% $\pm$ 1.15%	69.93% $\pm$ 1.26%	70.41%	617.4
FS-TRAP	70.43% $\pm$ 1.68%	68.48% $\pm$ 1.26%	70.45% $\pm$ 1.30%	69.79%	504.2
FS-ISMF	74.65% $\pm$ 0.91%	72.74% $\pm$ 1.18%	73.67% $\pm$ 1.28%	73.69%	559.7
Δ TRIA	6.60%	5.54%	3.99%	5.37%	16.20%
Δ GAUS	3.29%	5.37%	5.35%	4.65%	-9.35%
Δ TRAP	5.99%	6.22%	4.57%	5.59%	11.00%

The results listed in Table III show that the FS-ISMF is able to achieve approximately 5% increase in accuracy than fuzzy systems adopting common type MFs. The column “ Δ TRIA” indicate the increases in statistics (accuracy and training time) when comparing FS-TRIA with FS-ISMF expressed as a percentage of the former. It is the same case in the column “ Δ GAUS” and column “ Δ TRAP”.

4.3.3 Phoneme classification

This experiment is used to evaluate the fuzzy system adopting the proposed ISMFs on an very unbalanced dataset with large number of instances. The ELENA Phoneme dataset was used in the European Esprit project: ROARS, which tried to distinguish between nasal and oral vowels using a real time analytical system for French and Spanish speech recognition. It contains 5404 vowel patterns, among which 3818 are from the nasal class

and the other 1586 from the oral class. Each pattern is characterized with 5 attributes which are the amplitudes of the five first harmonics, normalized by the total energy (integrated on all the frequencies). The whole dataset is randomly partitioned into 3 different train-test groups. Because of the samples is abundant in this dataset, only 20% is used for training and the remaining 80% for testing. For this problem, $G_{\max}$ is set to 2000. The experimental results are listed in Table IV. Similar to the results of previous experiment, the FS-ISMF outperforms the fuzzy systems adopting common type MFs. Even though the rule base generated by W-M limits the accuracy of the fuzzy systems, the accuracy of 79.90% achieved by the FS-ISMF compares favorably with the accuracy of 75.89% and 73.27% respectively achieved by the FALCON-FKP and FALCON-PFKP [188], which adopt FKP and PFKP clustering algorithms to generate the trapezoidal MFs.

Table IV Result comparison on Phoneme classification problem

Phoneme	Div1 accuracy	Div2 accuracy	Div3 accuracy	Avg accuracy	Avg training time (sec)
FS-TRIA	76.26% $\pm$ 1.63%	75.86% $\pm$ 1.38%	76.81% $\pm$ 1.57%	76.61%	808.6
FS-GAUS	76.21% $\pm$ 1.52%	76.73% $\pm$ 1.33%	76.98% $\pm$ 1.15%	76.94%	957.7
FS-TRAP	75.20% $\pm$ 1.35%	76.03% $\pm$ 1.86%	76.11% $\pm$ 2.04%	76.08%	815.3
FS-ISMF	79.43% $\pm$ 0.86%	79.24% $\pm$ 1.27%	80.12% $\pm$ 0.93%	79.90%	883.7
Δ TRIA	4.14%	4.44%	4.29%	4.29%	9.28%
Δ GAUS	4.21%	3.26%	4.06%	3.84%	-7.72%
Δ TRAP	5.60%	4.21%	5.25%	5.02%	8.38%

4.3.4 Automobile MPG prediction

In this experiment, the automobile MPG (miles per gallon) dataset on UCI machine learning repository is used to benchmark the FS-ISMF against the fuzzy systems adopting common type MFs on a regression dataset with a small number of input features. An automobile's fuel consumption in terms of MPG is predicted according to the features including number of cylinders, displacement, horsepower, weight, acceleration and model year. The whole dataset contains 398 samples and is randomly partitioned into 3 different train-test groups (33% and 67%, respectively). For this problem, the number of maximum generations is set to 1000. According to the results shown in Table V, the FS-ISMF is able to provide more accurate prediction on the automobile MPG. The FS-ISMF reduces

the prediction mean squared error (MSE) by about 10% compared to FS-GAUS and by more than 30% compared to FS-TRIA and FS-TRAP.

Table V Result comparison on automobile MPG prediction

Auto MPG	Div1 MSE	Div2 MSE	Div3 MSE	Avg MSE	Avg training time (sec)
FS-TRIA	21.44 ± 3.21	19.07 ± 2.96	18.93 ± 3.06	19.81	95.9
FS-GAUS	14.56 ± 2.19	15.11 ± 2.42	13.85 ± 2.73	14.51	144.4
FS-TRAP	19.59 ± 2.12	19.51 ± 2.57	19.23 ± 2.28	19.44	100.7
FS-ISMF	12.60 ± 2.28	13.68 ± 2.16	12.92 ± 2.44	13.07	109.1
Δ TRIA	-41.23%	-28.26%	-31.75%	-34.05%	13.76%
Δ GAUS	-13.46%	-9.46%	-6.71%	-9.93%	-24.45%
Δ TRAP	-35.68%	-29.88%	-32.81%	-32.80%	8.34%

4.3.5 Solar activity prediction

This experiment is used to benchmark the FS-ISMF against the fuzzy systems adopting common type MFs on a complex real-world regression problem. Solar activity forecasting is an important topic for various areas, including long term climate variations, operations of low earth orbiting satellites, electric power lines, geophysical applications and high frequency radio communications. Nevertheless, the solar activity is very difficult to predict on the basis of time series of various proposed indicators, due to high frequency content, noise contamination, high dispersion level, high variability in phase and amplitude. The monthly smoothed sunspot number is one of the important indices characterizing the level of solar activity. The dataset of the monthly smoothed sunspot number used in this experiment is obtained from Solar influences data analysis center at royal observatory of Belgium (<http://sidc.oma.be/sunspot-data/>).

The difference between monthly smoothed sunspot number $y(t)$ and $y(t-1)$, denoted as $\Delta y(t)$, is predicted instead of the $y(t)$. According to the 6-tuples $[\Delta y(t-5), \dots, \Delta y(t-1), \Delta y(t)]$, the fuzzy rule-based systems predict the change in the smoothed monthly sunspot number in the next month $\Delta y(t+1)$. The time series of the monthly smoothed sunspot number $y(t)$ from January 1800 to December 1900 is used to form the training dataset. The time series from January 1901 to December 2005 is used to form the testing dataset. For this problem, $G_{\max}$ is set to 2000. Both of the square of Pearson correlation coefficient

(R^2) and mean squared error (MSE) are used to evaluate the accuracy of the prediction made by the fuzzy rule-based systems.

Figure 18 (a) The predicted and the actual time series of the change in the monthly smoothed sunspot number (b) the errors of the predicted time series

As shown in Table VI, the FS-ISMF is able to achieve an increase of 5% to 7% in R^2 and reduce the MSE by 9% to 40% than other benchmarked fuzzy systems. To provide a clearer view, the time series of the actual and predicted change in monthly smoothed sunspot number are shown in Figure 18(a). According to the given results, the FS-ISMF shows better capability in following the trend of monthly smoothed sunspot number and providing more accurate prediction. The errors in predicted time series are represented as the difference between the actual and predicted change in monthly smoothed sunspot number. As shown in Figure 18(b), the large prediction errors are only found at the months when drastic change in the sunspot number is observed.

Table VI Result comparison on solar activity prediction

Sunspot	R^2	MSE	Avg Training Time
FS-TRIA	0.700 ± 0.043	3.020 ± 0.19	2281.93
FS-GAUS	0.686 ± 0.032	4.014 ± 0.17	3023.36
FS-TRAP	0.707 ± 0.048	2.598 ± 0.22	2307.72
FS-ISMF	0.740 ± 0.052	2.371 ± 0.25	2678.96
Δ TRIA	5.71%	-20.49%	17.40%
Δ GAUS	7.87%	-40.93%	-11.39%
Δ TRAP	4.67%	-8.74%	16.09%

4.3.6 Analysis and discussions

Experimental results in the previous sections show that there is not a specific type among common type MFs which is superior to others in all experiments. This demonstrates the limitation of manually predefining a type of MF for a certain problem, which is the common practice in most works. The fuzzy system with ISMFs is able to achieve superior performance to other fuzzy systems that employ common types of membership functions in all the experiments. When adopting the ISMFs, the type of MF does not need to be selected because the ISMFs are sufficiently flexible to subsume the common types and provide superior performance. It is a principal advantage that facilitates automating the fuzzy rule-based system synthesis.

To closely examine the learned membership functions, some examples of the membership functions in the trained FS-ISMF and FS-TRIA for the Pima diabetes diagnosis and solar activity prediction are shown in left and right column in Figure 19, respectively. In each column, the ISMFs are shown in the left and the triangular MFs in the right. It is shown that all generated ISMFs contain less shoulder points on either side than the possible maximum number S , which is set to 8. That is because the GA tends to generate ISMFs with minimal required sampling points for describing the data distribution. Without basic shape constrains, the ISMFs are able to form specific shapes for different areas on the variable domain. As a direct result of adopting the ISMFs, the FS-ISMF is able to achieve higher accuracy in the classification and approximation than the fuzzy systems adopting common type MFs. As can be seen in Figure 19, the fuzzy partitions are complete to cover the range of the variable domains. The membership functions on the fuzzy partitions are distinguishable so that meaning linguistic terms can be assigned to them.

Therefore, it shows that the interpretability of the fuzzy partitions formed by ISMFs is carefully maintained.

Figure 19 Derived ISMFs and triangular MFs for (a) Pima diabetes diagnosis and (b) solar activity prediction

As shown in Table III-VI, it generally requires more training time for FS-ISMFs than FS-TRIA and FS-TRAP. When FS-ISMF is achieving about 4% to 34% increase in accuracy, the increase in the training time is in an acceptable range of 8% to 17%. This confirms that additional computational cost is the trade off for generating the ISMFs with higher flexibility, even though the proposed approach is devised to be efficient. However, because many applications can be handled in an offline mode, it is well worth spending more training time for the enhancement in system accuracy. It is interesting to find that the training time for FS-ISMF is still lower than that for FS-GAUS. The main reason for this observation is as follows. The total computational time in the training phase mainly consist of two parts: the time for evolving the MFs and the time for evaluating the MFs. Less time is required for evolving Gaussian MFs than that is required for evolving ISMFs because fewer parameters are tuned for Gaussian MFs. However, computing the membership value of an instance in an ISMF is less computational costly than computing

the membership value of an instance in a Gaussian MF. Because the membership value computation is highly frequent in the training phase, the total training time for a FS-GAUS is greater than that for a FS-ISMF.

In conclusion, as demonstrated in the experiments, the ISMF provides a generic form of fuzzy membership function for the fuzzy rule-based system designers' choice and constitutes a crucial component in the automation of fuzzy decision support system synthesis.

Chapter 5

Hierarchical Coevolutionary Genetic Algorithm

5.1 Overview of the Hierarchical Coevolutionary Model

In the HiCEFS, the hierarchical coevolutionary genetic algorithm (HCGA) serves as the major learning mechanism. Through a proper decomposition of the learning task into several subtasks, the candidate solutions are coevolved together in a number of populations, allowing for different kinds of chromosomes and genetic operators. For instance in [183], the complex task of learning a neural network ensemble is considered as a higher level task, which is further partitioned into the lower level subtasks of learning individual neural networks. These learning tasks are handled by the corresponding higher level and lower level populations. The chromosomes from these populations are interrelated because some genes in the higher level ones act as the control genes, which determine the lower level ones to be selected. During evaluation, the solutions to these subtasks are extracted and combined to form the solution to the whole task. Cooperation among these chromosomes is encouraged by rewarding them based on the fitness of the solutions they form.

The main advantage of HCGA over conventional evolutionary algorithms is that the HCGA is more suitable and efficient for solving those complex but highly structured problems by searching on the smaller components of the structure in parallel. However, the efficiency of the HCGA depends on a proper decomposition of the learning task into several subtasks, which in most cases relies on human expertise and domain analysis [51]. Based on how the learning task is decomposed, the parameter L_{max} determines the number of population levels in the HCGA. For instance, the L_{max} is set to 2 for automatic generation of the ISMF partitions in this chapter and in [189]. The Level-2 population evolves the combination of ISMF partitions on all input variables. Each Level-1 population is specialized in tuning the ISMF partition on an input variable. The L_{max} is set to 5 for automatic synthesis of the whole fuzzy decision support system, the details of which is elaborated in Chapter 6.

The evolutionary process of the HCGA is illustrated in Figure 20 and elaborated hereafter.

Figure 20 Flowchart of the evolutionary process of HCGA

In step 1 of the coevolution, the main parameters of HCGA are initialized, including the number of population levels $Lmax$, the number of populations NP_h ($h \in \{1, 2, \dots, Lmax\}$), the size of populations SP_h , the set of genetic operators GOP_h , the crossover rate $P_{c,h}$ and mutation rate $P_{m,h}$ from different levels and the number of maximum generation G_{max} . All the populations are initialized and the generation parameter G is set to 1.

In step 2, the evaluation of the chromosomes in current generation starts with the variable c , which denotes the index of the chromosome in the highest level (i.e. Level- $Lmax$) population, being set to 1.

In step 3, the set of lower level chromosomes selected by the c th chromosome in the Level- L_{max} population, denoted as $subch(L_{max}, c)$, are determined according to the control genes in the higher level chromosomes. The c th chromosome in the Level- L_{max} population together with $subch(L_{max}, c)$ is denoted as $fullch(L_{max}, c)$, which is the full set of chromosomes required to decode a candidate solution.

In step 4, the solutions to the subtasks, sub-solutions in short and denoted as $subsl(L_{max}, c)$, are decoded from the chromosomes $fullch(L_{max}, c)$ determined in previous step.

In step 5, the full solution to the search task $fullsl(L_{max}, c)$ is formed by merging the solutions to subtasks $subsl(L_{max}, c)$, which are derived in previous step.

In step 6, the full solution $fullsl(L_{max}, c)$ is subsequently evaluated and the fitness value is assigned to the involved chromosomes $fullch(L_{max}, c)$.

In step 7, if c is equal to the size of Level- L_{max} population, denoted as $SP_{L_{max}}$, it means all chromosomes in the highest level population are evaluated. Then, proper fitness values are assigned to those lower level chromosomes that are not evaluated in this generation. Otherwise, c is set to $c+1$ and the evolution process is branched to step 3 to start decoding the next chromosome in Level- L_{max} population.

In step 8, the evolution process of HCGA terminates if the $G=G_{max}$ or the learning goal is reached. Otherwise, appropriate fitness value is assigned to the set of unevaluated chromosomes in this generation, which is denoted as $uech(G)$. All populations are evolved with the corresponding genetic operators. Then, G is set to $G+1$ and the evolution process is branched to step 2 to start the next generation.

The HCGA's capability of solving complex problem is demonstrated with a concrete example in the remaining part of this chapter, where it is applied to select the input variables and derive proper number of ISMF for each selected input variable.

5.2 Coevolution of ISMF partitions

5.2.1 Genetic representation

In the evolutionary approach for deriving ISMF discussed in Section 4.2, a major drawback is that the number of ISMFs on each input variable needs to be predefined by the designer. This same shortcoming is shared by many clustering algorithms. Besides the shape of MFs, the number of MFs for each input is also an important factor in the performance of final fuzzy partitions. Over-refined fuzzy partitions can result in an excessively complex fuzzy rule-based system. On the other hand, a coarse fuzzy partition cannot accurately reflect the data distribution and therefore limits the system performance. The problem is that, for many datasets, it is very difficult or even impossible for human to set an appropriate number of membership functions without resorting to trial-and-error approaches. Furthermore, it may be more suitable to adopt different numbers of membership functions for different input variables, if optimal performance is desired. Typically, some inputs may need a less refined fuzzy partition, while others may need a more refined one.

Another issue with the existing approach is that, once the number of membership functions has been defined, these sampling points are confined and tuned within evenly partitioned intervals, which consequently constrains the distribution of the ISMFs.

The third limitation is that the ISMFs on all input variables are concatenated in a chromosome. When dealing with problems of large dimension, the existing genetic representation scheme would potentially result in very long chromosomes. In evolutionary algorithms, long chromosomes should be avoided, because they make the genetic search less efficient.

In this section, the HCGA is applied to overcome the above deficiencies, which undermine the achievement of high system accuracy as well as MF construction automation, by selecting the most relevant input variables and deriving proper number of ISMFs for each selected input variable.

The structure of the 2-level Hierarchical Coevolutionary GA adopted to generate the ISMF partitions is illustrated in Figure 21 and introduced hereafter.

Figure 21 Structure of the 2-Level HCGA for deriving ISMFs

In the HCGA, a Level-2 (L2) population is used to select the combination of the fuzzy partitions on all the input variables, which is termed fuzzy partition table in this thesis. Besides the Level-2 population, there are I Level-1 (L1) populations for evolving the fuzzy partitions formed by ISMFs, where I is the number of the input variables. Each Level-2 chromosome, which is shown in part-(a) of Figure 21, is applied to select the fuzzy partition on each input variable. In a Level-2 chromosome, the content of the i th gene is the index FP_i of the selected (FP_i)th chromosome in the i th Level-1 population as shown in part-(b) of Figure 21. If FP_i is equal to zero, the corresponding input variable will not be included in the fuzzy inference process. The i th Level-1 population evolves

the fuzzy partition for the i th input variable. Each Level-1 chromosome is applied to represent a possible fuzzy partition on the corresponding input variable as depicted in part-(c) of Figure 21. With this approach, the task of finding the ISMFs for each input variable is partitioned into several easier subtasks, the solutions to which are coevolved in several populations.

Instead of being predefined, the number of ISMFs on an input variable is to be decided during the evolution process, based on the positions of the pivots of the ISMFs. An important parameter N , which denotes maximum number of possible ISMFs covering the variable domain, is introduced in the approach with HCGA. If a large N is set, the chromosome will be unnecessarily long and complex. Furthermore, the fuzzy partition is more distinguishable when fewer fuzzy membership functions are used in a fuzzy partition. N is predefined to be 9 because the number of fuzzy membership functions in a fuzzy partition is generally preferred to be smaller than ten [130, 161]. These ISMFs are concatenated as a L1 chromosome as depicted in part-(c) of Figure 21.

In the previous chapter, the ISMFs on all the inputs are concatenated in a single chromosome, the length of which is computed as in (17). In HCGA, the length of L1 chromosomes and L2 chromosomes, denoted as Len_1 and Len_2 respectively, can be computed as in (31), where I is the number of input variables, S is the maximum number of shoulder points per side for an ISMF, N is the maximum number of ISMFs for an input variable. Because the ISMFs on each input variable are evolved in different L1 populations, each L1 chromosome only represents N ISMFs and long chromosomes are avoided even when dealing with high-dimension problems.

$$\begin{cases} Len_1 = N(4S - 1) - 4S - 2 \\ Len_2 = I \end{cases} \quad (31)$$

5.2.2 Extracting the ISMF partitions

All the Level-2 and Level-1 chromosomes are actually denoted as strings of integer or real values. The solutions they represent must be extracted before their fitness can be evaluated. Different from canonical GAs, the Hierarchical Coevolutionary GA divides

and evolves the candidate solutions in several populations. Thus, the extracting scheme is carefully devised to correctly derive the solutions from HCGA chromosomes, as elaborated hereafter.

Selecting input variables and determining number of ISMFs

In Level-2 chromosomes, which represent the fuzzy partition tables, the integer value FP_i in i th gene means that the (FP_i) th Level-1 chromosome (fuzzy partition on one input variable) in the i th L1 population (for the i th input variable) is chosen to form the fuzzy partition table. If the value FP_i is zero, the i th input variable will not be considered when constructing the fuzzy system.

When a L1 chromosome is selected, the ISMFs have to be decoded from this string of real values. A L1 chromosome contains N ISMFs. However, not all of the ISMFs must be selected to cover the input variable. Whether a certain ISMF is included in the fuzzy partition depends on whether its pivot exceeds the maximum of the input domain. The first gene of each ISMF segment is the horizontal per unit offset of the pivot from the pivot of the previous ISMF. If the ISMF is the leftmost one, the offset of its pivot is from the minimum of the input domain. The per unit offset of a pivot point is the percentage of the range R_D of a certain input domain.

The example of determining the positions of the pivots according to the pivot offsets is shown in Figure 22. The R_D , $d_{\min}$ and $d_{\max}$ are computed as per (32), where x_{ij} is the i th input value of the j th training sample. For the n th pivot, the per unit offset from the minimum of the domain $d_{\min}$ is denoted as pd_n and determined using (33), where po_i is the per unit offset of the pivot of the i th ISMF. A pivot exceeds the maximum of the dimension, if its per unit offset from the minimum of the domain $pd_n > 1$ as the p_4 in Figure 22. Hence, those ISMFs will be discarded when synthesizing the fuzzy rule-based system. If the number of selected ISMFs is equal to one, the corresponding input variable will also be discarded.

Figure 22 The example of determining the positions of the pivots according to their offsets

$$\begin{cases} R_D = |d_{\max} - d_{\min}| \\ d_{\min} = \min_j(x_{ij}) \\ d_{\max} = \max_j(x_{ij}) \end{cases} \quad (32)$$

$$pd_n = \sum_{i=1}^n po_i \quad (33)$$

This property ensures that the HCGA is capable of determining the selected input variables and the number of membership functions on each input variable. This feature selection capability helps to increase the performance of the final FDSS, because using too many input variables results in lower accuracy and interpretability of the model due to redundancy, non-informative input variables and noise [112]. Furthermore, it avoids the biases that are unavoidable in many existing techniques for deriving MFs.

Extracting the selected ISMFs

After finding the input variables and the ISMFs that participate in the fuzzy rule-based system, these selected ISMFs would be extracted from the gene expression. As described in Section 4.2.3, the horizontal per unit offset of the shoulder points is the percentage of the range RP between two nearby pivot points. Therefore, the RP need to be determined before computing the actual horizontal position of the shoulder points. The range $R_{PL,n}$ and $R_{PR,n}$ for the left and right shoulder points are computed as per (34).

$$\begin{cases} R_{PL,n} = p_n, & \text{for left shoulder points of } n\text{th ISMF} \\ R_{PR,n} = p_{n+1}, & \text{for right shoulder points of } n\text{th ISMF} \end{cases} \quad (34)$$

After determining the $R_{PL,n}$ and $R_{PR,n}$, the computation of the horizontal position and membership value of the sampling points is the same as that in Section 4.2.3. The per unit

offset of the left and right shoulder points on the range R_D can be computed by using (21). Based on these offsets, the x -axis value and membership value of the pivot point, left shoulder points and right shoulder points can be determined as per (22), (23) and (24), respectively.

5.2.3 Fitness evaluation and assignment

In the fitness evaluation process, all the chromosomes in the L2 population of the HCGA are evaluated. For each L2 chromosome, the corresponding fuzzy partitions formed by the ISMFs can be extracted by using the process described in Section 5.2.2. Then, a rule learning algorithm is adopted to derive the rule base after the genetically derived ISMFs are incorporated into the fuzzy rule-based system. The rule learning approach needs to be very efficient because it is taken for each possible fuzzy partition table and therefore invoked many times during the evolution. In this work, the Pseudo Outer-Product (POP) algorithm [16] is adopted because it is an efficient approach which can derive the rule base by making only one-pass learning on the training dataset. The resultant FRBS is presented with the samples in the training dataset. Based on the results, the fitness value f_{Acc} of the corresponding solution is evaluated with the fitness function (29) or (30) and then assigned to the L2 chromosome as in (35).

$$f_{L2,j} = f_{Acc} \quad (35)$$

However, it is possible that some L1 chromosomes may be either selected by several different L2 chromosomes or never selected by any L2 chromosome in a certain generation. Thus, the fitness assignment for L1 chromosomes is not as straightforward as that for L2 chromosomes. For the j th L1 chromosome, if it is selected by any L2 chromosome, its new fitness value $f_{L1,j}$ is evaluated as per (36), where $f_{L2,ij}$ is the fitness value of the i th L2 chromosome that selects the j th L1 chromosome. If the L1 chromosome is unchanged during last generation, $f_{L1,j}$ ' is its fitness value in the previous generation; otherwise, $f_{L1,j}$ ' is set to 0. In this way, $f_{L1,j}$ keeps track of the best fitness the j th L1 chromosome ever achieved.

$$f_{L1,j} = \max(\max_i(f_{L2,ij}), f_{L1,j}')$$
(36)

If a L1 chromosome is not selected by any L2 chromosome, the assignment of a fitness value of zero or a very low value is inappropriate. A specialized fitness assignment scheme for evaluating the new fitness value f_j of these L1 chromosomes is devised as per (37), where $f_{\min}$ and $f_{\max}$ are the minimum and maximum fitness in the current generation. A small α is preferred to assign the unevaluated chromosomes a fitness value slightly higher than the worst chromosome. The rationale for (37) is as follows. A certain L1 chromosome that is never selected in this generation may have participated in forming a very good solution in previous generations. In this case, the former fitness value of the chromosome f_j' is greater than 0 and should be kept as its new fitness value. Furthermore, this L1 chromosome may be newly evolved with initial fitness of zero and hence has not been evaluated. Still, these new chromosomes should be assigned a proper fitness value, so that they are given proper opportunity to be selected and evaluated in future generations. This fitness assignment scheme prevents the HCGA from eliminating good chromosomes and newly generated chromosomes too early in the coevolution stage.

$$f_j = \begin{cases} f_j', & f_j' > 0 \\ f_{\min} + \alpha(f_{\max} - f_{\min}), & \text{otherwise} \end{cases}$$
(37)

5.2.4 Settings of HCGA

The operators and parameters for L2 and L1 chromosomes are listed in Table VII, where I is the number of input variables, S is the maximum number of shoulder points for an ISMF, N is the maximum number of possible ISMFs. The settings of the 2-Level HCGA are further discussed as follows.

- For maintaining the balance between genetic diversity and search efficiency, a moderate size of 30 and 80 is selected for L2 and L1 populations respectively.
- Because the integer encoding is used in L1 populations, the search capability of crossover operator is reduced. To increase the chance of generating new solutions

and introducing new genetic material, the mutation rate is set to 10%. Several different combinations of the crossover rate ranging from 50% to 90% and the mutation rate ranging from 1% to 10% are selected in the exploratory experiments. Results show the system accuracy is not very sensitive to the rate selection.

- In the hierarchical coevolutionary model, the L2 population is randomly initialized. The L1 populations adopt specialized initialization method. For each L1 chromosome, the first n pivots are evenly positioned on the input domain. The parameter n is randomly selected within the range of $[2, N]$. Therefore, the L1 chromosomes are initialized to represent fuzzy partitions containing 2 to N ISMFs. This provides a better start point for evolution than blind random initialization.
- The number of maximum generations $G_{\max}$ is an important parameter specifies when the evolution process of HCGA should terminate. This parameter is heuristically selected within the range of $[500, 5000]$, according to the complexity of the problem. For each problem, a proper maximum number of generations will be chosen after trials.

Table VII The settings of the parameters in the 2-level HCGA

	L1 population	L2 population
Num of population(s)	I	1
Population size	80	30
Len of chromosomes	$N(4S - 1) - 4S - 2$	I
Coding method	Real	Integer
Content of genes	real number within $[0, 1]$	integer number within $[0, 80]$
Crossover rate	80%	80%
Mutation rate	1%	10%
Crossover operator	two-point crossover	two-point crossover
Mutation operator	random mutation	random mutation
Number of elite(s)	1	1

5.3 Experimental validation

In this section, five experiments are carefully selected to demonstrate the search efficiency of the 2-level Hierarchical Coevolutionary Genetic Algorithm. As an enhancement to the previous conventional GA-based approach, the ISMF generation

approach with the 2-level HCGA is able to automatically select informative input variables and derive proper number of ISMFs. Therefore, both approaches are benchmarked against each other in these experiments. The datasets used in the experiments include some datasets used in the previous chapter (Pima diabetes diagnosis, phoneme classification and automobile MPG prediction) and two new datasets (Ionosphere classification and traffic flow prediction).

5.3.1 Experimental settings

The general experimental procedures and settings are briefly described as follows. The previous and the coevolutionary approaches for ISMF generation are benchmarked based on the performance of the resulting fuzzy rule-based system. The fuzzy rule-based system adopting the ISMFs generation approach in the previous chapter is denoted as M1. Because the number of linguistic terms for each input needs to be predefined in M1, the M1 with 2, 3 or 4 linguistic terms per input are denoted as M1T2, M1T3 and M1T4, respectively. The FRBS adopting the hierarchical coevolutionary approach for ISMF generation is denoted as M2. During the fitness evaluation process, the fuzzy rule base can be induced by the rule learning techniques reviewed in [83] after the candidate ISMF partitions are decoded. In the experiments, the Pseudo Outer-Product rule learning algorithm [16] is adopted for both M1 and M2 because of its efficiency in deriving fuzzy rule base as mentioned in Section 5.2.3. The resultant fuzzy rule-based system is then evaluated using training dataset. When the evolution finishes, the best fuzzy rule-based system is then tested on the testing dataset. For each experiment case, the training and testing process is conducted 30 times on a PC with an Intel 2.66GHz CPU. With all other components, train-test groups and rule learning technique in M1 and M2 kept identical, a more accurate and concise fuzzy rule-based system can be considered the result of adopting the more accurate and appropriate fuzzy partitions.

5.3.2 Pima diabetes diagnosis

The dataset for Pima diabetes diagnosis is the same used as a benchmarking problem in Section 4.3.2. In this section, the whole dataset is randomly partitioned into 3 different

train-test groups (50% and 50%), which are denoted as Div1-3 respectively. The number of maximum generation for both approaches is set to 2000. The results of this experiment are listed in Table VIII. The “Avg” column shows the average and standard deviation of the results on all train-test groups. The “Best Avg” column shows the average of the best results found on all train-test groups, which is denoted as “best average” in short in the rest part of the section. The rows “ Δ M1T2” to “ Δ M1T4” show the change in accuracy and rule base size when comparing M2 with M1 fuzzy systems as a percentage of the latter.

In addition to accuracy and number of rules, the average rule length, which is represented by the number of antecedent conditions, is reported to demonstrate that the proposed 2-level HCGA is able to not only reduce the number but also the average length of the fuzzy rules. Because the approach for learning M1 uses all the input features, the rule length of the M1 is fixed and equals to the number of input variables in the Pima dataset, which is 6. The average rule length in M2 is 3.80 ± 0.42 and 36.67% shorter than that in M1. The average rule length in the best M2 on each training-testing group, denoted as best average rule length hereafter, is 3.30 ± 0.48 and 45% shorter than that in M1.

Table VIII Results on Pima diabetes diagnosis

	Avg		Best Avg	
	Accuracy(%)	Rules	Accuracy(%)	Rules
M1T2	73.08 ± 1.52	43.32 ± 9.31	74.74 ± 0.89	41.33 ± 4.04
M1T3	71.81 ± 1.61	151.88 ± 18.19	73.58 ± 1.05	155.33 ± 16.74
M1T4	65.67 ± 2.47	237.38 ± 35.50	68.76 ± 0.55	250.67 ± 22.37
M2	75.23 ± 1.06	14.69 ± 5.57	76.78 ± 1.01	15.00 ± 3.02
Δ M1T2	+2.94%	-66.09%	+2.73%	-63.71%
Δ M1T3	+4.76%	-90.33%	+4.35%	-90.34%
Δ M1T4	+14.56%	-93.81%	+11.66%	-94.02%

5.3.3 Phoneme classification

The ELENA Phoneme dataset is used as a benchmarking problem in Section 4.3.3. In this section, it is used to benchmark two approaches on an unbalanced dataset with large number of instances. Because of the abundance of the data, smaller portion of the data is

used in training phase and larger portion is used in testing compared with previous experiment. The dataset is randomly partitioned into 3 different train-test groups (20% and 80%), which are denoted as Div1-3 respectively. The number of maximum generation for both models is set to 2000. The results of this experiment are presented in Table IX. The rule length in all M1 is 5. The average rule length and best average rule length in M2 is 3.60 ± 0.51 and 3.50 ± 0.53 , respectively. In M2, the average rule length is shortened by 40% when it is compared to M1.

Table IX Results on phoneme classification

	Avg		Best Avg	
	Accuracy(%)	Rules	Accuracy(%)	Rules
M1T2	77.47 ± 1.69	21.46 ± 6.28	78.72 ± 0.49	17.33 ± 4.51
M1T3	77.53 ± 1.28	66.20 ± 19.95	78.73 ± 0.46	68.00 ± 13.23
M1T4	77.52 ± 1.69	116.29 ± 23.54	78.53 ± 1.29	118.67 ± 12.34
M2	79.54 ± 0.74	15.07 ± 8.04	80.64 ± 0.15	15.33 ± 8.33
Δ M1T2	+2.67%	-29.78%	+2.44%	-11.54%
Δ M1T3	+2.59%	-77.24%	+2.43%	-77.46%
Δ M1T4	+2.61%	-87.04%	+2.69%	-87.08%

5.3.4 Ionosphere classification

This experiment is used for benchmarking the previous approach and hierarchical coevolutionary-base approach on the capability of handling high-dimension problems. This dataset was collected by a radar system in Goose Bay, Labrador and is obtainable from the UCI Machine Learning Repository [187]. A total of 351 received radar signals in this dataset are described by 34 features. However, only 33 are used since the value of the second feature is always zero. Each received radar signal is classified to be good or bad based on whether it shows evidence of some type of structure in the ionosphere. The number of maximum generation for both approaches is set to 3000. The classification results of these models are shown in Table X. The rule length in all M1 is 33. The average rule length and best average rule length in M2 is 17.40 ± 0.97 and 16.50 ± 0.71 , respectively. In M2, the average rule length is shortened by about 50% when it is compared to M1.

Table X Results on ionosphere classification

	Avg		Best Avg	
	Accuracy (%)	Rules	Accuracy (%)	Rules
M1T2	81.66 ± 1.47	174.91 ± 4.34	83.96 ± 2.49	177.0 ± 8.72
M1T3	72.90 ± 1.97	201.0 ± 5.05	75.47 ± 2.83	203.33 ± 4.04
M1T4	71.35 ± 1.73	228.74 ± 4.11	74.84 ± 1.44	233.33 ± 1.53
M2	83.16 ± 2.07	164.43 ± 3.65	86.79 ± 4.99	156.67 ± 19.55
Δ M1T2	+1.84%	-5.99%	+3.37%	-11.49%
Δ M1T3	+14.07%	-18.19%	+15.0%	-22.95%
Δ M1T4	+16.55%	-28.11%	+15.97%	-32.85%

5.3.5 Automobile MPG prediction

In this experiment, both benchmarked approaches are used to evolve the fuzzy partitions in the fuzzy systems for solving a regression problem, automobile MPG (miles per gallon) prediction, which is also used in the experiment in Section 4.3.4. The whole dataset is randomly partitioned into 3 different train-test groups (33% and 67%), which are denoted as Div1-3 respectively. For this problem, the number of maximum generations is set to 1000. The mean squared error (MSE) is used as the accuracy index because the fuzzy systems are used to minimize the estimation error instead of trend prediction.

Table XI Results on automobile MPG prediction

	Avg		Best Avg	
	MSE	Rules	MSE	Rules
M1T2	11.94 ± 1.66	14.46 ± 3.04	10.40 ± 0.99	13.58 ± 2.04
M1T3	12.49 ± 1.74	42.62 ± 3.18	11.24 ± 0.41	44.00 ± 4.36
M1T4	18.38 ± 2.15	66.13 ± 6.66	16.49 ± 1.53	67.33 ± 3.21
M2	9.50 ± 1.28	13.36 ± 2.21	8.67 ± 0.56	11.67 ± 3.06
Δ M1T2	-22.44%	-7.61%	-16.63%	-14.06%
Δ M1T3	-23.94%	-68.65%	-22.86%	-73.48%
Δ M1T4	-48.31%	-79.80%	-47.42%	-82.67%

The results of this experiment are summarized in Table XI. The rule length in all M1 is 6. The average rule length and best average rule length in M2 is 4.80 ± 0.79 and 4.70 ± 0.67 , respectively. Compared to M1, the average rule length in M2 is about 20% shorter.

5.3.6 Traffic flow prediction

The traffic flow prediction experiment is taken to benchmark the previous approach and hierarchical coevolutionary-base approach in solving real-world applications. The data were collected from a five-lane section along the east-bound Pan Island Expressway in Singapore using loop detectors embedded beneath the road surface [190]. There are a total of 5 experiment cases devised to extensively benchmark the modeling and prediction capability of M1 and M2. The time t and the density of the three straight lanes were used to predict the density of Lane 1 at the moment of $t+\tau$, where τ is 5, 15, 30, 45 or 60 minutes. Three train-test groups are adopted in each case, each group comprising 550 data points in the training set and 821 data points in the testing set. The number of maximum generation for both models is set to 3000. The R^2 is used as the accuracy index. The mean and standard deviation of the best prediction results found on 3 train-test groups are given in Table XII. The length of the rules in all M1 is 4. For all 5 experiment cases with different τ , the average rule length in M2 is about 3.5, which is about 12.5% shorter than that in M1.

The actual traffic flow density time series on Lane 1 at 5 minutes later and the predicted time series by M2 are illustrated in Figure 23(a). The squared prediction error is plotted in Figure 23(b).

Table XII Prediction results of the traffic flow density on Lane 1

	M1T2		M1T3		M1T4		M2	
	R^2	Rules	R^2	Rules	R^2	Rules	R^2	Rules
5'	0.847 ± 0.037	7.73 ± 0.64	0.818 ± 0.049	22.55 ± 2.36	0.824 ± 0.033	32.00 ± 1.09	0.865 ± 0.034	7.05 ± 1.02
15'	0.790 ± 0.054	8.93 ± 0.38	0.780 ± 0.058	20.96 ± 1.94	0.780 ± 0.045	33.81 ± 3.54	0.809 ± 0.055	11.30 ± 3.39
30'	0.763 ± 0.064	9.12 ± 1.30	0.746 ± 0.063	20.33 ± 2.98	0.731 ± 0.055	36.72 ± 3.89	0.778 ± 0.059	10.03 ± 1.38
45'	0.727 ± 0.054	8.70 ± 1.05	0.713 ± 0.060	19.25 ± 1.04	0.703 ± 0.047	36.31 ± 6.08	0.739 ± 0.056	10.37 ± 0.29
60'	0.685 ± 0.064	9.40 ± 1.45	0.693 ± 0.054	19.42 ± 1.25	0.669 ± 0.049	36.05 ± 4.99	0.714 ± 0.064	9.67 ± 0.81

Figure 23 Prediction of the traffic density on Lane 1 at 5 minutes later by M2

5.3.7 Analysis and discussions

Performance by M1

For the results on Pima diabetes diagnosis shown in Table VIII, the performance of M1 drops with lower accuracy and larger number of fuzzy rules when the number of ISMF on each input increases. The best M1 is M1T2 which achieves the average accuracy of 73.08% with 43 rules and the best average accuracy of 74.74% with 41 rules. When the number of ISMF per input increases to 4, the average accuracy of M1T4 drops to 65.67% and the average rule number drastically increases to 237. The average best accuracy of M1T4 drops to 68.76% with about 250 rules, which is about 6 times of the rules in M1T2. For the results on ionosphere classification as shown in Table X, the accuracy of M1T4 drops about 10% with a 32% increase in rule base size when it is compared with M1T2.

A slightly different scenario is observed from the results on phoneme classification as shown in Table IX. All the M1 yield similar accuracy around 77%. However, M1 still suffers a performance drop when the number of ISMF increases because M1T3 and M1T4 use roughly 3 times and 6 times more rules than M1T2 to achieve a similar accuracy.

For the results on automobile MPG, shown in Table XI, the average MSE of the best M1 increases from 10.40 to 16.49 and the average rule number increases from 13.58 to 67.33 when the number of ISMFs per input increases from 2 to 4. This shows that 2 ISMFs per input variable is enough to describe the data distribution of this dataset. For the results on traffic flow prediction, shown in Table XII, no M1 is able to consistently excel other M1 in accuracy in all experiment cases. M1T2 yields slightly higher accuracy than other M1 in the experiment cases of predicting the traffic flow density at $\tau = 5, 15, 30, 45$ minutes later. In the experiment case of $\tau = 60$, M1T3 achieves 0.008 higher in R^2 than M1T2. However, the fuzzy rule base of M1T3 consists of about one time more rules than those used in M1T2.

The results discussed above are caused by two main deficiencies of the approach in the previous chapter. First, not all input variables in a dataset are equally informative and contributive to the final classification accuracy. As mentioned before, this causes great difficulties for fuzzy system designers to decide the proper number of MFs for each input variable. When allocating more MFs, the size of the fuzzy rule base drastically grows as shown in all discussed experiments. If many MFs are allocated on those less informative variables, the complexity and difficulty for the learning technique increases, which can cause a deterioration of system accuracy. Furthermore, this problem is more obvious for the datasets with many input variables. This is the main reason for the larger performance drop of M1 for Pima dataset than on phoneme, which is the dataset with less input variables. Second, the positions of pivots in the ISMFs are confined to corresponding intervals of the input variable domain. Therefore, the allocation of ISMFs is roughly restricted to the evenly partitioned intervals which may not be suitable for the actual data

distribution. This limitation reduces the flexibility and accuracy of ISMF fuzzy partition which is derived by the previous approach. Both of these limitations in the previous conventional GA-based approach are solved in the 2-Level HCGA-based approach proposed in this chapter.

Performance by M2

Compared with M1, M2 is able to consistently yield superior accuracy with a more concise fuzzy rule base in all experiments. For the results on Pima, shown in Table VIII, M2 manages to achieve an average performance of 75.23% accuracy with 14.69 rules. The average of the best M2 on all train-test groups reaches an accuracy of 76.78% with 15.00 rules. The row “ $\Delta M1T2$ ” shows that M2 is able to yield about 3% increase in accuracy with over 60% fewer fuzzy rules when it is compared with the best M1. The row “ $\Delta M1T4$ ” shows that M2 is able to yield over 11% increase in accuracy with over 93% fewer fuzzy rules compared with M1T4. Similar results on phoneme can be found in Table IX, where M2 gains 2.44% to 2.69% increase in accuracy with 11.54% to 87.08% fewer rules when compared with M1. For the results on ionosphere, shown in Table X, M2 boosts the accuracy for about 2% to 16% using about 11% to 32% less rules when compared with M1. For handling this problem with largest number of input variables used in the experiments, the 2-level HCGA in the proposed ISMF generation approach coevolves 33 L1 populations and 1 L2 population. The results show that the 2-level HCGA is able to efficiently co-evolve these populations simultaneously and yield good learning result. For the automobile MPG, M2 achieves about 16% smaller mean squared error with 14% fewer rules than the best M1 as shown in Table XI. According to the traffic flow prediction results which are listed in Table XII, M2 consistently gains an increase of about 0.02 in R^2 than the best M1 in all the experiment cases, although M2 uses slightly more rules for some experiment cases.

Besides increasing the accuracy and minimizing the number of rules, the approach based on 2-level HCGA is able to shorten the average rule length, especially for those datasets with high input dimensions, such as the Ionosphere dataset in Section 5.3.4. In all the experiments in Section 5.3, the average rule length of M2 is shorter than M1 by about

12% to 50%.

Analysis of the fuzzy partitions

In order to closely examine the ISMF partitions generated by both of the benchmarked approaches, the derived ISMFs in M1T2 and M2 for Pima diabetes diagnosis are assigned with linguistic terms and illustrated in Figure 24. As shown in Figure 24, no ISMF is derived on the first, the third and the eighth input variable when the hierarchical coevolutionary ISMF generation approach is adopted. This shows that the chromosomes in the L1 populations representing these input variables did not cooperate well with other populations during the coevolution process and did not contribute much to the fitness of the candidate solutions they participated in. That means these input variables are less informative for classifying the Pima diabetes cases than other input variables. For the second, the fourth, the fifth and the seventh input variables, both approaches derived 2 ISMFs to present the data distribution. For the sixth input variable, the coevolutionary-based approach evolved a more refined fuzzy partition formed by 3 ISMFs instead of 2 as on some other input variables. This shows that the coevolutionary-based ISMF generation approach is able to automatically allocate proper number of ISMFs for the input variable domains.

Figure 24 The ISMF fuzzy partitions in M1T2 and M2 in the Pima diabetes diagnosis

Together with the Figure 24 discussed above, the results on all experiments demonstrate that the 2-level HCGA is able to select the most informative input variables and derive appropriate number of ISMFs for them. The derived ISMFs form accurate and concise ISMF partitions which consequently result in a more concise fuzzy rule base.

Analysis of the efficiency of HCGA

Besides the performance of the derived ISMF partitions, the efficiency of the 2-level HCGA is also investigated in this subsection.

Table XIII The average training time (in seconds) of the different fuzzy systems in the experiments

	Pima	Phoneme	Ionosphere	AutoMPG	Traffic
M1T2	375.8	498.4	2350.2	46.3	267.5
M1T3	899.2	1042.0	2596.2	81.8	424.5
M1T4	1225.7	1862.4	2722.2	102.8	551.5
M2	547.9	668.8	1611.0	56.8	338.5
Δ M1T2	+45.8%	+34.2%	-31.5%	+22.7%	+26.5%
Δ M1T3	-39.1%	-35.8%	-38.0%	-30.6%	-20.2%
Δ M1T4	-55.3%	-64.1%	-40.1%	-44.7%	-38.6%

As shown in Table XIII, a longer average training time is required for M1 as the number of ISMFs increases. In Pima diabetes and phoneme classification, the training time for M1 is nearly tripled when the number of ISMF per input variable increases from 2 to 4. However, only an increase of 15.8% in training time is observed when comparing the M1T4 to M1T2 in ionosphere classification. When FS-ISMF is achieving about 4% to 34% increase in accuracy, the increase in the training time is in an acceptable range of 5% to 17%. This shows that additional computational cost is required for tuning more parameters (more ISMFs) in the chromosomes. Furthermore, when more ISMFs are adopted, the learned fuzzy rule base becomes more complex as shown in the experimental results. As a result, higher computational cost is required in rule learning and fitness evaluation, both of which form portions of the total training time. Therefore, the appropriate number of ISMFs helps to optimize the system performance not only in increasing accuracy and interpretability but also in shortening the total training time.

Compared with M1T2, the average training time for M2 is about 31% shorter in Ionosphere classification, whereas it is about 22% to 45% longer in other experiments. For instance, a total of 9 populations are evolved in the 2-level HCGA for M2 in Pima diabetes diagnosis, whereas M1T2 is evolved by a single-population GA. However, the average training time for M2 is only about 45% longer than M1T2. It clearly shows that the 2-level HCGA is able to efficiently evolve a number of populations with an acceptable increase in the computational cost. Furthermore, the 2-level HCGA can help to reduce the total training time. Compared with M1T3 and M1T4, the average training time for M2 is about 20% to 64% shorter in all experiments.

Effects of adopting ISMFs in a FRBS

In Chapter 4, the irregular-shaped membership functions are proposed and benchmarked against commonly used types of membership functions. In this chapter, the proposed HCGA is applied to automatically derive the proper number of ISMFs for each input variable.

Compared with commonly used membership functions, the ISMFs comprise more sampling points and are thus more flexible, in that they can better model the underlying data distribution. This in turn increases the accuracy of fuzzy rule-based systems, as shown in the experimental results reported in Section 4.3 and Section 5.3.

By adopting the HCGA approach to derive the ISMFs, the resultant partitions generally consist of fewer MFs than those fuzzy partitions formed by common type MFs. As a result, using ISMFs also helps to shorten the average rule length and reduces the number of rules in the resultant FRBS, as demonstrated in Section 5.3.

Besides the above-mentioned advantages, there are two limitations in adopting ISMFs in a FRBS. First, although repair schemes are applied to maintain the interpretability, the ISMFs still look more complex and are more difficult to interpret than common type MFs. Second, the ISMFs are generated by evolutionary algorithms and therefore require more computational cost than common type MFs.

According to the previous analysis, the positive and negative effects of using the ISMFs on fuzzy rule-based system are summarized in Table XIV.

Table XIV The positive and negative effects of adopting ISMFs in a FRBS

Positive effects	Negative effects
<ul style="list-style-type: none"> • Increases the conciseness of the fuzzy partitions • Shortens the average length of the rules in the FRBS • Reduces the number of rules in the FRBS • Increases the accuracy of the FRBS on training and testing datasets 	<ul style="list-style-type: none"> • Slightly reduces the interpretability of the individual MFs • Requires high computational cost

Summary

In this section, the experimental results demonstrate that the Hierarchical Coevolutionary Genetic Algorithm is able to properly divide the task of generating the ISMFs partitions into smaller subtasks of finding proper membership functions for each input variable. All these subtasks are efficiently coevolved and optimized by separate populations within the HCGA. The resultant fuzzy rule-based systems achieved high accuracy while maintaining a concise fuzzy rule base. In Chapter 6, we will further discuss the details of adopting the HCGA in the HiCEFS approach as the main learning mechanism for synthesizing fuzzy decision support systems.

Chapter 6

Synthesizing Fuzzy Decision Support Systems with the HiCEFS Approach

6.1 Introduction

To address the issue of decision support system synthesis, this thesis proposes a Hierarchical Co-Evolutionary Fuzzy system Synthesis (HiCEFS) approach for synthesizing from data a fuzzy decision support system with well balanced accuracy and interpretability while requiring minimal human intervention. An overview of the HiCEFS approach was given in Chapter 3, while its two salient features, the Irregular-Shaped Membership Functions (ISMF) and the Hierarchical Coevolutionary Genetic Algorithm (HCGA), were discussed in Chapter 4 and Chapter 5, respectively. In this chapter, the details of synthesizing fuzzy decision support systems with the HiCEFS approach are presented.

Specifically, Section 6.2 introduces the method to represent possible fuzzy decision support systems (FDSS) in a five-level HCGA, Section 6.3 elaborates the coevolutionary process in the HiCEFS approach for synthesizing a FDSS, and Section 6.4 describes the process of decoding the FDSS from the chromosomes from different populations in the HCGA, Section 6.5 details the genetic operators and settings adopted in the HCGA, Section 6.6 introduces a fuzzy rule discovery technique devised to facilitate the rule learning in HiCEFS, and Section 6.7 describes the scheme devised for evaluating candidate solutions and assigning proper fitness to the chromosomes in HCGA. Lastly, in Section 6.8, the performance of the HiCEFS approach is evaluated and demonstrated through applying it to synthesize fuzzy decision support systems for real-world applications in different areas.

6.2 Representing possible FDSS in HCGA

The Hierarchical Coevolutionary Genetic Algorithm coevolves the candidate solutions in a divide-and-conquer manner. Depending on the problem being solved, the total number

of levels in HCGA may vary from 2 to 5. For instance, a 2-level HCGA was adopted to derive the ISMFs in a fuzzy decision support system in Chapter 5. In this chapter, the parameter $Lmax$ in HCGA is set to 5 so that the ISMFs and fuzzy rule base in FDSS are simultaneously coevolved in several populations in a 5-level HCGA. The structures and relations of the chromosomes in these populations are depicted in Figure 25.

Figure 25 The structure of the 5-level HCGA for coevolving a FDSS

Each Level-1 (L1) chromosome ch_{L1} in the i th L1 population represents the fuzzy partition on i th input variable, which is formed by several ISMFs. The $ISMF_n$ in ch_{L1} represents a genetic segment, which encodes the positions of all the sampling points within the n th ISMF on the variable, as elaborated in Section 4.2.1.

Each Level-2 (L2) chromosome ch_{L2} represents the fuzzy partition table that consists of the fuzzy partitions encoded in L1 chromosomes. The genes FP_i in a ch_{L2} encode the indices of L1 chromosomes and act as selectors of the fuzzy partitions encoded in the corresponding L1 chromosomes. The content of FP_i is within the range of $[0, SP_1]$, where SP_1 is the size of a L1 population. For instance, if $FP_i=3$, the 3rd chromosome in the i th L1 population will be selected to form the fuzzy partition for the i th input variable. If $FP_i=0$, the i th input variable will be discarded. The L2 and L1 populations handle the subtask of evolving the fuzzy partition table of the fuzzy decision support system, which forms the *condition layer* in Figure 8.

Each Level-3 (L3) chromosome ch_{L3} encodes a Mamdani-type fuzzy rule. These fuzzy rules are represented as combinations of linguistic terms, which are related to corresponding fuzzy membership functions. The genes in a ch_{L3} are used to select the fuzzy membership functions, which are encoded in a ch_{L1} . Among the first I genes, the i th one XT_i represents the index of the membership function on the i th input variable. Among the remaining Z genes, the z th gene YT_z represents the index of the membership function on the z th output variable. The integer coding method is not suitable for directly representing the index of the membership function because the numbers of membership functions on each input variable are constantly changed and evolved by the L2 and L1 populations. For instance, assume that at a certain generation, there are three membership functions, representing *slow*, *medium* and *fast*, on the first input variable *Speed* and the first gene in a ch_{L3} is 3. It means that the condition part of the corresponding fuzzy rule contains “*Speed* IS *fast*”. Assume that after the populations are evolved into next generation, there are only 2 membership functions on the variable *Speed*, the first gene in the ch_{L3} with a content of 3 is no longer related to any valid membership function and hence the ch_{L3} is not a valid fuzzy rule. Therefore, instead of using integer coding, each gene in a ch_{L3} is a real number within the range of $[0, 1]$ to ensure that each ch_{L3} always represents a valid fuzzy rule. More details on mapping these real numbers to the indices of the membership functions are given in Section 6.4. The length of a Level-3 chromosome equals the number of inputs plus the number of outputs.

Each Level-4 (L4) chromosome ch_{L4} encodes a fuzzy rule base formulated from the candidate rules in the L3 population. Similar to its counterpart in ch_{L2} , each gene FR_r in a ch_{L4} selects a fuzzy rule by encoding the index of a ch_{L3} within the range of $[0, SP_3]$, where SP_3 is the size of the L3 population. If $FR_r=0$, no fuzzy rule is selected by this gene. The length of a Level-4 chromosome is set to be the maximum number of rules.

Each Level-5 (L5) chromosome ch_{L5} denotes an entire fuzzy decision support system with genes representing the selected fuzzy partition table and fuzzy rule base. In a ch_{L5} , the first gene FPT selects a fuzzy partition table by evolving the index of a ch_{L2} within the range of $[1, SP_2]$; the second gene FRB selects a fuzzy rule base by evolving the index of a ch_{L4} within the range of $[1, SP_4]$, where SP_h denotes the population size of a h th level population. Thus, the minimal length of a Level-5 chromosome is 2. Besides these first 2 genes, the fuzzy inference operators and the defuzzification operator can also be represented as Inf_1 , Inf_2 and Def in a ch_{L5} , which forms a future extension as described in Section 7.3.

Thus, a candidate FDSS is represented by a set of chromosomes from different level populations.

6.3 Coevolutionary synthesis process

The evolutionary process of the HCGA for evolving the whole FDSS is illustrated in Figure 26. Because the HiCEFS approach of synthesizing a FDSS is based on the evolutionary process of the generic HCGA shown in Figure 20, only the differences are pointed out, as follows. Note that the definitions of the symbols which also appear in Figure 20 are given in Section 5.1.

In Step 1 of the coevolution process, the number of population levels $Lmax$ is set to 5 for generating the ISMFs and fuzzy rule base simultaneously. Other parameters and genetic operators including number of populations NP_h , size of population SP_h , the set of genetic

Figure 26 The evolutionary process of the 5-level HCGA for the synthesis of a FDSS

maximum generation $G_{\max}$ are initialized, the settings of which are described in Section 6.5. Two new parameters, interval T_s and T_r , are added to overcome the fitness stagnation and discover candidate fuzzy rules. All the populations are initialized and the generation G is set to 1.

As described in Step 2 to Step 8 in Section 5.1, the candidate fuzzy decision support

systems are first decoded from the chromosomes in HCGA. Then, these candidate systems are evaluated and proper fitness values are assigned to the chromosomes. Subsequently, genetic operators are applied to these populations and evolve into next generation. The coevolution process terminates when the stopping criterion is reached. However, some special improvements are added in this 5-level HCGA for evolving FDSS as discussed hereafter.

Keeping the best chromosomes, which are also called elites, unchanged when evolving towards the next generation is the common method called elitism and adopted in canonical evolutionary techniques to maintain the best solution [34]. However, in HCGA, elites in the lower-level populations are highly likely to be selected by other higher level chromosomes and thus participate in the formation of other candidate solutions. If the canonical elitism is used and these elites cannot be changed, it may deter the possible improvements on these candidate solutions. Therefore, a novel method is proposed to preserve the best chromosomes instead of simply keeping them intact in the next generation. At the end of each generation, the fitness of the best solution in generation G , denoted as $f_{\max}(G)$, is compared with the fitness of the best solution in the previous generation. If $f_{\max}(G) > f_{\max}(G-1)$, the elite chromosomes that participated in forming the FDSS with the highest fitness in current generation are backed up and will not be affected by the genetic operations in evolution process. If $f_{\max}(G) < f_{\max}(G-1)$, the backup elites are restored to the populations at the exact places where they are located in the last generation. Otherwise, if $f_{\max}(G) = f_{\max}(G-1)$, there is no need to backup or restore the elites.

In the following step, a technique for alleviating the evolutionary stagnation [191] is incorporated. If equation (38) is satisfied, that means $f_{\max}(G)$ is not making sufficient improvement for a period of T_s , the chromosomes except the elites are randomly re-initialized. In this thesis, $T_s = G_{\max}/50$ and the desired minimal fitness increment ε is set to 0. The user can even set a higher ε to increase the chance of re-initialization. This mechanism helps to introduce new genetic material and break the fitness stagnation.

$$f_{\max}(G) - f_{\max}(G - T_s) \leq \varepsilon, \text{ where } \varepsilon \geq 0 \quad (38)$$

In the subsequent step, a newly proposed fuzzy rule base discovery process is performed for every predefined interval of T_r generations, as detailed in Section 6.6.

6.4 Extracting the solution from HCGA

Because a possible fuzzy decision support system (FDSS) is represented by a set of chromosomes from different level populations, the extracting process is more complex than extracting a solution from a chromosome in a canonical GA. When extracting a FDSS, the L5 chromosomes ch_{L5} are the first to be extracted because they are from the highest level population and select the chromosomes from lower level populations. The index of the selected ch_{L2} is determined according to the first gene FPT in ch_{L5} . Then, the fuzzy partitions formed by ISMFs in the corresponding L1 chromosomes can be extracted as explained in Section 5.2. After the ISMFs are formed, the numbers of fuzzy membership functions N_i and N_z on each i th input variable and z th output variable respectively are determined.

In the subsequent step, the ch_{L4} is selected according to the second gene FRB in ch_{L5} . Then, its corresponding ch_{L3} will be extracted. As mentioned previously, each gene in a ch_{L4} represents the index of a ch_{L3} . The first step is to extract the unique indices, which means pruning the redundant rules in the ch_{L4} . All the ch_{L3} corresponding to the unique indices are decoded. According to the genes, the index of the membership function can be mapped by using (39), where IXT_i is the index of the ISMF on the i th input variable and IYT_z is the index of the MF on the z th output variable; XT_i is the i th gene and YT_z is the $(I+z)$ th gene. For instance, assume that the i th input variable has a fuzzy partition containing 3 MFs with the linguistic term of $\{low, medium, high\}$, then the index of leftmost ISMF (*low*) is 1 and the index of rightmost one (*high*) is 3. If the gene $XT_i=0.2$, then the $IXT_i=1$ according to (39) and the gene represents “If x_i is *low*”.

$$\begin{cases} IXT_i = \lceil XT_i \cdot N_i \rceil \\ IYT_z = \lceil YT_z \cdot N_z \rceil \end{cases} \quad (39)$$

It is possible that conflicting fuzzy rules are generated that have the same fuzzy premise but a different consequent. Because their premises are the same, these conflicting fuzzy

rules cover the same set of training samples. The accuracy of these rules on the covered training samples can be computed in terms of the number of correctly classified cases or the mean squared error. Thus the rule that provides the highest accuracy will be included in the final fuzzy rule base and the others will be discarded. The fuzzy rule base extraction process keeps the knowledge in the FDSS non-duplicate and non-controversial.

6.5 Settings of the HCGA in HiCEFS

Selection operator

In HCGA, fitness-proportionate selection [34] is used to select the chromosomes for crossover and mutation. It gives those chromosomes with higher fitness a higher chance to participate in crossover and mutation.

Crossover operators

One-point crossover is adopted in the L5 population because of the relative short length of ch_{L5} , while two-point crossover is adopted in the L4, L3 and L2 populations. A hybrid crossover operator which consists of two crossover cases is tailored for the L1 populations. The first case is to exchange the corresponding ISMF genetic segments between two parents. In this case, the hybrid crossover attempts to identify superior fuzzy partition without destroying existing ISMFs. The second case is to fine tune the ISMFs with a variation of arithmetic crossover [112, 171] as per (40), where a_i, b_i are the i th gene of the two parents and a_i', b_i' are the i th gene of the offspring. The factor ξ_i is a normally distributed random variable $N(0, \sigma^2)$ with the variance σ^2 set to 0.2 so that its value most likely falls within the range of $[-1, 1]$ (with a probability of 97.4% according to the probability distribution function). In this case, it achieves balanced exploration and exploitation while keeping all genes in new offspring within the range of $[0, 1]$. When the hybrid crossover is applied, it randomly picks one from these two crossover cases. The results of preliminary trial runs show that the hybrid crossover only slightly increases the computational time when compared with 2-point crossover.

$$\begin{cases} a_i' = a_i - \xi_i(a_i - b_i) - \lfloor a_i - \xi_i(a_i - b_i) \rfloor \\ b_i' = b_i - \xi_i(b_i - a_i) - \lfloor b_i - \xi_i(b_i - a_i) \rfloor \end{cases}, \text{ where } \xi_i \sim N(0, \sigma^2) \quad (40)$$

Mutation operators

Random mutation operators are adopted in populations from all levels. The only difference is that the mutation operators in L5, L4 and L2 replace a gene with a randomly generated integer value within the range of $[1, SP_4]$, $[1, SP_3]$ and $[1, SP_1]$ respectively, while the mutation operator in L3 and L1 replaces a gene with a randomly generated real value within the range of $[0, 1]$.

6.6 Fuzzy rule base discovery process

The task of deriving a small fuzzy rule base with accuracy and interpretability is very difficult especially when dealing with a high-dimensional problem [192]. Thus, in HiCEFS approach, a novel fuzzy rule base discovery process (rule discovery in short) is incorporated into the HCGA, as shown in Figure 26, for facilitating the fuzzy rule base generation. The rule discovery scheme tries to find a better fuzzy rule base than the existing ones after every fixed interval, by deriving a fuzzy rule base for each fuzzy partition table in the Level-2 population with the data-driven rule learning techniques.

The flowchart of this fuzzy rule base discovery process is illustrated in Figure 27, where c_2 is the index of the ch_{L2} , SP_2 is the size of the L2 population, fit is the fitness of the resultant fuzzy decision support system, $fit_{\max}$ denotes the highest fitness found in this discovery process, c_{best} is the index of the ch_{L2} adopted in the best FDSS found in this discovery process, $f_{\max}(G)$ is the highest fitness in the current generation. Many existing rule learning techniques [83] can be adopted in this discovery process. The Pseudo Outer-Product rule learning algorithm [16] is adopted in this discovery process because it is very efficient and already implemented for deriving the rule base in the Section 5.2.3.

Figure 27 Flowchart of the fuzzy rule base discovery process in HiCEFS approach

During the rule discovery, if the fitness of the best found HiCEFS fit_{max} is greater than the highest fitness $f_{max}(G)$ in current generation, the found fuzzy rule base will be encoded in the HCGA. The fuzzy rules are encoded into L3 chromosomes as per (41), where IXT_i is the index of the ISMF on the i th input variable and IYT_z is the index of the MF on the z th output variable; XT_i is the i th gene and YT_z is the $(I+z)$ th gene. For instance, assume that in a FDSS for car braking, there are 5 membership functions (namely *very slow*, *slow*, *medium*, *fast* and *very fast*) on the input *Speed*, 3 (namely *short*, *medium* and *long*) on the input *Distance* and 2 (namely *weak* and *strong*) on the output *Braking force*. The Mamdani rule “IF *Speed* IS *fast* AND *Distance* IS *short* THEN *Braking force* IS *strong*” can be encoded into a string of “0.7, 0.33, 0.25” using (41).

$$\begin{cases} XT_i = (IXT_i - 0.5) / N_i \\ YT_z = (IYT_z - 0.5) / N_z \end{cases} \quad (41)$$

The chromosomes with the lowest fitness in the L3 population are replaced with these newly encoded ch_{L3} . The indices of these newly replaced L3 chromosomes are encoded into a ch_{L4} , which replaces the worst chromosome in the L4 population. Finally, a ch_{L5} that selects the newly encoded L4 chromosome and (c_{best})th L2 chromosome will replace the chromosome with the lowest fitness in the L5 population.

6.7 Fitness evaluation of the solution

In the fitness evaluation process, all the chromosomes in the L5 population of the HCGA are evaluated. The selected chromosomes from lower level populations are determined according to the decoding process described in Section 6.4. The fitness values of these chromosomes are set according to the accuracy and interpretability of the fuzzy decision support system they participated in. The accuracy of the corresponding FDSS f_{Acc} is measured by comparing the system outputs with the desired outputs and calculated as per (29) and (30). The interpretability of the FDSS is measured by the number of fuzzy rules. The accuracy is the first objective, while the interpretability is used as the second criterion to measure two systems with similar accuracy.

Evolutionary multiobjective optimization (EMO) [193, 194] is a promising technique for search a set of non-dominant fuzzy systems with different tradeoff between accuracy and interpretability [80, 139]. However, evolutionary multiobjective optimization is not suitable for the HiCEFS approach because of the following reasons. First, using EMO further increases the high computational cost in HCGA. Second, it is very difficult for the user to choose a proper FDSS for practical decision making from the set of candidate solutions derived by EMO.

The HCGA adopts a weighted sum fitness function as per (42), where f_s is the fitness value of the candidate solution, num_r is the number of rules in the derived FDSS, W_r is the weight of the number of rules. The aim of improving the accuracy of the FDSS while

keeping the number of rules low is achieved by simply setting a small positive number for W_r in (42). The default setting is $W_r=0.1$. However, the designer can change the setting to achieve a different balance between these two objectives. Subsequently, the f_S is assigned to the ch_{L5} as in (43).

$$f_S = f_{Acc} - W_r \cdot num_r \quad (42)$$

$$f_{L5,j} = f_S \quad (43)$$

Among the lower level chromosomes, the ch_{L4} and ch_{L2} are directly selected by ch_{L5} . Similarly, the ch_{L3} and ch_{L1} are selected by ch_{L4} and ch_{L2} , respectively. All these lower level chromosomes can be selected more than one time in a certain generation. The fitness of the best solution it participated in should be assigned to the chromosome. In this case, the fitness values of these chromosomes are computed in (44), where $f_{L5,ij}$ is the fitness of the i th ch_{L5} that selects the j th ch_{L4} or ch_{L2} , $f_{L4,ij}$ is the fitness of the i th ch_{L4} that selects the j th ch_{L3} , $f_{L2,ij}$ is the fitness of the i th ch_{L2} that selects the j th ch_{L1} ; f_j is the new fitness value and f_j' is the old fitness value. If the chromosome is not changed during this generation, f_j' equals to its fitness value at previous generation; otherwise, the chromosome is considered as a new one and f_j' is set to 0.

$$\begin{cases} f_{L4,j} = \max(\max_i(f_{L5,ij}), f_{L4,j}') \\ f_{L3,j} = \max(\max_i(f_{L4,ij}), f_{L3,j}') \\ f_{L2,j} = \max(\max_i(f_{L5,ij}), f_{L2,j}') \\ f_{L1,j} = \max(\max_i(f_{L2,ij}), f_{L1,j}') \end{cases} \quad (44)$$

As discussed in Section 5.2.3, it is also possible that some lower level chromosomes are not selected by any higher level chromosomes. Still, they should be assigned a proper fitness value, so that they can be evaluated in following generations. In this case, the fitness values of these chromosomes are computed as per (37).

6.8 Experimental validation and real-world applications

The Hierarchical Co-Evolutionary Fuzzy system Synthesis (HiCEFS) approach can be applied to synthesize fuzzy decision support systems for real-world applications in many different areas. The applications described in this chapter include the stock prediction and trading in finance [195], the Pima diabetes diagnosis in medicine [196], the gamma-ray signal detection in science [197] and the concrete compressive strength prediction in engineering [198].

6.8.1 Experimental settings

The experimental procedures and settings adopted in this chapter are similar to those used for M2, which are described in Section 5.3.1. The HiCEFS approach with a 5-Level HCGA is adopted for synthesizing the fuzzy decision support systems according to the training dataset. Then the best resultant FDSS is tested on testing dataset. Same training and testing procedure is conducted for other benchmarked techniques or systems.

6.8.2 Stock prediction and trading

Financial market prediction and trading presents a challenging task that attracts great interest from researchers and investors because success may result in substantial rewards [199]. Once the market trend is correctly identified or predicted, the investor can make profit by taking and maintaining a corresponding investment position of long (buy) or short (sell) until evidence indicates that the trend has reversed or is going to reverse. However, market trend is still very difficult to predict because it is a highly complex and dynamic system which involves the interactions taken by millions of investors and institutions [200]. Therefore, there is a need for a decision support system that can increase the accuracy in trend prediction and the facilitate investors in making trade decisions.

In this section, a novel financial trading decision support system (FTDSS) is proposed, which uses the fuzzy rule-based system synthesized by the HiCEFS process for predicting

financial market trend and employs a prudent trading strategy. The details of the proposed FTDSS are discussed hereafter.

To construct a FTDSS, the most important issue is to choose a suitable financial analysis method for identifying the market trend. There are generally two kinds of financial analysis: fundamental [201] and technical [202]. Fundamental analysis estimates the intrinsic value (target price) of the stock by analyzing a wide range of quantitative and non-quantitative information, such as the earnings and management of the company and other market news. It is difficult to automate the analysis of the non-quantitative information. Technical analysis only analyzes the historical quantitative data that are easier to obtain, such as the price and volume. Without the need to obtain and analyze the non-quantitative information, the technical analysis can be more conveniently automated and conducted in a decision support system than the fundamental analysis. It is thus used in the proposed FTDSS.

Among the indicators used in technical analysis, the moving averages are commonly used to reveal the trend line by smoothing the cyclical fluctuations in price line. One variation of moving average, exponential moving average (EMA), assigns the larger weights to more recent periods, which is considered more “adaptive” to the evolving dynamics in stock. The d -day EMA at time t represented by $EMA_d(t)$ can be derived from the stock price series using (45), where $y(t)$ is the price of the stock at time t . More importantly, it can be used to identify trend reversals. The market turns bullish or bearish when a short term EMA crosses a long term EMA from below or above, respectively. Based on the percentage difference between a S -day EMA and a L -day EMA ($S < L$), the percentage price oscillator (PPO) is computed using (46). The market turns bullish or bearish when PPO goes above or below zero respectively. The main advantage of PPO over EMA is that its values are comparable across time periods regardless of change in price levels in the long run. Therefore, PPO is adopted in the proposed FTDSS to signal trend reversals.

$$EMA_d(t) = EMA_d(t-1) + \frac{2}{d+1} [y(t) - EMA_d(t-1)] \quad (45)$$

$$\text{PPO}_{s,L}(t) = \frac{\text{EMA}_s(t) - \text{EMA}_L(t)}{\text{EMA}_L(t)} \quad (46)$$

Even without a predictive model, the financial trading decision support system can derive the trading signal $F(t)$ according to the current PPO as per (47), where ε is the width of the whipsaw signal filter. The filter is used here because whipsaw signals are sometimes formed, in which case the PPO will slightly fluctuate around 0 and generate false alarms of bullish reversal in a primary bearish trend, or conversely. According to the trading signal $F(t)$, the FTDSS take long position (buy) when $F(t) = 1$, or short position (sell) when $F(t) = -1$.

$$F(t) = \begin{cases} 1, & \text{when } \text{PPO}_{s,L}(t) > \varepsilon \\ -1, & \text{when } \text{PPO}_{s,L}(t) < -\varepsilon \\ F(t-1), & \text{else} \end{cases} \quad (47)$$

However, the PPO is a trend following indicator with a delay of a certain period after the actual trend reversal. In order to reduce the delay period and undertake prompt trading action, a predictive model derived by HiCEFS approach is adopted in the proposed FTDSS to forecast the following trend by predicting the future PPO. The PPO series is represented in U -tuples $[\text{PPO}(t-U+1), \dots, \text{PPO}(t-1), \text{PPO}(t)]$, where U denotes the embedding dimension. The U -tuples are used as inputs to the predictive model to predict the future PPO at V days later, which is represented as $\text{PPO}'(t+V)$.

Based on the predicted $\text{PPO}'_{s,L}(t+v)$, the trading signal $F(t)$ is generated in the trading model with a prudent trading rule shown in (48), where V is the horizon of the prediction and ε is the width of the whipsaw signal filter. The FTDSS will go long or short if all the latest $\lfloor V/2 \rfloor$ predicted PPO' is greater than ε , or smaller than $-\varepsilon$, respectively. Otherwise, it will just hold the previous trading position. With this modification to the trading strategy, the accuracy in assessing the predicted trend reversal is greatly improved and the unwanted loss caused by inappropriate trades can be reduced, as shown in the experimental results given in the remaining part of this section.

$$F(t) = \begin{cases} 1, & \forall v \in \{V - \lfloor V/2 \rfloor + 1, \dots, V\} : \text{PPO}'_{S,L}(t+v) > \varepsilon \\ -1, & \forall v \in \{V - \lfloor V/2 \rfloor + 1, \dots, V\} : \text{PPO}'_{S,L}(t+v) < -\varepsilon \\ F(t-1), & \text{else} \end{cases} \quad (48)$$

According to the trading signal $F(t)$, the stock price $y(t)$ and the transaction cost rate δ , the multiplicative return $R(t)$ [203] is adopted to measure the final portfolio return of the financial trading decision support systems and can be computed using (49).

$$R(t) = R(t-1) \{1 + r(t)F(t-1)\} \{1 - \delta|F(t) - F(t-1)|\} \quad (49)$$

where $r(t) = \{y(t)/y(t-1)\} - 1$

The overall architecture of the proposed FTDSS is illustrated in Figure 28, the details of which are discussed as above.

Figure 28 The proposed financial trading decision support system with a predictive model derived by HiCEFS approach

In the remaining part of this section, the proposed FTDSS with the predictive model synthesized by HiCEFS approach is adopted to provide trading decision support for the Hong Kong Hang Seng index (HSI). In order to demonstrate the forecasting performance of predictive model synthesized by HiCEFS approach, three well-known NFS's, including the evolving fuzzy neural network (EFuNN) [204], dynamic evolving neural-fuzzy inference system (DENFIS) [89] and rough set-based pseudo outer-product fuzzy neural network (RSPOP) [205], are also used as the predictive model. The performance of the proposed FTDSS is benchmarked against the simple buy-and-hold strategy, the FTDSS without prediction, the FTDSS with perfect prediction, and the FTDSS with other predictive model (EFuNN, DENFIS, RSPOP) using the historical data of HSI.

The settings of the FTDSS are as follows. The width of the whipsaw signal filter ε is heuristically set to 0.1% in (47) and (48). The proper number of parameter U is estimated to be 6 for HSI by adopting the method of “false nearest neighbors” [206]. The parameter V is customized to be 5 for the FTDSS with prediction. For the FTDSS without prediction, the trading signals are computed with (47). For trading systems with a predictive model, the trading signals are generated using (48). For trading system with perfect prediction, the trading signal are computed with (48), but the predicted $PPO'(t+i)$ are replaced with the actual future $PPO(t+i)$. All the trading systems enter the market with a starting portfolio value $R(0)$ of 1.0. According to the price series and the trading signals, the final portfolio value $R(T)$ can be computed using (49), where the transaction cost rate $\delta = 0.2\%$.

The data consists of 4941 daily closing price values obtained from the Yahoo finance website on the counter HSI from the period of Jan 2, 1987 to Oct 31, 2006. The training dataset for predictive models contains the PPO series constructed from 2480 price values before the year 1997. The out-of-sample dataset contains the PPO series constructed from the remaining 2461 price values. The PPO series is generated by using (46). The parameters S and L in (46) are heuristically set to 15 and 45 respectively, because there is no established method to determine these two parameters automatically or optimally.

The out-of-sample price series and the trading signals are shown in Figure 29. The series of portfolio value $R(t)$ for different trading systems are shown in Figure 30. Table XV shows the benchmarking results of different trading systems, including the final portfolio value $R(T)$, square of Pearson correlation R^2 between the actual and predicted PPO series and the number of rules in the predictive model. In Table XV, B&H represents the buy-and-hold strategy; FTDSS-WOP and FTDSS-PP denote the trading system without prediction and with perfect prediction, respectively; the trading system with EFuNN, DENFIS, RSPOP and the FRBS derived by HiCEFS approach are denoted as FTDSS-EFuNN, FTDSS-DENFIS, FTDSS-RSPOP and FTDSS-HiCEFS, respectively. The simple buy-and-hold strategy only achieved a final portfolio value $R(T)=1.424$. The trading system without a predictive model yielded a final portfolio value $R(T)=1.938$. The

trading system with HiCEFS predictive model achieved a final return of $R(T) = 6.326$. Compared with the trading system without prediction, the FTDSS-HiCEFS achieved an increase of 4.388 in $R(T)$. Compared with the trading system using EFuNN, DENFIS and RSPOP, the FTDSS-HiCEFS yielded an increase of 4.220, 1.935 and 3.520 in final portfolio value $R(T)$, respectively. Compared with the EFuNN and RSPOP predictive model, the predictive model derived by HiCEFS approach is able to achieve higher correlation with smaller number of rules. Although DENFIS is able to achieve a similar correlation with only 8 rules, the TSK-type rules in DENFIS are less interpretable than the Mamdani-type rules identified by HiCEFS. Besides the number of rules, the rule length is another measure of interpretability. Because there is no feature selection method used in EFuNN, DENFIS and RSPOP, the rule length in these systems equals to the number of input features, which is 6 in this study case. Comparing to these systems, HiCEFS is able to reduce the rule length by 1 and achieve a better interpretability.

Table XV Benchmarking results of the financial trading decision support systems on HSI

HSI	B&H	FTDSS-WOP	FTDSS-EFuNN	FTDSS-DENFIS	FTDSS-RSPOP	FTDSS-HiCEFS	FTDSS-PP
$R(T)$	1.424	1.938	2.106	4.391	2.806	6.326	22.120
R^2	N.A.	N.A.	0.832	0.905	0.877	0.922	N.A.
Rules	N.A.	N.A.	354	8	57	12	N.A.

Figure 29 Price of HSI and trading signals given by FTDSS-HiCEFS

Figure 30 Portfolio values on HSI achieved by the financial trading decision support systems

Figure 31 Enlarged part of Figure 29 from time $t=1000$ to 1250

The trading signals generated by FTDSS-HiCEFS can be more closely examined on Figure 31, which is the enlarged section of Figure 29 from time $t=1000$ to 1250. As shown in part (a) of Figure 31, it is possible that the trading system with HiCEFS generates delayed trading signals. Similar cases can also be found in the trading signals obtained when using the other predictive models. This is an intuitive result because it is impossible to forecast the financial series with perfect accuracy. Part (b) shows that the trading system with HiCEFS is able to successfully prune some unnecessary trading transactions. Furthermore, the trading system with HiCEFS is able to enter the long buying and short selling position earlier than the trading system without prediction because of a successful forecasting given by the predictive model as shown in part (c) and (d), respectively. The capability of giving more preemptive and less unnecessary or

delayed trading signals is the main reason for the higher revenue earned by the trading system with HiCEFS than the trading systems with other predictive model.

The derived membership functions for the input and output features are shown in Figure 32. With the HiCEFS approach, most MFs are generated on the $PPO(t)$ among all input variables whereas no MF is generated on the input $PPO(t-4)$. It shows that the PPO of today is the most important for predicting the trend of HSI, whereas the $PPO(t-4)$ can be ignored.

Figure 32 The derived membership functions in FTDSS-HiCEFS for HSI

In Table XVI, the derived fuzzy rules in HiCEFS are listed. From these rules, some decision-making logic can be extracted and analyzed.

Table XVI The fuzzy rule base in the FTDSS-HiCEFS for predicting HSI

Rule	$PPO(t-5)$	$PPO(t-3)$	$PPO(t-2)$	$PPO(t-1)$	$PPO(t)$	$PPO(t+5)$
R_1	low	low	low	Low	very low	very low
R_2	high	low	medium	Low	low	very low
R_3	low	low	low	Low	low	low
R_4	high	low	medium	Low	medium	low
R_5	low	low	medium	Low	medium	moderately low
R_6	low	low	medium	Low	low	slightly low
R_7	low	high	medium	Low	low	medium
R_8	low	low	medium	High	medium	slightly high
R_9	high	high	medium	High	medium	moderately high
R_{10}	low	high	high	High	high	high
R_{11}	high	high	high	High	high	high
R_{12}	low	high	medium	High	very high	very high

As shown in these results, the proposed fuzzy trading decision support system with the predictive model derived by the HiCEFS approach outperforms the simple buy-and-hold strategy, the trading system without prediction and the trading system with other predictive models (EFuNN, DENFIS and RSPOP) by providing the more accurate predictions and higher revenue. Furthermore, the decision-making logic in the predictive model derived by HiCEFS approach is succinct and can be easily analyzed by the human users.

6.8.3 Pima diabetes diagnosis

In this section, the HiCEFS approach is adopted to synthesize a fuzzy decision support system for a medical diagnosis problem. The dataset for Pima diabetes diagnosis is the same used as a benchmarking problem in Section 4.3.2 and 5.3.2. The experimental settings used here, including the train-test data partitions and $G_{\max}$, are identical to that of Section 4.3.2 and 5.3.2. Therefore, the improvement in performance of the FDSS derived by the HiCEFS approach can be clearly demonstrated.

The derived FDSS is benchmarked against other well-known statistical and neural techniques, including the k-nearest neighbors (kNN) [207], linear discriminant analysis (LDA) [208], logistic discriminant analysis (logdisc), C4.5 decision tree [9], naïve Bayes (NB) [209] and multi-layer perceptron (MLP) [75], the results by which are obtained from [210]. Furthermore, two neuro-fuzzy systems including the RSPOP [205] and DENFIS [89], one fuzzy classifier derived with an intelligent genetic algorithm-based method (IGA) [174] and a boosted genetic fuzzy classifier (GFC) [211] are also applied to the diabetes diagnosis.

The benchmarking results on this experiment are listed in Table XVII, in which the column “Div” shows how the dataset is partitioned into training-testing groups. The “10-fold CV” represents that the result is obtained by using 10-fold cross-validation [212]. The “50%-50%” indicates that 50% instances in the dataset are randomly chosen as training samples and remaining 50% is used as the testing cases. Among the benchmarked techniques, two statistical methods including LDA and Logdisc are able to achieve high

accuracy of over 77%. The statistical method kNN and neural network MLP achieved slightly lower accuracy of about 72% and 76% respectively. The main drawback of both statistical and neural network methods is that the knowledge in these derived systems can not be represented in a human-interpretable form. The neuro-fuzzy systems including RSPOP and DENFIS achieved relatively low accuracy of around 70%. One drawback of these systems is that they are not able to automatically determine the number of membership functions for each input variable. Even though a rough-set based technique is implemented in RSPOP to prune the redundant fuzzy rules, the system still contains a large fuzzy rule base with about 68 rules. Both the evolutionary-based fuzzy classifiers yield an accuracy of about 75% and the IGA contains only about 4 fuzzy rules.

Table XVII The diagnosis results on the Pima diabetes

	Accuracy (%)	Rules	Div
kNN	71.9	-	10-fold CV
LDA	77.5	-	10-fold CV
Logdisc	77.7	-	10-fold CV
C4.5	73.0	-	10-fold CV
NB	75.3	-	10-fold CV
MLP	76.4	-	10-fold CV
RSPOP	69.97 ± 2.48	68.0 ± 2.0	50%-50%
DENFIS	70.57 ± 1.35	67.67 ± 3.21	50%-50%
IGA	75.19	3.84	10-fold CV
GFC	75.47	20	50%-50%
FDSS	77.83 ± 1.16	5.66 ± 0.58	50%-50%

Compared with other techniques and systems, the FDSS derived by HiCEFS approach is able to provide the highest accuracy of 77.8%. The ISMFs in the FDSS are shown in Figure 33. Among eight input variables, only the plasma glucose concentration and body mass index are selected as inputs for the FDSS. Two and three ISMFs are derived on these variables respectively. It shows the HiCEFS approach is able to derive a proper number of ISMFs for the input variables that are important to Pima diabetes diagnosis. As a result, the average rule length in the FDSS derived by HiCEFS is about 2.89, which is much shorter than 5.03 in IGA, 8 in RSPOP and DENFIS.

Figure 33 The ISMFs in the FDSS derived by HiCEFS approach for Pima diabetes diagnosis

The compact fuzzy rule base in the FDSS consists of six Mamdani-type fuzzy rules, as shown in Table XVIII. Although there are two rules more than IGA, the fuzzy rule base in the FDSS is much smaller than those in RSPOP, DENFIS and GFC. The knowledge in these six rules is interpretable and can be easily analyzed by the experts.

Table XVIII The fuzzy rules in the FDSS derived by HiCEFS approach for Pima diabetes diagnosis

Rule	Plasma glucose concentration	body mass index	class
R_1	high	medium	malign
R_2	low	medium	benign
R_3	low	low	benign
R_4	low	high	malign
R_5	high	high	malign
R_6	high	low	benign

6.8.4 Gamma-ray signal detection

In the fields of observational sciences, such as astronomy and particle physics, one of the most important and challenging task is to extract a small number of interesting signal instances from an overwhelming presence of background signals, both of which have very similar characteristics. With the fast development in detector technology, more and more data is collected and waiting to be analyzed, which yields an ever increasing demand for decision support systems that can automatically analyze these data.

Gamma-ray burst is one the most luminous events in the universe and has been studied for more than three decades [213]. Ground based atmospheric Cherenkov telescopes using imaging techniques are applied to observe high energy gamma-rays. Depending on the energy of the primary gamma, a total of few hundreds to some 10000 Cherenkov

photons are collected in patterns called the shower image, allowing to discriminate statistically those caused by primary gammas (signal) from the images of hadronic showers initiated by cosmic rays in the upper atmosphere (background).

Figure 34 Illustration of some image parameters adapted from [197]

Table XIX The features of the sample data in MAGIC telescope data set

1.	fLength:	major axis of ellipse [mm]
2.	fWidth:	minor axis of ellipse [mm]
3.	fSize:	10-log of sum of content of all pixels [photon count]
4.	fConc2:	ratio of sum of two highest pixels over fSize [ratio]
5.	fConc1:	ratio of highest pixel over fSize [ratio]
6.	fAsym:	distance from highest pixel to center, projected onto major axis [mm]
7.	fM3long:	3rd root of third moment along major axis [mm]
8.	fM3trans:	3rd root of third moment along minor axis [mm]
9.	fAlpha:	angle of major axis with vector to origin [deg]
10.	fDist:	distance from origin to center of ellipse [mm]
11.	Class:	g for gamma (signal), h for hadron (background)

Typically, a shower image after some preprocessing is an elongated cluster. A principal component analysis is performed in the camera plane, which results in a correlation axis and defines an ellipse. An example is shown in Figure 34. If the depositions were distributed as a bivariate Gaussian, this would be an equidensity ellipse. The characteristic parameters of this ellipse and the energy depositions are among the image parameters that can be used for discrimination. The dataset, which is available online at UCI machine learning repository [187], consists of 12332 gamma event samples and 6688 hadron event samples. Each sample is described by 11 features which are listed in Table XIX. Accurately classifying this dataset is considered a challenging task for decision

support systems, due to the weak discrimination between signal (gamma) and background (hadrons) [197].

In this section, the HiCEFS approach is applied to synthesize a fuzzy decision support system that can detect the gamma-ray signals. The dataset is randomly divided into 3 stratified groups of training and testing datasets, which are denoted as Div1, Div2 and Div3. Because of the abundant of samples, 1901 (10%) samples of the dataset are used for training and the remaining 17119 (90%) samples for testing. For this problem, the number of maximum generation for HCGA is set at 3000.

The final FDSS is benchmarked with several well-established techniques and systems, including k-nearest neighbors (kNN) [207], linear discriminant analysis (LDA) [208], C4.5 decision tree [9], naïve Bayes (NB) [209], support vector machine (SVM) [5], Adaptive network based fuzzy inference system (ANFIS) [88], evolving fuzzy neural network (EFuNN) [204], dynamic evolving neural-fuzzy inference system (DENFIS) [89] and rough set-based pseudo outer-product fuzzy neural network (RSPOP) [205]. Among these techniques, kNN, LDA, EFuNN, DENFIS are the MATLAB versions collected from online resources; ANFIS is the original version in MATLAB; RSPOP is obtained from the original author; C4.5, NB and SVM are implemented in WEKA [214].

Table XX Classification results on MAGIC gamma telescope dataset

	Accuracy (%)	Sensitivity (%)	Specificity (%)	Rules
kNN (k=5)	76.58 ± 0.8	88.79 ± 5.1	54.07 ± 11.6	NA
LDA	78.15 ± 0.1	90.56 ± 0.4	55.27 ± 0.6	NA
C4.5	82.49 ± 0.3	90.73 ± 0.6	67.30 ± 1.9	73.0 ± 8.89 (C)
NB	72.71 ± 0.2	91.98 ± 0.7	37.16 ± 0.8	NA
SVM	78.91 ± 0.2	90.13 ± 0.2	58.22 ± 0.6	NA
ANFIS	83.39 ± 0.2	92.90 ± 0.4	65.86 ± 1.1	5.33 ± 0.58 (S)
EFuNN	75.66 ± 0.1	81.31 ± 0.5	65.25 ± 1.0	677.0 ± 53.70 (S)
DENFIS	83.69 ± 0.5	93.21 ± 0.0	66.13 ± 1.6	13.33 ± 1.53 (S)
RSPOP	76.09 ± 0.8	90.58 ± 1.8	49.40 ± 1.5	82.67 ± 8.08 (M)
FDSS	83.99 ± 0.6	92.75 ± 0.7	67.57 ± 1.8	52.67 ± 17.95 (M)

Results are listed in Table XX, where the first column indicates the overall accuracy of

these techniques on classifying the gamma-rays and hadrons, and the second and third column shows their sensitivity and specificity. The sensitivity is the number of correctly classified positive cases divided by the number of positive cases. A higher sensitivity indicates that more gamma signals are correctly detected. The specificity is the number of correctly classified negative cases divided by the number of negative cases. A higher specificity implies that less hadron (background) are erroneously classified as gamma signals. Although it is desirable to achieve both high sensitivity and high specificity, the specificity is relatively more important because classifying a background event as a gamma signal is worse than classifying a gamma signal as background. Finally, Table XX shows the number of derived rules, where (C) after the rule number indicates crisp rules, (S) Takagi-Sugeno-Kang (TSK) type fuzzy rules and (M) Mamdani type fuzzy rules.

As shown in Table XX, LDA, NB and SVM achieve a relatively low specificity that is below 60%, although they managed to achieve over 90% in sensitivity. The EFuNN achieves a specificity of 65.25% that is comparable with other techniques. However, its sensitivity is only 81.31%, which is the lowest among all benchmarked techniques. Similarly, the RSPOP achieves 90.58% sensitivity but with a low specificity of 49.40%. It shows that it is not easy to achieve both high sensitivity and high specificity because the knowledge in the techniques or systems can be dominated by a certain class while the other class is under-learned.

As seen in Table XX, the FDSS is able to yield an overall accuracy of 83.99% which is higher than the accuracy of other benchmarked techniques. In term of sensitivity, the FDSS correctly classifies 92.75% of the interest events caused by gamma-ray signals and compare favorable with the sensitivity of other techniques, only slightly lower than the 93.21% achieved by DENFIS. The FDSS is also able to achieve a specificity rate higher than other techniques. It shows that the FDSS is less likely to misclassify a background event caused by hadron as an interest event caused by gamma-ray.

In the aspect of interpretability, no human readable rule can be extracted from kNN, LDA, NB and SVM. The C4.5 derived a tree with average 73 crisp rules. Compared with this,

the FDSS derived by HiCEFS approach is able to represent the knowledge in the MAGIC gamma telescope dataset with fewer rules, about 52 Mamdani-type fuzzy rules in average. Even with the rough set-based rule reduction technique, the RSPOP derived a fuzzy rule base consisting of about 30 more rules than that of the FDSS. The ANFIS and DENFIS both derived a very concise fuzzy rule base with around 5 and 13 TSK-type fuzzy rules, respectively. However, with the consequence represented as a linear combination of inputs, these rules are more difficult to analyze than the crisp rules in C4.5 and Mamdani-type fuzzy rules in the FDSS. Furthermore, the average rule length is shortened from 10 to about 7 in the FDSS derived by HiCEFS.

6.8.5 Concrete compressive strength prediction

In this section, the HiCEFS approach is applied to synthesize a fuzzy decision support system to address the engineering problem of predicting the compressive strength of concrete according to its age and mixing components.

High-strength concrete (HSC) is a kind of concrete that has a compressive strength, which refers to the maximum resistance of a concrete sample to applied pressure. In addition to the three basic ingredients in conventional concrete, i.e., cement, aggregates, and water, the making of HSC needs to incorporate supplementary materials, such as blast furnace slag and fly ash, and chemical admixture, such as superplasticizer. The high-strength concrete is most commonly used for the construction of high-rise buildings [198].

The generally accepted rule is that an increase in the water-to-cement ratio (w/c) decreases the concrete compressive strength, whereas a decrease in the w/c increases the strength [215]. However, this rule is so general that it cannot provide robust prediction of the concrete compressive strength. Furthermore, several independent studies have shown that concrete compressive strength development is determined not only by the w/c ratio, but also influenced by other concrete ingredients [216]. Currently, there is still no reliable nondestructive testing technique that is able to provide robust evaluation of the concrete compressive strength [217]. Therefore, there is a need for a decision support system that can be conveniently used to predict the compressive strength of concrete according to its

ingredients and other features. Based on the study, the experts could even understand the nature of concrete better and further optimize the concrete mixture.

The dataset of concrete compressive strength used in this application is available at UCI machine learning repository [187]. The variables of each sample data are listed in Table XXI. Although each component in concrete is described only by a single term, these terms actually represent a variety of forms. Cement can be powdered to various degrees of finenesses and composed of several different chemical compositions. The concrete are influenced by different mixes portions and by the mixing techniques. Therefore, the prediction of the concrete compressive strength is a highly uncertain task.

Table XXI Ranges of the variables of the dataset of concrete compressive strength

Variable	Minimum	Maximum	Average
Cement (kg/m ³)	102.0	540.0	281.2
Blast Furnace Slag (kg/m ³)	0.0	359.4	73.9
Fly Ash (kg/m ³)	0.0	200.1	54.2
Water (kg/m ³)	121.8	247.0	181.6
Superplasticizer (kg/m ³)	0.0	32.2	6.2
Coarse Aggregate (kg/m ³)	801.0	1145.0	972.9
Fine Aggregate (kg/m ³)	594.0	992.6	773.6
Age (days)	1	365	45.7
Concrete compressive strength (MPa)	2.33	82.60	35.8

The dataset is randomly divided into three training-testing groups named Div1-3. In each group, 680 (66%) samples of the dataset are used for training and the remaining 340 (33%) samples for testing. In this application, the $G_{\max}$ in HCGA is set at 3000 according to the results of some trial runs.

Table XXII The results of the concrete compressive strength prediction

	MSE	R^2	Rules
EFuNN	102.29 ± 9.43	0.798 ± 0.031	554.50 ± 5.07
DENFIS	72.26 ± 7.60	0.851 ± 0.027	73.75 ± 4.50
RSPOP	100.50 ± 16.68	0.786 ± 0.055	397.75 ± 4.50
FDSS	74.08 ± 9.72	0.805 ± 0.018	31.50 ± 5.00

The results on this application are listed in Table XXII. The accuracy of the systems is measured in mean squared error (MSE) and square of the Pearson product-moment

correlation coefficient (R^2). In this application, the MSE is more important than R^2 because the objective of the application is to minimize the error in the predicted concrete compressive strength instead of predicting the trend of a time series. Among all the benchmarked systems, the DENFIS manages to achieve highest accuracy in terms of MSE and R^2 . The FDSS derived by the HiCEFS approach yielded accuracy slightly lower than DENFIS but the number of rules in the FDSS is only about 43% of that in DENFIS. Compared with EFuNN and RSPOP, the FDSS achieved about 28% reduction in MSE and about 2% increase in R^2 with a much small fuzzy rule base. Furthermore, the average rule length in the FDSS is shortened to 5.83. The results show that the FDSS derived by HiCEFS approach compares favorably with other benchmarked systems.

The examples of the membership functions and their linguistic meanings in the derived FDSS for concrete compressive strength prediction are shown in Table XXIII. As can be seen, two and three membership functions are synthesized for the input variable of “cement” and “water”, which are the most important components of concrete. The HiCEFS approach derived two membership functions on the input variables of “blast furnace slag”, “fine aggregate” and “age”. With its feature selection capability, the HiCEFS approach synthesizes no membership functions on the input variables of “fly ash”, “superplasticizer” and “coarse aggregate”. It shows that these variables are less important to prediction of the concrete strength in this dataset.

Some observations on derived membership functions conform to the knowledge on concrete. For instance, by observing the relation between the concrete compressive strength and the age shown in Figure 35, it can be found that the concrete compressive strength is higher than 20 MPa when the age is higher than 28 days. This finding helps to explain how the derived membership functions define “low” and “high” on the variable of “age”. Also, the reason that superplasticizer is not selected as input may lie in the fact that the main objective of adding superplasticizer to the concrete is to increase workability of the concrete at the low water-cement ratio.

Table XXIII Examples of the membership functions in the derived FDSS for concrete compressive strength prediction

<i>low, high</i>	<i>low, high</i>	<i>low, medium, high</i>
<i>low, high</i>	<i>low, high</i>	<i>very low, low, slightly low, medium, slightly high, high, very high</i>

Figure 35 The relation between the age and the concrete compressive strength

The most frequently fired fuzzy rules among a total of 28 rules in the FDSS derived by the HiCEFS approach for Div1 are listed in Table XXIV. After analyzing these fuzzy rules, we can find that the cement, water and age are three most important factors that roughly determine the range of concrete compressive strength, whereas the blast furnace slag and fine aggregate can slightly adjust the strength. In accord with the observation in Figure 35, there is no rule with the condition of “age” is *high* gives the consequence of “concrete compressive strength” is *very low* or *low*.

For analyzing the decision-making logic, human users can further simplify the rule base by merge some rules with the same consequence into a more descriptive rule. For instance,

the R_{12} and R_{13} are the only rules in the fuzzy rule base that give the consequence of *very high*. The only difference between these two rules is that the condition of “fine aggregate” is different. Thus, these rules can be merged into the rule “IF cement IS *high* AND blast furnace slag IS *high* AND water IS *medium* AND age IS *high* THEN concrete compressive strength IS *very high*”. Adding this rule base simplification scheme to the HiCEFS approach forms a promising future search direction.

Table XXIV The examples of the fuzzy rules in the derived FDSS for concrete compressive strength prediction

Rule	cement	blast furnace slag	water	fine aggregate	age	concrete compressive strength
R_1	<i>low</i>	<i>low</i>	<i>high</i>	<i>low</i>	<i>low</i>	<i>very low</i>
R_2	<i>low</i>	<i>low</i>	<i>medium</i>	<i>high</i>	<i>low</i>	<i>low</i>
R_3	<i>high</i>	<i>low</i>	<i>high</i>	<i>high</i>	<i>low</i>	<i>slightly low</i>
R_4	<i>low</i>	<i>high</i>	<i>high</i>	<i>high</i>	<i>low</i>	<i>slightly low</i>
R_5	<i>high</i>	<i>low</i>	<i>high</i>	<i>high</i>	<i>high</i>	<i>medium</i>
R_6	<i>low</i>	<i>high</i>	<i>medium</i>	<i>high</i>	<i>low</i>	<i>medium</i>
R_7	<i>high</i>	<i>low</i>	<i>high</i>	<i>low</i>	<i>low</i>	<i>medium</i>
R_8	<i>high</i>	<i>low</i>	<i>high</i>	<i>low</i>	<i>high</i>	<i>slightly high</i>
R_9	<i>high</i>	<i>low</i>	<i>medium</i>	<i>low</i>	<i>low</i>	<i>slightly high</i>
R_{10}	<i>high</i>	<i>low</i>	<i>medium</i>	<i>high</i>	<i>high</i>	<i>high</i>
R_{11}	<i>high</i>	<i>low</i>	<i>medium</i>	<i>low</i>	<i>high</i>	<i>high</i>
R_{12}	<i>high</i>	<i>high</i>	<i>medium</i>	<i>high</i>	<i>high</i>	<i>very high</i>
R_{13}	<i>high</i>	<i>high</i>	<i>medium</i>	<i>low</i>	<i>high</i>	<i>very high</i>

6.8.6 Analysis and discussions

In previous sections, results are reported and a corresponding analysis is given for each experiment. This section consolidates our findings by providing an overall analysis of the HiCEFS approach and the performance obtained.

As shown in the experimental results in Section 6.8.2 to 6.8.5, the FDSS derived by the proposed HiCEFS approach is able to achieve higher generalization ability than other benchmarked systems, which include the representatives of the state of the art. This is mainly due to the following two factors.

The first factor is the high search ability of the proposed HiCEFS. Adopting the Hierarchical Coevolutionary Genetic Algorithm (HCGA) as the learning mechanism, which is discussed in Section 5.1, the HiCEFS approach efficiently searches for good

solutions in a complex space. Compared with conventional single population evolutionary algorithms, the HCGA can divide the complex search task of synthesizing a FRBS into the subtasks of finding good fuzzy partitions and fuzzy rule base, as elaborated in Section 6.2-6.7.

The second factor is the high flexibility of the ISMFs. As it comprises a set of sampling points, the ISMF is able to efficiently represent various data distribution and reduce the complexity of the FDSS in terms of average rule length and rule base size, as shown in Section 6.8.

Chapter 7

Conclusion and Future Work

7.1 Achievements

Due to the increasing complexity of many essential application domains, there is a great need for decision support systems that can assist human users by improving the efficiency, accuracy and reliability of their decision-making. To address such need, this thesis focuses on the development of a novel and generic approach for the automatic synthesis of fuzzy decision support systems (FDSS) with following requirements. First, the fuzzy decision support systems derived by the proposed approach need to provide accurate decision support so as to be successfully applied in solving complex real-world problems. Second, the derived FDSS should provide human-readable decision-making logic, which can be analyzed and understood by the decision makers. Last, the proposed approach needs to be generic and require minimal human effort in the synthesis process, so that it can be utilized to solve a great variety of experimental and practical problems with different levels of complexity.

The achievements and contributions of this thesis are outlined as follows:

- A generic approach termed the Hierarchical Co-Evolutionary Fuzzy system Synthesis (HiCEFS) for automatically synthesizing from data samples a fuzzy decision support system with well-balanced accuracy and interpretability is proposed. The HiCEFS approach requires minimal effort from the designer and reduces the unwanted biases that may be introduced by human intervention. The overview of the approach is introduced in Chapter 3. The technical details and case studies are given in Chapter 6. In the proposed approach, the complex task of synthesizing a FDSS is divided into the subtasks of deriving two most important components in a FDSS, which include the fuzzy partition table and the fuzzy rule base. In this divide-and-conquer manner, the approach simultaneously handles the issues in synthesizing a FDSS, which include selecting the informative input variables, generating a proper number of fuzzy membership functions on each input variable and deriving a compact fuzzy rule base.

Being generic, the proposed HiCEFS approach can be easily applied to synthesize FDSS for solving problems with different complexity from various areas.

- A generic type of membership function termed Irregular-Shaped Membership Function (ISMF) is proposed in Chapter 4. An evolutionary-based learning technique dynamically and automatically allocates the proper number of sampling points in each ISMF and optimally tunes the position of these points. Its shape and position being determined by these sampling points, the ISMF is flexible and able to approximate any type of commonly used membership functions, such as triangular, trapezoidal, and Gaussian. Therefore, the ISMF subsumes those common types and can provide better data representation capability. Repairing schemes that require minimal computational cost are devised to maintain the validity of the derived ISMFs and the interpretability of the ISMF fuzzy partitions. The proposed ISMF provide a highly versatile type of membership function for the FDSS designers' choice and form a crucial component in the HiCEFS approach.
- A Hierarchical Coevolutionary Genetic Algorithm (HCGA) is proposed in Chapter 5 to serve as the main learning technique in the proposed HiCEFS approach for synthesizing a fuzzy decision support system. The HCGA can be specifically configured to coevolve the modules of the candidate solutions in separate populations. During the coevolution process, these components are combined again to form the candidate solutions and the fitness of these components is based on how well they collaborate in the candidate solutions.
- The proposed HiCEFS approach is applied to automatically synthesize the fuzzy decision support systems for diverse application domains including financial trading, medical diagnosis, scientific classification and engineering. The FDSSs derived by the HiCEFS approach are applied to provide decision support in these applications and compared against other established techniques. Results demonstrate that the FDSSs derived by the HiCEFS approach are able to provide more accurate decision support and more readable decision-making logic.

7.2 Limitations

Some limitations remain in the HiCEFS approach and it can be further improved in several aspects.

One limitation is that the computational cost is high or even prohibitive if the approach is directly applied to problems with over hundreds or even thousands of input features, such as gene expression analysis [218]. In such case, it is more cost-effective to adopt a pre-processing with a filter feature selection technique [219], after which the HCGA may further select informative features during coevolution process.

Another limitation is that the approach is not suitable for online applications due to the computational cost of the evolutionary techniques. Regardless, only a few real-world applications, for instance learning the behavior of the autonomous agents that interact in dynamic environments [220], require real-time learning capability. The majority of real-world applications can be conducted offline and this approach is for which suitable.

Lastly, although the HiCEFS approach has demonstrated its performance on various real-world applications in this thesis, there is no guarantee to synthesize the best FDSS for any problem, like all the other learning techniques.

7.3 Future research recommendations

In this section, several possible directions of future research are presented.

- **Improving the computational efficiency of the HiCEFS approach**

Further improving the learning efficiency of the HiCEFS approach forms the most important future research direction because it directly affects the performance of the derived fuzzy decision support system. The improvements can be applied in the following aspects in the HiCEFS approach.

First, how the candidate solutions (phenotypes) are represented in chromosomes (genotypes) can greatly affect the search efficiency of evolutionary techniques [221], hence the proposed structure of the multi-level Hierarchical Coevolutionary Genetic Algorithm may be altered for better representation of a FDSS. For instance, varying-length chromosomes [222] can be used in Level-1 populations. These chromosomes can start with short length, representing ISMF with maximal 2 shoulder pointers on each side. Then they can grow to longer to allow the synthesis of more complicated ISMFs when the fitness is not increasing.

Second, besides the hybrid crossover operator described in Section 6.5, other specifically designed genetic operators, including selection, crossover and mutation operators may be adopted for improving the performance of the HiCEFS approach. One possible method is adopting multiple mutation operators as proposed in [223].

Third, although the static control parameters employed in HCGA are demonstrated to be suitable for various problems, better performance may be achieved by dynamically altering these parameters according to the coevolution progress. One example is to dynamically increase the crossover rate and decrease the mutation rate when the similarity among the chromosomes is high, or conversely.

Fourth, alternative evolutionary techniques such as memetic algorithm [113, 114] and differential evolution [224] can be explored for further enhancing the search efficiency and reducing the computational cost of the HCGA in HiCEFS. A recent research work [225] proposed an evolutionary algorithm combining differential evolution (DE) and estimation of distribution algorithm (EDA). DE is able to search according to the local information extracted from the current population. EDA is able to extract directly the global statistical information about the search space from the search history. By guiding the DE search with the global information provided by EDA, the proposed method shows improved search efficiency. Adopting these hybrid methods in HCGA also forms an interesting direction for future research. Furthermore, different evolutionary techniques may be applied in populations from different levels

in HCGA, similar to how different encoding methods and genetic operators are applied in Chapter 6.

Last, because of the inherent parallel nature of HCGA, it is possible and more efficient to coevolve the populations simultaneously using multiple processors [116, 226]. Distributing the computation amidst several processors will help overcome the problem of long training time caused by the high computational cost mentioned in Section 7.2. Thus, it can achieve a parallelized HiCEFS approach that is capable of synthesizing a FDSS with higher performance within a similar training time when compared with other techniques.

- **Improving the representation accuracy of ISMF model**

In this thesis, the Irregular-Shaped Membership Functions are devised to be piece-wise linear for simplicity of computation. Thus, the cost of computing the membership value of an instance using an ISMF is equivalent to that of using a triangular or trapezoidal MF. The proposed ISMF model is computationally costly, only because it needs to encode and tune many values (sampling points). While it would be possible to replace the straight line segments in ISMFs with smoother curves by using splines [227] or other high-order approximation functions [228], doing so would greatly increase the computational cost. Nevertheless, the modeling capability and performance of higher-order ISMFs could be studied and benchmarked against the current piece-wise linear version.

- **Increasing interpretability of the fuzzy rule base**

As shown in the experimental results in Section 6.8, a compact fuzzy rule base is formed in the FDSS derived by the HiCEFS approach. However, the size of the fuzzy rule base can be further reduced by merging some similar fuzzy rules (example can be found in Section 6.8.5). One research direction is to devise an approach to merge these similar fuzzy rules into a more descriptive fuzzy rule without undermining the accuracy the final FDSS [229].

- **Extending further the HiCEFS approach for selecting fuzzy inference operators**

Besides the membership functions and fuzzy rule base, the fuzzy inference operators form another important component in a fuzzy decision support system. Different combination of fuzzy inference operators can result in different fuzzy inference schemes, including the Compositional Rule of Inference (CRI) [76], Truth Value Restriction (TVR) [230], Approximate Analogical Reasoning Schema (AARS) [231] and Yager inference [232]. One possible extension of the HiCEFS approach is to enable the automatic selection of the fuzzy inference operators in the FDSS by encoding them into the Level-5 chromosomes as described in Section 6.2.

Author's Publications

Journals

H. Huang, M. Pasquier and C. Quek, "Financial Market Trading System with a Hierarchical Coevolutionary Fuzzy Predictive Model", to be published in *IEEE Trans. Evolutionary Computation, Special issue on evolutionary computation for finance and economics*, DOI: 10.1109/TEVC.2007.911682. (#3 AI Journal with impact factor 3.77 at the date of acceptance)

H. Huang, M. Pasquier and C. Quek, "Decision Support System based on Hierarchical Coevolutionary Fuzzy Approach: A Case Study in Detecting Gamma Ray Signals", *submitted to IEEE Trans. Systems, Man, and Cybernetics, Part B: Cybernetics*, 2008.

H. Huang, M. Pasquier and C. Quek, "A Hierarchical Coevolutionary-based Approach for the Synthesis of Compact and Interpretable Fuzzy Partitions", *submitted to IEEE Trans. Evolutionary Computation*, 2007.

H. Huang, M. Pasquier and C. Quek, "Genetically Generated Irregular Shaped Membership Functions", *submitted to IEEE Trans. Evolutionary Computation*, 2007.

Conferences

H. Huang, M. Pasquier and C. Quek, "Application of a Hierarchical Coevolutionary Fuzzy System for Financial Prediction and Trading", in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC)*, Hong Kong, 2008, pp. 1252-1259.

H. Huang, M. Pasquier and C. Quek, "HiCEFS - A Hierarchical Coevolutionary Approach for the Dynamic Generation of Fuzzy System", in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC)*, Singapore, 2007, pp. 3426-3433.

H. Huang, M. Pasquier and C. Quek, "Optimally Evolving Irregular-Shaped Membership Functions for Fuzzy Systems", in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC)*, Vancouver, BC, Canada, 2006, pp. 3309-3316.

References

- [1] F. V. Jensen, *Introduction to Bayesian Networks*. New York: Springer, 1997.
- [2] G. A. Carpenter and S. Grossberg, *Pattern Recognition by Self-organizing Neural Networks*. Cambridge, MA: MIT Press, 1991.
- [3] S. Haykin, *Neural Networks: A Comprehensive Foundation*, 2nd ed. Englewood Cliffs, NJ: Prentice Hall, 1998.
- [4] V. Vapnik, *Statistical Learning Theory*. New York: Wiley, 1998.
- [5] C. J. C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, vol. 2, pp. 121-167, 1998.
- [6] K. Kristensen and I. A. Rasmussen, "The use of a Bayesian network in the design of a decision support system for growing malting barley without use of pesticides," *Computers and Electronics in Agriculture*, vol. 33, pp. 197-217, 2002.
- [7] P. Refenes, Y. Abu-Mostafa, J. Moody, and A. Weigend, "Neural Networks in Financial Engineering." Singapore: World Scientific, 1996.
- [8] K. Morik, M. Imboff, P. Brockhausen, T. Joachims, and U. Gather, "Knowledge discovery and knowledge validation in intensive care," *Artificial Intelligence in Medicine*, vol. 19, pp. 225-249, 2000.
- [9] J. R. Quinlan, *C4.5: Programs for Machine Learning*. San Mateo, CA: Morgan Kaufmann, 1993.
- [10] G. Klir and B. Yuan, *Fuzzy Sets and Fuzzy Logic: Theory and Applications*. Upper Saddle River, NJ: Prentice Hall, 1995.
- [11] J. Abonyi, J. A. Roubos, and F. Szeifert, "Data-driven generation of compact, accurate, and linguistically sound fuzzy classifiers based on a decision-tree initialization," *International Journal of Approximate Reasoning*, vol. 32, pp. 1-21, 2003.
- [12] J. Casillas, O. Cordon, F. Herrera, and L. Magdalena, "Accuracy Improvements in Linguistic Fuzzy Modeling," in *Studies in Fuzziness and Soft Computing*, vol. 129. Heidelberg, Germany: Springer-Verlag, 2003.
- [13] J. Casillas, O. Cordon, F. Herrera, and L. Magdalena, Eds., "Interpretability Issues in Fuzzy Modeling," in *Studies in Fuzziness and Soft Computing*, vol. 128. Heidelberg, Germany: Springer-Verlag, 2003.

- [14] A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*. Englewood Cliffs, NJ: Prentice Hall, 1988.
- [15] J. S. R. Jang, "Self-learning fuzzy controllers based on temporal backpropagation," *IEEE Transactions on Neural Networks*, vol. 3, pp. 714-723, 1992.
- [16] K. K. Ang, C. Quek, and M. Pasquier, "POPFNN-CRI(S): Pseudo outer product based fuzzy neural network using the compositional rule of inference and singleton fuzzifier," *IEEE Transactions on Systems Man and Cybernetics Part B-Cybernetics*, vol. 33, pp. 838-849, 2003.
- [17] W. L. Tung and C. Quek, "GenSoFNN: A generic self-organizing fuzzy neural network," *IEEE Transactions on Neural Networks*, vol. 13, pp. 1075-1086, 2002.
- [18] M. Setnes and H. Roubos, "GA-fuzzy modeling and classification: complexity and performance," *IEEE Transactions on Fuzzy Systems*, vol. 8, pp. 509-522, 2000.
- [19] H. Roubos and M. Setnes, "Compact and transparent fuzzy models and classifiers through iterative complexity reduction," *IEEE Transactions on Fuzzy Systems*, vol. 9, pp. 516-524, 2001.
- [20] E. Zhou and A. Khotanzad, "Fuzzy classifier design using genetic algorithms," *Pattern Recognition*, vol. 40, pp. 3401-3414, 2007.
- [21] J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithm*. New York: Plenum Press, 1981.
- [22] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," in *Parallel Distributed Processing*, vol. 1, D. E. Rumelhart and J. L. McClelland, Eds. Cambridge, MA: MIT Press, 1986.
- [23] J. A. Snyman, *Practical Mathematical Optimization: An Introduction to Basic Optimization Theory and Classical and New Gradient-Based Algorithms*. New York: Springer, 2005.
- [24] H. Ishibuchi, K. Nozaki, H. Tanaka, Y. Hosaka, and M. Matsuda, "Empirical study on learning in fuzzy systems by rice taste analysis," *Fuzzy Sets and Systems*, vol. 64, pp. 129-144, 1994.
- [25] K. Nozaki, H. Ishibuchi, and H. Tanaka, "A simple but powerful heuristic method for generating fuzzy rules from numerical data," *Fuzzy Sets and Systems*, vol. 86, pp. 251-270, 1997.
- [26] R. W. Zhou and C. Quek, "POPFNN: A pseudo outer-product based fuzzy neural network," *Neural Networks*, vol. 9, pp. 1569-1581, 1996.

- [27] H. Ichihashi, T. Shirai, K. Nagasaka, and T. Miyoshi, "Neuro-fuzzy ID3: a method of inducing fuzzy decision trees with linear programming for maximizing entropy and an algebraic method for incremental learning," *Fuzzy Sets and Systems*, vol. 81, pp. 157-167, 1996.
- [28] C. Quek and R. W. Zhou, "The POP learning algorithms: reducing work in identifying fuzzy rules," *Neural Networks*, vol. 14, pp. 1431-1445, 2001.
- [29] L. X. Wang and J. M. Mendel, "Generating Fuzzy Rules by Learning from Examples," *IEEE Transactions on Systems Man and Cybernetics*, vol. 22, pp. 1414-1427, 1992.
- [30] M. Flores, J. Gámez, and J. Puerta, "Learning Linguistic Fuzzy Rules by Using Estimation of Distribution Algorithms as the Search Engine in the COR Methodology," in *Towards a New Evolutionary Computation*, 2006, pp. 259-280.
- [31] T. Back, *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*. New York: Oxford University Press, 1996.
- [32] T. Back, D. B. Fogel, and Z. Michalewicz, *Handbook of Evolutionary Computation*. Bristol, UK: IOP Publishing Ltd, 1997.
- [33] D. B. Fogel, T. Bäck, and Z. Michalewicz, *Evolutionary Computation*. Bristol: Institute of Physics Pub, 2000.
- [34] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading, Mass.: Addison-Wesley, 1989.
- [35] X. Yao, "Evolutionary Computation: Theory and Applications." Singapore: World Scientific Publishing, 1999.
- [36] O. Cordon, F. Herrera, F. Hoffmann, and L. Magdalena, *Genetic Fuzzy Systems: Evolutionary Tuning and Learning of Fuzzy Knowledge Bases*. Singapore: World Scientific, 2001.
- [37] C. L. Karr, "Design of an adaptive fuzzy logic controller using a genetic algorithm," in *Proc. 4th Intl. Conf. on Genetic Algorithms*, pp. 450-457, 1991.
- [38] T. Murata and H. Ishibuchi, "Adjusting membership functions of fuzzy classification rules by genetic algorithms," in *Proc. Intl. Joint Conf. of the 4th IEEE Intl. Conf. on Fuzzy Systems and the 2nd Intl. Fuzzy Engineering Symposium*, vol. 4, pp. 1819 - 1824, 1995.
- [39] J. Casillas, O. Cordon, M. J. del Jesus, and F. Herrera, "Genetic tuning of fuzzy rule deep structures preserving interpretability and its

- interaction with fuzzy rule set reduction," *IEEE Transactions on Fuzzy Systems*, vol. 13, pp. 13-29, 2005.
- [40] F. Hoffmann and G. Pfister, "Evolutionary design of a fuzzy knowledge base for a mobile robot," *International Journal of Approximate Reasoning*, vol. 17, pp. 447-469, 1997.
- [41] O. Cordon, M. J. del Jesus, F. Herrera, and M. Lozano, "MOGUL: A methodology to obtain genetic fuzzy rule-based systems under the iterative rule learning," *International Journal of Intelligent Systems*, vol. 14, pp. 1123-1153, 1999.
- [42] A. Gonzalez and R. Perez, "SLAVE: A genetic learning system based on an iterative approach," *IEEE Transactions on Fuzzy Systems*, vol. 7, pp. 176-191, 1999.
- [43] H. Ishibuchi, T. Nakashima, and T. Murata, "Performance evaluation of fuzzy classifier systems for multidimensional pattern classification problems," *IEEE Transactions on Systems, Man and Cybernetics, Part B*, vol. 29, pp. 601-618, 1999.
- [44] F. Hoffmann, B. Baesens, C. Mues, T. Van Gestel, and J. Vanthienen, "Inferring descriptive and approximate fuzzy rules for credit scoring using evolutionary algorithms," *European Journal of Operational Research*, vol. 177, pp. 540-555, 2007.
- [45] C. L. Karr and E. J. Gentry, "Fuzzy control of pH using genetic algorithms," *IEEE Transactions on Fuzzy Systems*, vol. 1, pp. 46-53, 1993.
- [46] C.-H. Wang, T.-P. Hong, and S.-S. Tseng, "Integrating fuzzy knowledge by genetic algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 2, pp. 138-149, 1998.
- [47] I. Jagielska, C. Matthews, and T. Whitfort, "An investigation into the application of neural networks, fuzzy logic, genetic algorithms, and rough sets to automated knowledge acquisition for classification problems," *Neurocomputing*, vol. 24, pp. 37-54, 1999.
- [48] M. R. Delgado, F. Von Zuben, and F. Gomide, "Hierarchical genetic fuzzy systems," *Information Sciences*, vol. 136, pp. 29-52, 2001.
- [49] C. A. Pena-Reyes and M. Sipper, "Fuzzy CoCo: a cooperative-coevolutionary approach to fuzzy modeling," *IEEE Transactions on Fuzzy Systems*, vol. 9, pp. 727-737, 2001.
- [50] M. A. Potter and K. A. De Jong, "Cooperative coevolution: An architecture for evolving coadapted subcomponents," *Evolutionary Computation*, vol. 8, pp. 1-29, 2000.

- [51] V. R. Khare, Y. Xin, B. Sendhoff, J. Yaochu, and H. Wersing, "Coevolutionary modular neural networks for automatic problem decomposition," in *Proc. of the IEEE Congress on Evolutionary Computation (CEC)*, vol. 3, pp. 2691-2698, 2005.
- [52] P. N. Finlay, *Introducing Decision Support Systems*. Oxford, UK: NCC Blackwell, 1994.
- [53] D. Heckerman, D. Geiger, and D. M. Chickering, "Learning Bayesian networks: The combination of knowledge and statistical data," *Machine Learning*, vol. 20, pp. 197-243, 1995.
- [54] N. Friedman, D. Geiger, and M. Goldszmidt, "Bayesian network classifiers," *Machine Learning*, vol. 29, pp. 131-163, 1997.
- [55] B. Widrow and S. D. Stearns, *Adaptive Signal Processing*. Englewood Cliffs, NJ: Prentice Hall, 1985.
- [56] C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford: Clarendon Press, 1995.
- [57] I. T. Jolliffe, *Principal Component Analysis*. New York: Springer, 1986.
- [58] G. J. McLachlan, *Discriminant Analysis and Statistical Pattern Recognition*. New York: Wiley, 1992.
- [59] K. Crockett, Z. Bandar, D. McLean, and J. O'Shea, "On constructing a fuzzy inference framework using crisp decision trees," *Fuzzy Sets and Systems*, vol. 157, pp. 2809-2832, 2006.
- [60] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees*. Wadsworth, 1984.
- [61] P. Geurts, "Some enhancements of decision tree bagging," in *Proc. of the 4th European Conf. on Principles of Data Mining and Knowledge Discovery*, pp. 136-147, 2000.
- [62] K. Nozaki, H. Ishibuchi, and H. Tanaka, "Adaptive fuzzy rule-based classification systems," *IEEE Transactions on Fuzzy Systems*, vol. 4, pp. 238-250, 1996.
- [63] L. A. Zadeh, "Outline of a new approach to the analysis of complex systems and decision processes," *IEEE Transactions on Systems, Man and Cybernetics*, vol. SMC-3(1), pp. 28-44, 1973.
- [64] L. A. Zadeh, "Fuzzy sets," *Information and Control*, vol. 11, pp. 748-768, 1965.

- [65] L. Baron, S. Achiche, and M. Balazinski, "Fuzzy decision support system knowledge base generation using a genetic algorithm," *International Journal of Approximate Reasoning*, vol. 28, pp. 125-148, 2001.
- [66] D. Nauck and R. Kruse, "NEFCLASS - A neuro-fuzzy approach for the classification of data," in *Proc. of the ACM Symp. on Applied Computing, Nashville*, pp. 461 - 465, 1995.
- [67] D. Nauck and R. Kruse, "Neuro-fuzzy systems for function approximation," *Fuzzy Sets and Systems*, vol. 101, pp. 261 - 271, 1999.
- [68] L. X. Wang, *Adaptive Fuzzy Systems and Control: Design and Analysis*. Englewood Cliffs, NJ: Prentice-Hall, 1994.
- [69] H.-J. Zimmermann, *Fuzzy Set Theory and Its Applications*. New York: Springer, 2001.
- [70] W. Pedrycz, *Fuzzy Control and Fuzzy Systems*. New York: John Wiley & Sons, 1993.
- [71] Y. Jin, *Advanced Fuzzy Systems Design and Applications*. New York: Physica-Verlag, 2003.
- [72] E. H. Mamdani, "Application of fuzzy logic to approximate reasoning using linguistic synthesis," *IEEE Transactions on Computers*, vol. C-26, pp. 1182-1191, 1977.
- [73] M. Sugeno and M. Nishida, "Fuzzy control of model car," *Fuzzy Sets and Systems*, vol. 16, pp. 103-113, 1985.
- [74] W. Pedrycz, "Fuzzy Modelling: Paradigms and Practice." Norwell, MA: Kluwer Academic, 1996.
- [75] C.-T. Lin and C. S. G. Lee, *Neural Fuzzy Systems: A Neuro-Fuzzy Synergism to Intelligent Systems*. Englewood Cliffs, NJ: Prentice Hall, 1996.
- [76] L. A. Zadeh, "Calculus of fuzzy restrictions," in *Fuzzy Sets and Their Applications to Cognitive and Decision Processes*. New York: Academic, 1975, pp. 1-39.
- [77] H. Ishibuchi, K. Nozaki, N. Yamamoto, and H. Tanaka, "Selecting fuzzy if-then rules for classification problems using genetic algorithms," *IEEE Transactions on Fuzzy Systems*, vol. 3, pp. 260-270, 1995.
- [78] T. Murata, H. Ishibuchi, and M. Gen, "Construction of fuzzy classification systems using multiple fuzzy rule tables," in *Proc. of*

- IEEE Intl. Conf. on Systems, Man, and Cybernetics*, vol. 4, pp. 875-880, 1999.
- [79] Y. Jin, *Advanced Fuzzy Systems Design and Applications*, vol. 112: Physica-Verlag, 2003.
- [80] H. Ishibuchi and Y. Nojima, "Analysis of interpretability-accuracy tradeoff of fuzzy systems by multiobjective fuzzy genetics-based machine learning," *International Journal of Approximate Reasoning*, vol. 44, pp. 4-31, 2007.
- [81] J. G. Marin-Blazquez and Q. Shen, "From approximative to descriptive fuzzy classifiers," *IEEE Transactions on Fuzzy Systems*, vol. 10, pp. 484-497, 2002.
- [82] R. R. Yager and D. P. Filev, *Essentials of Fuzzy Modeling and Control*. New York: John Wiley & Sons, 1994.
- [83] S. Mitra and Y. Hayashi, "Neuro-fuzzy rule generation: Survey in soft computing framework," *IEEE Transactions on Neural Networks*, vol. 11, pp. 748-768, 2000.
- [84] J.-S. R. Jang and C. T. Sun, "Neuro-fuzzy modeling and control," *Proceedings of the IEEE*, vol. 83, pp. 378-406, 1995.
- [85] J.-S. Wang and C. S. G. Lee, "Self-adaptive neuro-fuzzy inference systems for classification applications," *IEEE Transactions on Fuzzy Systems*, vol. 10, pp. 790-802, 2002.
- [86] N. R. Pal, J. C. Bezdek, and E. C. K. Tsao, "Generalized Clustering Networks and Kohonen Self-Organizing Scheme (Vol 4, Pg 549, 1993)," *IEEE Transactions on Neural Networks*, vol. 6, pp. 521-521, 1995.
- [87] A. Abraham, "Neuro Fuzzy Systems: State-of-the-Art Modeling Techniques," in *Connectionist Models of Neurons, Learning Processes, and Artificial Intelligence*, 2001, pp. 269-276.
- [88] J.-S. R. Jang, "ANFIS: adaptive-network-based fuzzy inference system," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 23, pp. 665-685, 1993.
- [89] N. Kasabov and Q. Song, "DENFIS: Dynamic evolving neural-fuzzy inference system and its application for time-series prediction," *IEEE Transactions on Fuzzy Systems*, vol. 10, pp. 144-154, 2002.
- [90] D. Nauck and R. Kruse, "A neuro-fuzzy method to learn fuzzy classification rules from data," *Fuzzy Sets and Systems*, vol. 89, pp. 277-288, 1997.

- [91] J. H. Holland, *Adaptation in natural and artificial systems*. Ann Arbor, MI: University of Michigan Press, 1975.
- [92] J. R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. Cambridge, MA: MIT Press, 1992.
- [93] H.-P. Schwefel, *Evolution and Optimum Seeking*. New York: Wiley & Sons, 1995.
- [94] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. of IEEE Intl. Conf. on Neural Networks*, vol 4, pp. 1942-1948, 1995.
- [95] M. Dorigo and T. Stützle, "The ant colony optimization metaheuristic: algorithms, applications, and advances," in *Handbook of Metaheuristics*, 2003, pp. 250-285.
- [96] R. Storn and K. Price, "Differential evolution – A simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, pp. 341-359, 1997.
- [97] P. Larranaga and J. A. Lozano, *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*. Boston, MA: Kluwer Academic Publishers, 2002.
- [98] F. C.-H. Rhee and Y. J. Lee, "Unsupervised feature selection using a fuzzy-genetic algorithm," in *Proc. of IEEE Intl. Conf. on Fuzzy Systems*, vol. 3, pp. 1266 -1269, 1999.
- [99] E. C. C. Tsang, D. S. Yeung, and J. W. T. Lee, "Refinement of knowledge representation parameters in fuzzy production rules by genetic algorithms," in *Proc. of IEEE Intl. Conf. on Systems, Man, and Cybernetics*, vol. 2, pp. 1518 -1523, 1998.
- [100] F. Cheong and R. Lai, "Constraining the optimization of a fuzzy logic controller using an enhanced genetic algorithm," *IEEE Transactions on Systems, Man and Cybernetics, Part B*, vol. 30, pp. 31-46, 2000.
- [101] W. A. Farag, V. H. Quintana, and G. Lambert-Torres, "A genetic-based neuro-fuzzy approach for modeling and control of dynamical systems," *IEEE Transactions on Neural Networks*, vol. 9, pp. 756 -767, 1998.
- [102] F. Herrera, M. Lozano, and J. L. Verdegay, "Tackling real-coded genetic algorithms: operators and tools for behavioural analysis," *Artificial Intelligence Review*, vol. V12, pp. 265-319, 1998.
- [103] D. E. Goldberg and K. Deb, "A comparative analysis of selection schemes used in genetic algorithms," in *Foundations of Genetic Algorithms*, G. Rawlins, Ed. San Mateo, CA: Morgan Kaufmann, 1991, pp. 69-93.

- [104] T. Blickle and L. Thiele, "A comparison of selection schemes used in evolutionary algorithms," *Evolutionary Computation*, vol. 4, pp. 361-394, 1996.
- [105] H. Heider and T. Drabe, "A cascaded genetic algorithm for improving fuzzy-system design," *International Journal of Approximate Reasoning*, vol. 17, pp. 351-368, 1997.
- [106] C.-F. Juang, "A TSK-type recurrent fuzzy network for dynamic systems processing by neural network and genetic algorithms," *IEEE Transactions on Fuzzy Systems*, vol. 10, pp. 155 -170, 2002.
- [107] S. K. Sharma and M. O. Tokhi, "Genetic evolution: a dynamic fuzzy approach," in *Proc. of IEEE Intl. Conf. on Fuzzy Systems*, vol. 2, pp. 748 -752, 2000.
- [108] A. H. Wright, "Genetic algorithms for real parameter optimization," in *Foundations of Genetic Algorithms, First Workshop on the Foundations of Genetic Algorithms and Classifier Systems*, G. Rawlins, Ed. Los Altos, CA: Morgan Kaufmann, 1991, pp. 205-218.
- [109] E. Vonk, L. C. Jain, and R. P. Johnson, "Automatic generation of neural network architecture using evolutionary computation," *Advances in Fuzzy Systems Applications and Theory Vol. 14*, 1997.
- [110] Y. Tan and J. Wang, "Nonlinear blind source separation using higher order statistics and a genetic algorithm," *IEEE Transactions on Evolutionary Computation*, vol. 5, pp. 600-612, 2001.
- [111] J. Horn, *The nature of niching: genetic algorithms and the evolution of optimal, cooperative populations*: PhD thesis, University of Illinois at Urbana Champaign, Urbana, Illinois, 1997.
- [112] J. A. Roubos, M. Setnes, and J. Abonyi, "Learning fuzzy classification rules from labeled data," *Information Sciences*, vol. 150, pp. 77-93, 2003.
- [113] N. Krasnogor and J. Smith, "A tutorial for competent memetic algorithms: model, taxonomy, and design issues," *IEEE Transactions on Evolutionary Computation*, vol. 9, pp. 474-488, 2005.
- [114] E. Lumanpauw, M. Pasquier, and Q. Chai, "MNFS-FPM: A novel memetic neuro-fuzzy system based financial portfolio management," in *Proc. of IEEE Congress on Evolutionary Computation (CEC)*, pp. 2554-2561, 2007.
- [115] J. Paredis, "Coevolutionary computation," *Artificial Life*, vol. 2, pp. 355-375, 1995.

- [116] K. C. Tan, Y. J. Yang, and C. K. Goh, "A distributed cooperative coevolutionary algorithm for multiobjective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 10, pp. 527-549, 2006.
- [117] O. Cordon, F. Gomide, F. Herrera, F. Hoffmann, and L. Magdalena, "Genetic fuzzy systems. New developments," *Fuzzy Sets and Systems*, vol. 141, pp. 1-3, 2004.
- [118] M. Galea, Q. Shen, and J. Levine, "Evolutionary approaches to fuzzy modelling for classification," *The Knowledge Engineering Review*, vol. 19, pp. 27-59, 2004.
- [119] C. L. Karr, "Genetic algorithms for fuzzy controllers," *AI Expert*, vol. 6, pp. 27-33, 1991.
- [120] M. S. Chang and H. K. Chen, "A new encoding method of genetic algorithms towards parameter identification of fuzzy expert systems," *Proc. of the Asian Fuzzy Systems Symp.*, pp. 406-411, 1996.
- [121] Y.-C. Hu, "Determining membership functions and minimum fuzzy support in finding fuzzy association rules for classification problems," *Knowledge-Based Systems*, vol. 19, pp. 57-66, 2006.
- [122] A. Itagaki, M. Takashima, Y. Ashino, C. Nishio, and S. Nakanishi, "Fuzzy inference systems by genetic algorithm and factor analysis modeling for multivariate complex systems," in *Proc. of IEEE Symp. on Emerging Technologies and Factory Automation*, pp. 199-205, 1994.
- [123] C. C. Wong and C. C. Chen, "Generating fuzzy rules by a GA-based method from input-output data," in *Proc. of IEEE Intl. Conf. on Systems, Man, and Cybernetics*, vol. 5, pp. 278 - 283, 1999.
- [124] C. C. Wong and C. C. Chen, "A GA-based method for constructing fuzzy systems directly from numerical data," *IEEE Transactions on Systems, Man and Cybernetics, Part B*, vol. 30, pp. 904-911, 2000.
- [125] S. Matsushita, T. Furuhashi, and H. Tsutsui, "Rule extraction through fuzzy modeling using fuzzy neural network," in *Proc. of IEEE Intl. Joint Conf. on Neural Networks*, vol. 1, pp. 35-38, 1998.
- [126] T. Furuhashi, S. Matsushita, and H. Tsutsui, "Evolutionary fuzzy modeling using fuzzy neural networks and genetic algorithm," in *Proc. of IEEE Intl. Conf. on Evolutionary Computation (CEC)*, pp. 623 - 627, 1997.
- [127] C. C. Wong and S. M. Her, "A self-generating method for fuzzy system design," *Fuzzy Sets and Systems*, vol. 103, pp. 13-25, 1999.

- [128] M. A. Lee and H. Takagi, "Integrating design stages of fuzzy systems using genetic algorithms," in *Proc. of IEEE Intl. Conf. on Fuzzy Systems*, pp. 612-617, 1993.
- [129] D. Kim and H. Lee, "An optimal design of neuro-FLC by Lamarckian co-adaptation of learning and evolution," *Fuzzy Sets and Systems*, vol. 118, pp. 319-337, 2001.
- [130] Y. Jin, "Fuzzy modeling of high-dimensional systems: complexity reduction and interpretability improvement," *IEEE Transactions on Fuzzy Systems*, vol. 8, pp. 212-221, 2000.
- [131] R. Alcalá, J. Alcalá-Fdez, M. Gacto, and F. Herrera, "Rule base reduction and genetic tuning of fuzzy systems based on the linguistic 3-tuples representation," *Soft Computing*, vol. 11, pp. 401-419, 2007.
- [132] R. Alcalá, J. Alcalá-Fdez, F. Herrera, and J. Otero, "Genetic learning of accurate and compact fuzzy rule based systems based on the 2-tuples linguistic representation," *International Journal of Approximate Reasoning*, vol. 44, pp. 45-64, 2007.
- [133] L. B. Booker, D. E. Goldberg, and J. H. Holland, "Classifier systems and genetic algorithms," in *Machine learning: paradigms and methods*, J. Carbonell, Ed. Cambridge, MA: MIT Press, 1990, pp. 235-282.
- [134] S. F. Smith, "A Learning System Based on Genetic Algorithms," vol. Ph.D. dissertation. Pittsburgh, PA: Univ. Pittsburgh, 1980.
- [135] A. Bonarini, "Evolutionary learning of fuzzy rules: competition and cooperation," in *Fuzzy Modeling: Paradigms and Practice*, W. Pedrycz, Ed. Norwell, MA: Kluwer Academic Press, 1996, pp. 265-284.
- [136] P. Thrift, "Fuzzy Logic Synthesis with Genetic Algorithms," in *Procs. of the 4th Intl. Conf. on Genetic Algorithms*, pp. 509-513, 1991.
- [137] R. Alcalá, J. Casillas, O. Cordon, and F. Herrera, "Learning TSK Fuzzy Rule-Based Systems from Approximate Ones by means of MOGUL Methodology," *Technical Report DECSAI-000121, Dept. of Computer Science and A.I., University of Granada*, 2000.
- [138] A. Gonzalez and R. Perez, "Selection of relevant features in a fuzzy genetic learning algorithm," *IEEE Transactions on Systems, Man and Cybernetics, Part B*, vol. 31, pp. 417-425, 2001.
- [139] H. Ishibuchi and T. Yamamoto, "Effects of Three-Objective Genetic Rule Selection on the Generalization Ability of Fuzzy Rule-Based Systems," in *Evolutionary Multi-Criterion Optimization*, 2003, pp. 608-622.

- [140] H. Ishibuchi, T. Nakashima, and T. Kuroda, "A hybrid fuzzy genetics-based machine learning algorithm: hybridization of Michigan approach and Pittsburgh approach," in *Proc. of IEEE Intl. Conf. on Systems, Man, and Cybernetics*, vol. 1, pp. 296-301, 1999.
- [141] H. Ishibuchi, T. Yamamoto, and T. Nakashima, "Hybridization of fuzzy GBML approaches for pattern classification problems," *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 35, pp. 359-365, 2005.
- [142] L. Castillo, A. Gonzalez, and R. Perez, "Including a simplicity criterion in the selection of the best rule in a fuzzy genetic learning algorithm," *Fuzzy Sets and Systems*, vol. 120 (2), pp. 309-321, 2001.
- [143] O. Cordon, M. J. del Jesus, and F. Herrera, "Genetic learning of fuzzy rule-based classification systems co-operating with fuzzy reasoning methods," *International Journal of Intelligent Systems*, vol. 13, pp. 1025-1053, 1998.
- [144] A. F. Gomez-Skarmeta and F. Jimenez, "Fuzzy modeling with hybrid systems," *Fuzzy Sets and Systems*, vol. 104, pp. 199-208, 1999.
- [145] H. Ishibuchi, T. Murata, and I. B. Turksen, "Single-objective and two-objective genetic algorithms for selecting linguistic rules for pattern classification problems," *Fuzzy Sets and Systems*, vol. 89, pp. 135-150, 1997.
- [146] H. Ishibuchi, K. Nozaki, N. Yamamoto, and H. Tanaka, "Acquisition of fuzzy classification knowledge using genetic algorithms," in *Proc. of the 3rd IEEE Conf. on Fuzzy Systems*, vol. 3, pp. 1963-1968, 1994.
- [147] C. H. Wang, T. P. Hong, and S. S. Tseng, "Integrating fuzzy knowledge by genetic algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 2, pp. 138 - 149, 1998.
- [148] S. Ohtani, H. Kikuchi, R. R. Yager, and S. Nakanishi, "Fuzzy system identification for composite operation and fuzzy relation by genetic algorithms," in *Proc. of 1st Intl. Conf. on Knowledge-Based Intelligent Electronic Systems*, vol. 1, pp. 289 - 295, 1997.
- [149] M. R. Delgado, F. Von Zuben, and F. Gomide, "Modular and hierarchical evolutionary design of fuzzy systems," in *Proc. of the Genetic and Evolutionary Computation Conf.*, pp. 180-187, 1999.
- [150] M. R. Delgado, F. Von Zuben, and F. Gomide, "An extended evolutionary approach toward optimal design of fuzzy systems," in *Proc. of the 3rd. World Multi Conf. on Systemics, Cybernetics and Informatics*, pp. 493-500, 1999.

- [151] O. Cordon, M. J. del Jesus, and F. Herrera, "A proposal on reasoning methods in fuzzy rule-based classification systems," *International Journal of Approximate Reasoning*, vol. 20, pp. 21-45, 1999.
- [152] J. Kinzel, F. Klawonn, and R. Kruse, "Modifications of genetic algorithms for designing and optimizing fuzzy controllers," in *Proc. of the IEEE Congress on Evolutionary Computation (CEC)*, pp. 28-33, 1994.
- [153] J. M. F. Garrido and I. R. Ramos, "A methodology for constructing fuzzy rule-based classification systems," *Mathware & Soft Computing*, vol. 7, pp. 185-197, 2000.
- [154] S. E. Papadakis and J. B. Theocharis, "A GA-based fuzzy modeling approach for generating TSK models," *Fuzzy Sets and Systems*, vol. 131, pp. 121-152, 2002.
- [155] O. Cordon, F. Herrera, and P. Villar, "Generating the knowledge base of a fuzzy rule-based system by the genetic learning of the data base," *IEEE Transactions on Fuzzy Systems*, vol. 9, pp. 667-674, 2001.
- [156] O. Nelles, M. Fischer, and B. Muller, "Fuzzy rule extraction by a genetic algorithm and constrained nonlinear optimization of membership functions," in *Proc. of the IEEE Intl. Conf. on Fuzzy Systems*, vol. 1, pp. 213-219, 1996.
- [157] A. Homaifar and E. McCormick, "Simultaneous design of membership functions and rule sets for fuzzy controllers using genetic algorithms," *IEEE Transactions on Fuzzy Systems*, vol. 3, pp. 129-139, 1995.
- [158] J. Liska and S. S. Melsheimer, "Complete design of fuzzy logic systems using genetic algorithms," in *Proc. of the IEEE Intl. Conf. on Fuzzy Systems*, vol. 2, pp. 1377 - 1382, 1994.
- [159] C. C. Wong and N. S. Lin, "Rule extraction for fuzzy modeling," *Fuzzy Sets and Systems*, vol. 88, pp. 23-30, 1997.
- [160] D. Park, A. Kandel, and G. Langholz, "Genetic-based new fuzzy reasoning models with application to fuzzy control," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 24, pp. 39-47, 1994.
- [161] Y. Shi, R. Eberhart, and Y. Chen, "Implementation of evolutionary fuzzy systems," *IEEE Transactions on Fuzzy Systems*, vol. 7, pp. 109-119, 1999.
- [162] R. Alcalá, J. Alcalá-Fdez, and F. Herrera, "A proposal for the genetic lateral tuning of linguistic fuzzy systems and its interaction with rule selection," *IEEE Transactions on Fuzzy Systems*, vol. 15, pp. 616-635, 2007.

- [163] M. S. Kim, C. H. Kim, and J. J. Lee, "Evolving compact and interpretable Takagi-Sugeno fuzzy models with a new encoding scheme," *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 36, pp. 1006-1023, 2006.
- [164] T. K. Kohonen, *Self-Organization and Associative Memory*, 3rd ed. New York: Springer-Verlag, 1989.
- [165] S. G. Akl, *The Design and Analysis of Parallel Algorithms*. Englewood Cliffs, NJ: Prentice-Hall, 1991.
- [166] E. Alba and J. M. Troya, "A survey of parallel distributed genetic algorithms," *Complexity*, vol. 4, pp. 31-52, 1999.
- [167] I. Rojas, J. Gonzalez, H. Pomares, F. J. Rojas, F. J. Fernandez, and A. Prieto, "Multidimensional and multideme genetic algorithms for the construction of fuzzy systems," *International Journal of Approximate Reasoning*, vol. 26, pp. 179-210, 2001.
- [168] O. Castillo and P. Melin, "Evolutionary design and applications of hybrid intelligent systems," *International Journal of Innovative Computing and Applications*, vol. 1, pp. 48-62, 2007.
- [169] M. R. Delgado, F. Von Zuben, and F. Gomide, "Evolutionary design of Takagi-Sugeno fuzzy systems: a modular and hierarchical approach," in *Proc. of the IEEE Intl. Conf. on Fuzzy Systems*, pp. 447-452, 2000.
- [170] M. R. Delgado, F. Von Zuben, and F. Gomide, "Optimal parameterization of evolutionary Takagi-Sugeno fuzzy systems," in *Proc. of the Knowledge-Based Systems Conf.*, pp. 650-657, 2000.
- [171] X. Chang and J. H. Lilly, "Evolutionary design of a fuzzy classifier from data," *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 34, pp. 1894-1906, 2004.
- [172] M. Russo, "FuGeNeSys-a fuzzy genetic neural system for fuzzy modeling," *IEEE Transactions on Fuzzy Systems*, vol. 6, pp. 373-388, 1998.
- [173] A. Arslan and M. Kaya, "Determination of fuzzy logic membership functions using genetic algorithms," *Fuzzy Sets and Systems*, vol. 118, pp. 297-306, 2001.
- [174] S.-Y. Ho, H.-M. Chen, S.-J. Ho, and T.-K. Chen, "Design of accurate classifiers with a compact fuzzy-rule base using an evolutionary scatter partition of feature space," *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 34, pp. 1031-1044, 2004.

- [175] S. Medasani, J. Kim, and R. Krishnapuram, "An overview of membership function generation techniques for pattern recognition," *International Journal of Approximate Reasoning*, vol. 19, pp. 391-417, 1998.
- [176] J. M. Mendel, *Uncertain Rule-Based Fuzzy Logic Systems: Introduction and New Directions*. Englewood Cliffs: Prentice Hall, 2001.
- [177] J. M. Mendel and R. I. B. John, "Type-2 fuzzy sets made simple," *IEEE Transactions on Fuzzy Systems*, vol. 10, pp. 117-127, 2002.
- [178] H. Huang, M. Pasquier, and C. Quek, "Optimally evolving irregular-shaped membership functions for fuzzy systems," in *Proc. of the IEEE Congress on Evolutionary Computation (CEC)*, Vancouver, BC, pp. 3309-3316, 2006.
- [179] L. Pagie and M. Mitchell, "A comparison of evolutionary and coevolutionary search," *International Journal of Computational Intelligence & Application*, vol. 2, pp. 53-69, 2002.
- [180] C. D. Rosin and R. K. Belew, "New methods for competitive coevolution," *Evolutionary Computation*, vol. 5, pp. 1-29, 1997.
- [181] M. R. Delgado, F. Von Zuben, and F. Gomide, "Coevolutionary genetic fuzzy systems: A hierarchical collaborative approach," *Fuzzy Sets and Systems*, vol. 141, pp. 89-106, 2004.
- [182] C. A. Pena-Reyes, "Island fuzzy CoCo: Island model-based coevolution of fuzzy systems," presented at the Third European Symposium on Intelligent Technologies, Hybrid Systems and their implementation on Smart Adaptive Systems - EUNITE, 2003.
- [183] N. Garcia-Pedrajas, C. Hervas-Martinez, and D. Ortiz-Boyer, "Cooperative coevolution of artificial neural network ensembles for pattern classification," *IEEE Transactions on Evolutionary Computation*, vol. 9, pp. 271-302, 2005.
- [184] S. Guillaume, "Designing fuzzy inference systems from data: An interpretability-oriented review," *IEEE Transactions on Fuzzy Systems*, vol. 9, pp. 426-443, 2001.
- [185] J. V. de Oliveira, "Semantic constraints for membership function optimization," *IEEE Transactions on Systems Man and Cybernetics Part A-Systems and Humans*, vol. 29, pp. 128-138, 1999.
- [186] A. Lotfi, H. C. Andersen, and A. C. Tsoi, "Interpretation preservation of adaptive fuzzy inference systems," *International Journal of Approximate Reasoning*, vol. 15, pp. 379-394, 1996.

- [187] C. L. Blake and C. J. Merz, "UCI Repository of machine learning databases," 1998.
- [188] W. L. Tung and C. Quek, "Falcon: neural fuzzy control and decision systems using FKP and PFKP clustering algorithms," *IEEE Transactions on Systems, Man and Cybernetics, Part B*, vol. 34, pp. 686-695, 2004.
- [189] H. Huang, M. Pasquier, and C. Quek, "HiCEFS - A hierarchical coevolutionary approach for the dynamic generation of fuzzy system," in *Proc. of the IEEE Congress on Evolutionary Computation (CEC)*, Singapore, pp. 3426-3433, 2007.
- [190] C. Quek, M. Pasquier, and B. B. S. Lim, "POP-TRAFFIC: A novel fuzzy neural, approach to road traffic analysis and prediction," *Ieee Transactions on Intelligent Transportation Systems*, vol. 7, pp. 133-146, 2006.
- [191] K. Mathias, J. Schaffer, L. Eshelman, and M. Mani, "The effects of control parameters and restarts on search stagnation in evolutionary programming," in *Parallel Problem Solving from Nature — PPSN V*, 1998, pp. 398-407.
- [192] H. Ishibuchi, T. Nakashima, and T. Kuroda, "A hybrid fuzzy genetics-based machine learning algorithm: hybridization of Michigan approach and Pittsburgh approach," in *Proc. of IEEE Intl. Conf. on Systems, Man, and Cybernetics*, pp. 296-301, 1999.
- [193] C. A. Coello, "An updated survey of GA-based multiobjective optimization techniques," *ACM Computing Surveys*, vol. 32, pp. 109-143, 2000.
- [194] K. Deb, *Multi-Objective Optimization Using Evolutionary Algorithms*. New York: Wiley, 2001.
- [195] E. W. Saad, D. V. Prokhorov, and D. C. Wunsch, II, "Comparative study of stock trend prediction using time delay, recurrent and probabilistic neural networks," *IEEE Transactions on Neural Networks*, vol. 9, pp. 1456-1470, 1998.
- [196] J. W. Smith, J. E. Everhart, W. C. Dickson, W. C. Knowler, and R. S. Johannes, "Using the ADAP learning algorithm to forecast the onset of diabetes mellitus," in *Proc. of Symp. on Computer Applications and Medical Care*, pp. 261-265, 1998.
- [197] R. K. Bock, A. Chilingarian, M. Gaug, F. Hakl, T. Hengstebeck, M. Jirina, J. Klaschka, E. Kotrc, P. Savicky, S. Towers, A. Vaiciulis, and W. Wittek, "Methods for multidimensional event classification: a case study using images from a Cherenkov gamma-ray telescope," *Nuclear*

- Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 516, pp. 511-528, 2004.
- [198] I. C. Yeh, "Modeling of strength of high-performance concrete using artificial neural networks," *Cement and Concrete Research*, vol. 28, pp. 1797-1808, 1998.
- [199] G. Deboeck, *Trading on the Edge: Neural, Genetic, and Fuzzy Systems for Chaotic Financial Markets*. New York: Wiley, 1994.
- [200] R. Trippi, *Chaos and Nonlinear Dynamics in the Financial Markets: Theory, Evidence, and Applications*. Chicago, IL: Irwin Professional Pub., 1995.
- [201] J. S. Abarbanell and B. J. Bushee, "Fundamental analysis, future earnings, and stock prices," *Journal of Accounting Research*, vol. 35, pp. 1-24, 1997.
- [202] M. J. Pring, *Technical Analysis Explained: The Successful Investor's Guide to Spotting Investment Trends and Turning Points*, 4th ed. New York: McGraw-Hill, 2002.
- [203] J. Moody and M. Saffell, "Learning to trade via direct reinforcement," *IEEE Transactions on Neural Networks*, vol. 12, pp. 875-889, 2001.
- [204] N. Kasabov, "Evolving fuzzy neural networks - algorithms, applications and biological motivation," in *Methodologies for the Conception, Design and Application of Soft Computing*, T. Yamakawa and G. Matsumoto, Eds. Singapore: World Scientific, 1998, pp. 271-274.
- [205] K. K. Ang and C. Quek, "RSPOP: Rough set-based pseudo outer-product fuzzy rule identification algorithm," *Neural Computation*, vol. 17, pp. 205-243, 2005.
- [206] M. B. Kennel, R. Brown, and H. D. I. Abarbanel, "Determining embedding dimension for phase-space reconstruction using a geometrical construction," *Physical Review A*, vol. 45, pp. 3403-3411, 1992.
- [207] T. Denoeux, "A k-nearest neighbor classification rule based on Dempster-Shafer theory," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 25, pp. 804-813, 1995.
- [208] J. H. Friedman, "Regularized discriminant analysis," *Journal of the American Statistical Association*, vol. 84, pp. 165-175, 1989.

- [209] P. Langley, W. Iba, and K. Thompson, "An analysis of Bayesian classifiers," in *Proc. of the 10th National Conf. on Artificial Intelligence*, pp. 223-228, 1992.
- [210] W. Duch, R. Setiono, and J. M. Zurada, "Computational intelligence methods for rule-based data understanding," *Proceedings of the IEEE*, vol. 92, pp. 771-805, 2004.
- [211] F. Hoffmann, "Combining boosting and evolutionary algorithms for learning of fuzzy classification rules," *Fuzzy Sets and Systems*, vol. 141, pp. 47-58, 2004.
- [212] C. Schaffer, "Selecting a classification method by cross-validation," *Machine Learning*, vol. 13, pp. 135-143, 1993.
- [213] T. Piran, "The physics of gamma-ray bursts," *Reviews of Modern Physics*, vol. 76, pp. 1143-68, 2004.
- [214] I. H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*. Amsterdam: Morgan Kaufman, 2005.
- [215] D. A. Abrams, *Design of Concrete Mixtures: Structural Materials*, Research Laboratory, Lewis Institute, Chicago, 1918.
- [216] F. A. Oluokun, "Fly ash concrete mix design and the water-cement ratio law," *ACI Materials Journal*, vol. 91, pp. 362-371, 1994.
- [217] K. J. Bois, A. D. Benally, and R. Zoughi, "Microwave near-field reflection property analysis of concrete for material content determination," *IEEE Transactions on Instrumentation and Measurement*, vol. 49, pp. 49-55, 2000.
- [218] E.-J. Yeoh, M. E. Ross, S. A. Shurtleff, W. K. Williams, D. Patel, R. Mahfouz, F. G. Behm, S. C. Raimondi, M. V. Relling, A. Patel, C. Cheng, D. Campana, D. Wilkins, X. Zhou, J. Li, H. Liu, C.-H. Pui, W. E. Evans, C. Naeve, L. Wong, and J. R. Downing, "Classification, subtype discovery, and prediction of outcome in pediatric acute lymphoblastic leukemia by gene expression profiling," *Cancer Cell*, vol. 1, pp. 133-143, 2002.
- [219] I. Guyon and A. Elisseeff, "An Introduction to Variable and Feature Selection," *Journal of Machine Learning Research*, vol. 3, pp. 1157-1182, 2003.
- [220] H. Hagra, V. Callaghan, and M. Colley, "Online learning of fuzzy behaviour co-ordination for autonomous agents using genetic algorithms and real-time interaction with the environment," in *Proc. of the IEEE Intl. Conf. on Fuzzy Systems*, vol. 2, pp. 853-858, 2000.

- [221] K. A. De Jong, "Genetic algorithms are NOT function optimizers," in *Foundations of Genetic Algorithms 2*. San Mateo, CA: Morgan Kaufmann, 1993.
- [222] J. Hekanaho, "DOGMA: A GA-based relational learner," in *Inductive Logic Programming*, 1998, pp. 205-214.
- [223] T.-P. Hong, H.-S. Wang, and W.-C. Chen, "Simultaneously applying multiple mutation operators in genetic algorithms," *Journal of Heuristics*, vol. 6, pp. 439-455, 2000.
- [224] M. Pablo, "Memetic algorithms: a short introduction," in *New Ideas in Optimization*: McGraw-Hill Ltd., UK, 1999, pp. 219-234.
- [225] J. Sun, Q. Zhang, and E. P. K. Tsang, "DE/EDA: A new evolutionary algorithm for global optimization," *Information Sciences*, vol. 169, pp. 249-262, 2005.
- [226] E. Alba and M. Tomassini, "Parallelism and evolutionary algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 6, pp. 443-462, 2002.
- [227] L. Schumaker, *Spline Functions: Basic Theory*, 3rd ed. New York: Cambridge University Press, 2007.
- [228] G. M. Phillips, *Interpolation and Approximation by Polynomials*. New York: Springer-Verlag, 2003.
- [229] R. J. Oentaryo and M. Pasquier, "A reduced rule-based localist network for data comprehension," in *Proc. of the IEEE Intl. Joint Conf. on Neural Networks (IJCNN)*, Hong Kong, pp. 2661-2668, 2008.
- [230] R. L. Mantaras, *Approximate reasoning models*. Chichester, West Essex, England: Ellis Horwood Limited, 1990.
- [231] I. B. Turksen and Z. Zhong, "An approximate analogical reasoning scheme based on similarity measures and interval valued fuzzy sets," *Fuzzy Sets and Systems*, vol. 34, pp. 323-346, 1990.
- [232] J. M. Keller, R. R. Yager, and H. Tahani, "Neural network implementation of fuzzy logic," *Fuzzy Sets & Systems*, vol. 45, pp. 1-12, 1992.