Hierarchical Fuzzy Systems for Function Approximation on Discrete Input Spaces With Application

Xiao-Jun Zeng, *Member, IEEE*, John Yannis Goulermas, *Member, IEEE*, Panos Liatsis, *Member, IEEE*, Di Wang, *Member, IEEE*, and John A. Keane, *Member, IEEE*

Abstract—**This paper investigates the capabilities of hierarchical fuzzy systems to approximate functions on discrete input spaces. First, it is shown that any function on a discrete space has an arbitrary separable hierarchical structure and can be naturally approximated by hierarchical fuzzy systems. As a by-product of this result, a discrete version of Kolmogorov's theorem is obtained; second, it is proven that any function on a discrete space can be approximated to any degree of accuracy by hierarchical fuzzy systems with any desired separable hierarchical structure. That is, functions on discrete spaces can be approximated more simply and flexibly than those on continuous spaces; third, a hierarchical fuzzy system identification method is proposed in which human knowledge and numerical data are combined for system construction and identification. Finally, the proposed method is applied to the market condition performance modeling problem in site selection decision support and shows the better performance in both accuracy and interpretability than the regression and neural network approaches. In additions, the reason and mechanism why hierarchical fuzzy systems outperform regression and neural networks in this type of application are analyzed.**

Index Terms—**Discrete spaces, function approximation, hierarchical fuzzy systems, Kolmogorov's theorem, site selection decision support, universal approximation.**

I. Introduction

THE MAIN motivations for our research in hierarchical fuzzy systems for function approximation on discrete input spaces are as follows.

The first motivation is due to the lack of systematic investigation of the representation capabilities of fuzzy systems and hierarchical fuzzy systems to function approximation on discrete input spaces. In contrast, following the original work by Buckley [2], Kosko [15], and Wang [31], the capabilities of fuzzy and hierarchical fuzzy systems to approximate functions on continuous input spaces have been systematically investigated, for example [3], [12], [17], [35]–[37] for standard fuzzy systems and [10], [13], [28], [33], [39] for hierarchical fuzzy systems. Theoretically speaking, these results can be extended to approximate functions on discrete input spaces, as any function on discrete spaces can be expanded to be a continuous one [20] (which interpolates the given discrete function) and then be approximated by fuzzy or hierarchical fuzzy systems based on these existing results. However, such an approach is often less effective and more complicated as it does not take into account the special features of functions on discrete spaces. To overcome this weakness, a focus here is to recognize these special features and simplify approximation schemes that allow function approximation on discrete input spaces to be achieved both more simply and more effectively.

The second motivation is the widely used aggregation approach to complicated system modeling. In applications, such as applied social, economic, management, and marketing research areas, the complicated high-dimensional system modeling is often formed in two steps: given a system to be modeled, first, classify all input variables into several groups and aggregate the variables inside each group into an index that represents the overall impact of input variables in the group to the system output. Such aggregation schemes are often based on domain experts' intuition, knowledge, and experience. Second, a mathematical model that takes the aggregated indexes as its input variables is constructed to represent the system to be modeled. Despite widespread use, such an approach is basically heuristic and there has been little systematic analysis of its general applicability and effective usage. A focus here is to develop a theoretical foundation, using the framework of hierarchical fuzzy systems, to enhance both applicability and effectiveness of the aggregation approach.

The third motivation is the application of hierarchical fuzzy systems to site selection decision support. Site selection is a key long-term strategic decision faced by many retail and service firms as the disadvantages of poor location are difficult to overcome [7]. The fundamental issue here is to model how the site location and local market conditions affect the performances (sales, revenue, or/and profit) of retail stores, and then, use the identified model to provide the performance estimations of potential locations to support new store location selection. A special feature of building location–market condition

Manuscript received December 21, 2005; revised May 30, 2007 and September 23, 2007; accepted December 1, 2007. First published April 30, 2008; current version published October 8, 2008. This work was supported in part by the U.K. Engineering and Physical Sciences Research Council (EPSRC) under Grant EP/C513355/1 and in part by the Knowledge Support Systems (KSS) Ltd., Manchester, U.K.

X.-J. Zeng and J. A. Keane are with the School of Computer Science, University of Manchester, Manchester, M13 9PL, U.K. (e-mail: x.zeng@manchester.ac.uk; john.keane@manchester.ac.uk).

J. Y. Goulermas is with the Department of Electrical Engineering and Electronics, University of Liverpool, Liverpool L69 3GJ, U.K. (e-mail: j.y.goulermas@liverpool.ac.uk).

P. Liatsis is with the Information and Biomedical Engineering Centre, School of Engineering and Mathematical Sciences, City University, London EC1V 0HB, U.K. (e-mail: p.liatsis@city.ac.uk).

D. Wang is with ThinkAnalytics Ltd., Glasgow G3 7QF, U.K. (e-mail: dwang@thinkanalytics.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TFUZZ.2008.924343

performance (LMCP) models is that a number of location and local market factors that affect performance need to be considered, but there are only a limited number of data available for identifying such models (most retail chains in Europe have only a few hundred or less stores whose location condition and performance data pairs are available for modeling). In addition, for such an LMCP model, its input variables (i.e., the location and local market factors) usually only take, or are available as, discrete values. Therefore, building LMCP models for site selection is a modeling problem with high dimensions, discrete input spaces, and limited available data. A focus here is to develop a new fuzzy system approach to the LMCP modeling.

Based on the previous motivations, this paper will focus on investigating hierarchical fuzzy approach to function approximation on discrete input spaces. Distinguished from the existing research of hierarchical fuzzy systems (such as [10], [13], [28], [31], [33], and [39]), it is discovered in this paper that systems or functions on discrete input spaces have a special feature—the flexibility of hierarchical representation in the sense that they can be represented by any desired separable hierarchical form. Based on this feature, it is shown in the paper that, first, simpler two-level hierarchical fuzzy systems are universal approximators whereas multilevel hierarchical fuzzy systems are needed in the existing research; second and more importantly, for any given system or function on discrete input spaces and any desired separable hierarchical structure, a hierarchical fuzzy system with the desired hierarchical structure can always be found to approximate the system or function to any degree of accuracy (i.e., flexibility in hierarchical approximation). As a result, hierarchical fuzzy systems can always be designed to present systems to be modeled in such a way that is most suitable to users' need or understanding and makes the utilization of human knowledge easier and more effective during system identification. In other words, flexible hierarchical fuzzy systems presented in this paper allow system modeling or function approximation on discrete spaces to be achieved more simply and effectively than those existing ones.

Furthermore, it should be pointed out that mathematical approximation theory (such as polynomials and spline approximation, etc. [20]), neural networks [11], [19], [22], and statistical approaches (such as nonlinear regressions [23], support vector regressions [24], [29], and kernel methods [25]) also apply for function approximation. Similar to the situation in fuzzy systems, these approaches treat function approximation on discrete input spaces in the same way as on continuous input spaces. Therefore, the analysis and approach proposed in this paper may be extendable to these methods. Based on the discrete version of Kolmogorov's theorem (i.e., Theorem 2) obtained in this paper, some initial applications to neural networks for function approximation on discrete spaces can be found in [40] whereas based on the mixed discrete and continuous input spaces, they can be found in [30].

The rest of the paper is structured as follows. Section II introduces the LMCP modeling problem; Section III analyzes the hierarchical structure representation of functions on discrete spaces; Section IV analyzes the representation capabilities of hierarchical fuzzy systems to approximate functions on discrete spaces; Section V proposes an identification method of hierarchical fuzzy systems on discrete spaces; Section VI applies the proposed identification method to the LMCP modeling problem and compares it with regression and neural network approaches; finally, conclusions are presented in Section VII, and the proofs of all the lemmas and theorems presented are given in the Appendix.

II. LMCP Modeling Problem in Site Selection

This section briefly introduces the LMCP modeling problem that will be used to illustrate the various theoretic results obtained in the paper and to show how hierarchical fuzzy systems can be used to better solve the LMCP modeling problem than the existing regression and neural network methods.

The fundamental component of site selection decision support is to build the LMCP model, that is, a mathematical model that describes how the sales performance of a retail store is determined by its location and local market factors as follows:

$$S = G(x_1, x_2, \ldots, x_n)$$

where S represents the sales performance and $x_1, x_2, \ldots, x_n$ represent the different location and market factors that impact sales performance. Usually the location and market factors in LMCP modeling take discrete values. This is because, first, some market factors by nature are discrete, such as store opening hours or number of local competitors; second, some market variables are difficult to measure accurately, such as visibility and accessibility; and third, some market factors do not need to be measured to an exact accuracy, such as store size or income level. In practice, the granularity of the available data of market factors is usually coarse; for example, there are usually only a few values for visibility and accessibility, and population and average income are measured in thousands. Therefore, for a location or market factor $x_i (i = 1, \ldots, n)$, its possible value space (i.e., the input space) $U_i (i = 1, \ldots, n)$ is finite. As a result, the whole input space for all location–market factors can be denoted as $U = U_1 \times U_2 \times \cdots \times U_n = \mathop{\times}\limits_{i=1}^{n} U_i$ and is discrete. Therefore, the LMCP model is a function on a discrete space.

Once such an LMCP model is identified, then for each potential new site, the model can be used to estimate its sales performance. Based on the estimated sales performance and cost analysis, the profitability of a potential new store at the site and the return of investment can be analyzed and the site selection decision can be made by choosing those sites with the best profitability and return. Dependent on retail sectors and corporate strategies, the most commonly used sale performances are sale volume, profit, and revenue [7]. Sometimes, customer numbers [16] are used. Similarly, the location and market factors considered in the LMCP models can be varied depending on retail sectors and type of business. Normally, the list of the location and market factors includes the location- and trade-area-related factors (visibility, accessibility, location type, etc.), demographic and demand-related factors (population, income level, average consumption, etc.), store-facility- and character-related factors (store size, store facility, store attractiveness, opening hours, number of parking spots, etc.), and competition-related factors (number of competitors, competition strength, market share, etc.).

Typical features of LMCP models, and difficulties involved in their identification, include the following.

1) *High dimensionality:* There are many market factors that impact store performance; for example, 27 factors are listed for general location evaluation in [7], 25 factors are listed for industrial site selection in [34], and 43 factors are listed in [16] for convenient store location selection. Therefore, high dimensionality is a typical feature and difficulty for model identification.
2) *Nonlinearity:* Nonlinearity is evident in [4] and [8]. The existing method to handle nonlinearity is to use several linear models in which each model covers certain types of market conditions [4], [8], and more recently, use nonlinear models such as neural networks [6].
3) *Limited data:* In order to build the LMCP model, it is necessary to have data about different store performances under different location and market conditions. However, as most retail chains in Europe only have a few hundred stores or less, only a few hundred or less pairs of store location–market condition and performance data are available for modeling.

In the literature and practice, the most commonly used method is linear regression [8], [21], while neural networks have been used recently [6]. Analysis of linear regression methods can be found in [21]: the main weakness of the linear regression approach is that the nonlinearity cannot be modeled and the multicollinearity caused by limited data reduces the reliability of the model and forecasting. For neural networks, the concern is that there is not enough data for training, and thus, unrealistic forecasting results are often obtained. This combined with their lack of transparency and interpretability makes it difficult to convince retailers to use neural networks in practice.

III. Hierarchical Structure of Functions on Discrete Spaces

Throughout the rest of the paper, it is assumed that the system or function to be modeled or approximated is a multiinput–single-output (MISO) function defined on discrete spaces as follows:

$$y = G(X) = G(x_1, \ldots, x_n) \tag{1}$$

where $y \in V \subset R$ is the output variable and $X = (x_1, \ldots, x_n) \in U = U_1 \times \cdots \times U_n \subset R^n$ is the input variable vector in which $x_i \in U_i$ and U_i is a discrete space as

$$U_i = \{u_{i,k} | u_{i,k} \in R,\ k = 1, \ldots, N_i\}, \qquad i = 1, \ldots, n. \tag{2}$$

The LMCP model given in the last section is an example of such a function or system on discrete spaces.

In order to apply hierarchical fuzzy systems to approximate systems or functions on discrete spaces, this section discusses whether it is possible and how to decompose a general and high-dimensional function on discrete spaces into a composition of several simpler and lower dimensional functions. For this

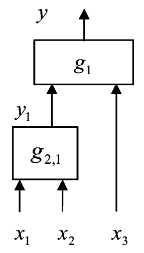


Fig. 1. System with a two-level hierarchical structure.

purpose, the concepts of hierarchical structure of functions are introduced first.

A. Concepts of Hierarchical Structure of Functions

Before formally introducing the concepts of hierarchical structure of functions, an example is discussed.

Example 1: Consider the following three variable functions defined on $U = [1,5] \times [1,5] \times [1,5]$ given in [12]

$$y = G(x_1, x_2, x_3) = \left(1 + x_1^{0.5} + x_2^{-1} + x_3^{-1.5}\right)^2. \tag{3}$$

Let g_1 and $g_{2,1}$ be $g_1(y_1, x_3) = (1 + y_1 + x_3^{-1.5})^2$ and $y_1 = g_{2,1}(x_1, x_2) = x_1^{0.5} + x_2^{-1}$, then $G(x_1, x_2, x_3) = g_1[g_{2,1}(x_1, x_2), x_3]$ which means that 3-D function $G(x_1, x_2, x_3)$ can be decomposed as a composition of 2-D functions $g_1(y_1, x_3)$ and $g_{2,1}(x_1, x_2)$. Based on such decomposition, $G(x_1, x_2, x_3)$ can be represented as a function with a two-level hierarchical structure given in Fig. 1. On the other hand, the function given in (3) can also be decomposed as $G(x_1, x_2, x_3) = g_1[g_{2,1}(x_1, x_3), x_2]$ with $g_1(y_1, x_2) = (y_1 + x_2^{-1})^2$ and $y_1 = g_{2,1}(x_1, x_3) = 1 + x_1^{0.5} + x_3^{-1.5}$.

The previous example shows that $y = G(x_1, x_2, x_3)$ given in (3) can be decomposed as a composition of two simpler or lower dimensional functions. Such a function is termed a function with hierarchical structure whose formal definition is as follows.

Definition 1: Let $G(X)$ be a MISO function given in (1). If there exist $m + 1$ functions

$$y = g_1(Y, X_1) = g_1\left(y_1, \ldots, y_m, x_{i_1^{(1)}}, \ldots, x_{i_{n_1}^{(1)}}\right)$$
$$1 \le i_k^{(1)} \le n, \qquad k = 1, \ldots, n_1 \tag{4}$$

$$y_j = g_{2,j}(X_{2,j}) = g_{2,j}\left(x_{i_1^{(2,j)}}, \ldots, x_{i_{n_{2,j}}^{(2,j)}}\right)$$
$$1 \le i_k^{(2,j)} \le n, \quad k = 1, \ldots, n_{2,j}, \quad j = 1, \ldots, m \tag{5}$$

which are simpler or lower dimensional functions than $G(X)$ such that

$$G(X) = g_1\left[g_{2,1}(X_{2,1}), \ldots, g_{2,m}(X_{2,m}), X_1\right] \tag{6}$$

then $G(X)$ is known as a function with hierarchical structure, where

$$Y = (y_1, \ldots, y_m), \qquad X_1 = \left(x_{i_1^{(1)}}, \ldots, x_{i_{n_1}^{(1)}}\right)$$

$$X_{2,j} = \left(x_{i_1^{(2,j)}}, \ldots, x_{i_{n_{2,j}}^{(2,j)}}\right), \qquad j = 1, \ldots, m. \tag{7}$$

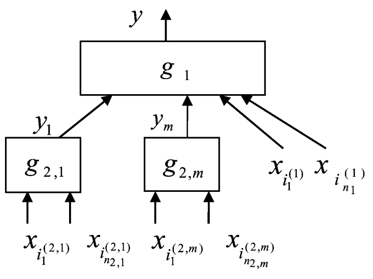


Fig. 2. Two-level hierarchical system.

Denote the input variable sets of subfunctions g_1 and $g_{2,j}$ $(j = 1,\ldots,m)$ as

$$G_1 = \left\{x_{i_1^{(1)}},\ldots,x_{i_{n_1}^{(1)}}\right\}, \qquad G_{2,j} = \left\{x_{i_1^{(2,j)}},\ldots,x_{i_{n_{2,j}}^{(2,j)}}\right\}. \tag{8}$$

Then $G(X)$ is known as a function with $G_1, G_{2,1},\ldots,G_{2,m}$ as its hierarchical structure. Further, if $G_1, G_{2,1},\ldots,G_{2,m}$ are disjoint [i.e., if the input variable sets of subfunctions g_1 and $g_{2,j}$ $(j = 1,\ldots,m)$ are disjoint] as follows:

$$G_1 \cap G_{2,j} = \oslash, \qquad j = 1,\ldots,m \tag{9}$$

$$G_{2,j} \cap G_{2,j'} = \oslash, \qquad j \neq j', \qquad j, j' = 1,\ldots,m \tag{10}$$

$$G_1 \cup G_{2,1} \cup \cdots \cup G_{2,m} = \{x_1,\ldots,x_n\} \tag{11}$$

then $G(X)$ is known as a function with a separable hierarchical structure.

When function $y = G(X) = G(x_1,\ldots,x_n)$ can be decomposed as a composition of $m+1$ functions g_1 and $g_{2,j}$ $(j = 1,\ldots,m)$ as given in (6), it can be represented by a hierarchical system as shown in Fig. 2 in which g_1 is known as the higher level subsystem or subfunction and $g_{2,j}$ $(j = 1,\ldots,m)$ as the lower level subsystems or subfunctions. The main feature of such a hierarchical system is that each output variable of a lower level subsystem is an input variable of the higher level system.

Remark 1: For simplification, Definition 1 is given based on two levels of hierarchical structure. If $g_{2,j}$ is a function with a hierarchical structure and Definition 1 is applied to $g_{2,j}$, then multilevel hierarchical structures can be obtained.

An interesting special case of the separable hierarchical structure for functions is that some functions can be decomposed into any predefined separable hierarchical structure. This case is formalized by the following definition.

Definition 2: Let $G(X)$ be a MISO function given in (1). If, for any disjoint grouping of input variables $\{x_1,\ldots,x_n\}$ into $m+1$ groups G_1 and $G_{2,j}$ $(j = 1,\ldots,m)$ as given by (8)–(11), there exist functions g_1 and $g_{2,j}$ $(j = 1,\ldots,m)$ such that $G(X) = g_1\left[g_{2,1}(X_{2,1}),\ldots,g_{2,m}(X_{2,m}),X_1\right]$, then $G(X)$ is known as a function with arbitrary separable hierarchical structure.

B. Hierarchical Structure Representation for Functions on Discrete Spaces

Based on the concepts of hierarchical structure of functions, in this section, the hierarchical structure representation theorem for functions on discrete spaces is given. For this purpose, a lemma is introduced first.

Lemma 1: Let $G(X)$ be a function given in (1) and (2). If G_1 and $G_{2,j}$ $(j = 1,\ldots,m)$ are a given disjoint grouping of input variables $\{x_1,\ldots,x_n\}$ satisfying (8)–(11) and $y_j = M_{2,j}(X_{2,j}) = M_{2,j}(x_{i_1^{(2,j)}},\ldots,x_{i_{n_{2,j}}^{(2,j)}})$ $(j = 1,\ldots,m)$ defined on $U_{G_{2,j}} = \overset{n_{2,j}}{\underset{k=1}{\times}} U_{i_k^{(2,j)}}$ is a one-to-one mapping from $U_{G_{2,j}}$ to R, then there exists a function $g_1\,(Y, X_1) = g_1(y_1,\ldots,y_m,X_1)$ such that $G(X) = g_1\left[M_{2,1}(X_{2,1}),\ldots,M_{2,m}(X_{2,m}),X_1\right]$. That is, $G(X)$ is a function with hierarchical structure as G_1 and $G_{2,j}$ $(j = 1,\ldots,m)$.

The previous lemma shows that, for any given disjoint grouping of $\{x_1,\ldots,x_n\}$ as G_1 and $G_{2,j}$ $(j = 1,\ldots,m)$, whether $G(X)$ can be represented as a function with the given G_1 and $G_{2,j}$ $(j = 1,\ldots,m)$ as its hierarchical structure is related to the existence of one-to-one mappings on discrete spaces. The following lemma shows that such one-to-one mappings not only exist but can also be realized by using some very simple functions.

Lemma 2: Let $U = U_1 \times \cdots \times U_n$ in which U_i is given in (2). Then there exists a real value function $y = M(X) = M(x_1,x_2,\ldots,x_n)$ defined on U such that $M : U \to R$ is a one-to-one mapping [i.e., if $X \neq X'$, then $M(X) \neq M(X')$]. In particular: 1) $M(X)$ can be a linear function, that is, there exists a linear one-to-one mapping from U to R; 2) $M(X)$ can be a fuzzy system, that is, there exists a fuzzy system that forms a one-to-one mapping from U to R.

Remark 2: The previous lemma shows that, for a discrete space $U \subset R^n\,(n \geq 2)$, there exist some simple functions such as linear functions that form one-to-one mappings from U to R. This is a property that holds only on discrete spaces but not on continuous spaces. This is because no continuous function could be found to form a one-to-one mapping from a multidimensional continuous space to R (see Proposition A in the Appendix) and as a result it is impossible to find a simple function that is a one-to-one mapping from multidimensional continuous space to R. This is the basic reason why functions on discrete spaces have very flexible hierarchical structure as shown in the following theorem.

Theorem 1 (Flexible Hierarchical Structure Representation Property): Let $G(X)$ be a MISO function given in (1) and (2). Then, for any disjoint grouping of the input variables $\{x_1,\ldots,x_n\}$ into $m+1$ groups G_1 and $G_{2,j}$ $(j = 1,\ldots,m)$ satisfying (8)–(11), there exist functions g_1 and $g_{2,j}$ $(j = 1,\ldots,m)$ (in which $g_{2,j}$ can be as simple as linear functions or fuzzy systems) such that $G(X) = g_1\left[g_{2,1}(X_{2,1}),\ldots,g_{2,m}(X_{2,m}),X_1\right]$ That is, any MISO function on discrete spaces has the arbitrary separable hierarchical structure.

The previous theorem has two implications.

1) As any function on discrete space has a simple and flexible hierarchical structure as shown in the previous theorem, it can be naturally approximated by hierarchical fuzzy systems based on the idea of hierarchical structure match. That is, for $G(X) =$

$g_1\,[g_{2,1}(X_{2,1}),\ldots,g_{2,m}(X_{2,m}),X_1]$, first, fuzzy subsystems g_1 or $g_{2,j}$ $(j=1,\ldots,m)$ can be constructed to approximate subfunctions g_1 and $g_{2,j}$ $(j=1,\ldots,m)$, respectively, and then, combine these fuzzy subsystems to form a hierarchical fuzzy system based on the same hierarchical structure as $G(X)$. In this way, the hierarchical fuzzy system can be formed as $F(X)=f_1\,[f_{2,1}(X_{2,1}),\ldots,f_{2,m}(X_{2,m}),X_1]$. Due to such a hierarchical structure match, it is not difficult to "guess" that $F(X)$ will be very close to $G(X)$ if f_1 or $f_{2,j}$ $(j=1,\ldots,m)$ are very close to g_1 and $g_{2,j}$ $(j=1,\ldots,m)$, respectively. Theorem 3, given in the next section, will verify this "guess."

2) For a given function $G(X)$ on a discrete space, as shown in Theorem 1, for any disjoint grouping of the input variables $\{x_1,\ldots,x_n\}$ (i.e., we can arbitrarily choose the number of groups and which input variables are in each group), $G(X)$ has a hierarchical representation based on the given grouping. Such a property gives flexibility during the construction of hierarchical fuzzy systems to approximate a given function or system based on modeling constraint (e.g., limited available data) and usability (ease of understanding and interpretation). This aspect will be discussed in more detail in the following sections.

As an application of the previous general hierarchical structure representation theorem, a special hierarchical structure will be considered in the next section that will lead to a simple discrete version of Kolmogorov's theorem.

C. Discrete Space Version of Kolmogorov's Theorem

Kolmogorov's theorem is the famous solution of Hilbert's 13th problem. This theorem has become well known in the last two decades due to its importance in the analysis and application of neural networks for function approximation [22]. In fact, the first result proving that neural networks are universal approximators by Hecht-Nielsen [9] and many representation capabilities and some algorithms of neural networks for function approximation are obtained based on this famous theorem (see [11], [22], and [27]). In addition, Kolmogorov's theorem has also been applied to nonlinear circuit and system theory, statistical pattern recognition, and image and multidimensional signal processing (see [5] and the references there). Therefore, Kolmogorov's theorem is a useful theoretic tool with wide applicability in many different fields.

Kolmogorov's Theorem: For any given real-value continuous function $G(x_1,\ldots,x_n)$ on $U=\prod_{i=1}^{n}[\alpha_i,\beta_i]=[0,1]^n$, there exist one variable continuous function g that depends on $G(x_1,\ldots,x_n)$ and ψ that is independent of $G(x_1,\ldots,x_n)$ such that

$$G(x_1,\ldots,x_n)=\sum_{q=0}^{2n} g\left[\sum_{i=1}^{n} a_i\psi(x_i+b_q)+c_q\right] \tag{12}$$

where $a_i=\lambda^{i-1}(i=1,\ldots,n)$, $b_q=rq$, and $c_q=q$ $(q=0,1,\ldots,2n)$ with λ and γ being real constants.

Remark 3: The previous theorem is a simplified version by Lorentz [18] and Sprecher [26] from Kolmogorov's original formula:

$$G(x_1,x_2,\ldots,x_n)=\sum_{q=0}^{2n} g_q\left[\sum_{i=1}^{n}\psi_{qi}(x_i)\right]. \tag{13}$$

Kolmogorov's theorem shows that a multidimensional function can be represented as a composition of two signal variable continuous functions g and ψ as shown in (12). In particular, ψ could be chosen to be independent on G. If such ψ can be found, then Kolmogorov's theorem shows that a multidimensional function approximation problem [i.e., to approximate $G(x_1,\ldots,x_n)$] can be transformed to a 1-D function approximation problem (i.e., to approximate the signal variable function g). Unfortunately, the construction of ψ is extremely complicated and all that has been known so far is that it does exist as the uniform limit of a sequence continuous nondecreasing piecewise-linear function (see [26] for the further details). In fact, this is the main difficulty in applying Kolmogorov's theorem in applications. Due to this reason, the applications of Kolmogorov's theorem to neural networks, nonlinear circuit and system, statistical pattern recognition, image and signal processing have been focused on finding suitable approximators to ψ (see [5], [11], [22], and [27] for more details).

The previous discussion shows the usefulness and limitation of Kolmogorov's theorem for function approximation and other applications. Then, when considering function approximation on discrete spaces, a natural and interesting question to ask is what the discrete version of Kolmogorov's theorem looks like and whether such a discrete version has the same limitation. By taking a special hierarchical structure in Theorem 1 as $m=1$, $G_1=\phi$, and $G_{2,1}=\{x_1,\ldots,x_n\}$ and choosing $g_{2,1}$ as a linear function, a simple discrete version of Kolmogorov's theorem as follows can be obtained immediately from Theorem 1.

Theorem 2 (A Discrete Version of Kolmogorov's Theorem): For any given real-value multidimensional function $G(X)$ given in (1) and (2), there exist a linear function $L(X)=a_0+\sum_{i=1}^{n}a_ix_i$ that could be independent of G and one variable function g that depends on G such that

$$G(X)=g[L(X)]=g\left(\sum_{i=1}^{n}a_ix_i+a_0\right). \tag{14}$$

Remark 4: It can be seen from the proof of Lemma 1 in the Appendix that, for any one-to-one linear or nonlinear function $L(X)$, there exists a one variable function g such that (14) holds. In other words, there are many different possible choices for $L(X)$.

Comparing (12) and (14), it can be found that the discrete version is much simpler in the following aspects.

1) Denote $g_q(X)=g\,[\sum_{i=1}^{n}a_i\psi(x_i+rq)+q]\,(q=0,1,\ldots,2n)$; then (12) in Kolmogorov's theorem can be represented as $G(X)=\sum_{q=0}^{2n}g_q(X)$. That is, for a function $G(X)$ on a multidimensional continuous space, it requires $2n+1$ terms as $g_q(X)$ in order to reproduce it. In contrast, for a function $G(X)$ on a multidimensional discrete space, one such term is enough to reproduce $G(X)$ as shown in (14).

2) In (12) of Kolmogorov's theorem, $\psi(x)$ is only known to exist but is complicated and how to construct it is unknown. In contrast, its counterpart in (14) is $\psi(x) = x$ as (14) can be reformulated as

$$G(X) = g\left(\sum_{i=1}^{n} a_i x_i + a_0\right) = g\left[\sum_{i=1}^{n} a_i \psi(x_i) + a_0\right].$$

In other words, $\psi(x)$ can be the simplest 1-D linear function (it should be noted that $\psi(x) = x$ is not the only choice but it is the simplest one). As $\psi(x)$ is explicitly given and very simple, the discrete version of Kolmogorov's theorem given in Theorem 2 is more convenient to use.

Now we analyze why functions on discrete spaces allow such simple structure decomposition as given in (14), whereas functions on continuous functions require much more complicated structure decomposition as shown in (12). The fundamental reason behind this is due to the fact (as shown in Lemma 2) that there exist some simple functions such as linear functions that form one-to-one mappings from a multidimensional discrete space U to R but this is impossible on continuous spaces (see aforementioned Remark 2). Therefore, in the proof of Kolmogorov's theorem (see the details in [26]), $2n+1$ functions $M_q(X) = \sum_{i=1}^{n} a_i \psi(x_i + rq) + q (q = 0, 1, \ldots, 2n)$ have to be constructed in order to form one-to-one mapping from $U = \mathop{\times}\limits_{i=1}^{n} [\alpha_i, \beta_i]$ $(n \geq 2)$ to R in the sense that, for $X \neq X'$, the corresponding images under $M_q(X) (q = 0, 1, \ldots, 2n)$ are distinct (see [23] for more details). This in fact is the reason why $2n + 1$ terms like $g_q(X) = g[\sum_{i=1}^{n} a_i \psi(x_i + rq) + q]$ $(q = 0, 1, \ldots, 2n)$ are needed in Kolmogorov's theorem.

Similar to Kolmogorov's theorem, its discrete version given in Theorem 2 may be more useful for theoretic analysis but less useful directly in some applications. For example, applying this theorem has led to several new results in neural networks, which include several simplified neural networks and hierarchical neural fuzzy systems as new universal approximators on discrete spaces in [40] and new training algorithms to identify neural networks with mixed discrete and continuous variables [30]. Also, the major results obtained in this paper can be derived from this discrete version of Kolmogorov's theorem if it is proved first.

However, there are two reasons that make this theorem perhaps less suitable to be applied directly in some applications. First, for a multidimensional function, the 1-D function $g(z)$ can be very complicated, as the variation in multidimensional spaces is embedded into a 1-D function. Second, the linear one-to-one mapping exists theoretically but is not always feasible for high-dimensional cases where there are many input variables (i.e., n is large) with each input variable having many possible values (i.e., N_j is large). The reason is as the total number of all possible values of input vector $X = (x_1, \ldots, x_n)$ are $\prod_{i=1}^{n} N_i$, then the total number of the possible function output values of a one-to-one mapping $z = L(X)$ is $\prod_{i=1}^{n} N_i$. When n and N_i $(i = 1, 2, \ldots, n)$ are large, such a one-to-one linear mapping is infeasible as all possible function output values are beyond the representation accuracy of float numbers. Therefore, more general hierarchical representation introduced in the last section is needed in which the large number of the input variables are divided into several groups G_1 and $G_{2,j}$ $(j = 1, \ldots, m)$ and then use several simpler functions such as linear functions or fuzzy systems to perform the one-to-one mappings on subspaces $U_{G_{2,j}} = \mathop{\times}\limits_{k=1}^{n_{2j}} U_{i_k^{(2,j)}} (j = 1, \ldots, m)$. In this way, the large n-dimensional function approximation or system modeling problem can be transformed into several smaller $n_{2,j} (j = 1, \ldots, m)$ and $(m + n_1)$-dimensional function approximation or system modeling problems, in which n_1 is the number of variables that take a large number of possible values (i.e., N_j is large) or even continuous variable.

IV. Approximation Capabilities of Hierarchical Fuzzy Systems

This section analyzes the capabilities of hierarchical fuzzy systems to approximate functions on discrete spaces. The first part of the section gives the formulation of hierarchical fuzzy systems; the second part gives the approximation property of hierarchical fuzzy systems; and then, the third part provides some numerical examples.

A. Formulation of Hierarchical Fuzzy System

Let $y = F(X) = F(x_1, \ldots, x_n)$ be a two-level hierarchical fuzzy system from the discrete input space $U = \mathop{\times}\limits_{i=1}^{n} U_i \subset R^n$ given in (2) to the output space $V \in R$ with the hierarchical structure given in Fig. 2 by replacing g_1 and $g_{2,j} (j = 1, \ldots, m)$ there by f_1 and $f_{2,j} (j = 1, \ldots, m)$, respectively.

In this hierarchical fuzzy system, the *j*th fuzzy subsystem at the second (i.e., lower) level is

$$y_j = f_{2,j}\left(x_{i_1^{(2,j)}}, \ldots, x_{i_{n_{2,j}}^{(2,j)}}\right) = f_{2,j}\left(X_{2,j}\right) \tag{15}$$

whose input variables are $X_{2,j} = (x_{i_1^{(2,j)}}, \ldots, x_{i_{n_{2,j}}^{(2,j)}}) \in U_{G_{2,j}} = \mathop{\times}\limits_{k=1}^{n_{2,j}} U_{i_k^{(2,j)}}$ (where $x_{i_k^{(2,j)}}$ denotes the *k*th variable of fuzzy subsystem $f_{2,j}$) and the output variable is $y_j \in V_j \subset R$.

The fuzzy subsystem at the first (i.e., higher) level of this hierarchical fuzzy system is

$$y = f_1\left(y_1, \ldots, y_m, x_{i_1^{(1)}}, \ldots, x_{i_{n_1}^{(1)}}\right) = f_1\left(Y, X_1\right) \tag{16}$$

which has two sets of input variables: the first set is $Y = (y_1, \ldots, y_m) \in V = \mathop{\times}\limits_{j=1}^{m} V_j$ in which each y_j is the output variable of the *j*th fuzzy subsystem $f_{2,j}$ at the lower level; the second set is $X_1 = (x_{i_1^{(1)}}, \ldots, x_{i_{n_1}^{(1)}}) \in U_{G_1} = \mathop{\times}\limits_{k=1}^{n_1} U_{i_k^{(1)}}$.

For the previous hierarchical fuzzy system, the rule base of fuzzy subsystem $f_{2,j} (j = 1, \ldots, m)$ at the second level is

$$R_{l_1 \ldots l_{n_{2,j}}}^{(2,j)}: \text{IF } x_{i_1^{(2,j)}} \text{ is } A_{i_1^{(2,j)}, l_1} \text{ and} \cdots \text{and } x_{i_{n_{2,j}}^{(2,j)}} \text{ is } A_{i_{n_{2,j}}^{(2,j)}, l_{n_{2,j}}},$$

$$\text{THEN } y_j \text{ is } C_{l_{1_2} \ldots l_{n_{2,j}}}^{(2,j)} \quad l_1 \cdots l_{n_{2,j}} \in I_{2,j} \tag{17}$$

where the index set of the rule base is defined by

$$I_{2,j} = \{l_1 \cdots l_{n_{2,j}} \,|\, l_k = 1, \ldots, N_{i_k^{(2,j)}}; k = 1, \ldots, n_{2,j}\} \tag{18}$$

and the mathematical formula of $f_{2,j} (j = 1, \ldots, m)$ can be written in the following form:

$$\begin{aligned} y_j &= f_{2,j}(X_{2,j}) \\ &= \sum_{l_1 \cdots l_{n_{2,j}} \in I_{2,j}} \frac{A^{(2,j)}_{l_1 \cdots l_{n_{2j}}}(X_{2,j})}{\sum\limits_{l_1 \cdots l_{n_{2,j}} \in I_{2,j}} A^{(2,j)}_{l_1 \cdots l_{n_{2,j}}}(X_{2,j})} y^{(2,j)}_{l_1 \cdots l_{n_{2,j}}} \end{aligned} \tag{19}$$

where $y^{(2,j)}_{l_1 \cdots l_{n_{2,j}}}$ is the centroid of the output fuzzy set $C^{(2,j)}_{l_1 \cdots l_{n_{2,j}}}$ and $A^{(2,j)}_{l_1 \cdots l_{n_{2,j}}}(X_{2,j}) = \prod_{k=1}^{n_{2,j}} A_{i_k^{(2,j)}, l_k}(x_{i_k^{(2,j)}})$.

Further, the rule base of fuzzy subsystem f_1 at the first level is

$R^{(1)}_{i_1 \cdots i_m l_1 \cdots l_{n_1}}$: IF y_1 is B_{1,i_1} and $\cdots$ and y_m is B_{m,i_m} , $x_{i_1^{(1)}}$ is $A_{i_1^{(1)}, l_1}$ and $\cdots$ and $x_{i_{n_1}^{(1)}}$ is $A_{i_{n_1}^{(1)}, l_{n_1}}$,

THEN y is

$$C^{(1)}_{i_1 \cdots i_m l_1 \cdots l_{n_1}} \; i_1 \ldots i_m \, l_1 \ldots l_{n_1} \in I_1 \tag{20}$$

where the index set of the rule base is defined by

$$\begin{aligned} I_1 = \{&i_1 \cdots i_m l_1 \cdots l_{n_1} | i_j = 1, \ldots, N_{2j}, \\ &j = 1, \ldots, m, l_k = 1, \ldots, N_{i_k^{(1)}}, k = 1, \ldots, n_1\} \end{aligned} \tag{21}$$

and the mathematical formula of f_1 can be written in the following form:

$$\begin{aligned} y &= f_1(Y, X_1) \\ &= \sum_{i_1 \cdots i_m l_1 \cdots l_{n_1} \in I_1} \frac{A_{i_1 \cdots i_m l_1 \cdots l_{n_1}}(Y, X_1)}{\sum\limits_{i_1 \cdots i_m l_1 \cdots l_{n_1} \in I_1} A_{i_1 \cdots i_m l_1 \cdots l_{n_1}}(Y, X_1)} y^{(1)}_{i_1 \cdots i_m l_1 \cdots l_{n_1}} \end{aligned} \tag{22}$$

where $y^{(1)}_{i_1 \cdots i_m l_1 \cdots l_{n_1}}$ is the centroid of the output fuzzy set $C^{(1)}_{i_1 \ldots i_m l_1 \ldots l_{n_1}}$, and

$$A_{i_1 \cdots i_m l_1 \cdots l_{n_1}}(Y, X_1) = \prod_{r=1}^{m} B_{r,l_r}(y_r) \times \prod_{k=1}^{n_1} A_{i_k^{(1)}, l_k}(x_{i_k^{(1)}}).$$

Finally, the mathematical formula of the hierarchical fuzzy system is

$$\begin{aligned} y &= F(X) = f_1(Y, X_1) \\ &= f_1[f_{2,1}(X_{2,1}), \ldots, f_{2,m}(X_{2,m}), X_1]^{\cdot} \end{aligned} \tag{23}$$

B. Capabilities of Hierarchical Fuzzy Systems to Approximate Functions on Discrete Spaces

The main result of this section is the *flexible universal approximation property* of hierarchical fuzzy systems to approximate functions on discrete spaces.

Theorem 3 (Flexible Universal Approximation Property): Let $y = G(X)$ given in (1) be a function on discrete input space $U = \times_{i=1}^{n} U_i$ given in (2) and $G_1, G_{2,1}, \ldots, G_{2,m}$ be any desired disjoint grouping of the input variables $\{x_1, \ldots, x_n\}$ satisfying (8)–(11). Then, for any given $\varepsilon > 0$, there exists a hierarchical fuzzy system $F(X) = f_1[f_{2,1}(X_{2,1}), \ldots, f_{2,m}(X_{2,m}), X_1]$ given in (15)–(23) such that

$$\|G - F\|_\infty = \max_{X \in U} |G(X) - F(X)| < \varepsilon.$$

That is, for any given degree of accuracy, there exists a hierarchical fuzzy system $F(X)$ with the desired hierarchical structure $G_1, G_{2,1}, \ldots, G_{2,m}$ to approximate $G(X)$ within the required degree of accuracy.

Theorem 3 shows that hierarchical fuzzy systems are generally applicable for the approximation of functions on discrete spaces as they can approximate any function to any degree of accuracy. Further, when compared to hierarchical fuzzy systems for function approximation on continuous spaces (see [10], [13], [28], [33], and [39]), a distinguishing and important feature for function approximation on discrete spaces is that *two-level* hierarchical fuzzy approximators are capable of approximating any function to any degree of accuracy with *any desired separable hierarchical structure*. That is, for a given function or system on discrete spaces, a hierarchical fuzzy system can be designed in any desired hierarchical view to enable system understanding while achieving the required approximation accuracy. With LMCP modeling for site selection, this means that a hierarchical fuzzy system can be built to approximate the location–market condition and performance relationship from any desired view. For example, for site selection of new stores, we may wish to understand the impact of the grouped location–market factors such as location factors, demand factors, store facility factors, and competition factors on store performance. This can be done, first, by grouping all the location–market factors into the corresponding four groups; second, by building four lower level fuzzy subsystems with each representing and aggregating the market factors of each group (e.g., the location factors such as accessibility and visibility are aggregated to be the location index; demand factors such as local population, income level, and consumption level are aggregated to be demand index, etc,); then, a higher level fuzzy subsystem is used to represent the impact of the grouped and aggregated market factors (via the indexes) to store performance. In this way, the top-level fuzzy subsystem is able to show how store performance is dependent on its location condition, demand level, overall store facility, and local competition strength, i.e., in whichever way we have chosen to understand how store performance is related to its location–market conditions. In contrast, if the purpose of modeling is for store refurbishment and improvement, then the market factors can be grouped in a different way such as store attractiveness factors (such as store outlook, entrance width, entrance hall size, floor condition, store decoration condition), store size, store service facility factors (such as opening hours, cash machine availability, helpdesk availability, car parking space, toilet availability), and other market factors (location, demand, and competition), and then, building the corresponding hierarchical fuzzy system to model the market condition and performance relationship. In short, Theorem 3 shows the simplicity, generality, and flexibility of hierarchical fuzzy systems when being applied to approximate functions or systems on discrete spaces.

C. Numerical Examples

In this section, several numerical examples are given to illustrate the feasibility and effectiveness of the hierarchical approximation scheme discussed earlier. These examples compare the approximation power of standard fuzzy systems and hierarchical fuzzy systems presented in Theorem 3 and show what advantages hierarchical fuzzy systems bring to function approximation on discrete spaces. All these examples are based on the approximation of 3-D functions $G(x_1, x_2, x_3)$ that have three different disjointed groups: 1) $G_1 = \{x_3\}$, $G_{2,1} = \{x_1, x_2\}$; 2) $G_1 = \{x_2\}$, $G_{2,1} = \{x_1, x_3\}$; and 3) $G_1 = \{x_1\}$, $G_{2,1} = \{x_2, x_3\}$.

Example 2: Consider the functions given as follows:

1) function *A* given in [12] (see also Example 1):

$$y = G_A(x_1, x_2, x_3) = (1 + x_1^{0.5} + x_2^{-1} + x_3^{-1.5})^2;$$

2) function *B* given in [33]:

$$y = G_B(x_1, x_2, x_3) = \frac{x_1 x_2 x_3 (x_1 + 2.5)}{1 + x_1^2 + x_2^2 + x_3^3};$$

3) function *C* given in [33]:

$$y = G_C(x_1, x_2, x_3) = \frac{1}{1 + \sum_{i=1}^{3} \sin^2[0.25\pi(x_i - 1)]}$$

in which each input variable $x_i (i = 1, 2, 3)$ takes its values on the discrete space $U_i = \{1, 2, 3, 4, 5\}$. Then, each function given earlier is defined by 125 input–output data as follows:

$$\{[X(t), y(t)] \mid X(t) = [x_1(t), x_2(t), x_3(t)], y(t) = G[X(t)], t = 1, 2, \ldots, T\}$$

with $T = 125$. In order to give a fair comparison to avoid the different algorithms leading to different membership functions and destroying interpretability, it is assumed that the membership functions for each variable are equally spaced on $U_i = \{1, 2, 3, 4, 5\}$.

First, the standard fuzzy system

$$F(X) = F(x_1, x_2, x_3) = \sum_{i_1=1}^{N_1} \sum_{i_2=1}^{N_2} \sum_{i_3=1}^{N_3} \frac{A_{i_1}(x_1) A_{i_2}(x_2) A_{i_3}(x_3)}{\sum_{i_1=1}^{N_1} \sum_{i_2=1}^{N_2} \sum_{i_3=1}^{N_3} A_{i_1}(x_1) A_{i_2}(x_2) A_{i_3}(x_3)} y_{i_1 i_2 i_3}$$

is used to approximate the previous functions with the approximation accuracy that is measured by the root mean-squared error (RMSE) given as follows:

$$\mathrm{RMSE}(G, F) = \frac{1}{T} \sqrt{\sum_{t=1}^{T} \{G[X(t)] - F[X(t)]\}^2} = \frac{1}{T} \sqrt{\sum_{t=1}^{T} \{y(t) - F[X(t)]\}^2}. \tag{24}$$

TABLE I
FUNCTION APPROXIMATED BY STANDARD FUZZY SYSTEMS: APPROXIMATION ERRORS, RULE, AND PARAMETER NUMBERS

Function	Approximation accuracy (i.e., RMSE)	Total number of rules and parameters
Function A	0.8627	27
Function B	0.1845	27
Function C	0.0326	27

Let each input space U_i $(i = 1, 2, 3)$ be partitioned by three equally spaced triangular membership functions (i.e., $N_i = 3$). Then, for each function $G(x_1, x_2, x_3)$ given earlier, by applying the standard least square method to identify the parameters $\{y_{i_1 i_2 i_3} | i_j = 1, 2, 3; j = 1, 2, 3\}$ (there are 27 parameters in total), the best standard fuzzy approximator in the sense of minimizing the RMSE given in (24) can be obtained, and Table I gives the corresponding achieved RMSEs.

Now hierarchical fuzzy systems given in Theorem 3 are used to approximate the given functions with the following questions. First, to achieve the same or better approximation accuracy as given in Table I, how many rules and parameters are needed for the hierarchical approximation scheme? The purpose for this question is to understand what benefit the hierarchical fuzzy approximation can bring. Second, for each possible disjointed group of the input variables: 1) $G_1 = \{x_3\}$, $G_{2,1} = \{x_1, x_2\}$; 2) $G_1 = \{x_2\}$, $G_{2,1} = \{x_1, x_3\}$; and 3) $G_1 = \{x_1\}$, $G_{2,1} = \{x_2, x_3\}$, does there always exist a hierarchical fuzzy system that has the given grouping as its hierarchical structure and can achieve the same or better accuracy than those given in Table I? The purpose for this is to give some numerical illustrations of Theorem 3.

For building hierarchical fuzzy systems, let each input space U_i $(i = 1, 2, 3)$ and the intermediate variable space V_1 be partitioned by equally spaced triangular membership functions. Then, for each function $G(x_1, x_2, x_3)$ given earlier, by applying the gradient descent algorithm (a modified version from [33]) to identify the parameters, the best hierarchical fuzzy approximator in the sense of minimizing the RMSE can be obtained, and Table II gives the corresponding achieved RMSEs and the total numbers of rules and parameters.

In Table II, the hierarchical fuzzy approximator as 1) $F(X) = f_1[f_{2,1}(x_1, x_2), x_3]$ or 2) $F(X) = f_1[f_{2,1}(x_1, x_3), x_2]$ or 3) $F(X) = f_1[f_{2,1}(x_2, x_3), x_1]$ is corresponding to the hierarchical structure as: a) $G_1 = \{x_3\}$, $G_{2,1} = \{x_1, x_2\}$; b) $G_1 = \{x_2\}$, $G_{2,1} = \{x_1, x_3\}$; or c) $G_1 = \{x_1\}$, $G_{2,1} = \{x_2, x_3\}$, respectively.

Comparing the approximation results given in Tables I and II for the three given functions as well as several other examples (omitted here due to space limitations), the following observations can be made. First, the hierarchical fuzzy approximation performs better in these examples than the standard fuzzy approximation in the sense that the former can achieve the same or better approximation accuracy with fewer rules and parameters (in other words, can achieve better accuracy with the same number of rules and parameters). Second, these examples illustrate and evidence the conclusion of Theorem 3 that any function on

TABLE II
FUNCTIONS APPROXIMATED BY HIERARCHICAL FUZZY SYSTEMS: APPROXIMATION ERRORS, RULE, AND PARAMETER NUMBERS

Function	Hierarchical fuzzy approximator	Number of rules and parameters of $f_{2,1}$	Number of rules and parameters of f_1	Total Number of rules and parameters	Approximation Accuracy (i.e., RMSE)
Function A	$F(X)=f_1[f_{2,1}(x_1,x_2),x_3]$	3x3=9	3x3=9	9+9=18	0.5240
	$F(X)=f_1[f_{2,1}(x_1,x_3),x_2]$	3x3=9	3x3=9	9+9=18	0.7019
	$F(X)=f_1[f_{2,1}(x_2,x_3),x_1]$	3x3=9	3x3=9	9+9=18	0.7112
Function B	$F(X)=f_1[f_{2,1}(x_1,x_2),x_3]$	4x4=16	3x3=9	16+9=25	01792
	$F(X)=f_1[f_{2,1}(x_1,x_3),x_2]$	4x4=16	3x3=9	16+9=25	0.1792
	$F(X)=f_1[f_{2,1}(x_2,x_3),x_1]$	4x4=16	3x3=9	16+9=25	0.1755
Function C	$F(X)=f_1[f_{2,1}(x_1,x_2),x_3]$	3x3=9	4x3=12	9+12=21	0.0147
	$F(X)=f_1[f_{2,1}(x_1,x_3),x_2]$	3x3=9	4x3=12	9+12=21	0.0147
	$F(X)=f_1[f_{2,1}(x_2,x_3),x_1]$	3x3=9	4x3=12	9+12=21	0.0147

discrete space can be approximated by two-level hierarchical fuzzy systems with any desired disjoint hierarchical structure. Third, the performance of hierarchical fuzzy approximators is dependent on the function to be approximated. The hierarchical fuzzy systems are especially suitable to approximate functions with the natural or explicit hierarchical decomposition such as function *A* (as shown by Example 1) and function *C* (it is called natural or explicit hierarchical structure here in contrast to the hierarchical decomposition given in Theorem 1 that is implicitly defined). For example, for functions *A* and *C*, hierarchical fuzzy systems can achieve better accuracy with much fewer rules and parameters than the standard fuzzy systems (18 or 21 rules and parameters comparing with 27 rules and parameters) as shown in Table II.

V. AN IDENTIFICATION METHOD FOR HIGH-DIMENSIONAL FUNCTIONS WITH LIMITED TRAINING DATA

This section is to develop a hierarchical fuzzy identification method for the case where a high-dimensional system needs to be identified or constructed but only limited numerical data about the system are available. The first part of this section proposes the identification method and the second part analyzes the basic design principles and guidances when applying the proposed method in applications. The proposed method and design principles will be used to solve an LMCP modeling problem in the next section.

A. Identification Method

When a system to be approximated is a high dimensional one but with only limited input output data, the proposed method to find a hierarchical fuzzy approximator is to combine human knowledge and available input/output data by the following steps.

1) *Construct the lower level fuzzy subsystems based on human knowledge.* First, divide all input variables $\{x_1,\ldots,x_n\}$ into several desired disjoint groups $G_1, G_{2,1},\ldots,G_{2,m}$, and then, based on human knowledge, construct the lower level fuzzy subsystems $f_{2,1}(X_{2,1}),\ldots,f_{2,m}(X_{2,m})$ to aggregate the impact of input variable groups $G_{2,1},\ldots,G_{2,m}$ on system output. Taking LMCP modeling as an example, this step divides all location–market factors into several groups such as location factors, demand factors, store facility factors, and competition factors, and then, for each factor group, expert knowledge is used to form the aggregation fuzzy rules. For example, consider a simplified location factor group; then an aggregation rule from domain experts might be:
If the visibility is good and the accessibility is average, then the location is above average.
Once all such rules are generated from experts, then the membership functions for each linguistic term (good, average, poor) can be constructed by assigning numerical parameters to these fuzzy terms, and the aggregated lower level fuzzy subsystem for this group can be formed based on (19). In this way, each lower level fuzzy subsystem aggregates each input variable group into a corresponding index such as the location (condition) index, demand index, store facility index, and competition index.
2) *Transform the available input–output data into the aggregated input (index)–output data.* If the available input–output data are $\{[X(t),y(t)]\,|\,t=1,\ldots,T\}$, then this step transforms the lower level fuzzy subsystems constructed in the previous step into the aggregated input–output data $\{([Y(t),X_1(t)],y(t))\,|t=1,\ldots,T\}$, where $Y(t)=[y_1(t),\ldots,y_m(t)]$ and $y_j(t)=f_{2,j}[X_{2,j}(t)]$ $(t=1,\ldots,T; j=1,\ldots,m)$. As each $y_j (j=1,\ldots,m)$ aggregates the impact of several input variables (i.e., the input variable group G_j), the number of the aggregated input variables (Y,X_1) can be much lower than the number of the original input variables. Taking LMCP modeling as an example, this step transforms the location–market factors and performance data $\{[X(t),y(t)]|t=1,\ldots,T\}$ into the location–market indexes and performance data

$\{[Y(t), y(t)] | t = 1, \ldots, T\}$ (as all location–market factors are divided into the input variable groups of the lower fuzzy subsystems; therefore, $G_1 = \phi$ and this is why there is no $X_1(t)$ here). In this way, 20–40 location–market factors and performance data pairs often can be transformed into four to six aggregated location–market indexes and performance data pairs.

3) *Identify the higher level fuzzy subsystem $f_1(Y, X_1)$ by learning from aggregated input–output data.* That is, $f_1(Y, X_1)$ is identified as the fuzzy system which best fits the available aggregated input (i.e., index)–output data $\{([Y(t), X_1(t)], y(t)) \mid t = 1, \ldots, T\}$.
4) *Refining the earlier identified hierarchical fuzzy system by gradient descent algorithm.* That is, treat the hierarchical fuzzy system identified from steps 1–3 as an initial hierarchical fuzzy system and then apply the gradient descent algorithm to refine the system parameters to improve the approximation accuracy.

The key idea behind the previous method is that, by constructing the lower level fuzzy subsystems based on human knowledge, the identification problem of a high-dimensional hierarchical fuzzy system $F(X) = f_1[f_{2,1}(X_{2,1}), \ldots, f_{2,m}(X_{2,m}), X_1]$ to approximate a given system or function is transformed into the identification problem of a lower dimensional standard fuzzy system $f_1(Y, X_1)$. It is the lower dimensionality of $f_1(Y, X_1)$ that allows it to be identified by using the limited available aggregated input–output data $\{([Y(t), X_1(t)], y(t)) \mid t = 1, 2, \ldots, T\}$ (notice here that the number of the aggregated input–output data is the same as that of the original input–output data but the fuzzy system (i.e., the higher level fuzzy subsystem) needed to be identified is a much lower dimensional one). Taking LMCP modeling as an example, the original LMCP function or model has about 20 to 40 input variables. By using experts' knowledge to aggregate the impact of each factor group into a corresponding index, the identification problem of the hierarchical fuzzy system to approximate the LMCP model is transformed into the identification problem of a standard fuzzy system $f_1(Y)$ (as $G_1 = \phi$ here) with four to six input variables.

It should be pointed out that the previous method is still applicable when the available human knowledge is not very accurate or not complete, as step 4 will further train or refine the system by using the gradient algorithm.

B. Methodology Analysis and Design Principles

In this section, the previous method is further analyzed and the design principles for using the proposed method are given.

As the lower level fuzzy subsystems are constructed based on human knowledge, then, to validate the proposed method, the first question requiring an answer is what conditions need to be met by these lower level fuzzy subsystems in order to maintain the capability of being able to approximate a given function or system to any degree of accuracy.

The answer to this question is that if the lower level fuzzy subsystems $y_j = f_{2,j}(X_{2,j}) (j = 1, \ldots, m)$ given in (19), constructed based on human knowledge from one-to-one mappings (based on Lemma 2, such one-to-one fuzzy systems exist), then there exists a higher level fuzzy subsystem $f_1(Y, X_1)$ such that the resulting hierarchical fuzzy system can approximate the given function or system to any degree of accuracy (see Theorem A.1 in the Appendix for the formal description of this property). In fact, when the lower level fuzzy subsystems are one-to-one mappings, it can be proved (similar to the proof of Lemma 1) that there exists a function $g_1(Y, X_1)$ such that the original function $G(X)$ can be represented as

$$\begin{aligned} G(X) &= g_1(Y, X_1) = g_1(y_1, y_2, \ldots, y_m, X_1) \\ &= g_1[f_{2,1}(X_{2,1}), \ldots, f_{2,m}(X_{2,m}), X_1]. \end{aligned} \tag{25}$$

That is, one-to-one mapping property of each lower level fuzzy subsystem ensures *no information loss* of the original function when aggregating or encoding the multidimensional input information into 1-D output. In this way, the approximation of a high-dimensional function $G(X)$ is transformed into the approximation of a lower dimensional function $g_1(Y, X_1)$. Step 3 of the proposed method is, in fact, to find a higher level fuzzy subsystem $f_1(Y, X_1)$ to approximate $g_1(Y, X_1)$.

Taking LMCP modeling as an example, the one-to-one mapping means that each lower level aggregation fuzzy subsystem based on human knowledge is able to distinguish the different location or market conditions. For example, if the accessibility and visibility of two stores are different, then the resulting location index value (i.e., the outputs of the location aggregation subsystem) for the two stores should be different. Even from an intuitive point of view, such a one-to-one mapping requirement for the construction of the lower-level fuzzy subsystems is quite natural because, if the lower level fuzzy subsystems cannot distinguish the different location or market conditions, it is unlikely that a hierarchical fuzzy system with these lower level fuzzy subsystems is able to distinguish the different performance of stores in different location and market conditions.

Although it is not difficult to find a one-to-one mapping in the mathematical sense, it is often infeasible in practice to convert human knowledge to one-to-one mappings if the number of possible values for input variables is high. For example, if each input variable group has five variables and each variable has ten possible values, then such conversion of human knowledge to a one-to-one mapping requires $10^5 = 100\,000$ different values being assigned. As this is impractical, so the more feasible solution is to construct the lower level fuzzy subsystems that are a one-to-one mapping only for the typical input cases. With LMCP modeling, if the lower level location index fuzzy subsystem is to be constructed based on the visibility and accessibility and the typical cases for the visibility and accessibility are *good*, *average*, and *poor*, then the rule base is designed as: *if visibility is good and accessibility is good, then the location is good; if visibility is good and accessibility is average, then the location is above average*; .... There are nine rules in this case and the mathematical formula of the location index fuzzy subsystem is

$$y_j = f_{2,j}(X_{2,j}) = f_{2,j}\left(x_{i_1^{(2,j)}}, x_{i_2^{(2,j)}}\right) = \sum_{l_2=1}^{3}\sum_{l_1=1}^{3} \frac{A_{l_1 l_2}^{(2,j)}(X_{2,j})}{\sum_{l_1 l_2 \in I_{2,j}} A_{l_1 l_2}^{(2,j)}(X_{2,j})} y_{l_1 l_2}^{(2,j)}$$

with $X_{2,j} = (x_{i_1^{(2,j)}}, x_{i_2^{(2,j)}})$ being the visibility and accessibility input variables. Then, the requirement that the location index fuzzy subsystem is a one-to-one mapping in the typical cases means that $y_{l_1 l_2}^{(2,j)} (l_1 = 1, 2, 3;\ l_2 = 1, 2, 3)$ are different. That is, the location index score at nine typical location conditions—(*good, good*), (*good, average*), (*good, poor*), $\ldots$, (*poor, poor*) are different. In this way, the lower level fuzzy subsystem is relatively easy to construct based on available human knowledge. At the same time, the location index subsystem maintains the one-to-one mapping requirement to a reasonable degree.

Based the previous analysis, the first design principle or guideline can be summarized as the following.

1) *Design lower level fuzzy subsystem to minimize the information loss:* Each lower level fuzzy subsystem should be designed as much as possible to be a one-to-one mapping to minimize the information loss when aggregating the multidimensional inputs into the output index of the given lower level subsystem. In particular, each lower level fuzzy subsystem should be able to distinguish the significant different or typical input values by assigning different output values.

Although any one-to-one lower level fuzzy subsystems are enough to preserve the approximation capability of hierarchical fuzzy systems, different construction or design of such one-to-one lower level fuzzy subsystems will lead to the different approximation complexity for the higher level fuzzy subsystem and different forecasting accuracy for the resulting hierarchical fuzzy system. For this reason, it is required to find an effective way to construct lower level fuzzy subsystems to reduce the complexity for the higher level fuzzy subsystem and improve the forecasting accuracy. For this purpose, the measurement of the complexity for functions on discrete spaces needs to be introduced. In continuous function approximation, usually the smoothness (i.e., differentiability) is used to measure the complexity of a function. However, there are two main difficulties in applying the smoothness concept here. First, such information is not available as only limited data related to the given function or system are available. Second, the smoothness or differentiability concept is not applicable to functions on discrete spaces. Therefore, a different measurement is needed. In this paper, the monotone property is proposed to measure the complexity of a function. For this purpose, the concept of monotone functions is introduced first.

Definition 3: Let $G(X)$ be a function on a discrete (or continuous) input space $U = \overset{n}{\underset{i=1}{\times}} U_i$. If for any $X, X' \in U$ and $X = (x_1, \ldots, x_n) \leq X' = (x'_1, \ldots, x'_n)$ (i.e., $x_i \leq x'_i$ for all $i = 1, \ldots, n$), $G(X) \leq G(X')$, then $G(X)$ is known as a monotone increase function. If $G(X) \geq G(X')$, then $G(X)$ is known as a monotone decrease function.

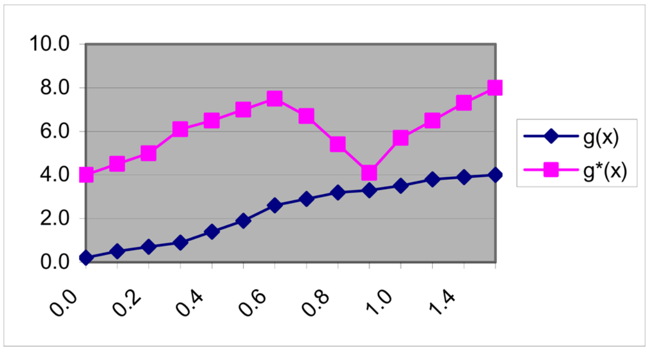


Fig. 3. Functions with different complexity: $g^*(x)$ is more complicated than $g(x)$.

It can be proved (see Proposition B in the Appendix) that a multidimensional function is monotone if and only if it is a monotone function related to each of its variables. Based on this fact, the following discussion will focus on 1-D functions to illustrate why the complexity of a function can be measured by its monotone property. Let $g(x)$ be a 1-D function on a discrete input space U, then $g(x)$ can be represented as a K piecewise monotone function as it is monotone between each pair of local min and max points, where K is a natural number. A function is more complicated if its corresponding K is bigger. The reason for such a measure of the complexity is illustrated by Fig. 3 where function $g^*(x)$ is more complicated than $g(x)$, as $g^*(x)$ needs three piecewise monotone functions to represent it, whereas $g(x)$ needs only one. Based on this definition, a monotone function is the simplest as $K = 1$. Another advantage for monotone functions is that the monotone property enables more accurate and reliable forecasting. The reason is that, for a monotone function, the value of the function at a given point is bounded by the function values at its neighbors. As a result, a reasonable estimation for a given point can be obtained if the function values at its neighbors are known.

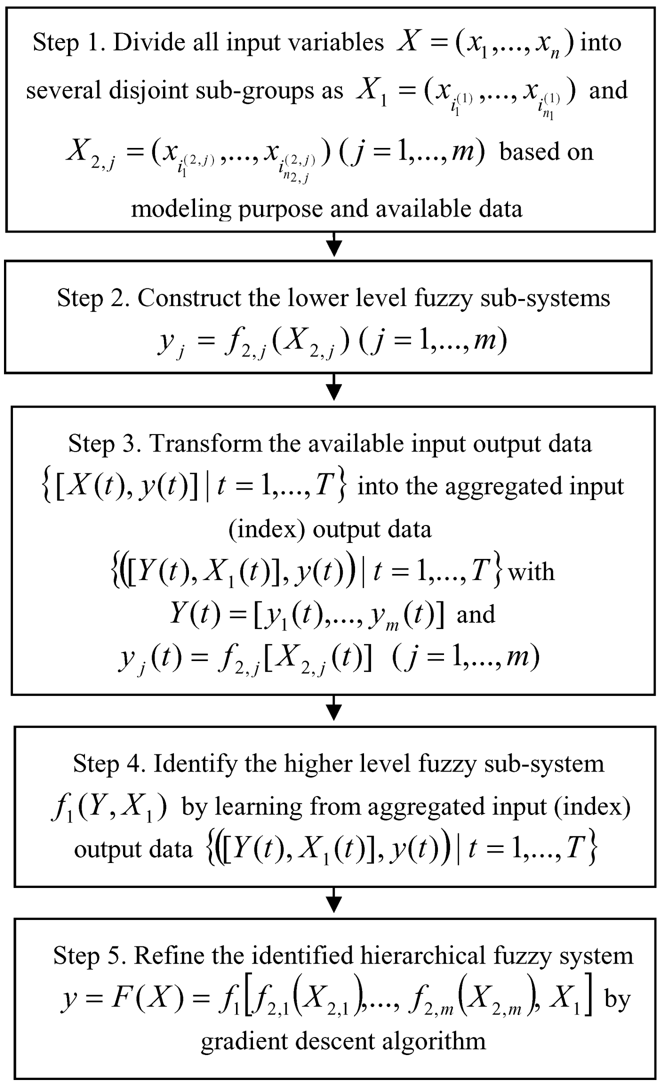
The previous analysis shows that, in order to reduce approximation complexity for the higher level fuzzy subsystem and improve forecasting accuracy, an effective approach is to construct the lower level fuzzy subsystems such that the resulting higher level function or system $g_1(Y, X_1)$ given in (25) is a monotone function related to Y or as monotone as possible. It can be proved (see Theorem A.2 in the Appendix) that such an approach is applicable under very mild conditions (i.e., applicable for a wide range of applications). In the following, we use the LMCP modeling problem to illustrate how such an approach can be achieved.

Consider the LMCP modeling problem under the grouping of all market factors into four groups as the location factors (such as visibility and accessibility), demand factors, store facility factors, and competition factors. When designing the lower level fuzzy subsystem for any given market factor group such as the location factors, we assume that all other market factors (i.e., demand factors, store facility factors, and competition factors) are the same, and then, ask the domain experts to give a higher location score or index value when a location condition is better for the sales performance (such as a location condition with good visibility and accessibility is assigned with a higher location index than one with poor visibility and accessibility). With such a natural and intuitive scoring method, the resulting higher level subsystem to be approximated becomes a monotone function related to the location index (i.e., the outputs of the lower level location index fuzzy subsystems).

As illustrated in the previous example of an LMCP modeling problem, to achieve the monotone property of the higher level subsystem, the available human knowledge should be capable of determining correctly the order of outputs corresponding to all different inputs for each lower level fuzzy subsystem. Similar to the construction of one-to-one mappings, it is often unlikely in practice that available human knowledge is able to determine such order correctly for all possible input values if the number of possible values for input variables is high. Therefore, the more realistic approach is to utilize the domain experts' knowledge to determine the output order only for the typical input cases, i.e., the higher level system is monotone for those typical input cases. Further, if the identified higher level fuzzy subsystem is found to have lost the monotone property, rejustifying and reassigning the values of the lower level fuzzy system will regain such a monotone property of the higher level system for those typical input cases.

Based the previuos analysis, the second design principle or guideline can be summarized as follows.

1) *Design lower level fuzzy subsystems by minimizing the complexity of the higher level subsystem*: Each lower level fuzzy subsystem should be designed by scoring the index values of the different inputs based on their corresponding sale performance, such that the higher level subsystem to be approximated becomes as monotone as possible in relating to the indexes (i.e., the outputs of the lower level subsystems). In particular, if the identified higher level fuzzy subsystem is found to have lost the monotone property at the typical input cases, rejustifying and reassigning the values of the lower level fuzzy subsystem will regain such a monotone property of the higher level system for the typical input cases.


Step 1. Divide all input variables $X=(x_1,\ldots,x_n)$ into several disjoint sub-groups as $X_1=(x_{i_1^{(1)}},\ldots,x_{i_{n_1}^{(1)}})$ and $X_{2,j}=(x_{i_1^{(2,j)}},\ldots,x_{i_{n_{2,j}}^{(2,j)}})$ $(j=1,\ldots,m)$ based on modeling purpose and available data

Step 2. Construct the lower level fuzzy sub-systems $y_j=f_{2,j}(X_{2,j})$ $(j=1,\ldots,m)$

Step 3. Transform the available input output data $\{[X(t),y(t)]\mid t=1,\ldots,T\}$ into the aggregated input (index) output data $\{([Y(t),X_1(t)],y(t))\mid t=1,\ldots,T\}$ with $Y(t)=[y_1(t),\ldots,y_m(t)]$ and $y_j(t)=f_{2,j}[X_{2,j}(t)]$ $(j=1,\ldots,m)$

Step 4. Identify the higher level fuzzy sub-system $f_1(Y,X_1)$ by learning from aggregated input (index) output data $\{([Y(t),X_1(t)],y(t))\mid t=1,\ldots,T\}$

Step 5. Refine the identified hierarchical fuzzy system $y=F(X)=f_1\big[f_{2,1}(X_{2,1}),\ldots,f_{2,m}(X_{2,m}),X_1\big]$ by gradient descent algorithm

Fig. 4. Procedures and steps for the identification of hierarchical fuzzy systems.

Based on the discussion and analysis in the last and in this section, the general steps for applying the proposed approach of hierarchical fuzzy systems are given in Fig. 4, and the general guidelines can be summarized as follows.

1) In step 1 of the input variable grouping, the three basic guidelines are, first, to divide the input variables to match domain experts' thinking, as this will lead to the benefits that human knowledge will be easier and better utilized in modeling, and the resulting system will be more easily understood and used; second, each group should be kept with the similar number of variables to reduce the overall complexity, as this will allow the system complexity to be evenly distributed into each subsystem and avoid some subsystems being too complicated or too high dimensional when compared with others; third, if there are more data, then more groups are allowed, as more model behaviors can be learned from the available objective data; if less data, then fewer groups have to be enforced, otherwise there are not enough data to identify the higher level fuzzy subsystem.

2) In step 2 of constructing lower level fuzzy subsystems, the basic design principles or guidelines (as analyzed in details earlier) are minimizing the information loss and the complexity of the higher level subsystem when using all the available human knowledge. In the case where there is little or no human knowledge available, random initialization can be applied as these subsystems will be further identified and tuned in step 5. However, it should be pointed out that even incomplete and inaccurate human knowledge will be much better than randomly initialization, as such knowledge will provide better initial fuzzy subsystems with simpler structure and better interpretability to be refined in step 5. Therefore, another basic guideline is to use as much as possible any available human knowledge.

3) In step 3 of transforming the input–output data into the aggregated input (index)–output data, the key idea here is to transform an identification problem in higher dimensional input space into an identification problem in lower dimensional aggregated input space based on the constructed lower level fuzzy subsystems in step 2. Such a step will improve the identifiability, especially when there are only limited training data available, and greatly reduce the complexity of the system identification especially in a high-dimensional situation.
4) In step 4 of identifying the higher level fuzzy subsystem $f_1(Y, X_1)$, most available fuzzy system learning and identification methods are usable as this is a relatively lower dimensional identification problem. However, the recommended approaches are the least square methods [3], [32] and the clustering-gradient-based methods [1], [3] as these will lead to better transparency and interpretability of the resulting hierarchical fuzzy system.
5) In step 5 of refining the identified hierarchical fuzzy system by the gradient descent algorithm, the main motivation in having such a step is to make the proposed hierarchical fuzzy system approach usable in a wide range of applications with good, poor, or no human knowledge. However, it should be noticed that the more accurate the human knowledge, the simpler and more accurate the refined hierarchical fuzzy systems. The reason is that the better human knowledge provides better initial system with simpler structure and better interpretability and then has much more chances to avoid the gradient descent algorithm leading to local optimal parameters.

VI. Application—A Hierarchical Fuzzy System Approach to LMCP Modeling

In this section, the proposed hierarchical fuzzy system identification method, the design principles, and guidelines given in Section V are to be used as a new method for LMCP modeling, and the obtained result will be compared with the most important and widely used LMCP modeling approaches—linear regressions and neural networks. This real application is based on real commercial data from a retail chain with about 400 stores. Due to the commercial confidential constraint, we are unable to disclose the data resource and will mainly focus on the technical analysis and comparison in the following experiment.

In this experiment, the mathematical relationship of LMCP is represented by

$$S = G(x_1, \ldots, x_{31}) \tag{26}$$

where S is the sale performance (store annual revenue) and $x_1, \ldots, x_{31}$ represent 31 input variables that include all related location and market factors. The objective of the experiment is to apply the approach proposed earlier to identify a hierarchical fuzzy system $F(x_1, \ldots, x_{31})$ to approximate $G(x_1, \ldots, x_{31})$. For this purpose, 90% of the data is used for training and 10% of the data is used for testing. This represents the real site selection situation where there are about 5%–10% new locations being selected for new stores.

TABLE III
Comparison of the Training and Testing Errors of Hierarchical Fuzzy System, Linear Regression, and Neural Network

Model	RMSE-training	RMSE-testing	MAPE-training	MAPE-testing
Linear Regression	46700	32745	18.8%	16.3%
Neural Network	20417	20593	8.16%	8.87%
Hierarchical fuzzy system	16037	18384	6.76%	7.65%

First, following step 1 in Fig. 4, based on the business domain analysis, 31 variables were divided into six location or market factor groups and each group is with five or six location and market factors. Second, steps 2–4 in Fig. 4 are applied to obtain the initial hierarchical fuzzy system in which the RMSE is used as the accuracy measure and the least square method is used to identify the system parameters. Finally, step 5 in Fig. 4 is applied to refine the obtained hierarchical fuzzy system by using the gradient descent algorithm to minimize the RMSE. After steps 1–5, the obtained final hierarchical fuzzy system is one with six lower level fuzzy subsystems and one higher level fuzzy subsystem as follows:

$$y = F(X) = f_1\left[f_{2,1}\left(X_{2,1}\right), \ldots, f_{2,6}\left(X_{2,6}\right)\right] \tag{27}$$

where $X_{2,j}$ represents the location or market factors of group j $(j = 1, \ldots, 6)$, $f_{2,j}(X_{2,j})$ is the *j*th fuzzy subsystem whose output y_j represents the aggregated index of the *j*th location or market factor group, and $f_1(y_1, \ldots, y_6)$ is the higher level fuzzy subsystem that represents the relationship between the aggregated location or market indexes and the sale performance. The training and testing RMSEs of the obtained hierarchical fuzzy system are given in Table III. As the RMSE is an absolute error measurement and less convenient for comparison, Table III also uses the commonly used relative error measurement—the mean absolute percentage error (MAPE)—as

$$\begin{aligned}\text{MAPE}(G, F) &= \frac{1}{T}\sum_{t=1}^{T}\left|\frac{G[X(t)] - F[X(t)]}{G[X(t)]}\right| \\ &= \frac{1}{T}\sum_{t=1}^{T}\left|\frac{y(t) - F[X(t)]}{y(t)}\right| \end{aligned}\tag{28}$$

to show training and testing errors of the obtained hierarchical fuzzy system. At the same time, for the convenience of the comparison, the training and testing RMSEs and MAPEs of the identified linear regression and neural network models based on the same data are also shown in Table III.

From Table III, we have the following observations.

1) The hierarchical fuzzy system and neural network achieve much better accuracy than the linear regression as the training and testing RMSEs and MAPEs of the formers are less than half of the latter. The reason behind this is that linear regression models are not complicated enough to catch up the nonlinear behavior of LMCP models. This is evidenced by the fact in Table III that the RMSE of

training of linear regression is much larger than the RMSE of testing (basically, the regression model is underfitted).

2) The hierarchical fuzzy system performs relatively significantly better than the neural network as the testing MAPE is 7.65% for the former and 8.87% for the latter. In addition to the better performance in the approximation accuracy, another important advantage of hierarchical fuzzy systems in comparing with neural networks is that the identified hierarchical fuzzy system can be represented as a knowledge rule base that can be understood and validated by users. This greatly increases the users' confidence in the obtained hierarchical fuzzy system. Also, such a knowledge rule base provides very useful insight about what kind of store sites are good and then gives the guidance for searching good store sites.

Although Table III shows that the hierarchical fuzzy system obtained by the proposed approach outperforms neural networks in LMCP modeling, a natural question is what the reason behind this is (i.e., whether such a better performance is coincident or there is some genuine reason for this).

The first and more obvious reason for the better performance of hierarchical fuzzy systems is that a better initial system model is used. Hierarchical fuzzy approach incorporates human knowledge into the model process, as stated in Section V. Therefore, it provides a better initial system model. Such a better initial model allows the training less chance to trap into local optimal parameters and greatly simplifies the model building effort. For neural networks, as the black box feature, human knowledge is much more difficult to integrate into the model initializing process, and therefore, training has much more of a chance to trap into local optimal parameters. As a result, neural networks have more of a chance to lead to a system model with poorer performance. As most LMCP modeling is carried out by business analysts who may not have the in-depth knowledge and experience in training neural networks, such an advantage of hierarchical fuzzy systems is very useful from an application point of view. However, from a theoretical point of view, it may be arguable that such an advantage is less significant, because the use of neural networks as a mathematical modeling tool or approach is not a weakness but a requirement for better training to users.

The second and much more fundamental reason (from theoretical and methodological points of view) for the better performance is the different approximation mechanisms between neural networks and hierarchical fuzzy systems. In the following, this point is analyzed in detail.

Mathematically, a forward multilayer neural network can be represented as

$$y = NN(X) = \sum_{i=1}^{N} w_i \sigma\left(a_i^\tau X + b_i\right) + c_0$$

and its approximation mechanism can be briefly summarized as follows. First, a number of different linear functions (i.e., $a_i^\tau X + b_i$) provide an irregular partition of the input spaces (see Fig. 5). Second, a number of different weighting factors (i.e., w_i) associating the active function [i.e., $\sigma(z)$] are assigned to different linear functions resulting in different active degrees of different linear functions (see [22] for more detailed analysis of this partition point of view on the approximation mechanism of neural networks). When approximating high-dimensional functions with limited training data, this mechanism with minimizing error training leads to a neural network with finer partitions in the regions with many training data (for example, in the middle region of Fig. 5) and sensible weighting factors, while more or less ignoring its behaviors in other regions (for example, those regions near the boundary in Fig. 5). As a result, a neural network tends to give reasonable forecasting when a new input is within the regions with many training data and poor or wild forecasting when a new input is isolated. Therefore, neural networks have much more of a chance to make poor or wild forecasting in the case of high-dimensional functions with limited training data as there are many regions with few or no training data.

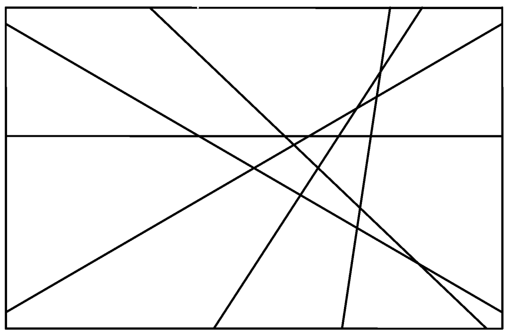


Fig. 5. Irregular partition used by neural networks.

On the other hand, hierarchical fuzzy systems have a very different approximation mechanism. First, the impacts of the several input variable groups are aggregated into corresponding index variables by the lower level fuzzy subsystems. Second, the aggregated indexes are used as the input variables to the higher level fuzzy subsystems. For approximating high-dimensional functions with limited training data, the first aggregation step maps the original input data sparsely distributed in the high-dimensional input spaces into very compact data distribution in the much lower dimensional index variable spaces. The point can be illustrated by a simplified example: suppose the training data includes the location data related to (*visibility, accessibility*) as (*good, good*), (*average, good*), (*average, average*), *and* (*poor, average*), shown in Fig. 6. With the location index fuzzy subsystem, these sparse data in the original input spaces could be converted into the location index score as *good*, *above average*, *average*, *below average* that is reasonably densely distributed in the location index space. This is in the case of conversion of data from 2-D data into one dimension. In the case of multiple dimension (≥ 3), such aggregation will lead to much denser distributed data in the index variable spaces. For example, in the considered experiment, about 360 location training data sparsely distributed in the 6-D (location factor) input space are aggregated into very densely distributed data in 1-D location index space. As a result of such an aggregation using the lower level subsystems, the forecasting based on the higher level fuzzy system is always bounded by the training data even when a new input is within a region with no or little training data

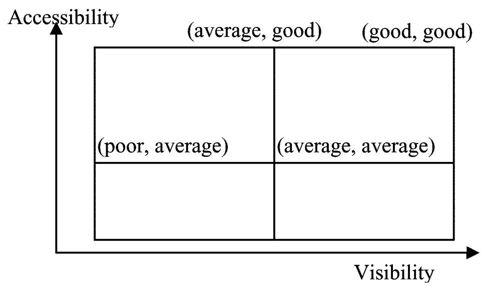


Fig. 6. Original input data sparsely distributed in input spaces.

(in fact, this is the benefit gained by applying the second design principle given in Section V). These behaviors can be explained by the following: suppose that an identified hierarchical fuzzy system has learned the sale performance from the training data at the location factors (*visibility, accessibility*) at (*good, good*), (*average, good*), (*average, average*), and (*poor, average*). In other words, the higher level fuzzy subsystem has learned the sale performance for the location index scores at *good*, *above average*, *average*, *below average*. Now suppose that we need to forecast the sale performance at a location as (good, average). As this new input is located between (*good, good*) and (*average, average*), the lower level location index fuzzy subsystem will map it into a location index score between *good* and *average*. As a result, the sale performance forecasting of this new input based on the higher level fuzzy subsystem will be between the sale performances of the location index scores at *good* and *average*. That is, the sale performance forecasting of the new input as (good, average) is bounded by the sale performances of two training data (*good, good*) and (*average, average*). Now suppose that the available training data include only (*good, good*), (*average, good*), and (*average, average*) and the sale performance of the new input data as (*poor, poor*) is to be forecasted. In this case, there is no any training data around the new input data (*poor, poor*). Based on the same analysis as earlier, it can be found that the sale forecasting of this new input is upper bounded by the sale performance at the training data (*average, average*) and such upper bound reduces the chance of unreasonable forecasting. These examples illustrate that even with the limited data distributed sparsely in the input spaces, the approximation mechanism of hierarchical fuzzy systems ensures the forecasting of new input data bounded by the training data (through the aggregation of the lower level fuzzy subsystems) and avoids unreasonable or wild forecasting as neural networks. This is a distinguished feature of hierarchical fuzzy systems and is not achievable by neural networks (due to the different approximation mechanism). In conclusion, the previous analysis shows that the better performance achieved by hierarchical fuzzy systems is not a coincidence but due to the distinguished approximation mechanism of hierarchical fuzzy systems obtained by the proposed approach.

VII. Conclusion

The main conclusions of the investigation in this paper can be summarized as follows.

1) *Functions on discrete spaces have a very flexible hierarchical structure representation as shown in Theorem 1.* That is, given a function on a discrete space, for any given separable hierarchical structure, the function can be represented in a hierarchical form that matches the given separable hierarchical structure. It is this flexibility that makes it possible that hierarchical fuzzy systems can be designed to approximate functions on discrete spaces more simply and more effectively than those fuzzy approximation schemes to approximate continuous functions. As an application of the flexible hierarchical structure representation theorem, a simple discrete version of Kolmogorov's theorem (i.e., Theorem 2) is obtained.
2) *For functions on discrete spaces, hierarchical fuzzy systems are universal approximators with very flexible hierarchical structure as shown in Theorem 3.* That is, for any given function on a discrete space and for any desired separable hierarchical structure, the function can be approximated to any degree of accuracy by a hierarchical fuzzy system with the desired hierarchical structure. Although hierarchical fuzzy systems for function approximation on continuous spaces are also universal approximators, such flexibility in hierarchical structure is a special feature of hierarchical fuzzy approximation on discrete spaces. This flexibility is a useful property for applications as it enables the design of a hierarchical fuzzy system to achieve the approximation goal while being easy to understand and justified by domain experts. Furthermore, available human knowledge can be more easily and effectively utilized during system modeling and identification.
3) *For system modeling in high dimensions with limited data, a hierarchical fuzzy system identification method combining human knowledge with available numerical data is proposed.* The theoretical foundation of the proposed method is provided by Theorems A.1 and A.2 and practical guidance is given in design principles 1 and 2 to enhance the effectiveness method. To a certain degree, the proposed method can be viewed as formalization (in the context of fuzzy systems) of the widely used intuitive aggregation–modeling method mentioned in Section I.
4) *As an application of the previous identification method, a hierarchical fuzzy system approach for LMCP modeling has been proposed.* Applying this approach to the real commercial data has shown better performance than widely used regression and neural network approaches. Further, it is analyzed and verified that such a better performance is not a coincidence but due to the distinguished approximation mechanism of hierarchical fuzzy systems obtained by the proposed approach.

Appendix

Proposition A: There is no mapping from $U = \underset{i=1}{\overset{n}{\times}} [-1, 1]$ to R that is both one-to-one and continuous.

Proof: Suppose that such a mapping exists. Consider the following two sets in $U = \times_{i=1}^{n} [-1, 1]$:

$$S_1 = \{X = (x_1, \ldots, x_n) | x_1 \in (-0.5, 0.5); x_i = 0, i \neq 1\},$$

$$S_2 = \{X = (x_1, \ldots, x_n) | x_2 \in (-0.5, 0.5); x_i = 0, i \neq 2\}$$

which intersect only at $X_0 = (0, \ldots, 0)$. Under the mapping, the image of each set is an open interval of R because the mapping is continuous. So, the image of the two sets consists of two open intervals of R that intersect at a single point (as the mapping is one-to-one), which is impossible. Therefore, such a mapping does not exist.

Proof of Lemma 1: For $X_{2,j} = (u_{i_1^{(2,j)},k_1}, u_{i_2^{(2,j)},k_2}, \ldots, u_{i_{n_{2,j}}^{(2,j)},k_{n_{2,j}}}) \in U_{G_{2,j}} = \times_{k=1}^{n_{2,j}} U_{i_k^{(2,j)}}$, let

$$y^{(2,j)}_{k_1 k_2 \cdots k_{n_{2,j}}} = M_{2,j}\left(u_{i_1^{(2,j)},k_1}, u_{i_2^{(2,j)},k_2}, \ldots, u_{i_{n_{2,j}}^{(2,j)},k_{n_{2,j}}}\right)$$
$$k_l = 1, 2, \ldots, N_{i_l^{(2,j)}}, \qquad l = 1, 2, \ldots, n_{2,j}.$$

That is, $y^{(2,j)}_{k_1 k_2 \ldots k_{n_{2,j}}}$ is the function value of $M_{2,j}(X_{2,j})$ at $X_{2,j}$ and the set of all such values is denoted as

$$V_j = \left\{ y^{(2,j)}_{k_1 k_2 \ldots k_{n_{2,j}}} \middle| k_l = 1, 2, \ldots, N_{i_l^{(2,j)}}, l = 1, 2, \ldots, n_{2,j} \right\}$$
$$j = 1, 2, \ldots, m$$

which is the output variable space of function $M_{2,j}(X_{2,j})$. As $M_{2,j}(X_{2,j})$ is one-to-one mapping, then all elements of V_j are different from each other and there exists only one element $X_{2,j}$ in $U_{G_{2,j}}$ such that $y^{(2,j)}_{k_1 k_2 \ldots k_{n_{2,j}}} = M_{2,j}(X_{2,j})$. Further, as $U_{G_{2,j}}$ is a discrete space, then V_j is a discrete space (i.e., it has only finite elements).

Now define function g_1 on $U_{g_1} = (\times_{j=1}^{m} V_j) \times (\times_{k=1}^{n_1} U_{i_k^{(1)}})$ as follows. For any given $(Y, X_1) = (y_1, \ldots, y_m, X_1) \in U_{g_1}$, as each $y_j \in V_j$ $(j = 1, 2, \ldots, m)$, then there exists a unique element $X_{2,j}$ in $U_{G_{2,j}}$ such that $y_j = M_{2,j}(X_{2,j})$. Further, it can be implied from (8)–(10) that all elements of vectors X_1 and $X_{2,j}$ $(j = 1, 2, \ldots, m)$ form a unique vector $X = (x_1, \ldots, x_n) \in U$. Now define the value of g_1 at the given $(Y, X_1) = (y_1, \ldots, y_m, X_1) \in U_{g_1}$ as the value of G at $X = (x_1, \ldots, x_n) \in U$. That is, $g_1(Y, X_1) = g_1(y_1, \ldots, y_m, X_1) = G(x_1, x_2, \ldots, x_n)$. After the function g_1 defined before, it can be proved by the reverse process that for all $X = (x_1, \ldots, x_n) \in U$, $G(X) = g_1[M_{21}(X_{2,1}), \ldots, M_{2m}(X_{2,m}), X_1]$.

Proof of Lemma 2: Given $U = U_1 \times \cdots \times U_n$ and $U_i = \{u_{i,k} | u_{i,k} \in R, k = 1, 2, \ldots, N_i\}$ $(i = 1, \ldots, n)$, without loss of generality, it is assumed that $u_{i,1} < u_{i,2} < \cdots < u_{i,N_i}$ $(i = 1, \ldots, n)$. Now define $\Delta_i = \min_{1 \le k \le N_1 - 1} \{u_{i,k+1} - u_{i,k}\}$, $u_{i,N_i+1} = u_{i,N_i} + \Delta_i$ $(i = 1, \ldots, n)$, and a linear function in the following:

$$y = M(X) = w_1\left(\frac{x_1 - u_{1,1}}{u_{1,N_1+1} - u_{1,1}}\right) + w_2\left(\frac{x_2 - u_{2,1}}{u_{2,N_2+1} - u_{2,1}}\right) + \cdots + w_n\left(\frac{x_n - u_{n,1}}{u_{n,N_n+1} - u_{n,1}}\right) \tag{A1}$$

in which the weighting factors w_i $(i = 1, \ldots, n)$ are constructed recursively as follows:

$$w_1 = 1, \quad 0 < w_{i+1} < w_i \frac{\Delta_i}{u_{i,N_i+1} - u_{i,1}}, \quad i = 1, \ldots, n-1.$$

For the previous construction of the weighting factors w_i $(i = 1, \ldots, n)$, it is implied that, for any given i, k, and l $(i = 1, 2, \ldots, n-1, k = 1, 2, \ldots, N_i, l = 1, 2, \ldots, N_{i+1})$

$$\begin{aligned} & w_i \frac{u_{i,k} - u_{i,1}}{u_{i,N_i+1} - u_{i,1}} + w_{i+1} \\ & < w_i \frac{u_{i,k} - u_{i,1}}{u_{i,N_i+1} - u_{i,1}} + w_i \frac{u_{i,k+1} - u_{i,k}}{u_{i,N_i+1} - u_{i,1}} \\ & = w_i \frac{u_{i,k+1} - u_{i,1}}{u_{i,N_i+1} - u_{i,1}} \le w_i. \end{aligned} \tag{A2}$$

Let X_0 and X_0' be any two different elements in U, that is, $X_0 = (u_{1,k_1}, u_{2,k_2}, \ldots, u_{n,k_n})$, $X_0' = (u_{1,k_1'}, u_{2,k_2'}, \ldots, u_{n,k_n'})$, and $X_0 \neq X_0'$. If $u_{1,k_1} \neq u_{1,k_1'}$, then, without loss of generality, assume that $u_{1,k_1} < u_{1,k_1'}$ (this means $u_{1,k_1+1} \le u_{1,k_1'}$). Now from (A1) and (A2), it is implied that

$$\begin{aligned} M(X_0) &= w_1\left(\frac{u_{1,k_1} - u_{1,1}}{u_{1,N_1+1} - u_{1,1}}\right) + w_2\left(\frac{u_{2,k_2} - u_{2,1}}{u_{2,N_2+1} - u_{2,1}}\right) \\ &\quad + \cdots + w_n\left(\frac{u_{n,k_n} - u_{n,1}}{u_{n,N_n+1} - u_{n,1}}\right) \\ &\overset{\text{by (A2)}}{<} w_1\left(\frac{u_{1,k_1} - u_{1,1}}{u_{1,N_1+1} - u_{1,1}}\right) \\ &\quad + w_2\left(\frac{u_{2,k_2} - u_{2,1}}{u_{2,N_2+1} - u_{2,1}}\right) \\ &\quad + \cdots + w_{n-1}\left(\frac{u_{n-1,k_{n-1}} - u_{n-1,1}}{u_{n-1,N_{n-1}+1} - u_{n-1,1}}\right) + w_n \\ &\overset{\text{by (A2)}}{<} \cdots < w_1\left(\frac{u_{1,k_1} - u_{1,1}}{u_{1,N_1+1} - u_{1,1}}\right) \\ &\quad + w_2 < w_1\left(\frac{u_{1,k_1+1} - u_{1,1}}{u_{1,N_1+1} - u_{1,1}}\right) \\ &\le w_1\left(\frac{u_{1,k_1'} - u_{1,1}}{u_{1,N_1+1} - u_{1,1}}\right) \le w_1\left(\frac{u_{1,k_1'} - u_{1,1}}{u_{1,N_1+1} - u_{1,1}}\right) \\ &\quad + w_2\left(\frac{u_{2,k_2'} - u_{2,1}}{u_{2,N_2+1} - u_{2,1}}\right) \\ &\quad + \cdots + w_n\left(\frac{u_{n,k_n'} - u_{n,1}}{u_{n,N_n+1} - u_{n,1}}\right) \\ &= M(X_0'). \end{aligned}$$

That is, $M(X_0) \neq M(X_0')$. If $u_{1,k_1} = u_{1,k_1'}$ but i_0 is the smallest i such that $u_{i_0,k_{i_0}} \neq u_{i_0,k'_{i_0}}$, then similar to that mentioned earlier, it can be proved that $M(X_0) \neq M(X_0')$. Therefore, if $X_0 \neq X_0'$, then $M(X_0) \neq M(X_0')$. That is, the linear function given in (A1) is a one-to-one mapping from U to R. Noting that any linear function can be exactly represented by a fuzzy system as shown in [38], then it is implied immediately that there is a fuzzy system that is a one-to-one mapping from U to R.

Proof of Theorem 1: Theorem 1 is implied immediately by combing Lemmas 1 and 2.

Proof of Theorem 3: Under the condition of Theorem 3, it can be obtained from Theorem 1 that there exist fuzzy subsystems $f_{2,j}(X_{2,j})$ on $U_{G_{2,j}}$ $(j = 1, 2, \ldots, m)$ and function $g_1(Y, X_1)$ on $U_{g_1} = \left(\underset{j=1}{\overset{m}{\times}} V_j\right) \times U_{G_1}$ such that

$$G(X) = g_1\left[f_{21}(X_{2,1}), \ldots, f_{2m}(X_{2,m}), X_1\right]$$

where $Y = (y_1, \ldots, y_m)$, $X_1 = (x_{i_1^{(1)}}, \ldots, x_{i_{n_1}^{(1)}})$, $U_{G_1} = \underset{k=1}{\overset{n_1}{\times}} U_{i_k^{(1)}}$, $X_{2,j} = (x_{i_1^{(2,j)}}, \ldots, x_{i_{n_{2,j}}^{(2,j)}})$, $U_{G_{2,j}} = \underset{k=1}{\overset{n_{2,j}}{\times}} U_{i_k^{(2,j)}}$, and $V_j = \{y^{(2,j)}_{k_1 k_2 \ldots k_{n_{2,j}}} | y^{(2,j)}_{k_1 k_2 \ldots k_{n_{2,j}}} = f_{2,j}(u_{i_1^{(2,j)},k_1}, \ldots, u_{i_{n_{2,j}}^{(2,j)},k_{n_{2,j}}}), k_l = 1, 2, \ldots, N_{i_l^{(2,j)}},\ l = 1, 2, \ldots, n_{2,j}\}$, $j = 1, \ldots, m$.

Now consider function $g_1(Y, X_1) = g_1(y_1, \ldots, y_m, X_1)$ on the discrete space $U_{g_1} = (\underset{j=1}{\overset{m}{\times}} V_j) \times (\underset{k=1}{\overset{n_1}{\times}} U_{i_k^{(1)}})$. It is known from [20] that any function in a discrete space can be extended to be a continuous function $\hat{g}_1(Y, X_1)$ on $\hat{U}_{g_1} = (\underset{j=1}{\overset{m}{\times}} [\underline{y}_j, \overline{y}_j]) \times (\underset{k=1}{\overset{n_1}{\times}} [\underline{u}_{i_k^{(1)}}, \overline{u}_{i_k^{(1)}}])$ [where $\underline{y}_j = \min_{y \in V_j} y$, $\overline{y}_j = \max_{y \in V_j} y$ $(j = 1, 2, \ldots, m)$ and $\underline{u}_{i_k^{(1)}} = \min_{u \in U_{i_k^{(1)}}} u$, $\overline{u}_{i_k^{(1)}} = \max_{u \in U_{i_k^{(1)}}} u$ $(k = 1, 2, \ldots, n_1)$] in the sense that $\hat{g}_1(Y, X_1) = g_1(Y, X_1)$ for any $(Y, X_1) \in U_{g_1} = (\underset{j=1}{\overset{m}{\times}} V_j) \times (\underset{k=1}{\overset{n_1}{\times}} U_{i_k^{(1)}})$. As $\hat{g}_1(Y, X_1)$ is a continuous function on $\hat{U}_{g_1}$, then it is implied immediately from the universal approximation property of fuzzy systems on continuous spaces that there exists a fuzzy system $\hat{f}_1(Y, X_1)$ on $\hat{U}_{g_1}$ such that $||\hat{g}_1 - \hat{f}_1||_\infty = \max_{(Y,X_1) \in \hat{U}_{g_1}} |\hat{g}_1(Y, X_1) - \hat{f}_1(Y, X_1)| < \varepsilon$. Now defining $f_1(Y, X_1) = \hat{f}_1(Y, X_1)$ and $F(X) = f_1\left[f_{21}(X_{2,1}), \ldots, f_{2m}(X_{2,m}), X_1\right]$ and noticing that $\hat{g}_1(Y, X_1) = g_1(Y, X_1)$ for any $(Y, X_1) \in U_{g_1} = (\underset{j=1}{\overset{m}{\times}} V_j) \times (\underset{k=1}{\overset{n_1}{\times}} U_{i_k^{(1)}})$, this implies for $X \in U$ that

$$\begin{aligned} &|G(X) - F(X)| \\ &\quad = |g_1\left[f_{2,1}(X_{2,1}), \ldots, f_{2,m}(X_{2,m}), X_1\right] \\ &\qquad - f_1\left[f_{2,1}(X_{2,1}), \ldots, f_{2,m}(X_{2,m}), X_1\right]| \\ &\quad \leq ||\hat{g}_1 - \hat{f}_1||_\infty < \varepsilon. \end{aligned}$$

Theorem A.1: Let $y = G(X)$ given in (1) be a function on the discrete input space $U = \underset{i=1}{\overset{n}{\times}} U_i$ given in (2) and $G_1, G_{2,1}, \ldots, G_{2,m}$ be any given disjoint grouping of the input variables $\{x_1, x_2, \ldots, x_n\}$ satisfying (8)–(11). If fuzzy subsystem $f_{2,j}(X_{2,j})(j = 1, 2, \ldots, m)$ given in (19) is a one-to-one mapping from $U_{G_{2,j}} = \underset{k=1}{\overset{n_{2,j}}{\times}} U_{i_k^{(2,j)}}$ to R, then, for any given $\varepsilon > 0$, there exists a fuzzy system $f_1(Y, X_1)$ given in (22) such that the resulting hierarchical fuzzy system $F(X) = f_1\left[f_{2,1}(X_{2,1}), \ldots, f_{2,m}(X_{2,m}), X_1\right]$ satisfies

$$||G - F||_\infty = \max_{X \in U} |G(X) - F(X)| < \varepsilon$$

where $V_j = \left\{y_i | y_i = f_{2,j}(X_{2,j}), X_{2,j} \in U_{G_{2,j}}\right\}$ $(j = 1, \ldots, m)$, $Y = (y_1, \ldots, y_m) \in V = \prod_{j=1}^m V_j$, and $X_1 = (x_{i_1^{(1)}}, \ldots, x_{i_{n_1}^{(1)}}) \in U_{G_1} = \prod_{k=1}^{n_1} U_{i_k^{(1)}}$.

Proof: The proof is similar to Theorem 3 and the details are omitted here in order to reduce the length of the paper.

Proposition B: $G(X)$ is a monotone increase (decrease) function on a discrete (or continuous) input space $U = \underset{i=1}{\overset{n}{\times}} U_i$ if and only if, for any given i and any fixed $(\bar{x}_1, \ldots, \bar{x}_{i-1}, \bar{x}_{i+1}, \ldots, \bar{x}_n) \in U_1 \times \cdots \times U_{i-1} \times U_{i+1} \times \cdots \times U_n$, $g(x_i) = G(\bar{x}_1, \ldots, \bar{x}_{i-1}, x_i, \bar{x}_{i+1}, \ldots, \bar{x}_n)$ is a 1-D monotone increase (decrease) function on U_i.

Proof of Proposition B: The proof can be obtained from the definition of monotone functions.

Theorem A.2: Let $y = G(X)$ given in (1) be a function on the discrete input space $U = \underset{i=1}{\overset{n}{\times}} U_i$ given in (2) and $G_1, G_{2,1}, \ldots, G_{2,m}$ be a desired disjoint grouping of the input variables $\{x_1, x_2, \ldots, x_n\}$ satisfying (8)–(11). Suppose that, for any given $X_j, X_j' \in U_{G_{2,j}} = \underset{k=1}{\overset{n_{2,j}}{\times}} U_{i_k^{(2,j)}}$, either

$$\begin{aligned} &G(X_1, \ldots, X_{j-1}, X_j, X_{j+1}, \ldots, X_m) \\ &\quad \geq G(X_1, \ldots, X_{j-1}, X_j', X_{j+1}, \ldots, X_m) \end{aligned}$$

for any fixed $X_1, \ldots, X_{j-1}, X_{j+1}, \ldots, X_m$ or the reverse inequality holds and can be determined by available human knowledge. Then, there exist one-to-one lower level fuzzy subsystems $y_j = f_{2,j}(X_{2,j})(j = 1, 2, \ldots, m)$ constructed based on available human knowledge such that $G(X) = g_1\left[f_{2,1}(X_{2,1}), \ldots, f_{2,m}(X_{2,m}), X_1\right]$ and $g_1(y_1, \ldots, y_m, X_1)$ is a monotone function related to $Y = (y_1, \ldots, y_m)$.

Proof: For the given input space $U = U_1 \times \cdots \times U_n$ with $U_i = \{u_{i,l} | u_{i,l} \in R, l = 1, \ldots, N_i\}$ $(i = 1, \ldots, n)$. Without loss of generality, it is assumed that $u_{i,1} < u_{i,2} < \cdots < u_{i,N_i}$ (otherwise we can rearrange the index such that this holds) and define $u_{i,0} = u_{i,1}, u_{i,N_i+1} = u_{i,N_i}$ $(i = 1, \ldots, n)$. Construct the triangular membership function $A_{i,l_i}(x_i) = \Delta_{i,l_i}\left(x_i, u_{i,l_i-1}, u_{i,l_i}, u_{i,l_i+1}\right)$ $(l_i = 1, 2, \ldots, N_i)$ and $A^{(2,j)}_{l_1 \cdots l_{n_{2,j}}}(X_{2,j}) = \prod_{k=1}^{n_{2,j}} A_{i_k^{(2,j)},l_k}(x_{i_k^{(2,j)}})$ and the lower level fuzzy systems

$$\begin{aligned} y_j &= f_{2,j}(X_{2,j}) \\ &= \sum_{l_1 \cdots l_{n_{2,j}} \in I_{2,j}} \frac{A^{(2,j)}_{l_1 \cdots l_{n_{2j}}}(X_{2,j})}{\sum_{l_1 \cdots l_{n_{2,j}} \in I_{2,j}} A^{(2,j)}_{l_1 \cdots l_{n_{2,j}}}(X_{2,j})} y^{(2,j)}_{l_1 \cdots l_{n_{2,j}}} \end{aligned}$$

where $y^{(2,j)}_{l_1 \cdots l_{n\,2,j}}$ $(l_k = 1, \ldots, N_{i_k^{(2,j)}}; k = 1, \ldots, n_{2,j})$ are constructed as follows. If

$$X_{2,j} = X^{(2,j)}_{l_1 \ldots l_{n\,2,j}} = \left(u_{i_1^{(2,j)}, l_1}, \ldots, u_{i_{n\,2,j}^{(2,j)}, l_{n\,2,j}}\right),$$

$$X'_{2,j} = X^{(2,j)}_{l'_1 \ldots l'_{n\,2,j}} = \left(u_{i_1^{(2,j)}, l'_1}, \ldots, u_{i_{n\,2,j}^{(2,j)}, l'_{n\,2,j}}\right)$$

and

$$G(X_1, X_{2,1}, \ldots, X_{2,j-1}, X_{2,j}, X_{2,j+1}, \ldots, X_{2,m})$$
$$\geq G(X_1, X_{2,1}, \ldots, X_{2,j-1}, X'_{2,j}, X_{2,j+1}, \ldots, X_{2,m})$$

then $y^{(2,j)}_{l_1 \cdots l_{n\,2,j}} > y^{(2,j)}_{l'_1 \cdots l'_{n\,2,j}}$. As

$$\begin{aligned} f_{2,j}\left(X^{(2,j)}_{l_1 \ldots l_{n\,2,j}}\right) &= f_{2,j}\left(u_{i_1^{(2,j)}, l_1}, \ldots, u_{i_{n\,2,j}^{(2,j)}, l_{n\,2,j}}\right) \\ &= y^{(2,j)}_{l_1 \ldots l_{n\,2,j}} > y^{(2,j)}_{l'_1 \ldots l'_{n\,2,j}} = f_{2,j}\left(X^{(2,j)}_{l'_1 \ldots l'_{n\,2,j}}\right) \\ &= f_{2,j}\left(u_{i_1^{(2,j)}, l'_1}, \ldots, u_{i_{n\,2,j}^{(2,j)}, l'_{n\,2,j}}\right) \end{aligned}$$

then $y_j = f_{2,j}(X_{2,j})$ is one-to-one mapping on U. Let $V_j = \{y^{(2,j)}_{l_1 \cdots l_{n\,2,j}} | y^{(2,j)}_{l_{1\,2} \cdots l_{n\,2,j}} = f_{2,j}(u^{(2,j)}_{i_1, l_1}, \ldots, u_{i_{n\,2,j}^{(2,j)}, l_{n\,2,j}}) l_k = 1, 2, \ldots, N_{i_l{}^{(2,j)}}, k = 1, 2, \ldots, n_{2,j}\}, j = 1, \ldots, m,$ $V = \underset{j=1}{\overset{m}{\times}} V_j$, and define $g_1(Y, X_1) = g_1(y_1, \ldots, y_m, X_1)$ on $U_{g_1} = V \times U_{G_1}$ as follows: $g_1(y_1, \ldots, y_m, X_1) = G(X_1, X_{2,1}, \ldots, X_{2,m})$ if $y_j = f_{2,j}(X_{2,j})$. As a result, $G(X) = g_1(Y, X_1) = g_1[f_{2,1}(X_{2,1}), \ldots, f_{2,m}(X_{2,m}), X_1]$. Further, for any $Y = (y_1, \ldots, y_{j-1}, y_j, y_{j+1}, \ldots, y_m) \in V$, $Y' = (y_1, \ldots, y_{j-1}, y'_j, y_{j+1}, \ldots, y_m) \in V$, and $y_j > y'_j$, based on the definition of V_j and $V = \underset{j=1}{\overset{m}{\times}} V_j$, there are $X_{2,k} \in U_{2,k}$ $(k = 1, \ldots, m)$ and $X'_{2,j} \in U_{2,j}$ such that $y_k = f_{2,k}(X_{2,k})(k = 1, 2, \ldots, m)$ and $y'_j = f_{2,j}(X'_{2,j})$. By the construction of $f_{2,j}(X_{2,j})$ as earlier, we have $g_1(y_1, \ldots, y_j, \ldots, y_m, X_1) = G(X_1, X_{2,1}, \ldots, X_{2,j}, \ldots, X_{2,m}) > G(X_1, X_{2,1}, \ldots, X'_{2,j}, \ldots, X_{2,m}) = g_1(y_1, \ldots, y'_j, \ldots, y_m, X_1)$. That is, when $y_1, \ldots, y_{j-1}, y_{j+1}, \ldots, y_m$ and X_1 are fixed, $g_1(y_1, \ldots, y_j, \ldots, y_m, X_1)$ is a monotone function on y_j. Then, from Proposition B, this immediately implies the theorem.

Acknowledgment

The authors would like to express their particular thanks to Dr. P. Cook, Head of Research at Knowledge Support Systems (KSS), for the support and many very useful discussions related to this research. The authors also would like to thank the reviewers for their constructive comments and suggestions, which have helped to improve the paper both in the presentation and methodology.

References

[1] J. Abonyi, R. Babuska, and F. Szeifert, "Modified Gath–Geva fuzzy clustering for identification of Takagi–Sugeno fuzzy models," *IEEE Trans. Syst., Man., Cybern.—B*, vol. 32, no. 5, pp. 612–621, Oct. 2002.

[2] J. J. Buckley, "Universal fuzzy controllers," *Automatica*, vol. 28, pp. 1245–1248, 1992.

[3] R. Babuska. (2002, Feb.). *Fuzzy systems, modeling and identification* [Online]. Available: http://www.dcsc.tudelft.nl/∼babuska/transp/fuzzmod.pdf

[4] R. L. David, "Evaluation of retail store attributes and sales performance," *Eur. J. Marketing*, vol. 7, pp. 89–102, 1973.

[5] R. J. P. de Figueiredo, "Implications and applications of Kolmogorov's superposition theorem," *IEEE Trans. Autom. Control.*, vol. 25, no. 6, pp. 1227–1231, Dec. 1980.

[6] M. M. Fischer and M. Reismann, "A methodology for neural spatial interaction modeling," *Geograph. Anal.*, vol. 34, pp. 93–111, 2002.

[7] A. Ghosh and S. L. McLafferty, *Location Strategies for Retail and Service Firms*. Toronto, ON, Canada: D.C. Heath and Company, 1987.

[8] T. Hernandez and D. Bennison, "The art and science of retail location decisions," *Int. J. Retail Distrib. Manage.*, vol. 28, no. 8, pp. 357–367, 2000.

[9] R. Hecht-Nielsen, "Kolmogorov's mapping neural network existence theorem," in *Proc. IEEE Int. Conf. Neural Netw.*, vol. 3, San Diego, CA, 1987, pp. 11–14.

[10] O. Huwendiek and W. Brockmann, "Function approximation with decomposed fuzzy systems," *Fuzzy Sets Syst.*, vol. 101, pp. 273–296, 1999.

[11] B. Igelnik and N. Parikh, "Kolmogorov's spline network," *IEEE Trans. Neural Netw.*, vol. 14, no. 4, pp. 725–733, Jul. 2003.

[12] J.-S. Jang, "ANFIS: Adaptive-network-based fuzzy inference system," *IEEE Trans. Syst., Man, Cybern.*, vol. 23, no. 3, pp. 665–685, May/Jun. 1993.

[13] M. G. Joo and J. S. Lee, "A class of hierarchical fuzzy systems with constraints on the fuzzy rules," *IEEE Trans. Fuzzy Syst.*, vol. 13, no. 2, pp. 194–203, Apr. 2005.

[14] M. Kumar, R. Stoll, and N. Stoll, "A min–max approach to fuzzy clustering, estimation, and identification," *IEEE Trans. Fuzzy Syst.*, vol. 14, no. 2, pp. 248–262, Apr. 2006.

[15] B. Kosko, "Fuzzy systems as universal approximators," in *Proc. IEEE Int. Conf. Fuzzy Syst.*, San Diego, CA, Mar. 1992, pp. 1153–1162.

[16] R. J. Kuo, S. C. Chi, and S. S. Kao, "A decision support system for selecting convenience store location through integration fuzzy AHP and artificial neural networks," *Comput. Ind.*, vol. 47, pp. 199–214, 2002.

[17] M. Landajo, M. J. Rio, and R. Perez, "A note on smooth approximation capabilities of fuzzy systems," *IEEE Trans. Fuzzy Syst.*, vol. 9, no. 2, pp. 229–236, Apr. 2001.

[18] G. G. Lorentz, *Approximation of Functions*. New York: Holt, Reinhart and Winston, 1966.

[19] T. Poggio and F. Girosi, "A theory of networks for approximation and learning," Artif. Intell. Lab., MIT, Cambridge, A.I. Memo No. 1140, 1989.

[20] M. J. D. Powell, *Approximation Theory and Methods*. Cambridge, U.K.: Cambridge Univ. Press, 1981.

[21] D. Rogers, "A review of sales forecasting models most commonly applied in retail site evaluation," *Int. J. Retail Distrib. Manage.*, vol. 20, no. 4, pp. 3–11, 1992.

[22] F. Scarselli and T. A. Chung, "Universal approximation using feedforward neural networks: A survey of some existing methods, and some new results," *Neural Netw.*, vol. 11, pp. 15–37, 1998.

[23] G. A. F. Seber and C. J. Wild, *Nonlinear Regression*. New York: Wiley, 2003.

[24] A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Stat. Comput.*, vol. 14, pp. 199–222, 2004.

[25] A. J. Smola and B. Schölkopf, "On a kernel-based method for pattern recognition, regression, approximation and operator inversion," *Algorithmica*, vol. 22, pp. 211–231, 1998.

[26] D. A. Sprecher, "A representation theorem for continuous functions of several variables," in *Proc. Amer. Math. Soc.*, 1965, vol. 16, pp. 200–203.

[27] D. A. Sprecher, "A numerical implementation of Kolmogorov's superpositions II," *Neural Netw.*, vol. 10, pp. 447–457, 1997.

[28] V. Torra, "A review of the construction of hierarchical fuzzy systems," *Int. J. Intell. Syst.*, vol. 17, pp. 531–543, 2002.

[29] V. Vapnik, S. Golowich, and A. Smola, "Support vector method for function approximation, regression estimation, and signal processing," in *Advances in Neural Information Processing Systems 9*, M. C. Mozer,

M. I. Jordan, and T. Petsche, Eds. Cambridge, MA: MIT Press, 1997, pp. 281–287.

[30] D. Wang, X.-J. Zeng, and J. A. Keane, "Hierarchical hybrid fuzzy-neural networks for approximation with mixed input variables," *Neurocomputing*, vol. 70, pp. 3019–3033, 2007.

[31] L.-X. Wang, "Fuzzy systems are universal approximators," in *Proc. IEEE Int. Conf. Fuzzy Syst.*, San Diego, CA, Mar. 1992, vol. SMC7, no. 10, pp. 1163–1170.

[32] L.-X. Wang, *A Course in Fuzzy Systems and Control*. Upper Saddle River, NJ: Prentice-Hall, 1997.

[33] L.-X. Wang, "Analysis and design of hierarchical fuzzy systems," *IEEE Trans. Fuzzy Syst.*, vol. 7, no. 5, pp. 617–624, Oct. 1999.

[34] F. Witlox and H. Timmermans, "MATISSE: A knowledge-based system for industrial site selection and evaluation," *Comput., Environ. Urban Syst.*, vol. 24, pp. 23–43, 2000.

[35] H. Ying, "Sufficient conditions on general fuzzy systems as function approximations," *Automatica*, vol. 30, pp. 521–525, 1994.

[36] K. Zeng, N. Y. Zhang, and W. L. Xu, "A comparative study on sufficient conditions for Takagi–Sugeno fuzzy systems as universal approximators," *IEEE Trans. Fuzzy Syst.*, vol. 8, no. 6, pp. 773–778, Dec. 2000.

[37] X.-J. Zeng and M. G. Singh, "Approximation accuracy analysis of fuzzy systems as function approximators," *IEEE Trans. Fuzzy Systems*, vol. 4, no. 1, pp. 44–63, Feb. 1996.

[38] X.-J. Zeng and M. G. Singh, "Decomposition property of fuzzy systems and its applications," *IEEE Trans. Fuzzy Syst.*, vol. 4, no. 2, pp. 149–165, May 1996.

[39] X.-J. Zeng and J. A. Keane, "Approximation capabilities of hierarchical fuzzy systems," *IEEE Trans. Fuzzy Syst.*, vol. 13, no. 5, pp. 659–672, Oct. 2005.

[40] X.-J. Zeng, J. Y. Goulerma, J. A. Keane, and P. Liatsis, "Approximation capabilities of hierarchical neural-fuzzy systems for function approximation on discrete spaces," *Int. J. Comput. Intell. Res.*, vol. 1, pp. 29–41, 2005.

Xiao-Jun Zeng (M'07) received the B.Sc. degree in mathematics and the M.Sc. degree in control theory and operation research from Xiamen University, Xiamen, China, and the Ph.D. degree in computation from the University of Manchester, Manchester, U.K., in 1982, 1985, and 1996, respectively.

Since 2002, he has been with the University of Manchester, where he is currently a Senior Lecturer in the School of Computer Science. From 1996 to 2002, he was with Knowledge Support Systems, Ltd. (KSS), Manchester, where he was a Scientific Developer, a Senior Scientific Developer, and the Head of Research, developing intelligent decision support systems. From 1985 to 1992, he was with the Department of Computer and Systems Sciences, Xiamen University, where he was a Lecturer and an Associate Professor. He is a member of the Editorial Board of the *International Journal of Computational Intelligence Research*. His current research interests include fuzzy systems and control, neural networks, machine learning, decision support systems, intelligent systems, and data mining.

Dr. Zeng is an Associate Editor of the IEEE TRANSACTIONS ON FUZZY SYSTEMS. He is a member of the Peer Review College of the U.K. Engineering and Physical Sciences Research Council. He was the recipient of the European Information Society Technologies (IST) Award in 1999 and the Microsoft European Retail Application Developer (RAD) Awards in 2001 and 2003.

John Yannis Goulermas (M'98) was born in Greece in 1970. He received the B.Sc.(Hons.) degree (with first class) in computation from the University of Manchester Institute of Science and Technology (UMIST), Manchester, U.K., in 1994, and the M.Phil. and Ph.D. degrees in electrical engineering from UMIST, in 1996 and 2000.

He was engaged in research in the area of image processing and machine vision. He was a full-time Scientific Developer and later a part-time Consultant in the area of pricing modeling and optimization in industry. He was with the University of Salford, where he was engaged in research on virtual reality with applications to engineering and biomechanics and intelligent gait analysis. In 2005, he joined the Department of Electrical Engineering and Electronics, University of Liverpool, Liverpool, U.K. His current research interests include pattern recognition, mathematical optimization, and machine vision, with application areas primarily including biomedical and clinical engineering and industrial monitoring.

Panos Liatsis (S'87–M'91) received the Diploma degree in electrical engineering from the Democritus University of Thrace, Xanthi, Greece, and the Ph.D. degree in electrical engineering from the University of Manchester Institute of Science and Technology (UMIST), Manchester, U.K.

In November 2003, he joined the School of Engineering and Mathematical Sciences, City University, London, U.K., where he is currently a Reader in sensor systems and the Director of the Information and Biomedical Engineering Centre. He was a Lecturer in the Control Systems Centre, UMIST. His current research interests include image processing, computer vision, intelligent systems, and pattern recognition, with applications to biomedical engineering and intelligent transportation systems.

Dr. Liatsis is a member of the International Programme Committee of major conferences in signal and image processing, and was the Programme Chair of the 9th International Workshop on Systems, Signals and Image Processing. He is a regular Project Assessor and a Reviewer for the European Commission. He is a member of the Peer Review College of the U.K. Engineering and Physical Sciences Research Council. He is also a member of the Institution of Engineering and Technology, the Institute of Measurement and Control, and the Technical Chamber of Greece.

Di Wang (S'03–M'05) received the Bachelor's degree in computer science and technique from Tianjin University, Tianjin, China, in 2001, and the Ph.D. degree in computer engineering form Nanyang Technological University (NTU), Singapore, 2005.

She is with ThinkAnalytics Ltd., Glasgow, U.K., as a Software Engineer Research Associate. From 2005 to 2007, she was with the School of Informatics, University of Manchester, as a Postdoctoral Research Associate. Her current research interests include fuzzy systems, neural networks, hybrid hierarchical neural fuzzy systems, statistical models, dynamic systems, and their application in time series forecasting.

John A. Keane (M'05) received the B.Sc. degree in computation and computer science from the University of Manchester Institute of Science and Technology (UMIST), Manchester, U.K., and the M.Sc. degree in computation and computer science from the University of Manchester, in 1985 and 1989, respectively.

He is the M. G. Singh Chair in Computing Science in the School of Computer Science, University of Manchester. He was in industry with the Trustees Savings Bank, Philips Data Systems, and ICL. His current research interests include the areas of data intensive systems and data mining. He is the Deputy Director of the U.K. National Center for Text Mining. He is a member of the Editorial Board of *Simulation Modelling*.

Prof. Keane is an Associate Editor of the IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART C, and was a member of both the U.K. Computer Science Research Strategy Group and the Engineering and Physical Sciences Research Council (EPSRC) Peer Review College.