Learning Nonlinear Multiregression Networks Based on Evolutionary Computation

Kwong-Sak Leung
Department of Computer Science and Engineering
The Chinese University of Hong Kong
Shatin
Hong Kong

Man-Leung Wong
Department of Computing and Decision Sciences
Lingnan University
Hong Kong

Wai Lam
Department of Systems Engineering and Engineering Management
The Chinese University of Hong Kong
Shatin
Hong Kong

Zhenyuan Wang
Department of Mathematics
University of Nebraska at Omaha,
Omaha, NE 68182,
U.S.A.

Kebin Xu
Department of Computer Science and Engineering
The Chinese University of Hong Kong
Shatin
Hong Kong

Abstract

This paper describes a novel knowledge discovery and data mining framework dealing with nonlinear interactions among domain attributes. Our network-based model provides an effective and efficient reasoning procedure to perform prediction and decision making. Unlike many existing paradigms based on linear models, the attribute relationship in our framework is represented by nonlinear nonnegative multiregressions based on the Choquet integral. This kind of multiregression is able to model a rich set of nonlinear interactions directly. Our framework involves two layers. The outer layer is a network structure consisting of network elements as its components, while the inner layer is concerned with a particular network element modeled by Choquet integrals. We develop a Fast Double Optimization Algorithm (FDOA) for learning the multiregression coefficients of a single network element. Using this local learning component and Multiregression-Residual-Cost Evolutionary Programming (MRCEP), we propose a global learning algorithm, called MRCEP-FDOA for discovering the network structures and their elements from databases. We have conducted a series of experiments to assess the effectiveness of our algorithm and investigate the performance under different parameter combinations as well as sizes of the training data sets. The empirical results demonstrate that our framework can successfully discover the target network structure and the regression coefficients.

1 Introduction

Knowledge discovery and data mining have received increasing attention due to its potential applications in a wide range of different areas [6]. One objective of knowledge discovery and data mining is to discover useful patterns or knowledge from data. Such knowledge can be used for performing intelligent tasks such as reasoning and problem solving. This paper describes a novel knowledge discovery and data mining framework dealing with interactions among domain attributes. Our network-based model provides an effective reasoning procedure to perform prediction and decision making. Unlike many existing paradigms based on linear models, the attribute relationship in our framework is represented by a network structure consisting of a number of network elements. Each network element is formed by a nonlinear nonnegative multiregression based on the Choquet integral. This kind of multiregression is able to model a rich set of attribute interaction directly. We develop a Fast Double Optimization Algorithm (FDOA) for learning the multiregression coefficients of a single network element. Using this local learning component, we propose a global learning algorithm, called MRCEP-FDOA, for discovering the network structure as well as the values of multiregression coefficients in all network elements from a given data set, where MRCEP (Multiregression-Residual-Cost Evolutionary Programming) employs Evolutionary Programming [7, 8, 10] to learn the network structure. It integrates a Multiregression-Residual-Cost (MRC) metric and several genetic operators including three structure-guided and a knowledge-guided operators.

A single nonlinear multiregression based on nonlinear integrals has been used to establish the relationship between an objective attribute and several predictive attributes [39, 41]. Such a single model is not powerful enough to represent intricate relations among attributes in a medium or large scale database. Employing a network structure can provide a richer representation of the problem space resulting in a more accurate model. For a network element, the multiregression may be modeled by some existing techniques, such as projection pursuit multiregression [12, 26]. In the projection pursuit multiregression, a

linear combination (it is the Lebesgue-like integral) with unknown coefficients is first applied to project the multi-dimensional data into one-dimensional line R. Then a nonlinear transformation from R to R is adopted. Finally, the sum of such a kind of transformations with respective to linear combination is used to fit the data. The one-dimensional nonlinear transformations may be approximated by natural splines. All unknown coefficients and parameters are optimally determined by training a neural network from given data. Thus, the nonlinearity of the projection pursuit multiregression is just embodied in the one-dimensional nonlinear transformations. Therefore, the effect of describing a nonlinear relation with inherent interaction among attributes is limited.

Wang proposed the interaction concept based on an nonadditive "importance measure" and a fuzzy integral [35]. Murofushi and Sugeno also proposed a definition for the notion of interaction based on the use of Choquet integral [29]. Later this notion was extended by Grabisch [15] and axiomatized by Grabisch and Roubens [16]. By using a nonadditive set function, our nonlinear multiregression based on the Choquet integral can directly represent the inherent interaction among predictive attributes to the objective attribute.

Another knowledge representation model known as Bayesian networks has been developed [19, 31, 24]. Unfortunately, most of them mainly cater for discrete attributes. Some Bayesian networks attempt to deal with continuous attributes [11], but, they mainly rely on the assumption of parametric or semi-parametric families such as Gaussian distributions. Our proposed network structure of nonlinear integrals can directly represent the nonlinear relationship of a set of continuous attributes. The network structure and the relevant learning framework are different from the hierarchical structure proposed in Kwon and Sugeno [20]. In our structure, any attribute could be objective attribute and any attribute could be predictive attribute. They are discovered by the learning algorithm. In contrast, in Kwon and Sugeno's hierarchical structure, the objective attribute is pre-defined.

We have conducted a series of experiments to demonstrate the effectiveness of our algorithm and investigate the performance under different parameter combinations as well as sizes of training data sets. The empirical results demonstrate that our framework can

successfully discover the target network structure and the regression coefficients. Moreover, our framework does not require a complete attribute ordering as input. If such an ordering is given, it is easy to incorporate such ordering information for mining.

This paper is organized as follows. Section 2 presents the knowledge representation we adopt in our framework. Section 3 covers the details of the basic nonlinear multiregression network element and a fast algorithm for learning the regression coefficients. Our EP-based network structure learning algorithm is given in Section 4. Section 5 describes the empirical results and the evaluation of our framework. We present the conclusions in Section 6.

2 Nonlinear Multiregression Networks

A nonlinear multiregression network attempts to represent the inter-dependencies among the attributes in a domain. Such a network consists of a set of nodes and directed edges. To be convenient for prediction, we require that there is no cycle in the network. Each node represents an attribute or a variable in the problem domain. Each attribute can take on a continuous value. There are directed edges connecting attributes forming a directed acyclic graph (DAG). Figure 1 shows an example of a nonlinear multiregression network with 5 attributes. The set of incoming edges of an attribute represents an association between the parent attributes and the current attribute. For example, N_1 and N_2 in Figure 1 are parent attributes of N_3. N_2, N_3, and N_4 are parent attributes of N_5. Figure 2 shows a practical example of a simple nonlinear multiregression network in the health care area.

A network element is defined as a basic subnetwork comprising a child attribute and the set of its parent attributes. For instance, the subnetwork concerned with N_1, N_2, and N_3 in Figure 1 forms a network element. One unique characteristic of a network element is that the association between the parent attributes and the child attribute is a nonlinear multiregression; it is not necessarily linear. Unlike traditional correlation models, our nonlinear multiregression is based on the Choquet integral with respect to a nonadditive set function. A novel feature of this nonlinear integral is that it can model a rich set of

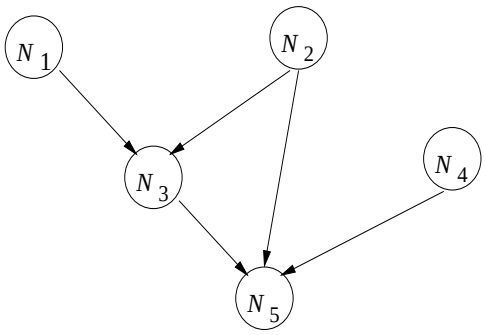

Figure 1: An example of a nonlinear multiregression network

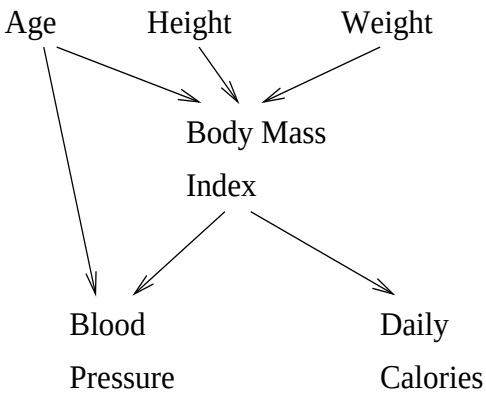

Figure 2: A practical example of a nonlinear multiregression network

inherent interactions directly among attributes. More details with illustrative examples are described in Section 3.1.

Once this network is constructed, it can be used for conducting inference and prediction in the problem domain. The inference process follows the direction of the edges in the network, i.e., from parents to child. Generally, an attribute can be a child for a network element and it can also be a parent for another network element. For example, consider the network concerned with the health care area depicted in Figure 2. Attribute "Body Mass Index" can be predicted from attributes "Age", "Height", and "Weight". Attribute "Body Mass Index" together with attribute "Age" can also help predict attribute "Blood Pressure". Indirect inference can be done via the network. For instance, if we have the observations for attributes "Age", "Height", and "Weight", we can predict attribute "Body Mass Index" which in turn can be used for the prediction of attributes "Blood Pressure" and "Daily Calories".

In principle, such nonlinear multiregression networks can be constructed by joint effort of domain experts and knowledge engineers. However it typically requires a lot of time and human resources due to the knowledge engineering bottleneck. A promising way to solve this problem is to automatically discover the network structure from raw data. Let $N_1, \ldots, N_n$ be the n attributes in the domain. The form of the raw data is given as follows:

N_1	N_2	$\cdots$	N_n
d_{11}	d_{12}	$\cdots$	d_{1n}
d_{21}	d_{22}	$\cdots$	d_{2n}
$\vdots$			
d_{L1}	d_{L2}	$\cdots$	d_{Ln}

where L is the number of records in the database. Each record corresponds to a case or a scenario observed in the real world. The goal is to automatically discover the best underlying nonlinear multiregression network that captures the inherent associations among the attributes. In addition to the structure, the learning method should also estimate the

corresponding coefficients associated with the structure.

We propose a learning framework, called MRCEP-FDOA. This framework consists of two layers. The inner layer is concerned with a basic network element modeled by a multiregression nonlinear integral. We develop a fast algorithm for learning the multiregression coefficients of a network element called Fast Double Optimization Algorithm (FDOA). The outer layer of our MRCEP-FDOA framework is concerned with the topological structure of a network. To discover the structure, we develop a learning algorithm called Multiregression-Residual-Cost Evolutionary Programming (MRCEP). In the following discussion, we first describe the form of association among attributes in a parent-child network element and FDOA. Then we give a detailed description of MRCEP.

3 Network Elements

3.1 Nonlinear Multiregression Based on the Choquet Integral

Figure 3 depicts a network element. Let $X = \{x_1, x_2, \ldots, x_m\}$ be the set of parent attributes and y be the child attribute. Parents $x_1, x_2, \ldots, x_m$ are called predictive attributes and the child y is called the objective attribute. All attributes can take on real numbers.

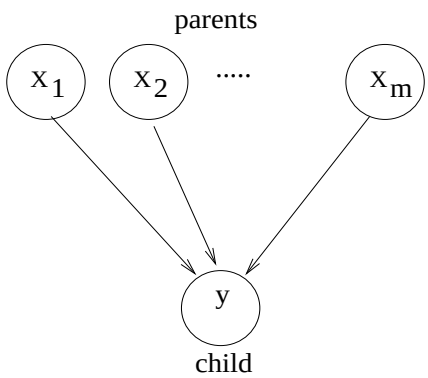

Figure 3: A network element

A traditional tool to describe how the objective attribute depends on the predictive

attributes is the linear multiregression that has the following model:

$$Y = \sum_{i=1}^{m} a_i f(x_i) + N(a_0, \sigma^2), \tag{1}$$

where Y is the value of y; $f(x_i)$ is the value of x_i; $a_0, a_1, a_2, \ldots, a_m$ are unknown regression coefficients; and $N(a_0, \sigma^2)$ is a random noise that has a normal distribution with mean a_0 and variance σ^2. It is a linear model. Without any loss of generality, we can assume that all regression coefficients are nonnegative (otherwise, we can switch the sign of corresponding data). Then, the above model can be expressed by the classical Lebesgue-like integral. In fact, if we define a set function μ, which indicates the importance of each x_i to the objective attribute y, on the class of all singletons $\{x_i\}$ by $\mu(\{x_i\}) = a_i/q$, $i = 1, 2, \ldots, m$, where $q = a_1 + a_2 + \ldots + a_m$, then μ can be uniquely extended to be an additive measure onto the power set of X, $\mathcal{P}(X)$, which is the class of all subsets of X [17]. Thus, the Lebesgue-like integral $\int f \mathrm{d}\mu$ can be used as a comprehensive numerical assessment to the objective attribute, where the integrand f is a function defined on X whose value at attribute x_i is $f(x_i), i = 1, 2, \ldots, m$. Hence, the above mentioned linear multiregression model can be expressed as:

$$Y = q \int f \mathrm{d}\mu + N(a_0, \sigma^2), \tag{2}$$

where the integral is the Lebesgue-like integral of f with respect to μ.

The Lebesgue-like integral is a linear functional. Using such a linear model needs a basic assumption that, to the objective attribute, there is no interaction among predictive attributes. This means that the influences to the objective attribute from predictive attributes are independent such that the global contribution of the set of all predictive attributes to the objective attribute is just the simple sum of their respective contributions. However, in many real-world problems, the interaction among predictive attributes to the objective attribute cannot be ignored [35, 38]. In this case, we should use a set function μ defined on $\mathcal{P}(X)$ to represent the importance of each individual predictive attribute as well as the joint importance for every possible combination of predictive attributes. Mathematically, set function μ can be expressed as $\mu : \mathcal{P}(X) \to [0, 1]$ satisfying $\mu(\emptyset) = 0$ and

$\mu(X) = 1$, where $\emptyset$ is the empty set. Here, condition $\mu(X) = 1$ is called the regularity of μ. Usually, set function μ is not additive, even not monotone. The nonadditivity of μ portrays the inherent interaction among predictive attributes to the objective attribute [38]. This kind of interaction is totally different from the concept of correlation in statistics. The latter describes the relation between the observations of attributes and it is an external relation among attributes.

Since X is finite, the continuity requirement on μ is insignificant. Therefore, when μ is monotone, that is, $A \subset B \Rightarrow \mu(A) \leq \mu(B)$ for any sets A and B in $\mathcal{P}(X)$, such a set function is also called a fuzzy measure [29, 35].

When the nonadditive set function is used to replace the additive measure, the classical Lebesgue-like integral fails. The Choquet integral, one type of nonlinear integrals, is available for our purpose in this situation. The Choquet integral is defined as follows [29, 36]:

$$\int f \mathrm{d}\mu = \int_0^\infty \mu(F_\alpha)\mathrm{d}\alpha \tag{3}$$

where f is a function on X with range $[0, \infty)$, and $F_\alpha = \{x | f(x) \geq \alpha, x \in X\}$, $\alpha \in [0, \infty)$, is the α-cut set of f. It is a generalization of the Lebesgue-like integral, that is, when μ is additive, the Choquet integral coincides with the Lebesgue-like integral. In fact, Equation 3 itself is one of the equivalent expressions of the Lebesgue-like integral of a nonnegative measurable function f with respect to a classical measure μ. In general, the Choquet integral is not linear. However, it possesses most good properties of the Lebesgue-like integral.

To calculate the value of the Choquet integral of a given function f, the values of f, $\{f(x_1), f(x_2), \ldots, f(x_m)\}$, should be first rearranged into a nondecreasing order, that is,

$$f(x_1^*) \leq f(x_2^*) \leq \ldots f(x_m^*), \tag{4}$$

where $(x_1^*, x_2^*, \ldots, x_m^*)$ is a permutation of $(x_1, x_2, \ldots, x_m)$. Then, the value of the Choquet integral can be obtained by computing expression

$$\int f \mathrm{d}\mu = \sum_{i=1}^{m} [f(x_i^*) - f(x_{i-1}^*)]\mu(\{x_i^*, x_{i+1}^*, \ldots, x_m^*\}), \tag{5}$$

where $f(x_0^*) = 0$.

The following is an example of using the Choquet integral as an aggregation tool [38]. The linear combination, a traditional aggregation tool, is just a special case of the Choquet integral where the set function is additive. The example also shows a very intuitive explanation to the Choquet integral. A similar example can also be found in [28].

Example 1. There are three workers x_1, x_2, and x_3 working for $f(x_1) = 10$, $f(x_2) = 15$, and $f(x_3) = 7$ days respectively to manufacture some kind of products. Their efficiencies of working alone are 5, 6, and 8 products per day respectively. If they worked separately, that is, there was no interaction among them, then the number of total products for such a working period would be a simple linear combination of the numbers of their working days:

$$10 \times 5 + 15 \times 6 + 7 \times 8 = 196.$$

However, they work together actually. Suppose that they begin to work from the same day. Their joint efficiencies are not the simple sum of the corresponding efficiencies given above, but are listed in the following:

$\{x_1, x_2\}$	14
$\{x_1, x_3\}$	7
$\{x_2, x_3\}$	16
$\{x_1, x_2, x_3\}$	25

These efficiencies can be regarded as a regular nonadditive set function (not necessarily monotone), μ, defined on the power set of $X = \{x_1, x_2, x_3\}$ with $\mu(\emptyset) = 0$ (the meaning is that there is no product if no worker) multiplied by a constant $q = 25$. For example, $\mu(\{x_1\}) = 5/25 = 0.2$ and $\mu(\{x_1, x_2\}) = 14/25 = 0.56$. Here inequality $\mu(\{x_1, x_2\}) > \mu(\{x_1\}) + \mu(\{x_2\})$ means that x_1 and x_2 have a good cooperation, while inequality $\mu(\{x_1, x_3\}) < \mu(\{x_1\}) + \mu(\{x_3\})$, and even $\mu(\{x_1, x_3\}) < \mu(\{x_3\})$, means that x_1 and x_3 have a very bad relationship and they are not suitable for working together. Set

function μ is not necessarily monotone and is called an efficiency measure. The nonadditivity of μ represents the interaction among three workers to the total products. In such a manner, during the first 7 days, all workers work together with efficiency $q \cdot \mu(\{x_1, x_2, x_3\})$, and the number of products is $f(x_3) \cdot q \cdot \mu(\{x_1, x_2, x_3\}) = 7 \times 25 = 175$; during the next $f(x_1) - f(x_3)$ days, workers x_1 and x_2 work together with efficiency $q \cdot \mu(\{x_1, x_2\})$, and the number of products is $[f(x_1) - f(x_3)] \cdot q \cdot \mu(\{x_1, x_2\}) = 3 \times 14 = 42$; during the last $f(x_2) - f(x_1)$ days, only x_2 works with efficiency $q \cdot \mu(\{x_2\})$, and the number of products is $[f(x_2) - f(x_1)] \cdot q \cdot \mu(\{x_2\}) = 5 \times 6 = 30$. Thus, value

$$q \int f \mathrm{d}\mu = 175 + 42 + 30 = 247$$

is just the total number of products manufactured by these workers during the period mentioned above.

If there was no interaction among these workers, then their joint efficiencies should be

$$\begin{aligned}
q\mu(\{x_1, x_2\}) &= q(\mu(\{x_1\}) + \mu(\{x_2\})) = 11, \\
q\mu(\{x_1, x_3\}) &= q(\mu(\{x_1\}) + \mu(\{x_3\})) = 13, \\
q\mu(\{x_2, x_3\}) &= q(\mu(\{x_2\}) + \mu(\{x_3\})) = 14, \\
q\mu(\{x_1, x_2, x_3\}) &= q(\mu(\{x_1\}) + \mu(\{x_2\}) + \mu(\{x_3\})) = 19.
\end{aligned}$$

Thus, the corresponding value of the Choquet integral would be

$$\begin{aligned}
& q \int f \mathrm{d}\mu = qf(x_3)\mu(\{x_1, x_2, x_3\}) + q[f(x_1) - f(x_3)]\mu(\{x_1, x_2\}) + q[f(x_2) - f(x_1)]\mu(\{x_2\}) \\
= \; & f(x_3)q\mu(\{x_1, x_2, x_3\}) + [f(x_1) - f(x_3)]q\mu(\{x_1, x_2\}) + [f(x_2) - f(x_1)]q\mu(\{x_2\}) \\
= \; & 7 \times 19 + 3 \times 11 + 5 \times 6 \\
= \; & 196.
\end{aligned}$$

This coincides with the resulting number of products when the linear combination is used. Here, the fact that the linear combination (essentially, it is the Lebesque-like integral) is a special case of the Choquet integral is verified. □

Thus, a new nonlinear multiregression based on the Choquet integral can be established as follows [38, 39]:

$$Y = q \int f \mathrm{d}\mu + N(c, \sigma^2), \tag{6}$$

where Y is the value of y, f is a function on X with $f(x_i)$ as its value at $x_i, i = 1, 2, ..., m, \mu$ is a nonnegative set function satisfying $\mu(\emptyset) = 0$ and $\mu(X) = 1$, q is a proportional divisor, and $N(c, \sigma^2)$ is a normally distributed random variable with mean c and variance σ^2. In such a model, c, q, and the value of μ at each set (except $\emptyset$ and X in $\mathcal{P}(X)$) are regarded as unknown regression coefficients. This nonlinear model is a generalization of the classical multiregression and is supported by a natural mechanism mentioned above. Given observed data of $x_1, x_2, \ldots, x_m$, and y, to estimate the values of regression coefficient is just the inverse problem of calculating the value of the Choquet integral. This is illustrated in the next example.

Example 2. Three workers x_1, x_2, and x_3 are hired for manufacturing some kind of products. Table 1 is a record of the numbers of their working days and the corresponding number of total products in each month during the last year. By using the data given in Table 1, we want to estimate their respective efficiency (the number of products per day) as well as their joint efficiencies. Suppose that they worked together in a similar manner shown in Example 1: they begin to work on the same day together and each worker continues to work until the last day of their respective working period without any break in each month. For instance, in January, all of them worked together for 10 days, then x_2 and x_3 worked together for 6 days and, finally, x_2 worked alone for 4 days. In such a manner, the Choquet integral based nonlinear multiregression can be used to solve this problem. Let Y be the monthly total number of products, f be the function defined on set $\{x_1, x_2, x_3\}$ to denote the numbers of working days for these workers, and $q \cdot \mu$ be the efficiency measure as mentioned in Example 1. Here, q is a proportional constant and such that $\mu(X) = 1$

Month	x_1	x_2	x_3	Total products
1	10	20	16	300
2	15	10	12	229
3	20	20	24	420
4	18	12	15	279
5	22	16	20	366
6	13	18	20	340
7	17	19	19	368
8	24	14	10	278
9	10	21	18	324
10	19	25	13	356
11	18	16	16	326
12	10	10	12	210

Table 1: The data of the numbers of working days for three workers

Set	value of μ
$\emptyset$	0
$\{x_1\}$	0.15
$\{x_2\}$	0.20
$\{x_1, x_2\}$	0.60
$\{x_3\}$	0.25
$\{x_1, x_3\}$	0.50
$\{x_2, x_3\}$	0.70
$\{x_1, x_2, x_3\}$	1

Table 2: The values of μ obtained in Example 2

which is required in our algorithm. Then

$$Y = q \int f \mathrm{d}\mu + N(c, \sigma^2). \tag{7}$$

By using an adaptive genetic algorithm described in [38], or a fast algorithm developed in this paper (Section 3.2), the value of c, q and the values of μ can be determined, and the value of σ^2 can also be estimated. In fact, this multiregression problem has a unique precise solution (with $\sigma^2 = 0$) : $c = 0$, $q = 20$, and the values of μ listed in Table 2.

Since the efficiency function is $q \cdot \mu$, we know that the efficiency of x_1 is 3 products per day, the efficiency of x_2 is 4 products per day, the efficiency of x_3 is 5 products per day, the joint efficiency of x_1 and x_2 is 12 products per day, the joint efficiency of x_1 and x_3 is 10 products per day, the joint efficiency of x_2 and x_3 is 14 products per day, and the joint efficiency of all three workers is 20 products per day. Here, we can see that the joint efficiency of x_1 and x_2 is much larger than the sum of the efficiency of x_1 and the efficiency of x_2. This indicates that there is a strong inherent interaction between x_1 and x_2 to the number of products. Such an interaction is totally different from the statistical concept of

correlation between variables x_1 and x_2. The latter can be estimated from the data given in the second and the third columns of Table 1. In fact, the correlation of x_1 and x_2 in this example is 0.0041, that is, x_1 and x_2 are almost independent statistically. $\square$

For the data given in Table 1, by using the projection pursuit multiregression, even if it is not impossible to get an approximation of the nonlinear relation between the total products and the numbers of the working days, the complexity will be very high, that is, a large number of nodes in the expressions of natural splines and a large number of terms in the sum mentioned above are needed. Moreover, our model can be used to replace the linear combination in the projection pursuit multiregression to reduce its complexity and has a good fitness in many real-world problems where the inherent interaction among predictive attributes exist significantly.

Now, we return to discuss how to optimally determine the unknown regression coefficients from data. Suppose that the data concerning the attributes of the parents and the child are available in the following form:

$$\begin{array}{ccccc} f_{11} & f_{12} & \cdots & f_{1m} & Y_1 \\ f_{21} & f_{22} & \cdots & f_{2m} & Y_2 \\ & & \vdots & & \\ f_{L1} & f_{L2} & \cdots & f_{Lm} & Y_L \end{array}$$

Then, the estimated values of these unknown regression coefficients c^*, q^* and the set function μ^* can be determined by minimizing regression residual.

$$e = \sqrt{\frac{1}{L}\sum_{j=1}^{L}(Y_j - Y_j^*)^2} \tag{8}$$

in which $Y_j^* = q^* \int f_j \mathrm{d}\mu^* + c^*$, $j = 1, 2, \ldots, L$, where f_j is a function on X with $f_j(x_i) = f_{ji}, i = 1, 2, \ldots, m$.

The estimation of these coefficients in each network element is an optimization problem. In our previous works [38, 41], an adaptive genetic algorithm was adopted to solve such an

optimization problem for a single nonlinear multiregression based on the Choquet integral and the Wang integral successfully. However, the adaptive genetic algorithm demands a large amount of computational resources. Fortunately, the calculation of the Choquet integral for a given function f_j concerns only the values of set function at a few sets which f_j involve. Murofushi and Grabisch [14] made use of this property to identify the set function μ used in the Choquet integral. Mori and Murofushi [27] proposed the idea of using the gradient on a chain. Inspired by these techniques, we develop a Fast Double Optimization Algorithm (FDOA) to estimate the regression coefficients (including the values of set function μ, proportional divisor q and mean c) for each network element. Here we use the word "fast" to make contrast to the adaptive genetic algorithm in our previous works.

3.2 Learning Nonlinear Multiregression Coefficients

To easily illustrate the algorithm determining the unknown coefficients in the nonlinear multiregression, we introduce a concept of complete nest in the power set $\mathcal{P}(X)$ as follows.

Definition. A subclass $\mathcal{N}$ of $\mathcal{P}(X)$ is called a nest if there exists a relation either $A \subset B$, or $B \subset A$ for any $A, B \in \mathcal{N}$. A nest $\mathcal{N}$ is said to be complete if it consists of $m+1$ different sets where m is the cardinality of X.

It is easy to see that there exists a permutation of $\mathcal{N}$, denoted as $A_0, A_1, \ldots, A_m$, satisfying conditions that $\emptyset = A_0 \subset A_1 \subset \cdots A_m = X$ and $A_i - A_{i-1}$ is a singleton, $i = 1, 2, \ldots, m$, if $\mathcal{N}$ is a complete nest. If μ is a fuzzy measure mentioned above, then $0 = \mu(A_0) \leq \mu(A_1) \leq \cdots \leq \mu(A_m) = 1$. These $m+1$ values are simply denoted by $u(0), u(1), \ldots, u(m)$.

For a given function f as mentioned above, first we rearrange its values into a nondecreasing order:

$$f(x_1^*) \leq f(x_2^*) \leq \ldots \leq f(x_m^*) \tag{9}$$

where $(x_1^*, x_2^*, \ldots, x_m^*)$ is a permutation of $(x_1, x_2, \ldots, x_m)$. This permutation corresponds to a complete nest $\mathcal{N}_f : A_0 = \emptyset, A_i = \{x_{m-i+1}^*, x_{m-i+2}^*, \ldots, x_m^*\}, i = 1, 2, \ldots, m$, and the values of μ at sets in the nest, $u(0), u(1), \ldots, u(m)$. Thus,

$$\int f \mathrm{d}\mu = \sum_{i=1}^{m} [f(x_i^*) - f(x_{i-1}^*)] \cdot u(m - i + 1) \tag{10}$$

Equation 10 indicates that the calculation of the value of the Choquet integral for a given function concerns only the values of μ at sets in nest $\mathcal{N}_f$. Hence, a Local-Revising Strategy (LRS) can be adopted in a fast iterative algorithm to find an approximate solution of the nonlinear multiregression when suitable data are available. In the algorithm, a double optimization technique is adopted. To be more precise, a local-revising on the values of μ is used to reduce errors first, then c^* and q^* are determined by the Least Square Method (LSM). We call such a new algorithm the Fast Double Optimization algorithm (FDOA). The algorithm is detailed as follows.

1. Input m; // m is the number of predictive attributes
2. Input L; // L is the number of records in the database
3. Input stop_e; // stop_e is a chosen small positive number
   // that is used for the stopping condition.
4. For $j = 1$ to L step 1
   input (f_{ji}) and (Y_j); // input the training examples
5. For each set $A \in \mathcal{P}(X)$
   $\mu^*(A) := |A|/m$; // initialize the values of μ^*
6. For $i = 0$ to m step 1
   $u^*(i) := i/m$; // initialize the values of u^*
7. $Num := 1$;
8. Calculate the residual e

i. $\bar{Y} := \frac{1}{L}\sum_{j=1}^{L} Y_j$;

ii. $\bar{I} := \frac{1}{L}\sum_{j=1}^{L} \int f_j \mathrm{d}\mu^*$;

iii. $q^* := \frac{\sum_{j=1}^{L}(\int f_j \mathrm{d}\mu^* - \bar{I})(Y_j - \bar{Y})}{\sum_{j=1}^{L}(\int f_j \mathrm{d}\mu^* - \bar{I})^2}$;

iv. $c^* := \bar{Y} - q^*\bar{I}$;

v. $e := \sqrt{\frac{1}{L}\sum_{j=1}^{L}(Y_j - q^* \int f_j \mathrm{d}\mu^* - c^*)^2}$;

9. While $e >= \text{stop\_}e$

   i. For $j = 1$ to L step 1

   // for each datum $(f_{ji}, Y_j), j = 1, 2, \ldots, L$, in the given data,

   // do the following steps

      a. $e_j := q^* \cdot \int f_j \mathrm{d}\mu^* + c^* - Y_j$; // calculate error e_j

      b. For $i = 1$ to m-1 step 1

      $w(i) := u^*(i) - \alpha \cdot e_j(f_j(x^*_{m-i+1}) - f_j(x^*_{m-i}))$;

      // $\alpha \in [0, 1]$ is a given constant.

      // Note that we have $u^*(0) = 0$ and $u^*(m) = 1$ at all time.

      c. If $e_j > 0$ then

      for $i = 1$ to m-1 step 1

         i. // For each $u^*(i)$, corresponding to $\mu^*(A_i)$, find $max\{\mu^*(B)\}$

         Find $max\{\mu^*(B) \mid B \subset A_i, \mu^*(B)$ has already been modified, $B \in \mathcal{P}(X)\}$;

         ii. // Modify $u^*(i)$ with a new value

         If $max\{\mu^*(B)\} < w(i)$ then

         $u^*(i) := w(i)$;

         Else

         $u^*(i) = max\{\mu^*(B)\}$;

      Else // $e_j <= 0$

      for $i = m$-1 to 1 step -1

i. // For each $u^*(i)$, corresponding to $\mu^*(A_i)$, find $min\{\mu^*(B)\}$

Find $min\{\mu^*(B) \mid B \supset A_i, \mu^*(B)$ has already been modified$\}$;

ii. // Modify $u^*(i)$ with a new value

If $min\{\mu^*(B)\} > w(i)$ then

$u^*(i) := w(i)$;

Else

$u^*(i) = min\{\mu^*(B)\}$;

ii. $Num := Num + 1$;

iii. Calculate the residual e

a. $\bar{Y} := \frac{1}{L}\sum_{j=1}^{L} Y_j$;

b. $\bar{I} := \frac{1}{L}\sum_{j=1}^{L} \int f_j \mathrm{d}\mu^*$;

c. $q^* := \frac{\sum_{j=1}^{L}(\int f_j \mathrm{d}\mu^* - \bar{I})(Y_j - \bar{Y})}{\sum_{j=1}^{L}(\int f_j \mathrm{d}\mu^* - \bar{I})^2}$;

d. $c^* := \bar{Y} - q^*\bar{I}$;

e. $e := \sqrt{\frac{1}{L}\sum_{j=1}^{L}(Y_j - q^* \int f_j \mathrm{d}\mu^* - c^*)^2}$;

10. For each set A

If $\mu^*(A)$ has not been modified in step 9

i. Find $min\{\mu^*(B) \mid A \subset B, \mu^*(B)$ has already been modified$, B \in \mathcal{P}(X)\}$;

ii. Find $max\{\mu^*(C) \mid C \subset A, \mu^*(C)$ has already been modified$, C \in \mathcal{P}(X)\}$;

iii. $\mu^*(A) := (max\{\mu^*(C)\} + min\{\mu^*(B)\})/2$;

// modify $\mu^*(A)$ with a new value

11. Output Num and the estimated regression coefficients c^*, q^*, and μ^*;

4 The Network Structure Learning Algorithm

4.1 Evolutionary Algorithms

Evolutionary algorithms are weak search techniques based on the principle of natural selection and evolution to achieve the goals of function optimization and machine learning [1, 2]. A potential solution to the problem is encoded as an individual. An evolutionary algorithm maintains a group of individuals, called the population, to explore the search space. A fitness function evaluates the performance of each individual to measure how close it is to the solution. The search space is explored by using genetic operators to evolve new individuals. The evolution is based on the Darwinian principle of evolution through natural selection: the fitter individual has a higher chance of survival, and tends to pass on its favorable traits to its offspring. Existing evolutionary algorithms include Genetic Algorithms [13, 18], Genetic Programming [21, 22, 23], Evolution Strategies [34, 4, 3], and Evolutionary Programming [7, 8, 10]. The various kinds of evolutionary algorithms differ mainly in the evolution models assumed, the evolutionary operators employed, the selection methods, and the fitness functions used [8].

Genetic Algorithms (GAs) use a fixed-length binary bit string as an individual. Three genetic operators are used to search for better individuals. Reproduction operator copies the unchanged individual to the next generation. Crossover operator exchanges bits between two parents. Mutation operator randomly changes individual bits. Genetic Programming (GP) extends GAs by using a tree structure as an individual. GAs and GP model evolution at genetic level. They emphasize on the acquisition of genetic structures at the symbolic level and regularities of the solutions.

Evolutionary Programming (EP) uses the highest level of abstraction by emphasizing on the adaptation of behavioral properties of various species. On the other hand, the idea of optimization is used in Evolution Strategies (ES) and the structures being optimized are the individuals of the population. Various behavioral properties of the individuals are parameterized and their values evolved as an optimization process.

Wong, Lam, and Leung [40] performed a series of experiments to compare their approach for learning Bayesian network structures using EP with the classical GA approach described in [25]. They observe that the EP approach is superior both in terms of quality of solutions and computational time in most data sets they tested. Thus, we apply EP to solve the problem of learning structures of nonlinear multiregression networks.

EP is a stochastic optimization and learning strategy that emphasizes the behavioral linkage between parents and their offspring rather than seeking to emulate specific genetic operators as observed in nature [7, 8, 10]. EP models the reproductive relationship between species behavior in successive generations. Consequently, EP only applies mutations to preserve behavioral similarity between offspring and their parents [7, 8, 10].

There are three important differences between EP and the classical GA. Firstly, there is no constraint on the representation. The classical GA involves encoding the problem solutions as fixed-length binary strings [13, 18]. In EP, the representation follows from the problem. For example, a Bayesian network can be represented in the same manner as it is implemented. Thus the mutation operation does not demand and assume any particular encoding method. Secondly, the mutation operators simply change aspects of the parent according to a statistical distribution. Minor modifications in the behavior of the offspring occur more frequently than substantial variations in the behavior of the offspring. Furthermore, the severity of mutations is often reduced as the global optimum is approached. Thirdly, EP employs mutation operators only while the classical GA applies mutation, crossover, and other genetic operators [3, 9, 32].

In the following subsections, we describe our approach to network structure learning based on EP. The fitness function which is derived from the residual, called Multiregression-Residual-Cost (MRC) metric, is also presented.

4.2 The Multiregression-Residual-Cost Metric

The network structure learning algorithm makes use of a cost measure, called Multiregression-Residual-Cost (MRC) metric, that can measure the fitness of a candidate network structure

to the data set. One characteristic of this metric is that it gives a lower value for a network element that is close to the true association among the parent set and the child attributes. This cost metric can be decomposed into each individual attribute. Let $C(B)$ be the cost metric of a network structure B; Π_{N_i} be the parent set of N_i. With overloading of the notation $C()$, it can be expressed as:

$$C(B) = \sum_{N_i \in N} C(N_i, \Pi_{N_i}) \tag{11}$$

As N_i and Π_{N_i} form a network element such as the network shown in Figure 3, we can apply the method described in Section 3.2 to estimate the unknown regression coefficients and the corresponding minimum error. Consequently, we define:

$$C(N_i, \Pi_{N_i}) = e^2 \tag{12}$$

where e is given in Equation 8. The rationale is that the network capturing appropriate dependency relationships among the attributes would allow prediction of an attribute with low error.

Ideally we would like to find a network structure that has the lowest C. We call such a network an optimal network. In situations where an optimal solution cannot be obtained due to limited computing resources, we wish to find a network with C as low as possible.

4.3 The EP-based Learning Algorithm

Our approach, called MRCEP-FDOA, adopts the MRC cost metric and Evolutionary Programming (EP) [7, 8, 10] to learn nonlinear multiregression network structures. This learning problem is difficult because the number of network structures increases exponentially with the number of attributes. The size of the search space is given by the formula [33]:

$$f(n) = \sum_{i=1}^{n} (-1)^{i+1} \begin{pmatrix} n \\ i \end{pmatrix} 2^{i(n-i)} f(n-i), f(0) = 1, f(1) = 1 \tag{13}$$

The size of the search space for different numbers of attributes is given in Table 3. Our MRCEP-FDOA algorithm starts with an initial population of directed acyclic graphs

Number of attributes	Size of the search space
1	1
2	3
3	25
4	543
5	29281
6	3781503
7	1.139e9
8	7.837e11
9	1.213e15
10	4.175e18
11	3.160e22
12	5.219e26
13	1.868e31
14	1.439e36
15	2.377e41
20	2.344e72
25	2.659e111
30	2.714e158

Table 3: The size of the search space

(DAGs) called parental network structures. Each parental network structure is evaluated by using the cost metric described above. Next, each parental network structure creates an offspring by performing a series of mutations to the parental network structure. The probabilities of executing 1, 2, 3, 4, 5, or 6 instances of mutation are 0.2, 0.2, 0.2, 0.2, 0.1, and 0.1, respectively. If mutations generate an invalid offspring that is cyclic, the algorithm deletes the edges of the offspring that invalidate the DAG conditions. It can detect cycles and delete the related edges in $O(|N| + |E|)$ time, where $|N|$ is the number of nodes and $|E|$ is the number of edges in the graph [5]. The new offspring are then evaluated by using the cost metric. The next generation of parental network structures are selected from the current generation of parental network structures and offspring. The algorithm performs this selection by requiring each DAG to compete against p randomly chosen DAGs. If the cost metric of the former is lower than or equal to the chosen opponent in each competition, the former receives one score. The algorithm retains the groups of DAGs with the highest scores as parental network structures of the next generation. It then repeats this process

until the maximum number of generations G is reached. The algorithm is summarized as follows:

1. Set t to 0.
2. Create an initial population, Pop(t), of PS random DAGs. The initial population size is PS.
3. Each DAG in the population Pop(t) is evaluated using the cost metric.
4. While t is smaller than the maximum number of generations G
   i. Each DAG in Pop(t) produces one offspring by performing a number of mutation operations. If the offspring has cycles, delete the edges of the offspring that violate the DAG condition.
   ii. The DAGs in Pop(t) and all new offspring are stored in the intermediate population Pop$'(t)$. Thus, the size of Pop$'(t)$ is $2 * PS$.
   iii. Conduct a number of pairwise competitions over all DAGs in Pop$'(t)$. Let B_i be the DAG being conditioned upon, p opponents are selected randomly from Pop$'(t)$ with equal probability, where $p < |\text{Pop}'(t)|$. Let $B_{ij}, 1 \leq j \leq p$, be the randomly selected opponent DAGs. The B_i gets one more score if $C(B_i) \leq C(B_{ij})$, $1 \leq j \leq p$, where $C(B_i)$ is the cost metric function of a DAG B_i. Thus, the maximum score of a DAG is p.
   iv. Select PS DAGs with the highest scores from Pop$'(t)$ and store them in the new population Pop($t+1$).
   v. Increase t by 1.
5. Return the DAG with lowest cost metric found in any generation of a run as the result of the algorithm.

The learning algorithm uses a variety of structure-guided mutation operators and a knowledge-guided mutation operator to produce new offspring from existing DAGs. Formally, let B be an existing DAG to be mutated, $N = \{N_1, N_2, \ldots, N_n\}$ be the set of attributes in a domain, and E be the set of edges in B. The operators generate a new offspring by modifying E. These operators are detailed in the following discussions.

4.4 Structure-Guided Mutation Operators

Simple Mutation:

This operator randomly adds an edge e_{ij} from attributes N_j to N_i, where $i \neq j$, if the edge does not already exist. Otherwise, it deletes the edge from E. For example, the network structure in Figure 4 is obtained from the network structure in Figure 2 by adding an edge from attribute "Weight" to attribute "Daily Calories". The network structure in Figure 5 is obtained from the network structure in Figure 2 by deleting the edge from attribute "Age" to attribute "Blood Pressure".

Reversion:

This operator randomly selects an edge, says e_{ij}, from E, and modifies the direction of the edge. In other words, the set of edges, E', of the offspring is:

$$E' = (E - \{e_{ij}\}) \cup \{e_{ji}\} \tag{14}$$

For example, the network structure in Figure 6 is obtained from the network structure in Figure 2 by reversing the edge from attribute "Body Mass Index" to attribute "Daily Calories".

Move:

This operator modifies the parent set of an attribute, says N_i, if Π_{N_i} is not empty. Specifically, it deletes an attribute N_k where $N_k \in \Pi_{N_i}$, from the parent set of N_i randomly,

and adds a new attribute N_j to Π_{N_i}, if $N_j \notin (\Pi_{N_i} \cup N_i)$. For example, the network structure in Figure 7 is obtained from the network structure in Figure 2 by deleting the edge from attribute "Body Mass Index" to attribute "Daily Calories" and then adding an edge from attribute "Weight" to attribute "Daily Calories".

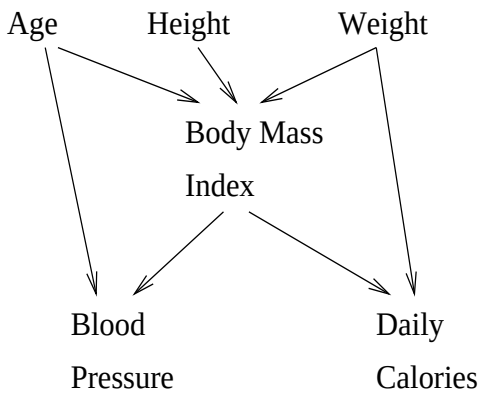

Figure 4: An example of using simple mutation to add an edge to a network structure. It is obtained from the network structure in Figure 2 by adding an edge from attribute "Weight" to attribute "Daily Calories"

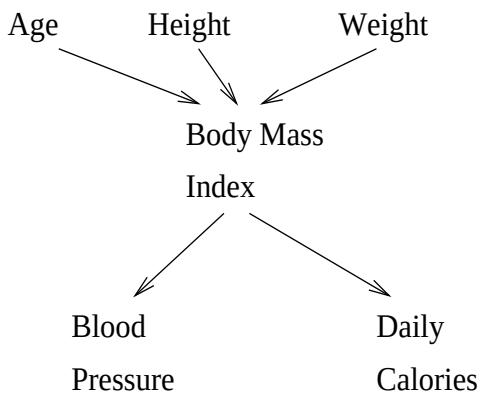

Figure 5: An example of using simple mutation to delete an edge from a network structure. It is obtained from the network structure in Figure 2 by deleting the edge from attribute "Age" to attribute "Blood Pressure"

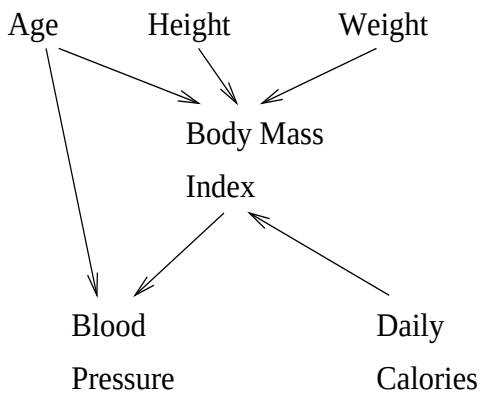

Figure 6: An example of using reversion mutation to modify the direction of an edge in a network structure. It is obtained from the network structure in Figure 2 by reversing the edge from attribute "Body Mass Index" to attribute "Daily Calories"

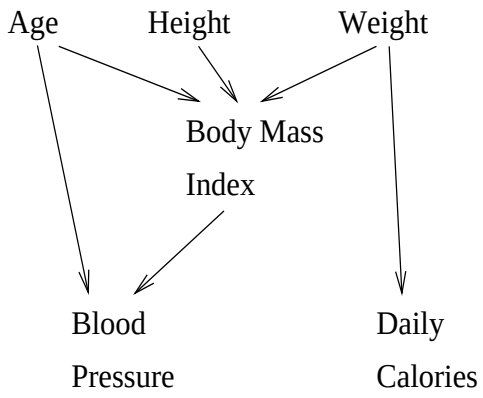

Figure 7: An example of using move mutation to move an edge in a network structure. It is obtained from the network structure in Figure 2 by deleting the edge from attribute "Body Mass Index" to attribute "Daily Calories" and then adding an edge from attribute "Weight" to attribute "Daily Calories"

4.5 Knowledge-Guided Mutation

This operator is similar to the simple mutation operator. It removes an existing edge from a nonlinear multiregression network or adds an edge if there is no edge between the corresponding attributes. The main difference between these two operators is that knowledge-guided mutation considers the cost metrics of all possible edges and determines which edge should be removed or inserted. The cost metric of an edge from N_j to N_i, where $i \neq j$, is computed by using $C(N_i, \{N_j\})$. Before the learning algorithm is executed, the cost metrics of all possible edges is computed and stored. When the knowledge-guided mutation operator determines that an existing edge of the parental network structure B should be removed, it retrieves the stored cost metrics of all edges in E and those edges with higher cost metrics are deleted with higher probabilities. On the other hand, if the knowledge-guided mutation operator decides to add an edge to the parental network structure, it gets the stored cost metrics of the edges not in E, and the edges with lower cost metrics will have higher probabilities of being added.

5 Empirical Evaluation

We have conducted a number of experiments to evaluate the performance of our approach. Synthetic data sets are used in the experiments since we can validate the quality of the results under different conditions of complexity levels. The method for the construction of the data set will be described in detail in the following subsection. In each experiment, our learning framework (MRCEP-FDOA) attempts to discover a network structure and the corresponding nonlinear multiregression coefficients from data. The learning algorithm takes the data set only as input and does not know the original networks that generate the data set in any way during the learning process. MRCEP-FDOA is employed to search the optimal network structure, while the algorithm FDOA described in Section 3.2 is employed to learn the multiregression coefficients for every network element. The goodness of the network structure is evaluated by the MRC cost metric C. The lower the value, the better

the network. Various combinations of different parameters as well as the size of training data are investigated to test the effectiveness and efficiency of our approach. The experiments are conducted on Sun Ultra 1/170 workstations.

5.1 Data Set Construction

To generate a data set, we start with a given network structure. The network topology is first determined. Then we set the values of coefficients such as μ, c, and q for each network element. Next, we generate the data records. To be more precise, consider a nonlinear multiregression network, $N = \{N_1, N_2, \ldots, N_n\}$. We use a connection matrix M to represent the network structure. M is an n by n matrix with its element e_{ij}. Each e_{ij} is a Boolean variable valued in {0,1}. The expression $e_{ij} = 1$ means that there is an edge from N_j to N_i, while $e_{ij} = 0$ means there is no edge from N_j to N_i. M is an anti-symmetric matrix. Before going into the details of the data construction procedure, we introduce the term *level* to describe the role of an attribute in the network structure.

The root *level*, denoted by V_0 , is defined as

$$V_0 = \{i | e_{ij} = 0, \forall j = 1, \ldots, n.\} \tag{15}$$

Usually, we call the attributes belonging to V_0 as source nodes. The first *level*, V_1, is defined as

$$V_1 = \{i | \{j | e_{ij} = 1\} \subset V_0; i \notin V_0.\} \tag{16}$$

Generally, the kth *level*, V_k, is iteratively defined as

$$V_k = \{i | \{j | e_{ij} = 1\} \subset \bigcup_{p=0}^{k-1} V_p; i \notin \bigcup_{p=0}^{k-1} V_p\} \tag{17}$$

for $k > 0$ until $k = K$ for which the constraint $\bigcup_{p=0}^{K} V_p = \{1, \cdots, n\}$ is satisfied. For each N_i, $i \in \bigcup_{p=1}^{k} V_p$, there is a corresponding network element NE_i with child N_i and parent set $\Pi_{N_i} = \{N_j | e_{ij} = 1\}$. The data construction procedure is detailed as follows:

1. Select a network structure represented by a connection matrix M. Determine the number of records L.

2. Use a random number generator to create the data for source nodes independently. Each of them is uniformly distributed on [0,1].

3. Set $p = 1$.

4. While p is not greater than K, for each NE_i whose $N_i \in V_p$, do the followings:

   i. Let N_i be the objective attribute y, and Π_{N_i} be the predictive attributes $X = \{x_1, \cdots, x_m\}$, where m is the cardinality of Π_{N_i}.

   ii. Assign the values of μ on $\mathcal{P}(X)$ and the respective regression coefficients c and q.

   iii. Calculate the values of the objective attribute y, $Y_j, j = 1, 2, \ldots, L$, according to the expression of the nonlinear multiregression in Equation (6).

   iv. Let Y_j be the value of attribute N_i in the data set.

   v. Add 1 to p.

5.2 Experimental Results

The data set used in the first experiment is derived from a nonlinear multiregression network with 8 attributes, $N = \{N_1, N_2, \ldots, N_8\}$. The network structure is shown in Figure 8 and its corresponding connection matrix is

$$M = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \end{bmatrix}$$

It is composed of 3 network elements and 8 edges. 50 records are generated according to the data construction procedure described above. Then we employ our learning method to discover the network structure as well as estimate 20 coefficients of all network elements. The parameters in the MRCEP-FDOA algorithm are set as follows: the population size $PS = 50$, the terminating number of generations $G = 200$, and the parameter $p = 7$. The probabilities of applying simple mutation, reversion, move, and knowledge-guided mutation are 0.25 each. Ten trials are performed with different seeds for the data set.

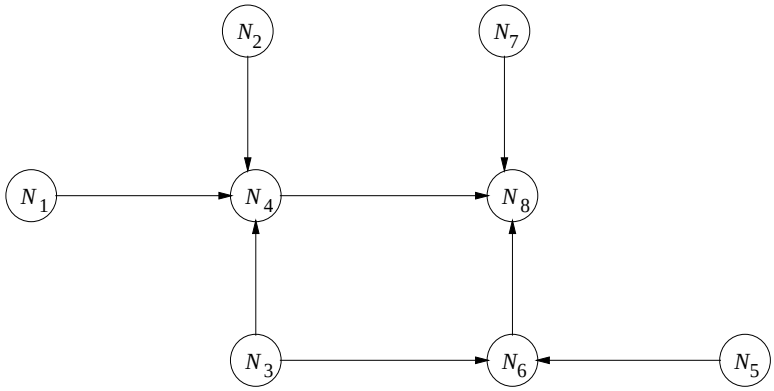

Figure 8: A nonlinear multiregression network with 8 attributes and 8 edges

The experiment results show that all ten trials are able to discover the original network structure. The cost metric of the learned structure is 3.178. Figure 9 depicts the lowest cost metric in each generation during the learning process. It shows that the quality of the learned network increases as the algorithm progresses. When the algorithm terminates, the estimated multiregression coefficients of the three network elements are also obtained. Table 4 depicts the actual coefficients and the estimated solutions. We can see that the estimated values are very close to the actual ones. Moreover, as the population size is 50 and the terminating number of generations is 200, at most 10050 different network structures are generated. Compare this number with the size of the search space which is 7.837e11, it can be concluded that the solution can be found by only exploring a small portion of the search space (10050/7.837e11 = 1.2824e-8).

The second set of experiments is concerned with a more complex nonlinear multire-

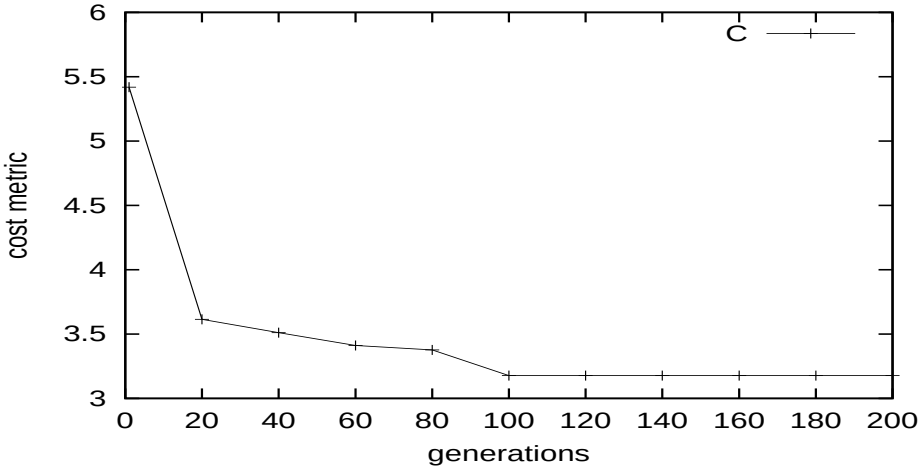

Figure 9: The lowest cost metric versus generations during the learning process for the 8-attribute network

NE_4			NE_6			NE_8		
	preset	estimated		preset	estimated		preset	estimated
c	0.075439	0.075446	c	0.440912	0.440909	c	0.762085	0.764012
q	0.569946	0.569949	q	0.348480	0.348483	q	0.950012	0.952366
Set	μ	μ^*	Set	μ	μ^*	Set	μ	μ^*
$\{\phi\}$	0.0	0.0	$\{\phi\}$	0.0	0.0	$\{\phi\}$	0.0	0.0
$\{N_1\}$	0.4589843	0.458946	$\{N_3\}$	0.104492	0.104505	$\{N_4\}$	0.381836	0.350345
$\{N_2\}$	0.894531	0.894493	$\{N_5\}$	0.151367	0.151378	$\{N_6\}$	0.321289	0.310680
$\{N_1, N_2\}$	0.961914	0.961894	$\{N_3, N_5\}$	1.0	1.0	$\{N_4, N_6\}$	0.941406	0.935823
$\{N_3\}$	0.667969	0.667941				$\{N_7\}$	0.470703	0.467758
$\{N_1, N_3\}$	0.843750	0.843721				$\{N_4, N_7\}$	0.752930	0.677577
$\{N_2, N_3\}$	0.934570	0.934544				$\{N_6, N_7\}$	0.877930	0.860434
$\{N_1, N_2, N_3\}$	1.0	1.0				$\{N4, N_6, N_7\}$	1.0	1.0

Table 4: The actual and estimated values of the regression coefficients for the 8-attribute network (NE: Network Element)

gression network with 14 attributes, 6 network elements and 19 edges. Figure 10 shows the original network structure. The number of data records L is 500. Setting $PS = 500$, $G = 500$, and other parameters similar to the first experiment, we employ our learning algorithm to discover the network structure and the regression coefficients. Ten trials are tested.

The experimental results show that most trials are able to discover the original network structure. The cost metric is 4.22. The lowest cost metric versus generation in the learning process is depicted in Figure 11. The actual and the estimated coefficients for each network element are listed in Table 5. From this table, we can see that even for such a complex network containing 60 coefficients, the estimated coefficients are very close to the actual ones. Furthermore, as the population size is 500 and the terminating number of generations is 500, at most 250500 different network structures are generated. Compare this number with the size of the search space which is 1.439e36, it can be observed that the solution can be discovered by only examining a very small fraction of the search space (250500/1.439e36 = 1.7408e-31). It indicates that our learning algorithm is very effective for discovering the network structure and the FDOA algorithm is very effective to estimate the nonlinear multiregression coefficients based on the Choquet integral.

Once the network structure and the multiregression coefficients of network elements are learned, we can conduct inference and prediction. The inference direction basically follows the direction of the edges, i.e., from parents to child. In each network element, a child attribute can be inferred from its parent set. Generally, an attribute may be a child for a network element, while it can also be a parent for one or several other network elements at the same time. For any network element NE_i, if the data of all predictive attributes are available in a new observation, we can predict the value of the objective attribute N_i directly by its predictive attributes. For example, N_{12} is the objective attribute in network element NE_{12}. From Figure 10, N_{12} can be predicted by N_2, N_4, N_9, and N_{10} directly. If the values of N_2, N_4, N_9, and N_{10} are available in a new observation, we can estimated the value of N_{12} by calculating the nonlinear multiregression with the estimated multiregression

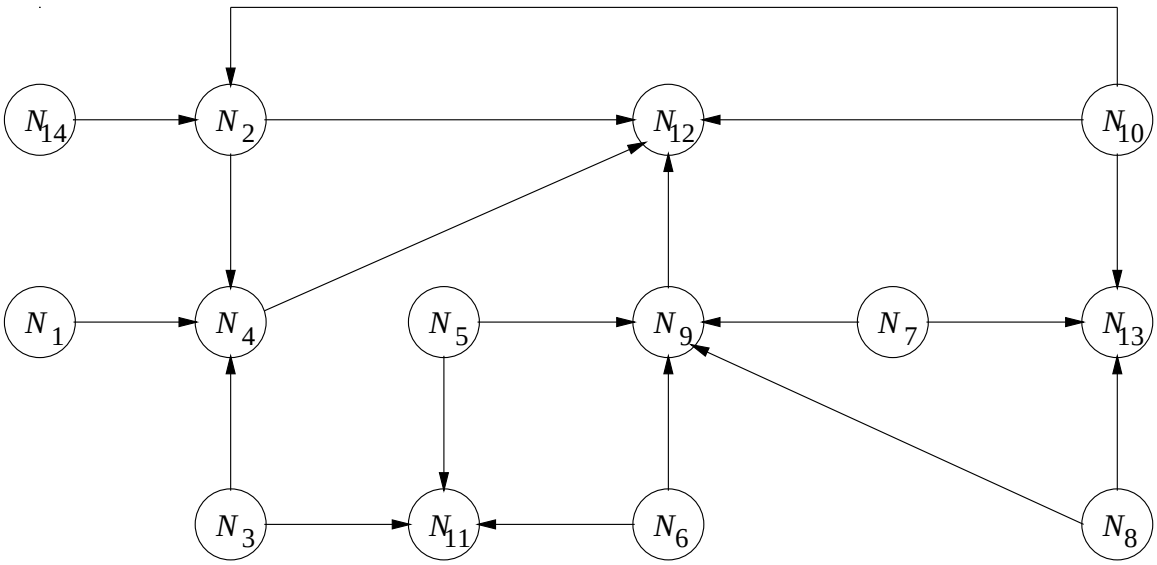

Figure 10: A nonlinear multiregression network with 14 attributes and 19 edges

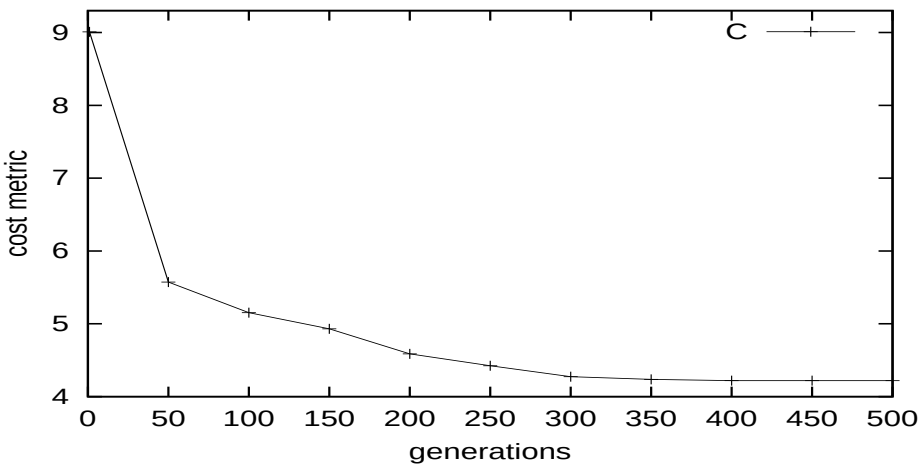

Figure 11: The lowest cost metric versus generations during the learning process for the 14-attribute network

NE_2			NE_4			NE_{11}		
	preset	estimated		preset	estimated		preset	estimated
c	0.3	0.300001	c	0.4	0.399996	c	2	2.000000
q	0.6	0.600001	q	0.7	0.700003	q	1.3	1.300000
Set	μ	μ^*	Set	μ	μ^*	Set	μ	μ^*
$\{\phi\}$	0.0	0.0	$\{\phi\}$	0.0	0.0	$\{\phi\}$	0.0	0.0
$\{N_{10}\}$	0.5	0.499995	$\{N_1\}$	0.1	0.100008	$\{N_3\}$	0.3	0.299997
$\{N_{14}\}$	0.8	0.799994	$\{N_2\}$	0.2	0.200011	$\{N_5\}$	0.5	0.499996
$\{N_{10}, N_{14}\}$	1.0	1.04	$\{N_1, N_2\}$	0.5	0.500006	$\{N_3, N_5\}$	0.5	0.499997
			$\{N_3\}$	0.3	0.300007	$\{N_6\}$	0.7	0.699997
			$\{N_1, N_3\}$	0.6	0.600009	$\{N_3, N_6\}$	0.8	0.799997
			$\{N_2, N_3\}$	0.8	0.800005	$\{N_5, N_6\}$	0.9	0.899997
			$\{N1, N_2, N_3\}$	1.0	1.0	$\{N3, N_5, N_6\}$	1.0	1.0

NE_{13}			NE_9			NE_{12}		
	preset	estimated		preset	estimated		preset	estimated
c	0.8	0.800001	c	0.1	0.0856692	c	5.0	4.985990
q	3.0	0.300000	q	0.8	0.796283	q	2.0	2.020000
Set	μ	μ^*	Set	μ	μ^*	Set	μ	μ^*
$\{\phi\}$	0.0	0.0	$\{\phi\}$	0.0	0.0	$\{\phi\}$	0.0	0.0
$\{N_7\}$	0.6	0.599999	$\{N_5\}$	0.5	0.532208	$\{N_2\}$	0.4	0.404843
$\{N_8\}$	0.6	0.599999	$\{N_6\}$	0.2	0.219235	$\{N_4\}$	0.25	0.262294
$\{N_7, N_8\}$	0.8	0.799999	$\{N_5, N_6\}$	0.6	0.63295	$\{N_2, N_4\}$	0.5	0.487397
$\{N_{10}\}$	0.1	0.099992	$\{N_7\}$	0.1	0.116373	$\{N_9\}$	0.5	0.531700
$\{N_7, N_{10}\}$	0.8	0.799999	$\{N_5, N_7\}$	0.7	0.737091	$\{N_2, N_9\}$	0.5	0.531676
$\{N_8, N_{10}\}$	0.9	0.899998	$\{N_6, N_7\}$	0.5	0.530009	$\{N_4, N_9\}$	0.6	0.622438
$\{N_7, N_8, N_{10}\}$	1.0	1.0	$\{N_5, N_6, N_7\}$	0.8	0.818365	$\{N_2, N_4, N_9\}$	0.6	0.622274
			$\{N_8\}$	0.3	0.332515	$\{N_{10}\}$	0.6	0.59513
			$\{N_5, N_8\}$	0.6	0.632677	$\{N_2, N_{10}\}$	0.7	0.697006
			$\{N_6, N_8\}$	0.6	0.64156	$\{N_4, N_{10}\}$	0.6	0.594931
			$\{N_5, N_6, N_8\}$	0.7	0.729668	$\{N_2, N_4, N_{10}\}$	0.8	0.801305
			$\{N_7, N_8\}$	0.6	0.633673	$\{N_9, N_{10}\}$	0.6	0.601174
			$\{N_5, N_7, N_8\}$	0.9	0.935828	$\{N_2, N_9, N_{10}\}$	0.9	0.869875
			$\{N_6, N_7, N_8\}$	0.8	0.815284	$\{N_4, N_9, N_{10}\}$	0.8	0.823851
			$\{N_5, N_6, N_7, N_8\}$	1.0	1.0	$\{N_2, N_4, N_9, N_{10}\}$	1.0	1.0

Table 5: The actual and estimated values of the regression coefficients for the 14-attribute network (NE: Network Element)

coefficients of the network element listed in the right bottom part of Table 5. This is an example of a direct prediction.

In some applications, not all parents observation are available for a child. For instance, in network element NE_{12}, in case the values of N_4 and N_2 are not available, the prediction for attribute N_{12} can be done via the observations of N_1, N_3, and N_{14} if they are available instead. From the network structure, the value of N_2 can be estimated by N_{10} and N_{14} first according to the estimated multiregression coefficients in the left top part of Table 5. N_4 can be estimated by N_1, N_2, and N_3 according to the estimated multiregression coefficients in center top part of Table 5. Then N_{12} can be predicted by the observation values of N_9 and N_{10} and the estimated values of N_2 and N_4. This is an example of an indirect prediction.

5.3 Study on Parameters

We have also conducted an extensive investigation on the performance of our learning algorithm by using different number of training records and varying the parameters in the algorithm. To study the effect of our learning framework on different sizes of data sets, we generate two more data sets of 100 and 300 records from the same 14-attribute network structure with similar multiregression coefficients. We then apply our algorithm on these three data sets of 100, 300, and 500 records respectively. Furthermore, we also vary the population sizes, PS (200 and 500), of MRCEP-FDOA and the number of generations, G (200, 500, and 1000). For each combination of L, PS, and G, 5 trials with different seeds are conducted. To study the performance, we collected the average cost metric AC, the averaged structure difference ASD, and the average running time (in seconds) AT, for each combination. Here the structure difference, SD, is defined as $\sum_{i=1}^{n} \delta_i$, where δ_i is the cardinality of the symmetric difference of parent sets in the discovered network and the original network for attribute N_i. The experiment results are summarized in Table 6. From the table, we can make the following observations:

			data set size, L		
PS	G		100	300	500
200	200	AC	4.69503	4.52189	4.73370
		ASD	9	8.6	9.8
		AT	1954	5766	9600
	500	AC	4.21382	4.28310	4.23535
		ASD	1.2	1.2	0.6
		AT	2689	8615	14533
	1000	AC	4.17312	426319	4.22072
		ASD	0	0	0
		AT	3432	11213	18706
500	200	AC	4.596212	4.55986	4.56675
		ASD	8	7.8	6.2
		AT	3770	12094	18690
	500	AC	4.18419	4.21823	4.22677
		ASD	1	0.4	0.2
		AT	4739	14478	22350
	1000	AC	4.1732	4.17312	4.22072
		ASD	0	0	0
		AT	5489	17027	25565

Table 6: Learning Performance from the data set with 100, 300, and 500 records under different combination of parameters

1. ASD generally decreases with the size of data sets. It implies that the quality of learning improves if the size of data set increases.

2. The running time is approximately proportional to the size of the training data.

3. Generally speaking, the algorithm can obtain the best result (SD=0) when the number of generations G is set to be 1000.

6 Conclusions

A novel multiregression network model to represent nonlinear relationships among continuous attributes has been proposed. Our network-based structure provides an effective

reasoning procedure to perform prediction and decision making. Unlike many existing paradigms based on linear models, the attribute relationship in our framework is represented by nonlinear nonnegative multiregressions based on the Choquet integral. This kind of multiregression is able to model a rich set of nonlinear attribute interactions directly. We have developed a Fast Double Optimization Algorithm (FDOA) for learning the multiregression coefficients of a single network element. Using this local learning component, we propose a global learning algorithm, called MRCEP-FDOA for discovering the network structures from databases. Several effective genetic operators have been designed for this learning task. We have conducted a series of experiments to assess the effectiveness of our algorithm and investigate the performance under different parameter combinations as well as the sizes of the training data sets. The empirical results demonstrate that our framework can successfully discover the target network structures and the regression coefficients.

Appendix

The data set used in the first experiment

N_1	N_2	N_3	N_4	N_5	N_6	N_7	N_8
0.002	0.967	0.787	0.586	0.668	0.678	0.871	1.482
0.876	0.629	0.254	0.490	0.935	0.565	0.872	1.428
0.374	0.485	0.893	0.503	0.963	0.756	0.951	1.538
0.746	0.350	0.640	0.442	0.844	0.675	0.177	1.238
0.844	0.080	0.558	0.426	0.459	0.604	0.471	1.245
0.959	0.540	0.375	0.489	0.895	0.599	0.087	1.238
0.730	0.826	0.582	0.537	0.075	0.486	0.016	1.216
0.539	0.610	0.383	0.416	0.570	0.584	0.604	1.306
0.838	0.916	0.011	0.575	0.964	0.495	0.104	1.240
0.996	0.767	0.636	0.570	0.696	0.666	0.151	1.309
0.156	0.180	0.935	0.464	0.610	0.665	0.841	1.449
0.864	0.737	0.914	0.576	0.473	0.622	0.348	1.310
0.413	0.099	0.673	0.382	0.766	0.680	0.379	1.216
0.533	0.147	0.835	0.460	0.658	0.677	0.273	1.255
0.013	0.401	0.942	0.496	0.919	0.762	0.196	1.298
0.131	0.568	0.601	0.395	0.063	0.482	0.428	1.182
0.365	0.409	0.906	0.496	0.134	0.516	0.786	1.371
0.614	0.964	0.981	0.618	0.075	0.500	0.871	1.435
0.371	0.256	0.784	0.434	0.316	0.568	0.664	1.329
0.038	0.432	0.791	0.444	0.318	0.569	0.075	1.201
0.382	0.467	0.521	0.359	0.878	0.641	0.175	1.179
0.335	0.019	0.591	0.336	0.780	0.657	0.261	1.175
0.952	0.853	0.561	0.581	1.000	0.660	0.205	1.317
0.861	0.988	0.727	0.628	0.037	0.479	0.189	1.255
0.792	0.856	0.646	0.556	0.916	0.680	0.022	1.299
0.728	0.396	0.206	0.384	0.578	0.532	0.976	1.449
0.133	0.954	0.371	0.575	0.393	0.571	0.825	1.419
0.582	0.404	0.454	0.363	0.945	0.625	0.432	1.224
0.822	0.198	0.638	0.448	0.580	0.645	0.776	1.411
0.096	0.344	0.786	0.430	0.436	0.606	0.862	1.432
0.533	0.960	0.749	0.602	0.183	0.525	0.912	1.454
0.243	0.541	0.616	0.401	0.709	0.660	0.374	1.221
0.151	0.232	0.312	0.235	0.207	0.517	0.053	1.061
0.492	0.176	0.138	0.257	0.860	0.527	0.616	1.271
0.221	0.833	0.012	0.509	0.832	0.488	0.238	1.219
0.446	0.296	0.553	0.357	0.510	0.620	0.584	1.302
0.034	0.416	0.280	0.295	0.024	0.458	0.963	1.404
0.473	0.593	0.032	0.397	0.336	0.468	0.927	1.404
0.079	0.757	0.532	0.476	0.860	0.644	0.186	1.249
0.404	0.753	0.601	0.488	0.348	0.571	0.099	1.229
0.775	0.079	0.671	0.432	0.881	0.686	0.361	1.246
0.794	0.338	0.163	0.384	0.276	0.504	0.852	1.383
0.334	0.802	0.474	0.508	0.652	0.615	0.211	1.261
0.445	0.315	0.008	0.282	0.711	0.481	0.144	1.083
0.840	0.673	0.018	0.488	0.851	0.491	0.286	1.216
0.199	0.582	0.618	0.407	0.336	0.568	0.621	1.307
0.820	0.383	0.029	0.400	0.107	0.455	0.086	1.142
0.379	0.455	0.359	0.330	0.790	0.589	0.537	1.264
0.386	0.133	0.591	0.351	0.226	0.533	0.388	1.171
0.965	0.958	0.996	0.636	0.555	0.650	0.487	1.363

Acknowledgment

The work described in this paper was partially supported by grants from the Research Grant Council of the Hong Kong Special Administrative Region. Projects: CUHK 4212/01E, CUHK 4187/01E, and LU 3012/01E. It was also partially supported by the Direct Grant Project No. 2050179 of the Engineering Faculty in the Chinese University of Hong Kong.

References

[1] P. Angeline. *Evolutionary Algorithms and Emergent Intelligence.* Ph.D. Thesis, The Ohio State University, 1993.

[2] P. Angeline. Genetic programming and emergent intelligent. In *Advances in Genetic Programming*, editor K. E. Kinnear, Jr., Cambridge MA: MIT Press, pp. 75-97, 1994.

[3] T. Bäck. *Evolutionary Algorithms in Theory and Practice : Evolution strategies, Evolutionary Programming, Genetic algorithms.* New York NY: Oxford University Press, 1996.

[4] T. Bäck, F. Hoffmeister and H. P. Schwefel. Evolutionary Computation: Comments on the History and Current State. *IEEE Transactions on Evolutionary Computation*, 1(1), pp. 3-17, 1997.

[5] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms.* Cambridge MA: MIT Press, 1990.

[6] U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors. *Advances in Knowledge Discovery and Data Mining.* Menlo Park CA: AAAI Press, 1996.

[7] D. B. Fogel. *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence, second edition.* New York, NY: IEEE Press, 2000.

[8] D. B. Fogel. An introduction to simulated evolutionary optimization. *IEEE Transactions on Neural Networks*, 5, pp. 3-14, 1994.

[9] D. B. Fogel and K. Chellapilla. Revisiting Evolutionary Programming. *Proc. AeroSense'98: Aerospace/Defense Sensing and Controls*, Orlando, 1998.

[10] L. Fogel, A. Owens, and M. Walsh. *Artificial Intelligence through Simulated Evolution.* New York: John Wiley and Sons, 1966.

[11] N. Friedman and I. Nachman, Gaussian process networks. In *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, pp. 211-219, 2000.

[12] J. H. Friedman and W. Stuetzle, Projection pursuit regression, *Ann. Statist.*, 76, pp. 817-823, 1981.

[13] D. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning.* Reading MA: Addison-Wesley, 1989.

[14] M. Grabisch. A new algorithm for identifying fuzzy measures and its application to pattern recognition. *Proc. FUZZY-IEEE/IFES'95*, Yokohama, pp.145-150, 1995.

[15] M. Grabisch. K-older-additive discrete fuzzy measures and their representation. *Fuzzy Sets and Systems*, 92, pp. 167-189, 1997.

[16] M. Grabisch and M. Roubens. An axiomatic approach to the concept of interaction among players in co-operative games. *International Journal of Game Theory*, 28, pp. 547-565, 1999.

[17] P. R. Halmos. *Measure Theory.* New York: Van Nostrand, 1967.

[18] J. Holland. *Adaptation in Natural and Artificial Systems.* Cambridge MA: MIT Press, 1992.

[19] F. V. Jensen, *An Introduction to Bayesian Networks.* University College London Press, 1996.

[20] S. H. Kwon and M. Sugeno. A hierarchical subjective evaluation model using non-monotonic measures and the Choquet integral. In *Fuzzy Measures and Integrals: Theory and Applications*, editors M. Grabisch, T. Murofushi, and M. Sugeno, Springer Verlag, pp. 375-391, 2000.

[21] J. R. Koza. *Genetic Programming: on the Programming of Computers by Means of Natural Selection*. Cambridge MA: MIT Press, 1992.

[22] J. R. Koza. *Genetic Programming II: Automatic Discovery of Reusable Programs*. Cambridge MA: MIT Press, 1994.

[23] J. R. Koza, F. H. Bennett III, D. Andre, and M. A. Keane. *Genetic Programming III: Darwinian Invention and Problem Solving*. San Francisco CA: Morgan Kaufmann, 1999.

[24] W. Lam. Bayesian network refinement via machine learning approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(3), pp. 240-251, 1998.

[25] P. Larrañaga, M. Poza, Y. Yurramendi, R. Murga, and C. Kuijpers. Structure Learning of Bayesian Network by Genetic Algorithms: A Performance Analysis of Control Parameters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(9), pp. 912-926, 1996.

[26] O. C. Lingjarde and K. Liestol, Generalized projection pursuit regression, *SIAM J. Sci. Comput*, 20(3), pp. 844-857, 1998.

[27] Mori and T. Murofushi. An analysis of evaluation model using fuzzy measure and the Choquet integral. *5th Fuzzy System Symposium*, Kobe, Japan, (in Japanese), June 1989.

[28] T. Murofushi, M. Sugeno and M. Machida. Non-monotonic fuzzy measures and the Choquet integral. *Fuzzy Sets and Systems*, 64, pp. 73-86, 1994.

[29] T. Murofushi and M. Sugeno. A theory of fuzzy measures: representations, the Choquet integral, and null sets. *Journal of Mathematical Analysis and applications*, 159, pp. 532-549, 1991.

[30] T. Murofushi and M. Sugeno. Fuzzy measures and integrals. In *Fuzzy measures and integrals - Theory and Applications*, M. Grabisch, T. Murofushi, and M. Sugeno (eds.), Physical Verlag, 2000.

[31] J. Pearl, *Probabilistic Reasoning in Intelligent Systems : Networks of Plausible Inference*. Morgan Kaufmann, 1988.

[32] V. W. Porto. Evolutionary Programming. In *Evolutionary Computation 1: Basic Algorithms and Operators*, editors T. Bäck, D. B. Fogel, and Z. Michalwicz, Institute of Physics Publishing, pp. 89-102, 2000.

[33] R. W. Robinson. Counting Unlabeled Acyclic Digraphs. In C. H. C. Little, ed., *Lectures Notes in Mathematics 622: Combinatorial Mathematics V*, pp. 28-43, New York NY: Springer-Verlag, 1977.

[34] H. P. Schewefel. *Numerical Optimization of Computer Models*. New York: John Wiley and sons, 1981.

[35] Z. Wang and G. J. Klir. *Fuzzy Measure Theory*. New York: Plenum, 1992.

[36] Z. Wang and G. J. Klir. Choquet integrals and natural extensions of lower probabilities. *International Journal of Approximate Reasoning*, 16, pp. 137-147, 1997.

[37] Z. Wang, K.S. Leung and J. Wang. A genetic algorithms for determining nonadditive set functions in information fusion. *Fuzzy Sets and Systems*, 102, pp. 463-469, 1999.

[38] Z. Wang, K.S. Leung, M.L. Wong, J. Fang, and K. Xu. Nonlinear nonnegative multiregressions based on Choquet integrals. *International Journal of Approximate Reasoning*, 25, pp. 71-87, 2000.

[39] Z. Wang, K.S. Leung, and K. Xu. A new nonlinear regression model used for multisource-multisensor data fusion: an application of nonlinear integrals and genetic algorithms. *FUSION '98*, pp. 299-306, 1998.

[40] M.L. Wong, W. Lam, and K.S. Leung. Using evolutionary programming and minimum description length principle for data mining of Bayesian networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(2), pp.174-178, 1999

[41] K. Xu, Z. Wang, K.S. Leung. Using a new type of nonlinear integral for multiregression: an application of evolutionary algorithms in data mining. *Proc. of 1998 IEEE International Conference on System, Man and Cybernetics*, pp. 2326-2331, 1998.