

Chapter III

Generalization Data Mining in Fuzzy Object-Oriented Databases

Rafal Angryk, Tulane University, USA

Roy Ladner, Stennis Space Center, USA

Frederick E. Petry, Tulane University & Stennis Space Center, USA

Abstract

In this chapter, we consider the application of generalization-based data mining to fuzzy similarity-based object-oriented databases (OODBs). Attribute generalization algorithms have been most commonly applied to relational databases, and we extend these approaches. A key aspect of generalization data mining is the use of a concept hierarchy. The objects of the database are generalized by replacing specific attribute values by the next higher-level term in the hierarchy. This will then eventually result in generalizations that represent a summarization of the information in the database. We focus on the generalization of similarity-based simple fuzzy attributes for an OODB using approaches to the fuzzy concept hierarchy

developed from the given similarity relation of the database. Then consideration is given to applying this approach to complex structure-valued data in the fuzzy OODB.

Introduction

Data mining and knowledge discovery have increasing importance as the amount of data from various sources has rapidly increased. Awash in such volumes of data, data mining techniques attempt to make sense of this data by formulating information of value for decision making. This can vary from deciding on commercial sales promotions to environmental planning to national security decisions. Much of the current work is in the context of conventional relational databases. In this chapter, we will discuss how to apply one valuable data mining approach — attribute-oriented generalization — to a similarity-based fuzzy OODB.

Background

In this section, we survey the general area of data mining, discuss some of the relevant work in fuzzy data mining, and then describe the specific technique of attribute-oriented induction for generalization, which is the focus of this chapter. Additionally, we describe the fuzzy object-oriented model based on similarity relationships that is the context in which we investigate data generalization.

Data Mining

Data mining or knowledge discovery generally refers to a variety of techniques that have developed in the fields of databases, machine learning, and pattern recognition. The intent is to uncover useful patterns and associations from large databases.

Although we are primarily interested here in specific algorithms for knowledge discovery, we will first review the overall process of data mining (Feelders, Daniels, & Holsheimer, 2000). The initial steps of data mining are concerned with preparation of data, including data cleaning intended to resolve errors and missing data and integration of data from multiple heterogeneous sources. Next are the steps needed to prepare for actual data mining. These include selection

of the specific data relevant to the task and transformation of this data into a format required by the data mining approach. These steps are sometimes considered to be those in the development of a data warehouse, i.e., an organized format of data available for various data mining tools. There is a wide variety of specific knowledge discovery algorithms that were developed (Han & Kamber, 2000). These discover patterns that can then be evaluated based on some “interestingness” measure used to prune the huge number of available patterns. Finally, as true for any decision aid system, an effective user interface with visualization and alternative representations must be developed for presentation of the discovered knowledge.

Specific data mining algorithms can be considered as belonging to two categories: descriptive and predictive data mining. In the descriptive category are class description, association rules, and classification. Class description can provide characterization or generalization of data or comparisons between data classes to provide class discriminations. Data generalization is a process of grouping data, enabling transformation of similar item sets, stored originally in a database at the low (primitive) level, into more abstract conceptual representations. This process is a fundamental element of attribute-oriented induction, a descriptive database mining technique, allowing compression of the original data set into a generalized relation, which provides concise and summarative information about the massive set of task-relevant data.

Association rules correspond to correlations among the data items (Agrawal, Imielinski, & Swami, 1993). They are often expressed in rule form, showing attribute-value conditions that commonly occur at the same time in some set of data. An association rule of the form $X \rightarrow Y$ can be interpreted as meaning that the tuples in the database that satisfy the condition X also are “likely” to satisfy Y , so that the “likely” implies this is not a functional dependency in the formal database sense. Finally, a classification approach analyzes the training data (data with known class membership) and constructs a model for each class based on the features in the data. Commonly, the outputs generated are decision trees or sets of classification rules. These can be used for the characterization of the classes of existing data and to allow the classification of data in the future, and so can also be considered predictive.

Predictive analysis is also a very developed area of data mining. One common approach is clustering. Clustering analysis identifies the collections of data objects that are similar to each other. The similarity metric is often a distance function given by experts or appropriate users. A good clustering method produces high-quality clusters to yield low intercluster similarity and high intracluster similarity. Prediction techniques are used to predict possible missing data values or distributions of values of some attributes in a set of objects. First, one must find the set of attributes relevant to the attribute of interest and then

predict a distribution of values based on the set of data similar to the selected objects. A large variety of techniques is used, including regression analysis, correlation analysis, genetic algorithms, and neural networks, to mention a few.

Finally, a particular case of predictive analysis is time-series analysis. This technique considers a large set of time-based data to discover regularities and interesting characteristics. One can search for similar sequences or subsequences, then mine sequential patterns, periodicities, trends, and deviations.

Fuzzy Data Mining

An early and continuing significant application of fuzzy sets has been in pattern recognition, especially fuzzy clustering algorithms (Bezdek, 1974). Hence, much of the effort in fuzzy data mining has been made by using fuzzy clustering and fuzzy set approaches in neural networks and genetic algorithms (Hirota & Pedrycz, 1999). In fuzzy set theory, an important consideration is the treatment of data from a linguistic viewpoint. From this, an approach was developed that uses linguistically quantified propositions to summarize the content of a database by providing a general characterization of the analyzed data (Yager, 1991; Kacprzyk, 1999; Dubois & Prade, 2000; Feng & Dillon, 2003). Fuzzy gradual rules for data summarization were also considered (Cubero et al., 1999). A common organization of data for data mining is the multidimensional data cube in data warehouse structures. Treating the data cube as a fuzzy object has provided another approach for knowledge discovery (Laurent et al., 2000).

Fuzzy data mining for generating association rules was considered by a number of researchers. There are approaches using the set-oriented mining (SETM) algorithm (Shu et al., 2001) and other techniques (Bosc & Pivert, 2001), but most have been based on the Apriori algorithm (Delgado et al., 2003). Extensions included fuzzy set approaches to quantitative data (Zhang, 1999; Kuok et al., 1998), hierarchies or taxonomies (Chen et al., 2000; Lee, 2001), weighted rules (Gyenesei, 2001), and interestingness measures (de Graaf et al., 2001; Gyenesei, 2001; Au & Chan, 2003).

Generalization Data Mining

The basis of a generalization data mining approach rests on three aspects (Han & Kamber, 2000): the set of data relevant to a given data mining task; the expected form of knowledge to be discovered; and the background knowledge, which usually supports the whole process of knowledge acquisition. Generalization of data is typically performed with utilization of concept hierarchies, which

in ordinary databases are considered to be part of background knowledge, and are indispensable for the process.

Despite the progress in research on data mining algorithms, the phase of data generalization remains a crucial activity. The choice of data to be analyzed as well as of the concepts for its generalization has a fundamental influence on retrieved results, regardless of applied knowledge acquisition techniques. Although certain dependencies among data can be discovered at the primitive concept level, much stronger and often far more interesting dependencies can be determined at a higher concept level. With data generalization executed at the initial stage of data mining, the process of knowledge extraction can be more effective and bring concise results directly at the abstraction level desired by a user. Moreover, many relations occurring at the lower level may not match the requirement of minimum support assigned by data analysts to eliminate infrequent regularities, whereas after summarization via generalization they may occur often enough to have significant meaning.

The idea of using concept hierarchies for attribute-oriented induction in data mining was investigated by several research groups (Han et al., 1992; Han, 1995; Carter & Hamilton, 1998; Hilderman et al., 1999). Generalization of database objects is performed on an attribute-by-attribute basis, applying a separate concept hierarchy for each of the generalized attributes included in the relation of task-relevant data.

The basic steps and guidelines for attribute-oriented generalization in an OODB are summarized below (Han, Nishio, & Kawano, 1994):

1. An initial query to the fuzzy OODB with a given similarity threshold provides the starting generalization class G_0 , which contains the set of data that is relevant to the user's generalization interest.
2. Generalization should be performed on the smallest decomposable components (or attributes) of the data objects in each generalization class G_i .
3. If there is a large set of distinct values for an attribute but there is no higher-level concept provided for the attribute, the attribute should be removed in the generalization process.
4. If there a higher-level concept exists in the concept tree for an attribute value of an object, the substitution of the value by its higher-level concept generalizes the object. Minimal generalization should be enforced by ascending the tree one level at a time.
5. Two generalized objects may become similar enough to be merged (see the next section for merging of objects in a fuzzy OODB). So we include an added attribute, *count*, to keep track of how many objects were merged to form the current generalized object. The value of the count of an object should be carried to its generalized object, and the counts should be accumulated when merging identical objects in generalization.

6. The generalization is controlled by providing levels that specify how far the process should proceed. If the number of distinct values of an attribute in the given class is larger than the generalization threshold value, further generalization on this attribute should be performed. If the number of objects of a generalized class is larger than the generalization threshold value, the generalization should proceed further.

Attribute generalization should not be mistaken for simple record summarization. Summaries of data usually have a more simplified character and tend to omit data that do not occur originally in large quantities in order to simplify the final report. Gradual generalization through concept hierarchies allows, in contrast, detailed tracking of all data objects and can lead to the discovery of interesting patterns among data at the lowest possible abstraction level of their occurrence, decreasing, at the same time, the risk of omitting them due to overgeneralization. The appropriate attribute-oriented generalization allows extraction of knowledge on a specific abstraction level but without omitting even rare attribute values. It might occur that such atypical values, despite being initially (at a low level of the generalization hierarchy) infrequent, can sum up to impressive cardinalities when generalized to an efficiently high abstraction level, which can then sometimes strongly influence the suspected proportions among the original data.

Depending on the approach and the intention of data analysts, generalization of collected data can be treated as a final step of data mining (e.g., summary tables are presented to users, allowing them to interpret overall information) or as an introduction to further knowledge extraction (e.g., extraction of abstract association rules directly from the generalized data).

Fuzzy Object-Oriented Model

The OODB model and object-oriented programming languages arose out of the necessity of dealing with the complexity of large software systems. Object-oriented systems view the universe as consisting of objects and try to model the interaction between objects. The object-oriented model is characterized by its properties of abstraction, encapsulation, modularity, hierarchy, typing concurrency, and persistence.

The object-oriented model is a natural successor to record-based models with explicit mechanisms to overcome their disadvantages (Bertino & Martino, 1991). The object-oriented data model (OODM) models composite objects, thereby capturing the IS-PART-OF concept, and relationships directly. Data are organized into classes, and classes are organized into an inheritance hierarchy. This

methodology is useful in capturing similarities among classes and data and abstracting them to higher levels.

An object is completely specified by its identity, behavior, and state. The state of an object consists of the values of its attributes. Its behavior is specified by the set of methods that operate on the state. An object identifier maintains the identity of an object, thereby distinguishing it from all others. The use of object identifiers permits three different types of object equality (Khoshafian & Copeland, 1986):

1. Identity (**=**): The identity predicate corresponds to the equality of references or pointers in conventional languages.
2. Shallow equality (**se**): Two objects are shallow equal if their states or contents are identical, i.e., corresponding instance variables need not be the same object, contents must be identical objects.
3. Deep equality (**de**): This ignores object identities and checks whether two objects are instances of the same class (i.e., same structure or type) and whether the values of the corresponding base objects are the same.

It is clear that identity is stronger than shallow equality, and shallow equality is stronger than deep equality. If identity holds, the same can be said of shallow and deep equality; if shallow equality holds, so does deep equality.

The most powerful aspect of an OODM is its ability to model inheritance. A class may inherit all the methods and attributes of its superclass. When a class inherits from one superclass, this is known as single inheritance. The situation in which a class inherits from more than one superclass is called multiple inheritance, and the inheritance structure forms a lattice. The class–subclass relationships form a class hierarchy similar to a generalization–specialization relationship. Another hierarchy that may originate at an attribute is the class composition hierarchy (Kim, 1989). The class composition hierarchy is distinct and orthogonal to the class hierarchy.

A Fuzzy Class Hierarchy

In this approach (George, Buckles, & Petry, 1993), two levels of imprecision may be represented: first, the impreciseness of object membership in class values (fuzzy class extents); and second, the fuzziness of object attribute values. The class composition schema is enhanced to incorporate the similarities between object instances, and the effects of the merge operator on class memberships were considered.

A class is characterized by structure, methods and extension so a class is a pair $C_i = (t_i, \text{ext}(t_i))$, where t is a type. Next, C_i is a subclass of C_i' ($C_i \subseteq_s C_i'$) iff:

1. The structure of C_i' is less equally defined (more general) in comparison to C_i .
2. A class possesses every method owned by its superclasses, though the methods may be refined in the class.

A class hierarchy models class-subclass relationships and may be represented as:

$$C_i \subseteq_s C_{i+1} \subseteq_s \dots \subseteq_s C_n$$

where C_n represents the root (basic) class, and C_i is the most refined (leaf) class. Analysis of class-subclass relations indicates that they can be broadly divided into two different types:

1. Specialization subclasses (also referred to as partial subclass or object-oriented subclass), where the subclass is a specialization of its immediate superclass, i.e., computer science is a specialization of engineering.
2. Subclasses that are subsets of its immediate superclass, i.e., the class of employees is a subset subclass of the class of persons.

A fuzzy hierarchy exists whenever it is judged subjectively that a subclass or instance is not a full member of its immediate class. Consideration of a fuzzy representation of the class hierarchy should take into account the different requirements and characteristics of the class-subclass relations. We associate with a subclass a grade of membership in its immediate class $C_i \subseteq_s C_{i+1}$, represented as $\mu_{C_i}(C_{i+1})$. A subclass is represented now by a pair $(C_i, \mu(C_{i+1}))$, the second element of which represents the membership of C_i in its immediate class C_{i+1} . The class hierarchy is now:

$$(o_i, \mu(C_i)) \subseteq_s (C_i, \mu(C_{i+1})) \subseteq_s (C_{i+1}, \mu(C_{i+2})) \subseteq_s \dots \subseteq_s (C_n, \mu(C_{n+1}))$$

The nature of class-subclass relationships also depends on the type of ISA links existing between the two. It is possible to have strong and weak ISA relationships between a class and its subclass. In a weak ISA relationship, the membership of a class in its superclasses is monotonically nonincreasing, while for the strong ISA link, the membership is nondecreasing. A fuzzy hierarchy possesses the following properties:

1. Membership of an instance/subclass in any of the superclasses in its hierarchy is constant, monotonically nonincreasing, or monotonically nondecreasing. If the membership is constant, the hierarchy is a subset hierarchy; if nonincreasing, a weak ISA specialization hierarchy; and if nondecreasing, a strong ISA specialization hierarchy
2. For a weak ISA specialization hierarchy and a strong ISA specialization hierarchy:

$$\mu_{C_i}(C_n) = f(\mu_{C_i}(C_{i+1}), \mu_{C_{i+1}}(C_{i+2}), \dots, \mu_{C_{n-1}}(C_n)).$$

The function f , which is application dependent, may be a product, min, max, etc.

3. For two objects o and o' such that $o, o' \in \text{ext}(C_i)$, if o **de** o' or o **se** o' , then $\mu_o(C_i) = \mu_{o'}(C_i)$. In other words, two objects have the same membership in a class (and all its superclasses) if they are value equal.

We prescribed a fuzzy hierarchy in which each instance/subclass is described as a member in its immediate superclass with a degree of membership. And, we described the membership of an instance in a class as function of the membership of the instance in the immediate classes that lie between the instance and the class of interest. However, this may not be possible because the hierarchies are not always “pure” and mixed hierarchies are more the rule. In some applications, it might be necessary to assume that the membership of an object (class) in its class (superclass) is list directed.

Thus, the expression for the class hierarchy can be generalized to account for the different types of links that can exist within an object hierarchy:

$$(o_i, \{ \mu(C_i), \mu(C_{i+1}), \dots, \mu(C_n) \}) \subseteq^s (C_i, \{ \mu(C_{i+1}), \mu(C_{i+2}), \dots, \mu(C_n) \}) \subseteq^s \dots \subseteq^s (C_n, \mu(C_{n+1}))$$

Fuzzy Class Schema

The OODM permits data to be viewed at different levels of abstraction based on the semantics of the data and their interrelationships. By extending the model to incorporate fuzzy and imprecise data, we allow data in a given class to be viewed through another layer of abstraction, this time one based on data values. This ability of the data model to chunk information further enhances its utility. In developing the fuzzy class schema, the merge operator is defined, which

combines two object instances of a class into a single object instance, provided predefined level values are achieved. The merge operator at the same time maintains the membership relationship existing between the object/class and its class/superclass.

Assume for generality two object members of a given class C_i with list-directed class/superclass memberships:

$$o = (i, \langle a_{k1}:i_{k1}, a_{k2}:i_{k2}, \dots, a_{km}:i_{km} \rangle, \langle \mu_o(C_i), \mu_o(C_{i+1}), \dots, \mu_o(C_n) \rangle)$$

$$o' = (i', \langle a_{k1}:i_{k1}2, a_{k2}:i_{k2}2, \dots, a_{km}:i_{km}2 \rangle, \langle \mu_o'(C_i), \mu_o'(C_{i+1}), \dots, \mu_o'(C_n) \rangle)$$

So o is a fuzzy object in C_i if $o \in \text{ext}(C_i)$ and $\mu_o(C_i)$ takes values in the range $[0,1]$.

Now we must consider how the data values as described by similarity relations behave (Petry, 1996). Assume attribute a_{kj} of class C_i with a noncomposite domain D_j . By definition of fuzzy object, the domain of a_{kj} is $d_{kj} \subseteq D_j$. So the similarity threshold of D_j is:

$$\text{Thresh}(D_j) = \min \{ \min_{x,y \in d_{jk}} [s(x,y)] \}$$

where $o \in \text{ext}(C_i)$ and x, y are atomic elements.

The threshold of a composite object is undefined. A composite domain is constituted of simple domains (at some level), each of which has a threshold value, i.e., the threshold for a composite object is a vector. The threshold value represents the minimum similarity of the values an object attribute may take. If the attribute domain is strictly atomic for all objects of the class (i.e., cardinality of a_{ij} is 1), then the threshold = 1. As the threshold value ranges toward 0, larger chunks of information are grouped together, and the information conveyed about the particular attribute of the class decreases. A level value given a priori determines the objects that may be combined by the set union of the respective domains. Note that the level value may be specified via the query language with the constraint that it may never exceed the threshold value.

Merging Objects

For object o_i and o_i' , assume \forall_{akj} , the domain (a_{kj}) is noncomposite:

$$o_i'' = \text{Merge}(o_i, o_i') = (i'', \langle a_{k1}:i_{k1}'', a_{k2}:i_{k2}'', \dots, a_{kj}:i_{kj}'', \dots, a_{km}:i_{km}'' \rangle, \langle \mu_o''(C_i), \mu_o''(C_{i+1}), \dots, \mu_o''(C_n) \rangle)$$

where $o_{kj}'' = (i_{kj}'', \{i_{kj}'', i_{kj}'\})$ and $\mu_{o''}(C_m) = f((C_m), (C_m)) \forall m, m = 1, \dots, n$ such that $\forall \text{val}(i_{kj}''), \text{val}(i_{kj}') \in d_{ij} \cup d_{ij}': \min[s(\text{val}(i_{kj}''), \text{val}(i_{kj}')) > \text{Level}(D_j)]$ and $\text{Level}(D_j) \leq \text{Thres}(D_j)$.

The merge operator permits a reorganization of the objects belonging to a class scheme by grouping them according to the similarity of an attribute object to another. As in the definition of threshold, the definition can be extended to composite objects.

Two objects in an OODBMS can be nonredundant even if they are shallow equal. By introducing fuzziness into the model, however, we weaken this property. Two objects that are shallow equal are redundant, as are objects exhibiting deep equality. But equality alone does not determine redundancy, and the following is the characteristic of redundancy:

Two objects o_i and o_i' are redundant iff $\forall_j, j = 1, 2, \dots, m$ and $\text{Level}(D_j)$ given a priori

$$\forall \text{val}(i_{kj}''), \text{val}(i_{kj}') \in d_{ij} \cup d_{ij}': \min[s(\text{val}(i_{kj}''), \text{val}(i_{kj}')) > \text{Level}(D_j)]$$

This property of redundancy (Buckles & Petry, 1982) is directly responsible for the property of value abstraction exhibited by the fuzzy database. It also ensures that the results of database operations are unique.

Other Fuzzy Object-Oriented Approaches

For OODBs, Zicari (1990) considered issues of incompleteness, albeit without use of fuzzy concepts. In particular, incomplete data in an object are handled by the introduction of explicit null values in a similar manner to the relational and nested relational models. Several researchers have been developing fuzzy OODB approaches and studying related issues for a number of years (de Clauwe, 1997; Lee et al., 1999; Pasi & Yager, 1999; Bordogna et al., 2000; de Tre et al., 2000; Marin et al., 2000; Cao, 2001; Koyuncu & Yazici, 2003; Ma, 2000, 2004). Significant applications of fuzzy object modeling are in the areas of complex spatial data and GIS (George et al., 1992; Morris & Petry, 1998; Cross & Firat, 2000).

Generalization in Fuzzy OODB

The starting point for all generalization approaches must be based on the most frequently encountered attribute values — single-valued nonnumeric and numeric data values. We will extensively consider the issues related to generalization for single-valued data and then show how this may extend to structured data and class hierarchy issues.

Attribute Generalization and Concept Hierarchies

For the purpose of attribute-oriented generalization, the concept of hierarchy is critical and in an environment of fuzzy data may lead to different interpretations for generalization. Each concept hierarchy reflects background knowledge about the domain to be generalized. These hierarchies should permit gradual, similarity-based, aggregation of attribute values in the objects. Typically, a hierarchy is built in the bottom-up manner, progressively increasing the abstraction of the generalization concepts at each new level. Creation of new concept levels in generalization hierarchies is accompanied by an increase of the concept abstraction and the decrease of cardinality (each higher level includes less data descriptors, but the descriptors have more general meanings).

Hierarchical grouping (Han, 1995) was based on tree-like generalization hierarchies, where each of the concepts at the lower level of the generalization hierarchy was allowed to have just one abstract concept at the level directly above it. Fuzzy ISA hierarchies were later applied to data summarization (Lee & Kim, 1997), allowing a single concept (attribute value) to partially belong to more than one of the concepts placed at the next abstract level (direct abstracts). However, this and a similar approach (Raschia & Moudaddib, 2002) lack certain properties (exact count/vote propagation) that we find are needed in the attribute-oriented generalization.

Because of the nature of fuzzy OODBs, we can restructure the original data (by merging objects considered to be identical at a certain α -cut level, according to a given similarity relation) in order to begin attribute-oriented generalization from a desired level of detail, the initial set G_o . This approach, when removing unnecessary detail, must be applied with caution. When merging objects according to the equivalence at the given similarity level (e.g., by using queries with a high threshold level), we are not able to keep track of the number of original objects to be merged to one object. This may result in a significant change of balance among the objects in a class and lead to the erroneous (not reflecting reality) information presented later in the form of support and confidence of the extracted knowledge. This problem, which we refer to as a *count dilemma* in the

count propagation, can easily be avoided by performing extraction of initial working class G_0 at a detailed level (i.e., $\alpha = 1.0$), where only identical values are merged (e.g., Bleached and Light Blond will be unified), but no considerable number of objects would be lost as the result of such redundancy removal.

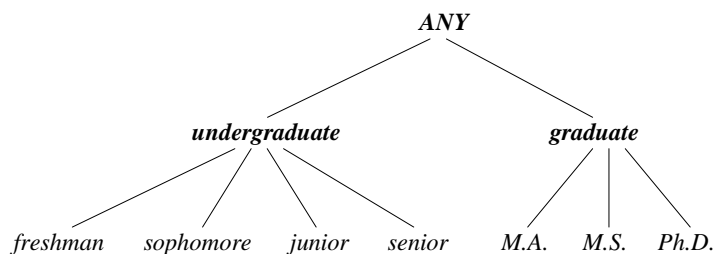
Another issue that must be emphasized is an *exact count propagation dilemma* (also derived from the principle of count propagation). When generalizing data for data mining purposes, we have to preserve the number of objects and the relationships between them in identical proportions at each level of generalization. In other words, we have to assure that each object from the original class will be counted once at each of the levels of the generalization hierarchy. This leads to the two following properties, which must be maintained at each level of the generalization hierarchy:

1. The set of concepts at each level of hierarchy should cover all of the attribute values that occurred in the original database (so we are guaranteed not to lose the number of objects when generalizing their values).
2. Never allow any attribute value (or its abstract) to be counted more or less than once at each level of the generalization hierarchy. (When we allow a concept to partially belong to more than one of its direct abstracts, we have to check each time that the sum of fractional memberships is equal to 1.0). This aspect is especially important when we plan to apply attribute-oriented generalization as a pre-analysis tool, to compress the initial data set to a form more appropriate for the application of computationally complex data mining algorithms (e.g., association rules mining).

For the purpose of further analysis, we distinguish three basic types of generalization hierarchies:

1. *Crisp concept hierarchy* (Han, 1995; Hilderman et al., 1999): Here each attribute variable (concept) at each level of the hierarchy can have only one direct abstract (its direct generalization) to which it fully belongs. (There is no consideration of the degree of relationship, e.g., {master of art, master of science, doctorate} \subset graduate, {freshman, sophomore, junior, senior} \subset undergraduate.) This is as shown in the tree in Figure 1.
2. *Fuzzy concept hierarchy* (Lee & Kim, 1997; Raschia & Mouaddib, 2002): The hierarchy of concepts here reflects the degree with which one concept belongs to its direct abstract and more than one direct abstract of a single concept is allowed. Because of the lack of guarantee of exact count propagation, such a hierarchy seems to be more appropriate for simplified data summarization, or for the cases when subjective results are to be emphasized (when we purposely want to modify the roles or influences of certain objects). Utilization of the four popular text editors could be

Figure 1. Crisp concept hierarchy



generalized as follows (Lee & Kim, 1997). We denote fuzzy generalization of concept a to its direct abstract b with membership degree c as $a \prec b/c$:

- First level of abstraction: $\{emacs \prec editor/ 1.0; emacs \prec documentation/ 0.1; vi \prec editor/ 1.0; vi \prec documentation/ 0.3; word \prec documentation / 1.0; word \prec spreadsheet/ 0.1; wright \prec spreadsheet/ 1.0\}$
 - Second level of hierarchy: $\{editor \prec engineering/ 1.0; documentation \prec engineering / 1.0; documentation \prec business/ 1.0; spreadsheet \prec engineering / 0.8; spreadsheet \prec business/ 1.0\}$
 - Third level of hierarchy: $\{engineering \prec any / 1.0; businessp any / 1.0\}$
3. *Consistent fuzzy concept hierarchy* (recently proposed in Angryk & Petry, 2003): Each degree of membership is normalized to preserve an exact count propagation for each object when being generalized.

Extraction of Concept Hierarchies from Similarity Relations

Here we consider the nature of similarity relations as a mechanism for attribute-oriented generalization. Commonly, the generalization of concepts in data mining is based on the two types of ontological relations: (1) “Part-Of” (e.g., wheels, oil, oil filter, and brake pads sold by Wal-Mart could be generalized to “auto-service items”) and (2) “Is-A” (e.g., red, auburn, ruby, and scarlet could be described in general as “reddish colors”). “Part-Of” emphasizes the similarity of concepts to their abstract, while “Is-A” accentuates the similarity occurring between the values from lower level of abstraction, trying then to define the descriptor fitting its character. In practice we may find loose hybrids of these relations, because the structure of generalization hierarchy strongly depends on the character of the

Table 1. Proximity table for a domain HAIR COLOR

	Black	Dark Brown	Auburn	Red	Blond	Bleached
Black	1.0	0.8	0.7	0.5	0.5	0.5
Dark brown	0.8	1.0	0.7	0.7	0.5	0.5
Auburn	0.7	0.7	1.0	0.8	0.5	0.5
Red	0.7	0.7	0.8	1.0	0.5	0.5
Blond	0.5	0.5	0.5	0.5	1.0	0.8
Bleached	0.5	0.5	0.5	0.5	0.8	1.0

data mining task or personal preferences of the analyst. Each of these relations among concepts can be reflected in a similarity relation, because the user or data-mining analyst can be allowed to modify the values in the similarity table in the individual's user view of the database to represent the similarity between the concepts (attribute values) in the context of interest.

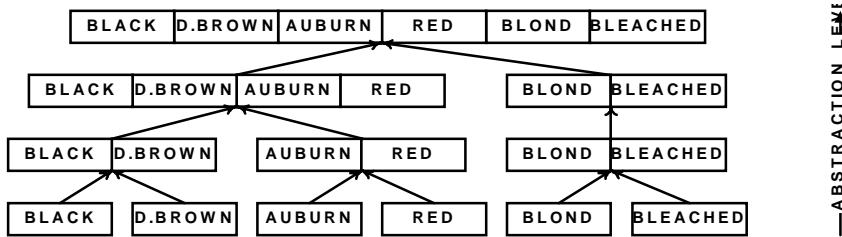
The existence of a similarity relation modeled for a particular domain can lead to the extraction of a crisp concept hierarchy, allowing attribute-oriented generalization. Let S_α be the α -cut of the similarity relation S , presented in Table 1. It can be shown (Zadeh, 1970) that if S is a similarity relation on a given domain D_j (which is a single attribute in our case), then $\forall \alpha \in (0,1]$ each S_α creates equivalence classes in the domain D_j . Now, let Π_α denote the equivalence class partition induced on domain D_j by S_α . Clearly, $\Pi_{\alpha'}$ is a refinement of Π_α if $\alpha' \geq \alpha$. A nested sequence of partitions $\Pi_{\alpha^1}, \Pi_{\alpha^2}, \dots, \Pi_{\alpha^k}$ may be represented diagrammatically in the form of a *partition tree*.

The nested sequence of partitions in the form of a tree has a structure identical to the crisp concept hierarchy for data mining generalization purposes (Figure 2). The increase of abstraction in the partition tree is denoted by decreasing values of α ; lack of abstraction during generalization (0-abstraction level at the bottom of generalization hierarchy) complies with the 1-cut of the similarity relation ($\alpha = 1.0$), and can be denoted as $S_{1.0}$.

An advantage of attribute-oriented generalization with OODBs using similarity relations is that such an hierarchy is implicit in the object-oriented fuzzy model and can be extracted automatically, even by a user who has no background knowledge about the particular domain. Experienced analysts not satisfied with an existing similarity relation may then define their own similarity tables in user views to better reflect their knowledge about the attribute values.

The only difference in Figure 2 from crisp concept hierarchies is their lack of abstract concepts used as labels characterizing the sets of generalized (grouped) concepts. In our example, we could generalize the values *blond* and *bleached* to one common descriptor *BLONDISH*, *auburn* and *red* to *REDDISH*, and *black* and *dark brown* to *DARKISH* (to maintain consistency of the naming

Figure 2. Partition tree of domain HAIR COLOR for similarity relation (Table 1)



convention at the first level of abstraction). At the next level of the generalization hierarchy, we can keep the concept *BLONDISH*, because there is no change in its components; however, according to the taxonomy presented in Figure 2, the concepts *DARKISH* and *REDDISH* should be generalized and should have a new descriptor, which we call *DARK* to emphasize the change. A term *ANY* is usually placed at the highest level of concept hierarchy, to emphasize that the name describes all values possibly occurring in the particular domain. When defining abstract names for generalized sets of attribute values, we need to remember that the lower cut of the similarity relation (smaller values of α) represents a higher abstraction of generalization descriptors.

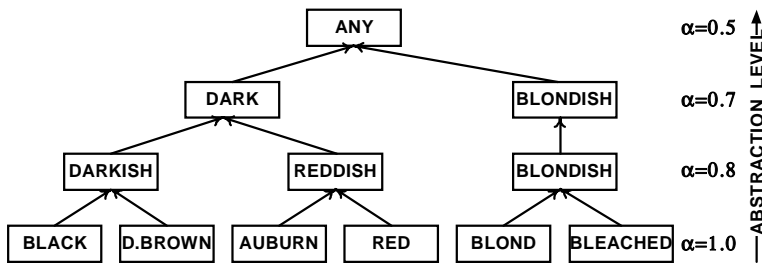
Due to the nested character of partitions as a result of α -cuts of a similarity relation, to specify a complete set of abstract descriptors it is sufficient to choose one value of the attribute per equivalence class partition at each level of the hierarchy, represented by α in Table 2. This is sufficient to build the generalization hierarchy in Figure 3.

Because the similarity relation can generate only a nested sequence of equivalence partitions via a decrease in similarity level, we cannot extract a fuzzy concept hierarchy from the similarity table. The disjoint character of equivalence classes generated from the similarity relation does not allow any concept in the

Table 2. Abstract descriptors, for the generalization hierarchy in Figure 2, where abstraction level is represented by value of α

Original Attribute Value	Abstraction Level	Abstract Descriptor
<i>Black</i>	0.8	<i>DARKISH</i>
<i>Red</i>	0.8	<i>REDDISH</i>
<i>Blond</i>	0.8	<i>BLONDISH</i>
<i>Black</i>	0.7	<i>DARK</i>
<i>Blond</i>	0.7	<i>BLONDISH</i>
<i>Black</i>	0.5	<i>ANY</i>

Figure 3. Crisp generalization hierarchy formed using Tables 1 and 2



hierarchy to have more than one direct abstract at every level of the generalization hierarchy. A similarity table can be utilized to form a crisp generalization hierarchy. Such an hierarchy can be successfully applied as a foundation to the development of a fuzzy concept hierarchy. Data-mining analysts can extend the crisp hierarchy with additional edges to represent partial membership of the lower-level concepts in their direct abstract descriptors. Depending on the assigned memberships, reflecting preferences of the user, they can create consistent or inconsistent fuzzy concept hierarchies.

Utilizing Similarity Relations to Define Abstract Concepts

A similarity relation can be interpreted in terms of fuzzy similarity classes $S(x)$ (Zadeh, 1970), where the membership of attribute variables in the class $S(x)$ is equal to the similarity level between these variables and the fuzzy similarity class. In other words, the grade of membership of y in the fuzzy class $S(x)$, denoted by $\mu_{S(x)}(y)$, is xSy .

Based on this consideration, one can define abstract concepts by choosing their *basic representative attribute values* (i.e., *typical representative specializers*) and then using a similarity table to extract a more precise definition of such abstract classes. For such extraction, we assume a certain level of similarity (α), which should be interpreted as a level of precision reflected in our abstract concept definition.

Typically, the more abstract the concepts to be used in data generalization, the less certain experts are at assigning particular lower-level concepts to them; often, some values can be easily generalized to abstracts, but others may raise doubts among experts. In analyzing the problem of imprecise information, it was noted (Dubois & Prade, 1991) that each attribute has a *domain* (allowed values),

a *range* (actually occurring values), and a *typical range* (most common values), and we apply this classification to the generalization process. With an abstract concept we can usually identify its *typical direct specializers*, the elements clearly belonging to it (e.g., we all would probably agree here that *black* hair can be generalized to the descriptor *DARK* with 100% accuracy). This can be represented as a core of the fuzzy set (abstract concept). However, there are also lower-level concepts that cannot be definitely assigned to only one of their direct abstracts (e.g., assigning *blond* fully to the abstract concept *LIGHT* hair is problematic because there are many people with *dark blond* hair). We term such cases *possible direct specializers*, concepts in the group of lower-level concepts characterized by the given abstract descriptor (fuzzy set) with membership $\mu \leq 1$. These are the support of a fuzzy set and are interpreted as the *range of the abstract concept*.

Now we define each abstract concept as a set of its typical original attribute values with the level of doubt about its other possible specializers reflected by the value of α . Then we select the fuzzy similarity class created from the α -cut of similarity relation for these predefined typical specializers and analyze if this fits our needs. For instance, define the abstract concept *LIGHT* hair by the attribute variable *bleached* with the level of similarity $\alpha = 0.8$ to spread the range of this abstract descriptor (*LIGHT* is predefined as the similarity class *BLEACHED*_{0.8}). From the similarity relation presented in Table 1, we can derive:

$$LIGHT = BLEACHED_{0.8} = \{bleached/1.0; blond/0.8\}$$

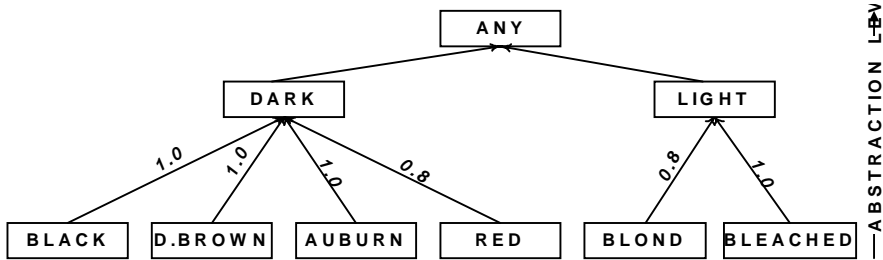
Of course, each of the abstract concepts can be defined by more than one typical representative element (in such a case we may also choose an intersection operator, as best fits our preferences). Assume the descriptor *DARK* to be principally defined by the following original values of the HAIR COLOR domain: *black*, *d.brown*, and *auburn*. Assuming the similarity level to be 0.7, we would obtain:

$$\begin{aligned} DARK &= MAX(BLACK_{0.7}; D. BROWN_{0.7}; AUBURN_{0.7}) \\ &= \{black/1.0; d.brown/1.0; auburn/1.0; red/0.8\} \end{aligned}$$

Using both of these abstract concepts, with assumption that only *DARK* and *LIGHT* colors occur at the given level of HAIR COLOR generalization, we construct the fuzzy generalization hierarchy (Figure 4).

The hierarchy in Figure 4 is called a simplified fuzzy concept hierarchy, because the fractional memberships of low-level concepts to their abstract descriptors

Figure 4. Simplified fuzzy generalization hierarchy for the attribute HAIR COLOR



make it similar to the fuzzy concept hierarchy described previously. Each of the original attribute values belongs to only one direct abstract, creating a simplified (crisp-hierarchy-like) structure. For instance, define an abstract concept *BLACKISH* as the α -cut from the similarity table for *black* at the level 0.7:

$$BLACKISH = BLACK_{0.7} = \{black|1.0; d.brown|0.8; auburn|0.7; red|0.7\}$$

Simultaneously introduce the abstract class *BROWNISH* at the same α -level:

$$BROWNISH = D.BROWN_{0.7} = \{black|0.8; d.brown|1.0; auburn|0.7; red|0.7\}$$

We can derive the fuzzy concept hierarchy and even modify the generalization model to become consistent through the normalization of derived memberships:

$$BLACKISH = BLACK_{0.7} = \{black|\frac{1.0}{1.8}; d.brown|\frac{0.8}{1.8}; auburn|\frac{0.7}{1.4}; red|\frac{0.7}{1.4}\} = \{black|0.6; d.brown|0.4; auburn|0.5; red|0.5\}$$

$$BROWNISH = D.BROWN_{0.7} = \{black|\frac{0.8}{1.8}; d.brown|\frac{1.0}{1.8}; auburn|\frac{0.7}{1.4}; red|\frac{0.7}{1.4}\} = \{black|0.4; d.brown|0.6; auburn|0.5; red|0.5\}$$

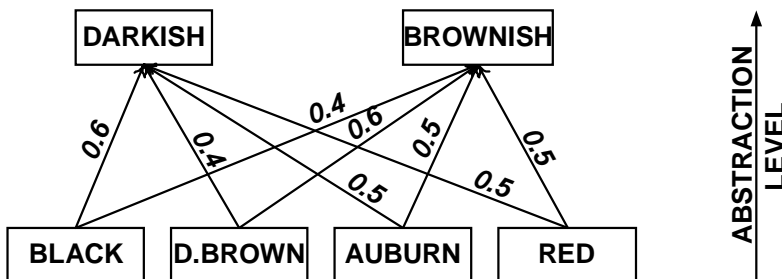
Despite the formally correct appearance, this mechanism may be inappropriate. We characterized two new generalization concepts (*BLACKISH* and *BROWNISH*) with a low level of imprecision (each had only one typical direct specializer), simultaneously choosing a relatively high degree of abstraction ($\alpha = 0.7$) when extracting α -cuts from the similarity relation. This resulted in two fuzzy similarity

classes ($BLACK_{0.7}$ and $D.BROWN_{0.7}$) that were overlapping and led to the consistent fuzzy concept hierarchy in Figure 5 (derived through the normalization of membership degrees). Extraction of two fuzzy classes from the similarity table at the similarity level where they were considered to be equivalent ($black$ and $d.brown$ belong to the same equivalence class partition at the similarity level 0.7), despite being formally possible, often may not be semantically meaningful. This situation may occur when the abstract concepts are characterized incorrectly at the particular level of generalization (which is the case here) or the similarity relation represents the similarity between these concepts in the perspective not compatible with the context represented in the particular generalization hierarchy. It makes no sense to define two or more general concepts at a level of abstraction so high that they are interpreted as identical. This rationale found its natural reflection in the distribution of memberships presented in the consistent fuzzy concept hierarchy (Figure 5), where both of the introduced abstract concepts have almost identical compositions of their direct specializers.

Some guidelines are needed when characterizing abstract concepts via their typical direct specializers and trying to extract their full definition (range of possible direct specializers) using a similarity table:

1. We need to assure that the intuitively assumed value of α extracts the cut (subset) of attribute values that corresponds closely to the definition of the abstract descriptor for which we were looking. The strategy for choosing the most appropriate level of α -cut when extracting the abstract concept definitions arises from the guideline of minimal generalization (the minimal concept tree ascension strategy described in the second section). Based on this strategy, we would recommend always choosing a definition extracted at the highest possible level of similarity (biggest α), where all predefined typical components of the desired abstract descriptor are already embraced (where they occur for the first time).

Figure 5. Consistent fuzzy concept hierarchy for the attribute HAIR COLOR



2. The problem of selecting appropriate representative elements without external knowledge about a particular attribute remains; however, it can now be supported by the analysis of the values stored in the similarity table. Choosing typical values and then extracting a detailed definition with all possible components of the desired abstract concepts from the similarity table seems to be easier than describing generalized components in detail.
3. Moreover, we should be aware that if low-level concepts, predefined as typical components of the particular abstract descriptor, do not occur in the common similarity class, then the contexts of the generalized descriptor and of the similarity relation are not in agreement, and revision of the similarity table (or the abstract) is recommended.
4. We cannot directly specify a restriction stating that all abstract concepts in the generalization hierarchy have to be at the same level of similarity, when extracted from the similarity relation. Moreover, definitions extracted to the example presented in Figure 4 show that this situation is acceptable. However, when using this approach, we should generally not put at the same level of generalization hierarchy the abstract descriptors that overlap with others. This can easily occur when trying to place an abstract defined via the original concepts on the given level of the generalization hierarchy. This abstract is already represented on that level by the generalized concept derived from the equivalence class partition with the higher similarity level. We have to remember that the abstract concepts derived from the similarity relation have nested character, and placing one concept simultaneously with the other may lead to the partial overlapping of partitions (because it is its actual refinement), which contradicts the character of similarity relation.

The approach described here seems to allow us to form only flat (one-level) generalization hierarchies or to derive the generalized concepts at the first level of abstraction in the concept hierarchy. Each abstract concept defined with this method is a generalization of original attribute values, and therefore cannot be placed at the higher level of the concept hierarchy. However, there is no obstacle preventing these concepts from being further generalized.

The lack of ability to derive multilevel hierarchical structures does not prevent this approach from being appropriate, and actually convenient, for rapid data summarization or something we term “selective attribute-oriented generalization.” To summarize the given data set, we may prefer to not perform gradual (hierarchical) generalization but replace it with a one-level hierarchy covering a whole domain of attribute values. Such an appropriately built “flat hierarchy” would represent the majority of dependencies between the original low-level concepts, which are to be generalized, by the propagation of fractions of counts

coming from each attribute value, instead of having to perform detailed hierarchical generalization.

In selective generalization, we generalize all attribute values from a specific point of view, which is dictated by the character of the data mining task. Assume that we are interested in association rules regarding only people who have dark hair. Using the similarity relation, we derive the following:

$$\begin{aligned} DARKISH &= MAX(BLACK_{0.7}; D.BROWN_{0.7}) \\ &= \{black/1.0; d.brown/1.0; auburn/0.7; red/0.7\} \end{aligned}$$

This reflects the following interpretation: All people who have *black* or *dark brown* hair are considered to have *DARKISH* hair, and 70% of *redheads* and people with *auburn* hair have it in a dark shade. This is sufficient to explain the difference between selective generalization and the application of data selection when building the initial data-mining class G_o . In both cases, we omit all objects with hair; however, in the case of selective generalization, 70% of each count represented by each object with red or auburn hair color remains. This is obviously not equivalent to the extraction of all objects with values “red” and “auburn” and then randomly choosing 70% of them for further generalization. With selective generalization, we do not omit the objects but decrease their influence to an appropriate representation of their importance for the given data-mining problem.

We should finally point out that consistent fuzzy hierarchies are not appropriate tools for selective attribute-oriented generalization. In this case, we do not want to have normalization of counts’ values to preserve exact count dilemma, we instead want to preserve an unbalanced relation between the objects, as this reflects dependencies occurring in real-life data. The ordinary fuzzy hierarchies seem to be the most appropriate for such purposes.

Although we focused on nonnumeric data in this discussion of fuzzy concept hierarchies, the generalization of numeric attributes can be performed in a similar manner. Of course, the numeric hierarchy can be based on similarity relationships for fuzzy numbers, such as was already developed for fuzzy databases (Buckles & Petry, 1984; Petry 1996), and used as described above for nonnumeric data. In the case of numeric data, it is possible to analyze the data distribution characteristics. It may then not be necessary to have predefined concept hierarchies. For example, consider an income range study in which the incomes can be clustered into several groups, $\{<20K, 20-35K, 35-45K, 45-50K, >50K\}$, based on some statistical clustering tool. Obviously, further clustering can be

done on these groupings to form a multilevel hierarchy. Linguistic terms can be assigned to groups, {very low, low, medium, high, very high}, to provide labels in the hierarchy for generalization. This may be a crisp hierarchy, but it is also possible to formulate a fuzzy hierarchy by techniques such as use of fuzzy agglomerative clustering (Yager, 2000).

Generalization of Structured Data Values and Class Hierarchies

In general, we may have complex structured data such as set and list valued data or data with nested structures. First let us consider an attribute that may be multi- or set-valued. Each value in a set can first be generalized into its higher-level concept. For example, if we have the multivalued attribute “*skills*” for an employee, we might have the set of values: {German, Programming, Pilot}. If the next level in a concept hierarchy were to classify skills as mental or physical, then we would have the set {(Physical Skills, $count_p$), (Mental Skills, $count_m$)}, where $count_p$ is the value 1, and $count_m$ 2, but each is scaled as appropriate depending on the type of the concept hierarchy being utilized.

For more complex data, we still base the approach on set-valued data as above. A list-valued attribute can be generalized in the same manner as that for the list elements, except that a generalized form of the list order must be used in the generalization process. For structured data, we can consider that same approach but must evaluate alternatives to structure generalization. When generalizing individual attribute values, we may maintain the shape of the structure or provide some generalization of the structure, such as flattening the structure or removing low-level values and summarizing them. Recall also that we have a fuzzy class hierarchy:

$$(o_i, \mu(C_i)) \subseteq^s (C_i, \mu(C_{i+1})) \subseteq^s (C_{i+1}, \mu(C_{i+2})) \subseteq^s \dots \subseteq^s (C_n, \mu(C_{n+1})),$$

so that when we generalize the object o_i , we must account for the degree of membership in its particular class. This can be done scaling the object’s count, $o_i.count$, by the membership $\mu(C_i)$ for the current class of o_i . If the generalization of o_i moves up through the hierarchy, then the appropriate weighting must be taken into account for $o_i.count$.

Conclusions

We considered in detail the issues relative to concept hierarchies for attribute generalization, as the use of a concept hierarchy is the essential component of the generalization process. As we have seen, there are several approaches that can be taken depending on the exact intention of the data-mining application. This allows one to be more flexible in dealing with fuzzy objects in the similarity-based fuzzy OODB model we described, in particular, due to the ability to create hierarchies from the given similarity relationships for the data domains.

There are several directions that can be profitably followed in this area for OODBs that we have not considered to date. Two of particular interest that we are currently studying are the issues of generalization of methods and the use of aggregation as a structuring mechanism. As an application area, the problem of generalization of multimedia data, especially spatial data (Ladner, Petry, & Cobb, 2003), in a fuzzy OODB is of particular interest. Also, we have been considering the extension of fuzzy hierarchy development in a database utilizing proximity relationships (Angryk & Petry, 2003) and plan on extending the fuzzy OODM to accommodate generalization via proximity relations.

ACKNOWLEDGMENTS

We would like to thank the Naval Research Laboratory's Base Program, Program Element No. 0602435N for sponsoring this research.

References

- Agrawal, R., Imielinski, T., & Swami, A. (1993). Mining association rules between sets of items in large databases. In *Proceedings of the 1993 ACM-SIGMOD International Conference on Management of Data* (pp. 207–216). New York: ACM Press.
- Angryk, R., & Petry, F. (2003). Consistent fuzzy concept hierarchies for attribute generalization. In *Proceedings IASTED International Conference on Information and Knowledge Sharing (IKS 2003)* (pp. 158–193).
- Angryk, R., & Petry, F. (2003). Data mining fuzzy databases using attribute-oriented generalization. In *Proceedings of the IEEE International Con-*

- ference Data Mining Workshop on Foundations and New Directions in Data Mining* (pp. 8–15). Melbourne, FL.
- Au, W., & Chan, K. (2003). Mining fuzzy association rules in a bank-account database. *IEEE Transactions on Fuzzy Systems*, *11*(2), 238–248.
- Bertino, E., & Martino, L. (1991). Object-oriented database management systems: Concepts and issues. *IEEE Computer*, *24*, 65–81.
- Bezdek, J. (1974). Cluster validity with fuzzy sets. *Journal of Cybernetics*, *3*, 58–72.
- Bordogna, G., Leporati, A., Lucarella, D., & Pasi, G. (2000). The fuzzy object-oriented database management system. In G. Bordogna, & G. Pasi (Eds.), *Recent issues on fuzzy databases* (pp. 209–236). Heidelberg: Physica-Verlag.
- Bosc, P., & Pivert, O. (2001). On some fuzzy extensions of association rules. In *Proceedings of IFSA-NAFIPS 2001* (pp. 1104–1109). Piscataway, NJ: IEEE Press.
- Buckles, B., & Petry, F. (1982). A fuzzy representation for relational data bases. *International Journal of Fuzzy Sets and Systems*, *7*, 213–226.
- Buckles, B., & Petry, F. (1984). Extending the fuzzy database with fuzzy numbers. *Information Sciences*, *34*, 45–55.
- Cao, T. (2001). Uncertain inheritance and recognition as probabilistic default reasoning. *International Journal of Intelligent Systems*, *16*, 781–803.
- Carter, C., & Hamilton, H. (1998). Efficient attribute-oriented generalization for knowledge discovery from large databases. *IEEE Transactions on Knowledge and Data Engineering*, *10*(2), 193–208.
- Chaudhri, A., & Lommis, M. (Eds.). (1998). *Object databases in practice*. New York: Prentice Hall.
- Chen, G., Wei, Q., & Kerre, E. (2000). Fuzzy data mining: Discovery of fuzzy generalized association rules. In G. Bordogna, & G. Pasi (Eds.), *Recent issues on fuzzy databases* (pp. 45–66). Heidelberg: Physica-Verlag.
- Cross, V., & Firat, A. (2000). Fuzzy objects for geographical information systems. *International Journal of Fuzzy Sets and Systems*, *113*, 19–36.
- Cubero, J., Medina, J., Pons, O., & Vila, M. (1999). Data summarization in relational databases through fuzzy dependencies. *Information Sciences*, *121*(3–4), 233–270.
- de Clauwe, R. (Ed.). (1997). *Fuzzy and uncertain object-oriented databases: Concepts and models*. Singapore: World Scientific.
- de Graaf, J., Kosters, W., & Witteman, J. (2001). Interesting fuzzy association rules in quantitative databases. In *Principles of Data Mining and*

- Knowledge Discovery LNAI 2168* (pp. 140–151). Heidelberg: Springer-Verlag.
- de Tre, G., de Clauwe, R., & Van der Cruyssen, B. (2000). A generalized object-oriented database model. In G. Bordogna, & G. Pasi (Eds.), *Recent issues on fuzzy databases* (pp. 155–182). Heidelberg: Physica-Verlag.
- Delgado, M., Marin, N., Sanchez, D., & Vila, M. (2003). Fuzzy association rules: General model and applications. *IEEE Transactions on Fuzzy Systems*, *11*(2), 214–225.
- Dubois, D., & Prade, H. (2000). Fuzzy sets in data summaries — outline of a new approach, In *Proceedings of the Eighth International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems* (pp. 1035–1040). Madrid, Spain.
- Dubois, D., Prade, H., & Rossazza, J. (1991). Vagueness, typicality and uncertainty in class hierarchies. *International Journal of Intelligent Systems*, *6*, 167–183.
- Feelders, A., Daniels, H., & Holsheimer, M. (2000). Methodological and practical aspects of data mining. *Information and Management*, *37*, 271–281.
- Feng, L., & Dillon, T. (2003). Using fuzzy linguistic representations to provide explanatory semantics for data warehouses. *IEEE Transactions on Knowledge and Data Engineering*, *15*(1), 86–102.
- George, R., Buckles, B., Petry, F., & Yazici, A. (1992). Uncertainty modeling in object-oriented geographical information systems. In *1992 Proceedings of Conference on Database & Expert System Applications* (pp. 77–86). Heidelberg: Springer-Verlag.
- George, R., Buckles, B., & Petry, F. (1993). Modeling class hierarchies in the fuzzy object-oriented data model. *Int. J. of Fuzzy Sets and Systems*, *60*, 259–272.
- Gyenesei, A. (2001a). A fuzzy approach for mining quantitative association rules. *Acta Cybernetica*, *15*, 305–320.
- Gyenesei, A. (2001b). Interestingness measures for fuzzy association rules. In *Principles of data mining and knowledge discovery — LNAI 2168* (pp. 152–164). Heidelberg: Springer-Verlag.
- Han, J. (1995). Mining knowledge at multiple concept levels. In *Proceedings of the Fourth International Conference on Information and Knowledge Management* (pp. 19–24). New York: ACM Press.
- Han, J., & Kamber, M. (2000). *Data mining: Concepts and techniques*. San Diego, CA: Academic Press.

- Han, J., Nishio, S., & Kawano, W. (1994). Knowledge discovery in object-oriented and active databases. In F. Fuchi, & T. Yokoi (Eds.), *Knowledge building and knowledge sharing* (pp. 221–230). Singapore: IOS Press.
- Han, J., Nishio, S., Kawano, H., & Wang, W. (1998). Generalization-based data mining in object-oriented databases using an object-cube model. *Data and Knowledge Engineering*, 25(1–2), 55–97.
- Hilderman, R., Hamilton, H., & Cercone, N. (1999). Data mining in large databases using domain generalization graphs. *Journal of Intelligent Information Systems*, 13(3), 195–234.
- Hirota, K., & Pedrycz, W. (1999). Fuzzy computing for data mining. In *Proceedings of the IEEE*, 87, 1575–1599.
- Kacprzyk, J. (1999). Fuzzy logic for linguistic summarization of databases. In *Proceedings of the Eighth International Conference on Fuzzy Systems* (pp. 813–818). Seoul, Korea.
- Kacprzyk, J., & Zadrozny, S. (2000). On combining intelligent querying and data mining using fuzzy logic concepts. In G. Bordogna, & G. Pasi (Eds.), *Recent issues on fuzzy databases* (pp. 67–81). Heidelberg: Physica-Verlag.
- Khoshafian, S., & Copeland, G. (1986). Object identity. In *Proceedings of the OOPSLA '86 Conference* (pp. 406–416). New York: ACM Press.
- Kim, W. (1989). A model of queries for object-oriented databases. In *Proceedings of 15th International Conference on Very Large Databases* (pp. 45–54).
- Koyuncu, M., & Yazici, A. (2003). IFOOD: An intelligent fuzzy object-oriented database architecture. *IEEE Transactions Knowledge and Data Engineering*, 15(5), 1137–1154.
- Kuok, C., Fu, A., & Wong, H. (1998). Mining fuzzy association rules in databases. *ACM SIGMOD Record*, 27, 41–46.
- Ladner, R., Petry, F., & Cobb, M. (2003). Fuzzy set approaches to spatial data mining of association rules. *Transactions on GIS*, 7(1), 123–138.
- Laurent, A., Bouchon-Meunier, B., Doucet, A., Gancarski, S., & Marasal, C. (2000). Fuzzy data mining from multidimensional databases. *Studies in Fuzziness and Soft Computing*, 54, *Proceedings of ISCI* (pp. 245–256).
- Lee, D., & Kim, M. (1997). Database summarization using fuzzy ISA hierarchies. *IEEE Transactions On Systems, Man, and Cybernetics — Part B*, 27(1), 68–78.
- Lee, J., Xue, N., Hsu, K., & Yang, J. (1999). Modeling imprecise requirements with fuzzy objects. *Inf. Sci.*, 118, 101–119.

- Lee, K. (2001). Mining generalized fuzzy quantitative association rules with fuzzy generalization hierarchies. In *Proceedings of IFSA-NAFIPS 2001* (pp. 2977–2982). Piscataway, NJ: IEEE Press.
- Ma, Z., Zhang, W., Ma, W., & Chen, G. (2001). Conceptual design of fuzzy object-oriented databases using extended entity–relationship model. *International Journal of Intelligent Systems*, *16*, 697–711.
- Ma, Z., Zhang, W., & Ma, W. (2004). Extending object-oriented databases for fuzzy information modeling. To appear in *Information Systems*.
- Marín, N., Vila, M., & Pons, O. (2000). Fuzzy types: A new concept of type for managing vague structures. *International Journal of Intelligent Systems*, *15*, 1061–1085.
- Morris, A., Petry, F., & Cobb, M. (1998). Fuzzy object-oriented database modeling of spatial data. In *Proceedings IPMU Conference* (pp. 604–611). Paris: EDK Press.
- Pasi, G., & Yager, R. (1999). Calculating attribute values using inheritance structures in fuzzy object-oriented data models. *IEEE Transactions on Systems, Man, and Cybernetics — Part B*, *29*(4), 556–564.
- Petry, F. (1996). *Fuzzy databases: Principles and applications*. Boston, MA: Kluwer Academic Publishers.
- Raschia, G., & Mouaddib, N. (2002). SAINTETIQ: A fuzzy set-based approach to database summarization. *Fuzzy Sets and Systems*, *129*, 137–162.
- Shu, J., Tsang, E., & Yeung, D. (2001). Query fuzzy association rules in relational databases. In *Proceedings of IFSA-NAFIPS 2001* (pp. 2989–2993). Piscataway, NJ: IEEE Press.
- Yager, R. (1991). On linguistic summaries of data. In G. Piatetsky-Shapiro, & Frawley (Eds.), *Knowledge discovery in databases* (pp. 347–363). Boston, MA: MIT Press.
- Yager, R. (2000). Intelligent control of the hierarchical agglomerative clustering process. *IEEE Transactions on Systems, Man, and Cybernetics — Part B*, *30*(6), 835–845.
- Zadeh, L. (1970). Similarity relations and fuzzy orderings. *Information Sciences*, *3*, 177–200.
- Zhang, W. (1999). Mining fuzzy quantitative association rules. In *Proceedings of IEEE International Conference on Tools with Artificial Intelligence* (pp. 99–102). Piscataway, NJ: IEEE Press.
- Zicari, R. (1990). Incomplete information in object-oriented databases. *SIGMOD RECORD*, *19*, 33–40.