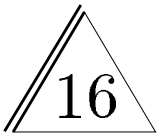

Exploration of the Power of Attribute-Oriented Induction in Data Mining

Jiawei Han
Yongjian Fu
Simon Fraser University

Abstract

Attribute-oriented induction is a set-oriented database mining method which generalizes the task-relevant subset of data attribute-by-attribute, compresses it into a generalized relation, and extracts from it the general features of data. In this chapter, the power of attribute-oriented induction is explored for the extraction from relational databases of different kinds of patterns, including characteristic rules, discriminant rules, cluster description rules, and multiple-level association rules. Furthermore, it is shown that the method is efficient, robust, with wide applications, and extensible to knowledge discovery in advanced database systems, including object-oriented, deductive, and spatial database systems. The implementation status of DB-Miner, a system prototype which applies the method, is also reported here.

16.1 Introduction

With an upsurge of the application demands and research activities on knowledge discovery in databases (Matheus, Chan and Piatetsky-Shapiro 1993; Piatetsky-Shapiro and Frawley 1991), an *attribute-oriented induction method* (Cai, Cercone and Han 1991; Han, Cai and Cercone 1993) has been developed as an interesting technique for mining knowledge from data. The method integrates a machine learning paradigm (Michalski 1983), especially learning-from-examples techniques, with database operations, extracts generalized rules from an interesting set of data, and discovers high-level data regularities.

The development of the attribute-oriented induction method is motivated by the following observations. First, although certain regularities, such as association rules, can be discovered and expressed at the primitive concept level by interesting data mining

techniques (Piatetsky-Shapiro 1991; Agrawal, Imielinski and Swami 1993), stronger and often more interesting regularities can be discovered at high concept levels and expressed in concise terms. Thus it is often necessary to generalize low level primitive data in databases to relatively high level concepts for effective data mining. Second, since autonomous discovery often generates too many rules without focus, it is recommended to initiate a knowledge discovery process by a user's request which represents relatively constrained search on a specified subset of data for desired knowledge. Third, the availability of certain background knowledge, such as conceptual hierarchies, not only improves the efficiency of a discovery process but also expresses user's preference for guided generalization, which may lead to an efficient and desirable generalization process.

Based on these observations, a set-oriented, generalization-based induction method, called *attribute-oriented induction*, has been developed, which collects an interesting set of data as an initial data relation, performs on it the induction process including attribute removal, concept hierarchy ascension, generalization operator application, etc. attribute-by-attribute, eliminates duplications among generalized tuples (but with *counts* accumulated), and derives generalized relations and rules. The method has been implemented in a data mining system prototype, DBMiner (previously called DBLearn, and been tested successfully against large relational databases.

In this chapter, the power of attribute-oriented induction in data mining is examined in two aspects: (1) attribute-oriented induction for mining different kinds of rules, including characteristic rules, discriminant rules, association rules, and cluster description rules, and (2) extension of the method for data mining in advanced database systems, including object-oriented, deductive, and spatial database systems. The study outlines the high potential of the method for efficient and effective knowledge discovery in large databases.

The remainder of the chapter is organized as follows. In Section 2, a brief summary of the methodologies of attribute-oriented induction is presented. In Section 3, the extraction of different kinds of rules by the induction method is examined. In Section 4, the extensions of the method towards knowledge discovery in advanced database systems are discussed. The implementation of DBMiner is discussed in Section 5, and the study is summarized in Section 6. Because of the broad scope of the discussion, the chapter is descriptive in nature. More rigorous treatment can be found in the references.

16.2 A Brief Summary of Attribute-Oriented Induction Methodology

A data mining request should include the specification of the interesting subset of data and the kind of rules to be discovered, as shown in a DBMiner query below.

Example 16.1 The following query, presented in an SQL-like syntax, requests to use the database "NSERC94" (containing the information about 1994-1995 NSERC[1] research grants), and find a characteristic rule from two relations, `award` and `organization`, which satisfies the condition specified in the where-clause and in relevance to four attributes: `province, amount, percentage(count), percentage(amount)`, where `percentage`(x) = $x/total\_x\%$ (the ratio of x versus *total* x). A *characteristic rule* is an assertion that characterizes all or most of the data undergoing examination (called the *target class*).

```
use NSERC94
find characteristic rule for 'CS_Grants'
from award A, organization O
where A.org_code = O.org_code and A.disc_code = 'Computer Science'
in relevance to province, amount, percentage(count), percentage(amount)
```

The process first collects the interesting set of data (target class) by executing an SQL query based on the condition specified in the where-clause which contains a high-level concept "Computer Science". Since the database stores only the detailed discipline code such as 25502 for "database management", the concept hierarchy for discipline code should be consulted in the data retrieval. Similarly, concept hierarchies should be consulted in the generalization of data in the attributes `province` and `amount`. □

A *concept hierarchy* defines a sequence of mappings from a set of concepts to their higher-level correspondences. It is usually partially ordered according to a general-to-specific ordering, with the most general concept usually defined by a reserved word "*any*" and the most specific concepts corresponding to the specific data in the database.

Concept hierarchies represent necessary background knowledge which directs the generalization process. Using a concept hierarchy, the discovered rules can be represented in terms of generalized concepts and stated in a simple and explicit form, which is desirable to most users.

Many concept hierarchies, such as `geo_location(city, province, country)`, are stored in the database implicitly, which can be made explicit by specifying certain attribute mapping rules. Some hierarchies can be provided by knowledge engineers or domain experts, which is reasonable even for large databases since a concept hierarchy registers only the *distinct* discrete attribute values or ranges of numerical values for an attribute, which is, in general, not very large. To certain extent, concept hierarchies can also be generated automatically (Fisher 1987; Chu and Chiang 1994) based on data semantics and/or data distribution statistics. Moreover, a given concept hierarchy may

[1] the Natural Sciences and Engineering Research Council of Canada

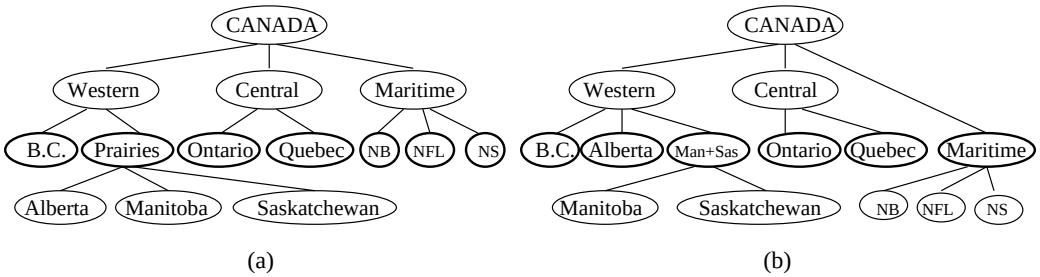

Figure 16.1
The given and refined concept hierarchies for the attribute "province".

not be best suited for a particular learning task, which therefore often needs to be dynamically refined for desired learning results (Han and Fu 1994).

Example 16.2 The given concept hierarchy for *province* (Figure 16.1 (a)), based on the geographic and administrative regions of Canada, may not reflect the characteristics of research grant distribution of Computer Science in Canada. Such a hierarchy needs to be dynamically refined based on the query, attribute threshold (the desired number of higher level attribute values), and data distribution. The refinement is performed by identifying and promoting "big" nodes and grouping the small ones while maximally preserving the original shape of the hierarchy (thus the semantic meaning) (Han and Fu 1994). The refined hierarchy for the query of Example 16.1 is described in Figure 16.1 (b). □

Different concept hierarchies can be constructed on the same attribute based on different viewpoints or preferences. For example, the birthplace can be organized according to administrative regions, geographic locations, size of cities, etc. Usually, a popularly referenced hierarchy is associated with an attribute as the default one. Other hierarchies can be chosen explicitly by users in a data mining process. A concept hierarchy can also be in the shape of lattice or DAG (directed acyclic graph). Moreover, it is sometimes preferable to perform induction in parallel along with more than one hierarchy and select a promising one based on some intermediate generalization results.

16.2.1 Attribute-oriented induction for mining characteristic rules

The attribute-oriented induction method is examined here using the DBMiner query presented in Example 16.1.

In a generalized relation, some or all of its attribute values are generalized data, that is, nonleaf nodes in the concept hierarchies. It is usually desired that an attribute in a (generalized) relation contains only a small number of distinct values. A system may set a small integer as a default, desired *attribute threshold* for each attribute, which

should be adjustable by users, because a user may know better which particular attribute should have more distinct values (thus a larger threshold) than others. An attribute is at the desirable level if it contains no more distinct values than its attribute threshold. Moreover, the attribute is at the *minimum desirable level* if it would contain more distinct values than the threshold when it were specialized to a level lower than the current one. A particular generalized relation R' of an initial relation R is the *prime relation* of R if every attribute in R' is at the minimum desirable level. Besides threshold control, one may also allow a user to specify explicitly a level in the hierarchy as the level to be generalized.

For mining characteristic rules, the attribute-oriented induction is performed in the following steps (Han and Fu 1994).

1. **Initial data collection:** The data mining request is transformed into an SQL query and executed to collect the set of data relevant to the data mining task (as an initial relation).

2. **Derivation of the generalization plan for each attribute:** If there is a large set of distinct values in an attribute of the initial relation, the attribute should be generalized by either attribute removal or attribute generalization. The former is performed when there is no generalization operator on the attribute, or its higher-level concepts are expressed in another attribute. The latter is performed otherwise by (1) determining the *prime level* (generalized) concepts for each attribute, which is the level (possibly determined after the refinement of concept hierarchy) that makes the number of distinct generalized values just within its attribute threshold, and (2) linking them with the data in the initial relation to form generalization pairs.

3. **Prime relation derivation:** Perform attribute-oriented generalization, by substituting lower level concepts with its corresponding prime level concepts, which leads to a prime relation, by eliminating duplicated tuples and accumulating the counts in the retained generalized tuples.

The method described above integrates relational database operations with attribute-oriented generalization and leads to an efficient induction process. Step 1 is a typical relational query whose optimization relies on the well-developed relational technology. Let the initial relation and the derived prime relation contain n and p tuples respectively. Step 2 involves one scan (or less if a sampling technique is adopted) of the initial relation, with the worst-case time complexity of $O(n)$. Step 3 scans the initial relation, generalizes the relation attribute by attribute, and inserts generalized tuples into the result relation, which takes $O(n)$ time if p is small, or $O(n \log p)$ time if the tuples in the result relation are ordered and a binary or tree-based search is applied.

Conceptually, the induction process described above can be viewed as a data generalization and compression process, which compresses an initial relation into a usually much smaller prime relation expressed at high concept levels.

Knowledge rules or general data distributions can be extracted from the prime relation by statistics and/or machine learning tools. There are many techniques for extraction of interesting generalized information from a prime relation:

1. *Direct presentation of the prime relation.* This is effective if the prime relation is small. Otherwise, further reduction of the prime relation is often desirable.
2. *Reduction of the number of attributes in the prime relation.* A simple method is to project on different sets of attributes to extract different *generalized feature tables.* Example 16.3 shows a prime relation and a feature table for `count%` in relevance to `amount` and `province`. Moreover, many techniques developed in previous studies on machine learning (Michalski, Carbonell and Mitchell 1986), statistics, fuzzy set and rough set theories (Ziarko 1994), etc. can be applied to the evaluation of the importance of different sets of attributes and projection on the appropriate ones.
3. *Further generalization of the prime relation.* A *relation threshold* can be specified (or set by default) to indicate the maximum number of tuples that the *final generalized relation* should contain. There are usually alternative choices to select a candidate attribute for further generalization. The interestingness of the final generalized relation relies on the selection of the attributes to be generalized and the selection of generalization operators, based on data semantics, user preference, generalization efficiency, etc. Criteria, such as the preference of a larger reduction ratio on the number of tuples or the number of distinct attribute values, the simplicity of the final discovered rules, etc., can also be used for selection.

The above discussion shows that knowledge rules can be represented in the form of generalized relations, generalized feature tables, or generalized rule(s), in which a tuple in the generalized relation is transformed into conjunctive normal form, and multiple tuples are transformed into disjunctive normal form. Interesting rules can often be discovered by following different paths leading to several generalized relations for comparison. Following different paths corresponds to the way in which different people may learn differently from the same set of examples. The generalized relations can be examined by users or experts interactively to filter out trivial rules and preserve interesting ones (Zytkow and Baker 1991). Therefore, a user-friendly graphical interface is implemented in DBMiner for users to try different alternatives for interactive mining of desired rules.

Example 16.3 Following Example 16.1, part of an original relation, `award`, is shown in Table 16.1. Part of the prime relation is in Table 16.2. A feature table of count%

grantee_id	name	department	org_code	year	amount	...
156274	BAERLOCHER FJ	Biology	1620	1994	16000	...
...	...	...	...	...	...	...

Table 16.1
A portion of the original table *award*.

amount	province	count%	amount%
0-20Ks	Alberta	4.53%	3.26%
0-20Ks	British Columbia	7.41%	4.64%
...	...	...	...

Table 16.2
A portion of the *prime relation*.

extracted from Table 16.2 is in Table 16.3. Each entry in Table 16.3 represents the percentage of the `count` of the corresponding "feature" (pattern) with respect to the `total count`. □

16.2.2 Feasibility of attribute-oriented induction

A major strength of attribute-oriented induction over tuple-oriented one is at its efficiency in the induction process. The former performs generalization on each attribute *uniformly* for all the tuples in the initial relation in the formation of the prime relation, which contrasts with the latter that explores different possible combinations for a large number of tuples in the same generalization phase. An exploration of different possible generalization paths for different tuples at the early generalization stage will not be productive since these combinations will be merged in further generalization. Different possible combinations should be explored only when the relation has been generalized to a relatively small prime relation.

Attribute-oriented induction is robust and handles noise and/or exceptional cases elegantly because it incorporates statistical information (using count) and generates disjunctive rules. The association of count with each disjunct leads naturally to mining *approximate rules*, for which the conditions with negligible weight can be dropped in generalization and rule formation since a negligible weight implies a minimal influence to the conclusion.

Count association facilitates incremental learning (Fisher 1987) in large databases as well: when a new tuple is inserted into a database, its concepts (attribute values) are first generalized to the same concept level as those in the generalized relation and then merged naturally into the generalized relation.

Furthermore, with the association of count information, data sampling (Kivinen and Mannila 1994) and parallelism can be explored in knowledge discovery. Attribute-

amount	province						total
	Alberta	B.C.	Maritime	Ontario	Quebec	Sas+Man	
0-20Ks	4.53%	7.41%	6.79%	24.49%	13.79%	3.70%	60.70%
20Ks-40Ks	3.70%	5.35%	1.03%	12.76%	5.14%	1.65%	29.63%
40Ks-60Ks	0.21%	1.23%	0.00%	5.14%	1.03%	0.00%	7.61%
60Ks-	0.21%	0.21%	0.00%	1.23%	0.21%	0.21%	2.06%
Total	8.64%	14.20%	7.82%	43.62%	20.16%	5.56%	100.00%

Table 16.3
A feature table of *count%* extracted from Table 16.2

oriented induction can be performed by sampling a subset of data from a huge set of relevant data or by first performing induction in parallel on several partitions of the relevant data set and then merging the generalized results.

As a generalization-based method, attribute-oriented induction confines its power to the discovery of knowledge rules at general concept levels. There are applications which require the discovery of knowledge at primitive concept levels, such as finding (primitive) association or dependency rules, or finding functional relationships (e.g., "$y = f(x)$") between two numerical attributes (e.g., height and weight). Attribute-oriented induction does not suit such applications. Moreover, attribute-oriented induction needs reasonably informative concept hierarchies for generalization of nonnumerical attributes, and over-generalization should be guarded against by setting thresholds flexibly and/or interactively. Nevertheless, the method is useful in most database-oriented applications which need to generalize some or all of the relevant attributes.

16.3 Discovery of different kinds of rules

Besides mining characteristic rules, attribute-oriented induction can be used to discover discriminant rules, cluster description rules, and multi-level association rules, as illustrated in this subsection. Moreover, other kinds of rules, including data evolution regularities and deviation rules, can also be discovered based on the similar philosophy.

16.3.1 Mining discriminant rules

A *discriminant rule* is an assertion which discriminates concepts of the class being examined (the *target class*) from other classes (called *contrasting classes*). For example, to distinguish one disease from others, a discriminant rule should summarize the symptoms that discriminate this disease from others.

A discriminant rule can be discovered by generalizing the data in both target class and contrasting class(es) *synchronously* in an attribute-oriented fashion and excluding the

properties that overlap in both classes in the generalized rule. Usually, a property, which is quite good at discriminating a class, may still have minor overlaps with other classes in a large data set. Thus it is often preferable to associate quantitative information with a rule or with the tuples in a generalized relation to indicate how precisely a property can be used to distinguish a target class from others. Such a quantitative measurement, called *discriminating weight* (Han, Cai and Cercone 1993), can be defined as the ratio of the frequency that a property occurring in the target class versus that occurring in both (target and contrasting) classes. Obviously, a property with a discriminating weight close to 1 is a good discriminator; whereas that close to 0 is a negative discriminator (the properties unlikely occurring in the target class).

Based on these considerations, the method for discovery of discriminant rules is outlined as follows.

1. Collect the relevant set of data respectively into the target class and the contrasting classes.

2. Extract the *prime target relation* (the prime relation corresponding to the initial relation in the target class) in a similar way as the attribute-oriented induction at learning characteristic rules. Then generalize the concepts of the initial relation(s) in the contrasting class(es) to the same level as those in the prime target relation, which results in the *prime contrasting relation*(*s*).

3. To generate qualitative discriminant rules, compare the tuples in the prime target relation against those in the prime contrasting relation(s), and mark those tuples which overlap in both relations. The discriminating properties are represented by the unmarked tuples.

4. To generate quantitative discriminant rules, compute the discriminating weight for each property and output the properties whose discriminating weight is close to 1 (together with the discriminating weight).

This method summarizes the major differences of a target class from a contrasting class at a high concept level. Notice that machine learning techniques, such as a decision-tree method like ID3 (Quinlan 1986) and C4.5 (Quinlan 1992), have been used to classify objects and find discriminating behaviors. To these algorithms, the attribute-oriented induction can be viewed as a preprocessing stage which performs a preliminary generalization using domain knowledge (such as concept hierarchies) to make rules expressible at a relatively high concept level to achieve both processing efficiency and representation clarity. Examples in ID3 or C4.5 algorithms assume that concepts to be classified are already at a relatively high level, such as "mild" (temperature) and "high" (humidity) (Quinlan 1986), which may not be the cases for the actual data stored in large databases

but can be easily achieved by attribute-oriented generalization. After such generalization, one may choose either presenting directly the characteristic and/or discriminant rules as described above (which are in the relational form, desirable for some applications, and allowing the same tuples appearing in different classes but with weights associated), or integrating with an ID3-like algorithm for further concept classification and compact rule generation.

16.3.2 Mining cluster description rules

In a large database, it is often desirable to cluster data according to data semantics (called *conceptual clustering* (Michalski and Stepp 1983)) and associate (description) rules with such clusters. For example, students in a university can be clustered (i.e., classified) based on different attribute(s), including major, age, height, academic performance, etc., however, clustering on one attribute may generate more meaningful and concise descriptions than that on another. It is important to determine desired classifying attribute(s) and desired concept level(s) for conceptual clustering in a particular set of data.

Attribute-oriented induction can be applied to conceptual clustering as follows (Han, Cai and Cercone 1991). First, the interested set of data is collected by a database query, and the attribute-oriented induction generalizes the set of data to an appropriate high concept level, which results in a generalized relation. Second, this generalized relation is mapped into a table with the generalized attribute values as rows and columns, and the occurrence frequency (i.e., *count*) as the table entry value. After filtering out the entries with very low frequency (treated as noise), the nonempty entries can be merged into bigger slots and mapped into a set of candidate schemes with each corresponding to a set of rules to be generated. Based on a combined measurement of *sparseness* (the problem space covered by the generated rules but not covered by the evidence data) and *complexity* of the generated rules, the candidate scheme which minimizes the combined measurement can be selected as the *classifying scheme*, and the set of rules which distinguish one class from others can be associated with the class, as *cluster description rules*. Such a process iterates until no further classification effort is necessary.

This iterative process leads to the generation of a preferred hierarchy for the interested set of data, with a set of description rules associated with each node of the hierarchy.

An alternative (but of the same spirit) conceptual clustering method is to first perform attribute-oriented induction as described above, and then construct a decision tree from the generalized relation using a method similar to ID3 (Quinlan 1986) and associate the corresponding rules with each node in the tree.

16.3.3 Mining multiple-level association rules

An *association rule* represents an association relationship among a set of patterns (values) in a database (Agrawal, Imielinski and Swami 1993). For example, an association rule discovered from a shopping transaction database may disclose that a customer who buys milk will have 90% possibility to buy bread as well.

The association relationship can be characterized by *support* σ and *confidence* φ. The support of pattern A occurring in set S is denoted as $\sigma(A/S)$. The confidence that "if A occurs in S, then B occurs in S" can be derived by the formula, $\varphi((A \rightarrow B)/S) = \sigma((A \wedge B)/S)/\sigma(A/S)$.

In a database with a large number of concrete patterns, there may not exist many *strong* associations (i.e., with *large* support and *high* confidence) between concrete patterns. However, there many exist many interesting association relationships for the patterns at high concept levels. For example, although there may not exist noticeable associations between particular brands, categories, and sizes of milk and bread, there may exist strong evidence to associate milk and bread together. Therefore, it is interesting to generalize patterns to different concept levels and discover multiple-level association rules.

The following algorithm demonstrates a top-down technique (from a high level to lower ones) for deriving multiple-level association rules: find all the large patterns $A_1, \ldots, A_m$ in set S (i.e., $\sigma(A_1 \wedge \ldots \wedge A_m/S) \geq \sigma_l$, where σ_l is the minimum support at level l). Notice that a user may set different σ_l's at different levels based on data characteristics or search interests and adjust them interactively based on the patterns returned.

1. Collect the task-relevant set S from the database by a query, and generalize the data in S to a high concept level by attribute-oriented induction.

2. At this concept level, search S to find large single-item sets. Combine them to form candidate large two-item sets and check against the database to find large two-item sets, and so on, until all the large k-item sets are found.

3. Go down one level l, find all the large item sets at level l based on the same method of Step 2 for all the children of the large items at level $l-1$, i.e., one level higher.

4. Progressively walk down the hierarchy to find large item sets at each level until no large one-item set is found or it reaches the leaf level of the hierarchy.

The method first finds large patterns at a high concept level and progressively deepens the search to find such patterns among their descendants at lower concept levels until it reaches the primitive-level concepts in the database. Notice that the search at a lower concept level is confined to only those patterns whose ancestors have passed the testing at their corresponding higher concept levels. Also, the corresponding rules in the form

of "$A_1 \wedge A_k \rightarrow A_m$" (and their confidence ratios) can be derived straightforwardly. A detailed algorithm is presented in (Han and Fu 1995).

16.4 Towards knowledge discovery in advanced database systems

In this section, we briefly examine the extension of attribute-oriented induction to knowledge discovery in advanced and/or special purpose databases, including object-oriented, deductive, and spatial databases.

16.4.1 Attribute-oriented induction in object-oriented databases

An object-oriented database organizes a large set of complex objects into classes which are in turn organized into class/subclass hierarchies with rich data semantics. Each object in a class is associated with (1) an object-identifier, (2) a set of attributes which may contain sophisticated data structures, set- or list- valued data, class composition hierarchies, multimedia data, etc., and (3) a set of methods which specify the computational routines or rules associated with the object class.

Knowledge discovery in object-oriented databases can be performed by first generalizing a set of complex data components into relatively simple generalized concepts and then applying the attribute-oriented induction method to generalize them into a generalized prime relation (Han, Nishio and Kawano 1994).

To facilitate generalization of complex data objects, it is important to implement efficiently a set of generalization operators on the components of object-oriented databases, including object identifiers, unstructured and structure values, class composition hierarchies, inherited and derived data, methods, etc., illustrated as follows.

1. **Generalization of object identifiers**: Since objects belong to certain classes which in turn are organized into certain class/subclass hierarchies, an object identifier can be first generalized to its corresponding lowest subclass name which can in turn be generalized to a higher level class/subclass name by climbing up the class/subclass hierarchy.

2. **Generalization of structured data**: The generalization of complex structure-valued data, such as set-valued or list-valued data and data with nested structures, can be explored in several ways in order to extract interesting patterns. Take set-valued attributes as an example. A set-valued attribute can be typically generalized in two ways: (1) generalization of each value in a set into its corresponding higher level concepts, or (2) derivation of the general behavior of a set, such as the number of elements in the set, the types or value ranges in the set, the weighted average for numerical data, etc. For example, the *hobby* of a person, such as {*tennis, hockey, chess, violin, nintendo*}, can be

generalized into a set of high level concepts, such as {*sports, music, computer_games*}, or into 5 (the number of hobbies in the set), etc.

3. **Generalization on inherited and derived properties**: In an OODB, an attribute or a method in an object class may not be explicitly specified in the class itself but is inherited from its higher level classes or derivable (computable) by applying some deduction rules or methods. From the knowledge discovery point of view, it is unnecessary to distinguish the data stored within the class from those inherited from its superclass(es) or derived using rules or methods. As long as the set of relevant data are collected by query processing, the knowledge discovery process will treat them in the same way as those stored in the object class and perform generalization accordingly.

4. **Generalization on class composition hierarchies**: An attribute value in an object may itself be an object, whose attributes may in turn be composed by other objects, thus forming a class composition hierarchy. Generalization on a class composition hierarchy can be viewed as generalization on a set of (possibly infinite, if the nesting is recursive) nested structured data. Although the reference to a composite object may traverse via a long sequence of references along the corresponding class composition hierarchy, the longer sequence it traverses, the weaker semantic linkage between the original object and its referenced ones in most cases. Therefore, in order to discover relatively interesting knowledge, generalization should be performed only on the composite objects closely related to the currently focused class(es) but not on those which have only remote and rather weak semantic linkages.

Besides object-oriented data generalization, attribute-oriented induction can also be applied to schema formation and schema evolution in object-oriented databases. This can be done by first generalizing object-oriented data and then performing conceptual clustering on the generalized data, a process similar to mining cluster description rules discussed in Section 16.3.2.

16.4.2 Integration of knowledge discovery and deductive database techniques

A *deductive database* is a database which consists of data, rules, and integrity constraints and performs deductive reasoning on large sets of data. Integration of deduction and induction mechanisms not only leads to discovery of new knowledge in deductive databases but also enhances the power of knowledge discovery mechanisms.

There are at least four cases that a knowledge discovery mechanism may interact with the deductive database technology.

1. **Data derived by applying deduction rules:** With the availability of deduction rules, new data can be derived by deductive query processing. Knowledge discovery

can be performed on the data so derived, which is the case that deduction rules are used for the collection of the task-relevant set of data; whereas the knowledge discovery mechanism itself remains intact.

2. **Rules representing (partially) generalized data:** With the availability of conceptual hierarchy and knowledge discovery mechanisms, it is possible that a deduction rule plays the role of a generalized rule which defines a portion of generalized data. If a knowledge discovery query is to discover the regularity of a set of data which is a superset of the data covered by the rule, the rule can be merged with the corresponding (intermediate) generalized relation of the other portion of the data for further generalization.

3. **Deduction rule-specified concept hierarchy:** A rule may also be used to specify concept hierarchies (Dhar and Tuzhilin 1993). For example, a rule which claims that "*a student is excellent if (s)he is an undergraduate student with GPA* $\geq$ *3.5 or if (s)he is a graduate student with GPA* $\geq$ *3.75*" defines a portion of the concept hierarchy for the attribute GPA. Such a rule can be stored as a part of the concept hierarchy and be used in generalization (Cheung, Fu and Han 1994).

4. **Rule-directed knowledge discovery:** Certain rules can be used for directing a knowledge discovery process. For example, "$A \wedge B \rightarrow C$" can be used as a meta-rule to inform the system to find rules in such a logical form from a large set of data (Shen et al. 1994). A rule "*buy milk* $\rightarrow$ *buy bread*" can be used as a guide for finding similar kinds of rules, which may result in the association of credibility with the rule, such as "*if one buys milk, there is a 90% possibility that (s)he will buy bread*" or deepening the rule by associating it with lower level concepts, such as "*if one buys 2% Dairyland milk, there is a 45% possibility that (s)he will buy Wonder whole wheat bread.*"

16.4.3 Attribute-oriented induction in spatial databases

A *spatial database* system stores, manages and manipulates spatial (i.e., space-related) data and, in most cases, nonspatial data as well for geographic information systems and many other applications (Elmasri and Navathe 1994).

Spatial data mining refers to the discovery of interesting relationships between spatial and nonspatial data and/or among spatial data in spatial databases. An example for the former (spatial-nonspatial) is the finding of weather patterns related to different geographic regions; whereas that for the latter is the finding of general spatial relationships (e.g., nearby, far-away, etc.) between service stations and highways.

The discovery of general relationships between spatial and nonspatial data can be performed by attribute-oriented induction in two ways: (1) spatial-dominant generalization, and (2) nonspatial-dominant generalization.

The *spatial-dominant generalization* first performs generalization on task-relevant spatial data using user/expert-provided spatial data hierarchies (such as geographic regions, etc.), hierarchical spatial data structures (such as R-trees, quad-trees, etc.), or spatial clustering algorithms (Ng and Han 1994) to generalize or cluster spatial data, and then generalize the nonspatial data associated with each spatial cluster or partition.

On the other hand, the *nonspatial-dominant generalization* first performs attribute-oriented induction on task-relevant nonspatial data, which generalizes nonspatial primitives to high level concepts and merges database tuples and their associated spatial data pointers into a small number of generalized entries. Then spatial merging operations can be performed on the spatial data in each generalized entry to form a small number of merged spatial entities (e.g., regions) for each entry.

Example 16.4 Weather patterns related to geographic regions in British Columbia can be discovered by either spatial-dominant or nonspatial-dominant generalization.

The former merges scattered small regions into several major regions in the province (based on the administrative or geographic hierarchies), which partitions accordingly the associated nonspatial data, such as temperature and precipitation. Attribute-oriented induction can then be performed on each partition to extract general weather patterns associated with each generalized region.

The latter generalizes the climate data, such as temperature and precipitation, into a small number of generalized entries (such as wet and cold), or concrete range values, which collects the associated spatial data pointers when merging nonspatial components. Then spatial merging (or region growing) can be performed on the spatial entities in the same generalized entry.

The selection of the two methods can be based on the availability of background knowledge, generalization efficiency, and the preference of generalization results. □

Besides concept tree ascension and spatial data handling techniques, aggregation and approximation play an important role in spatial data mining. When generalizing scattered regions into clustered regions, it is necessary not only to merge or cluster the regions of similar types within the same general class but also to ignore some scattered regions with different types if they are unimportant to the study. For example, different pieces of land for different agricultural usages can be merged into one large piece of land by spatial merge. However, such an agricultural land may contain highways, houses, small stores, etc. If the majority land is used for agriculture, the scattered spots for other purposes can be ignored, and the whole region can be claimed as an agricultural area by approximation.

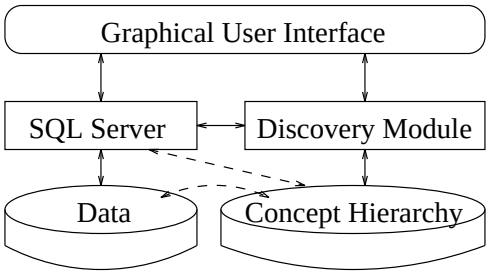

Figure 16.2
The configuration of the DBMINER system

16.5 Integration of attribute-oriented induction with DBMS

Based on the methodology of attribute-oriented induction, a data mining system prototype, DBMINER, has been developed in our research and integrated with commercial relational database systems.

Fig. 16.2 shows the configuration of the DBMINER system, which consists of several major modules, including graphical user interface, knowledge discovery, concept hierarchy manipulation, and SQL server modules. These modules are tightly coupled together for efficient, set-oriented processing.

The system has the following main features. First, it integrates data mining with relational database systems and provides an SQL-like knowledge discovery interface for relational system-based data retrieval and relational operation-based data generalization, and thus facilitates its integration with the existing commercial relational database systems. Second, a graphical user interface has been constructed for the specification of parameters and for flexible and interactive data mining. Moreover, the system can perform automatic generation of concept hierarchies for numeric data and dynamic adjustment of hierarchies based on data statistics.

The prototype is implemented in C on the UNIX system, using a client-server architecture, accessing relational databases via a Sybase server. A PC database system-based version is also under construction and experimentation. Experiments have been conducted on relational databases, including NSERC research grant databases and several very large ones provided by industry firms, with good performance and satisfactory results. The system has also been used for experiments on spatial data mining. The system discovers successfully the characteristic rules and discriminant rules. The extraction of other kinds of rules in relational database systems is currently under implementation and experimentation.

16.6 Conclusions

We have explored in this chapter the power of an interesting induction method, the attribute-oriented induction, in data mining. As a set-oriented, generalization-based data mining method, attribute-oriented induction extracts effectively different kinds of knowledge rules, including characteristic rules, discriminant rules, cluster description rules, and multiple-level association rules in relational database systems. Furthermore, it is shown that the method is efficient, robust, with wide applications, and extensible to knowledge discovery in advanced database systems, including object-oriented, deductive, and spatial databases. However, the method is a generalization-based technique and requires the availability of some background knowledge (such as concept hierarchies). Thus it may not be suitable for mining knowledge without generalization or without background knowledge.

Knowledge discovered by attribute-oriented induction has interesting applications at querying database knowledge, cooperative query answering, multiple layered database construction, and semantic query optimization, as reported in our previous studies (Han, Fu and Ng 1994). In general, data mining provides a powerful tool for automatic generation and verification of knowledge in the construction of large knowledge-bases.

Knowledge discovery represents an important and promising direction in the development of data and knowledge-base systems. Our preliminary study of the attribute-oriented induction mechanism leads to an efficient implementation of a data mining system. Besides further advance of our study on the attribute-oriented induction methodology, we are also investigating other data mining methods (Agrawal, Imielinski and Swami 1993; Chu and Chiang 1994; Kivinen and Mannila 1994; Matheus, Chan and Piatetsky-Shapiro 1993; Michalski et al. 1992; Piatetsky-Shapiro and Frawley 1991; Shen et al. 1994; Uthurusamy, Fayyad and Spangler 1991; Ziarko 1994) and working on the integrated method for discovery of various kinds of knowledge in different kinds of database and information systems.

Acknowledgements

Research is partially supported by the Natural Sciences and Engineering Research Council of Canada under the grant OGP0037230 and by the Networks of Centres of Excellence Program (with the participation of PRECARN association) under the grant IRIS-HMI-5. The authors would like to express their thanks to Yandong Cai and Yue Huang for their implementations of the early versions of the DBLearn system and their experiments of the attribute-oriented induction method for knowledge discovery in relational databases. Also, the authors would like

to express their thanks to them and to Colin Carter, Nick Cercone, David Cheung, Son Dao, Ada Fu, Randy Goebel, Howard Hamilton, Xiaohua Hu, Hiroyuki Kawano, Rizwan Kheraj, Kris Koperski, Peter Leung, Ling Liu, Wei Lu, Gabor Melli, Wendy Moore, Raymond T. Ng, Shojiro Nishio, Beng-Chin Ooi, Simon Tang, Wei Wang, and Osmar R. Zaïane for their discussions on the research issues related to knowledge discovery in databases, and finally, to Gregory Piatetsky-Shapiro whose comments have substantially improved the quality of this chapter.

Bibliography

Agrawal, R., Imielinski, T. and Swami, A. 1993. Mining association rules between sets of items in large databases. In Proc. 1993 ACM-SIGMOD Int. Conf. Management of Data, 207–216, Washington, D.C.: ACM Press.

Cai, Y., Cercone, N., and Han, J. 1991. Attribute-oriented induction in relational databases. In *Knowledge Discovery in Databases*, ed. G. Piatetsky-Shapiro and W. J. Frawley, 213–228. AAAI/MIT Press.

Cheung, D.W., Fu, A. W.-C. and Han., J. 1994. Knowledge discovery in databases: A rule-based attribute-oriented approach. In Proc. 1994 Int'l Symp. on Methodologies for Intelligent Systems, 164-173, Charlotte, North Carolina.

Chu, W. W. and Chiang, K. 1994. Abstraction of high level concepts from numerical values in databases. In Proc. AAAI'94 Workshop on Knowledge Discovery in Databases (KDD'94), 37–48, Seattle, WA.: AAAI Press.

Dhar, V. and Tuzhilin, A. 1993. Abstract-driving pattern discovery in databases. *IEEE Trans. Knowledge and Data Engineering*, 5(6): 926–938.

Elmasri, R. and Navathe, S. B. 1994. *Fundamentals of Database Systems, 2nd ed.* Benjamin/Cummings.

Fisher, D. H. 1987. Knowledge acquisition via incremental conceptual clustering. *Machine Learning*, 2:139–172.

Han, J., Cai, Y., and Cercone, N. 1991. Concept-based data classification in relational databases. In Proc. 1991 AAAI Workshop on Knowledge Discovery in Databases, 77–94, Anaheim, CA.: AAAI Press

Han, J., Cai, Y., and Cercone, N. 1993. Data-driven discovery of quantitative rules in relational databases. *IEEE Trans. Knowledge and Data Engineering*, 5:29–40.

Han, J. and Fu, Y. 1994. Dynamic generation and refinement of concept hierarchies for knowledge discovery in databases. In Proc. AAAI'94 Workshop on Knowledge Discovery in Databases (KDD'94, 157–168, Seattle, WA.: AAAI Press

Han, J. and Fu, Y. 1995. Discovery of multiple-level association rules from large databases. In Proc. 1995 Int. Conf. Very Large Data Bases, 420-431, Zürich, Switzerland.

Han, J., Fu, Y., and Ng, R. 1994. Cooperative query answering using multiple-layered databases. In Proc. 2nd Int. Conf. Cooperative Information Systems, 47–58, Toronto, Canada.

Han, J., Nishio, S., and Kawano, H. 1994. Knowledge discovery in object-oriented and active databases. In *Knowledge Building and Knowledge Sharing*, ed. F. Fuchi and T. Yokoi, 221–230. Ohmsha, Ltd. and IOS Press.

Kivinen, J. and Mannila, H. 1994. The power of sampling in knowledge discovery. In Proc. 13th ACM Symp. Principles of Database Systems, 77–85, Minneapolis, MN.: ACM Press.

Matheus, C., Chan, P. K. and Piatetsky-Shapiro, G. 1993. Systems for knowledge discovery in databases. *IEEE Trans. Knowledge and Data Engineering*, 5(6): 903–913.

Michalski, R. S. 1983. A theory and methodology of inductive learning. In *Machine Learning: An Artificial Intelligence Approach, Vol. 1*, ed. Michalski et al., 83–134. Morgan Kaufmann.

Michalski, R. S., Carbonell, J. G. and Mitchell, T. M. 1986. *Machine Learning, An Artificial Intelligence Approach, Vol. 2.* Morgan Kaufmann.

Michalski, R. S., Kerschberg, L., Kaufman, K. A. and Ribeiro, J. S. 1992. Mining for knowledge in databases: The INLEN architecture, initial implementation and first results. *J. Int. Info. Systems*, 1:85–114.

Michalski, R. S. and Stepp, R. 1983. Automated construction of classifications: Conceptual clustering versus numerical taxonomy. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 5:396–410.

Ng, R. and Han, J. 1994. Efficient and effective clustering method for spatial data mining. In Proc. 1994 Int. Conf. Very Large Data Bases, 144–155, Santiago, Chile.

Piatetsky-Shapiro, G. and Frawley, W. J. 1991. *Knowledge Discovery in Databases.* AAAI/MIT Press.

Piatetsky-Shapiro, G. 1991. Discovery, analysis, and presentation of strong rules. In *Knowledge Discovery in Databases*, ed. G. Piatetsky-Shapiro and W. J. Frawley, 229–238. AAAI/MIT Press.

Quinlan, J. R. 1992. *C4.5: Programs for Machine Learning.* Morgan Kaufmann.

Quinlan, J. R. 1986. Induction of decision trees. *Machine Learning*, 1:81–106.

Shen, W., Mitbander, B., Ong, K. and Zaniolo, C. 1994. Using metaqueries to integrate inductive learning and deductive database technology. In Proc. AAAI'94 Workshop on Knowledge Discovery in Databases (KDD'94), 335–346, Seattle, WA.: AAAI Press.

Uthurusamy, R., Fayyad, U. M. and Spangler, S. 1991. Learning useful rules from inconclusive data. In *Knowledge Discovery in Databases*, ed. G. Piatetsky-Shapiro and W. J. Frawley, 141–158. AAAI/MIT Press.

Ziarko, W. 1994. *Rough Sets, Fuzzy Sets and Knowledge Discovery*. Springer-Verlag.

Zytkow J. and Baker, J. 1991. Interactive mining of regularities in databases. In *Knowledge Discovery in Databases*, ed. G. Piatetsky-Shapiro and W. J. Frawley, 31–54. AAAI/MIT Press.