Data Mining: An Overview from a Database Perspective

Ming-Syan Chen, *Senior Member, IEEE*, Jiawei Han, *Senior Member, IEEE*, and Philip S. Yu, *Fellow, IEEE*

Abstract—Mining information and knowledge from large databases has been recognized by many researchers as a key research topic in database systems and machine learning, and by many industrial companies as an important area with an opportunity of major revenues. Researchers in many different fields have shown great interest in data mining. Several emerging applications in information providing services, such as data warehousing and on-line services over the Internet, also call for various data mining techniques to better understand user behavior, to improve the service provided, and to increase the business opportunities. In response to such a demand, this article is to provide a survey, from a database researcher's point of view, on the data mining techniques developed recently. A classification of the available data mining techniques is provided, and a comparative study of such techniques is presented.

Index Terms—Data mining, knowledge discovery, association rules, classification, data clustering, pattern matching algorithms, data generalization and characterization, data cubes, multiple-dimensional databases.

1 Introduction

Recently, our capabilities of both generating and collecting data have been increasing rapidly. The widespread use of bar codes for most commercial products, the computerization of many business and government transactions, and the advances in data collection tools have provided us with huge amounts of data. Millions of databases have been used in business management, government administration, scientific and engineering data management, and many other applications. It is noted that the number of such databases keeps growing rapidly because of the availability of powerful and affordable database systems. This explosive growth in data and databases has generated an urgent need for new techniques and tools that can intelligently and automatically transform the processed data into useful information and knowledge. Consequently, *data mining* has become a research area with increasing importance [30], [70], [76].

Data mining, which is also referred to as *knowledge discovery in databases*, means a process of nontrivial extraction of implicit, previously unknown and potentially useful information (such as knowledge rules, constraints, regularities) from data in databases [70]. There are also many other terms, appearing in some articles and documents, carrying a similar or slightly different meaning, such as *knowledge mining from databases, knowledge extraction, data archaeology, data dredging, data analysis*, etc. By knowledge discovery in databases, interesting knowledge, regularities, or high-level information can be extracted from the relevant sets of data in databases and be investigated from different angles, and large databases thereby serve as rich and reliable sources for knowledge generation and verification. Mining information and knowledge from large databases has been recognized by many researchers as a key research topic in database systems and machine learning, and by many industrial companies as an important area with an opportunity of major revenues. The discovered knowledge can be applied to information management, query processing, decision making, process control, and many other applications. Researchers in many different fields, including database systems, knowledge-base systems, artificial intelligence, machine learning, knowledge acquisition, statistics, spatial databases, and data visualization, have shown great interest in data mining. Furthermore, several emerging applications in information providing services, such as on-line services and World Wide Web, also call for various data mining techniques to better understand user behavior, to meliorate the service provided, and to increase the business opportunities.

In response to such a demand, this article is to provide a survey on the data mining techniques developed in several research communities, with an emphasis on database-oriented techniques and those implemented in applicative data mining systems. A classification of the available data mining techniques is also provided, based on the kinds of databases to be mined, the kinds of knowledge to be discovered, and the kinds of techniques to be adopted. This survey is organized according to one classification scheme: the kinds of knowledge to be mined.

- *M.-S. Chen is with the Electrical Engineering Department, National Taiwan University, Taipei, Taiwan, Republic of China. E-mail: mschen@cc.ee.ntu.edu.tw.*
- *J. Han is with the School of Computing Science, Simon Fraser University, British Columbia, V5A 1S6, Canada. E-mail: han@cs.sfu.ca.*
- *P.S. Yu is with the IBM Thomas J. Watson Research Center, PO Box 704, Yorktown, NY 10598. E-mail: psyu@watson.ibm.com.*

Manuscript accepted Aug. 28, 1996.
For information on obtaining reprints of this article, please send e-mail to: transkde@computer.org, and reference IEEECS Log Number K96075.

1.1 Requirements and Challenges of Data Mining

In order to conduct effective data mining, one needs to first examine what kind of features an applied knowledge discovery system is expected to have and what kind of challenges one may face at the development of data mining techniques.

1) *Handling of different types of data.*

   Because there are many kinds of data and databases used in different applications, one may expect that a knowledge discovery system should be able to perform effective data mining on different kinds of data. Since most available databases are relational, it is crucial that a data mining system performs efficient and effective knowledge discovery on *relational* data. Moreover, many applicable databases contain *complex data types,* such as structured data and complex data objects, hypertext and multimedia data, spatial and temporal data, transaction data, legacy data, etc. A powerful system should be able to perform effective data mining on such complex types of data as well. However, the diversity of data types and different goals of data mining make it unrealistic to expect one data mining system to handle all kinds of data. Specific data mining systems should be constructed for knowledge mining on specific kinds of data, such as systems dedicated to knowledge mining in relational databases, transaction databases, spatial databases, multimedia databases, etc.

2) *Efficiency and scalability of data mining algorithms.*

   To effectively extract information from a huge amount of data in databases, the knowledge discovery algorithms must be *efficient* and *scalable* to large databases. That is, the running time of a data mining algorithm must be predictable and acceptable in large databases. Algorithms with exponential or even medium-order polynomial complexity will not be of practical use.

3) *Usefulness, certainty, and expressiveness of data mining results.*

   The discovered knowledge should accurately portray the contents of the database and be useful for certain applications. The imperfectness should be expressed by measures of uncertainty, in the form of approximate rules or quantitative rules. Noise and exceptional data should be handled elegantly in data mining systems. This also motivates a systematic study of measuring the quality of the discovered knowledge, including interestingness and reliability, by construction of statistical, analytical, and simulative models and tools.

4) *Expression of various kinds of data mining requests and results.*

   Different kinds of knowledge can be discovered from a large amount of data. Also, one may like to examine discovered knowledge from different views and present them in different forms. This requires us to express both the data mining requests and the discovered knowledge in *high-level languages* or graphical user interfaces so that the data mining task can be specified by nonexperts and the discovered knowledge can be understandable and directly usable by users. This also requires the discovery system to adopt expressive knowledge representation techniques.

5) *Interactive mining knowledge at multiple abstraction levels.*

   Since it is difficult to predict what exactly could be discovered from a database, a high-level data mining query should be treated as a probe which may disclose some interesting traces for further exploration. *Interactive discovery* should be encouraged, which allows a user to interactively refine a data mining request, dynamically change data focusing, progressively deepen a data mining process, and flexibly view the data and data mining results at multiple abstraction levels and from different angles.

6) *Mining information from different sources of data.*

   The widely available local and wide-area computer network, including Internet, connect many sources of data and form huge distributed, heterogeneous databases. Mining knowledge from different sources of formatted or unformatted data with diverse data semantics poses new challenges to data mining. On the other hand, data mining may help disclose the high-level data regularities in heterogeneous databases which can hardly be discovered by simple query systems. Moreover, the huge size of the database, the wide distribution of data, and the computational complexity of some data mining methods motivate the development of *parallel and distributed data mining algorithms.*

7) *Protection of privacy and data security.*

   When data can be viewed from many different angles and at different abstraction levels, it threatens the goal of protecting data security and guarding against the invasion of privacy. It is important to study when knowledge discovery may lead to an invasion of privacy, and what security measures can be developed for preventing the disclosure of sensitive information.

Notice that some of these requirements may carry conflicting goals. For example, the goal of protection of data security may conflict with the requirement of interactive mining of multiple-level knowledge from different angles. Moreover, this survey addresses only some of the above requirements, with an emphasis on the efficiency and scalability of data mining algorithms. For example, the handling of different types of data are confined to relational and transactional data, and the methods for protection of privacy and data security are not addressed (some discussions could be found elsewhere, such as [22], [63]). Nevertheless, we feel that it is still important to present an overall picture regarding to the requirements of data mining.

2 An Overview of Data Mining Techniques

Since data mining poses many challenging research issues, direct applications of methods and techniques developed in related studies in machine learning, statistics, and database systems cannot solve these problems. It is necessary to perform dedicated studies to invent new data mining methods or develop integrated techniques for efficient and effective data mining. In this sense, data mining itself has formed an independent new field.

2.1 Classifying Data Mining Techniques

There have been many advances on researches and developments of data mining, and many data mining techniques and systems have recently been developed. Different classification schemes can be used to categorize data mining methods and systems based on the kinds of databases to be studied, the kinds of knowledge to be discovered, and the kinds of techniques to be utilized, as shown below.

- *What kinds of databases to work on.*
  A data mining system can be classified according to the kinds of databases on which the data mining is performed. For example, a system is a relational data miner if it discovers knowledge from relational data, or an object-oriented one if it mines knowledge from object-oriented databases. In general, a data miner can be classified according to its mining of knowledge from the following different kinds of databases: relational databases, transaction databases, object-oriented databases, deductive databases, spatial databases, temporal databases, multimedia databases, heterogeneous databases, active databases, legacy databases, and the Internet information-base.
- *What kind of knowledge to be mined.*
  Several typical kinds of knowledge can be discovered by data miners, including association rules, characteristic rules, classification rules, discriminant rules, clustering, evolution, and deviation analysis, which will be discussed in detail in the next subsection.

  Moreover, data miners can also be categorized according to the abstraction level of its discovered knowledge which may be classified into generalized knowledge, primitive-level knowledge, and multiple-level knowledge. A flexible data mining system may discover knowledge at multiple abstraction levels.
- *What kind of techniques to be utilized.*
  Data miners can also be categorized according to the underlying data mining techniques. For example, it can be categorized according to the driven method into autonomous knowledge miner, data-driven miner, query-driven miner, and interactive data miner. It can also be categorized according to its underlying data mining approach into generalization-based mining, pattern-based mining, mining based on statistics or mathematical theories, and integrated approaches, etc.

Among many different classification schemes, this survey follows mainly one classification scheme: *the kinds of knowledge to be mined* because such a classification presents a clear picture on different data mining requirements and techniques. Methods for mining different kinds of knowledge, including association rules, characterization, classification, clustering, etc., are examined in depth. For mining a particular kind of knowledge, different approaches, such as machine learning approach, statistical approach, and large database-oriented approach, are compared, with an emphasis on the database issues, such as efficiency and scalability.

2.2 Mining Different Kinds of Knowledge from Databases

Data mining is an application-dependent issue and different applications may require different mining techniques to cope with. In general, the kinds of knowledge which can be discovered in a database are categorized as follows.

Mining association rules in transactional or relational databases has recently attracted a lot of attention in database communities [4], [7], [39], [57], [66], [73], [78]. The task is to derive a set of strong association rules in the form of "$A_1 \wedge \ldots \wedge A_m \Rightarrow B_1 \wedge \ldots \wedge B_n$," where A_i (for $i \in \{1, \ldots, m\}$) and B_j (for $j \in \{1, \ldots, n\}$) are sets of attribute-values, from the relevant data sets in a database. For example, one may find, from a large set of transaction data, such an association rule as if a customer buys (*one brand of*) milk, he/she usually buys (*another brand of*) bread in the same transaction. Since mining association rules may require to repeatedly scan through a large transaction database to find different association patterns, the amount of processing could be huge, and performance improvement is an essential concern. Efficient algorithms for mining association rules and some methods for further performance enhancements will be examined in Section 3.

The most popularly used data mining and data analysis tools associated with database system products are *data generalization* and *summarization* tools, which carry several alternative names, such as on-line analytical processing (OLAP), multiple-dimensional databases, data cubes, data abstraction, generalization, summarization, characterization, etc. Data generalization and summarization presents the general characteristics or a summarized high-level view over a set of user-specified data in a database. For example, the general characteristics of the technical staffs in a company can be described as a set of characteristic rules or a set of generalized summary tables. Moreover, it is often desirable to present generalized views about the data at multiple abstraction levels. An overview on multilevel data generalization, summarization, and characterization is presented in Section 4.

Another important application of data mining is the ability to perform classification in a huge amount of data. This is referred to as mining classification rules. *Data classification* is to classify a set of data based on their values in certain attributes. For example, it is desirable for a car dealer to classify its customers according to their preference for cars so that the sales persons will know whom to approach, and catalogs of new models can be mailed directly to those customers with identified features so as to maximize the business opportunity. Some studies in classification rules will be reviewed in Section 5.

In Section 6, we discuss the techniques on *data clustering*. Basically, data clustering is to group a set of data (without a predefined class attribute), based on the conceptual clustering principle: *maximizing the intraclass similarity and minimizing the interclass similarity*. For example, a set of commodity objects can be first clustered into a set of classes and then a set of rules can be derived based on such a classification. Such clustering may facilitate *taxonomy formation*, which means the organization of observations into a hierarchy of classes that group similar events together.

Temporal or spatial-temporal data constitutes a large portion of data stored in computers [9], [80]. Examples of this type of database include: financial database for stock price index, medical databases, and multimedia databases, to name a few. Searching for similar patterns in a temporal or spatial-temporal database is essential in many data mining operations [1], [3], [56] in order to discover and predict the risk, causality, and trend associated with a specific pattern. Typical queries for this type of database include identifying companies with similar growth patterns, products with similar selling patterns, stocks with similar price movement, images with similar weather patterns, geological features, environmental pollutions, or astrophysical patterns. These queries invariably require similarity matches as opposed to exact matches. The approach of pattern-based similarity search is reviewed in Section 7.

In a distributed information providing environment, documents or objects are usually linked together to facilitate interactive access. Understanding user access patterns in such environments will not only help improving the system design but also be able to lead to better marketing decisions. Capturing user access patterns in such environments is referred to as *mining path traversal patterns*. Notice, however, that since users are traveling along the information providing services to search for the desired information, some objects are visited because of their locations rather than their content, showing the very difference between the traversal pattern problem and others which are mainly based on customer transactions. The capability of mining path traversal patterns is discussed in Section 8.

In addition to the issues considered above, there are certainly many other aspects on data mining that are worth studying. It is often necessary to use a data mining query language or graphical user interface to specify the interesting subset of data, the relevant set of attributes, and the kinds of rules to be discovered. Moreover, it is often necessary to perform interactive data mining to examine, transform, and manipulate intermediate data mining results, focus at different concept levels, or test different kinds of thresholds. Visual representation of data and knowledge may facilitate interactive knowledge mining in databases.

3 Mining Association Rules

Given a database of sales transactions, it is desirable to discover the important associations among items such that the presence of some items in a transaction will imply the presence of other items in the same transaction. A mathematical model was proposed in [4] to address the problem of mining association rules. Let $I = \{i_1, i_2, ..., i_m\}$ be a set of literals, called items. Let D be a set of transactions, where each transaction T is a set of items such that $T \subseteq I$. Note that the quantities of items bought in a transaction are not considered, meaning that each item is a binary variable representing if an item was bought. Each transaction is associated with an identifier, called TID. Let X be a set of items. A transaction T is said to contain X if and only if $X \subseteq T$. An association rule is an implication of the form $X \Rightarrow Y$, where $X \subset I$, $Y \subset I$, and $X \cap Y = \phi$. The rule $X \Rightarrow Y$ holds in the transaction set D with *confidence c* if c% of transactions in D that contain X also contain Y. The rule $X \Rightarrow Y$ has *support* s in the transaction set D if s% of transactions in D contain $X \cup Y$.

Confidence denotes the strength of implication and support indicates the frequencies of the occurring patterns in the rule. It is often desirable to pay attention to only those rules which may have reasonably large support. Such rules with high confidence and strong support are referred to as *strong rules* in [4], [68]. The task of mining association rules is essentially to discover strong association rules in large databases. In [4], [7], [66], the problem of mining association rules is decomposed into the following two steps:

1) Discover the large itemsets, i.e., the sets of itemsets that have transaction support above a predetermined minimum support s.
2) Use the large itemsets to generate the association rules for the database.

It is noted that the overall performance of mining association rules is determined by the first step. After the large itemsets are identified, the corresponding association rules can be derived in a straightforward manner. Efficient counting of large itemsets is thus the focus of most prior work. Here, algorithms Apriori and DHP, developed in [7], [66], respectively, are described to illustrate the nature of this problem.

3.1 Algorithm Apriori and DHP

Consider an example transaction database given in Fig. 1.[1] In each iteration (or each pass), Apriori constructs a candidate set of large itemsets, counts the number of occurrences of each candidate itemset, and then determines large itemsets based on a predetermined minimum support [7]. In the first iteration, Apriori simply scans all the transactions to count the number of occurrences for each item. The set of candidate 1-itemsets, C_1, obtained is shown in Fig. 2. Assuming that the minimum transaction support required is two (i.e., $s = 40\%$), the set of large 1-itemsets, L_1, composed of candidate 1-itemsets with the minimum support required, can then be determined. To discover the set of large 2-itemsets, in view of the fact that any subset of a large itemset must also have minimum support, Apriori uses $L_1 * L_1$ to generate a candidate set of itemsets C_2 where * is an operation for concatenation in this case.[2] C_2 consists of

$$\binom{|L_1|}{2}$$

2-itemsets. Next, the four transactions in D are scanned and the support of each candidate itemset in C_2 is counted. The middle table of the second row in Fig. 2 represents the result from such counting in C_2. The set of large 2-itemsets, L_2, is therefore determined based on the support of each candidate 2-itemset in C_2.

1. This example database is extracted from [7].
2. For the general case, $L_k * L_k = \{X \cup Y \mid X, Y \in L_k, |X \cap Y| = k - 1\}$.

Database D

TID	Items
100	A C D
200	B C E
300	A B C E
400	B E

Fig. 1. An example transaction database for data mining.

The set of candidate itemsets, C_3, is generated from L_2 as follows. From L_2, two large 2-itemsets with the same first item, such as $\{BC\}$ and $\{BE\}$, are identified first. Then, Apriori tests whether the 2-itemset $\{CE\}$, which consists of their second items, constitutes a large 2-itemset or not. Since $\{CE\}$ is a large itemset by itself, we know that all the subsets of $\{BCE\}$ are large and then $\{BCE\}$ becomes a candidate 3-itemset. There is no other candidate 3-itemset from L_2. Apriori then scans all the transactions and discovers the large 3-itemsets L_3 in Fig. 2. Since there is no candidate 4-itemset to be constituted from L_3, Apriori ends the process of discovering large itemsets.

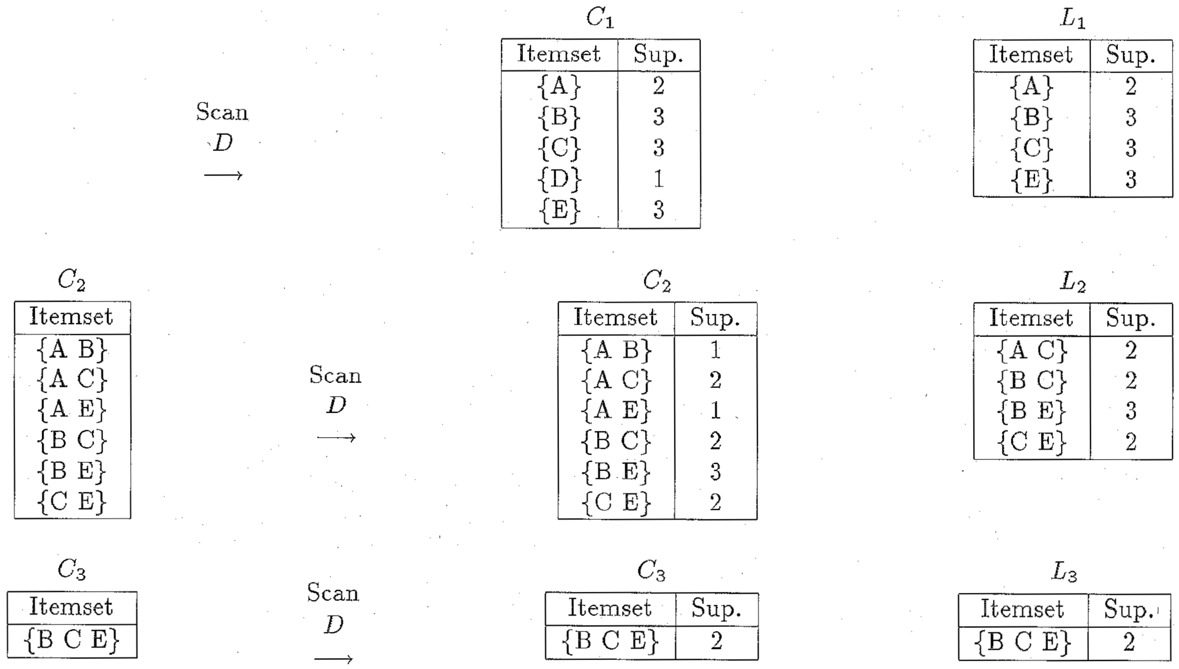
Similar to Apriori, DHP in [66] also generates candidate k-itemsets from L_{k-1}. However, DHP employs a hash table, which is built in the previous pass, to test the eligibility of a k-itemset. Instead of including all k-itemsets from $L_{k-1} * L_{k-1}$ into C_k, DHP adds a k-itemset into C_k only if that k-itemset is hashed into a hash entry whose value is larger than or equal to the minimum transaction support required. As a result, the size of candidate set C_k can be reduced significantly. Such a filtering technique is particularly powerful in reducing the size of C_2. DHP also reduces the database size progressively by not only trimming each individual transaction size but also pruning the number of transactions in the database. We note that both DHP and Apriori are iterative algorithms on the large itemset size in the sense that the large k-itemsets are derived from the large $(k-1)$-itemsets. These large itemset counting techniques are in fact applicable to dealing with other data mining problems [8], [19].

3.2 Mining Generalized and Multiple-Level Association Rules

In many applications, interesting associations among data items often occur at a relatively high concept level. For example, the purchase patterns in a transaction database may not show any substantial regularities at the primitive data level, such as at the bar-code level, but may show some interesting regularities at some high concept level(s), such as milk and bread. Therefore, it is important to study mining association rules at a generalized abstraction level [78] or at multiple levels [39].

Information about multiple abstraction levels may exist in database organizations. For example, a class hierarchy [50] may be implied by a combination of database attributes, such as *day, month, year*. It may also be given explicitly by users or experts, such as *Alberta* $\subset$ *Prairies*.

Consider the class hierarchy in Fig. 3 for example. It could be difficult to find substantial support of the purchase patterns at the primitive concept level, such as the bar codes of 1 *gallon Dairyland* 2% *milk* and 1 *lb Wonder wheat bread*. However, it could be easy to find 80% *of customers that purchase* milk *may also purchase* bread. Moreover, it could be informative to also show that 70% *of people buy* wheat bread *if they buy* 2% milk. The association relationship in the latter statement is expressed at a lower concept level but often carries more specific and concrete informa-


$\xrightarrow{\text{Scan } D}$

C_1

Itemset	Sup.
{A}	2
{B}	3
{C}	3
{D}	1
{E}	3

L_1

Itemset	Sup.
{A}	2
{B}	3
{C}	3
{E}	3

C_2

Itemset
{A B}
{A C}
{A E}
{B C}
{B E}
{C E}

$\xrightarrow{\text{Scan } D}$

C_2

Itemset	Sup.
{A B}	1
{A C}	2
{A E}	1
{B C}	2
{B E}	3
{C E}	2

L_2

Itemset	Sup.
{A C}	2
{B C}	2
{B E}	3
{C E}	2

C_3

Itemset
{B C E}

$\xrightarrow{\text{Scan } D}$

C_3

Itemset	Sup.
{B C E}	2

L_3

Itemset	Sup.
{B C E}	2

Fig. 2. Generation of candidate itemsets and large itemsets.

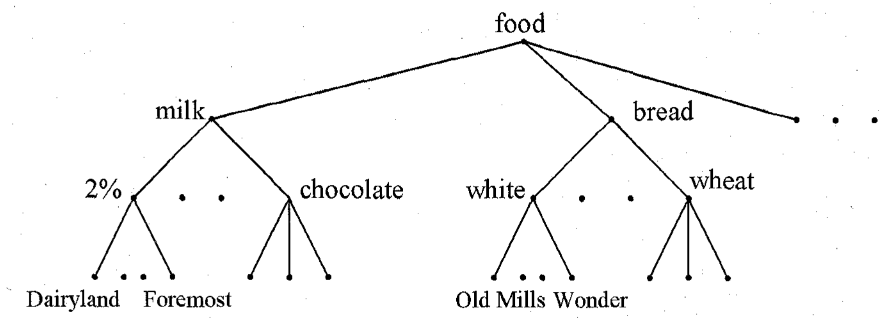


Fig. 3. An example of concept hierarchies for mining multiple-level association rules.

tion than that in the former. Therefore, it is important to mine association rules at a generalized abstraction level or at multiple concept levels.

In [39], the notion of mining multiple-level association rules are introduced: Low-level associations will be examined only when their high-level parents are large at their corresponding levels, and different levels may adopt different minimum support thresholds. Four algorithms are developed for efficient mining of association rules based on different ways of sharing of multiple-level mining processes and reduction of the encoded transaction tables. In [78], methods for mining associations at generalized abstraction level are studied by extension of the Apriori algorithm.

Besides mining multiple-level and generalized association rules, the mining of quantitative association rules [79] and meta-rule guided mining of association rules in relational databases [33], [75] are also studied recently, with efficient algorithms developed.

3.3 Interestingness of Discovered Association Rules

Notice that not all the discovered *strong* association rules (i.e., passing the minimum support and minimum confidence thresholds) are interesting enough to present.

For example, consider the following case of mining the survey results in a school of 5,000 students. A retailer of breakfast cereal surveys the students on the activities they engage in the morning. The data show that 60% of students (i.e., 3,000 students) play basketball, 75% of students (i.e., 3,750 students) eat cereal, and 40% of them (i.e., 2,000 students) both play basketball and eat cereal. Suppose that a data mining program for discovering association rules is run on the data with the following settings: the minimal student support is 2,000 and the minimal confidence is 60%. The following association rule will be produced: "(play basketball) $\Rightarrow$ (eat cereal)," since this rule contains the minimal student support and the corresponding confidence

$$\frac{2000}{3000} = 0.66$$

is larger than the minimal confidence required. However, the above association rule is misleading since the overall percentage of students eating cereal is 75%, even larger than 66%. That is, playing basketball and eating cereals are in fact negatively associated; i.e., being involved in one decreases the likelihood of being involved in the other. Without fully understanding this aspect, one could make wrong business decisions from the rules derived.

To filter out such kind of misleading associations, one may define that an association rule "A $\Rightarrow$ B" is *interesting* if its confidence exceeds a certain measure, or formally,

$$\frac{P(A \cap B)}{P(A)}$$

is greater than d, a suitable constant. However, the simple argument we used in the example above suggests that the right heuristic to measure association should be

$$\frac{P(A \cap B)}{P(A)} - P(B) > d,$$

or alternatively, $P(A \cap B) - P(A) * P(B) > k$, where k is a suitable constant. The expressions above essentially represent tests of statistical independence. Clearly, the factor of statistical dependence among various user behaviors analyzed has to be taken into consideration for the determination of the usefulness of association rules.

There have been some interesting studies on the interestingness or usefulness of discovered rules, such as [68], [78], [77]. The notion of interestingness on discovered generalized association rules is introduced in [78]. The subjective measure of interestingness in knowledge discovery is studied in [77].

3.4 Improving the Efficiency of Mining Association Rules

Since the amount of the processed data in mining association rules tends to be huge, it is important to devise efficient algorithms to conduct mining on such data. In this section, some techniques to improve the efficiency of mining association rules are presented.

3.4.1 Database Scan Reduction

In both Apriori and DHP, C_3 is generated from $L_2 * L_2$. In fact, a C_2 can be used to generate the candidate 3-itemsets. Clearly, a C_3' generated from $C_2 * C_2$, instead of from $L_2 * L_2$, will have a size greater than $|C_3|$ where C_3 is generated from $L_2 * L_2$. However, if $|C_3'|$ is not much larger than $|C_3|$, and both C_2 and C_3' can be stored in main memory, we can find L_2 and L_3 together when the next scan of the database is performed, thereby saving one round of database scan. It can be seen that using this concept, one can determine all L_ks by as few as two scans of the database (i.e., one initial scan to determine L_1 and a final scan to determine all other large itemsets), assuming that C'_k for $k \geq 3$ is generated from C'_{k-1} and all C'_k for $k > 2$ can be kept in the memory. This technique is called *scan-reduction.* In [19], the technique of scan-reduction was utilized and shown to result in prominent performance improvement. Clearly, such a technique is applicable to both Apriori and DHP.

3.4.2 Sampling: Mining with Adjustable Accuracy

Several applications require mining the transaction data to capture the customer behavior in a very frequent basis. In those applications, the efficiency of data mining could be a more important factor than the requirement for a complete accuracy of the results. In addition, in several data mining applications the problem domain could only be vaguely defined. Missing some marginal cases with confidence and support levels at the borderline may have little effect on the quality of the solution to the original problem. Allowing imprecise results can in fact significantly improve the efficiency of the mining algorithms. As the database size increases nowadays, sampling appears to be an attractive approach to data mining. A technique of relaxing the support factor based on the sampling size is devised in [65] to achieve the desired level of accuracy. As shown in [65], the relaxation factor, as well as the sample size, can be properly adjusted so as to improve the result accuracy while minimizing the corresponding execution time, thereby allowing us to effectively achieve a design trade-off between accuracy and efficiency with two control parameters. As a means to improve efficiency, sampling has been used in [78] for determining the cut-off level in the class hierarchy of items to collect occurrence counts in mining generalized association rules. Sampling was discussed in [57] as a justification for devising algorithms and conducting experiments with data sets of small sizes.

3.4.3 Incremental Updating of Discovered Association Rules

Since it is costly to find the association rules in large databases, incremental updating techniques should be developed for maintenance of the discovered association rules to avoid redoing data mining on the whole updated database.

A database may allow frequent or occasional updates and such updates may not only invalidate some existing strong association rules but also turn some weak rules into strong ones. Thus it is nontrivial to maintain such discovered rules in large databases. An incremental updating technique is developed in [21] for efficient maintenance of discovered association rules in transaction databases with data insertion. The major idea is to reuse the information of the old large itemsets and to integrate the support information of the new large itemsets in order to substantially reduce the pool of candidate sets to be re-examined.

3.4.4 Parallel Data Mining

It is noted that data mining in general requires progressive knowledge collection and revision based on a huge transaction database. How to achieve efficient parallel data mining is a very challenging issue, since, with the transaction database being partitioned across all nodes, the amount of internode data transmission required for reaching global decisions can be prohibitively large, thus significantly compromising the benefit achievable from parallelization. For example, in a shared nothing type parallel environment like SP2 [44], each node can directly collect information only from its local database partition, and the process to reach a global decision from partial knowledge collected at individual nodes could itself be complicated and communication intensive. An algorithm for parallel data mining, called PDM, was reported in [67]. Under PDM, with the entire transaction database being partitioned across all nodes, each node will employ a hashing method to identify candidate k-itemsets (i.e., itemsets consisting of k items) from its local database. To reduce the communication cost incurred, a *clue-and-poll* technique was devised in [67] to resolve the uncertainty due to the partial knowledge collected at each node by judiciously selecting a small fraction of the itemsets for the information exchange among nodes.

4 Multilevel Data Generalization, Summarization, and Characterization

Data and objects in databases often contain detailed information at primitive concept levels. For example, the "item" relation in a "sales" database may contain attributes about the primitive level item information such as item number, item name, date made, price, etc. It is often desirable to summarize a large set of data and present it at a high concept level. For example, one may like to summarize a large set of the items related to some sales to give a general description. This requires an important functionality in data mining: *data generalization.*

Data generalization is a process which abstracts a large set of relevant data in a database from a low concept level to relatively high ones. The methods for efficient and flexible generalization of large data sets can be categorized into two approaches:

1) data cube approach [35], [43], [83], [84], and
2) attribute-oriented induction approach [37], [40].

4.1 Data Cube Approach

The data cube approach has a few alternative names or a few variants, such as, "multidimensional databases," "materialized views," and "OLAP (On-Line Analytical Processing)." The general idea of the approach is to materialize certain expensive computations that are frequently inquired, especially those involving aggregate functions, such as count, sum, average, max, etc., and to store such materialized views in a multidimensional database (called a "data cube") for decision support, knowledge discovery, and many other applications. Aggregate functions can be precomputed according to the grouping by different sets or subsets of attributes. Values in each attribute may also be grouped into a hierarchy or a lattice structure. For example, "date" can be grouped into "day," "month," "quarter," "year," or "week," which form a lattice structure. Generalization and specialization can be performed on a multiple dimensional data cube by "roll-up" or "drill-down" operations, where a roll-up operation reduces the number of dimensions in a data cube or generalizes attribute values to high-level concepts, whereas a drill-down operation does the reverse. Since many aggregate functions may often need to be computed repeatedly in data analysis, the storage of precomputed results in a multiple dimensional data cube may ensure fast response time and flexible views of data from different angles and at different abstraction levels.

For example, a relation with the schema *"sales(part, supplier, customer, sale_price)"* can be materialized into a set of eight views as shown in Fig. 4 (extracted from [43]), where psc indicates a view consisting of aggregate function values (such as *total_sales*) computed by grouping three attributes *part, supplier,* and *customer, p* indicates a view consisting of the corresponding aggregate function values computed by grouping *part* alone, etc.

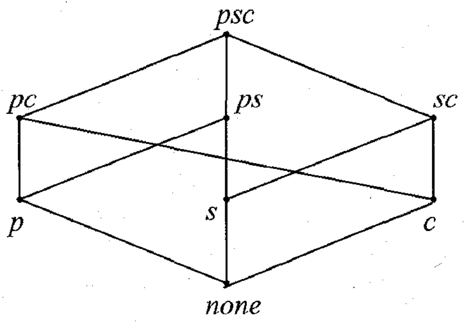


Fig. 4. Eight views of data cubes for sales information.

There are commonly three choices in the implementation of data cubes:

1) *physically materialize the whole data cube,*
2) *materialize nothing,* and
3) *materialize only part of the data cube.*

The problem of materialization of a selected subset of a very large number of views can be modeled as a lattice of views. A recent study [43] has shown that a greedy algorithm, which, given what views have already been materialized, selects for materializing those views that offer the most improvement in average response time, is able to lead to results within 63% of those generated by the optimal algorithm in all cases. As a matter of fact, in many realistic cases, the difference between the greedy and optimal solutions is essentially negligible.

Data cube approach is an interesting technique with many applications [83]. Techniques for indexing multiple dimensional data cubes and for incremental updating of data cubes at database updates have also been studied [83], [86]. Notice that data cubes could be quite sparse in many cases because not every cell in each dimension may have corresponding data in the database. Techniques should be developed to handle sparse cubes efficiently. Also, if a query contains constants at even lower levels than those provided in a data cube (e.g., a query refers time in the unit of "hour" whereas the lowest concept level on time in the cube is "day"), it is not clear how to make the best use of the precomputed results stored in the data cube.

4.2 Attribute-Oriented Induction Approach

The data warehousing approach which uses materialized views are "off-line" database computation which may not correspond to the most up-to-date database contents. An alternative, on-line, generalization-based data analysis technique, is called *attribute-oriented induction approach* [37], [40]. The approach takes a data mining query expressed in an SQL-like data mining query language and collects the set of relevant data in a database. Data generalization is then performed on the set of relevant data by applying a set of data generalization techniques [37], [40], [60], including attribute-removal, concept-tree climbing, attribute-threshold control, propagation of counts and other aggregate function values, etc. The generalized data is expressed in the form of a generalized relation on which many other operations or transformations can be performed to transform generalized data into different kinds of knowledge or map them into different forms [40]. For example, drill-down or roll-up operations can be performed to view data at multiple abstraction levels [36]; the generalized relation can be mapped into summarization tables, charts, curves, etc., for presentation and visualization; characteristic rules which summarize the generalized data characteristics in quantitative rule forms can be extracted; discriminant rules which contrast different classes of data at multiple abstraction levels can be extracted by grouping the set of comparison data into contrasting classes before data generalization; classification rules which classify data at different abstraction levels according to one or a set of classification attributes can be derived by applying a decision tree classifier [72] on the generalized data [42]; and association rules which associate a set of generalized attribute properties in a logic implication rule by integration of attribute-oriented induction and the methods for mining association rules [7], [39], [66], [78]. Moreover, statistical pattern discovery can also be performed using attribute-oriented induction [24].

The core of the attribute-oriented induction technique is on-line data generalization which is performed by first examining the data distribution for each attribute in the set of relevant data, calculating the corresponding abstraction level that data in each attribute should be generalized to, and then replacing each data tuple with its corresponding generalized tuple. The generalization process scans the data set only once and collects the aggregate values in the corresponding generalized relation or data cube. It is a highly efficient technique since the worst-case time complexity of the process is $O(n)$, if a cube structure is used (desirable when the data cube is dense), or $O(n \log(p))$, if a generalized relation is used (desirable when the corresponding cube is rather sparse), where n is the number of tuples in the set of relevant data and p is the size (i.e., number of tuples) of the generalized relation [40].

To support multiple-level data mining, especially the drill-down operations, the attribute-oriented induction technique should generalize the set of relevant data to a minimal abstraction level and maintain such a *minimally generalized relation* (if expressed by a relational structure) or a *minimally generalized cube* (if expressed by a cube structure) to facilitate the traversals among multiple abstraction spaces. The roll-up of a generalized relation may simply start with this relation; however, the drill-down of the relation may start with the minimal generalized relation and perform data generalization to the corresponding abstraction levels [36].

The essential background knowledge applied in attribute-oriented induction is concept hierarchy (or lattice) associated with each attribute [37]. Most concept hierarchies are stored implicitly in databases. For example, a set of attributes in *address* (*number, street, city, province, country*) in a database schema represents the concept hierarchies of the attribute *address*. A set of attributes in a data relation, though seemingly no strong semantic linkages exist, may also form concept hierarchies (or lattices) among their supersets or subsets. For example, in the schema *item*(*id, name, category, producer, date_made, cost, price*), "{*category, producer, date_made*} ⊂ {*category, date_made*}" indicates the former forms a lower level concept than the latter. Moreover, rules and view definitions can also be used as the definitions of concept hierarchies [24]. Conceptual hierarchies for numerical or ordered attributes can be generated automatically based on the analysis of data distributions in the set of relevant data [38]. Moreover, a given hierarchy may not be best suited for a particular data mining task. Therefore, such hierarchies should be adjusted dynamically in many cases based on the analysis of data distributions of the corresponding set of data [38].

An an example, one may use data mining facilities to study the general characteristics of the Natural Science and Engineering Research Council of Canada (NSERC) research grant database. To compare the research grants between '*British Columbia*' and '*Alberta*' (two neighbor provinces in Western Canada) in the discipline of '*Computer* (Science)' according to the attributes: *disc_code* (discipline code) and *grant_category*, the following data mining query can be specified in a data mining query language DMQL [42] as follows:

```
use NSERC95
mine discriminant rule for 'BC_Grants'
where O.Province = 'British Columbia'
in contrast to 'Alberta_Grants'
where O.Province = 'Alberta'
from Award A, Organization O, Grant_type G
where A.grant_code = G.grant_code and
    A.OrgID = O.OrgID and A.disc_code =
    'Computer'
related to disc_code, grant_category,
count(*)%
```

The execution of this data mining query generates Table 1, which presents the differences between the two provinces in terms of disc_code, grant_category and the number of the research grants. The column *support%* represents the number of research grants in this category vs. the total number of grants in its own province; where the *comparison%* represents the number of research grants in this category in one province vs. that of the other. For example, the first (composite) row indicates that for *Computer Science Infrastructure Grants* in the range of 40Ks to 60Ks, British Columbia takes 2.00% of its total number of Computer Science Grants, whereas Alberta takes 1.72%; however, in comparison between these two provinces, British Columbia takes 66.67% of the share (in number) whereas Alberta takes only 33.33%. Other rows have similar interpretations.

Notice with interactive data mining facilities, such as those provided in DBMiner, a user may perform roll-up or

TABLE 1
MINING DISCRIMINANT RULES: A COMPARISON OF RESEARCH GRANTS IN TWO PROVINCES

class	discipline	grant_category	amount_category	support%	comparison%
B.C.	Computer	Infrastructure Grant	40Ks-60Ks	2.00	66.67
Alberta				1.72	33.33
B.C.	Computer	Other	20Ks-40Ks	2.00	66.67
Alberta				1.72	33.33
B.C.	Computer	Other	60Ks-	2.00	50.00
Alberta				3.45	50.00
B.C.	Computer	Research Grant: Individual	0-20Ks	38.00	63.33
Alberta				37.93	36.67
B.C.	Computer	Research Grant: Individual	20Ks-40Ks	28.00	56.00
Alberta				37.93	44.00
B.C.	Computer	Research Grant: Individual	40Ks-60Ks	6.00	75.00
Alberta				3.45	25.00
B.C.	Computer	Research Grant: Individual	60Ks-	3.00	100.00
Alberta				0.00	0.00
B.C.	Computer	Scholarship	0-20Ks	19.00	76.00
Alberta				10.34	24.00

drill-down operations conveniently. For example, one may drill down on 'grant_category' to examine even finer grant categories such as 0-10Ks, 10-15Ks, 15-20Ks, etc., or roll-up on disc_code to group together *Infrastructure Grant, Research Grant: Individual*, etc., into one category '*Any* (Grant)'.

Overall, attribute-oriented induction is a technique for generalization of any subset of on-line data in a relational database and extraction of interesting knowledge from the generalized data. The generalized data may also be stored in a database, in the form of a generalized relation or a generalized cube, and be updated incrementally upon database updates [37]. The approach has been implemented in a data mining system, DBMiner, and been experimented in several large relational databases [40], [42]. The approach can also be extended to generalization-based data mining in object-oriented databases [41], spatial databases [53], [56], and other kinds of databases. The approach is designed for generalization-based data mining. It is not suitable for mining specific patterns at primitive concept levels although it may help guiding such data mining by first finding some traces at high concept levels and then progressively deepening the data mining process to lower abstraction levels [53].

5 Data Classification

Data classification is the process which finds the common properties among a set of objects in a database and classifies them into different *classes*, according to a classification model. To construct such a classification model, a sample database E is treated as the *training set*, in which each tuple consists of the same set of multiple attributes (or features) as the tuples in a large database W, and additionally, each tuple has a known class identity (label) associated with it. The objective of the classification is to first analyze the training data and develop an accurate description or a model for each class using the features available in the data. Such class descriptions are then used to classify future test data in the database W or to develop a better description (called *classification rules*) for each class in the database. Applications of classification include medical diagnosis, performance prediction, selective marketing, to name a few.

Data classification has been studied substantially in statistics, machine learning, neural networks, and expert systems [82], and is an important theme in data mining [30].

5.1 Classification Based on Decision Trees

A decision-tree-based classification method, such as [71], [72], has been influential in machine learning studies. It is a supervised learning method that constructs decision trees from a set of examples. The quality (function) of a tree depends on both the classification accuracy and the size of the tree. The method first chooses a subset of the training examples (a window) to form a decision tree. If the tree does not give the correct answer for all the objects, a selection of the exceptions is added to the window and the process continues until the correct decision set is found. The eventual outcome is a tree in which each leaf carries a class name, and each interior node specifies an attribute with a branch corresponding to each possible value of that attribute.

A typical decision tree learning system, ID-3 [71], adopts a top-down irrevocable strategy that searches only part of the search space. It guarantees that a simple, but not necessarily the simplest, tree is found. ID-3 uses an *information-theoretic approach* aimed at minimizing the expected number of tests to classify an object. The attribute selection part of ID-3 is based on the plausible assumption that the complexity of the decision tree is strongly related to the amount of information conveyed by this message. An information-based heuristic selects the attribute providing the highest information gain, i.e., the attribute which minimizes the information needed in the resulting subtrees to classify the elements. An extension to ID-3, C4.5 [72], extends the domain of classification from categorical attributes to numerical ones.

The ID-3 system [71] uses information gain as the evaluation functions for classification, with the following evaluation function,

$$i = \sum (p_i ln(p_i)),$$

where p_i is the probability that an object is in class i. There are many other evaluation functions, such as *Gini index, chi-square test*, and so forth [14], [52], [68], 82]. For example, for Gini index [14], [59], if a data set T contains examples from n classes, $gini(T)$ is defined as,

$$gini(T) = 1 - \sum p_i^2 .$$

where p_i is the relative frequency of class i in T. Moreover, there are also approaches for tranforming decision trees into rules [72] and transforming rules and trees into comprehensive knowledge structures [34].

There have been many other approaches on data classification, including statistical approaches [18], [26], 68], rough sets approach [87], etc. Linear regression and linear discriminant analysis techniques are classical statistical models [26]. Methods have also been studied for scaling machine learning algorithms by combining base classifiers from partitioned data sets [18]. There have also been some studies of classification techniques in the context of large databases [2], [10]. An interval classifier has been proposed in [2] to reduce the cost of decision tree generation. The neural network approach for classification and rule extraction in databases has also been studied recently [55].

5.2 Methods for Performance Improvement

Most of the techniques developed in machine learning and statistics may encounter the problem of scaling-up. They may perform reasonably well in relatively small databases but may suffer the problem of either poor performance or the reduction of classification accuracy when the training data set grows very large, even though a database system has been taken as a component in some of the above methods. For example, the interval classifier proposed in [2] uses database indices to improve only the efficiency of data retrieval but not the efficiency of classification since the classification algorithm itself is essentially an ID-3 algorithm.

A direct integration of attribute-oriented induction with the ID-3 algorithm may help discovery of classification rules at high abstraction levels [40]. It, though efficient, may reduce the classification accuracy since the classification interval may have been generalized to a rather high level. A multiple-level classification technique and a level adjustment and merge technique have been developed in DBMiner to improve the classification accuracy in large databases by the integration of attribute-oriented induction and classification methods [42].

Recently, Mehta et al. [59] has developed a fast data classifier, called supervised learning in QUEST (SLIQ), for mining classification rules in large databases. SLIQ is a decision tree classifier that can handle both numeric and categorical attributes. It uses a novel presorting technique in the tree growing phase. This sorting procedure is integrated with a breadth-first tree growing strategy to enable classification of disk-resident datasets. SLIQ also uses a new tree-pruning algorithm that is inexpensive, and results in compact and accurate trees. The combination of these techniques enables it to scale for large data sets and classify data sets irrespective of the number of classes, attributes, and examples. An approach, called meta-learning, was proposed in [17]. In [17], methods to learn how to combine several base classifiers, which are learned from subsets of data, were developed. Efficient scaling-up to larger learning problems can hence be achieved.

Notice that in most prior work on decision tree generation, a single attribute is considered at each level for the branching decision. However, in some classification tasks, the class identity in some cases is not so dependent on the value of a single attribute, but instead, depends upon the combined values of a set of attributes. This is particularly true in the presence of those attributes that have strong inference among themselves. In view of this, a two-phase method for multiattribute extraction was devised in [20] to improve the efficiency of deriving classification rules in a large training dataset. A feature that is useful in inferring the group identity of a data tuple is said to have a good *inference power* to that group identity. Given a large training set of data tuples, the first phase, referred to as *feature extraction phase*, is applied to a subset of the training database with the purpose of identifying useful features which have good inference power to group identities. In the second phase, referred to as *feature combination phase*, those features extracted from the first phase are combinedly evaluated and multiattribute predicates with strong inference power are identified. It is noted that the inference power can be improved significantly by utilizing multiple attributes in predicates, showing the advantage of using multiattribute predicates.

6 Clustering Analysis

The process of grouping physical or abstract objects into classes of similar objects is called *clustering* or *unsupervised classification*. Clustering analysis helps construct meaningful partitioning of a large set of objects based on a "divide and conquer" methodology which decomposes a large scale system into smaller components to simplify design and implementation.

As a data mining task, data clustering identifies clusters, or densely populated regions, according to some distance measurement, in a large, multidimensional data set. Given a large set of multidimensional data points, the data space is usually not uniformly occupied by the data points. Data clustering identifies the sparse and the crowded places, and hence discovers the overall distribution patterns of the data set.

Data clustering has been studied in statistics [18], [47], machine learning [31], [32], spatial database [11], and data mining [18], [27], [62], [85] areas with different emphases.

As a branch of statistics, clustering analysis has been studied extensively for many years, mainly focused on distance-based clustering analysis. Systems based on statistical classification methods, such as AutoClass [18] which uses a Bayesian classification method, have been used in clustering in real world databases with reported success.

The distance-based approaches assume that all the data points are given in advance and can be scanned frequently. They are global or semiglobal methods at the granularity of data points. That is, for each clustering decision, they inspect all data points or all currently existing clusters equally no matter how close or far away they are, and they use global measurements, which require scanning all data points or all currently existing clusters. Hence, they do not have linear scalability with stable clustering quality.

In machine learning, clustering analysis often refers to *unsupervised learning*, since which classes an object belongs to are not prespecified, or *conceptual clustering*, because the distance measurement may not be based on geometric distance, but be based on that a group of objects represents a certain conceptual class. One needs to define a measure of similarity between the objects and then apply it to determine classes. Classes are defined as collections of objects whose intraclass similarity is high and interclass similarity is low.

The method of clustering analysis in conceptual clustering is mainly based on probability analysis. Such approaches, represented by [31], [32], make the assumption that probability distributions on separate attributes are statistically independent of each other. This assumption is, however, not always true since correlation between attributes often exists. Moreover, the probability distribution representation of clusters makes it very expensive to update and store the clusters. This is especially so when the attributes have a large number of values since their time and space complexities depend not only on the number of attributes, but also on the number of values for each attribute. Furthermore, the probability-based tree (such as [31]) that is built to identify clusters is not height-balanced for skewed input data, which may cause the time and space complexity to degrade dramatically.

Clustering analysis in large databases has been studied recently in the database community.

6.1 Clustering Large Applications Based on Randomized Search

Ng and Han presented a clustering algorithm, CLARANS (Clustering Large Applications Based upon Randomized Search) [62], based on randomized search and originated from two clustering algorithms used in statistics, PAM (Partitioning Around Medoids) and CLARA (Clustering Large Applications) [48].

The CLARANS algorithm [62] integrates PAM and CLARA by searching only the subset of the data set but not confining itself to any sample at any given time. While CLARA has a fixed sample at every stage of the search, CLARANS draws a sample with some randomness in each step of the search. The clustering process can be presented as searching a graph where every node is a potential solution, i.e., a set of k medoids. The clustering obtained after replacing a single medoid is called the *neighbor* of the current clustering. If a better neighbor is found, CLARANS moves to the neighbor's node and the process is started again, otherwise the current clustering produces a local optimum. If the local optimum is found, CLARANS starts with new randomly selected nodes in search for a new local optimum. CLARANS has been experimentally shown to be more effective than both PAM and CLARA. The computational complexity of every iteration in CLARANS is basically linearly proportional to the number of objects [27], [62]. It should be mentioned that CLARANS can be used to find the most natural number of clusters k_{nat}. A heuristic is adopted in [62] to determine k_{nat}, which uses *silhouette coefficients*,[3] introduced by Kaufman and Rousseeuw [48]. CLARANS also enables the detection of outliers, e.g., points that do not belong to any cluster.

Based upon CLARANS, two spatial data mining algorithms were developed in a fashion similar to the attribute-oriented induction algorithms developed for spatial data mining [56], [37]: spatial *dominant approach*, SD(CLARANS) and *nonspatial dominant approach*, NSD(CLARANS). Both algorithms assume that the user specifies the type of the rule to be mined and relevant data through a learning request in a similar way as DBMiner [40]. Experiments show that the method can be used to cluster reasonably large data sets, such as houses in the Vancouver area, and the CLARAN algorithm outperforms PAM and CLARA.

6.2 Focusing Methods

Ester et al. [27] pointed out some drawbacks of the CLARANS clustering algorithm [62] and proposed new techniques to improve its performance.

First, CLARANS assumes that the objects to be clustered are all stored in main memory. This assumption may not be valid for large databases and disk-based methods could be required. This drawback is alleviated by integrating CLARANS with efficient spatial access methods, like R*-tree [11]. R*-tree supports the focusing techniques that Ester et al. proposed to reduce the cost of implementing CLARANS. Ester et al. showed that the most computationally expensive step of CLARANS is calculating the total distances between the two clusters. Thus, they proposed two approaches to reduce the cost of this step.

The first one is to reduce the number of objects considered. A *centroid query* returns the most central object of a leaf node of the R*-tree where neighboring points are stored. Only these objects are used to compute the medoids of the clusters. Thus, the number of objects taken for consideration is reduced. This technique is called *focusing on representative objects*. The drawback is that some objects, which may be better medoids, are not considered, but the sample is drawn in the way which still enables good quality of clustering.

The other technique to reduce computation is to restrict the access to certain objects that do not actually contribute to the computation, with two different focusing techniques: *focus on relevant clusters*, and *focus on a cluster*. Using the R*-tree structure, computation can be performed only on pairs of objects that can improve the quality of clustering instead of checking all pairs of objects as in the CLARANS algorithm.

Ester et al. applied the focusing on representative objects to a large protein database to find a segmentation of protein surfaces so as to facilitate the so-called *docking queries*. They reported that when the focusing technique was used the effectiveness (the average distance of the resulting clustering) decreased just from 1.5% to 3.2% whereas the efficiency (CPU time) increased by a factor of 50.

6.3 Clustering Features and CF Trees

R-trees are not always available and their construction may be time consuming. Zhang et al. [85] presented another algorithm, BIRCH (Balanced Iterative Reducing and Clustering), for clustering large sets of points. The method they presented is an incremental one with the possibility of adjustment of memory requirements to the size of memory that is available.

Two concepts, *Clustering Feature* and *CF tree*, are introduced.

A *Clustering Feature* CF is the triplet summarizing information about subclusters of points. Given N d-dimensional points in subcluster: $\{X_i\}$, CF is defined as

$$\mathrm{CF} = (N, \vec{LS}, SS)$$

where N is the number of points in the subcluster, $\vec{LS}$ is the linear sum on N points, i.e.,

$$\sum_{i=1}^{N} \vec{X}_i ,$$

and SS is the square sum of data points,

$$\sum_{i=1}^{N} \vec{X}_i^2 .$$

The *Clustering Features* are sufficient for computing clusters and they constitute an efficient storage information method as they summarize information about the subclusters of points instead of storing all points.

A CF tree is a balanced tree with two parameters: branching factor B and threshold T. The branching factor specifies the maximum number of children. The threshold parameter specifies the maximum diameter of subclusters stored at the leaf nodes. By changing the threshold value we can change the size of the tree. The nonleaf nodes are storing sums of their children's *CF*s, and thus, they summarize the information about their children. The *CF tree* is build dynamically as data points are inserted. Thus, the method is an incremental one. A point is inserted to the closest leaf entry (subcluster). If the diameter of the subcluster stored in the leaf node after insertion is larger than the threshold value,

3. It is a property of an object that specifies how much the object truly belongs to the cluster.

then, the leaf node and possibly other nodes are split. After the insertion of the new point, the information about it is passed towards the root of the tree. One can change the size of the *CF tree* by changing the threshold. If the size of the memory that is needed for storing the *CF tree* is larger than the size of the main memory, then a larger value of threshold is specified and the *CF tree* is rebuilt. The rebuild process is performed by building a new tree from the leaf nodes of the old tree. Thus, the process of rebuilding the tree is done without the necessity of reading all the points. Therefore, for building the tree, data has to be read just once. Some heuristics and methods are also introduced to deal with outliers and improve the quality of CF trees by additional scans of the data.

Zhang et al. claimed that any clustering algorithm, including CLARANS can be used with CF trees. The CPU and I/O costs of the BIRCH algorithm are of order $O(N)$. A good number of experiments reported in [85] show linear scalability of the algorithm with respect to the number of points, insensibility to the input order, and good quality of clustering of the data.

7 Pattern-Based Similarity Search

Next, we discuss data mining techniques based on pattern-based similarity search. When searching for similar patterns in a temporal or spatial-temporal database, two types of queries are usually encountered in various data mining operations:

- Object-relative similarity query (i.e., range query or similarity query) in which a search is performed on a collection of objects to find the ones that are within a user-defined distance from the queried object.
- All-pair similarity query (i.e., spatial join) where the objective is to find all the pairs of elements that are within a user-specified distance from each other.

Significant progress has recently been made in sequence matching for temporal databases [1], [5], [28], [29], [54], [57] and for speech recognition techniques such as dynamic time warping [81]. Two types of similarity queries for temporal data have emerged thus far: *whole matching* [1] in which the target sequence and the sequences in the database have the same length; *subsequence matching* [29] in which the target sequence could be shorter than the sequences in the database and the match can occur at any arbitrary point. Various approaches proposed in the literature differ in the following aspects. The first one is the similarity measure chosen. The second is whether the comparison is performed in time domain vs. transformed domain. The third is the generality of the approach whether subsequence of arbitrary length, scaling and translation are allowed in the matching. Finally, different techniques have been explored to reduce the number of comparisons or search space during mining.

7.1 Similarity Measures

Different similarity measures have been considered, mainly the Euclidean distance [1], [29], [28] and the correlation [54]. The Euclidean distance between two sequences is defined as follows. Let $\{x_i\}$ be the target sequence and $\{y_i\}$ be a sequence in the database. Let n be the length of $\{x_i\}$, N be the length of $\{y_i\}$, and assume $n \leq N$.

Consider only subsequences of length n of $\{y_i\}$. The Jth subsequence of length n of $\{y_i\}$ is denoted as $\{z_i^J\}$, where J is the offset. A metric for measuring the similarity between $\{x_i\}$ and $\{y_i\}$ can be defined as

$$\min_J \sum_{i=1}^{n} (x_i - K_J z_i^J)^2 \tag{7.1}$$

where K_J be a scaling factor. Assume that the subsequences are generated dynamically from the original sequence at query time. For each sequence of length N, a total of $N - n$ subsequences of length n need to be considered at query time.

The Euclidean distance is only meaningful for measuring the distance between two vectors with the same dimension. Another possible similarity measure is the correlation between two sequences as considered in [54]. This measure not only gives the relative similarity as a function of location but also eliminates the need to generate all the subsequences of given length n of each time series in the database.

The linear correlation between a target sequence $\{x_i\}$ and a sequence in the database $\{y_i\}$ is defined as

$$c_i = \frac{\sum_{j=1}^{n} x_j y_{i+j}}{\sqrt{\sum_{j=1}^{n} x_j^2}\sqrt{\sum_{j=1}^{N} y_j^2}}, \tag{7.2}$$

for $i = 1, \ldots, N + n - 1$. Such an operation can be expensive, especially for long target sequences $\{x_i\}$. Yet, when this is the case, the convolution theorem for Fourier transforms offers an appealing solution to the problem. First, zeros can be added at the end of the sequences $\{x_i\}$ and $\{y_i\}$, thus generating new sequences $\{x_i'\}$ and $\{y_i'\}$ both with length $l = N + n - 1$. Then, the Discrete Fourier Transforms (DFT), $\{X_i\}$ and $\{Y_i\}$ of $\{x_i'\}$ and $\{y_i'\}$, can be calculated. Finally, the correlation coefficient can be obtained by multiplying pointwise $\{X_i\}$ and $\{Y_i\}$ and inverting the result, as

$$c_i = \frac{\mathcal{F}^{-1}\left\{X_j^* Y_j\right\}}{\sqrt{\sum_{j=1}^{n} X_j^2}\sqrt{\sum_{j=1}^{N} Y_j^2}} \tag{7.3}$$

where $\{X_j'\}$ denotes the complex conjugate of X_j. The denominators of (7.2) and (7.3) are equal in virtue of Parseval's Theorem 1. If both $\{x_i\}$ and $\{y_i\}$ are properly normalized, the value of correlation coefficient c_i is a similarity measure of two sequences and ranges from –1 to 1, where 1 indicates a perfect match. With noisy signals, the correlation value is always smaller than one and the peaks of the sequence $\{c_i\}$ give the locations of possible matches.

7.2 Alternative Approaches

The straightforward approach for whole matching is to consider all of the data points of a sequence simultaneously. In [46], the concept of mapping an object to a point in the feature space and then applying multidimensional indexing method to perform similarity search is explored. A fast whole matching method generalizing this idea to sequence matching is proposed in [1], where the similarity between a stored sequence in the database and a target sequence is measured by the Euclidean distance between the features extracted from these two sequences in the Fourier domain.

Extending the above concept, an innovative approach is proposed in [29] to match subsequences by generating the first few Fourier coefficients of *all* possible subsequences of a given length for each stored sequence in the database. The idea is to match each sequence into a small set of multidimensional rectangles in the feature space. These rectangles can then be indexed by spatial access methods like R-tree family of indexes. This approach uses a moving window of a given length to slice each sequence at every possible position, and extracts features for each subsequence in the window. (The window size determines not only the minimum target sequence length that can be matched, but also the effectiveness of the search.) Thus each sequence maps into a trail in the feature space when sliding the moving window across the sequence. To reduce bookkeeping, each trail of feature vectors is divided into subtrails and each of these subtrails is represented with its minimum bounding rectangle.

Fourier transformation is by no means the best method of feature extraction. It is known that the a priori relative importance of the features can be optimally determined from the singular value decomposition (SVD) or the Karhunen-Loeve transformation on the covariance matrix of the collection of the time sequences [64]. A fast heuristic algorithm which approximates this dimensionality reduction process is proposed in [28]. Even with the significant dimensionality reduction resulting from the algorithm proposed in [28] and the compression of the representations of the subsequences in the feature space using the method proposed in [29], generating all subsequences from each time series is still a daunting task for a database of long sequences.

In [54], an enhancement is proposed of the feature extraction and matching method discussed in [29]. This new approach on subsequence matching is referred to as *HierarchyScan*. It uses the *correlation coefficient* as an alternative similarity measure between the target sequence and the stored sequences, and performs an adaptive scan on the extracted features of the stored sequences based on the target sequence. Because of the use of correlation as a similarity measure, the method is insensitive to the possible scale and phase differences between the stored sequences and the target sequence. It also eliminates the need of generating all subsequences from each stored sequence for subsequence matching. To improve the search efficiency, *HierarchyScan* first selects the subset of features with the greatest discriminating capability (i.e., the features with the largest filtering effect) to perform the matching (correlating). Only a small fraction of the sequences is expected to pass the test. Then the next most discriminating set of features is used for matching. This process is iterated until all of the features are exhausted. Compared to the method proposed in [29], *HierarchyScan* performs a hierarchical scan instead of using a tree structure for indexing. Different transformations were considered in [54].

In [5], another approach is introduced to determine all similar sequences in a set of sequences. It is also applicable to find all subsequences similar to a target sequence. The similarity measure considered is the Euclidean distance between the sequences and the matching is performed in the time domain. An R-tree type index is maintained for all subsequences of a given length (say, ω). The matching of all similar sequences consists of three steps. In the first step, all matching pairs of subsequences of length ω are identified using the R-tree type index. In the second step, for each pair of sequences, the matching pairs in step one are stitched into long subsequence matches. The third step linearly orders the long subsequence matches found in the second step to determine the one with the longest match length. This approach allows the amplitude of one of the two sequences to be scaled and its offset adjusted properly. It also permits nonmatching gaps in the matching subsequences.

8 Mining Path Traversal Patterns

In a distributed information providing environment, documents or objects are usually linked together to facilitate interactive access. Examples for such information providing environments include World Wide Web (WWW) [23] and on-line services, such as Prodigy, CompuServe, and America Online, where users, when seeking for information of interest, travel from one object to another via the corresponding facilities (i.e., hyperlinks and URL addresses) provided. Clearly, understanding user access patterns in such environments will not only help improving the system design (e.g., providing efficient access between highly correlated objects, better authoring design for pages, etc.) but also be able to lead to better marketing decisions (e.g., putting advertisements in proper places, better customer/user classification and behavior analysis, etc.). Capturing user access patterns in such environments is referred to as *mining path traversal patterns*. Note that although some efforts have been elaborated upon analyzing the user behavior [12], [15], [16], mining path traversal patterns is still in its infancy. This can be in part explained by the reason that these information providing services, though with great potential, are mostly in their infancy and their customer analysis may still remain in a coarser level such as the frequency of a page visited and user occupation/age study. Note that, as pointed out in [19], since users are traveling along the information providing services to search for the desired information, some objects are visited because of their locations rather than their content, showing the very difference between the traversal pattern problem and others which are mainly based on customer transactions. This unique feature of the traversal pattern problem unavoidably increases the difficulty of extracting meaningful information from a sequence of traversal data, and explains the reason that current web usage analysis products are only able to provide statistical information for traveling points, but not for traveling *paths*. However, as these information providing services are becoming increasingly popular nowadays, there is a growing demand for capturing user traveling behavior and improving the quality of such services.

In response to this demand, the problem of mining traversal patterns was explored in [19], where the proposed solution procedure consists of two steps. First, an algorithm was devised to convert the original sequence of log data into a set of traversal subsequences. Each traversal subsequence represents a maximal forward reference from the starting point of a user access. It is noted that this step of converting the original log sequence into a set of maximal forward references will filter out the effect of backward references which are mainly made for ease of traveling, and enable us to concentrate on mining meaningful user access sequences. Second, algorithms to determine the frequent traversal patterns, termed *large reference sequences*, from the maximal forward references obtained above were developed, where a large reference sequence is a reference sequence that appeared a sufficient number of times in the database. For example, suppose the traversal log contains the following traversal path for a user:

{A, B, C, D, C, B, E, G, H, G, W, A, O, U, O, V},

as shown in Fig. 5. Then, the set of maximal forward references for this user is

{ABCD, ABEGH, ABEGW, AOU, AOV}.

After maximal forward references for all users are obtained, the problem of finding frequent traversal patterns is mapped into the one of finding frequent occurring consecutive subsequences among all maximal forward references. After large reference sequences are determined, *maximal reference sequences* can then be obtained in a straightforward manner. A maximal reference sequence is a large reference sequence that is not contained in any other maximal reference sequence. Suppose that {*AB*, *BE*, *AD*, *CG*, *GH*, *BG*} is the set of large 2-references and {*ABE*, *CGH*} is the set of large 3-references. Then, the resulting maximal reference sequences are *AD*, *BG*, *ABE*, and *CGH*. A maximal reference sequence corresponds to a frequently accessed pattern in an information providing service.

It is noted that the problem of finding large reference sequences is similar to that of finding large itemsets for association rules [4] where a large itemset is a set of items appearing in a sufficient number of transactions. However, they are different from each other in that a reference sequence in mining traversal patterns has to be consecutive references in a maximal forward reference whereas a large itemset in mining association rules is just a combination of items in a transaction. As a consequence, the very difference between these two problems calls for the use of different algorithms for mining the knowledge required. As the popularity of internet explodes nowadays, it is expected that how to effectively discover knowledge on the web will be one of the most important data mining issues for years to come.

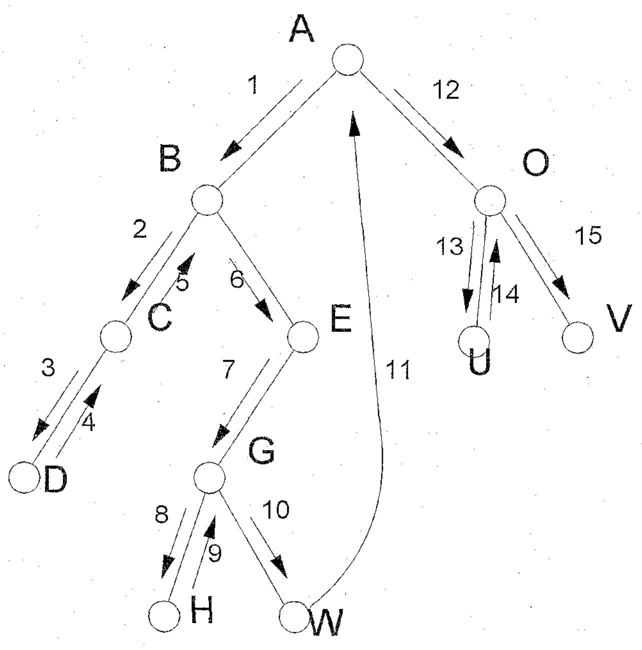


Fig. 5. An illustrative example for traversal patterns.

9 Summary

Data mining is a fast expanding field with many new research results reported and new systems or prototypes developed recently. Researchers and developers in many fields have contributed to the state of the art of data mining [30], [70]. Therefore, it is a challenging task to provide a comprehensive overview of the data mining methods within a short article. This article is an attempt to provide a reasonably comprehensive survey, from a database researcher's point of view, on the data mining techniques developed recently. An overview of data mining and knowledge discovery, from some data mining and machine learning researchers, has been performed recently [69]. The major difference of our survey from theirs is the focus of this survey is on the techniques developed by database researchers, with an emphasis on efficient methods for data mining in very large databases. A classification of the available data mining techniques is provided and a comparative study of such techniques has been presented.

Based on the diversity of data mining methods and rich functionalities of data mining investigated so far, many data mining systems or prototypes have been developed recently, some of which have been used successfully for mining knowledge in large databases. Here we briefly introduce some data mining systems reported in recent conferences and journals. However, this introduction is by no means complete. Appendices are welcome, and a comprehensive overview of such systems is necessary.

QUEST is a data mining system developed at the IBM Almaden Research Center by Agrawal et al. [6], which discovers various kinds of knowledge in large databases, including association rules [7], [78], [79], sequential patterns [8], classification rules [59], pattern matching and analysis [5], etc. KEFIR is a knowledge discovery system developed at the GTE Labs by Piatetsky-Shapiro [68] and

Piatetsky-Shapiro et al. [58] for the analysis of health care data. SKICAT is a knowledge discovery system, developed at Jet Propulsion Laboratory, which automatically detects and classifies sky objects from image data resulting from a major astronomical sky survey. DBMiner is a relational data mining system developed at Simon Fraser University by Han et al. [37] and Han and Fu [39], [40], for mining multiple kinds of rules at multiple abstraction levels, including characteristic rules, discriminant rules, association rules, classification rules, etc. KnowledgeMiner is a data mining system, developed by Shen et al. [75], which integrates data mining with deductive database techniques and using the meta-pattern to guide the data mining process. INLEN is a system, developed by Michalski et al. [61], which integrates multiple learning paradigms in data mining. Explora is a multipattern and multistrategy discovery assistant developed by Klösgen [52]. IMACS is a data mining system developed at AT&T Laboratory by Brachman and Anand [13], using sophisticated knowledge representation techniques. DataMine is system exploring interactive ad hoc query-directed data mining, developed by Imielinski and Virmani [45]. IDEA, developed at AT&T Laboratory by Selfridge et al. [74], performs interactive data explorations and analysis. There have been many other data mining systems reported by machine learning and statistics researchers. Moreover, data warehousing systems have been seeking data mining tools for further enhancement of data analysis capabilities, and it is likely to see a trend of integration of the techniques of data warehousing and data mining in the near future.

Besides the work done by database researchers, there have been fruitful results on data mining and knowledge discovery reported in many others fields. For example, researchers in statistics have developed many techniques which may benefit data mining [26]. Inductive logic programming [25] is a fast growing subfield in logic programming which is closely related to data mining. There have also been active studies on data mining using data visualization techniques [49] and visualization of data mining results [51]. For lack of space, the results and issues in these related fields cannot be presented in this short article. An overview of data mining and knowledge discovery in a broader spectrum is left as a future exercise.

As a young and promising field, data mining still faces many challenges and unsolved problems which pose new research issues for further study. For example, further development of efficient methods for mining multiple kinds of knowledge at multiple abstraction levels, a flexible and convenient data mining language or interface, the development of data mining techniques in advanced database systems, such as active databases, object-oriented databases, spatial databases, and multimedia databases, data mining in Internet information systems, the application of discovered knowledge, the integration of discovered and expert-defined knowledge, and the method to ensure security and protection of privacy in data mining, are important issues for further study.

ACKNOWLEDGMENTS

Jiawei Han was supported, in part, by Research Grant NSERC-A3723 from the Natural Sciences and Engineering Research Council of Canada, Research Grant NCE: IRIS/Precarn-HMI5 from the Networks of Centers of Excellence of Canada, and research grants from MPR Teltech Ltd. and Hughes Research Laboratories.

REFERENCES

[1] R. Agrawal, C. Faloutsos, and A. Swami, "Efficient Similarity Search in Sequence Databases," *Proc. Fourth Int'l Conf. Foundations of Data Organization and Algorithms*, Oct. 1993.

[2] R. Agrawal, S. Ghosh, T. Imielinski, B. Iyer, and A. Swami, "An Interval Classifier for Database Mining Applications," *Proc. 18th Int'l Conf. Very Large Data Bases*, pp. 560–573, Aug. 1992.

[3] R. Agrawal, T. Imielinski, and A. Swami, "Database Mining: A Performance Perspective," *IEEE Trans. Knowledge and Data Eng.*, pp. 914–925, Dec. 1993.

[4] R. Agrawal, T. Imielinski, and A. Swami, "Mining Association Rules between Sets of Items in Large Databases," *Proc. ACM SIGMOD*, pp. 207–216, May 1993.

[5] R. Agrawal, K.-I. Lin, H.S. Sawhney, and K. Shim, "Fast Similarity Search in the Presence of Noise, Scaling, and Translation in Time-Series Databases," *Proc. 21th Int'l Conf. Very Large Data Bases*, pp. 490–501, Sept. 1995.

[6] R. Agrawal, M. Mehta, J. Shafer, R. Srikant, A. Arning, and T. Bollinger, "The QUEST Data Mining System," *Proc. Int'l Conf. Data Mining and Knowledge Discovery (KDD '96)*, pp. 244-249, Portland, Ore., Aug. 1996.

[7] R. Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules in Large Databases," *Proc. 20th Int'l Conf. Very Large Data Bases*, pp. 478–499, Sept. 1994.

[8] R. Agrawal and R. Srikant, "Mining Sequential Patterns," *Proc. 11th Int'l Conf. Data Eng.*, pp. 3–14, Mar. 1995.

[9] K.K. Al-Taha, R.T. Snodgrass, and M.D. Soo, "Bibliography on Spatiotemporal Databases," *ACM SIGMOD Record*, vol. 22, no. 1, pp. 59–67, Mar. 1993.

[10] T.M. Anwar, H.W. Beck, and S.B. Navathe, "Knowledge Mining by Imprecise Querying: A Classification-Based Approach," *Proc. Eighth Int'l Conf. Data Eng.*, pp. 622–630, Feb. 1992.

[11] N. Beckmann, H.-P. Kriegel, R. Schneider, and B. Seeger, "The R*-Tree: An Efficient and Robust Access Method for Points and Rectangles," *Proc. ACM SIGMOD Int'l Conf. Management Data*, pp. 322–331, Atlantic City, N.J., June 1990.

[12] M. Bieber and J. Wan, "Backtracking in a Multiple-Window Hypertext Environment," *Proc. ACM European Conf. Hypermedia Technology*, pp. 158–166, 1994.

[13] R. Brachman and T. Anand, "The Process of Knowledge Discovery in Databases: A Human-Centered Approach," U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, eds., *Advances in Knowledge Discovery and Data Mining*, pp. 37–58. AAAI/MIT Press, 1996.

[14] L. Breiman, J. Friedman, R. Olshen, and C. Stone, *Classification of Regression Trees*. Wadsworth, 1984.

[15] E. Caramel, S. Crawford, and H. Chen, "Browsing in Hypertext: A Cognitive Study," *IEEE Trans. Systems, Man, and Cybernetics*, vol. 22, no. 5, pp. 865–883, Sept. 1992.

[16] L.D. Catledge and J.E. Pitkow, "Characterizing Browsing Strategies in the World-Wide Web," *Proc. Third WWW Conf.*, Apr. 1995.

[17] P.K. Chan and S.J. Stolfo, "Learning Arbiter and Combiner Trees from Partitioned Data for Scaling Machine Learning," *Proc. First Int'l Conf. Knowledge Discovery and Data Mining (KDD '95)*, pp. 39–44, Aug. 1995.

[18] P. Cheeseman and J. Stutz, "Bayesian Classification (AutoClass): Theory and Results," U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, eds., *Advances in Knowledge Discovery and Data Mining*, pp. 153–180, AAAI/MIT Press, 1996.

[19] M.-S. Chen, J.-S. Park, and P.S. Yu, "Data Mining for Path Traversal Patterns in a Web Environment," *Proc. 16th Int'l Conf. Distributed Computing Systems*, pp. 385–392, May 27-30, 1996.

[20] M.-S. Chen and P.S. Yu, "Using Multi-Attribute Predicates for Mining Classification Rules," *IBM Research Report*, 1995.

[21] D.W. Cheung, J. Han, V. Ng, and C.Y. Wong, "Maintenance of Discovered Association Rules in Large Databases: An Incremental Updating Technique," *Proc. Int'l Conf. Data Eng.*, New Orleans, La., Feb. 1996.

[22] C. Clifton and D. Marks, "Security and Privacy Implications of Data Mining," *Proc. 1996 SIGMOD '96 Workshop Research Issues Data Mining and Knowledge Discovery (DMKD '96)*, pp. 15–20, Montreal, Canada, June 1996.

[23] J. December and N. Randall, *The World Wide Web Unleashed*. SAMS Publishing, 1994.

[24] V. Dhar and A. Tuzhilin, "Abstract-Driven Pattern Discovery in Databases," *IEEE Trans. Knowledge and Data Eng.*, pp. 926–938, Dec. 1993.

[25] S. Dzeroski, "Inductive Logic Programming and Knowledge Discovery," U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, eds., *Advances in Knowledge Discovery and Data Mining*, pp. 117–152. AAAI/MIT Press, 1996.

[26] J. Elder IV and D. Pregibon, "A Statistical Perspective on Knowledge Discovery in Databases," U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, eds., *Advances in Knowledge Discovery and Data Mining*, pp. 83–115. AAAI/MIT Press, 1996.

[27] M. Ester, H.-P. Kriegel, and X. Xu, "Knowledge Discovery in Large Spatial Databases: Focusing Techniques for Efficient Class Identification," *Proc. Fourth Int'l Symp. Large Spatial Databases (SSD '95)*, pp. 67–82, Portland, Maine, Aug. 1995.

[28] C. Faloutsos and K.-I. Lin, "FastMap: A Fast Algorithm for Indexing, Data-Mining and Visualization of Traditional and Multimedia Datasets," *Proc. ACM SIGMOD*, pp. 163–174, May 1995.

[29] C. Faloutsos, M. Ranganathan, and Y. Manolopoulos, "Fast Subsequence Matching in Time-Series Databases," *Proc. ACM SIGMOD*, Minneapolis, Minn., pp. 419–429, May 1994.

[30] U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, *Advances in Knowledge Discovery and Data Mining*. AAAI/MIT Press, 1996.

[31] D. Fisher, "Improving Inference Through Conceptual Clustering," *Proc. AAAI Conf.*, pp. 461–465, Seattle, July 1987.

[32] D. Fisher, "Optimization and Simplification of Hierarchical Clusterings," *Proc. First Int'l Conf. Knowledge Discovery and Data Mining (KDD '95)*, pp. 118–123, Montreal, Canada, Aug. 1995.

[33] Y. Fu and J. Han, "Meta-Rule-Guided Mining of Association Rules in Relational Databases," *Proc. First Int'l Workshop Integration of Knowledge Discovery with Deductive and Object-Oriented Databases (KDOOD '95)*, pp. 39–46, Singapore, Dec. 1995.

[34] B.R. Gains, "Tranforming Rules and Trees into Comprehensive Knowledge Structures," U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, eds., *Advances in Knowledge Discovery and Data Mining*, pp. 205–228. AAAI/MIT Press, 1996.

[35] A. Gupta, V. Harinarayan, and D. Quass, "Aggregate-Query Processing in Data Warehousing Environment," *Proc. 21st Int'l Conf. Very Large Data Bases*, pp. 358–369, Zurich, Sept. 1995.

[36] J. Han, "Mining Knowledge at Multiple Concept Levels," *Proc. Fourth Int'l Conf. Information and Knowledge Management*, pp. 19–24, Baltimore, Md., Nov. 1995.

[37] J. Han, Y. Cai, and N. Cercone, "Data-Driven Discovery of Quantitative Rules in Relational Databases," *IEEE Trans. Knowledge and Data Eng.*, vol. 5, pp. 29-40, 1993.

[38] J. Han and Y. Fu, "Dynamic Generation and Refinement of Concept Hierarchies for Knowledge Discovery in Databases," *Proc. AAAI '94 Workshop Knowledge Discovery in Databases (KDD '94)*, pp. 157–168, Seattle, July 1994.

[39] J. Han and Y. Fu, "Discovery of Multiple-Level Association Rules from Large Databases," *Proc. 21th Int'l Conf. Very Large Data Bases*, pp. 420–431, Sept. 1995.

[40] J. Han and Y. Fu, "Exploration of the Power of Attribute-Oriented Induction in Data Mining," U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, eds., *Advances in Knowledge Discovery and Data Mining*, pp. 399–421, AAAI/MIT Press, 1996.

[41] J. Han, S. Nishio, and H. Kawano, "Knowledge Discovery in Object-Oriented and Active Databases," F. Fuchi and T. Yokoi, eds., *Knowledge Building and Knowledge Sharing*, pp. 221–230, Ohmsha, Ltd. and IOS Press, 1994.

[42] J. Han, Y. Fu, W. Wang, J. Chiang, W. Gong, K. Koperski, D. Li, Y. Lu, A. Rajan, N. Stefanovic, B. Xia, and O.R. Zaiane, "DBMiner: A System for Mining Knowledge in Large Relational Databases," *Proc. Int'l Conf. Data Mining and Knowledge Discovery (KDD '96)*, pp. 250-255, Portland, Ore., Aug. 1996.

[43] V. Harinarayan, J.D. Ullman, and A. Rajaraman, "Implementing Data Cubes Efficiently," *Proc. 1996 ACM SIGMOD Int'l Conf. Management Data*, pp. 205-216, Montreal, Canada, June 1996.

[44] IBM, "Scalable POWERparallel Systems," Technical Report GA23-2475-02, Feb. 1995.

[45] T. Imielinski and A. Virmani, "DataMine—Application Programming Interface and Query Language for kdd Applications," *Proc. Int'l Conf. Data Mining and Knowledge Discovery (KDD '96)*, pp. 256-261, Portland, Ore., Aug. 1996.

[46] H.V. Jagadish, "A Retrieval Technique for Similar Shapes," *Proc. ACM SIGMOD*, pp. 208–217, 1991.

[47] A.K. Jain and R.C. Dubes, *Algorithms for Clustering Data*. Prentice Hall, 1988.

[48] L. Kaufman and P.J. Rousseeuw, *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley and Sons, 1990.

[49] D. Keim, H. Kriegel, and T. Seidl, "Supporting Data Mining of Large Databases by Visual Feedback Queries," *Proc. 10th Int'l Conf. Data Eng.*, pp. 302–313, Houston, Feb. 1994.

[50] W. Kim, *Introduction to Objected-Oriented Databases*. Cambridge, Mass.: MIT Press, 1990.

[51] M. Klemettinen, H. Mannila, P. Ronkainen, H. Toivonen, and A.I. Verkamo, "Finding Interesting Rules from Large Sets of Discovered Association Rules," *Proc. Third Int'l Conf. Information and Knowledge Management*, pp. 401–408, Gaithersburg, Md., Nov. 1994.

[52] W. Klösgen, "Explora: A Multipattern and Multistrategy Discovery Assistant," U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, eds., *Advances in Knowledge Discovery and Data Mining*, pp. 249–271. AAAI/MIT Press, 1996.

[53] K. Koperski and J. Han, "Discovery of Spatial Association Rules in Geographic Information Databases," *Proc. Fourth Int'l Symp. Large Spatial Databases (SSD '95)*, pp. 47–66, Portland, Maine, Aug. 1995.

[54] C-S. Li, P.S. Yu, and V. Castelli, "HierarchyScan: A Hierarchical Similarity Search Algorithm for Databases of Long Sequences," *Proc. 12th Int'l Conf. Data Eng.*, Feb. 1996.

[55] H. Lu, R. Setiono, and H. Liu, "NeuroRule: A Connectionist Approach to Data Mining," *Proc. 21th Int'l Conf. Very Large Data Bases*, pp. 478–489, Sept. 1995.

[56] W. Lu, J. Han, and B.C. Ooi, "Knowledge Discovery in Large Spatial Databases," *Proc. Far East Workshop Geographic Information Systems*, pp. 275–289, Singapore, June 1993.

[57] H. Mannila, H. Toivonen, and A. Inkeri Verkamo, "Efficient Algorithms for Discovering Association Rules," *Proc. AAAI Workshop Knowledge Discovery in Databases*, pp. 181–192, July 1994.

[58] C.J. Matheus, G. Piatetsky-Shapiro, and D. McNeil, "Selecting and Reporting What is Interesting: The KEFIR Application to Health-Care Data," U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, eds., *Advances in Knowledge Discovery and Data Mining*, pp. 495–516. AAAI/MIT Press, 1996.

[59] M. Mehta, R. Agrawal, and J. Rissanen, "SLIQ: A Fast Scalable Classifier for Data Mining," *Proc. Int'l Conf. Extending Database Technology (EDBT '96)*, Avignon, France, Mar. 1996.

[60] R.S. Michalski, "A Theory and Methodology of Inductive Learning," Michalski et al., eds., *Machine Learning: An Artificial Intelligence Approach*, vol. 1, pp. 83–134. Morgan Kaufmann, 1983.

[61] R.S. Michalski, L. Kerschberg, K.A. Kaufman, and J.S. Ribeiro, "Mining for Knowledge in Databases: The INLEN Architecture, Initial Implementation, and First Results," *J. Int'l Information Systems*, vol. 1, pp. 85–114, 1992.

[62] R. Ng and J. Han, "Efficient and Effective Clustering Method for Spatial Data Mining," *Proc. Int'l Conf. Very Large Data Bases*, pp. 144–155, Santiago, Chile, Sept. 1994.

[63] D.E. O'Leary, "Knowledge Discovery as a Threat to Database Security," G. Piatetsky-Shapiro and W.J. Frawley, eds., *Knowledge Discovery in Databases*, pp. 507–516, AAAI/MIT Press, 1991.

[64] A. Papoulis, *Probability, Random Variable, and Stochastic Process*. New York: McGraw Hill, 1984.

[65] J.-S. Park, M.-S. Chen, and P.S. Yu, "Mining Association Rules with Adjustable Accuracy," *IBM Research Report*, 1995.

[66] J.-S. Park, M.-S. Chen, and P.S. Yu, "An Effective Hash Based Algorithm for Mining Association Rules," *Proc. ACM SIGMOD*, pp. 175–186, May 1995.

[67] J.-S. Park, M.-S. Chen, and P.S. Yu, "Efficient Parallel Data Mining for Association Rules," *Proc. Fourth Int'l Conf. Information and Knowledge Management*, pp. 31–36, Nov. 29-Dec. 3, 1995.

[68] G. Piatetsky-Shapiro, "Discovery, Analysis, and Presentation of Strong Rules," G. Piatetsky-Shapiro and W. J. Frawley, eds., *Knowledge Discovery in Databases*, pp. 229–238. AAAI/MIT Press, 1991.

[69] G. Piatetsky-Shapiro, U. Fayyad, and P. Smith, "From Data Mining to Knowledge Discovery: An Overview," U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, eds., *Advances in Knowledge Discovery and Data Mining*, pp. 1–35. AAAI/MIT Press, 1996.

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

[71] J.R. Quinlan, "Induction of Decision Trees," *Machine Learning*, vol. 1, pp. 81–106, 1986.

[72] J.R. Quinlan, *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.

[73] A. Savasere, E. Omiecinski, and S. Navathe, "An Efficient Algorithm for Mining Association Rules in Large Databases," *Proc. 21th Int'l Conf. Very Large Data Bases*, pp. 432–444, Sept. 1995.

[74] P.G. Selfridge, D. Srivastava, and L.O. Wilson, "IDEA: Interactive Data Exploration and Analysis," *Proc. ACM SIGMOD Int'l Conf. Management Data*, pp. 24-34, Montreal, Canada, June 1996.

[75] W. Shen, K. Ong, B. Mitbander, and C. Zaniolo, "Metaqueries for Data Mining," U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, eds., *Advances in Knowledge Discovery and Data Mining*, pp. 375–398. AAAI/MIT Press, 1996.

[76] A. Silberschatz, M. Stonebraker, and J.D. Ullman, "Database Research: Achievements and Opportunities into the 21st Century," *Report NSF Workshop Future of Database Systems Research*, May 1995.

[77] A. Silberschatz and A. Tuzhilin, "On Subjective Measure of Interestingness in Knowledge Discovery," *Proc. First Int'l Conf. Knowledge Discovery and Data Mining (KDD '95)*, pp. 275–281, Montreal, Canada, Aug. 1995.

[78] R. Srikant and R. Agrawal, "Mining Generalized Association Rules," *Proc. 21th Int'l Conf. Very Large Data Bases*, pp. 407–419, Sept. 1995.

[79] R. Srikant and R. Agrawal, "Mining Quantitative Association Rules in Large Relational Tables," *Proc. 1996 ACM SIGMOD Int'l Conf. Management Data*, pp. 1-12, Montreal, Canada, June 1996.

[80] R. Stam and R. Snodgrass, "A Bibliography on Temporal Databases," *IEEE Bull. Data Eng.*, vol. 11, no. 4, Dec. 1988.

[81] Y. Stettiner, D. Malah, and D. Chazan, "Dynamic Time Warping with Path Control and Nonlocal Cost," *Proc. 12th IAPR Int'l Conf. Pattern Recognition*, pp. 174–177, Oct. 1994.

[82] S.M. Weiss and C.A. Kulikowski, *Computer Systems that Learn: Classification and Prediction Methods from Statistics, Neural Nets, Machine Learning, and Expert Systems*. Morgan Kaufman, 1991.

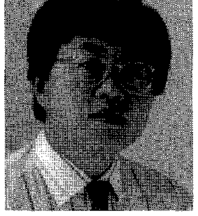
[83] J. Widom, "Research Problems in Data Warehousing," *Proc. Fourth Int'l Conf. Information and Knowledge Management*, pp. 25–30, Baltimore, Nov. 1995.

[84] W.P. Yan and P. Larson, "Eager Aggregation and Lazy Aggregation," *Proc. 21st Int'l Conf. Very Large Data Bases*, pp. 345–357, Zurich, Sept. 1995.

[85] T. Zhang, R. Ramakrishnan, and M. Livny, "BIRCH: An Efficient Data Clustering Method for Very Large Databases," *Proc. 1996 ACM SIGMOD Int'l Conf. Management Data*, Montreal, Canada, June 1996.

[86] Y. Zhuge, H. Garcia-Molina, J. Hammer, and J. Widom, "View Maintenance in a Warehousing Environment," *Proc. 1995 ACM SIGMOD Int'l Conf. Management Data*, pp. 316–327, San Jose, Calif., May 1995.

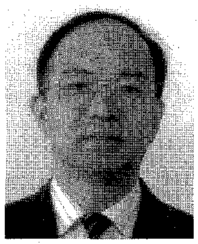
[87] W. Ziarko, *Rough Sets, Fuzzy Sets and Knowledge Discovery*. Springer-Verlag, 1994.



Ming-Syan Chen received the BS degree in electrical engineering from National Taiwan University in Taipei, Republic of China, in 1982; and the MS and PhD degrees in computer, information, and control engineering from the University of Michigan, Ann Arbor, in 1985 and 1988, respectively.

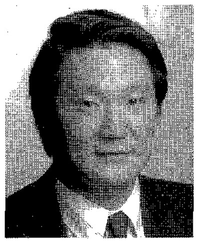
Dr. Chen is a faculty member in the Electrical Engineering Department at National Taiwan University. His research interests include network databases, multimedia technologies, and Internet applications. He was a research staff member at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York, form 1988 to 1996, primarily involved in projects related to parallel databases, multimedia systems, and digital libraries. He has published more than 70 refereed international journal/conference papers in his research areas.

Dr. Chen is a guest co-editor of this special issue of *IEEE Transactions on Knowledge and Data Engineering* dedicated to data mining. He is the inventor of many international patents in the areas of interactive video playout, video server design, interconnection networks, and concurrency and coherency control protocols. He received the Outstanding Innovation Award from IBM in 1994 for his contributions to parallel transaction design of a major database product, and has received numerous awards for his inventions and patent applications. He is a senior member of the IEEE and a member of the IEEE Computer Society and ACM.



Jiawei Han received his PhD from the University of Wisconsin, Madison, in 1985. He is now a professor in the School of Computer Science and director of the Database Systems Research Laboratory at Simon Fraser University in British Columbia, Canada.

Dr. Han has conducted research in the areas of knowledge discovery in databases, deductive and object-oriented databases, spatial databases, multimedia databases, and logic programming. He has published more than 100 journal and conference papers. He is a project leader of the Canada NCE/IRIS:HMI-5 project (data mining and knowledge discovery in large databases) and has served or is currently serving on program committees of more than 30 international conferences and workshops—including ICDE '95 (Program Committee vice chair), DOOD '95, ACM SIGMOD '96, VLDB '96, CIKM '96, KDD '96 (Program Committee co-chair), and SSD '97. He has also served as editor of *IEEE Transactions on Knowledge and Data Engineering*, the *Journal of Intelligent Information Systems*, and the *Journal of Data Mining and Knowledge Discovery*. He is a senior member of the IEEE and a member of the IEEE Computer Society.



Philip S. Yu (S'76-M'78-SM'87-F'93) received the BS degree in electrical engineering from National Taiwan University, Taipei, Taiwan, Republic of China, in 1972; the MS and PhD degrees in electrical engineering from Stanford University in 1976 and 1978, respectively; and the MBA degree from New York University in 1982.

Dr. Yu has been with the IBM Thomas J. Watson Research Center, Yorktown Heights, New York, since 1978, and is currently manager of the Software Tools and Techniques group in the Internet Department. His current research interests include database systems, data mining, multimedia systems, transaction and query processing, parallel and distributed systems, disk arrays, computer architecture, performance modeling, and workload analysis. He has published more than 200 papers in refereed journals and conferences, plus more than 140 research reports and 90 invention disclosures. He holds, or has applied for, 31 U.S. patents.

Dr. Yu is a fellow of the IEEE and the ACM, and a member of the IEEE Computer Society. He is an editor of *IEEE Transactions on Knowledge and Data Engineering*. In addition to serving as a program committee member of various conferences, he has served as program chair of the Second International Workshop on Research Issues on Data Engineering; Transaction and Query Processing and as program co-chair of the 11th International Conference on Data Engineering. He has received several IBM and external honors, including awards for Best Paper, IBM Outstanding Innovation, Outstanding Technical Achievement, Research Division, and 15 Invention Achievements.