An Overview of Data Mining and Knowledge Discovery

Fan Jianhua (范建华) and Li Deyi (李德毅)*

Department of Computer Science, Communication Engineering Institute Nanjing 210016, P.R. China

**System Engineering Institute, Beijing 100039, P.R. China*

Received September 15, 1997; revised March 27, 1998.

Abstract

With massive amounts of data stored in databases, mining information and knowledge in databases has become an important issue in recent research. Researchers in many different fields have shown great interest in data mining and knowledge discovery in databases. Several emerging applications in information providing services, such as data warehousing and on-line services over the Internet, also call for various data mining and knowledge discovery techniques to understand user behavior better, to improve the service provided, and to increase the business opportunities. In response to such a demand, this article is to provide a comprehensive survey on the data mining and knowledge discovery techniques developed recently, and introduce some real application systems as well. In conclusion, this article also lists some problems and challenges for further research.

Keywords: Knowledge discovery in databases, data mining, machine learning, association rule, classification, data clustering, data generalization, pattern searching.

1 Introduction

Across a wide variety of fields, data are being collected and accumulated at a dramatic pace. The corporate, governmental, and scientific communities are being overwhelmed with an influx of data that is routinely stored in on-line databases. Analyzing these data and extracting meaningful patterns in a timely fashion are intractable without computer assistance and powerful analytical tools. Standard computer-based statistical and analytical packages alone, however, are of limited benefit without the guidance of trained statisticians to apply them correctly and domain experts to filter and interpret the results. There is an urgent need for a new generation of computational techniques and tools to assist humans in extracting useful information (knowledge) from the rapidly growing volumes of data. These techniques and tools are the subject of the emerging field of knowledge discovery in databases (KDD)[1,2].

1.1 Research History and Development of KDD

Historically the notion of finding useful patterns in data has been given a variety of names including data mining, knowledge extraction, information discovery, information harvesting, data archaeology, and data pattern processing. The term data mining has been mostly used by statisticians, data analysts and the management information systems (MIS) communities. It has also gained popularity in the database field. The term KDD was coined at the first KDD workshop in 1989 to emphasize that "knowledge" is the end product of a data-driven

discovery. It has been popularized in artificial intelligence and machine learning. The following table shows the history of KDD research development.

Table 1 Academic Meetings on KDD

Time	Name of meeting	Place of meeting	Accepted papers	Submitted papers
Jun. 1989	Workshop on KDD	Detroit, Michigan, USA	29	69
Jul. 1991	Workshop on KDD	Anaheim, California, USA	25	46
Jul. 1993	Workshop on KDD	Washington, USA	28	40
1995	KDD95	Montreal, Canada	40	135
Aug. 1996	KDD96	Portland, Oregon, USA	45	220
Feb. 1997	PAKDD97	Singapore	35	97
Aug. 1997	KDD97	California, USA		
Apr. 1998	PAKDD98	Melbourne, Australia		
Aug. 1998	KDD98	New York, USA		

The latest KDD97, which was held on August 14–17 in Newport Beach, CA, attracted over 600 people[3], and the topics included data mining, data visualization, text mining, OLAP, and statistical approaches. Moreover, KDD CUP competition was held at the same time, and a large number of companies exhibited their data mining systems and helped to generate a good interaction between researchers have and developers[3].

In all, with the exponentially increasing of the data and databases, KDD is becoming more and more important in many business and scientific domains. More and more researchers have turned their interests to this field.

1.2 The Relationship Between Data Mining and KDD

Generally, KDD is defined (by Fayyad, Piatetsky-Shapiro & Smyth[1]) as *the nontrivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data.*

Here data are a set of facts, (e.g., cases in a database) and pattern is an expression in some language (natural language or formal language) describing a subset of the data or a model applicable to that subset. Hence, in our usage here, extracting a pattern also means fitting a model to data, finding structure from data, or in general any high-level description of a set of data. The term process implies that KDD is composed of many steps, which involve data preparation, search for patterns, knowledge evaluation, and refinement, all repeated in multiple iteration. By non-trivial we mean that some search of inference is involved, i.e., it is not a straightforward computation of predefined quantities like computing the average value of a set of numbers. The discovered patterns should be valid for new data with some degree of certainty. We also want patterns to be novel and potentially useful, e.g., leading to some benefit to the user or task. Finally, the patterns should be understandable, if not immediately then after some post-processing.

Also according to [1], Data Mining is *a step in the KDD process consisting of applying data analysis and discovery algorithms that, under acceptable computational efficiency limitations, produce a particular enumeration of patterns over the data.*

So, the data mining is just a step within the whole KDD process, and is the application of specific algorithms for extracting patterns from data. It is concerned with the algorithmic means by which patterns are extracted and enumerated from data. The additional steps in the KDD process (Fig.1), such as data preparation, data selection, data cleaning, incorporation of appropriate prior knowledge, and proper interpretation of the results of mining, are essential to ensure that useful knowledge is derived from the data. Blind application of data mining methods can be a dangerous activity easily leading to the discovery of meaningless patterns.

In the following sections, we will provide a comprehensive survey on the data mining techniques, and briefly introduce some well-known application systems. Finally, we point out some research challenges and highlight issues for further study.

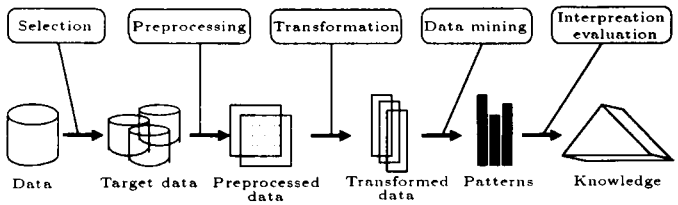

Fig.1. An overview of the steps that comprise the KDD process[4].

2 Data Mining Techniques

The knowledge discovery goals are defined by the intended use of the system. We can distinguish two types of goals: Verification, where the system is limited to verifying the user's hypothesis, and Discovery, where the system autonomously finds new patterns. We further subdivide the Discovery goal into Prediction, where the system finds patterns for the purpose of predicting the future behavior of some entities, and Description, where the system finds patterns for the purpose of presenting them to a user in a human-understandable form. Whatever they are, the goals can be achieved via the following methods: database search, generalization, knowledge representation, mathematical or statistical modeling, etc.

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 operation. In this sense, data mining itself has formed an independent new field.

In general, data mining is an application-dependent issue and different applications explored may require different mining techniques to cope with. The techniques, which are usually applied during data mining process, are categorized as follows.

2.1 Mining Association Rules

Mining association rules is to derive a set of strong association rules in the form of "$A_1 \wedge A_2 \wedge \cdots \wedge A_m \Rightarrow B_1 \wedge B_2 \wedge \cdots \wedge B_n$", 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 that an association rule, such 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 at mining such rules.

Here, algorithms Apriori and DHP, developed in [5,6] respectively, are described to illustrate the nature of this problem.

Algorithm Apriori

Consider an example transaction database given in Fig.2. In Apriori, in each iteration (or each pass) it constructs a candidate set of large itemsets, counts the number of occurrences of each candidate itemset, and then determines large itemsets based on a pre-determined minimum support. 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.3. Assuming that the minimum transaction support required is 2 (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. 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.3 represents the result form 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.

Database D

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

Fig.2. 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 {B, C} and {B, E}, are identified first. Then, Apriori tests whether the 2-itemset {C, E}, which consists of their second items, constitutes a large 2-itemset or not. Since {C, E} is a large itemset by itself, we know that all the subsets of {B, C, E} are large and then {B, C, E} becomes a candidate 3-itemset. There are no other candidate 3-itemsets from L_2. Apriori then scans all the transactions and discovers the large 3-itemsets L_3 in Fig.3. Since there is no candidate 4-itemset to be constituted from L_3, Apriori ends the process of discovering large itemsets.

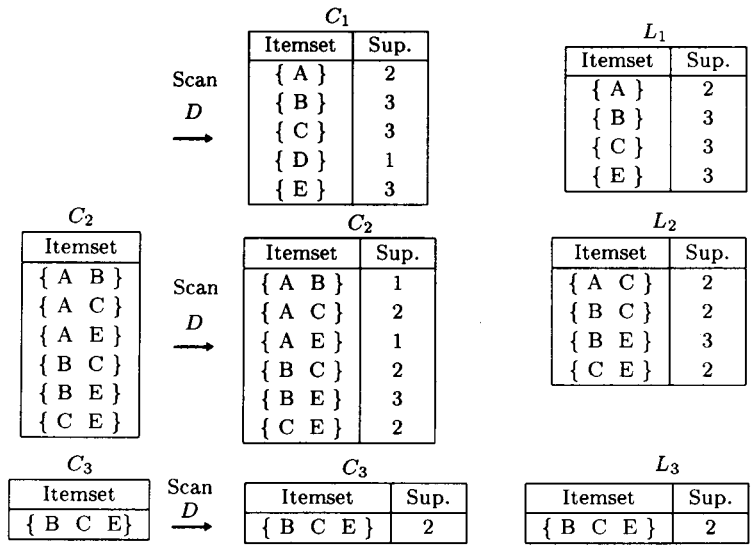

Fig.3. Generation of candidate itemsets and large itemsets.

Algorithm DHP

Similar to Apriori, DHP in [6] also generates candidate k-itemsets from L_{k-1}. However, DHP is unique in that it 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, thus leading to better performance. 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.

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[7] or at multiple level[8].

In [8], the notion of mining multiple-level association rules is 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 association rules based on different ways of sharing of multiple-level mining processes and reduction of the encoded transaction tables. In [7], 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[9] and meta-rule guided mining of association rules in relational databases[10,11] are also studied recently, with efficient algorithms developed.

Notice that not all the discovered strong association rules (i.e., passing the minimum support and minimum confidence thresholds) are interesting enough to present. There have been some interesting studies on the interestingness or usefulness of discovered rules, such as [7, 12, 13]. The notion of interestingness on discovered generalized association rules is introduced in [7]. The subjective measure of interestingness in knowledge discovery is studied in [13].

2.2 Multi-Level Data Generalization

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 three approaches: (1) data cube approach[15–17], (2) attribute-oriented induction approach[17–20], (3) approach based on knowledge discovery state space theory[21,22].

2.2.1 Data Cube Approach

The data cube approach has a few alternative names or a few variants, such as, "data warehousing", "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, maximum, etc., and 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.

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[14] 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.

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.

2.2.2 Attribute-Oriented Induction Approach

The data warehousing approach which uses materialized views is "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[17–20]. 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, including attribute-removal, concept-tree climbing, attribute-threshold, propagation of counts and other aggregate function values, etc. The generalized data are 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.

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 by 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 complexity of the process is $O(n)$, if a cube structure is used (desirable when the data cube is dense), or $O(np)$, 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.

The essential background knowledge applied in attribute-oriented induction is concept hierarchy (or lattice) associated with each attribute. Most concept hierarchies are stored implicitly in databases. Conceptual hierarchies for numerical or ordered attributes can be generated automatically based on the analysis of data distributions in the set of relevant data[23]. 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[23].

The approach has been implemented in a data mining system, DBMiner, and experimented in several large relational databases[23]. The approach can also be extended to generalization-based data mining in object-oriented databases, spatial databases, and other kinds of database. This 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.

2.2.3 Knowledge Discovery State Space

Now most of the database systems are based on relational data model and the Discovery State Space Theory, which was put forward by Dr. Deyi Li, mostly focuses on the RDBMS.

Data in a relational database are organized through certain relation tables, and they can be linked together in logic. So all the data in the database are composed of a single 2-dimensional Universal Relation, shown in Fig.4. The horizontal of the Universal Table is the attributes, and the vertical is the records or tuples. Through specific interface, the data in the database can be seen as a single flat file by external application, so the existing machine learning and discovery algorithms can be applied directly.

According to the discovery tasks that are put forward by users, with SQL mapping and projecting functions, the related data can be extracted. If the amount of extracted data is too large to handle, sampling can be done firstly. Finally, the initial data set D that is related to discovery tasks is formed.

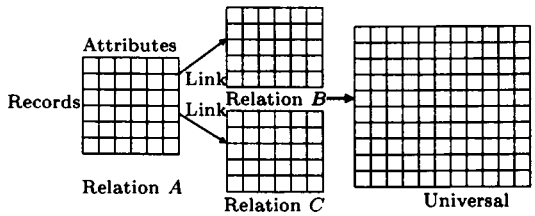

Fig.4. Relations in database and Universal Table.

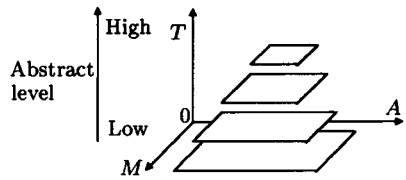

Fig.5. The knowledge discovery state space.

In this initial data set D, some records are induced to one Macro Tuple, and all the Macro Tuples are composed of the initial Knowledge Template. Then, three directional operations are carried out on this Knowledge Template: Attribute Oriented Operation, Macro Tuple Oriented Operation, and Template Oriented Operation.

Attribute Oriented Operation (OA) is to find and discover the relationship between attributes, while the Macro Tuple Oriented Operation (OM) is to find and discover the accordance and differences between Macro Tuples. For Template Oriented Operation (OT), its main task is to abstract the template to a higher level. Thus a 3-dimensional discovery state space is set up, and many the discovery algorithms are carried out within this space, shown as Fig.5.

Suppose the Universal Table has m attributes and n tuples, and each attribute has V_i different values ($1 \leq i \leq n$). So the value space for the Universal Table is set:

$$U = V_1 \times V_2 \times \cdots \times V_i$$

Each tuple has a correspondent real point in the U, and the n points form a set R. The distributed data patterns in the U reflect the principles and relationships among data.

2.3 Data Classification

Data classification is the process, which finds the common properties of a set of objects in a database and classifies them into different classes. To construct a classification model, a small 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 present 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.

2.3.1 Classification Based on Decision Trees

In studies of machine learning, the decision-tree classification method, developed by Quinlan[24,25], has been influential. It is a supervised learning method that constructs decision trees from a set of examples. The method first chooses a subset of the training examples 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[24], 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 that minimizes the information needed in the resulting subtrees to classify the elements. An extension to ID-3, C4.5[25], extends the domain of classification from categorical attributes to numerical ones.

The ID-3 system uses information gain as the evaluation functions for classification, e.g., the following evaluation function,

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

where p_i is the probability that an object is in class i.

2.3.2 Classification Using Rough Sets

Databases are often highly incomplete, that is, in the selected attribute-value representation many possible tuples are not present in the database; and in particular, insufficient tuples exist in the database to reliably estimate the probability distributions. Rough sets (or fuzzy sets) take the advantage to deal with such kind of problems.

Shan *et al.*[26] proposed an approach, which uses rough sets to ensure the completeness of the classification and the reliability of the probability estimate prior to rule induction.

A relational database can be viewed as an information system. Formally, an information system S is a quadruple $\langle U, A, V, F\rangle$, where U is a nonempty set of objects called universe; A is a finite set of attributes consisting of condition attributes C and decision attributes D such that $A = C \cup D$ and $C \cap D = \emptyset$, $V = \cup_{pA} V_p$ is a nonempty finite set of values

of attributes A and V_p is the domain of the attribute p (the set of values of attribute p); $F : U \times A \rightarrow V$ is an information function which assigns particular values from the domain of attributes A to objects such that $f(x_i, p) \in V_p$ for all $x_i \in U$ and $p \in A$.

Shan *et al.* introduced the notion of classification complexity, defined as the number of equivalence classes in the classification. In practice, this number is usually not known in advance. Instead, a crude upper bound on the classification complexity for a subset of attributes $B \subseteq C$, can be computed by:

$$TC(B, V) = \prod_{p \in B} \text{card}\,(V_p)$$

The quantity $TC(B, V)$ is called the theoretical complexity of the set of attributes B given the set of values V of the attributes B.

Briefly, Shan's method performs three steps prior to rule induction. First, it generalizes the condition attributes as necessary to increase the credibility of the classification. It applies attribute-oriented concept tree ascension[18,19] to reduce the complexity of an information system, and generalizes a condition attribute to a certain level based on the attribute's concept tree, which is provided by knowledge engineers or domain experts. The number of possible values at a higher level of an attribute is always smaller than at a lower level, so the theoretical complexity is reduced.

Then, the method identifies clusters of interacting attributes, i.e., searches for credible classifications of the database tuples based on these clusters. A classification is credible if it is complete or almost complete with respect to the domain from which the database was collected.

Finally, it searches for acceptable classifications. Classifications, which result in the good approximation of the concept of the interest, in the rough sets sense, are subsequently selected to obtain the classification rules.

There have been many other approaches to data classification, such as statistical approaches[12]. There have also been some studies on classification techniques in the context of databases[27]. An interval classifier has been proposed in [27] to reduce the cost of decision tree generation. An attribute-oriented induction method has been developed for mining classification rules in relational databases[17]. The work in [27] explores rule extraction in a database based on neural networks.

2.4 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.

2.4.1 Clustering Based on Randomized Search

Ng and Han presented a clustering algorithm, CLARANS (Clustering Large Applications based upon RANdomized Search)[28] based on randomized search and originated from two clustering algorithms used in statistics, PAM (Partitioning Around Medoids) and CLARA (Clustering LARge Applications)[29].

The PAM algorithm finds k clusters in n objects by first finding a representative object for each cluster. Such a representative, which is the most centrally located point in a cluster, is called a medoid. After selecting k medoids, the algorithm repeatedly tries to make a

better choice of medoids by analyzing all possible pairs of objects such that one object is a medoid and the other is not. The measure of clustering quality is calculated for each such combination. The best choice of points in one iteration is chosen as the medoids for the next iteration. The cost of a single iteration is $O(k(n-k)^2)$. It is therefore computationally quite inefficient for large values of n and k.

The CLARA algorithm accomplishes the same task but is based upon sampling. Only a small portion of the real data is chosen as the representative of the data and medoids are chosen from this sample using PAM. The idea is that if the sample is selected in a fairly random manner, then it correctly represents the whole data set and therefore, the representative objects (medoids) chosen will be similar as if they are chosen from the whole data set. CLARA draws multiple samples and outputs the best clustering out of these samples. As expected, CLARA can deal with larger data sets than PAM. The complexity of each iteration now becomes $O(kS^2 + k(n-k))$, where S is the size of the sample. The authors indicated through their experimental results that samples of size $40+2k$ gave good results.

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[17]: spatial dominant approach, SD (CLARANS) and non-spatial 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. Experiments show that the method can be used to cluster of reasonably large data sets, such as houses in the Vancouver area, and the CLARAN algorithm outperfoms PAM and CLARA.

Ester *et al.* pointed out some of the drawbacks of the CLARANS clustering algorithm[30], and proposed two approaches to reduce the cost of the algorithm. One is called *focusing on representative objects*, and the other is called *focusing on relevant clusters* and *focusing on a cluster*[30]. They reported that when the focusing technique was used the effectiveness decreased just from 1.5% to 3.2% whereas the efficiency (CPU time) increased by a factor of 50.

2.4.2 Clustering Based on CF Trees

Zhang *et al.*[31] presented another algorithm, BIRCH (Balanced Iterative Reducing and Clustering), for clustering large sets of points. The method they presented is the incremental one with the possibility of adjustment of memory requirements to the size of memory that is available. They introduced two concepts, Clustering Feature and CF tree.

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

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

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, i.e., $\sum_{i=1}^{N} \vec{X}_i^2$. 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 non-leaf nodes are storing sums of their children's CFs, and thus, they summarize the information about their children. The CF tree is built 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, 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 the threshold is specified and the CF tree is rebuilt. The rebuilding 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 have 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 CLARNS, may be used with CF trees. The CPU and I/O costs of the BIRCH algorithm are $O(n)$.

2.5 Pattern Searching

Searching similar patterns in databases is an important aspect of data mining, and usually is applied in a temporal or spatial-temporal database. Significant progress has recently been made, and various approaches have been studied also[32−35]. Here we discuss the similarity measures and the genetic algorithm.

2.5.1 Similarity Measures

Different similarity measures have been considered, mainly the Euclidean distance[32,33] and the correlation[35]. 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 the length of $\{y_i\}$, and assume $n \leq N$. There is a total of $N(N-1)/2$ subsequences of $\{y_i\}$ with length ranging from 2 to N. The j-th subsequence of $\{y_i\}$ is denoted as $z_i^{j,k,l}$, where k is the offset and l is the length. A metric for measuring the similarity between $\{x_i\}$ and $\{y_i\}$ can be defined as

$$\min_j \sum_{i=1}^{n} (x_i - K_j z_i^{j,k,l})^2 \tag{2.1}$$

where K_j is 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 [35]. 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

$$c_i = \frac{\sum_{j=1}^{n} x_i y_{i+j}}{\sqrt{\sum_{j=1}^{n} X_j^2 \sqrt{\sum_{j=1}^{N} Y_j^2}}} \tag{2.2}$$

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 having 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

$$c_i = \frac{F^{-1}\{X_j^* Y_j\}}{\sqrt{\sum_{j=1}^{n} X_j^2}\sqrt{\sum_{j=1}^{N} Y_j^2}} \tag{2.3}$$

where X_j^* denotes the complex conjugate of X_j. The denominators of Eqs.(2.2) and (2.3) are equal in virture of Parseval's Theorem[32]. 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 less than one and the peaks of the sequence $\{c_i\}$ give the locations of possible matches.

Two types of similarity queries for temporal data have emerged thus far: whole matching[32] in which the target sequence and the sequences in the databases have the same length; subsequence matching[33] in which the target sequence could be shorter than the sequences in the database and the match can occur at any arbitrary point.

2.5.2 Genetic Algorithm

Genetic algorithms have been used successfully in a variety of search and optimization problems. In data mining field, they also play important roles, and many data mining researchers adopt them to extract patterns from large amount of raw data.

The famous one is the algorithm proposed by Ian W. Flockhart *et al.* in GA-MINER[36]. It uses a structured population model in which each gnome's reproductive partner is selected from within its local neighborhood (using tournament selection). This helps prolong diversity within the population and encourages local niching, which tends to result in exploration of several areas of the search space and matches well with the goal of finding several patterns in a single run.

The crossover operator is defined at a variety of levels, reflecting the structure of the representation. Disjunct clauses, clauses, terms and attribute values each comprise a single gene at the appropriate level. Within subset descriptions, crossover at the disjunct clause level is based on uniform crossover and enforces positional alignment of component clauses. Both uniform and single-point crossovers are used at the clause level, while crossover at the term level is again based on uniform crossover. Mutation is also defined at a variety of levels, with separate probabilities specified for mutating each of the component parts. Clauses, terms and values are added or deleted with specified probabilities and can be regarded as distinct specialization and generalization operators.

GA-MINER collects sets of patterns during the run of the algorithm for presentation to the user. A simple heuristic is used for updating the set, based on a strategy of continually replacing either the lowest fitness rule or the most similar rule (if the similarity is over a given threshold) by a higher fitness rule.

3 KDD Systems

As we discussed in Section 1, the data mining is just a step in the KDD process that consists of applying data analysis and discovery algorithms that, under acceptable computational efficiency limitations, produce a particular enumeration of patterns over data. KDD

focuses on the overall process of knowledge discovery from data, including how the data are stored and accessed, how algorithms can be scaled to massive data sets and still run efficiently, how results can be interpreted and visualized, and how the overall man-machine interaction can usefully be modeled and supported.

3.1 A Framework of KDD Systems

Until now, many KDD systems have been set up, and all of them have similar structure. Generally, a KDD system comprises a collection of components that can efficiently identify and extract interesting and useful new patterns from data stored in real-world databases. While the components may differ among systems, certain basic functions are usually identical. Matheus *et al.*[37] incorporated these as components in a model of an idealized KDD system as shown in Fig.6. The model components include:

- Discovery Controller: controls the invocation and parameterization of the other components
- Databases Interface: generates and processes database queries
- Knowledge Base: repository of domain specific information
- Focusing or Data Sampling: determines what portions of the data to analyze
- Pattern Extraction: a collection of pattern-extracting algorithms
- Evaluation: evaluates the degree of interests and utility of extracted patterns

Information flows into the system from three sources: the user issues high-level commands to the controller through human interface, and the DBMS provides the raw data; domain knowledge from various sources is deposited into the system's knowledge base. Raw data are selected from the databases through DBMS and then processed by the extraction algorithms which produce candidate patterns. These patterns are then evaluated and some are identified as interesting discoveries. In addition to presenting the results to the user, discovered patterns may be deposited into the system's knowledge base to support subsequent discoveries.

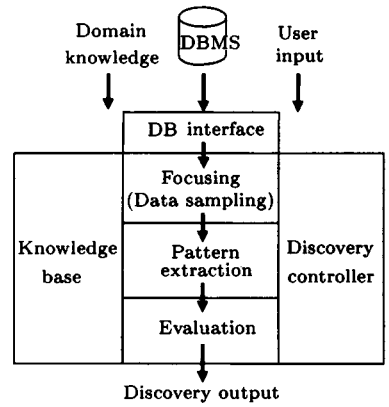

Fig.6. A framework of KDD system.

The framework above represents an abstraction of what occurs in KDD systems. In an actual system, it is not always possible to clearly separate the individual components of the model.

3.2 Real KDD Systems

Data mining and knowledge discovery are a fast expanding field with many new research results reported and new systems or prototypes developed recently, some of which have been successfully used for mining knowledge in very large real databases. Here we briefly introduce some systems reported in recent conferences and journals, and of course by no means complete.

3.2.1 CoverStory

CoverStory is a commercial system developed by Information Resources, Inc. In cooperation with John D. C. Little of MIT Sloan School of Management[38], its purpose is to

analyze supermarket scanner data and produce a marketing report summarizing the "key events."

CoverStory performs a top-down analysis of the raw scanner data beginning with aggregate products lines, decomposing them into product components, and finally comparing these with competitor products. At each stage the system ranks the products according to volume change, selects the top few to report, and identifies the causal factors having the highest scores as defined by the equation:

$$Score = Percent\text{-}Change * Factor\text{-}Weight * Market\text{-}Weight$$

The Percent-Change here is the percent of change in product volume, Factor-Weight is a constant weight indicating the relative importance of a marketing factor (derived from the original marketing analysis), and Market-Weight takes into account the market size (heuristically set to the square root of market size). After analyzing all market changes, CoverStory produces a report using natural-language templates to generate text.

The system gets its data directly from a scanner database. Its domain knowledge is built into the linear model of causal relationships and the top-down analysis algorithm. The system's controller follows the simple four-step algorithm outlined above. The causal model provides focus by identifying a small set of relevant features. The extraction algorithm falls into the class of deviation-detection methods: it identifies where the data deviate most from previous periods, other regions, or competitors performance, and then attempts to explain the deviations by identifying the factors that most strongly influence the results. Evaluation of the results relies on the ranking of changes according to the percent of change and the causal factor scores.

CoverStory is fully automated once the initial domain knowledge for a particular distributor has been entered. However, it is fairly limited in its applicability, being tied closely to the scanner data and marketing models.

3.2.2 EXPLORA

EXPLORA was developed by Hoschka and Klosgen[39]. It is "an integrated system for conceptually analyzing data and searching for interesting relationships." It operates by performing a graph search through a network of pattern templates searching for "facts". A fact is a data instant of a pattern template (also called statement types) that satisfies statistical criteria specified in an associated verification method. Redundancy rules use taxonomic information to reduce the search, and generalization and selection criteria condense the resulting set of discovered facts. Using the interactive browser, an end-user can take the ordered set of facts and generate a customized, final report. The user may also intervene throughout the discovery process to create new statement types, modify verification methods, and generally guide the search path.

After the system completes its search through the network of patterns, inductive generalization rules are applied to the set of discovered facts to integrate related statements. One form of generalization, for example, collects patterns with similar observations; another identifies regularities among similar statements across time periods. Application-dependent selection rules provide a final filter of the facts before the user turns them into a written report using the browser tool.

It is unclear from the literature whether EXPLORA has a direct DBMS interface. Its knowledge base is well defined, comprising generic statement types, verification methods, filtering rules, and applications-specific object taxonomies. EXPLORA's controller combines human guidance with a heuristic search through the space of instantiated statement types. Focus is provided by the instantiations of statement types which specify the fields to

access from specific subsets of records. EXPLORA's extraction algorithm is fundamentally a deviation detector that identifies significant differences between populations or across time periods. Evaluation is based primarily on statistical measures with additional, application-specific constraints optionally provided by the user.

EXPLORA is specifically designed to work with data that change "regularly and often". Within this context, the statement types are generic enough for most deviation-detection problems, but they need to be supplemented with object taxonomies specific to each application. Once these are defined, the search algorithm can be turned loose to operate autonomously, although human guidance is required to fine tune the results for the best performance.

3.2.3 Knowledge Discovery Workbench

The Knowledge Discovery Workbench (KDW) is a collection of tools for the interactive analysis of large databases[40]. Its components have evolved through three versions (KDW, KDW II, and KDW ++), all of which provide a graphical user interface to a suite of tools for accessing database tables, creating new fields, defining a focus, plotting data and results, applying discovery algorithms, and handling domain knowledge. The current version of the system is embedded with an extensible command interpreter based on tcl (an embeddable command language) which enables the user to interactively control the discovery process or call up intelligent scripts to automate discovery tasks. The following extraction algorithms have been incorporated into one or more versions of the KDW: clustering for identifying simple linearly-related classes; classification for finding rules using a decision-tree algorithm; summarization for characterizing classes or records: deviation detection for identifying significant differences between classes of records; dependency analysis for finding and displaying probabilistic dependencies. The details of most of these algorithms can be found in [41–43].

The KDW has direct access to a DBMS through its SQL-based query interface. Its knowledge base contains information specific to a database regarding important field groups, record groups, functional dependencies, and SQL-query statements. Most of the domain knowledge is used to provide focus by guiding the access of information from the database. Control in the KDW is provided exclusively by the user, who may define scripts to automate frequently repeated operations. The pattern-extraction algorithms range from clustering to classification to deviation detection. Each of these provides significance measures for their results, although final evaluation is left to the user.

3.2.4 DBMiner

DBMiner was developed by Jiawei Han and his colleagues (Simon Fraser University of Canada), and based on data mining techniques and the prototype, DBLearn. This system integrates data mining techniques with database technologies, and discovers various kinds of knowledge at multiple concept levels from large relational databases efficiently and effectively[44–46].

The system has the following distinct features:

1. It incorporates several interesting data mining techniques, including attribute-oriented induction, statistical analysis, progressive deepening for mining multiple-level rules, and meta-rule guided knowledge mining. It also implements a wide spectrum of data mining functions including generalization, characterization, association, classification, and prediction.
2. It performs interactive data mining at multiple concept levels on any user-specified set of data in a database using an SQL-like Data Mining Query Language, DMQL, or a graphical user interface. Users may interactively set and adjust various thresholds, control a data

mining process, perform roll-up or drill-down at multiple concept levels, and generate different forms of outputs, including generalized relations, generalized feature tables, multiple forms of generalized rules, visual presentation of rules, charts, curves, etc.

3. Efficient implementation techniques have been explored using different data structures, including generalized relations and multiple-dimensional data cubes. The implementations have been integrated smoothly with relational database systems.
4. The data mining process may utilize user- or expert-defined set-grouping or schema-level concept hierarchies which can be specified flexibly, adjusted dynamically based on data distribution, and generated automatically for numerical attributes. Concept hierarchies are being taken as an integrated component of the system and are stored as a relation in the database.
5. Both UNIX and PC (Windows/NT) versions of the system adopt a client/server architecture. The latter may communicate with various commercial database systems for data mining using the ODBC technology.

The general architecture of DBMiner, shown in Fig.7, tightly integrates a relational database system, such as a Sybase SQL server, with a concept hierarchy module, and a set of knowledge discovery modules. The discovery modules of DBMiner include characterizer, discriminator, classifier, association rule finder, meta-rule guided miner, predictor, evolution evaluator, deviation evaluator, and some planned future modules[44].

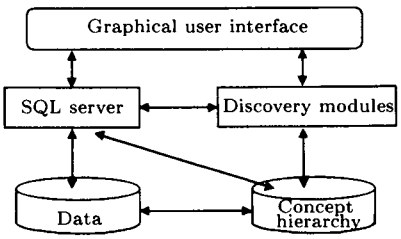

Fig.7. General architecture of DBMiner.

Now, the DBMiner system is being extended in several directions, including enhancing data mining efficiency and power, improving the ability of integration, maintenance and application of discovered knowledge, and extending data mining technique towards advanced and/or special purpose database systems. Currently, two such data mining systems, GeoMiner and WebMiner, for mining in spatial databases and the Internet information-base respectively, are under design and construction[44].

3.2.5 Quest

The Quest Data Mining System was developed by Rakesh Agrawal and his colleagues, IBM Almaden Research Center[47-49]. The goal is to develop technology to enable a new breed of data-intensive decision-support applications, so the algorithms in this system have the following features:

- discover patterns in very large databases, rather than simply verify that a pattern exists;
- have a completeness property that guarantees that all patterns of certain types have been discovered;
- have high performance and near-linear scaling on very large (multiple gigabytes) real-life databases.

The system architecture of the Quest system is shown as Fig.8.

The mining algorithms run on the server close to the data source. Users interact with the system through a GUI that can run on the same workstation or on a different client machine. There is an open API using which the user can optionally input results of any mining operation into software of choice. An interesting aspect of the Quest architecture is its I/O architecture. There is a standard stream interface defined for all accesses to input, insulating the algorithm code from data repository details, which are encapsulated in a data access API. Thus, it is easy to add new data repository types to the Quest system.

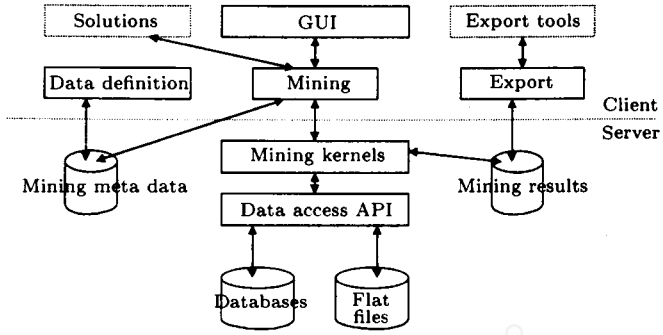

Fig.8. The Quest system architecture[47].

The Quest system runs on both AIX and MVS platforms, against data in flat files as well as DB2 family of database products. Databases can be accessed in a loosely-coupled mode using dynamic SQL.

4 Research and Application Challenges

The ever-increasing quantity of data in every computing environment presents both an opportunity to extract useful information and a challenge to process the massive volume of data effectively. As a young and promising field, data mining and knowledge discovery still face many challenges and unsolved problems which pose new research issues for further study[50−53]. Here we list some challenges and highlight issues which are important for further study:

Larger databases: Databases with hundreds of fields and tables, millions of records, and multigigabyte size are quite commonplace, and terabyte (10^{12} bytes) databases are beginning to appear. Methods for dealing with large data volumes include more efficient algorithms, sampling, approximation methods, and massively parallel processing[53,61].

High dimensionality: Not only are there often a very large number of records in the database, but there can also be a very large number of fields (attributes, variables) so that the dimensionality of the problem is high. A high dimensional data set creates problems in terms of increasing the size of the search space for model induction in a combinatorially explosive manner. In addition, it increases the chances that a data mining algorithm will find spurious patterns that are not valid in general. Approaches to this problem include methods to reduce the effective dimensionality of the problem and the use of prior knowledge to identify irrelevant variables[53].

Complex data: Some types of information, including text, images, audio, video, and anything related to the Web present an even grander challenge with potentially great rewards[53,58].

Changing data and knowledge: Rapidly changing (non-stationary) data may make previously discovered patterns invalid. In addition, the variables measured in a given application database may be modified, deleted, or augmented with new measurements over time. Possible solutions include incremental methods for updating the patterns and treating change as an opportunity for discovery by using it to cue the search for patterns of change only.

Missing and noisy data: This problem is especially acute in business databases. U.S. census data reportedly have error rates of up to 20%. Important attributes may be missing if the database was not designed with discovery in mind. Possible solutions include more sophisticated statistical strategies to identify hidden variables and dependencies[57,58].

Complex relationships between fields: Hierarchically structured attributes or values, relations between attributes, and more sophisticated means for representing knowledge about the contents of a database will require algorithms that can effectively utilize such information. Historically, data mining algorithms have been developed for simple attribute-value records, although new techniques for deriving relations between variables are being developed[61].

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[62,63].

Accuracy, uncertainty and expressiveness of data mining results: The discovered knowledge should accurately portray the contents of the database. The imperfectness should be expressed by measures of uncertainty, in the form of approximate rules or quantitative rules. That is, the data mining results should be accurate, and, each derived rule should be expressed by certain quantitative measurement to reflect the degree of the preciseness of the rule. Noise and exceptional data should be handled elegantly in data mining systems. This also motivates a systematic study of the measurement of the quality of the discovered knowledge, including interestingness and reliability, by construction of statistical, analytical, and simulation models and tools[57,58].

Interactive mining of knowledge at multiple concept 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.

Integration with other systems: A stand-alone discovery system may not be very useful. Typical integration issues include integration with a DBMS (e.g. via a query interface), integration with spreadsheets and visualization tools, and accommodating real-time sensor reading[65].

In conclusion, we provide an overview of this growing research area, outline the basic techniques, and provide brief coverage of several representative application systems. Because data mining and knowledge discovery are a fast expanding field with many other new research results reported and new systems or prototypes developed recently, and researchers and developers in many other fields have contributed to this field, it is a challenging task to provide a comprehensive overview of the data mining methods and knowledge discovery techniques in a short article. This article is an attempt to provide a reasonably comprehensive survey on the data mining techniques and knowledge discovery systems developed recently. Given the broad spectrum viewpoints, our overview is inevitably limited in scope. But we hope this paper will lead to a better understanding of the variety of techniques and approaches in this multi-disciplinary field.

Acknowledgments We would like to thank Mr. Gregory Piatetsky-Shapiro, Mr. Jiawei Han and other KDD-96 referees for their valuable suggestions and ideas.

References

[1] Fayyad U, Piatetsky-Shapiro G, Smyth P. Knowledge discovery and data mining: Towards a unifying framework. In *Proc. KDD-96: Second International Conference on Knowledge Discovery & Data Mining,* Menlo Park, CA: AAAI Press, 1996, pp.82–88.

[2] Matheus Christopher J, Chan Philip K, Piatetsky-Shapiro G. Systems for knowledge discovery in databases. *IEEE Trans. Knowl. Data Eng.*, 1993, 5(6).

[3] Knowledge Discovery Nuggets on the Internet: http://www.nuggets.com/.

[4] Fayyad U, Piatetsky-Shapiro G, Smyth P. From data mining to knowledge discovery in databases. *AI Magazine*, Fall, 1996, pp.37–54.

[5] Agrawal R, Srikant R. Fast algorithms for mining association rules. In *Proc. Int'l Conf. Very Large Databases*, 1994, pp.487–499.

[6] Agrawal R, Imielinski T, Swami A. Mining association rules between sets of items in large databases. In *Proc. ACM SIGMOD*, May 1993, pp.207–216.

[7] Srikant R, Agrawal R. Mining generalized association rules. In *Proc. 21st Int'l Conf. Very Large Databases*, September 1995, pp.407–419.

[8] Han Jiawei, Fu Yong. Discovery of multi-level association rules from large databases. In *Proc. Int'l Conf. Very Large Databases*, 1995, pp.420–431.

[9] Shen K, Ong, Mitbander B, Zaniolo C. Metaqueries for data mining. In *Advances in Knowledge Discovery and Data Mining*, Fayyad R U (*eds.*), AAAI/MIT Press, 1996, pp.375–398.

[10] Park J S, Chen M S, Yu P S. An effective hash based algorithm for mining association rules. In *Proc. ACM SIGMOD*, May 1995, pp.175–186.

[11] Fu Y, Han J. Meta-rule-guided mining of association rules in relational databases. In *Proc. 1st Int'l Workshop on Integration of Knowledge Discovery with Deductive and Object-Oriented Databases (KDOOD'95)*, Singapore, Dec. 1995, pp.39–46.

[12] Piatetsky-Shapiro G. Discovery, analysis, and presentation of strong rules. In *Knowledge Discovery in Databases*, Piatetsky-Shapiro G, Frawley W J (eds.), AAAI/MIT Press, 1991, pp.229–238.

[13] Silberschatz A, Tuzhilin A. On subjective measure of interestingness in knowledge discovery. In *Proc. 1st Int'l Conf. Knowledge Discovery and Data Mining (KDD95)*, Montreal, Canada, Aug. 1995, pp.275–281.

[14]] Harinarayan V, Ullman J D, Rajaraman A. Implementing data cubes efficiently. In *Proc. 1996 ACM-SIGMOD Int'l Conf. Management of Data*, Montreal, Canada, June 1996.

[15] Gupta A, Harinarayan V, Quass D. Aggregate-query processing in data warehousing environment. In *Proc. 21st Int'l Conf. Very Large Data Bases*, Zurich, Switzerland, September 1995, pp.358–369.

[16] Widom J. Research problems in data warehousing. In *Proc. 4th Int'l Conf. Information and Knowledge Management*, Baltimore, Maryland, Nov. 1995, pp.25–30.

[17] Han J, Cai Y, Cercone N. Data-driven discovery of quantitative rules in relational databases. *IEEE Trans. Knowledge and Data Engineering*, 1993, 3: 29–40.

[18] Cai Y, Cercone N, Han J. Attribute-Oriented Induction in Relational Databases. In *Knowledge Discovery in Database*, 1991, pp.213–228.

[19] Han J, Fu Y. Exploration of the Power of Attribute-Oriented Induction in Data Mining. In *Advances in Knowledge Discovery and Data Mining*, Fayyad U M, Piatetsky-Shapiro G *et al.* (eds.), AAAI/MIT Press, 1996, pp.399–421.

[20] Han J, Cai Y, Cercone N. Knowledge discovery in databases: An attribute-oriented approach. In *Proc. 18th International Conference on Very Large Databases*, Aug. 1992, pp.547–559.

[21] Li Deyi, Shi Xuemei, Meng Haijun. Membership clouds and clouds generators. *The Research and Development of Computers*, 1995, 42(8): 32–41.

[22] Han Ke. The discovery state space theory and its applications. Ph.D. dissertation, Communication Engineering Institute, Nanjing, China, 1996.

[23] Han J, Fu Y. Dynamic generation and refinement of concept hierarchies for knowledge discovery in databases. In *Proc. AAAI'94 Workshop Knowledge Discovery in Databases*, Seattle, July 1994, pp.157–168.

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

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

[26] Shan Ning, Ziarko W, Hamilton H J, Cercone N. Discovering classification knowledge in databases using rough sets. In *Proc. KDD-96: Second International Conference on Knowledge Discovery & Data Mimng*, Menlo Park, CA: AAAI Press, 1996, pp.271–274.

[27] Agrawal A, Ghosh S, Imielinkski T, Iyer B, Swami A. An interval classifier for database mining applications. In *Proc. 18th Int'l Conf. Very Large Data Bases,* August 1992, pp.560–573.

[28] Ng R, Han J. Efficient and effective clustering method for spatial data mining. In *Proc. International Conference on Very Large Databases,* Santiago, Chile, September 1994, pp.144–155.

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

[30] Ester M, Kriegel H P, Xu X. Knowledge discovery in large spatial databases: Focusing techniques for efficient class identification. In *Proc. 4th Int'l Symposium on Large Spatial Databases (SSD95),* Portland, Maine, August 1995, pp.67–82.

[31] Zhang T, Ramakrishnan R, Livny M. BIRCH: An efficient data clustering method for very large databases. In *Proc. ACM-SIGMOD,* Montreal, Canada, June 1996.

[32] Agrawal R, Faloutsos C, Swami A. Efficient similarity search in sequence databases. In *Proc. 4th Int'l Conf. Foundations of Data Organization and Algorithms,* October 1993.

[33] Faloutsos C, Ranganathan M, Manolopoulos Y. Fast subsequence matching in time-series databases. In *Proc. ACM SIGMOD,* Minneapolis, MN, May 1994, pp.419–429.

[34] Agrawal R, Lin K I, Sawhney H S, Shim K. Fast similarity search in the presence of noise, scaling, and translation in time-series databases. In *Proc. 21st Int'l Conf. Very Large Databases,* September 1995, pp.490–501.

[35] Li C S, Yu P S, Castelli V. HierarchyScan: A hierarchical similarity search algorithm for databases of long sequences. In *Proc. 12th Int'l Conf. Data Engineering,* February 1996.

[36] Flockhart I W, Radcliffe N J. A genetic algorithm-based approach to data mining. In *Proc. KDD-96: Second Int'l Conf. Knowledge Discovery & Data Mining,* Menlo Park, CA: AAAI Press, 1996, pp.299–302.

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

[38] Schmitz J, Armstrong G, Little J D C. CoverStory automated news finding in marketing. In *DSS Transactions,* Institute of Management Sciences, Providence, RI, 1990.

[39] Hoschka P, Klosgen W. A support system for interpreting statistical data. In *Knowledge Discovery in Databases,* Piatetsky-Shapiro G, Frawley W (eds.), Cambridge, MA: AAAI/MIT, 1991, pp.325–345.

[40] Piatetsky-Shapiro G, Matheus C J. Knowledge discovery workbench: An exploratory environment for discovery in business databases. In *Workshop Notes from the 9th National Conference on Artificial Intelligence: Knowledge Discovery in Databases,* Anaheim, CA, July 1991, pp.11–24.

[41] Piatetsky-Shapiro G. Discovery, Analysis, and Presentation of Strong Rules. In *Knowledge Discovery in Databases,* Cambridge, MA: AAAI/MIT, 1991, pp.229–248.

[42] Piatetsky-Shapiro G (ed.). Workshop Notes from the 9th Nar. Conf. Art. Intell.: Knowledge Discovery in Databases, Anaheim, CA, July, 1991.

[43] Piatetsky-Shapiro G. Probabilistic data dependencies. In *Proc. Mach. Discovery Work,* (9th Mach. Learn. Conf.), Aberdeen, Scotland, 1992, pp.11–17.

[44] Han Jiawei, Fu Yongjian *et al.* DBMiner: A system for mining knowledge in large relational database. In *Proc. KDD-96: Second Int'l Conf. Knowledge Discovery & Data Mining,* Menlo Park, CA: AAAI Press, 1996, pp.250–255.

[45] Roddick John F, Craske Noel G, Richards Thomas J. Handling discovered structure in database systems. *IEEE Trans. Knowledge and Data Engineering,* April 1996, 8(2): 227–240.

[46] Han Jiawei, Huang Yue, Cercone Nick, Fu Yongjian. Intelligent query answering by knowledge discovery techniques. *IEEE Trans. Knowledge and Data Engineering,* June 1996, 8(3): 373–390.

[47] Rakesh Agrawal, Manish Mehta, John Shafer, Ramakrishnan Srikant. The quest data mining system. In *Proc. KDD-96: Second Int'l Conf. Knowledge Discovery & Data Mining,* Menlo Park, CA: AAAI Press, 1996, pp.244–249.

[48] Srikant R, Agrawal R. Mining quantitative association rules in large relational tables. In *Proc. ACM SIGMOD Conf. Management of Data,* 1996.

[49] Srikant R, Agrawal R. Mining sequential patterns: Generalizations and performance improvements. In *Proc. Fifth Int'l Conf. Extending Database Technology (EDBT),* 1996.

[50] Piatetsky-Shapiro G, Brachman R, Khabaza T *et al.* An overview of issues in developing industrial data mining and knowledge discovery applications. In *Proc. KDD-96: Second Int'l Conf. Knowledge Discovery & Data Mining,* Menlo Park, CA: AAAI Press, 1996, pp.89–95.

[51] Selinger P G. Predictions and challenges for database systems in the year 2000. In *Proc. 19th Int'l Conf. Very Large Databases,* Agrawal R, Baker S, Bell D (eds.), Dublin, Ireland, 1993, pp.667–675.

[52] Fayyad U, Haussler D, Stolorz P. KDD for science data analysis: Issues and examples. In *Proc. KDD-96: Second Int'l Conf. Knowledge Discovery & Data Mining,* Menlo Park, CA: AAAI Press, 1996, pp.50–56.

[53] Frawley W J, Piatetsky-Shapiro G, Matheus C J. Knowledge discovery in databases: An overview. In *Knowledge Discovery in Databases,* Cambridge, MA: AAAI/MIT, 1991, pp.1–27. Reprinted in *AI Magazine,* 1992, 13(3): 1–27.

[54] Jain A K, Dubes R C. Algorithms for Clustering Data. Prentice-Hall, 1988.

[55] Fisher D. Optimization and simplification of hierarchical clustering. In *Proc. 1st Int'l Conf. Knowledge Discovery and Data Mining (KDD95),* Montreal, Canada, August 1995, pp.118-123.

[56] Cheeseman P, Stutz J. Bayesian classification (AutoClass): Theory and results. In *Advances in Knowledge Discovery and Data Mining,* Fayyad U M, Piatetsky-Shapiro G *et al.* (eds.), AAAI/MIT Press, 1996, pp.153–180.

[57] Li D, Shi X, Ward P, Gupta M M. Soft inference mechanism based on cloud models. In *Proc. First Int'l Workshop on Logic Programming and Soft Computing,* Francesca Arcelli Fontana, Ferrante Formato and Trevor P. Martin (eds.), Bonn, Germany, Sept. 6, 1996, pp.38–62.

[58] Li Deyi, Shi Xuemei, Vincent N G. On representing uncertainty in commonsense knowledge. In *Proc. Joint 1997 Pacific Asia Conf. Expert Systems/Singapore Int'l Conf. Intelligent Systems (PACES/SPICIS 97),* Orchard Hotel, Singapore, Feb. 1997, pp.291–298.

[59] Kaufman K A, Michalski R S, Kerschberg L. Mining for Knowledge in Databases: Goals and General Description of the INLEN System. In *Knowledge Discovery in Databases,* Piatetsky-Shapiro G, Frawley W J (eds.), AAAI/MIT Press, 1991, pp.449–462.

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

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

[62] Mehta M, Agrawal R, Rissanen J. A fast scaleable classifier for data mining. In *Proc. Fifth Int'l Conf. Extending Database Technology,* 1996.

[63] Arning A, Agrawal R. A linear method for deviation detection in large databases. In *Proc. 2nd Int'l Conf. Knowledge Discovery in Databases and Data Mining,* 1996.

[64] O'Leary D E. Knowledge discovery as a threat to database security. In *Knowledge Discovery in Databases,* Piatetsky-Shapiro G, Frawley W J (eds.), AAAI/MIT Press, 1991, pp.229–238.

[65] Piatetsky-Shapiro G, Matheus C J. Knowledge discovery workbench fo exploring business databases. *Int'l J. Intelligent Systems,* 1992, 7: 675–686.

Fan Jianhua is a Ph.D. candidate in Department of Computer Science, Nanjing Communications Engineering Institute. His current research interests include data mining, C^3I systems.

Li Deyi graduated from Department of Electronic Engineering, Southeast University in 1967, and received his Ph.D. degree in computer science from Heriot-Watt University, Edinburgh in 1983. He is presently a Professor and the Chief-Engineer in the Institute of Beijing Electroinc System Engineering. His research interests include data mimng, fuzzy control, system simulation and C^3I systems.