UNIT – V

Advanced Concepts: Basic concepts in Mining data streams–Mining Time–series data—Mining sequence patterns in Transactional databases– Mining Object– Spatial– Multimedia–Text and Web data – Spatial Data mining– Multimedia Data mining–Text Mining– Mining the World Wide Web.
Data Mining – Concepts and Techniques – Jiawei Han & Micheline Kamber Second Edition

1.Explain how mining can be performed on stream data?

Data Stream mining: Data Stream is a continuous, fast-changing, and ordered chain of data transmitted at a very high speed. It is an ordered sequence of information for a specific interval. The sender's data is transferred from the sender's side and immediately shows in data streaming at the receiver's side. Streaming does not mean downloading the data or storing the information on storage devices.

What are Data Streams in Data Mining?

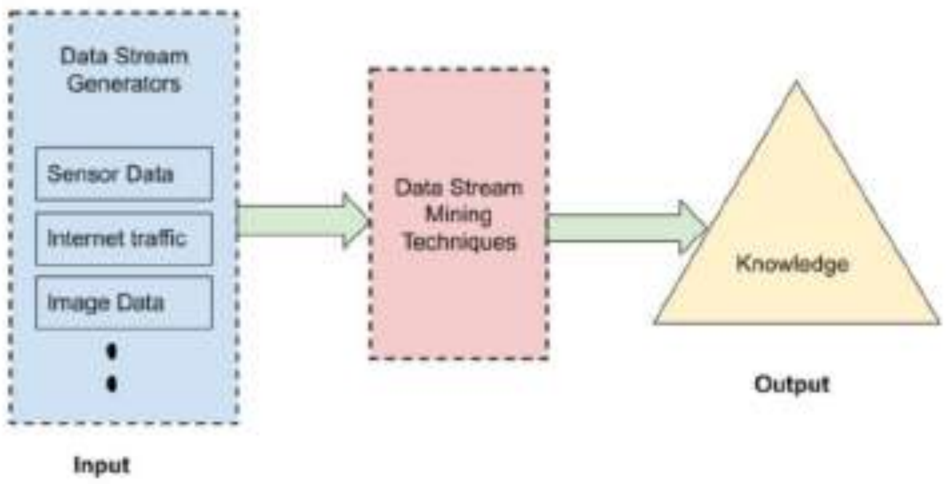


Image Source: SelfData Streams in Data Mining is extracting knowledge and valuable insights from a continuous stream of data using stream processing software. Data Streams in Data Mining can be considered a subset of general concepts of machine learning, knowledge extraction, and data mining. In Data Streams in Data Mining, data analysis of a large amount of data needs to be done in real-time. The structure of knowledge is extracted in data steam mining represented in the case of models and patterns of infinite streams of information.

Characteristics of Data Stream in Data Mining

Data Stream in Data Mining should have the following characteristics:

- **Continuous Stream of Data**: The data stream is an infinite continuous stream resulting in big data. In data streaming, multiple data streams are passed simultaneously.

- **Time Sensitive**: Data Streams are time-sensitive, and elements of data streams carry timestamps with them. After a particular time, the data stream loses its significance and is relevant for a certain period.
- **Data Volatility**: No data is stored in data streaming as It is volatile. Once the data mining and analysis are done, information is summarized or discarded.
- **Concept Drifting**: Data Streams are very unpredictable. The data changes or evolves with time, as in this dynamic world, nothing is constant.

Data Stream is generated through various data stream generators. Then, data mining techniques are implemented to extract knowledge and patterns from the data streams. Therefore, these techniques need to process multi-dimensional, multi-level, single pass, and online data streams.

Data Based Techniques

Sampling is based on selecting subset of data uniformly distributed.

- Reservoir Sampling *(Fixed Size Sample)*This algorithm sample a subset *m* of the data from the stream. After the i'th item is chosen with probability 1/i and if the i element is already sampled its randomly replace a sampled item

Sketching is based on reducing the dimensionality of the dataset. Sketch is based on performing a linear transformation of the data delivering a summarization of the Stream.

Sliding windows: The Sliding window is a problem-solving technique of data structure and algorithm for problems that apply arrays or lists. These problems are painless to solve using a brute force approach in O(n^2) or O(n^3). However, the Sliding window technique can reduce the time complexity to O(n)

Histograms: A histogram is a graph that shows the frequency of numerical data using rectangles. The height of a rectangle (the vertical axis) represents the distribution frequency of a variable (the amount, or how often that variable appears).

Multiresolution methods: A common way to deal with a large amount of **data** is through the use of **data** reduction methods

Randomised Algorithms:in the form of sampling and sketching are often used to deal with massive ,high dimensional data streams.

Chebyshevs inequality let x be a random variable with mean µand and standard deviation then

$P(|x-\mu|>k)<=(sd)^2/k^2$

For any positive real number k

Data stream Management system:They arrive online and are continuous ,temporally ordered and potentially infinite.

Stream queries ,includes three parts end user,query processor and scratch space.continious query is evaluated continuously as data stream continue to arrive.

2.Explain the how Mining can be performed on Time-Series Data.

“What is a time-series database?” A time-series database consists of sequences of values or events obtained over repeated measurements of time. The values are typically measured at equal time intervals (e.g., hourly, daily, weekly). Time-series databases are popular in many applications, such as stock market analysis, economic and sales forecasting, budgetary analysis, utility studies, inventory studies, yield projections, workload projections, process and quality control, observation of natural phenomena (such as atmosphere, temperature, wind, earthquake), scientific and engineering experiments, and medical treatments. A time-series database is also a sequence database. However, a sequence database is any database that consists of sequences of ordered events, with or without concrete notions of time. For example, Web page traversal sequences and customer shopping transaction sequences are sequence data, but they may not be time-series data.

With the growing deployment of a large number of sensors, telemetry devices, and other on-line data collection tools, the amount of time-series data is increasing rapidly, often in the order of gigabytes per day (such as in stock trading) or even per minute (such as from NASA space programs). How can we find correlation relationships within time-series data? How can we analyze such huge numbers of time series to find similar or regular patterns, trends, bursts (such as sudden sharp changes), and outliers, with fast or even on-line real-time response.This has become an increasingly important and challenging problem. In this section, we examine several aspects of mining time-series databases, with a focus on trend analysis and similarity search.

1. Trend Analysis

A time series involving a variable Y, representing, say, the daily closing price of a share in a stock market, can be viewed as a function of time t, that is, Y = F(t). Such a function can be illustrated as a time-series graph, as shown in Figure 8.4, which describes a point moving with the passage of time.
"How can we study time-series data?" In general, there are two goals in time-series analysis: (1) modeling time series (i.e., to gain insight into the mechanisms or underlying forces that generate the time series), and (2) forecasting time series (i.e., to predict the future values of the time-series variables).
Trend analysis consists of the following four major components or movements for characterizing time-series data:
Trend or long-term movements: These indicate the general direction in which a time series graph is moving over a long interval of time. This movement is displayed by a trend curve, or a trend line. For example, the trend curve of Figure 8.4 is indicated by a dashed curve. Typical methods for determining a trend curve or trend line include the weighted moving average method and the least squares method, discussed later.

Cyclic movements or cyclic variations: These refer to the cycles, that is, the long-term oscillations about a trend line or curve, which may or may not be periodic. That is, the cycles need not necessarily follow exactly similar patterns after equal intervals of time.

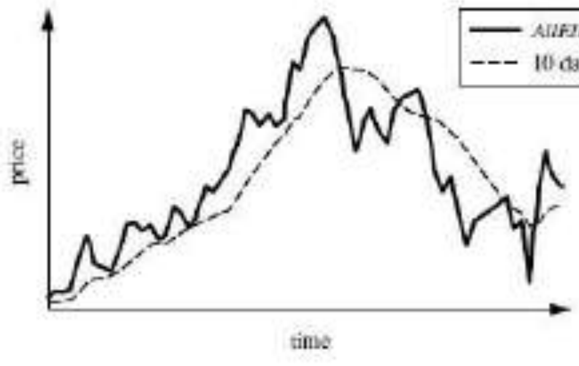


Figure 8.4 Time-series data of the stock price of *AllElectronics* over time. The *trend* is shown with a dashed curve, calculated by a moving average.

Seasonal movements or seasonal variations: These are systematic or calendar related. Examples include events that recur annually, such as the sudden increase in sales of chocolates and flowers before Valentine's Day or of department store items before Christmas. The observed increase in water consumption in summer due to warm weather is another example. In these examples, seasonal movements are the identical

or nearly identical patterns that a time series appears to follow during corresponding months of successive years.
Irregular or random movements: These characterize the sporadic motion of time series due to random or chance events, such as labor disputes, floods, or announced personnel changes within companies.

3. Explain how Mining can be performed on Sequence Patterns in Transactional Databases.

A sequence database consists of sequences of ordered elements or events, recorded with or without a concrete notion of time. There are many applications involving sequence data. Typical examples include customer shopping sequences, Web clickstreams, biological sequences, sequences of events in science and engineering, and in natural and social developments. In this section, we study sequential pattern mining in transactional databases. In particular, we start with the basic concepts of sequential pattern mining in Section 8.3.1. Section 8.3.2 presents several scalable methods for such mining. Constraint-based sequential pattern mining is described in Section 8.3.3. Periodicity analysis for sequence data is discussed in Section 8.3.4. Specific methods for mining sequence patterns in biological data are addressed in Section 8.4.

1 Sequential Pattern Mining: Concepts and Primitives

"What is sequential pattern mining?" Sequential pattern mining is the mining of frequently occurring ordered events or subsequences as patterns. An example of a sequential pattern is "Customers who buy a Canon digital camera are likely to buy an HP color printer within a month." For retail data, sequential patterns are useful for shelf placement and promotions. This industry, as well as telecommunications and other businesses, may also use sequential patterns for targeted marketing, customer retention, and many other tasks. Other areas in which sequential patterns can be applied include Web access pattern analysis, weather prediction, production processes, and network intrusion detection. Notice that most studies of sequential pattern mining concentrate on categorical (or symbolic) patterns,whereas numerical curve analysis usually belongs to the scope of trend analysis and forecasting in statistical time-series analysis, as discussed in Section 8.2.

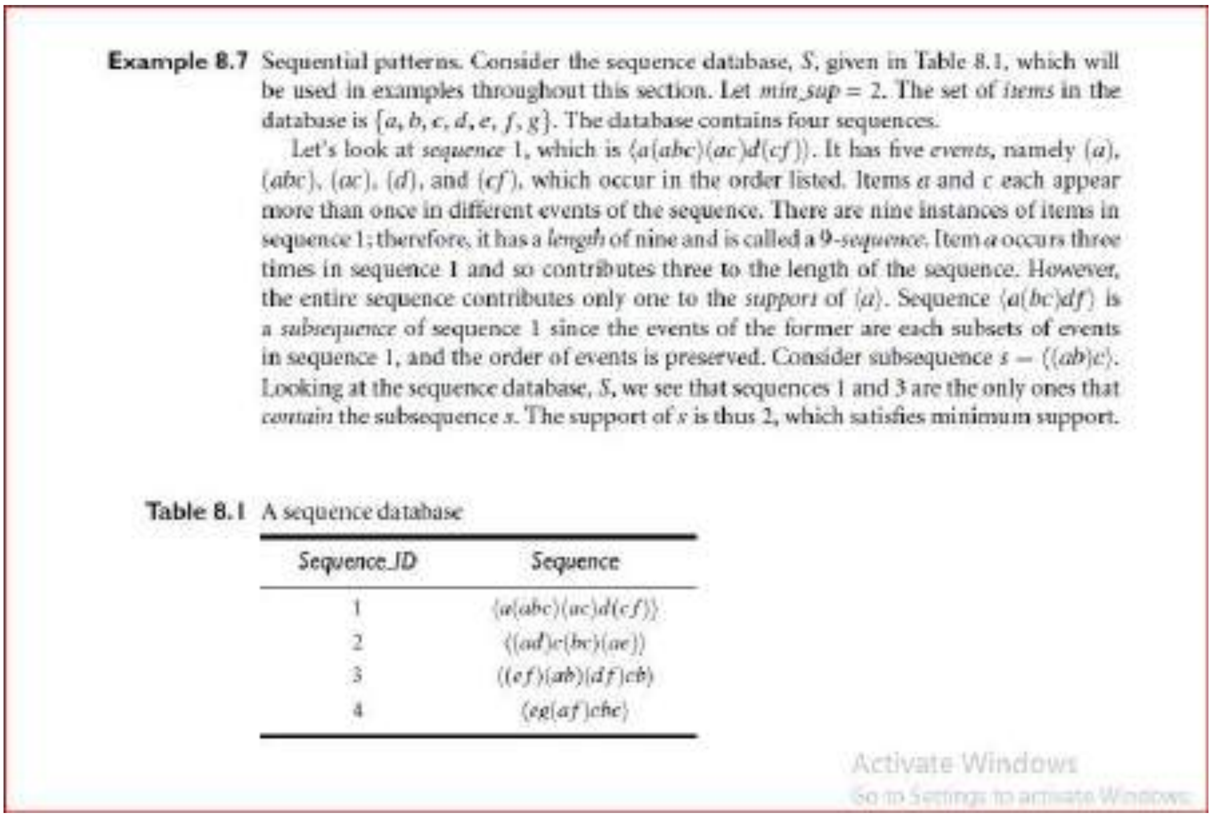
The sequential pattern mining problem was first introduced by Agrawal and Srikant in 1995 [AS95] based on their study of customer purchase sequences, as follows: "Given a set of sequences, where each sequence consists of a list of events (or elements) and each event consists of a set of items, and given a user-specified minimum support threshold of min sup, sequential pattern mining finds all frequent subsequences, that is, the subsequences whose occurrence frequency in the set of sequences is no less than min sup."

Let's establish some vocabulary for our discussion of sequential pattern mining. Let

Let's establish some vocabulary for our discussion of sequential pattern mining. Let $I = \{I_1, I_2, \ldots, I_p\}$ be the set of all *items*. An **itemset** is a nonempty set of items. A **sequence** is an ordered list of events. A sequence s is denoted $\langle e_1 e_2 e_3 \cdots e_l \rangle$, where event e_1 occurs before e_2, which occurs before e_3, and so on. Event e_j is also called an **element** of s. In the case of customer purchase data, an event refers to a shopping trip in which a customer bought items at a certain store. The event is thus an itemset, that is, an unordered list of items that the customer purchased during the trip. The itemset (or event) is denoted $(x_1 x_2 \cdots x_q)$, where x_k is an item. For brevity, the brackets are omitted if an element has only one item, that is, element (x) is written as x. Suppose that a customer made several shopping trips to the store. These ordered events form a sequence for the customer. That is, the customer first bought the items in s_1, then later bought

the items in s_2, and so on. An item can occur at most once in an event of a sequence, but can occur multiple times in different events of a sequence. The number of instances of items in a sequence is called the **length** of the sequence. A sequence with length l is called an l**-sequence**. A sequence $\alpha = \langle a_1 a_2 \cdots a_n \rangle$ is called a **subsequence** of another sequence $\beta = \langle b_1 b_2 \cdots b_m \rangle$, and β is a **supersequence** of α, denoted as $\alpha \sqsubseteq \beta$, if there exist integers $1 \leq j_1 < j_2 < \cdots < j_n \leq m$ such that $a_1 \subseteq b_{j_1}, a_2 \subseteq b_{j_2}, \ldots, a_n \subseteq b_{j_n}$. For example, if $\alpha = \langle (ab), d \rangle$ and $\beta = \langle (abc), (de) \rangle$, where a, b, c, d, and e are items, then α is a subsequence of β and β is a supersequence of α.

A **sequence database**, S, is a set of tuples, $\langle SID, s \rangle$, where SID is a *sequence_ID* and s is a sequence. For our example, S contains sequences for all customers of the store. A tuple $\langle SID, s \rangle$ is said to contain a sequence α, if α is a subsequence of s. The **support** of a sequence α in a sequence database S is the number of tuples in the database containing α, that is, $support_S(\alpha) = |\ \{\langle SID, s \rangle | (\langle SID, s \rangle \in S) \wedge (\alpha \sqsubseteq s)\}\ |$. It can be denoted as $support(\alpha)$ if the sequence database is clear from the context. Given a positive integer min_sup as the **minimum support threshold**, a sequence α is frequent in sequence database S if $support_S(\alpha) \geq min\_sup$. That is, for sequence α to be frequent, it must occur at least min_sup times in S. *A frequent sequence is called a* **sequential pattern**. A sequential pattern with length l is called an l**-pattern**. The following example illustrates these concepts.



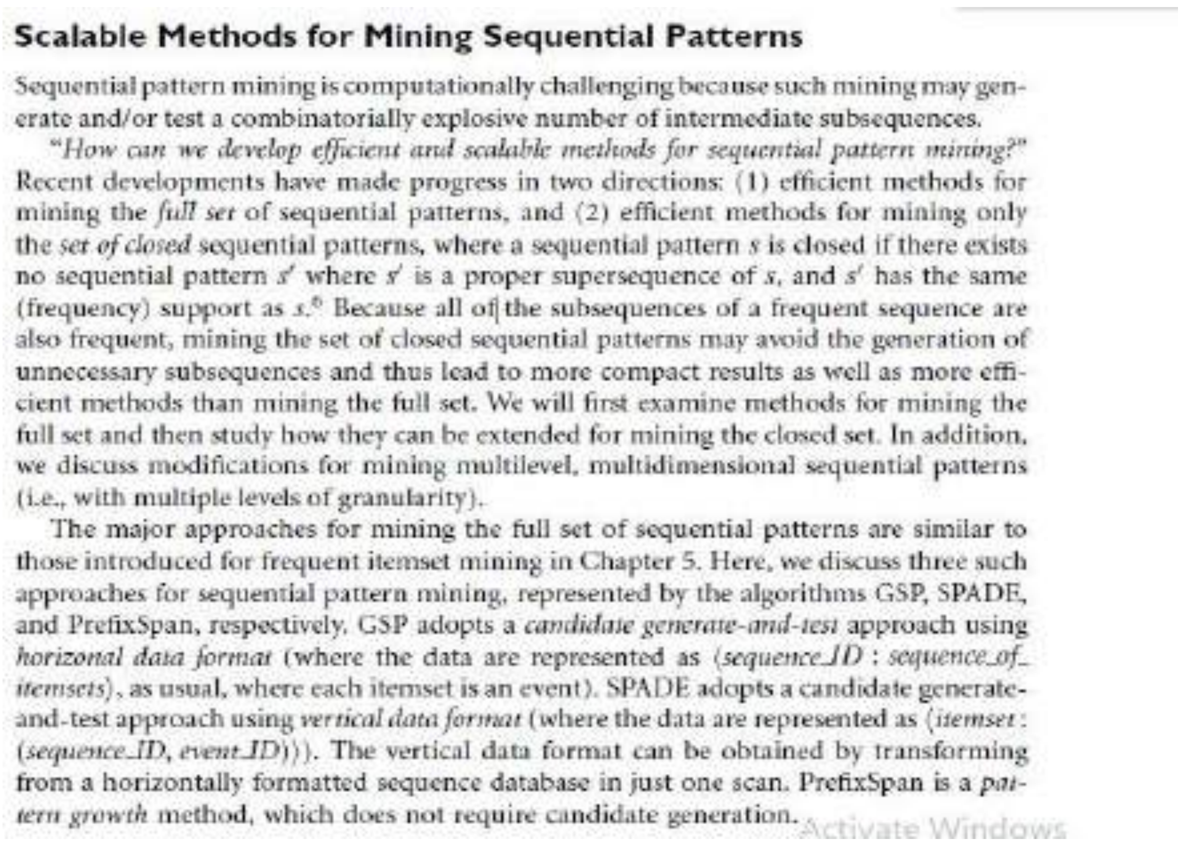
Example 8.7 Sequential patterns. Consider the sequence database, S, given in Table 8.1, which will be used in examples throughout this section. Let *min_sup* = 2. The set of *items* in the database is $\{a, b, c, d, e, f, g\}$. The database contains four sequences.

Let's look at *sequence* 1, which is $\langle a(abc)(ac)d(cf)\rangle$. It has five *events*, namely (a), (abc), (ac), (d), and (cf), which occur in the order listed. Items a and c each appear more than once in different events of the sequence. There are nine instances of items in sequence 1; therefore, it has a *length* of nine and is called a 9-*sequence*. Item a occurs three times in sequence 1 and so contributes three to the length of the sequence. However, the entire sequence contributes only one to the *support* of $\langle a\rangle$. Sequence $\langle a(bc)df\rangle$ is a *subsequence* of sequence 1 since the events of the former are each subsets of events in sequence 1, and the order of events is preserved. Consider subsequence $s = \langle (ab)c\rangle$. Looking at the sequence database, S, we see that sequences 1 and 3 are the only ones that *contain* the subsequence s. The support of s is thus 2, which satisfies minimum support.

Table 8.1 A sequence database

Sequence_ID	Sequence
1	$\langle a(abc)(ac)d(cf)\rangle$
2	$\langle (ad)c(bc)(ae)\rangle$
3	$\langle (ef)(ab)(df)cb\rangle$
4	$\langle eg(af)cbc\rangle$

Therefore, s is frequent, and so we call it a sequential pattern. It is a 3-pattern since it is a sequential pattern of length three.



Scalable Methods for Mining Sequential Patterns

Sequential pattern mining is computationally challenging because such mining may generate and/or test a combinatorially explosive number of intermediate subsequences.

"How can we develop efficient and scalable methods for sequential pattern mining?" Recent developments have made progress in two directions: (1) efficient methods for mining the *full set* of sequential patterns, and (2) efficient methods for mining only the *set of closed* sequential patterns, where a sequential pattern s is closed if there exists no sequential pattern s' where s' is a proper supersequence of s, and s' has the same (frequency) support as s.[8] Because all of the subsequences of a frequent sequence are also frequent, mining the set of closed sequential patterns may avoid the generation of unnecessary subsequences and thus lead to more compact results as well as more efficient methods than mining the full set. We will first examine methods for mining the full set and then study how they can be extended for mining the closed set. In addition, we discuss modifications for mining multilevel, multidimensional sequential patterns (i.e., with multiple levels of granularity).

The major approaches for mining the full set of sequential patterns are similar to those introduced for frequent itemset mining in Chapter 5. Here, we discuss three such approaches for sequential pattern mining, represented by the algorithms GSP, SPADE, and PrefixSpan, respectively. GSP adopts a *candidate generate-and-test* approach using *horizonal data format* (where the data are represented as $\langle$*sequence_ID* : *sequence_of_itemsets*$\rangle$, as usual, where each itemset is an event). SPADE adopts a candidate generate-and-test approach using *vertical data format* (where the data are represented as $\langle$*itemset* : (*sequence_ID*, *event_ID*)$\rangle$). The vertical data format can be obtained by transforming from a horizontally formatted sequence database in just one scan. PrefixSpan is a *pattern growth* method, which does not require candidate generation.

All three approaches either directly or indirectly explore the Apriori property, stated as follows: every nonempty subsequence of a sequential pattern is a sequential pattern. (Recall that for a pattern to be called sequential, it must be frequent. That is, it must satisfy minimum support.) The Apriori property is anti monotonic (or downward-closed) in that, if a sequence cannot pass a test (e.g., regarding minimum support), all of its super sequences will also fail the test. Use of this property to prune the search space can help make the discovery of sequential patterns more efficient.

Mining Multidimensional, Multilevel Sequential Patterns

Sequence identifiers (representing individual customers, for example) and sequence items (such as products bought) are often associated with additional pieces of information. Sequential pattern mining should take advantage of such additional information to discover interesting patterns in multidimensional, multilevel information space. Take customer shopping transactions, for instance. In a sequence database for such data, the additional information associated with sequence IDs could include customer age, address, group, and profession. Information associated with items could include item category, brand, model type, model number, place manufactured, and manufacture date. Mining multidimensional, multilevel sequential patterns is the discovery of interesting patterns in such a broad dimensional space, at different levels of detail.

Example 8.12 Multidimensional, multilevel sequential patterns. The discovery that "Retired customers who purchase a digital camera are likely to purchase a color printer within a month" and that "Young adults who purchase a laptop are likely to buy a flash drive within two weeks" are examples of multidimensional, multilevel sequential patterns. By grouping customers into "retired customers" and "young adults" according to the values in the age dimension, and by generalizing items to, say, "digital camera" rather than a specific model, the patterns mined here are associated with additional dimensions and are at a higher level of granularity.

"Can a typical sequential pattern algorithm such as PrefixSpan be extended to efficiently mine multidimensional, multilevel sequential patterns?" One suggested modification is to associate the multidimensional, multilevel information with the sequence ID and item ID, respectively, which the mining method can take into consideration when finding frequent subsequence. For example, (Chicago, middle aged, business) can be associated with sequence ID 1002 (for a given customer), whereas (Digital camera, Canon, Supershot, SD400, Japan, 2005) can be associated with item ID 543005 in the sequence. A sequential pattern mining algorithm will use such information in the mining process to find sequential patterns associated with multidimensional, multilevel information.

Constraint-Based Mining of Sequential Patterns

As shown in our study of frequent-pattern mining in Chapter 5, mining that is performed without user- or expert-specified constraints may generate numerous patterns that are of no interest. Such unfocused mining can reduce both the efficiency and usability of frequent-pattern mining. Thus, we promote constraint-based mining, which incorporates user-specified constraints to reduce the search space and derive only patterns that are of interest to the user.

Constraints can be expressed in many forms. They may specify desired relationships between attributes, attribute values, or aggregates within the resulting patterns mined. Regular expressions can also be used as constraints in the form of "pattern templates," which specify the desired form of the patterns to be mined. The general concepts introduced for constraint-based frequent pattern mining in Section 5.5.1 apply to constraint-based sequential pattern mining as well. The key idea to note is that these kinds of constraints can be used during the mining process to confine the search space, thereby improving (1) the efficiency of the mining and (2) the interestingness of the resulting patterns found. This idea is also referred to as "pushing the constraints deep into the mining process."

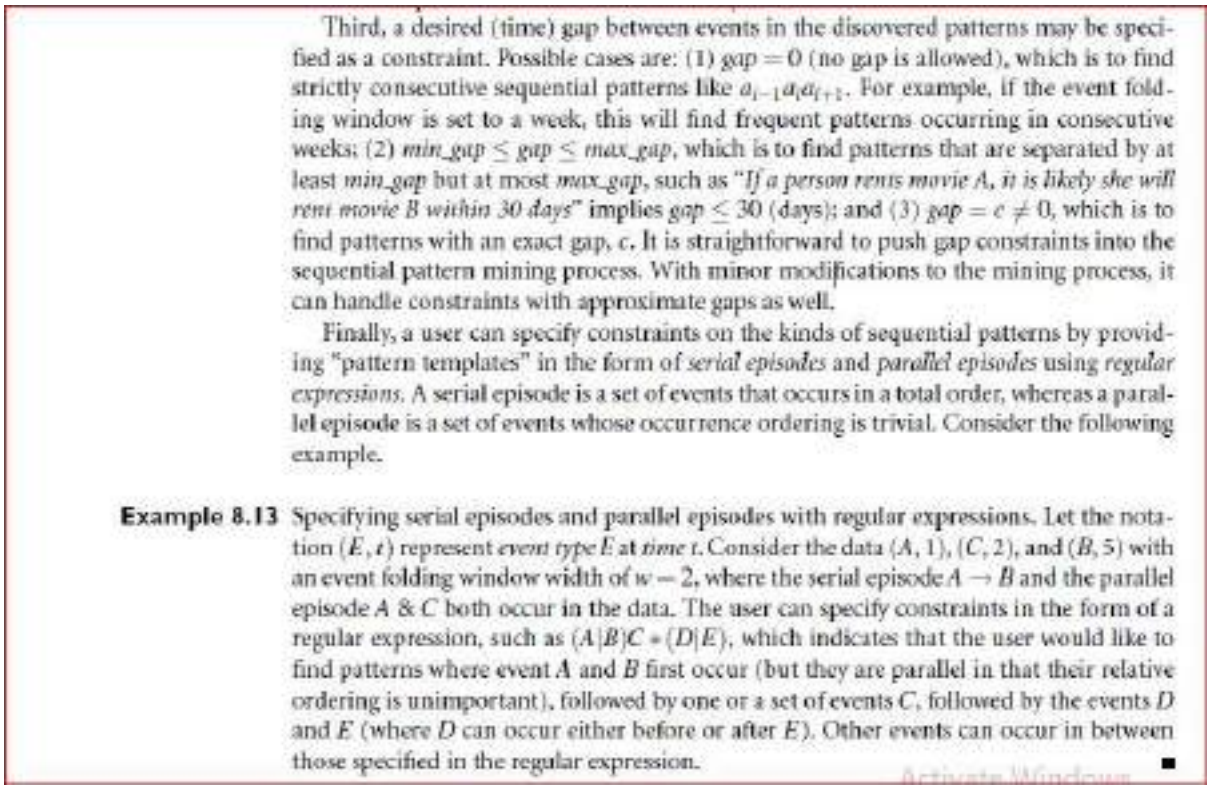
We now examine some typical examples of constraints for sequential pattern mining. First, constraints can be related to the duration, T, of a sequence. The duration may be the maximal or minimal length of the sequence in the database, or a user-specified duration related to time, such as the year 2005. Sequential pattern mining can then be confined to the data within the specified duration, T.

Constraints relating to the maximal or minimal length (duration) can be treated as antimonotonic or monotonic constraints, respectively. For example, the constraint T _10 is ant monotonic since, if a sequence does not satisfy this constraint, then neither will any of its super sequences (which are, obviously, longer). The constraint T >10 is monotonic. This means that if a sequence satisfies the constraint, then all of its super sequences will also satisfy the constraint. We have already seen several examples in this chapter of how antimonotonic constraints (such as those involving minimum support) can be pushed deep into the mining process to prune the search space. Monotonic constraints can be used in a way similar to its frequent-pattern counterpart as well.

Constraints related to a specific duration, such as a particular year, are considered succinct constraints. A constraint is succinct if we can enumerate all and only those sequences that are guaranteed to satisfy the constraint, even before support counting begins. Suppose, here, T = 2005. By selecting the data for which year = 2005, we can enumerate all of the sequences guaranteed to satisfy the constraint before mining begins. In other words, we don't need to generate and test. Thus, such constraints contribute toward efficiency in that they avoid the substantial overhead of the generate-and-test paradigm.

Durations may also be defined as being related to sets of partitioned sequences, such as every year, or every month after stock dips, or every two weeks before and after an earthquake. In such cases, periodic patterns (Section 8.3.4) can be discovered.

Second, the constraint may be related to an event folding window, w. A set of events occurring within a specified period can be viewed as occurring together. If w is set to be as long as the duration, T, it finds time-insensitive frequent patterns—these are essentially frequent patterns, such as "In 1999, customers who bought a PC bought a digital camera as well" (i.e., without bothering about which items were bought first). If w is set to 0 (i.e., no event sequence folding), sequential patterns are found where each event occurs at a distinct time instant, such as "A customer who bought a PC and then a digital camera is likely to buy an SD memory chip in a month." If w is set to be something in between (e.g., for transactions occurring within the same month or within a sliding window of 24 hours), then these transactions are considered as occurring within the same period, and such sequences are "folded" in the analysis.



Third, a desired (time) gap between events in the discovered patterns may be specified as a constraint. Possible cases are: (1) $gap = 0$ (no gap is allowed), which is to find strictly consecutive sequential patterns like $a_{i-1}a_ia_{i+1}$. For example, if the event folding window is set to a week, this will find frequent patterns occurring in consecutive weeks; (2) $min\_gap \leq gap \leq max\_gap$, which is to find patterns that are separated by at least *min_gap* but at most *max_gap*, such as "*If a person rents movie A, it is likely she will rent movie B within 30 days*" implies $gap \leq 30$ (days); and (3) $gap = c \neq 0$, which is to find patterns with an exact gap, c. It is straightforward to push gap constraints into the sequential pattern mining process. With minor modifications to the mining process, it can handle constraints with approximate gaps as well.

Finally, a user can specify constraints on the kinds of sequential patterns by providing "pattern templates" in the form of *serial episodes* and *parallel episodes* using *regular expressions*. A serial episode is a set of events that occurs in a total order, whereas a parallel episode is a set of events whose occurrence ordering is trivial. Consider the following example.

Example 8.13 Specifying serial episodes and parallel episodes with regular expressions. Let the notation (E, t) represent *event type E* at *time t*. Consider the data $(A, 1)$, $(C, 2)$, and $(B, 5)$ with an event folding window width of $w = 2$, where the serial episode $A \rightarrow B$ and the parallel episode A & C both occur in the data. The user can specify constraints in the form of a regular expression, such as $(A|B)C*(D|E)$, which indicates that the user would like to find patterns where event A and B first occur (but they are parallel in that their relative ordering is unimportant), followed by one or a set of events C, followed by the events D and E (where D can occur either before or after E). Other events can occur in between those specified in the regular expression. ■

Mining Object, Spatial, Multimedia, Text, and Web Data

Q4.What is Multidimensional Analysis and Descriptive Mining of Complex Data Objects?

Many advanced, data-intensive applications, such as scientific research and engineering design, need to store, access, and analyze complex but relatively structured data objects. These objects cannot be represented as simple and uniformly structured records (i.e., tuples) in data relations. Such application requirements have motivated the design and development of object-relational and object-oriented database systems. Both kinds of systems deal with the efficient storage and access of vast amounts of disk-based complex structured data objects. These systems organize a large set of complex data objects into classes, which are in turn organized into class/subclass hierarchies. Each object in a class is associated with

(1) an object-identifier,

(2) a set of attributes that may contain sophisticated data structures, set- or list-valued data, class composition hierarchies, multimedia data, and

(3) a set of methods that specify the computational routines or rules

associated with the object class. There has been extensive research in the field of database systems on how to efficiently index, store, access, and manipulate complex objects in object-relational and object-oriented database systems.

One step beyond the storage and access of massive-scaled, complex object data is the systematic analysis and mining of such data. This includes two major tasks: (1) construct multidimensional data warehouses for complex object data and perform online analytical processing (OLAP) in such data warehouses, and (2) develop effective and scalable methods for mining knowledge from object databases and/or data warehouses.

The second task is largely covered by the mining of specific kinds of data (such as spatial, temporal, sequence, graph- or tree-structured, text, and multimedia data), since these data form the major new kinds of complex data objects.
In this chapter we study methods for mining complex data. Thus, our focus in this section will be mainly on how to construct object data warehouses and perform OLAP analysis on data warehouses for such data.
A major limitation of many commercial data warehouse and OLAP tools for multidimensional
database analysis is their restriction on the allowable data types for dimensions and measures. Most data cube implementations confine dimensions to nonnumeric data, and measures to simple, aggregated values. To introduce data mining and multidimensional data analysis for complex objects, this section examines how to perform generalization on complex structured objects and construct object cubes for OLAP and mining in object databases.
To facilitate generalization and induction in object-relational and object-oriented databases, it is important to study how each component of such databases can be generalized, and how the generalized data can be used for multidimensional data analysis and data mining.

a)Generalization of Structured Data

An important feature of object-relational and object-oriented databases is their capability of storing, accessing, and modeling complex structure-valued data, such as set- and list-valued data and data with nested structures.
"How can generalization be performed on such data?" Let's start by looking at the generalization of set-valued, list-valued, and sequence-valued attributes.
A set-valued attribute may be of homogeneous or heterogeneous type. Typically, set-valued data can be generalized by (1) generalization of each value in the set to its corresponding higher-level concept, or (2) derivation of the general behavior of the set, such as the number of elements in the set, the types or value ranges in the set, the weighted average for numerical data, or the major clusters formed by the set. Moreover, generalization can be performed by applying different generalization operators to explore alternative generalization paths. In this case, the result of generalization is a heterogeneous set.

Example1: Generalization of a set-valued attribute.
Suppose that the hobby of a person is a set-valued attribute containing the set of values {tennis, hockey, soccer, violin, SimCity}. This set can be generalized to a set of high-level concepts, such as{sports, music, computer games} or into the number 5 (i.e., the number of hobbies in the set). Moreover, a count can be associated with a generalized value to indicate how many elements are generalized to that value, as in {sports(3),music(1), computer games(1)}, where sports(3) indicates three
kinds of sports, and so on.

Example2: Generalization of list-valued attributes.
Consider the following list or sequence of data for a person's education record: "((B.Sc. in Electrical Engineering, U.B.C., Dec., 1998), (M.Sc. in Computer Engineering, U. Maryland, May, 2001), (Ph.D. in Computer Science, UCLA, Aug., 2005))". This can be generalized by dropping less important descriptions (attributes) of each tuple in the list, such as by dropping the month attribute to obtain "((B.Sc., U.B.C., 1998), : : :)", and/or by retaining only the most important tuple(s) in the list, e.g., "(Ph.D. in Computer Science, UCLA, 2005)".

b)Aggregation and Approximation in Spatial and Multimedia Data Generalization

Aggregation and approximation are another important means of generalization. They
are especially useful for generalizing attributes with large sets of values, complex structures,
and spatial or multimedia data.
Let’s take spatial data as an example. We would like to generalize detailed geographic
points into clustered regions, such as business, residential, industrial, or agricultural
areas, according to land usage. Such generalization often requires the merge
of a set of geographic areas by spatial operations, such as spatial union or spatial
clustering methods. Aggregation and approximation are important techniques for
this form of generalization. In a spatial merge, it is necessary to not only merge the
regions of similar types within the same general class but also to compute the total
areas, average density, or other aggregate functions while ignoring some scattered
regions with different types if they are unimportant to the study. Other spatial operators,
such as spatial-union, spatial-overlapping, and spatial-intersection (which may
require the merging of scattered small regions into large, clustered regions) can also
use spatial aggregation and approximation as data generalization operators.

Example:3 Spatial aggregation and approximation. Suppose that we have different pieces of land for various purposes of agricultural usage, such as the planting of vegetables, grains, and fruits. These pieces can be merged or aggregated into one large piece of agricultural land by a spatial merge. However, such a piece of agricultural land may contain highways, houses, and small stores. If the majority of the land is used for agriculture, the scattered regions for other purposes can be ignored, and the whole region can be claimed as an agricultural area by approximation.

5.What is Spatial Data Mining?

A spatial database stores a large amount of space-related data, such as maps, pre-processed
remote sensing or medical imaging data, and VLSI chip layout data. Spatial
databases have many features distinguishing them from relational databases. They
carry topological and/or distance information, usually organized by sophisticated,
multidimensional spatial indexing structures that are accessed by spatial data access
methods and often require spatial reasoning, geometric computation, and spatial
knowledge representation techniques.
Spatial data mining refers to the extraction of knowledge, spatial relationships, or
other interesting patterns not explicitly stored in spatial databases. Such mining demands
an integration of data mining with spatial database technologies. It can be used for understanding spatial data, discovering spatial relationships and relationships between spatial and nonspatial data, constructing spatial knowledge bases, reorganizing spatial databases, and optimizing spatial queries. It is expected to have wide applications in geographic information systems, geomarketing, remote sensing, image database exploration, medical imaging, navigation, traffic control, environmental studies, and many other areas where spatial data are used. A crucial challenge to spatial data mining is the exploration of efficient spatial data mining techniques due to the huge amount of spatial data and the complexity of spatial data types and spatial access methods.
“What about using statistical techniques for spatial data mining?” Statistical spatial data

analysis has been a popular approach to analyzing spatial data and exploring geographic information. The term geostatistics is often associated with continuous geographic space, whereas the term spatial statistics is often associated with discrete space. In a statistical model that handles nonspatial data, one usually assumes statistical independence among different portions of data. However, different from traditional data sets, there is no such independence among spatially distributed data because in reality, spatial objects are often interrelated, or more exactly spatially co-located, in the sense that the closer the two objects are located, the more likely they share similar properties. For example, nature resource, climate, temperature, and economic situations are likely to be similar in geographically closely located regions. People even consider this as the first law of geography: "Everything is related to everything else, but nearby things are more related than distant things." Such a property of close interdependency across nearby space leads to the notion of spatial autocorrelation. Based on this notion, spatial statistical modelling methods have been developed with good success. Spatial data mining will further develop spatial statistical analysis methods and extend them for huge amounts of spatial data, with more emphasis on efficiency, scalability, cooperation with database and data warehouse systems, improved user interaction, and the discovery of new types of knowledge.

1. **Spatial Data Cube Construction and Spatial OLAP**

"Can we construct a spatial data warehouse?" Yes, as with relational data, we can integrate spatial data to construct a data warehouse that facilitates spatial data mining. A spatial data warehouse is a subject-oriented, integrated, time-variant, and non-volatile collection of both spatial and nonspatial data in support of spatial data mining and spatial-data related decision-making processes.

Let's look at the following example.

Example:5 Spatial data cube and spatial OLAP. There are about 3,000 weather probes distributed in British Columbia (BC), Canada, each recording daily temperature and precipitation for a designated small area and transmitting signals to a provincial weather station. With a spatial data warehouse that supports spatial OLAP, a user can view weather patterns on a map by month, by region, and by different combinations of temperature and precipitation, and can dynamically drill down or roll up along any dimension to explore desired patterns, such as "wet and hot regions in the Fraser Valley in Summer 1999."

There are three types of dimensions in a spatial data cube:
A nonspatial dimension contains only nonspatial data. Nonspatial dimensions temperature and precipitation can be constructed for the warehouse in Example 10.5, since each contains nonspatial data whose generalizations are nonspatial (such as "hot" for temperature and "wet" for precipitation).
A spatial-to-nonspatial dimension is a dimension whose primitive-level data are spatial but whose generalization, starting at a certain high level, becomes nonspatial. For example, the spatial dimension city relays geographic data for the U.S. map. Suppose that the dimension's spatial representation of, say, Seattle is generalized to the string "pacific northwest." Although "pacific northwest" is a spatial concept, its representation is not spatial (since, in our example, it is a string). It therefore plays the role of a nonspatial dimension.
Aspatial-to-spatial dimension is a dimension whose primitive level and all of its high-level

generalized data are spatial. For example, the dimension equi temperature region
contains spatial data, as do all its generalizations, such as with regions covering
0-5 degrees (Celsius), 5-10 degrees, and so on.
We distinguish two types of measures in a spatial data cube:
A numerical measure contains only numerical data. For example, one measure in a
spatial data warehouse could be the monthly revenue of a region, so that a roll-up may
compute the total revenue by year, by county, and so on. Numerical measures can be
further classified into distributive, algebraic, and holistic, as discussed in Chapter 3.
A spatial measure contains a collection of pointers to spatial objects. For example,
in a generalization (or roll-up) in the spatial data cube of Example 10.5, the regions
with the same range of temperature and precipitation will be grouped into the same
cell, and the measure so formed contains a collection of pointers to those regions.
A nonspatial data cube contains only nonspatial dimensions and numerical measures.
If a spatial data cube contains spatial dimensions but no spatial measures, its OLAP
operations, such as drilling or pivoting, can be implemented in a manner similar to that
for nonspatial data cubes.
"But what if I need to use spatial measures in a spatial data cube?" This notion raises
some challenging issues on efficient implementation, as shown in the following example.

Example:6 Numerical versus spatial measures.
A star schema for the BC weather warehouse of Example 10.5 is shown in Figure 10.2.

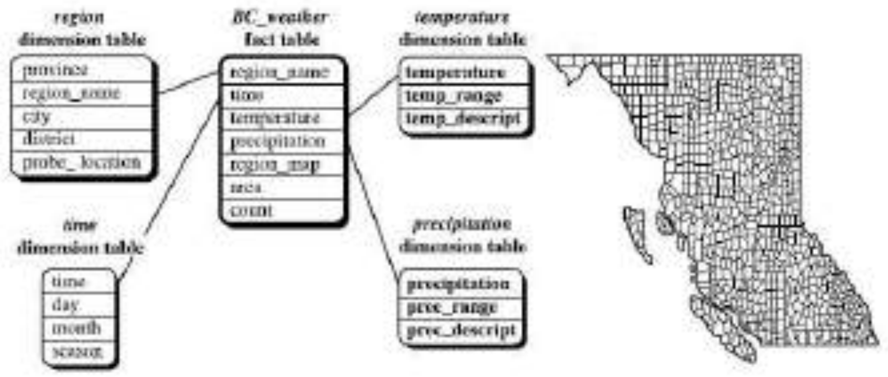
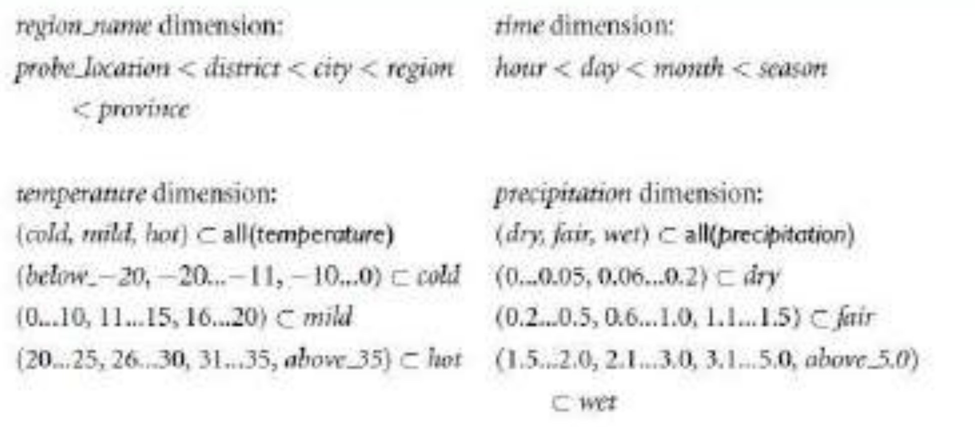


Figure 10.2 A star schema of the *BC_weather* spatial data warehouse and corresponding BC weather probes map.

It consists of four dimensions: region temperature, time, and precipitation, and three measures: region map, area, and count. A concept hierarchy for each dimension can be created by users or experts, or generated automatically by data clustering analysis.

Figure 10.3 presents hierarchies for each of the dimensions in the BC weather warehouse.



region_name dimension:
probe_location < *district* < *city* < *region* < *province*

time dimension:
hour < *day* < *month* < *season*

temperature dimension:
(*cold, mild, hot*) ⊂ all(*temperature*)
(*below_−20*, −20...−11, −10...0) ⊂ *cold*
(0...10, 11...15, 16...20) ⊂ *mild*
(20...25, 26...30, 31...35, *above_35*) ⊂ *hot*

precipitation dimension:
(*dry, fair, wet*) ⊂ all(*precipitation*)
(0...0.05, 0.06...0.2) ⊂ *dry*
(0.2...0.5, 0.6...1.0, 1.1...1.5) ⊂ *fair*
(1.5...2.0, 2.1...3.0, 3.1...5.0, *above_5.0*) ⊂ *wet*

Figure 10.3 Hierarchies for each dimension of the *BC_weather* data warehouse.

Of the three measures, area and count are numerical measures that can be computed similarly as for nonspatial data cubes; region map is a spatial measure that represents a collection of spatial pointers to the corresponding regions. Since different spatial OLAP operations result in different collections of spatial objects in region map, it is a major challenge to compute the merges of a large number of regions flexibly and dynamically. For example, two different roll-ups on the BC weather map data (Figure 10.2) may produce two different generalized region maps, as shown in Figure 10.4,

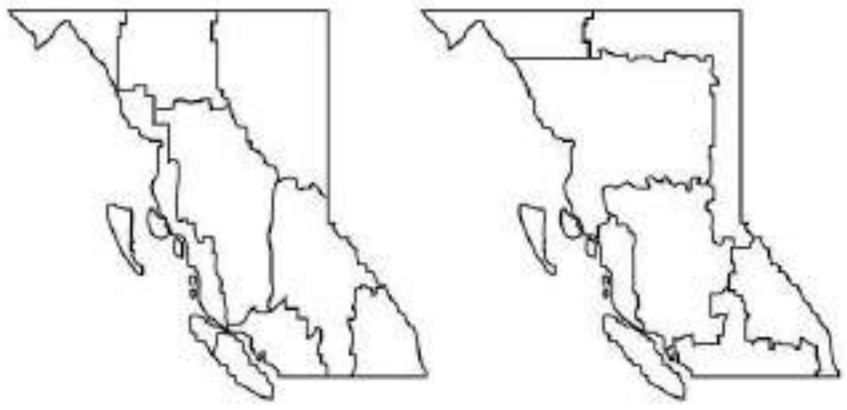


Figure 10.4 Generalized regions after different roll-up operations.

each being the result of merging a large number of small (probe) regions from Figure 10.2.

6.What is Multimedia Data Mining?

"What is a multimedia database?" A multimedia database system stores and manages a large collection of multimedia data, such as audio, video, image, graphics, speech, text,

document, and hypertext data, which contain text, text markups, and linkages. Multimedia database systems are increasingly common owing to the popular use of audio video equipment, digital cameras, CD-ROMs, and the Internet. Typical multimedia database systems include NASA's EOS (Earth Observation System), various kinds of image and audio-video databases, and Internet databases.
In this section, our study of multimedia data mining focuses on image data mining. Mining text data and mining the World Wide Web are studied in the two subsequent sections. Here we introduce multimedia data mining methods, including similarity search in multimedia data, multidimensional analysis, classification and prediction analysis, and mining associations in multimedia data.

1 **Similarity Search in Multimedia Data**

"When searching for similarities in multimedia data, can we search on either the data description or the data content?" That is correct. For similarity searching in multimedia data, we consider two main families of multimedia indexing and retrieval systems: (1) description-based retrieval systems, which build indices and perform object retrieval based on image descriptions, such as keywords, captions, size, and time of creation; and (2) content-based retrieval systems, which support retrieval based on the image content, such as color histogram, texture, pattern, image topology, and the shape of objects and their layouts and locations within the image. Description-based retrieval is labour-intensive if performed manually. If automated, the results are typically of poor quality. For example, the assignment of keywords to images can be a tricky and arbitrary task. Recent development of Web-based image clustering and classification methods has improved the quality of description-based Web image retrieval, because image surrounded text information as well as Web linkage information can be used to extract proper description and group images describing a similar theme together. Content-based retrieval uses visual features to index images and promotes object retrieval based on feature similarity, which is highly desirable in many applications.
In a content-based image retrieval system, there are often two kinds of queries: image sample-based queries and image feature specification queries. Image-sample-based queries find all of the images that are similar to the given image sample. This search compares the feature vector (or signature) extracted from the sample with the feature vectors of images that have already been extracted and indexed in the image database. Based on this comparison, images that are close to the sample image are returned. Image feature specification queries specify or sketch image features like color, texture, or shape, which are translated into a feature vector to be matched with the feature vectors of the images in the database. Content-based retrieval has wide applications, including medical diagnosis, weather prediction, TV production, Web search engines for images, and e-commerce. Some systems, such as QBIC (Query By Image Content), support both sample-based and image feature specification queries. There are also systems that support both content based and description-based retrieval.
Several approaches have been proposed and studied for similarity-based retrieval in image databases, based on image signature:

Color histogram–based signature: In this approach, the signature of an image includes color histograms based on the color composition of an image regardless of its scale or orientation. This method does not contain any information about shape, image topology, or texture. Thus, two images with similar color composition but that contain very different shapes or textures may be identified as similar, although they could be completely unrelated semantically.

Multifeature composed signature: In this approach, the signature of an image includes a composition of multiple features: color histogram, shape, image topology, and texture. The extracted image features are stored as metadata, and images are indexed based on such metadata. Often, separate distance functions can be defined for each feature and subsequently combined to derive the overall results. Multidimensional content-based search often uses one or a few probe features to search for images containing such (similar) features. It can therefore be used to search for similar images. This is the most popularly used approach in practice.

Wavelet-based signature: This approach uses the dominant wavelet coefficients of an image as its signature. Wavelets capture shape, texture, and image topology information in a single unified framework.1 This improves efficiency and reduces the need for providing multiple search primitives (unlike the second method above). However, since this method computes a single signature for an entire image, it may fail to identify images containing similar objects where the objects differ in location or size.

Wavelet-based signature with region-based granularity: In this approach, the computation and comparison of signatures are at the granularity of regions, not the entire image. This is based on the observation that similar images may contain similar regions, but a region in one image could be a translation or scaling of a matching region in the other. Therefore, a similarity measure between the query image Q and a target image T can be defined in terms of the fraction of the area of the two images covered by matching pairs of regions from Q and T. Such a region-based similarity search can find images containing similar objects, where these objects may be translated or scaled.

2 **Multidimensional Analysis of Multimedia Data**

"Can we construct a data cube for multimedia data analysis?" To facilitate the multidimensional analysis of large multimedia databases, multimedia data cubes can be designed and constructed in a manner similar to that for traditional data cubes from relational data.
A multimedia data cube can contain additional dimensions and measures for multimedia information, such as color, texture, and shape.
Let's examine a multimedia data mining system prototype called Multimedia Miner, which extends the DBMiner system by handling multimedia data. The example database tested in the Multimedia Miner system is constructed as follows. Each image contains two descriptors: a feature descriptor and a layout descriptor. The original image is not stored directly in the database; only its descriptors are stored. The description information encompasses fields like image file name, image URL, image type (e.g., gif, tiff, jpeg, mpeg, bmp, avi), a list of all known Web pages referring to the image (i.e., parent URLs), a list of keywords, and a thumbnail used by the user interface for image and video browsing. The feature descriptor is a set of vectors for each visual characteristic.
The main vectors are a color vector containing the color histogram quantized to 512 colors (8X8X8 for R_G_B), an MFC(Most Frequent Color) vector, and an MFO(Most Frequent Orientation) vector. The MFC and MFO contain five color centroids and five edge orientation centroids for the five most frequent colors and five most frequent orientations,

respectively. The edge orientations used are 0 deg, 22:5deg, 45deg, 67:5deg, 90deg, and so on. The layout descriptor contains a color layout vector and an edge layout vector. Regardless of their original size, all images are assigned an 8_8 grid. The most frequent color for each of the 64 cells is stored in the color layout vector, and the number of edges for each orientation in each of the cells is stored in the edge layout vector. Other sizes of grids, like 4x4, 2x2, and 1x1, can easily be derived.

3 **Classification and Prediction Analysis of Multimedia Data**

Classification and predictive modelling have been used for mining multimedia data, especially in scientific research, such as astronomy, seismology, and geoscientific research. In general, all of the classification methods discussed in Chapter 6 can be used in image analysis and pattern recognition. Moreover, in-depth statistical pattern analysis methods are popular for distinguishing subtle features and building high-quality models.
Example 10.8 Classification and prediction analysis of astronomy data. Taking sky images that have been carefully classified by astronomers as the training set, we can construct models for the recognition of galaxies, stars, and other stellar objects, based on properties like magnitudes, areas, intensity, image moments, and orientation. A large number of sky images taken by telescopes or space probes can then be tested against the constructed models in order to identify new celestial bodies. Similar studies have successfully been performed to identify volcanoes on Venus.

The popular use of the World Wide Web has made the Web a rich and gigantic repository Of multimedia data. The Web not only collects a tremendous number of photos, pictures, albums, and video images in the form of on-line multimedia libraries, but also has numerous photos, pictures, animations, and other multimedia forms on almost every Web page. Such pictures and photos, surrounded by text descriptions, located at the different blocks of Web pages, or embedded inside news or text articles, may serve rather different purposes, such as forming an inseparable component of the content, serving as an advertisement, or suggesting an alternative topic. Furthermore, these Web pages are linked with other Web pages in a complicated way. Such text, image location, and Web linkage information, if used properly, may help understand the contents of the text or assist classification and clustering of images on the Web. Data mining by making good use of relative locations and linkages among images, text, blocks within a page, and page links on the Web becomes an important direction in Web data analysis.

7.What is Mining Associations in Multimedia Data?

"What kinds of associations can be mined inmultimedia data?" Association rules involving multimedia objects can be mined in image and video databases. At least three categories can be observed:

Associations between image content and nonimage content features:
A rule like "If at least 50% of the upper part of the picture is blue, then it is likely to represent sky" belongs to this category since it links the image content to the keyword sky.

Associations among image contents that are not related to spatial relationships: A rule like "If a picture contains two blue squares, then it is likely to contain one red circle As well" belongs to this category since the associations are all regarding image contents.

Associations among image contents related to spatial relationships: A rule like "If a red triangle is between two yellow squares, then it is likely a big oval-shaped object is underneath" belongs to this category since it associates objects in the image with spatial relationships.

To mine associations among multimedia objects, we can treat each image as a transaction and find frequently occurring patterns among different images.
"What are the differences between mining association rules in multimedia databases versus in transaction databases?" There are some subtle differences. First, an image may contain multiple objects, each with many features such as color, shape, texture, keyword, and spatial location, so there could be many possible associations. In many cases, a feature may be considered as the same in two images at a certain level of resolution, but different at a finer resolution level. Therefore, it is essential to promote a progressive resolution refinement approach. That is, we can first mine frequently occurring patterns at a relatively rough resolution level, and then focus only on those that have passed the minimum support threshold when mining at a finer resolution level. This is because the patterns that are not frequent at a rough level cannot be frequent at finer resolution levels. Such a multi resolution mining strategy substantially reduces the overall data mining cost without loss of the quality and completeness of data mining results. This leads to an efficient methodology for mining frequent item sets and associations in large multimedia databases.
Second, because a picture containing multiple recurrent objects is an important feature in image analysis, recurrence of the same objects should not be ignored in association analysis. For example, a picture containing two golden circles is treated quite differently from that containing only one. This is quite different from that in a transaction database, where the fact that a person buys one gallon of milk or two may often be treated the same as "buys milk." Therefore, the definition of multimedia association and its measurements, such as support and confidence, should be adjusted accordingly.
Third, there often exist important spatial relationships among multimedia objects, such as above, beneath, between, nearby, left-of, and so on. These features are very useful for exploring object associations and correlations. Spatial relationships together with other content-based multimedia features, such as color, shape, texture, and keywords, may form interesting associations. Thus, spatial data mining methods and properties of topological spatial relationships become important for multimedia mining.

8 What is Audio and Video Data Mining?

Besides still images, an incommensurable amount of audiovisual information is becoming available in digital form, in digital archives, on the World Wide Web, in broadcast data streams, and in personal and professional databases. This amount is rapidly growing.
There are great demands for effective content-based retrieval and data mining methods for audio and video data. Typical examples include searching for and multimedia editing of particular video clips in a TV studio, detecting suspicious persons or scenes in surveillance videos, searching for particular events in a personal multimedia repository such as MyLifeBits, discovering patterns and outliers in weather radar recordings, and finding a particular melody or tune in your MP3 audio album.
To facilitate the recording, search, and analysis of audio and video information from multimedia data, industry and standardization committees have made great strides toward developing a set of standards for multimedia information description and compression.
For example, MPEG-k (developed by MPEG: Moving Picture Experts Group)

and JPEG are typical video compression schemes. The most recently released MPEG-7, formally named "Multimedia Content Description Interface," is a standard for describing the multimedia content data. It supports some degree of interpretation of the information meaning, which can be passed onto, or accessed by, a device or a computer.
MPEG-7 is not aimed at any one application in particular; rather, the elements that MPEG-7 standardizes support as broad a range of applications as possible. The audiovisual data description in MPEG-7 includes still pictures, video, graphics, audio, speech, three-dimensional models, and information about how these data elements are combined in the multimedia presentation.
The MPEG committee standardizes the following elements in MPEG-7: (1) a set of descriptors, where each descriptor defines the syntax and semantics of a feature, such as color, shape, texture, image topology, motion, or title; (2) a set of descriptor schemes, where each scheme specifies the structure and semantics of the relationships between its components (descriptors or description schemes); (3) a set of coding schemes for the descriptors, and (4) a description definition language (DDL) to specify schemes and descriptors. Such standardization greatly facilitates content-based video retrieval and video data mining.

It is unrealistic to treat a video clip as a long sequence of individual still pictures and analyze each picture since there are too many pictures, and most adjacent images could be rather similar. In order to capture the story or event structure of a video, it is better to treat each video clip as a collection of actions and events in time and first temporarily segment them into video shots. A shot is a group of frames or pictures where the video content from one frame to the adjacent ones does not change abruptly. Moreover, the most representative frame in a video shot is considered the key frame of the shot. Each key frame can be analyzed using the image feature extraction and analysis methods studied above in the content-based image retrieval. The sequence of key frames will then be used to define the sequence of the events happening in the video clip. Thus the detection of shots and the extraction of key frames from video clips become the essential tasks in video processing and mining.
Video data mining is still in its infancy. There are still a lot of research issues to be solved before it becomes general practice. Similarity-based preprocessing, compression, indexing and retrieval, information extraction, redundancy removal, frequent pattern discovery, classification, clustering, and trend and outlier detection are important data mining tasks in this domain.

9.What is Text Mining?

Most previous studies of data mining have focused on structured data, such as relational, transactional, and data warehouse data. However, in reality, a substantial portion of the available information is stored in text databases (or document databases), which consist of large collections of documents from various sources, such as news articles, research papers, books, digital libraries, e-mail messages, and Web pages. Text databases are rapidly growing due to the increasing amount of information available in electronic form, such as electronic publications, various kinds of electronic documents, e-mail, and the World Wide Web (which can also be viewed as a huge, interconnected, dynamic text database). Nowadays most of the information in government, industry, business, and other institutions are stored electronically, in the form of text databases.

Data stored in most text databases are semi structured data in that they are neither completely unstructured nor completely structured. For example, a document may contain a few structured fields, such as title, authors, publication date, category, and so on, but also contain some largely unstructured text components, such as abstract and contents. There have been a great deal of studies on the modelling and implementation of semi structured data in recent database research. Moreover, information retrieval techniques, such as text indexing methods, have been developed to handle unstructured documents.

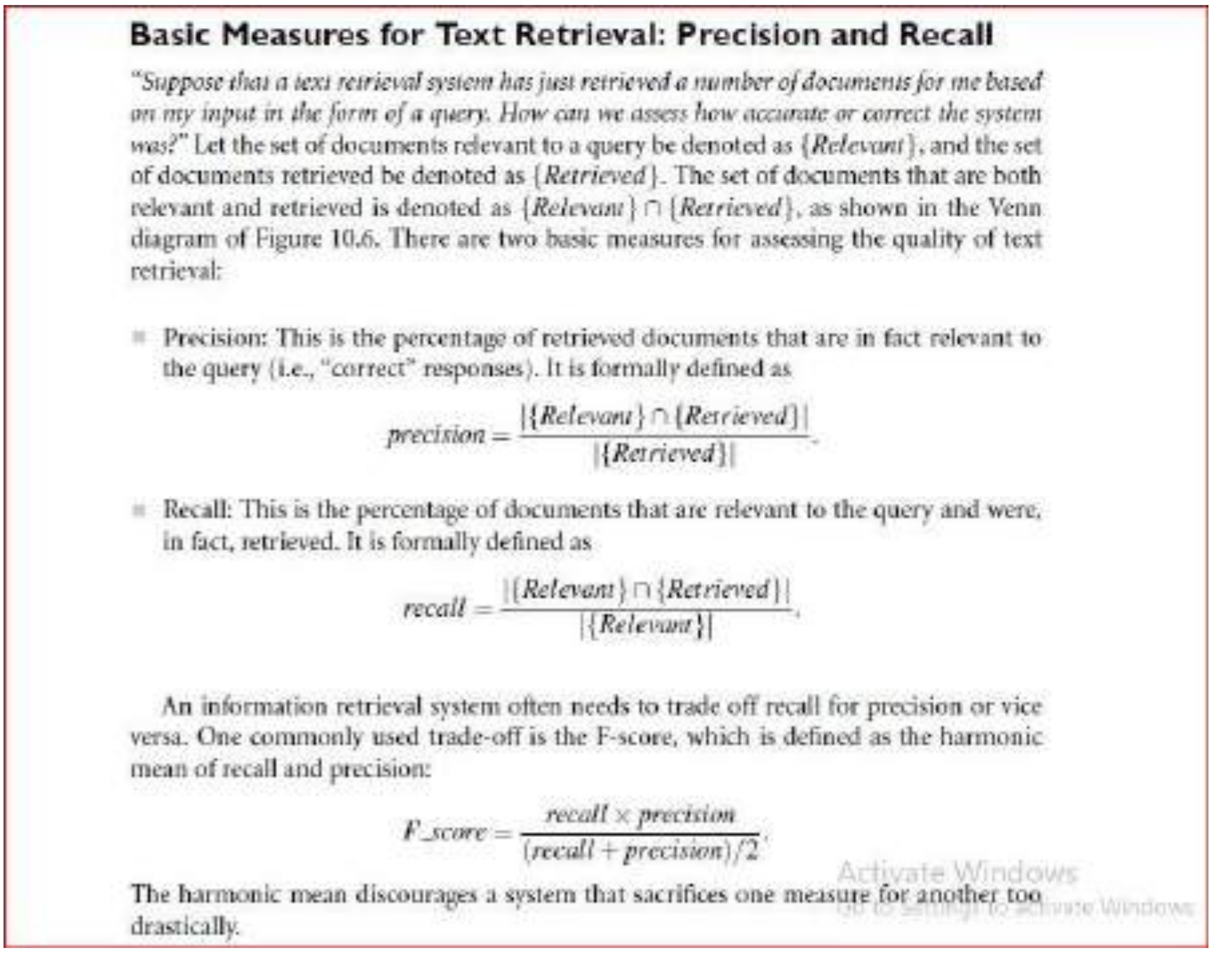
Traditional information retrieval techniques become inadequate for the increasingly vast amounts of text data. Typically, only a small fraction of the many available documents will be relevant to a given individual user. Without knowing what could be in the documents, it is difficult to formulate effective queries for analyzing and extracting useful information from the data. Users need tools to compare different documents, rank the importance and relevance of the documents, or find patterns and trends across multiple documents. Thus, text mining has become an increasingly popular and essential theme in data mining.

1 Text Data Analysis and Information Retrieval

"What is information retrieval?" Information retrieval (IR) is a field that has been developing in parallel with database systems for many years. Unlike the field of database systems, which has focused on query and transaction processing of structured data, information retrieval is concerned with the organization and retrieval of information from a large number of text-based documents. Since information retrieval and database systems each handle different kinds of data, some database system problems are usually not present in information retrieval systems, such as concurrency control, recovery, transaction management, and update. Also, some common information retrieval problems are usually not encountered in traditional database systems, such as unstructured documents, approximate search based on keywords, and the notion of relevance.

Due to the abundance of text information, information retrieval has found many applications. There exist many information retrieval systems, such as on-line library catalog systems, on-line document management systems, and the more recently developed Web search engines.

A typical information retrieval problem is to locate relevant documents in a document collection based on a user's query, which is often some keywords describing an information need, although it could also be an example relevant document. In such a search problem, a user takes the initiative to "pull" the relevant information out from the collection; this is most appropriate when a user has some ad hoc (i.e., short-term) information need, such as finding information to buy a used car. When a user has a long-term information need (e.g., a researcher's interests), a retrieval system may also take the initiative to "push" any newly arrived information item to a user if the item is judged as being relevant to the user's information need. Such an information access process is called information filtering, and the corresponding systems are often called filtering systems or recommender systems. From a technical viewpoint, however, search and filtering share many common techniques. Below we briefly discuss the major techniques in information retrieval with a focus on search techniques.



Basic Measures for Text Retrieval: Precision and Recall

"Suppose that a text retrieval system has just retrieved a number of documents for me based on my input in the form of a query. How can we assess how accurate or correct the system was?" Let the set of documents relevant to a query be denoted as {*Relevant*}, and the set of documents retrieved be denoted as {*Retrieved*}. The set of documents that are both relevant and retrieved is denoted as {*Relevant*} ∩ {*Retrieved*}, as shown in the Venn diagram of Figure 10.6. There are two basic measures for assessing the quality of text retrieval:

- Precision: This is the percentage of retrieved documents that are in fact relevant to the query (i.e., "correct" responses). It is formally defined as

$$precision = \frac{|\{Relevant\} \cap \{Retrieved\}|}{|\{Retrieved\}|}.$$

- Recall: This is the percentage of documents that are relevant to the query and were, in fact, retrieved. It is formally defined as

$$recall = \frac{|\{Relevant\} \cap \{Retrieved\}|}{|\{Relevant\}|}.$$

An information retrieval system often needs to trade off recall for precision or vice versa. One commonly used trade-off is the F-score, which is defined as the harmonic mean of recall and precision:

$$F\_score = \frac{recall \times precision}{(recall + precision)/2}.$$

The harmonic mean discourages a system that sacrifices one measure for another too drastically.

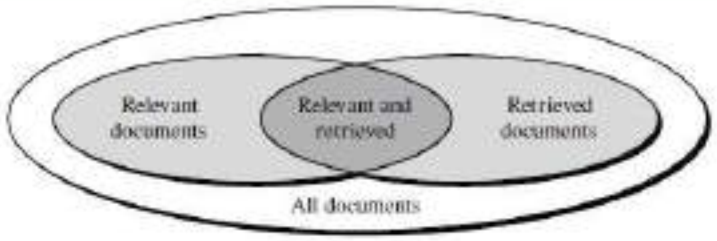


Figure 10.6 Relationship between the set of relevant documents and the set of retrieved documents.

Precision, recall, and F-score are the basic measures of a retrieved set of documents. These three measures are not directly useful for comparing two ranked lists of documents because they are not sensitive to the internal ranking of the documents in a retrieved set. In order to measure the quality of a ranked list of documents, it is common to compute an average of precisions at all the ranks where a new relevant document is returned. It is also common to plot a graph of precisions at many different levels of recall; a higher curve represents a better-quality information retrieval system.

Text Retrieval Methods

“What methods are there for information retrieval?” Broadly speaking, retrieval methods fall into two categories: They generally either view the retrieval problem as a document selection problem or as a document ranking problem.
In document selection methods, the query is regarded as specifying constraints for selecting relevant documents. A typical method of this category is the Boolean retrieval model, in which a document is represented by a set of keywords and a user provides a Boolean expression of keywords, such as “car and repair shops,” “tea or coffee,” or “database systems but not Oracle.” The retrieval systemwould take such a Boolean query and return documents that satisfy the Boolean expression. Because of the difficulty in prescribing a user’s information need exactly with a Boolean query, the Boolean retrieval method generally only works well when the user knows a lot about the document collection and can formulate a good query in this way.
Document ranking methods use the query to rank all documents in the order of relevance. For ordinary users and exploratory queries, these methods are more appropriate than document selection methods. Most modern information retrieval systems present a ranked list of documents in response to a user’s keyword query. There are many different ranking methods based on a large spectrum of mathematical foundations, including algebra, logic, probability, and statistics. The common intuition behind all of these methods is that we may match the keywords in a query with those in the documents and score each document based on how well it matches the query. The goal is to approximate the degree of relevance of a document with a score computed based on information such as the frequency of words in the document and the whole collection. Notice that it is inherently difficult to provide a precise measure of the degree of relevance between a set of keywords. For example, it is difficult to quantify the distance between data mining and data analysis. Comprehensive empirical evaluation is thus essential for validating any retrieval method.

The vector space model for text retrieval:
The basic idea of the vector space model is the following: We represent a document and a query both as vectors in a high-dimensional space corresponding to all the keywords and use an appropriate similarity measure to compute the similarity between the query vector and the document vector. The similarity values can then be used for ranking documents.
“How do we tokenize text?” The first step in most retrieval systems is to identify keywords for representing documents, a preprocessing step often called tokenization. To avoid indexing useless words, a text retrieval system often associates a stop list with a set of documents. A stop list is a set of words that are deemed “irrelevant.” For example, a, the, of, for, with, and so on are stop words, even though they may appear frequently. Stop lists may vary per document set. For example, database systems could be an important keyword in a newspaper.However, it may be considered as a stop word in a set of research papers presented in a database systems conference.
A group of different words may share the same word stem. A text retrieval system needs to identify groups of words where the words in a group are small syntactic variants of one another and collect only the common word stem per group. For example, the group of words drug, drugged, and drugs, share a common word stem, drug, and can be viewed as different occurrences of the same word.

"How can we model a document to facilitate information retrieval?" Starting with a set of d documents and a set of t terms, we can model each document as a vector v in the t dimensional space $\Re^t$, which is why this method is called the **vector-space model**. Let the **term frequency** be the number of occurrences of term t in the document d, that is, $freq(d,t)$. The (weighted) term-frequency matrix $TF(d,t)$ measures the association of a term t with respect to the given document d: it is generally defined as 0 if the document does not contain the term, and nonzero otherwise. There are many ways to define the term-weighting for the nonzero entries in such a vector. For example, we can simply set $TF(d,t) = 1$ if the term t occurs in the document d, or use the term frequency $freq(d,t)$, or the relative term frequency, that is, the term frequency versus the total number of occurrences of all the terms in the document. There are also other ways to normalize the term frequency. For example, the Cornell SMART system uses the following formula to compute the (normalized) term frequency:

$$TF(d,t) = \begin{cases} 0 & \textit{if } freq(d,t) = 0 \\ 1 + \log(1 + \log(freq(d,t))) & \textit{otherwise.} \end{cases} \quad (10.3)$$

Besides the term frequency measure, there is another important measure, called **inverse document frequency (IDF)**, that represents the scaling factor, or the importance, of a term t. If a term t occurs in many documents, its importance will be scaled down due to its reduced discriminative power. For example, the term *database systems* may likely be less important if it occurs in many research papers in a database system conference. According to the same Cornell SMART system, $IDF(t)$ is defined by the following formula:

$$IDF(t) = \log \frac{1 + |d|}{|d_t|}, \quad (10.4)$$

where d is the document collection, and d_t is the set of documents containing term t. If $|d_t| \ll |d|$, the term t will have a large IDF scaling factor and vice versa.

In a complete vector-space model, TF and IDF are combined together, which forms the TF-IDF measure:

$$TF\text{-}IDF(d,t) = TF(d,t) \times IDF(t). \quad (10.5)$$

Let us examine how to compute similarity among a set of documents based on the notions of term frequency and inverse document frequency.

Example 10.9 Term frequency and inverse document frequency. Table 10.5 shows a term frequency matrix where each row represents a document vector, each column represents a term, and each entry registers $freq(d_i,t_j)$, the number of occurrences of term t_j in document d_i. Based on this table we can calculate the TF-IDF value of a term in a document. For example, for t_6 in d_4, we have

$$TF(d_4,t_6) = 1 + \log(1 + \log(15)) = 1.3377$$

$$IDF(t_6) = \log \frac{1+5}{3} = 0.301.$$

■

Therefore,

$$TF\text{-}IDF(d_4,t_6) = 1.3377 \times 0.301 = 0.403$$

"How can we determine if two documents are similar?" Since similar documents are expected to have similar relative term frequencies, we can measure the similarity among a set of documents or between a document and a query (often defined as a set of keywords), based on similar relative term occurrences in the frequency table. Many metrics have been proposed for measuring document similarity based on relative term occurrences or document vectors. A representative metric is the **cosine measure**, defined as follows. Let v_1 and v_2 be two document vectors. Their cosine similarity is defined as

Let v_1 and v_2 be two document vectors. Their cosine similarity is defined as

$$sim(v_1, v_2) = \frac{v_1 \cdot v_2}{|v_1||v_2|}, \tag{10.6}$$

where the inner product $v_1 \cdot v_2$ is the standard vector dot product, defined as $\Sigma_{i=1}^{t} v_{1i} v_{2i}$, and the norm $|v_1|$ in the denominator is defined as $|v_1| = \sqrt{v_1 \cdot v_1}$.

Table 10.5 A term frequency matrix showing the frequency of terms per document.

document/term	t_1	t_2	t_3	t_4	t_5	t_6	t_7
d_1	0	4	10	8	0	5	0
d_2	5	19	7	16	0	0	32
d_3	15	0	0	4	9	0	17
d_4	22	3	12	0	5	15	0
d_5	0	7	0	9	2	4	12

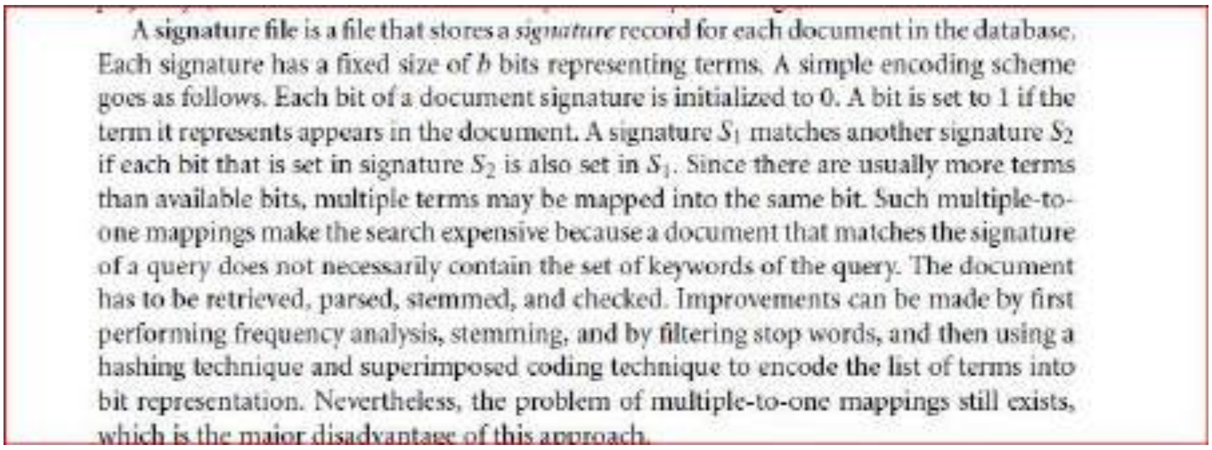
Text Indexing Techniques

There are several popular text retrieval indexing techniques, including *inverted indices* and *signature files*.

An inverted index is an index structure that maintains two hash indexed or B+-tree indexed tables: *document_table* and *term_table*, where

- *document_table* consists of a set of document records, each containing two fields: *doc_id* and *posting_list*, where *posting_list* is a list of terms (or pointers to terms) that occur in the document, sorted according to some relevance measure.
- *term_table* consists of a set of term records, each containing two fields: *term_id* and *posting_list*, where *posting_list* specifies a list of document identifiers in which the term appears.

With such organization, it is easy to answer queries like "*Find all of the documents associated with a given set of terms*," or "*Find all of the terms associated with a given set of documents*." For example, to find all of the documents associated with a set of terms, we can first find a list of document identifiers in *term_table* for each term, and then intersect them to obtain the set of relevant documents. Inverted indices are widely used in industry. They are easy to implement. The *posting_lists* could be rather long, making the storage requirement quite large. They are easy to implement, but are not satisfactory at handling *synonymy* (where two very different words can have the same meaning) and *polysemy* (where an individual word may have many meanings).

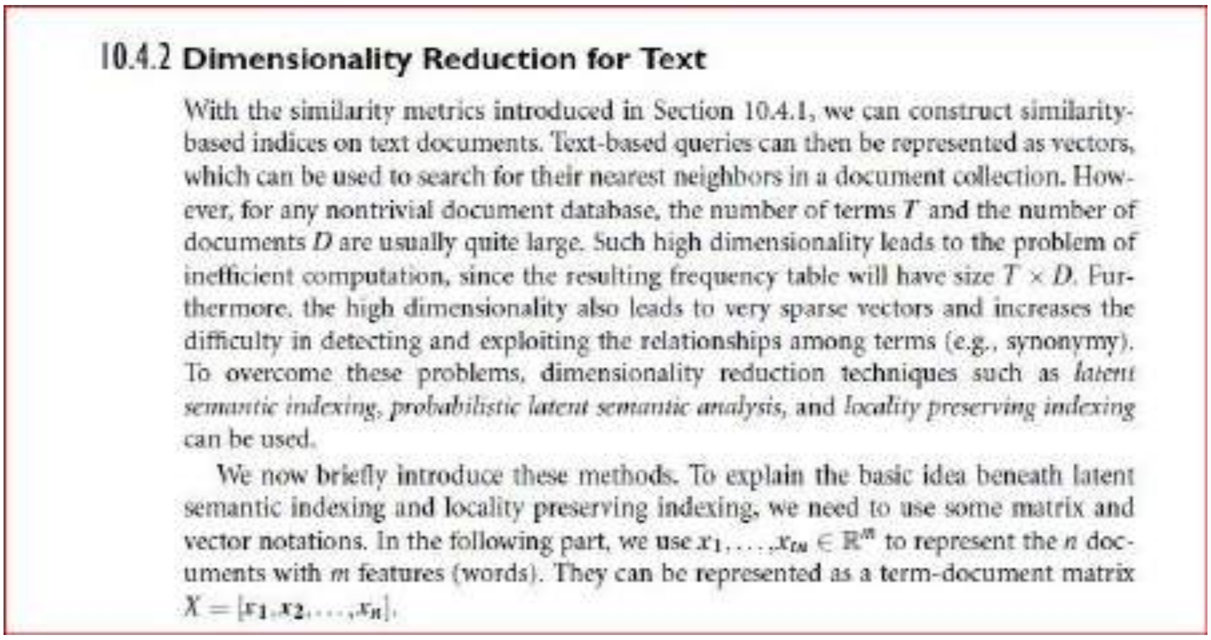


A signature file is a file that stores a *signature* record for each document in the database. Each signature has a fixed size of b bits representing terms. A simple encoding scheme goes as follows. Each bit of a document signature is initialized to 0. A bit is set to 1 if the term it represents appears in the document. A signature S_1 matches another signature S_2 if each bit that is set in signature S_2 is also set in S_1. Since there are usually more terms than available bits, multiple terms may be mapped into the same bit. Such multiple-to-one mappings make the search expensive because a document that matches the signature of a query does not necessarily contain the set of keywords of the query. The document has to be retrieved, parsed, stemmed, and checked. Improvements can be made by first performing frequency analysis, stemming, and by filtering stop words, and then using a hashing technique and superimposed coding technique to encode the list of terms into bit representation. Nevertheless, the problem of multiple-to-one mappings still exists, which is the major disadvantage of this approach.

Query Processing Techniques

Once an inverted index is created for a document collection, a retrieval systemcan answer a keyword query quickly by looking up which documents contain the query keywords. Specifically, we will maintain a score accumulator for each document and update these accumulators as we go through each query term. For each query term, we will fetch all of the documents that match the term and increase their scores.

When examples of relevant documents are available, the system can learn from such examples to improve retrieval performance. This is called *relevance feedback* and has proven to be effective in improving retrieval performance. When we do not have such relevant examples, a system can *assume* the top few retrieved documents in some initial retrieval results to be relevant and extract more related keywords to expand a query. Such feedback is called *pseudo-feedback* or *blind feedback* and is essentially a process of mining useful keywords from the top retrieved documents. Pseudo-feedback also often leads to improved retrieval performance.
One major limitation of many existing retrieval methods is that they are based on exact keyword matching. However, due to the complexity of natural languages, keyword based retrieval can encounter two major difficulties. The first is the synonymy problem: two words with identical or similar meanings may have very different surface forms. For example, a user’s query may use the word “automobile,” but a relevant document may use “vehicle” instead of “automobile.” The second is the polysemy problem: the same keyword, such as *mining*, or *Java*, may mean different things in different contexts.
We now discuss some advanced techniques that can help solve these problems as well as reduce the index size.



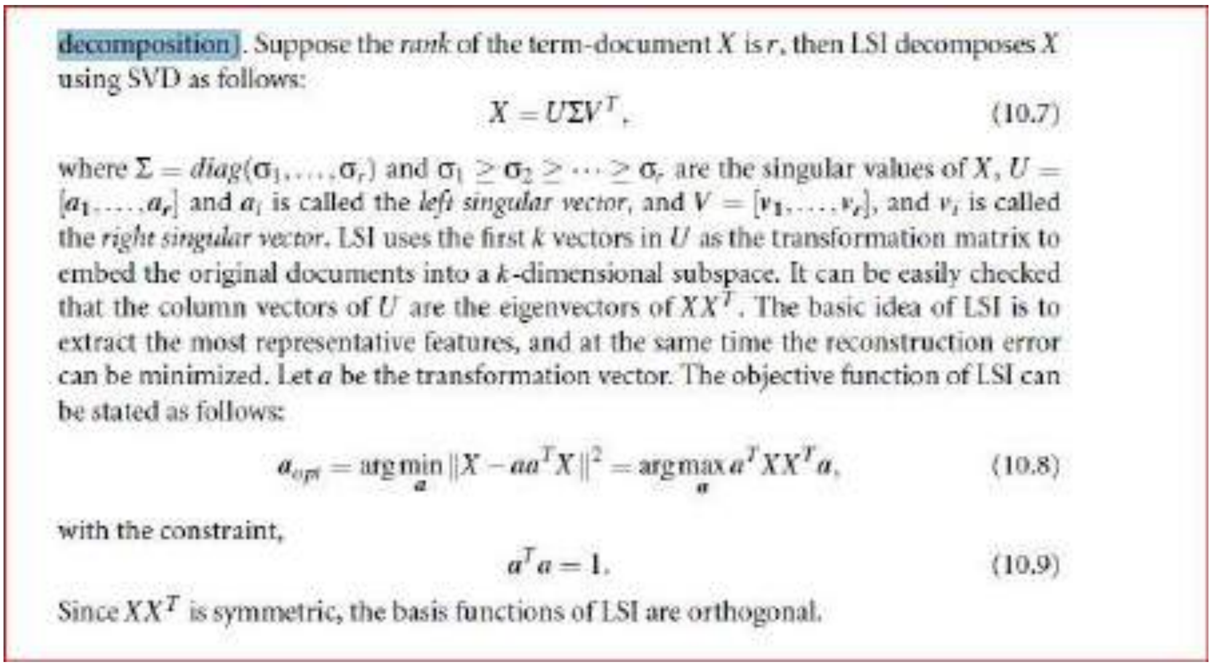
10.4.2 Dimensionality Reduction for Text

With the similarity metrics introduced in Section 10.4.1, we can construct similarity-based indices on text documents. Text-based queries can then be represented as vectors, which can be used to search for their nearest neighbors in a document collection. However, for any nontrivial document database, the number of terms T and the number of documents D are usually quite large. Such high dimensionality leads to the problem of inefficient computation, since the resulting frequency table will have size $T \times D$. Furthermore, the high dimensionality also leads to very sparse vectors and increases the difficulty in detecting and exploiting the relationships among terms (e.g., synonymy). To overcome these problems, dimensionality reduction techniques such as *latent semantic indexing*, *probabilistic latent semantic analysis*, and *locality preserving indexing* can be used.

We now briefly introduce these methods. To explain the basic idea beneath latent semantic indexing and locality preserving indexing, we need to use some matrix and vector notations. In the following part, we use $x_1, \ldots, x_n \in \mathbb{R}^m$ to represent the n documents with m features (words). They can be represented as a term-document matrix $X = [x_1, x_2, \ldots, x_n]$.

Latent Semantic Indexing

Latent semantic indexing (LSI) is one of the most popular algorithms for document dimensionality reduction. It is fundamentally based on SVD (singular value decomposition).



decomposition). Suppose the *rank* of the term-document X is r, then LSI decomposes X using SVD as follows:

$$X = U\Sigma V^T, \tag{10.7}$$

where $\Sigma = diag(\sigma_1, \ldots, \sigma_r)$ and $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r$ are the singular values of X, $U = [a_1, \ldots, a_r]$ and a_i is called the *left singular vector*, and $V = [v_1, \ldots, v_r]$, and v_i is called the *right singular vector*. LSI uses the first k vectors in U as the transformation matrix to embed the original documents into a k-dimensional subspace. It can be easily checked that the column vectors of U are the eigenvectors of XX^T. The basic idea of LSI is to extract the most representative features, and at the same time the reconstruction error can be minimized. Let a be the transformation vector. The objective function of LSI can be stated as follows:

$$a_{opt} = \arg\min_a \|X - aa^T X\|^2 = \arg\max_a a^T XX^T a, \tag{10.8}$$

with the constraint,

$$a^T a = 1. \tag{10.9}$$

Since XX^T is symmetric, the basis functions of LSI are orthogonal.

10.4.3 Text Mining Approaches

There are many approaches to text mining, which can be classified from different perspectives, based on the inputs taken in the text mining system and the data mining tasks to be performed. In general, the major approaches, based on the kinds of data they take as input, are: (1) the **keyword-based approach**, where the input is a set of keywords or terms in the documents, (2) the **tagging approach**, where the input is a set of tags, and (3) the **information-extraction** approach, which inputs semantic information, such as events, facts, or entities uncovered by information extraction. A simple keyword-based approach may only discover relationships at a relatively shallow level, such as rediscovery of compound nouns (e.g., "database" and "systems") or co-occurring patterns with less significance (e.g., "terrorist" and "explosion"). It may not bring much deep understanding to the text. The tagging approach may rely on tags obtained by *manual tagging* (which is costly and is unfeasible for large collections of documents) or by some *automated categorization algorithm* (which may process a relatively small set of tags and require defining the categories beforehand). The information-extraction approach is more advanced and may lead to the discovery of some deep knowledge, but it requires semantic analysis of text by natural language understanding and machine learning methods. This is a challenging knowledge discovery task.

Various text mining tasks can be performed on the extracted keywords, tags, or semantic information. These include document clustering, classification, information extraction, association analysis, and trend analysis. We examine a few such tasks in the following discussion.

Keyword-Based Association Analysis

"*What is keyword-based association analysis?*" Such analysis collects sets of keywords or terms that occur frequently together and then finds the association or correlation relationships among them.

Like most of the analyses in text databases, association analysis first preprocesses the text data by parsing, stemming, removing stop words, and so on, and then evokes association mining algorithms. In a document database, each document can be viewed as a transaction, while a set of keywords in the document can be considered as a set of items in the transaction. That is, the database is in the format

$$\{document\_id, a\_set\_of\_keywords\}.$$

The problem of keyword association mining in document databases is thereby mapped to item association mining in transaction databases, where many interesting methods have been developed, as described in Chapter 5.

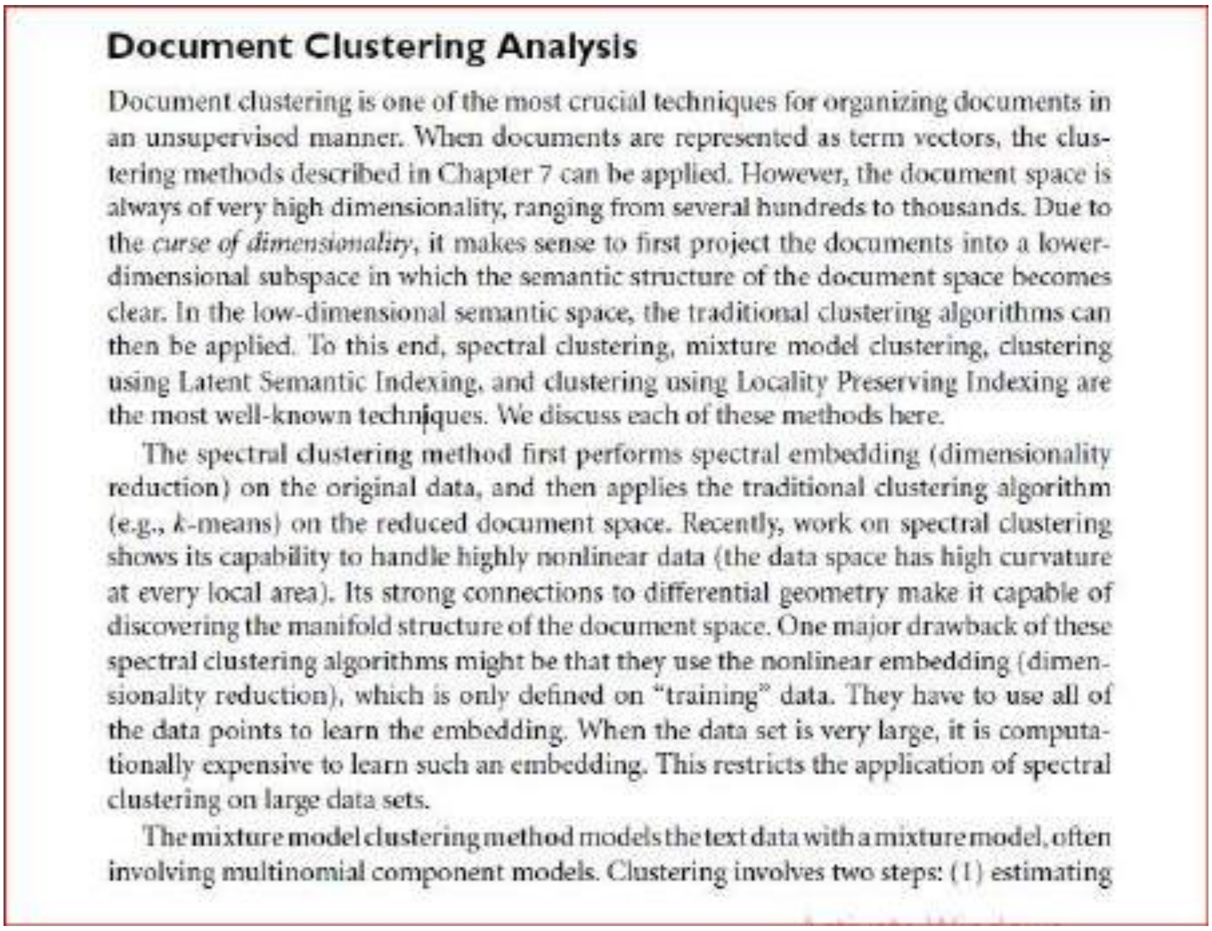
Notice that a set of frequently occurring consecutive or closely located keywords may form a *term* or a *phrase*. The association mining process can help detect **compound associations**, that is, domain-dependent terms or phrases, such as [Stanford, University] or [U.S., President, George W. Bush], or **noncompound associations**, such as [dollars, shares, exchange, total, commission, stake, securities]. Mining based on these associations is referred to as "*term-level* association mining" (as opposed to mining on individual words). Term recognition and term-level association mining enjoy two advantages in text analysis: (1) terms and phrases are automatically tagged so there is no need for human effort in tagging documents; and (2) the number of meaningless results is greatly reduced, as is the execution time of the mining algorithms.

With such term and phrase recognition, term-level mining can be evoked to find associations among a set of detected terms and keywords. Some users may like to find associations between pairs of keywords or terms from a given set of keywords or phrases, whereas others may wish to find the maximal set of terms occurring together. Therefore, based on user mining requirements, standard association mining or max-pattern mining algorithms may be evoked.

Document Classification Analysis

Automated document classification is an important text mining task because, with the existence of a tremendous number of on-line documents, it is tedious yet essential to be able to automatically organize such documents into classes to facilitate document retrieval and subsequent analysis. Document classification has been used in automated topic tagging (i.e., assigning labels to documents), topic directory construction, identification of the document writing styles (which may help narrow down the possible authors of anonymous documents), and classifying the purposes of hyperlinks associated with a set of documents.

"How can automated document classification be performed?" A general procedure is as follows: First, a set of preclassified documents is taken as the training set. The training set is then analyzed in order to derive a classification scheme. Such a classification scheme often needs to be refined with a testing process. The so-derived classification scheme can be used for classification of other on-line documents.



Document Clustering Analysis

Document clustering is one of the most crucial techniques for organizing documents in an unsupervised manner. When documents are represented as term vectors, the clustering methods described in Chapter 7 can be applied. However, the document space is always of very high dimensionality, ranging from several hundreds to thousands. Due to the *curse of dimensionality*, it makes sense to first project the documents into a lower-dimensional subspace in which the semantic structure of the document space becomes clear. In the low-dimensional semantic space, the traditional clustering algorithms can then be applied. To this end, spectral clustering, mixture model clustering, clustering using Latent Semantic Indexing, and clustering using Locality Preserving Indexing are the most well-known techniques. We discuss each of these methods here.

The spectral clustering method first performs spectral embedding (dimensionality reduction) on the original data, and then applies the traditional clustering algorithm (e.g., *k*-means) on the reduced document space. Recently, work on spectral clustering shows its capability to handle highly nonlinear data (the data space has high curvature at every local area). Its strong connections to differential geometry make it capable of discovering the manifold structure of the document space. One major drawback of these spectral clustering algorithms might be that they use the nonlinear embedding (dimensionality reduction), which is only defined on "training" data. They have to use all of the data points to learn the embedding. When the data set is very large, it is computationally expensive to learn such an embedding. This restricts the application of spectral clustering on large data sets.

The mixture model clustering method models the text data with a mixture model, often involving multinomial component models. Clustering involves two steps: (1) estimating

10.What do you understand from Mining the World Wide Web

The World Wide Web serves as a huge, widely distributed, global information service center for news, advertisements, consumer information, financial management, education, government, e-commerce, and many other information services. The Web also contains a rich and dynamic collection of hyperlink information and Web page access and usage information, providing rich sources for data mining. However, based on the following observations, the Web also poses great challenges for effective resource and knowledge discovery.

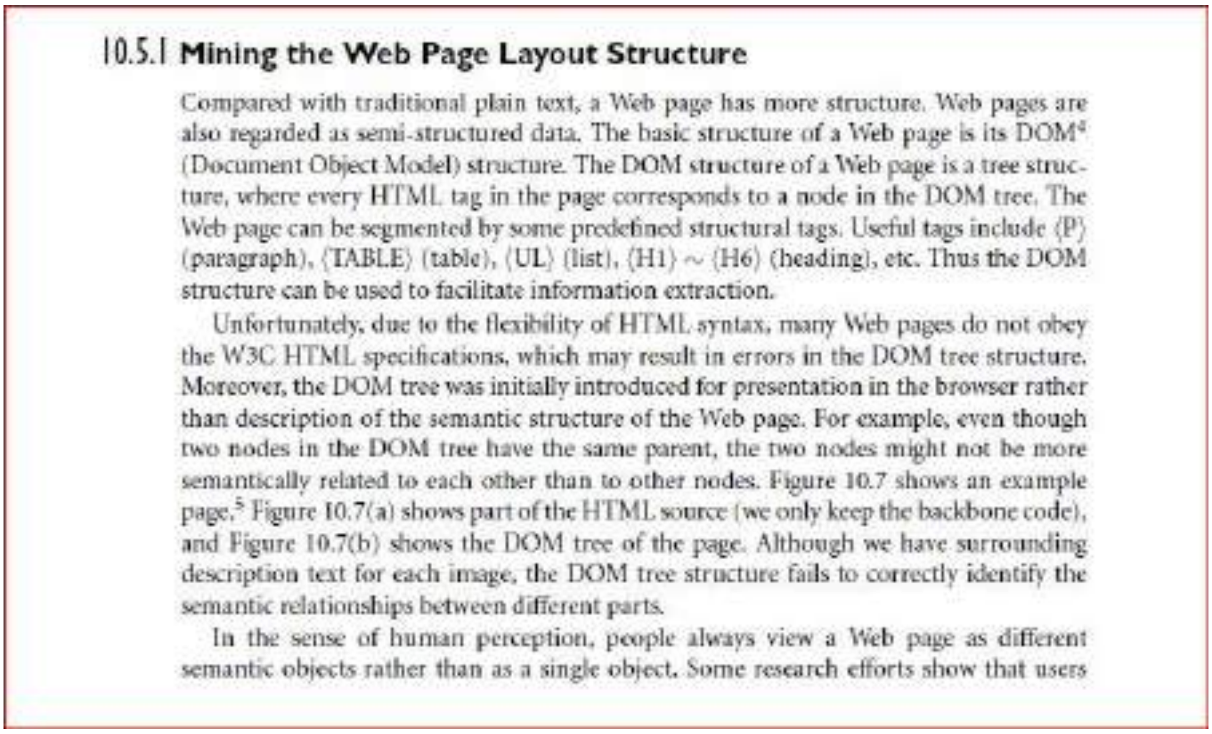
The Web seems to be too huge for effective data warehousing and data mining. The size of the Web is in the order of hundreds of terabytes and is still growing rapidly. Many organizations and societies place most of their public-accessible information on the Web. It is barely possible to set up a data warehouse to replicate, store, or integrate all of the data on the Web.

The complexity of Web pages is far greater than that of any traditional text document collection. Web pages lack a unifying structure. They contain far more authoring style and content variations than any set of books or other traditional text-based documents. The Web is considered a huge digital library; however, the tremendous number of documents in this library are not arranged according to any particular sorted order. There is no index by category, nor by title, author, cover page, table of contents, and so on. It can be very challenging to search for the information you desire in such a library!

The Web is a highly dynamic information source. Not only does the Web grow rapidly, but its information is also constantly updated. News, stock markets, weather, sports, shopping, company advertisements, and numerous other Web pages are updated regularly on the Web. Linkage information and access records are also updated frequently.

The Web serves a broad diversity of user communities. The Internet currently connects more than 100 million workstations, and its user community is still rapidly expanding. Users may have very different backgrounds, interests, and usage purposes. Most users may not have good knowledge of the structure of the information network and may not be aware of the heavy cost of a particular search. They can easily get lost by groping in the "darkness" of the network, or become bored by taking many access "hops" and waiting impatiently for a piece of information.

Only a small portion of the information on the Web is truly relevant or useful. It is said that 99% of the Web information is useless to 99% of Web users. Although this may not seem obvious, it is true that a particular person is generally interested in only a tiny portion of the Web, while the rest of the Web contains information that is uninteresting to the user and may swamp desired search results. How can the portion of the Web that is truly relevant to your interest be determined? How can we find high quality Web pages on a specified topic?
These challenges have promoted research into efficient and effective discovery and use of resources on the Internet.



10.5.1 Mining the Web Page Layout Structure

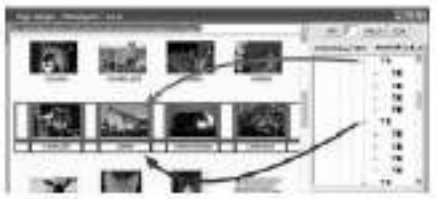
Compared with traditional plain text, a Web page has more structure. Web pages are also regarded as semi-structured data. The basic structure of a Web page is its DOM[4] (Document Object Model) structure. The DOM structure of a Web page is a tree structure, where every HTML tag in the page corresponds to a node in the DOM tree. The Web page can be segmented by some predefined structural tags. Useful tags include ⟨P⟩ (paragraph), ⟨TABLE⟩ (table), ⟨UL⟩ (list), ⟨H1⟩ ~ ⟨H6⟩ (heading), etc. Thus the DOM structure can be used to facilitate information extraction.

Unfortunately, due to the flexibility of HTML syntax, many Web pages do not obey the W3C HTML specifications, which may result in errors in the DOM tree structure. Moreover, the DOM tree was initially introduced for presentation in the browser rather than description of the semantic structure of the Web page. For example, even though two nodes in the DOM tree have the same parent, the two nodes might not be more semantically related to each other than to other nodes. Figure 10.7 shows an example page.[5] Figure 10.7(a) shows part of the HTML source (we only keep the backbone code), and Figure 10.7(b) shows the DOM tree of the page. Although we have surrounding description text for each image, the DOM tree structure fails to correctly identify the semantic relationships between different parts.

In the sense of human perception, people always view a Web page as different semantic objects rather than as a single object. Some research efforts show that users

```
<tr>
<td><img src="t1.jpg"></td><td><img src="t2.jpg"></td>
<td><img src="t3.jpg"></td><td><img src="t4.jpg"></td>
</tr>
<tr>
<td>Timber Wolf</td><td>Giraffes</td>
<td>Elephant Sunrise</td><td>Prowling Teen</td>
</tr>
```

(a) Part of HTML source (only keep the backbone)



(b) The DOM tree structure (The picture area and caption area are two different TR nodes)

Figure 10.7 The HTML source and DOM tree structure of a sample page. It is difficult to extract the correct semantic content structure of the page.

always expect that certain functional parts of a Web page (e.g., navigational links or an advertisement bar) appear at certain positions on the page. Actually, when a Web page is presented to the user, the spatial and visual cues can help the user unconsciously divide the Web page into several semantic parts. Therefore, it is possible to automatically segment the Web pages by using the spatial and visual cues. Based on this observation, we can develop algorithms to extract the Web page content structure based on spatial and visual information.

Here, we introduce an algorithm called VIsion-based Page Segmentation (VIPS). VIPS aims to extract the semantic structure of a Web page based on its visual presentation. Such semantic structure is a tree structure: each node in the tree corresponds to a block. Each node will be assigned a value (Degree of Coherence) to indicate how coherent is the content in the block based on visual perception. The VIPS algorithm makes full use of the page layout feature. It first extracts all of the suitable blocks from the HTML DOM tree, and then it finds the separators between these blocks. Here separators denote the horizontal or vertical lines in a Web page that visually cross with no blocks. Based on these separators, the semantic tree of the Web page is constructed. A Web page can be represented as a set of blocks (leaf nodes of the semantic tree). Compared with DOM-based methods, the segments obtained by VIPS are more semantically aggregated. Noisy information, such as navigation, advertisement, and decoration can be easily removed because these elements are often placed in certain positions on a page. Contents with different topics are distinguished as separate blocks. Figure 10.8 illustrates the procedure of VIPS algorithm, and Figure 10.9 shows the partition result of the same page as in Figure 10.7.

10.5.2 Mining the Web's Link Structures to Identify Authoritative Web Pages

"What is meant by authoritative Web pages?" Suppose you would like to search for Web pages relating to a given topic, such as financial investing. In addition to retrieving pages that are relevant, you also hope that the pages retrieved will be of high quality, or *authoritative* on the topic.

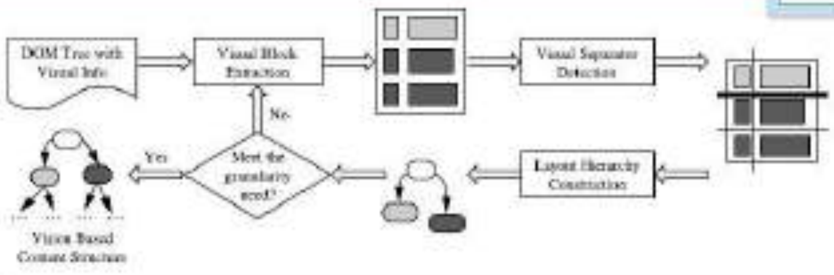
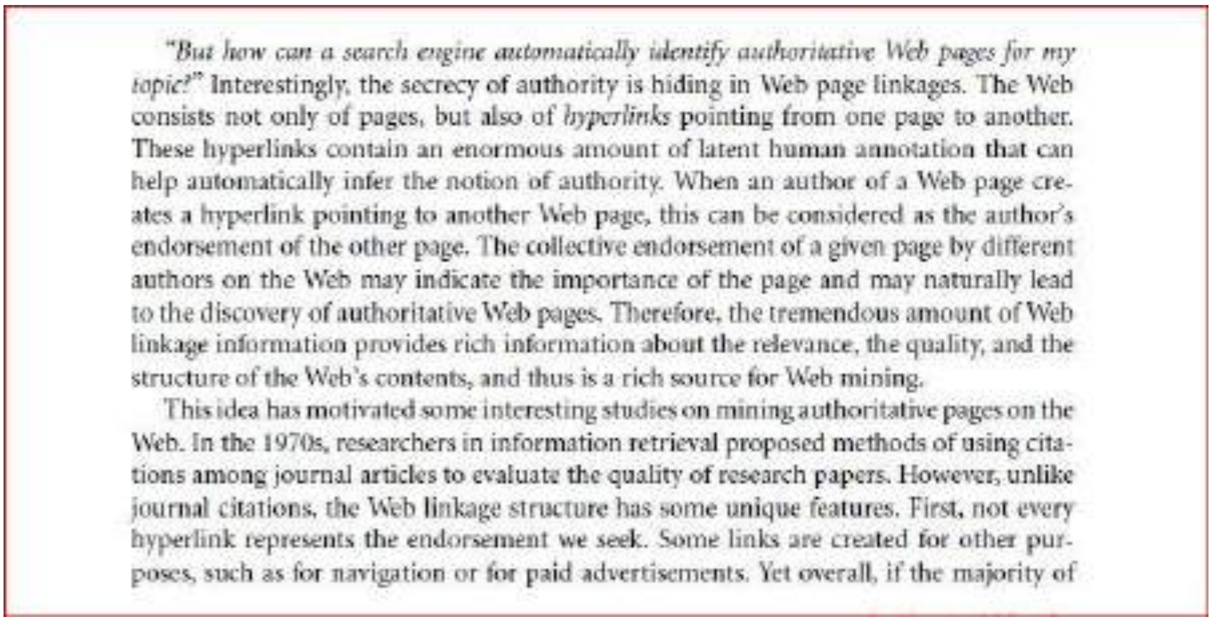

DOM Tree with Visual Info
Visual Block Extraction
Visual Separator Detection
Layout Hierarchy Construction
Meet the granularity need?
No
Yes
Vision Based Content Structure

Figure 10.8 The process flow of vision-based page segmentation algorithm.

Figure 10.9 Partition using VIPS (The image with their surrounding text are accurately identified)



"But how can a search engine automatically identify authoritative Web pages for my topic?" Interestingly, the secrecy of authority is hiding in Web page linkages. The Web consists not only of pages, but also of *hyperlinks* pointing from one page to another. These hyperlinks contain an enormous amount of latent human annotation that can help automatically infer the notion of authority. When an author of a Web page creates a hyperlink pointing to another Web page, this can be considered as the author's endorsement of the other page. The collective endorsement of a given page by different authors on the Web may indicate the importance of the page and may naturally lead to the discovery of authoritative Web pages. Therefore, the tremendous amount of Web linkage information provides rich information about the relevance, the quality, and the structure of the Web's contents, and thus is a rich source for Web mining.

This idea has motivated some interesting studies on mining authoritative pages on the Web. In the 1970s, researchers in information retrieval proposed methods of using citations among journal articles to evaluate the quality of research papers. However, unlike journal citations, the Web linkage structure has some unique features. First, not every hyperlink represents the endorsement we seek. Some links are created for other purposes, such as for navigation or for paid advertisements. Yet overall, if the majority of

hyperlinks are for endorsement, then the collective opinion will still dominate. Second, for commercial or competitive interests, one authority will seldom have its Web page point to its rival authorities in the same field. For example, *Coca-Cola* may prefer not to endorse its competitor *Pepsi* by not linking to *Pepsi*'s Web pages. Third, authoritative pages are seldom particularly descriptive. For example, the main Web page of Yahoo! may not contain the explicit self-description *"Web search engine."*

These properties of Web link structures have led researchers to consider another important category of Web pages called a *hub*. A hub is one or a set of Web pages that provides collections of links to authorities. Hub pages may not be prominent, or there may exist few links pointing to them; however, they provide links to a collection of prominent sites on a common topic. Such pages could be lists of recommended links on individual home pages, such as recommended reference sites from a course home page, or professionally assembled resource lists on commercial sites. Hub pages play the role of implicitly conferring authorities on a focused topic. In general, a good hub is a page that points to many good authorities; a good authority is a page pointed to by many good hubs. Such a mutual reinforcement relationship between hubs and authorities helps the mining of authoritative Web pages and automated discovery of high-quality Web structures and resources.

3 **Mining Multimedia Data on the Web**

A huge amount of multimedia data are available on the Web in different forms. These include video, audio, images, pictures, and graphs. There is an increasing demand for effective methods for organizing and retrieving such multimedia data.
Compared with the general-purpose multimedia data mining, the multimedia data on the Web bear many different properties. Web-based multimedia data are embedded on the Web page and are associated with text and link information. These texts and links can also be regarded as features of the multimedia data. Using some Web page layout mining techniques (like VIPS), a Web page can be partitioned into a set of semantic blocks. Thus, the block that contains multimedia data can be regarded as a whole. Searching and organizing the Web multimedia data can be referred to as searching and organizing the multimedia blocks.

4 **Automatic Classification of Web Documents**

In the automatic classification of Web documents, each document is assigned a class label from a set of predefined topic categories, based on a set of examples of preclassified documents. For example, Yahoo!'s taxonomy and its associated documents can be used as training and test sets in order to derive a Web document classification scheme. This scheme may then be used to classify new Web documents by assigning categories from the same taxonomy.

5 Web Usage Mining

"What is Web usage mining?" Besides mining Web contents and Web linkage structures, another important task for Web mining is Web usage mining, which mines Weblog records to discover user access patterns of Web pages. Analyzing and exploring regularities in Weblog records can identify potential customers for electronic commerce, enhance the quality and delivery of Internet information services to the end user, and improve Web server system performance.
A Web server usually registers a (Web) log entry, or Weblog entry, for every access of a Web page. It includes the URL requested, the IP address from which the request originated, and a timestamp. For Web-based e-commerce servers, a huge number of Web access log records are being collected. Popular websites may register Weblog records in the order of hundreds of megabytes every day. Weblog databases provide rich information about Web dynamics. Thus it is important to develop sophisticated Weblog mining techniques.