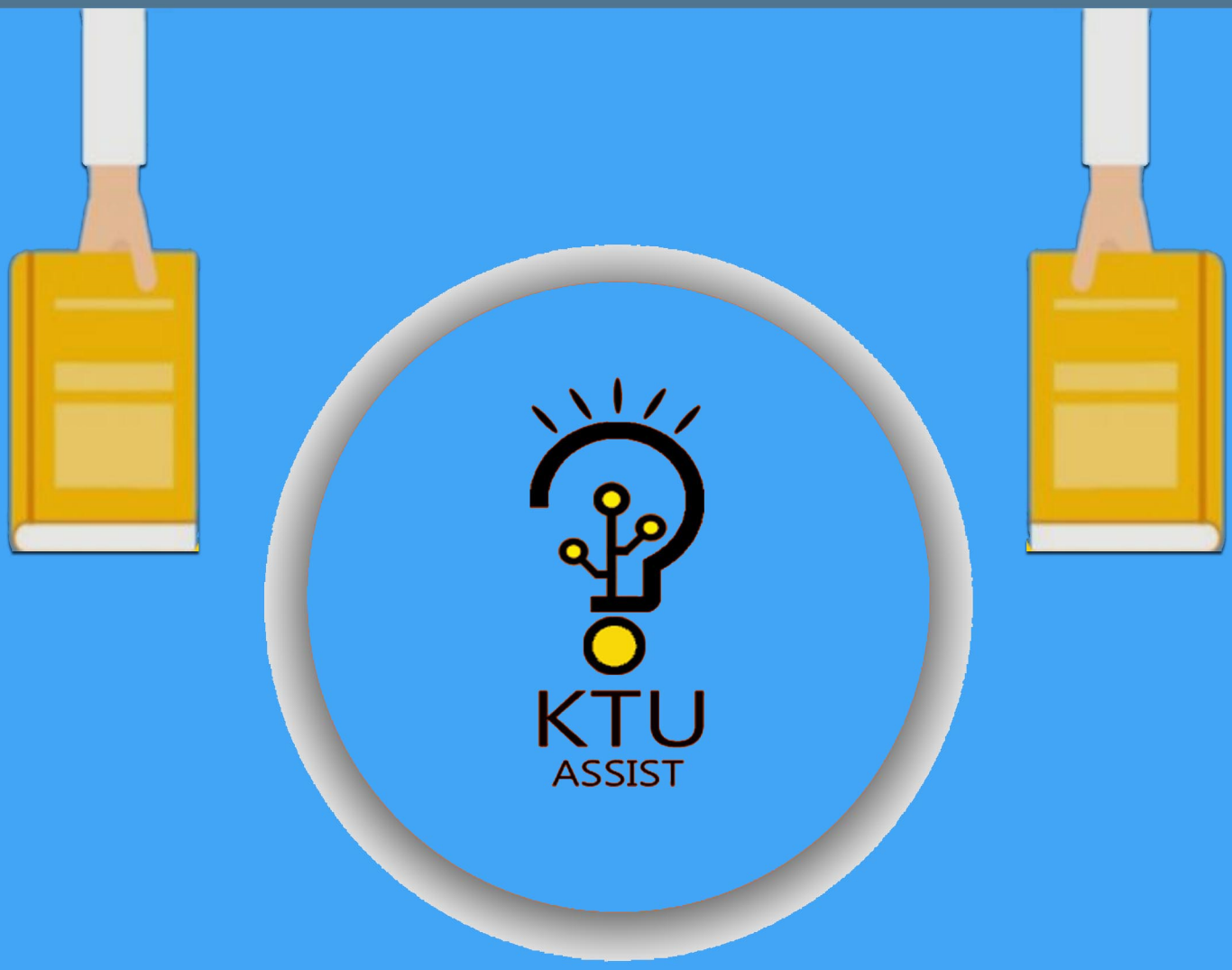
APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY

STUDY MATERIALS



Module 6

Hierarchical Clustering method: BIRCH. Density-Based Clustering –DBSCAN and OPTICS.

Advanced Data Mining Techniques: Introduction, Web Mining- Web Content Mining, Web Structure Mining, Web Usage Mining.

Text Mining. Graph mining:- Apriori based approach for mining frequent subgraphs. Social Network Analysis:- characteristics of social networks.

Link mining:- Tasks and challenges

Hierarchical Clustering method: BIRCH

1. Hierarchical methods

A hierarchical clustering method works by grouping data objects into a **tree of clusters.** Hierarchical clustering methods can be further classified into agglomerative and divisive hierarchical clustering, depending on whether the hierarchical decomposition is formed in a bottom-up or top-down fashion. ***The quality of a pure hierarchical clustering method suffers from its inability to perform adjustment once a merge or split decision has been executed.***

Agglomerative and divisive hierarchical clustering

In general, there are two types of hierarchical clustering methods:

1. **Agglomerative hierarchical clustering:** This bottom-up strategy starts by placing each object in its own cluster and then merges these atomic clusters into larger and larger clusters, until all of the objects are in a single cluster or until certain termination conditions are satisfied. Most hierarchical clustering methods belong to this category. They differ only in their definition of inter-cluster similarity.

2. **Divisive hierarchical clustering:** This top-down strategy does the reverse of agglomerative hierarchical clustering by starting with all objects in one cluster. It subdivides the cluster into smaller and smaller pieces, until each object forms a cluster on its own or until it satisfies certain termination conditions, such as a desired number of clusters is obtained or the distance between the two closest clusters is above a certain threshold distance.

BIRCH: Balanced Iterative Reducing and Clustering using Hierarchies

BIRCH is designed for clustering a large amount of numerical data by integration of hierarchical clustering and other clustering methods such as iterative partitioning**.**

It overcomes the two difficulties of agglomerative clustering methods:

(1) scalability
(2) the inability to undo what was done in the previous step

It introduces two concepts: ***clustering feature and clustering feature tree (CF tree***), which are used to summarize cluster representations.
Let's look closer at the above-mentioned structures. Given "n" d-dimensional data objects or points in a cluster, we can define **the centroid x_0, radius R, and diameter D** of the cluster as follows:

$$x_0 = \frac{\sum_{i=1}^{n} x_i}{n}$$

$$R = \sqrt{\frac{\sum_{i=1}^{n} (x_i - x_0)^2}{n}}$$

$$D = \sqrt{\frac{\sum_{i=1}^{n} \sum_{j=1}^{n} (x_i - x_j)^2}{n(n-1)}}$$

where R is the average distance from member objects to the centroid, and D is the average pairwise distance within a cluster. Both R and D reflect the tightness of the cluster around the centroid. A **clustering feature** (**CF**) is a three-dimensional vector summarizing information about clusters of objects. Given n d-dimensional objects or points in a cluster, $\{x_i\}$, then the CF of the cluster is defined as

$$CF = \langle n, \boldsymbol{LS}, \boldsymbol{SS} \rangle, \tag{7.27}$$

where n is the number of points in the cluster, $\boldsymbol{LS}$ is the linear sum of the n points (i.e., $\sum_{i=1}^{n} x_i$), and $\boldsymbol{SS}$ is the square sum of the data points (i.e., $\sum_{i=1}^{n} x_i^2$).

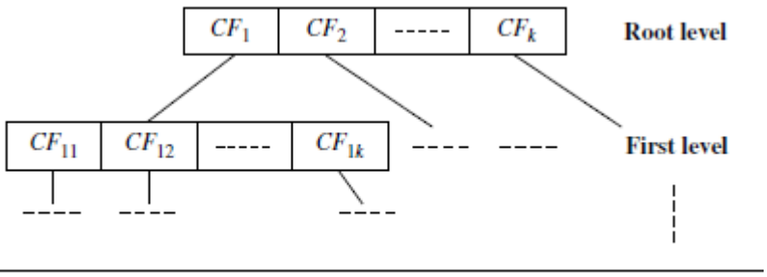
Example 7.10 **Clustering feature.** Suppose that there are three points, (2, 5), (3, 2), and (4, 3), in a cluster, C_1. The clustering feature of C_1 is

$$CF_1 = \langle 3, (2+3+4, 5+2+3), (2^2+3^2+4^2, 5^2+2^2+3^2) \rangle = \langle 3, (9, 10), (29, 38) \rangle.$$

Suppose that C_1 is disjoint to a second cluster, C_2, where $CF_2 = \langle 3, (35, 36), (417, 440) \rangle$. The clustering feature of a new cluster, C_3, that is formed by merging C_1 and C_2, is derived by adding CF_1 and CF_2. That is,

$$CF_3 = \langle 3+3, (9+35, 10+36), (29+417, 38+440) \rangle = \langle 6, (44, 46), (446, 478) \rangle. \quad \blacksquare$$

A **CF tree** is a height-balanced tree that stores the clustering features for a hierarchical clustering. An example is shown in Figure 7.8. By definition, a nonleaf node in a tree has descendants or "children." The nonleaf nodes store sums of the CFs of their children, and thus summarize clustering information about their children. A CF tree has two parameters: *branching factor*, B, and *threshold*, T. The branching factor specifies the maximum number of children per nonleaf node. The threshold parameter specifies the maximum diameter of subclusters stored at the leaf nodes of the tree. These two parameters influence the size of the resulting tree.



CF-tree structure.

«How does the BIRCH algorithm work?" It consists of two phases:

Phase 1: BIRCH scans the database to build an initial in-memoryCF tree, which can be viewed as a multilevel compression of the data that tries to preserve the inherent clustering structure of the data.

Phase 2: BIRCH applies a (selected) clustering algorithm to cluster the leaf nodes of the CF-tree.

2.Density-based methods

To discover **clusters with arbitrary shape**, density-based clustering methods have been developed. These typically regard clusters as dense regions of objects in the data space which are separated by regions of low density (representing noise).

DBSCAN: A density-based clustering method based on connected regions with sufficiently high density

DBSCAN is a density-based clustering algorithm. The algorithm grows regions with sufficiently high density into clusters, and discovers clusters of arbitrary shape in spatial databases with noise. It defines a cluster as a maximal set of density-connected points.

The basic ideas of density-based clustering involve a number of new definitions. The neighborhood within a radius **ε** of a given object is called the **ε** -neighborhood of the object.

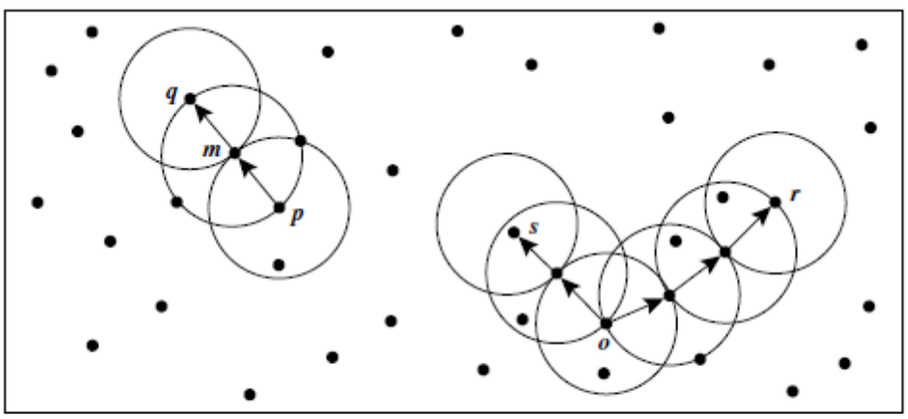
- The neighborhood within a radius ε of a given object is called the **ε-neighborhood** of the object.
- If the ε-neighborhood of an object contains at least a minimum number, *MinPts*, of objects, then the object is called a **core object**.
- Given a set of objects, *D*, we say that an object ***p*** is **directly density-reachable** from object ***q*** if ***p*** is within the ε-neighborhood of ***q***, and ***q*** is a core object.

- An object p is **density-reachable** from object q with respect to ε and *MinPts* in a set of objects, D, if there is a chain of objects $p_1, \ldots, p_n$, where $p_1 = q$ and $p_n = p$ such that p_{i+1} is directly density-reachable from p_i with respect to ε and *MinPts*, for $1 \leq i \leq n$, $p_i \in D$.
- An object p is **density-connected** to object q with respect to ε and *MinPts* in a set of objects, D, if there is an object $o \in D$ such that both p and q are density-reachable from o with respect to ε and *MinPts*.

Density reachability is the transitive closure of direct density reachability, and this relationship is asymmetric. Only core objects are mutually density reachable. Density connectivity, however, is a symmetric relation.

Example 8.5 Consider Figure 8.9 for a given **ε** represented by the radius of the circles, and, say, let MinP ts = 3.

Based on the above definitions:



Density-reachability and density-connectivity in density-based clustering. *Source:* Based on Ester, Kriegel, Sander, and Xu [EKSX96].

- Of the labeled points, M, P, O, and R are core objects since each is in an ε^- neighborhood containing at least three points.
- M is directly density-reachable from P, and Q is directly density-reachable from M.
- Based on the previous observation, Q is (indirectly) density-reachable from P. However, P is not density-reachable from Q. Similarly, R and S are density-reachable from O.
- O, R and S are all density-connected.

A density-based cluster is a set of density-connected objects that is maximal with respect to density-reachability. Every object not contained in any cluster is considered to be noise.

"How does DBSCAN find clusters?" DBSCAN checks the ε-neighborhood of each point in the database. If the ε-neighborhood of a point p contains more than MinPts, a new cluster with p as a core object is created. DBSCAN then iteratively collects directly density-reachable objects from these core objects, which may involve the merge of a few density-reachable clusters. The process terminates when no new point can be added to any cluster

Algorithm: DBSCAN: a density-based clustering algorithm.

Input:

- D: a data set containing n objects,
- ϵ: the radius parameter, and
- *MinPts*: the neighborhood density threshold.

Output: A set of density-based clusters.

Method:

```
(1)  mark all objects as unvisited;
(2)  do
(3)      randomly select an unvisited object p;
(4)      mark p as visited;
(5)      if the ε-neighborhood of p has at least MinPts objects
(6)          create a new cluster C, and add p to C;
(7)          let N be the set of objects in the ε-neighborhood of p;
(8)          for each point p' in N
(9)              if p' is unvisited
(10)                 mark p' as visited;
(11)                 if the ε-neighborhood of p' has at least MinPts points,
                         add those points to N;
(12)             if p' is not yet a member of any cluster, add p' to C;
(13)         end for
(14)         output C;
(15)     else mark p as noise;
(16) until no object is unvisited;
```

DBSCAN algorithm.

OPTICS: Ordering Points To Identify the Clustering Structure

Although DBSCAN (the density-based clustering algorithm) can cluster objects given input parameters such as ε and MinPts, it still leaves the user with the responsibility of selecting parameter values that will lead to the discovery of acceptable clusters. Actually, this is a problem associated with many other clustering algorithms. Such parameter settings are usually empirically set and difficult to determine, especially for real-world, high-dimensional data sets. Most algorithms are very sensitive to such parameter values: slightly different settings may lead to very dfferent clusterings of the data. Moreover, high-dimensional real data sets often have very skewed distributions.

There does not even exist a global parameter setting for which the result of a clustering algorithm may accurately describe the intrinsic clustering structure.

To help overcome this difficulty, a cluster ordering method called OPTICS (Ordering Points To Identify the Clustering Structure) was proposed. Rather than produce a data set clustering explicitly, OPTICS computes an augmented cluster ordering for automatic and interactive cluster analysis.

By examining DBSCAN, one can easily see that for a constant MinPts -value, density-based clusters with respect to a higher density (i.e., a lower value for **ε**) are completely contained in density-connected sets obtained with respect to a lower density.

OPTICS require two values need to be stored for each object **core-distance and reachability-distance**:

- The **core-distance** of an object p is the smallest **ε′** value that makes p a core object. If p is not a core object, the core-distance of p is undefined.
- The **reachability-distance** of an object p and another object o is the greater value of the core-distance of p and the Euclidean distance between p and q. If p is not a core object, the reachability-distance between p and q is undefined.

«How are these values used?" The OPTICS algorithm creates an ordering of the objects in a database, additionally storing the core-distance and a suitable reachability-distance for each object. Such information is sufficient for the extraction of all density-based clusterings with respect to any distance **ε′** that is smaller than the distance **ε** used in generating the order.

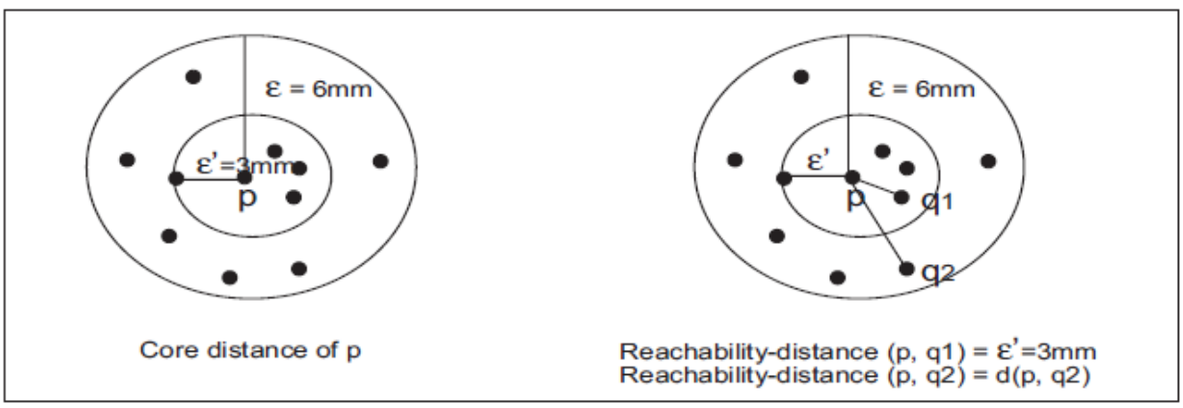


Figure 8.10: OPTICS terminology.

Figure 8.10 illustrates the concepts of core-distance and reachability-distance. Suppose that **ε** = 6mm and MinPts = 5. The core-distance of p is the distance, **ε′**, between p and the fourth closest data object.

The cluster ordering of a data set can be represented graphically, which helps in its understanding. For example, Figure above is the reachability plot for a simple 2-

dimensional data set, which presents a general overview of how the data are structured and clustered. Methods have also been developed for viewing clustering structures of high- dimensional data.

Because of the structural equivalence of the OPTICS algorithm to DBSCAN, the OPTICS algorithm has the same run-time complexity as that of DBSCAN, i.e., O(n log n). Spatial indexing structures can be used to further enhance its performance.

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, andWeb 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 WorldWideWeb (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 semistructured 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.

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.

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}.***

There are three basic measures for assessing the quality of text retrieval:

Precision, recall, and F-score

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.

- **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.

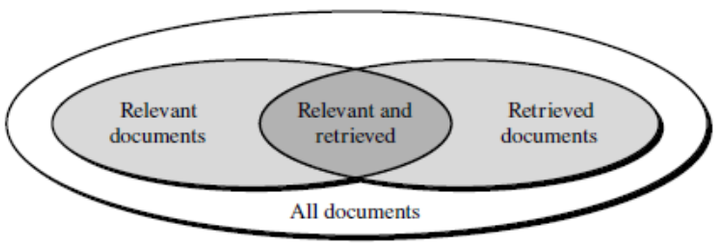


Figure 10.6 Relationship between the set of relevant documents and the set of retrieved documents.

Web Mining

INTRODUCTION

Determining the size of the World Wide Web is extremely difficult. The Web can be viewed. as the the largest database available and presents a challenging task for effective design and access, Here we use the term database quite loosely because, of course, there is no real structure or schema to the Web. Thus, data mining applied to the Web has the potential to be quite beneficial.

Web mining is mining of data related to the World Wide Web.

Web data can be classified into the following classes :

- Content of actual Web pages.
- Intrapage structure includes the HTML or XML code for the page.
- Interpage structure is the actual linkage structure between Web pages.
- Usage data that describe how Web pages are accessed by visitors.
- User profiles include demographic and registration information obtained about users. This could also include information found in cookies.

Web mining tasks can be divided into several classes. Figure 7.1 shows one taxonomy of Web mining activities.

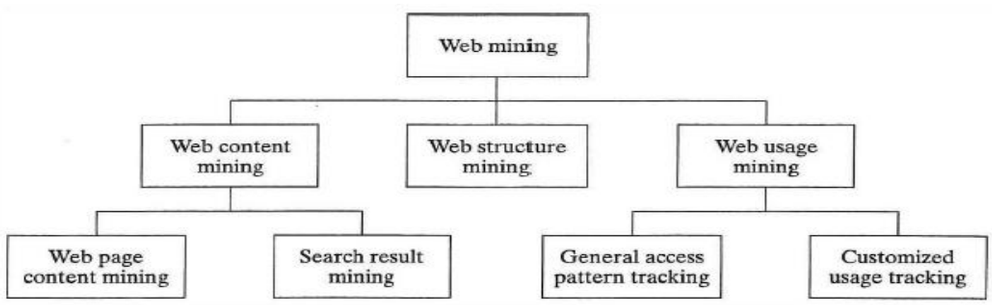


FIGURE 7.1: Web mining taxonomy (modified from [Zaï99]).

Web content mining examines the content of Web pages as well as results of Web searching. The content includes text as well as graphics data.

Web content mining is further divided into **Web page content mining and search results mining**.

The first is traditional searching of Web pages via content, while the second is a further search of pages found from a previous search.

Content mining is similar to the work performed by basic Information Retrieval techniques (IR), but it usually goes farther than simply employing keyword searching. For example, clustering may be applied to Web pages to identify similar pages. The

intrapage structure includes links within the page as well as the code (HTML, XML) for the page.

Web usage mining looks at logs of Web access. General access pattern tracking is a type of usage mining that looks at a history of Web pages visited. This usage may be general or may be targeted to specific usage or users. Besides identifying what the traffic patterns look like, usage mining also involves the mining of these sequential patterns.

For example, patterns can be clustered based on their similarity. This in turn can be used to cluster users into groups based on similar access behavior.

WEB CONTENT MINING

Web content mining can be thought of as extending the work performed by basic search engines. There are many different techniques that can be used to search the Internet. Only a few of these techniques are discussed here. Most search engines are keyword based. Web content mining goes beyond this basic IR technology. It can improve on traditional search engines through such techniques as concept hierarchies and synonyms, user profiles, and analyzing the links between pages. Traditional search engines must have crawlers to search the Web and gather information, indexing techniques to store the information, and query processing support to provide fast and accurate information to users. Data mining techniques can be used to help search engines provide the efficiency, effectiveness, and scalability needed.

One problem associated with retrieval of data from Web documents is that they are not structured as in traditional databases. There is no schema or division into attributes.

Traditionally, Web pages are defined using hypertext markup language (HTML). Web pages created using HTML are only semistructured, thus making querying more difficult than with well-formed databases containing schemas and attributes with defined domains. HTML ultimately will be replaced by extensible markup language (XML), which will provide structured documents and facilitate easier mining.

Crawlers

A robot (or spider or crawler) is a program that traverses the hypertext structure in the Web. The page (or set of pages) that the crawler starts with are referred to as the seed URLs. By starting at one page, all links from it are recorded and saved in a queue. These new pages are in turn searched and their links are saved. As these robots search the Web, they may collect information about each page, such as extract keywords and store in indices for users of the associated search engine.

A crawler may visit a certain number of pages and then stop, build an index, and replace the existing index. This type of crawler is referred to as a **periodic crawler** because it is activated periodically.

Recent research has examined how to use an **incremental crawler**. Traditional crawlers usually replace the entire index or a section thereof. An incremental crawler

selectively searches the Web and only updates the index incrementally as opposed to replacing it. Because of the tremendous size of the Web, it has also been proposed that a focused crawler be used.

A **focused crawler** visits pages related to topics of interest. This concept is illustrated in Figure 7.3.

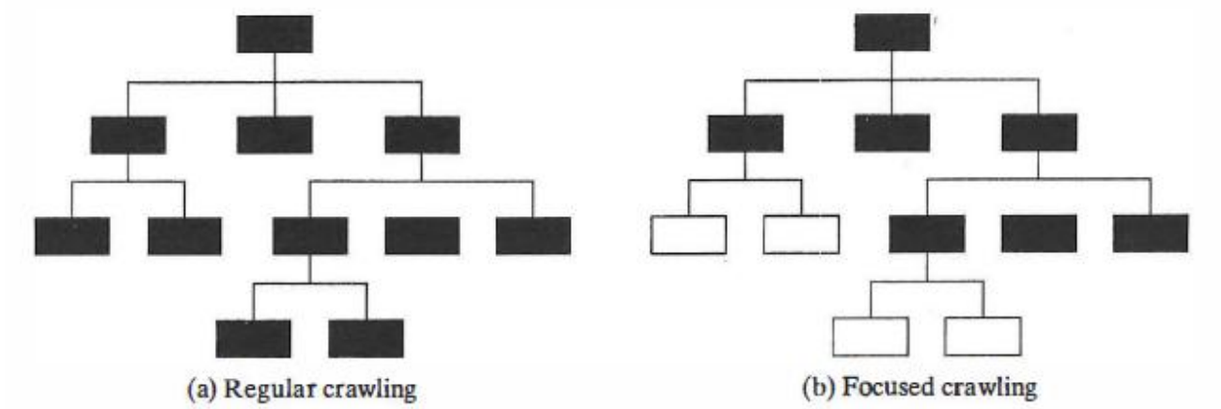


FIGURE 7.3: Focused crawling.

Figure 7.3(a) illustrates what happens with regular crawling, while Figure 7.3(b) illustrates focused crawling. The shaded boxes represent pages that are visited. With focused crawling, if it is determined that a page is not relevant or its links should not be followed, then the entire set of possible pages underneath it are pruned and not visited. With thousands of focused crawlers, more of the Web can be covered than with traditional crawlers. This facilitates better scalability as the Web grows. The focused crawler architecture consists of three primary components.

Harvest System

The Harvest system is based on the use of caching, indexing, and crawling. Harvest is actually a set of tools that facilitate gathering of information from diverse sources. The Harvest design is centered around the use of gatherers and brokers. A gatherer obtains information for indexing from an Internet service provider, while a broker provides the index and query interface. The relationship between brokers and gatherers can vary. Brokers may interface directly with gatherers or may go through other brokers to get to the gatherers. Indices in Harvest are topic-specific, as are brokers. This is used to avoid the scalability problems found without this approach.

Virtual Web View

One proposed approach to handling the large amounts of somewhat unstructured data on the Web is to create a multiple layered database (MLDB) on top of the data in the Web (or a portion thereof). This database is massive and distributed. Each layer of this database is more generalized than the layer beneath it. Unlike the lowest level (the Web), the upper levels are structured and can be accessed (and mined) by an SQL-like query language. The MLDB provides an abstracted and condensed view

of a portion of the Web. A view of the MLDB, which is called a Virtual Web View (VWV), can be constructed.

The layer- 1 data can be viewed as a massive distributed database. The higher levels of the database become less distributed and more summarized as they move up the hierarchy. Generalization tools are proposed, and concept hierarchies are used to assist in the generalization process for constructing the higher levels of the MLDB . These hierarchies can be created using the WordNet Semantic Network. WordNet is a database of the English language. Nouns, adjectives, verbs, and adverbs are listed, divided into groups of synonyms, and linked together using both lexical and semantic relationships.

A Web data mining query language, **WebML** is proposed to provide data mining operations on the MLDB . WebML is an extension of DMQL. Documents are accessed using data mining operations and lists of keywords. A major feature of WebML are four primitive operations based on the use of concept hierarchies for the keywords [ZaY99] :

1. COVERS : One concept covers another if it is higher (ancestor) in the hierarchy. This coverage is extended to include synonyms as well.

2. COVERED BY: This is the reverse of COVERS in that it reverses to descendents.

3. LIKE: The concept is a synonym.

4. CLOSE TO: One concept is close to another if it is a sibling in the hierarchy. Again, this is extended to include synonyms.

The following example illustrates WebML. The query finds all documents at the level of "www .engr.smu.edu" that have a keyword that covers the keyword cat:

```
SELECT *

FROM document in  ' ' www . engr . srnu . edu ' '

WHERE ONE OF keywords COVERS  ' ' cat ' '
```

WebML allows queries to be stated such that the WHERE clause indicates selection based on the links found in the page, keywords for the page, and information about the domain Where the document is found. Because WebML is an extension of DMQL, data mining functions such as classification, summarization, association rules, clustering, and prediction are included.

Personalization

Another example of Web content mining is in the area of **personalization**. With personalization, Web access or the contents of a Web page are modified to better fit the desires of the user. This may involve actually creating Web pages that are unique per user or using the desires of a user to determine what Web documents to retrieve.

With personalization, advertisements to be sent to a potential customer are chosen based on specific knowledge concerning that customer. Unlike targeting, personalization may be performed on the target Web page. The goal here is to entice a current customer to purchase something he or she may not have thought about purchasing. Perhaps the simplest example of personalization is the use of a visitor' s name when he or she visits a page. Personalization is almost the opposite of targeting. With targeting, businesses display advertisements at other sites visited by their users. With personalization, when a particular person visits a Web site, the advertising can be designed specifically for that person.

.

Personalization can be viewed as a type of clustering, classification, or even prediction. Through classification, the desires of a user are determined based on those for the class. With clustering, the desires are determined based on those users to which he or she is determined to be similar. Finally, prediction is used to predict what the user really wants to see. There are three basic types of Web page: personalization

• Manual techniques perform personalization through user registration preferences or via the use of rules that are used to classify individuals based on profiles or demographics.

• Collaborative filtering accomplishes personalization by recommending information (pages) that have previously been given high ratings from similar users.

• Content-based filtering retrieves pages based on similarity between them and user profiles.

WEB STRUCTURE MINING

Web structure mining can be viewed as creating a model of the Web organization or a portion thereof. This can be used to classify Web pages or to create similarity measures between documents. We have already seen some structure mining ideas presented in the content mining section. These approaches used structure to improve on the effectiveness of search engines and crawlers.

PageRank

The PageRank technique was designed to both increase the effectiveness of search engines and improve their efficiency. PageRank is used to measure the importance of a page and to prioritize pages returned from a traditional search engine using keyword searching. The effectiveness of this measure has been demonstrated by the success of Google.

The PageRank value for a page is calculated based on the number of pages that point to it. This is actually a measure based on the number of backlinks to a page. A backlink is a link pointing to a page rather than pointing out from a page. The measure is not simply a count of the number of backlinks because a weighting is used to provide more importance to backlinks coming from important pages. Given a page p,

we use Bp to be the set of pages that point to p, and Fp to be the set of links out of p. The PageRank of a page p is defined as [PBMW98]

$$\mathrm{PR}(p) = c \sum_{q \in B_p} \frac{\mathrm{PR}(q)}{N_q} \qquad (7.4)$$

(7.4)

Here $N_q = I\ F_q\ |$. The constant c is a value between 0 and 1 and is used for normalization. A problem, called ***rank sink***, that exists with this PageRank calculation is that when a cyclic reference occurs (page A points to page B and page B points to page A), the PR value for these pages increases. This problem is solved by adding an additional term to the formula:

$$\mathrm{PR}'(p) = c \sum_{q \in B_p} \frac{\mathrm{PR}(q)}{N_q} + cE(v) \qquad (7.5)$$

where c is maximized. Here E (v) is a vector that adds an artificial link. This simulates a random surfer who periodically decides to stop following links and jumps to a new page. E(v) adds links of small probabilities between every pair of nodes. The PageRank technique is different from other approaches that look at links. It does not count all links the same. The values are normalized by the number of links in the page.

Graph Mining

Graphs become increasingly important in modeling complicated structures, such as circuits, images, chemical compounds, protein structures, biological networks, social networks, the Web, workflows, and XML documents. With the increasing demand on the analysis of large amounts of structured data, graph mining has become an active and important theme in data mining.

Among the various kinds of graph patterns, ***frequent substructures*** are the very basic patterns that can be discovered in a collection of graphs. They are useful for characterizing graph sets, discriminating different groups of graphs, classifying and clustering graphs, building graph indices, and facilitating similarity search in graph databases.

Methods for Mining Frequent Subgraphs

We denote the **vertex set** of a graph g by $V(g)$ and the **edge set** by $E(g)$. A label function, L, maps a vertex or an edge to a label. A graph g is a **subgraph** of another graph g' if there exists a subgraph isomorphism from g to g'. Given a labeled graph data set, $D = \{G_1, G_2, \ldots, G_n\}$, we define $support(g)$ (or $frequency(g)$) as the percentage (or number) of graphs in D where g is a subgraph. A **frequent graph** is a graph whose support is no less than a minimum support threshold, min_sup.

Example: Frequent subgraph.

Figure 9.1 shows a sample set of chemical structures.

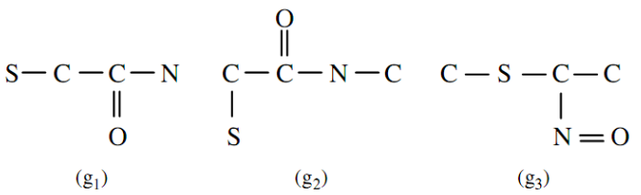


Figure 9.1 A sample graph data set.

Figure 9.2 depicts two of the frequent subgraphs in this data set, given a minimum support of 66.6%.

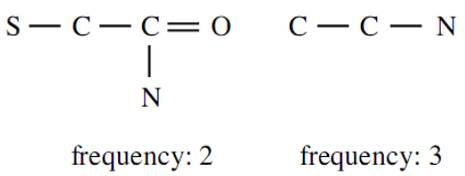


Figure 9.2 Frequent graphs.

"How can we discover frequent substructures?"

The discovery of frequent substructures usually consists of two steps. In the first step, we generate frequent substructure candidates. The frequency of each candidate is checked in the second step.

Most studies on frequent substructure discovery focus on the optimization of the first step, because the second step involves a subgraph isomorphism test whose computational complexity is excessively high (i.e., NP-complete).

Apriori-based Approach

Apriori-based frequent substructure mining algorithms share similar characteristics with Apriori-based frequent itemset mining algorithms. The search for frequent graphs starts with graphs of small "size," and proceeds in a bottom-up manner by generating candidates having an extra vertex, edge, or path. The definition of graph size depends on the algorithm used.

The general framework of Apriori-based methods for frequent substructure mining is outlined in Figure 9.3.

We refer to this algorithm as AprioriGraph. $\mathbf{S_k}$ is the frequent substructure set of size **k**. AprioriGraph adopts a ***level-wise*** mining methodology. At each iteration, the size of newly discovered frequent substructures is increased by one. These new substructures

are first generated by joining two similar but slightly different frequent subgraphs that were discovered in the previous call to AprioriGraph. This candidate generation procedure is outlined on line 4. The frequency of the newly formed graphs is then checked. Those found to be frequent are used to generate larger candidates in the next round.

Algorithm: AprioriGraph. Apriori-based frequent substructure mining.

Input:

- D, a graph data set;
- min_sup, the minimum support threshold.

Output:

- S_k, the frequent substructure set.

Method:

$S_1 \leftarrow$ frequent single-elements in the data set;
Call AprioriGraph(D, min_sup, S_1);

procedure AprioriGraph(D, min_sup, S_k)

(1) $S_{k+1} \leftarrow \varnothing$;
(2) **for each** frequent $g_i \in S_k$ **do**
(3) **for each** frequent $g_j \in S_k$ **do**
(4) **for each** size $(k+1)$ graph g formed by the merge of g_i and g_j **do**
(5) **if** g is frequent in D and $g \notin S_{k+1}$ **then**
(6) insert g into S_{k+1};
(7) **if** $s_{k+1} \neq \varnothing$ **then**
(8) *AprioriGraph*(D, min_sup, S_{k+1});
(9) **return**;

Figure 9.3 AprioriGraph.

Social Network Analysis

The notion of social networks, where relationships between entities are represented as links in a graph, has attracted increasing attention in the past decades. Thus social network analysis, from a data mining perspective, is also called link analysis or link mining.

What Is a Social Network?

From the point of view of data mining, a social network is a heterogeneous and multi-relational data set represented by a graph.

The graph is typically very large, with nodes corresponding to objects and edges corresponding to links representing relationships or interactions between objects. Both

nodes and links have attributes. Objects may have class labels. Links can be one-directional and are not required to be binary.

Characteristics of Social Networks

Social networks are rarely static. Their graph representations evolve as nodes and edges are added or deleted over time. In general, social networks tend to exhibit the following phenomena:

1. Densification power law: Previously, it was believed that as a network evolves, the number of degrees grows linearly in the number of nodes. However, extensive experiments have shown that networks become increasingly dense over time with the average degree increasing (and hence, the number of edges growing super linearly in the number of nodes).

2. Shrinking diameter: It has been experimentally shown that the effective diameter tends to decrease as the network grows. This contradicts an earlier belief that the diameter slowly increases as a function of network size.

3. Heavy-tailed out-degree and in-degree distributions: The number of out-degrees for a node tends to follow a heavy-tailed distribution by observing the power law, $1/n^a$, where n is the rank of the node in the order of decreasing out-degrees and typically, $0 < a < 2$ (Figure 9.17). The smaller the value of a, the heavier the tail. This phenomena is represented in the preferential attachment model, where each new node attaches to an existing network by a constant number of out-links, following a "rich-get-richer" rule. The in-degrees also followa heavy-tailed distribution, although it tends be more skewed than the out-degrees distribution.

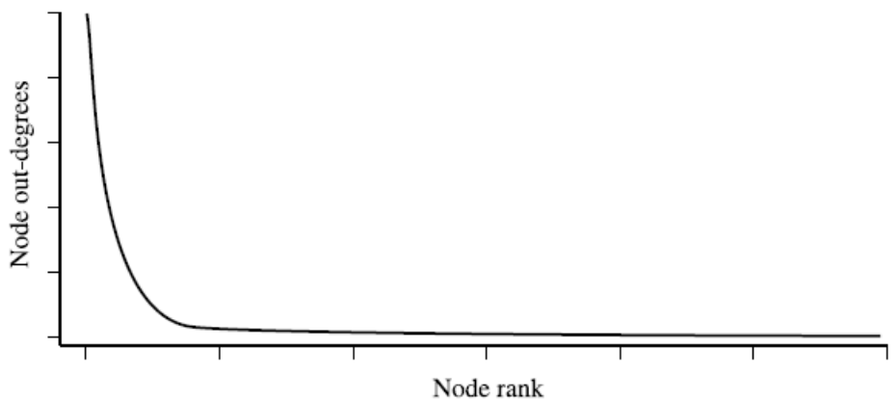


Figure 9.17 The number of out-degrees (*y*-axis) for a node tends to follow a heavy-tailed distribution. The node rank (*x*-axis) is defined as the order of deceasing out-degrees of the node.

A Forest Fire model for graph generation was proposed, which captures these characteristics of graph evolution over time. It is based on the notion that new nodes attach to the network by "burning" through existing edges in epidemic fashion. It uses two parameters, forward burning probability, p, and backward burning ratio, r, which

are described below. Suppose a new node, v, arrives at time t. It attaches to G_t , the graph constructed so far, in the following steps:

1. It chooses an ambassador node, w, at random, and forms a link to w.

2. It selects x links incident to w, where x is a random number that is binomially distributed with mean $(1-p)^{-1}$. It chooses from out-links and in-links of w but selects in-links with probability r times lower than out-links. Let w1 , w2 , ... wx denote the nodes at the other end of the selected edges.

3. Our new node, v, forms out-links to w1 , w2 , ... wx and then applies step 2 recursively to each of w1 , w2 , ... wx. Nodes cannot be visited a second time so as to prevent the construction from cycling. The process continues until it dies out.

Link Mining: Tasks and Challenges

"How can we mine social networks?"

Traditional methods of machine learning and data mining, taking, as input, a random sample of homogenous objects from a single relation, may not be appropriate here. The data comprising social networks tend to be heterogeneous, multi-relational, and semi-structured. As a result, a new field of research has emerged called **link mining**.

Link mining is a confluence of research in social networks, link analysis, hypertext and Web mining, graph mining, relational learning, and inductive logic programming. It embodies descriptive and predictive modeling.

By considering links (the relationships between objects), more information is made available to the mining process. This brings about several new tasks. Here, we list these tasks with examples from various domains:

1. Link-based object classification. In traditional classification methods, objects are classified based on the attributes that describe them. Link-based classification predicts the category of an object based not only on its attributes, but also on its links, and on the attributes of linked objects.

Web page classification is a well-recognized example of link-based classification. It predicts the category of a Web page based on word occurrence (words that occur on the page) and anchor text (the hyperlink words, that is, the words you click on when you click on a link), both of which serve as attributes.

2. Object type prediction. This predicts the type of an object, based on its attributes and its links, and on the attributes of objects linked to it. In the bibliographic domain, we may want to predict the venue type of a publication as either conference, journal, or workshop. In the communication domain, a similar task is to predict whether a communication contact is by e-mail, phone call, or mail.

3. Link type prediction. This predicts the type or purpose of a link, based on properties of the objects involved. Given epidemiological data, for instance, we may try

to predict whether two people who know each other are family members, coworkers, or acquaintances. In another example, we may want to predict whether there is an advisor-advisee relationship between two coauthors. Given Web page data, we can try to predict whether a link on a page is an advertising link or a navigational link.

4. Predicting link existence. Unlike link type prediction, where we know a connection exists between two objects and we want to predict its type, instead we may want to predict whether a link exists between two objects. Examples include predicting whether there will be a link between two Web pages, and whether a paper will cite another paper. In epidemiology, we can try to predict with whom a patient came in contact.

5. Link cardinality estimation. There are two forms of link cardinality estimation. First, we may predict the number of links to an object. This is useful, for instance, in predicting the authoritativeness of aWeb page based on the number of links to it (in-links). Similarly, the number of out-links can be used to identifyWeb pages that act as hubs, where a hub is one or a set ofWeb pages that point to many authoritative pages of the same topic. In the bibliographic domain, the number of citations in a paper may indicate the impact of the paper—the more citations the paper has, the more influential it is likely to be. In epidemiology, predicting the number of links between a patient and his or her contacts is an indication of the potential for disease transmission.

6. Object reconciliation. In object reconciliation, the task is to predict whether two objects are, in fact, the same, based on their attributes and links. This task is common in information extraction, duplication elimination, object consolidation, and citation matching, and is also known as record linkage or identity uncertainty. Examples include predicting whether two websites are mirrors of each other, whether two citations actually refer to the same paper, and whether two apparent disease strains are really the same.

7. Group detection. Group detection is a clustering task. It predicts when a set of objectsbelong to the same group or cluster, based on their attributes as well as their link structure. An area of application is the identification of Web communities, where a Web community is a collection of Web pages that focus on a particular theme or topic. A similar example in the bibliographic domain is the identification of research communities.

8. Subgraph detection. Subgraph identification finds characteristic subgraphs within networks. This is a formof graph search and was described in Section 9.1. An example from biology is the discovery of subgraphs corresponding to protein structures. In chemistry, we can search for subgraphs representing chemical substructures.

9. Metadata mining. Metadata are data about data. Metadata provide semi-structured data about unstructured data, ranging from text and Web data to multimedia databases. It is useful for data integration tasks in many domains. Metadata mining can be used for schema mapping schema discovery, which generates schema from semi-structured data; and schema reformulation, which refines the schema based on the mined metadata.

Link Mining challenges:

1. Logical versus statistical dependencies. Two types of dependencies reside in the graph—link structures (representing the logical relationship between objects) and probabilistic dependencies (representing statistical relationships, such as correlation between attributes of objects where, typically, such objects are logically related). The coherent handling of these dependencies is also a challenge for multirelational data mining, where the data to be mined exist in multiple tables. We must search over the different possible logical relationships between objects, in addition to the standard search over probabilistic dependencies between attributes. This takes a huge search space, which further complicates finding a plausible mathematical model. Methods developed in inductive logic programmingmay be applied here,which focus on search over logical relationships.

2. Feature construction. In link-based classification, we consider the attributes of an object as well as the attributes of objects linked to it. In addition, the links may also have attributes. The goal of feature construction is to construct a single feature representing these attributes. This can involve feature selection and feature aggregation. In feature selection, only the most discriminating features are included. Feature aggregation takes amultiset of values over the set of related objects and returns a summary of it. This summary may be, for instance, the mode (most frequently occurring value); the mean value of the set (if the values are numerical); or the median or "middle" value (if the values are ordered). However, in practice, this method is not always appropriate.

3. Instances versus classes. This alludes to whether the model refers explicitly to individuals or to classes (generic categories) of individuals. An advantage of the former model is that it may be used to connect particular individuals with high probability. An advantage of the latter model is that it may be used to generalize to new situations, with different individuals.

4. Collective classification and collective consolidation. Consider training a model for classification, based on a set of class-labeled objects. Traditional classification methods consider only the attributes of the objects. After training, suppose we are given a new set of unlabeled objects. Use of the model to infer the class labels for the new objects is complicated due to possible correlations between objects—the labels of linked objects may be correlated. Classification should therefore involve an additional iterative step that updates (or consolidates) the class label of each object based on the labels of objects linked to it. In this sense, classification is done collectively rather than independently.

5. Effective use of labeled and unlabeled data. Arecent strategy in learning is to incorporate a mix of both labeled and unlabeled data.Unlabeled data can help infer the object attribute distribution. Links between unlabeled (test) data allowus to use attributes of linked objects. Links between labeled (training) data and unlabeled (test) data induce dependencies that can help make more accurate inferences.

6. Link prediction. A challenge in link prediction is that the prior probability of a particular link between objects is typically extremely low. Approaches to link prediction have been proposed based on a number of measures for analyzing the proximity of nodes in a network. Probabilistic models have been proposed as well. For large data sets, it may be more effective to model links at a higher level.

7. Closed versus open world assumption. Most traditional approaches assume that we know all the potential entities in the domain. This "closed world" assumption is unrealistic in real-world applications.Work in this area includes the introduction of a language for specifying probability distributions over relational structures that involve a varying set of objects.

8. Community mining from multi-relational networks. Typical work on social network analysis includes the discovery of groups of objects that share similar properties. This is known as community mining. Web page linkage is an example, where a discovered community may be a set of Web pages on a particular topic. Most algorithms for community mining assume that there is only one social network, representing a relatively homogenous relationship. In reality, there exist multiple, heterogeneous social networks, representing various relationships. A new challenge is the mining of hidden communities on such heterogeneous social networks, which is also known as community mining on multirelational social networks.

These challenges will continue to stimulate much research in link mining.

try it now

A KTU
STUDENTS
PLATFORM

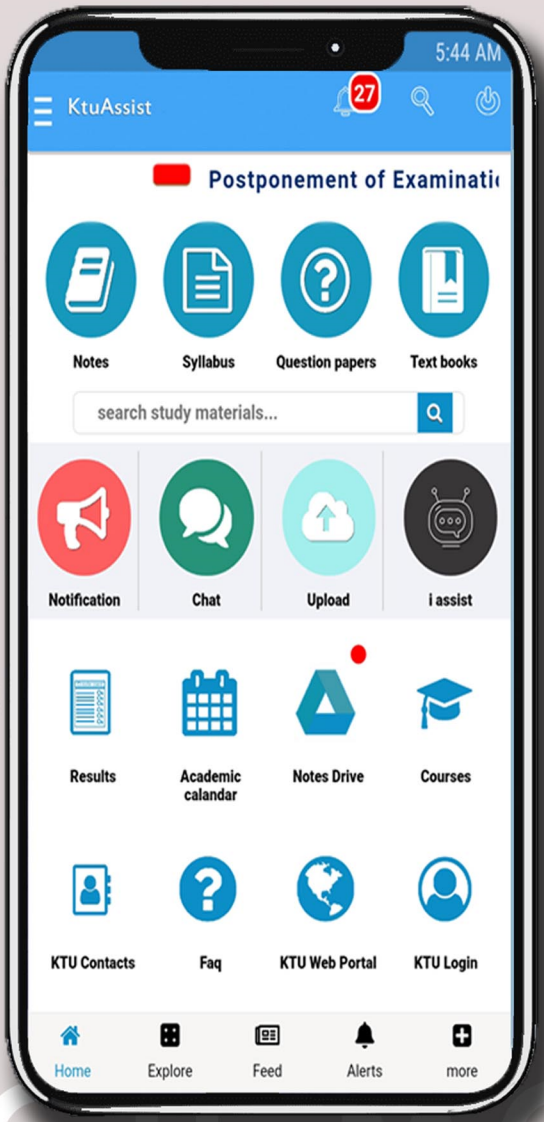
SYLLABUS

NOTES

TEXT BOOKS

QUESTION PAPERS

KTU NOTIFICATION



DOWNLOAD
IT
FROM
GOOGLE PLAY

CHAT

A FAQ

LOGIN

CALENDAR

MUCH MORE

DOWNLOAD APP

ktuassist.in

instagram.com/ktu_assist

facebook.com/ktuassist