Chapter 2

WEB SEARCH ENGINES AND WEB MINING: A SURVEY

2.1 WWW

The World Wide Web, abbreviated as WWW [1] and commonly known as the Web, is a system of interlinked hypertext documents accessed via the Internet. The Internet is a global system of interconnected computer networks that use the standard Internet Protocol Suite [16] (TCP/IP) to serve billions of users worldwide. WWW resources are organized to allow users to navigate easily from one resource to another. The navigation is done by WWW using an application known as a WWW browser client. The browser presents formatted text, images, sound, or other objects, such as hyperlinks, in the form of a WWW page on a computer screen. The user can click on a hyperlink to navigate to other WWW pages. These links exist across the global Internet to form a large-scale, distributed, multimedia knowledge base that relates words, phrases, images, or other information.

In 1989, Timothy Berners-Lee [17] developed a hyperlinked structure to enable information to be shared among internationally dispersed teams of researchers at the European Laboratory for Particle Physics (formerly known by the acronym CERN) near Geneva, Switzerland. It subsequently became a platform for the development of WWW. Its further development is guided by the WWW Consortium based at the Massachusetts Institute of Technology in Cambridge, Massachusetts.

The recent explosion of WWW has transformed not only the disciplines of computer-related sciences and engineering but also the lifestyles of people and economies of countries. Since its onset, the traffic on WWW has ever been increasing. In fact, between 1995 and 2006, the total amount of web traffic increased from about 10 terabytes a month to 1,000,000 terabytes (or 1 exabyte where 1 EB = 10^{18} bytes). According to Cisco [18,
19], total internet traffic rose from about 1 exabyte to 7 exabytes between 2005 and 2010. The exponential growth in terms of actual total traffic is shown in the graph of Fig. 2.1:

![Fig. 2.1 Proportion of Total Internet Traffic [19]](image)

The WWW has emerged as the most widely used tool for information access and dissemination throughout the Internet. It has evolved rapidly from a simple information-sharing mechanism offering only static text and images to a rich assortment of dynamic and interactive services, such as video/audio conferencing, e-commerce, and distance learning. It enables computer users to view and interact with a variety of information, including magazine archives, public and university-library resources, current world and business news etc. As a result, the web has become a sea of all kinds of data, making any manual search into the huge information reservoir extremely difficult. This is where the need and the role of search engines come into picture. It is a program that searches for the documents for a specified query and returns the list of documents where the query keywords were found. The following section aims to survey the search engines, their history and the basic working architecture.

### 2.2 SEARCH ENGINES

Before the search engines [20] were invented, users of the net confined to visiting the web sites they already knew of in the hopes of finding a useful link, or finding what they
wanted through word of mouth. This may have been adequate in the early days of the Internet, but as the WWW continued to grow exponentially, it became necessary to develop automatic means of locating desired content. At first, search services were quite rudimentary, but over the years, they have grown quite sophisticated [21]. Not to mention how popular they are, search services are now among the most frequented sites on the web with millions of hits every day. As stated by ‘comscore’ in its press release in 2010 [22], the total worldwide search market boasted more than 131 billion searches conducted by people of age 15 or older from home and work locations in December 2009, representing a 46-percent increase in the past year.

### 2.2.1 HISTORY OF SEARCH ENGINES

1. **Gerard Salton (1960s - 1990s)**

‘Gerard Salton’ [23] was the father of modern search technology. His teams at Harvard and Cornell developed the SMART informational retrieval system. Salton’s ‘Magic Automatic Retriever of Text’ included important concepts like the vector space model, Inverse Document Frequency (IDF), Term Frequency (TF), term discrimination values, and relevancy feedback mechanisms. He authored a 56 page book called “A Theory of Indexing” which does a great job explaining many of his tests upon which search is still largely based.


‘Archie’ [24, 25] originally known as “ARCHIVES” evolved from a university student project into commercial enterprise revolutionizing the field as it grew. Created in 1990 by Alan Emtage, a McGill University student, Archie archived what at the time was the most popular repository of Internet files and anonymous FTP sites. What ‘Archie’ did for FTP sites, ‘Veronica’ did for Gopherspace. Gopherspace is a term used to describe the aggregate of all the information on the thousands of Gopher servers in the world. This information consists of individual files (abstracts, full-length documents or papers, lists, and other file types) organized as a hierarchy of subject categories that can be limited to one server or span multiple Gopher servers. Veronica was created in 1993 at the
University of Nevada. Soon another user interface name Jughead appeared with the same purpose as Veronica. Both of these were used for files sent via Gopher, which was created as an Archie alternative by Mark McCahill at the University of Minnesota in 1991.


In response to the problems with automated indexing of the Web, Martijn Koster in Oct. 1993 created ‘Aliweb’ [26], which stands for ‘Archie Like Indexing for the Web’. This was the first attempt to create a directory for just the Web. Instead of a robot, webmasters submit a file with their URL and their own description of it. This allowed for a more accurate, detailed listing. Unfortunately, the application file was difficult to fill out so many websites were never listed with Aliweb.


The first browsable Web directory was ‘ElNet Galaxy’, now known as ‘Tradewave Galaxy’ [27], which went online January 1994. It made good use of categories and subcategories and so on. Users could narrow their search until presumably they found something that caught their eye. It still exists today and offers users the opportunity to help coordinate directories, becoming an active participant in cataloging the Internet in their field.

‘Yahoo’ [27] grew out of two Stanford University students, David Filo’s and Jerry Yang’s, web-pages with their favorite links (such pages were quite popular back then). Started in April 1994 as a way to keep track of their personal interests, Yahoo soon became too popular for the university server. Yahoo’s user-friendly interface and easy to understand directories have made it the most used search directory. But because everything is reviewed and indexed by people, their database is relatively small and hence accounting for approximately 1% of web-pages.
5. The Big-Guns (1995)

When a search fails on Yahoo it automatically defaults to ‘AltaVista’s’ search [28]. AltaVista was late onto the scene in December 1995, but made up for it in scope. AltaVista was not only big, but also fast. It was the first to adopt natural language queries as well as Boolean search techniques. And to aid in this, it was the first to offer "Tips" for good searching prominently on the site. These advances made for unparalleled accuracy and accessibility.


The next important step in search engines is the rise of meta-engines. Essentially nothing new was offered by them. They just simultaneously compile search results from various different search engines. Then list the results according to the collective relevancy. The first meta-engine was MetaCrawler released in 1995. Now called Go2net.com [29], it was developed in 1995 by Eric Selburg, a Masters student at the University of Washington.


‘Google’ [30] began in March 1996 as a PhD research project by Larry Page and Sergey Brin, students of Stanford University. They began working on BackRub, a search engine which utilized backlinks for search. It ranked pages using citation notation which meant that any mention of a web site on another site would be counted as a vote towards the mentioned site. The website’s reliability came from how many people linked to that site and how trustworthy the linking sites were.

Today Google is one of the most sought after search engines. Google Sites are ranked as the top search property worldwide [22] with 87.8 billion searches in December, or 66.8 percent of the global search market. Google Sites achieved a 58-percent increase in search query volume over the past year.

Prior to ‘Direct Hit’ [31], launched in the summer of 1998, there were two types of search engines:

- Author controlled services, such as AltaVista and Excite, in which the results are ranked by keyword relevancy and

- Editor-controlled, such as directories like Yahoo and LookSmart, in which people manually decide on placement.

Direct Hit, as inventor Gary Culliss relates: "represents a third kind of search, one that's user-controlled, because search rankings are dependent on the choices made by other users."

As users choose to go to a listed link, they keep track of that data and use the collected hit-ratio to calculate the relevancy. So ‘the more people go to the site from Direct Hit, the higher it will appear on their results.’

As the Web continues to grow rapidly, the need for better search engines only increases.

The next subsection discusses the working of the search engine in detail.

2.2.2 WORKING OF SEARCH ENGINES

A search engine [32, 33, 34, 35, 36 and 37] is the popular term for an information retrieval (IR) [38] system. It allows the users to ask for content in the form of a query consisting of words/phrases and it retrieves a list of references that contain those keywords/phrases. For this to happen, it uses regularly updated indexes where an index consists of the words contained in document, plus pointers to their locations within the documents. In fact, the index is generally organized as an inverted file. Fig. 2.2 shows the search engine architecture. A search engine or IR system comprises five essential modules:
A. Crawler

The crawler [39] is a program that traverses the WWW in a meticulous manner in order to provide up to date data in the form of web pages to a search engine. The crawling process starts with the seed URLs that the search engine provides. The crawler then visits each web page and identifies all the hyperlinks on the page, adding them to the list of pages to crawl. Different search engines follow different policies to crawl the web.

![Search engine Architecture](image)

Fig. 2.2 Search engine Architecture
**Crawling Policies:** There are three important characteristics of the web due to which crawling is difficult. They are:

- Heavy Internet traffic.
- Fast rate of change of contents of documents.
- Dynamic Page generation.

Therefore, search engines have to prioritize the web pages to be crawled. This prioritization can be summed up in four different policies:

- *The selection policy:* This policy identifies the pages to be downloaded.
- *The re-visit policy:* This policy sets the re-visit frequency for the crawlers i.e. when to check web pages for changes.
- *The politeness policy:* This policy formulates the guidelines so as to how to avoid overloading of websites by its crawler(s).
- *The parallelization policy:* This policy states as to how to co-ordinate among the distributed web crawlers of the search engines with a view to reduce duplication and redundancy.

The web pages collected by the crawler are indexed by another component called ‘indexer’. Indexing is an important part of IR wherein an index is built over a collection of documents. Typically, an inverted index is the reference structure for storing indexes in Web full-text IR systems. [40] Depending upon how the index is organized, it may also contain information on how to efficiently access the index. This meta-information is generated during processing of the documents as explained below:

**B. Document Processor**

The document processor performs some or all of the following steps:

1. Normalizes the document stream to a predefined format.
2. Breaks the document stream into desired retrievable units.  
3. Isolates and metatags subdocument pieces.  

} Preprocessing
4. Identifies potential indexable elements in documents.
5. Deletes stop words.
7. Extracts index entries.
8. Computes weights.
9. Creates and updates the main inverted file against which the search engine searches in order to match queries to documents.

Steps 1-3: Preprocessing - It [41] prepares the document into a single consistent data structure. The need for a well-formed, consistent format is of relative importance in direct proportion to the sophistication of later steps of document processing. These first three steps are very important because they simply standardize the multiple formats of the documents encountered while deriving them from various servers or web sites.

Step 4: Identify elements to index - The quality of the search carried out by the search engines can drastically improve if the web documents are properly indexed [42]. Each search engine depends on a set of rules that its document processor must execute to determine what action is to be taken by the "tokenizer". The tokenizer extracts terms called tokens suitable for indexing.

Step 5: Deleting stop words: Stop words are break terms that have little value in finding useful documents in response to a user's query. For example, ‘a’, ‘an’, ‘the’, ‘of’, etc are stop words. Deleting stop words help save system resources and time spent in their processing and potential matching. However, in today’s era, when memory has become so much cheaper and systems so much faster, it does not effect much, but since stop words may comprise up to 40 percent of text words in a document, it still has some significance as far as processing and match time is concerned.

Step 6: Term Stemming: This process [43] removes word suffixes, perhaps recursively in layer after layer of processing. The process has two goals.
1. The main objective of stemming is to reduce the number of unique words present in the index, which in turn reduces the storage space required for the index and hence speeds up the search process.

2. As far as *effectiveness* is concerned, stemming improves *recall* by reducing all forms of the word to a base or stemmed form. Recall is the measure for evaluating the correctness of the pattern recognition system. It can be computed as the fraction of correct instances among all instances that *actually* belong to the relevant subset. For example, if a user asks for *analyze*, they may also want documents which contain *analysis*, *analyzing*, *analyzer*, *analyzes*, and *analyzed*. Therefore, the document processor stems document terms to *analy*- so that documents which include various forms of *analy*- will have equal likelihood of being retrieved. This would not occur if the engine only indexed variant forms separately and required the user to enter all.

However, stemming does have a drawback in the sense that it may negatively affect *precision* wherein all forms of a stem will match. Precision is the fraction of correct instances among those that the algorithm *believes* to belong to the relevant subset.

*Step 8: Term weight assignment:* Weights are assigned to terms in the index file. The simplest of search engines just assign a binary weight - 1 for presence and 0 for absence. The more sophisticated the search engine, the more complex the weighting scheme. Extensive experience in information retrieval research over many years has clearly demonstrated that the optimal weighting comes from use of “tf/idf” which the algorithm uses to measure the frequency of occurrence of each term within a document. Thereafter, it compares the computed frequency against the frequency of occurrence in the entire database. The “tf/idf” weighting scheme assigns higher weights to those terms that really distinguish one document from the others.

*Step 9: Create index:* The index or inverted file is the internal data structure that stores the index information [44] and is consulted for each query. Inverted files range from a simple listing of every alpha-numeric sequence to a more linguistically complex list of
entries such as ‘tf/idf’ weights, and pointers to where inside each document the term occurs. The more complete the information in the index, the better the search results.

C. Query and Result Processor

The goal of this component [45] is to analyze the human language to identify the context of the searcher’s intent in order to return the most relevant set of results. Document processing shares many steps with query processing. More steps and more documents make the process more expensive for processing in terms of computational resources and responsiveness. However, the longer the wait for results, the higher is the quality of results. Thus, search system designers must choose what is most important to their users - time or quality. Publicly available search engines usually choose time over very high quality. The reason being they have too many documents to download and to process.

The steps followed in query and result processing are as:

As can be seen from Fig. 2.3, the task of search engine begins by the time user enters the query into its search interface. This query then moves into the Query Processing Module which performs the following tasks:

1. **Tokenizing and Parsing:** Tokenizing is the process of breaking a query into understandable segments. These are also known as tokens. A token is defined as an alpha-numeric string that occurs between white space (and/or) punctuation. As soon as a user inputs a query, the search engine must tokenize the query stream. Parsing forms the next important step. Since users may employ special operators in their query including Boolean, adjacency, or proximity operators, the system needs to parse the query into query terms and operators. The result of this step is the combination of tokens and the stop-words which are then treated further.

2. **Stop-word listing and Query Stemming:** This is the next step taken by the ‘Query Processor Module’ wherein it deletes stop-words from the query. Since most publicly available search engines encourage very short queries,
the engines may drop these two steps. The output of this step is the refined intermediate query.

3. **Query expansion:** It becomes highly probable that the information required by the user could be expressed using synonyms rather than the exact query terms. Therefore, more sophisticated systems may expand the query into all possible synonymous terms and perhaps even broader and narrower terms.
4. **Query term weighting**: (assuming more than one query term). The final step in query processing involves computing weights for the terms in the query. Sometimes the user controls this step by indicating either how much to weight each term or simply which term or concept in the query matters most and *must* appear in each retrieved document to ensure relevance. But this criterion is not commonly used as research has shown that users are not particularly good at determining the relative importance of terms in their queries [46]. They can't make this determination for several reasons as:

   a. First, they don't know what else exists in the database and document terms are weighted by being compared to the database as a whole.
   b. Second, most users seek information about an unfamiliar subject, so they may not know the correct terminology.

Few search engines implement system-based query weighting [47], but some do an implicit weighting by treating the first term(s) in a query as having higher significance. The engines use this information to provide a list of documents/pages to the user.

After this final step, the search engine consults the index for a given query. The matched query is then expanded and weighted against the indexed documents. The raw results of the search process are reported to the ‘Results-Processing Module’ wherein the duplicates are removed. Even the results from different search nodes are merged followed by the sorting and ranking of the new results. The next subsections explain the searching, matching and ranking functions performed by the search engine.

**D. Search and Matching**

Searching the inverted file for documents that meets the query requirements is referred to as "matching" [48]. It is typically a standard binary search. Once the documents that match the query issued by the user are determined, a similarity score is computed between the query and each document. This score is based on the scoring algorithm used by the system. Scoring algorithms [49] ranking is based on the presence/absence of query
term(s), term frequency, tf/idf, or query term weights. Some search engines use scoring algorithms based on relations among documents or past retrieval history of documents/pages rather than on the content of the document.

The next step is to rank these documents and the list of such documents is then presented to the user. The user now simply clicks and follows the system's internal pointers to reach to the selected document/page.

More sophisticated systems rather help the user iterate i.e. to provide some relevance feedback to modify their query based on the earlier results. Accordingly, the system adjusts its query representation to reflect this value-added feedback helping the searching module re-run with the improved query to produce more relevant set of documents. In certain cases, re-ranking of documents may be carried out.

**E. Ranking**

A Ranking algorithm [50] locates the keywords in a web document/page. Based on certain criteria, the rank of the web pages is computed. For instance, if the search keywords appear near the top of a web page, such as in the headline or in the first few paragraphs of text, it is assumed that the page is more relevant to the topic.

Frequency is other major factor in the determination of relevancy. A search engine analyzes as to how often keywords appear in relation to other words in a web page. A page with higher frequency of keywords is considered more relevant than other web pages.

No two search engines perform ranking in similar manner because each one may adopt a different criteria for ranking of pages. For instance, some search engines may collect more web pages and then index while others may index web pages more frequently with lesser web pages each time. Therefore, no search engine has the exact same collection of web pages to search through. These naturally produce differences while comparing their results.
Unfortunately, search engines’ designers have to be wary of web masters who try to misuse its ranking criteria in order to promote the ranking of certain websites for vested interests. Therefore they have to constantly keep changing their ranking methods. Following are few ways in which the web masters and search engines counteract each other:

- **Misuse:** Web masters may introduce ‘spamming’. It is a technique wherein a word is repeated hundreds of times on a page in order to increase the frequency and propel the page higher in the listings.

  *Action:* Search engines may penalize such pages or exclude them from the index if they detect ‘spamming’.

- **Misuse:** They may "reverse engineer" the location/frequency criteria used by a particular search engine.

  *Action:* Because of this, all major search engines have started making use of "off the page" ranking criteria. Off the page factors are those that a webmasters cannot easily influence.

  - Chief among them is the ‘link analyses’. By analyzing how pages link to each other, a search engine can both determine what a page is about and whether that page is ‘important’ and if it really deserves a boost in its ranking. In addition, sophisticated techniques are used to screen out attempts by webmasters to build ‘artificial links’ designed to boost their rankings.

  - Another off the page factor is ‘click-through measurement’. In this method, a search engine may watch what pages someone selects for a particular search. It then eventually drops high-ranking pages that aren’t attracting clicks while promoting lower-ranking pages that do pull in visitors.
How efficiently the search engines perform their task depend on how efficiently the crawlers have collected the documents from the WWW. Even after applying the most sophisticated methods, search engines provide the users with huge list of documents for their submitted query. This leads to the problem of ‘information overkill’. To minimize this problem, automated techniques/tools need to be developed that filter/prune the large list to obtain the desired information resources. Moreover, it may track and analyze their usage patterns also. Of course, these usage patterns can be found with the help of Data mining.

Data mining is primarily used today by companies with a strong consumer focus - retail, financial, communication, and marketing organizations. It enables these companies to determine relationships among "internal" factors such as price, product positioning, or staff skills, and "external" factors such as economic indicators, competition, and customer demographics. And, it enables them to determine the impact on sales, customer satisfaction, and corporate profits. Conclusively, it enables them to "drill down" into summary information to view detail transactional data.

With data mining, a retailer could use point-of-sale records of customer purchases to send targeted promotions based on an individual's purchase history. By mining demographic data from warranty cards, the retailer could develop products and promotions to appeal to specific customer segments.

For example, Shopper Stop can mine its online shopping database to recommend special offers on clothing to its customers. HDFC bank can suggest products to its credit cardholders based on analysis of their monthly expenditures.

A brief discussion on data mining is given in next section.

2.3 DATA MINING

Data mining [51, 52] is the process of extracting patterns from large data sets by combining methods from statistics and artificial intelligence with database management. Data mining tools predict future trends and behaviours, allowing businesses to make
proactive, knowledge-driven decisions. It allows users to analyze data from many different dimensions, categorize it, and summarize the relationships identified. Technically, *data mining is the process of finding correlations or patterns among dozens of fields in large relational databases.*

Data mining techniques analyze relationships and patterns in stored transaction data based on user queries. Generally, any four of the following types of relationships are sought:

1. *Classes:* Stored data is used to locate data in predetermined groups. For example, a restaurant chain could mine customer purchase data to determine when customers visit and what they typically order. This information could be used to increase traffic by having daily specials.

2. *Clusters:* Data items are grouped according to logical relationships or consumer preferences. For example, data can be mined to identify market segments or consumer affinities.

3. *Associations:* Data can be mined to identify associations. The market-basket example is an example of associative mining.

4. *Sequential patterns:* Data is mined to anticipate behavior patterns and trends. For example, an online travel portal can predict the likelihood of hotel reservation by the user on the purchase of an air-ticket to a particular destination.

Data mining consists of five major steps as shown in Fig 2.4. A brief discussion on each data mining steps is given below:

1. Extract and transform raw data into preprocessed data after performing data cleaning steps. Data extraction and data cleaning are guided by adapting good data lifecycle management policies.

2. Store and manage the data in a multidimensional data warehouse system.
3. Prepare data for data modelling and perform data modelling through either of these techniques:

- **Artificial neural networks:** Non-linear predictive models that learn through training and resemble biological neural networks in structure.
- **Genetic algorithms:** Optimization techniques that use process such as genetic combination, mutation, and natural selection in a design based on the concepts of natural evolution.
- **Decision trees:** Tree-shaped structures that represent sets of decisions. These decisions generate rules for the classification of a dataset.
- **Classification:** A technique that classifies each record in a dataset based on a combination of the classes of the $k$ record(s) most similar to it in a historical dataset (where $k=1$). It is also known as $k$-nearest neighbour technique.
- **Rule induction:** The extraction of useful if-then rules from data based on statistical significance.
• Data visualization: The visual interpretation of complex relationships in multidimensional data. Graphics tools are used to illustrate data relationships.

4. Finally, the patterns obtained after data modelling are converted into knowledge and can be represented in the form of a graph or a table.

The data operated upon by data mining techniques is structured and is presented by well-defined tables, rows, columns, keys and constraints whereas the data available on web is rich and dynamic in features and patterns. Thus, data mining techniques operating on WWW needs to be modified/adjusted for evolving usage patterns in real time and therefore, are termed as WEB MINING techniques. Web mining is the extraction of interesting and potentially useful patterns and implicit information from the activity related to the WWW. A survey of web mining and its taxonomy is presented in the next subsection.

2.3.1 WEB MINING

It has become increasingly necessary for users to utilize automated tools to find the desired information resources and to track and analyze their usage patterns. This has given rise to the necessity of creating server-side, proxy-side and client-side intelligent systems that can effectively mine valuable knowledge. One of the key steps in Knowledge Discovery in Databases (KDD) [53] is to create a suitable target data set for web mining tasks. Each type of data collection differs not only in the location of the data source but also in the segment of users from whom the data is collected, the nature of data composed and in their methods of implementations.

Web Mining can be broadly divided into three distinct categories- web content mining, web structure mining and web usage mining according to the kinds of data to be mined [54] as shown in Fig. 2.5.

• Web content mining: It is the automatic search of information resources available online.
- **Web Structural Mining:** It generates the structural summary for the web sites and web pages.
- **Web Usage Mining:** It is the discovery of user access patterns from Web servers.

1. **Web Content Mining:** Web Content Mining [55] is the process of extracting useful information from the contents of Web documents. Content data corresponds to the collection of facts that a web page was designed to communicate to the users. It may consist of text, images, audio, video, or structured records such as lists and tables. Text mining and its application to web content has been the most widely researched issue. Some of the research issues addressed in text mining are topic discovery, extracting association patterns, clustering of web documents and classification of Web Pages.

Web content mining studies can be divided into two main approaches [56]:

a. *Agent based approach and*

b. *Database approach* as shown in fig 2.4.

a. Agent-based web mining systems can be placed into three categories:

i. *Intelligent Search Engines* uses domain characteristics and user profiles to organize and interpret the discovered information such as Harvest [57], Parasite [58] and Shop-Boot [59].

ii. *Information Filtering/Categorization* uses various information retrieval techniques [60] and characteristics of open web documents to automatically retrieve filter and categorize them.

iii. *Personalized Web Agents* learn user preferences and discover web information sources based on their preferences and those of other individuals with similar interests such as WebWatcher [61] Sykill & Webert [62].

b. The database approaches to web mining organize semi-structured web pages into more structured collections of resources.
2. **Web Structure Mining:** The structure of a typical Web graph [63] consists of Web pages as nodes, and hyperlinks as edges connecting between two related pages. Web Structure Mining can be regarded as the process of discovering structural information from the Web. This type of mining can be further divided into two types based on the kind of structural data used - hyperlinks and document structure.
a. **Hyperlinks**: A Hyperlink is a structural unit that connects a Web page to different location, either within the same webpage or to a different webpage. A hyperlink that connects to a different part of the same page is called an **Intra-Document Hyperlink**, and a hyperlink that connects two different pages is called an **Inter-Document Hyperlink**. There has been a significant body of work on hyperlink analysis, of which Chakrabarty et al [64] provides an up-to-date survey.

b. **Document Structure**: In addition, the content within a webpage can also be organized in a tree-structured format, based on the various HTML and XML tags within the page. Mining efforts have focused on automatically extracting document object model (DOM) structures out of documents. Most research on the web structure mining can be thought of a mixture of content and structure mining and add content information to the link structures such as Clever System [65] and Google [66].

3. **Web Usage Mining**: It is the application of data mining techniques [67, 68] to discover interesting usage patterns from web data, in order to understand and better serve the needs of Web-based applications. Usage data captures the identity or origin of Web users along with their browsing behaviour at a Web site e.g. IP addresses, page references, and the date and time of references. The usage data can also be split into three different kinds on the basis of the source of its collection: the server side, the client side, and the proxy side as described in fig. 2.5.

An overview of the web mining categories is given in Table 2.1. It may be noted that among the different classifications of web mining, web usage mining is the most important for the different organizations as it helps them to:

- Determine the life-time value of the clients,
- Determine the cross-marketing strategies,
- Optimize the functionality of web based applications,
• Provide more personalized content to the visitors etc.

Table 2.1: Web Mining Categories

<table>
<thead>
<tr>
<th>Web mining</th>
<th>Web content mining</th>
<th>Web Structure mining</th>
<th>Web usage mining</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IR View</td>
<td>DB View</td>
<td></td>
</tr>
<tr>
<td>View of Data</td>
<td>-Unstructured</td>
<td>-Semi structured</td>
<td>-Links Structure</td>
</tr>
<tr>
<td></td>
<td>-Semi structured</td>
<td>-Website as DB</td>
<td>-Interactivity</td>
</tr>
<tr>
<td>Main Data</td>
<td>-Text documents</td>
<td>-Hypertext Documents</td>
<td>-Links Structure</td>
</tr>
<tr>
<td></td>
<td>-Hypertext documents</td>
<td></td>
<td>-Server logs</td>
</tr>
<tr>
<td>Representation</td>
<td>-Bag of words, n-gram</td>
<td>-Edge-labelled-graph(OEM)</td>
<td>-Browser Logs</td>
</tr>
<tr>
<td></td>
<td>-Terms and Phrases</td>
<td>-Relational</td>
<td></td>
</tr>
<tr>
<td></td>
<td>-Concepts or Ontology</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>-Relational</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Method</td>
<td>-TF/IDF and variants</td>
<td>-Proprietary Algos</td>
<td>-M/C Learning</td>
</tr>
<tr>
<td></td>
<td>-Machine learning</td>
<td>-Association Rules</td>
<td>-Statistical</td>
</tr>
<tr>
<td></td>
<td>-Statistical(including NLP)</td>
<td></td>
<td>Association Rules</td>
</tr>
<tr>
<td>Application Categories</td>
<td>-Categorization</td>
<td>-Finding Frequent</td>
<td>-Site Construction,</td>
</tr>
<tr>
<td></td>
<td>-Clustering</td>
<td>-Substructures</td>
<td>Adaptation,</td>
</tr>
<tr>
<td></td>
<td>-Finding Extraction Rules</td>
<td>-Website</td>
<td>Management</td>
</tr>
<tr>
<td></td>
<td>-Finding Patterns in text</td>
<td>-Schema</td>
<td>-Marketing</td>
</tr>
<tr>
<td></td>
<td>-User modelling</td>
<td>-Discovery</td>
<td>-User Modeling</td>
</tr>
</tbody>
</table>

The goal of the web usage mining is to capture, model, and analyze the behavioral patterns and profiles of the users who interact with the web sites. Thus web usage mining is a critical issue that needs to be addressed. The next section attempts to survey the web usage mining process in detail.

2.4 WEB USAGE MINING PROCESS

Web usage mining process is carried out in following three phases as shown in Fig. 2.6.
1. Preprocessing Phase
   a. Data Cleaning
   b. User and Session Identification
   c. Path Completion
2. Pattern Discovery Phase - Mining the Data
3. Pattern Analysis Phase - Recommending URLs/Presenting Analysis results

The first phase is mostly identical in all systems and is known as the preprocessing phase but the mining and recommendation phases can be designed differently in different systems.

1. **Preprocessing Phase:**

Preprocessing converts the usage, content and structure information contained in the various available data sources into the data abstractions necessary for pattern discovery. The adequacy of this phase enables the mining algorithms to provide reliable results for pattern analysis. However, usage preprocessing is a difficult task in web usage mining as it is of incomplete nature.

Fig. 2.6 Web Usage mining process
a. Data cleaning

Web server and proxy logs contain millions of entries out of which all the entries in a log file are not necessary. This large search space must be reduced to create an effective result space. This can be done by data cleaning which is performed by the following two steps:

- **By removing all the image files:** To get an idea of user behavior, it is necessary to keep track of the web pages that the user specifically requested. A log file marks an entry for every user-requested web page. Each web page may consist of several components like text, pictures etc. whereas it would be useful if we could keep only a single entry about the page in the log for each page view. Therefore, all entries of web pages in the log which contains images or audio (like .jpg, .gif, .wav, etc.) are removed. However, it suffers from a major drawback i.e. sometimes users specifically want to see a picture on a separate page which will be deleted by this method.
- **By removing the records with the failed HTTP status code:** By examining the Status field of every record in the web access log, the records with status codes above 299 or under 200 are removed.

b. User and session identification

The task of user and session identification is to find out the different user sessions from the original web access log. User’s identification is, to identify who accessed web site and which pages were accessed. The goal of session identification is to divide the page accesses of each user at a time into individual sessions. A session is a series of web pages user browse in a single access. The difficulties to accomplish this step are introduced by using proxy servers, e.g. different users may have same IP address in the log. A referrer-based method can be used to solve these problems. The user sessions can be distinguished by adopting following rules:

- The different IP addresses distinguish different users.
• If the IP addresses are same, the different browsers and operation systems indicate different users [69].

• If all of the IP address, browsers and operating systems are same, the referrer information should be taken into account. The Refer URI (Uniform Resource Identifier) field is checked, and a new user session is identified if the URL in the Refer URI field hasn’t been accessed previously, or there is a large interval (usually more than 10 seconds) between the accessing time of this record and the previous one if the Refer URI field is empty [70].

• The session identified by rule 3 may contains more than one visit by the same user at different time, the time oriented heuristics is then used to divide the different visits into different user sessions. After grouping the records in web logs into user sessions, the path completion algorithm should be used for acquiring the complete user access path. Table 2.2 shows the sample log.

Consider table 2.2 which shows the sample log. It may be noted that the IP address 123.456.78.9 is responsible for three server sessions while IP address 209.654.76.9 is responsible for the fourth session.

It may be further noted that the following three sessions have been formed using the combination of referrer and agent field:

1. A-B-F-O-G,
2. L-R, and
3. A-B-C-J.

A critical look at the sample log further indicates that from the entries consisting of the IP address 209.654.76.9 (i.e. the last two lines), it is not possible to determine that this particular IP address belongs to one session. However, using cookies or embedded session ID or a client side tracking can overcome this difficulty.
### Table 2.2: Sample Log

<table>
<thead>
<tr>
<th># IP Address</th>
<th>User id</th>
<th>Time</th>
<th>Method/ URL/ Protocol</th>
<th>Status</th>
<th>Size</th>
<th>Referrer</th>
<th>Agent</th>
</tr>
</thead>
<tbody>
<tr>
<td>123.456.78.9</td>
<td>-</td>
<td>[25/Apr/2006:03:04:41</td>
<td>&quot;GET A.html HTTP/1.0&quot;</td>
<td>200</td>
<td>3290</td>
<td>-</td>
<td>Mozilla/3.04 (Win98, I)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-0500]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>123.456.78.9</td>
<td>-</td>
<td>[25/Apr/2006:03:05:34</td>
<td>&quot;GET A.html HTTP/1.0&quot;</td>
<td>200</td>
<td>2050</td>
<td>A.html</td>
<td>Mozilla/3.04 (Win98, I)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-0500]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>123.456.78.9</td>
<td>-</td>
<td>[25/Apr/2006:03:05:39</td>
<td>&quot;GET A.html HTTP/1.0&quot;</td>
<td>200</td>
<td>4130</td>
<td>-</td>
<td>Mozilla/3.04 (Win98, I)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-0500]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>123.456.78.9</td>
<td>-</td>
<td>[25/Apr/2006:03:06:02</td>
<td>&quot;GET A.html HTTP/1.0&quot;</td>
<td>200</td>
<td>5096</td>
<td>B.html</td>
<td>Mozilla/3.04 (Win98, I)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-0500]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>123.456.78.9</td>
<td>-</td>
<td>[25/Apr/2006:03:07:42</td>
<td>&quot;GET A.html HTTP/1.0&quot;</td>
<td>200</td>
<td>3290</td>
<td>-</td>
<td>Mozilla/3.04 (X11, I, IRIX6.2, IP22)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-0500]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>123.456.78.9</td>
<td>-</td>
<td>[25/Apr/2006:03:07:55</td>
<td>&quot;GET A.html HTTP/1.0&quot;</td>
<td>200</td>
<td>250</td>
<td>A.html</td>
<td>Mozilla/3.04 (X11, I, IRIX6.2, IP22)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-0500]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>123.456.78.9</td>
<td>-</td>
<td>[25/Apr/2006:03:09:50</td>
<td>&quot;GET A.html HTTP/1.0&quot;</td>
<td>200</td>
<td>8140</td>
<td>L.html</td>
<td>Mozilla/3.04 (Win98, I)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-0500]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>123.456.78.9</td>
<td>-</td>
<td>[25/Apr/2006:03:10:02</td>
<td>&quot;GET A.html HTTP/1.0&quot;</td>
<td>200</td>
<td>1820</td>
<td>A.html</td>
<td>Mozilla/3.01 (X11, I, IRIX6.2, IP22)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-0500]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>123.456.78.9</td>
<td>-</td>
<td>[25/Apr/2006:03:10:45</td>
<td>&quot;GET A.html HTTP/1.0&quot;</td>
<td>200</td>
<td>2270</td>
<td>F.html</td>
<td>Mozilla/3.04 (Win98, I)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-0500]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>123.456.78.9</td>
<td>-</td>
<td>[25/Apr/2006:03:12:23</td>
<td>&quot;GET A.html HTTP/1.0&quot;</td>
<td>200</td>
<td>9430</td>
<td>C.html</td>
<td>Mozilla/3.01 (X11, I, IRIX6.2, IP22)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-0500]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>123.456.78.9</td>
<td>-</td>
<td>[25/Apr/2006:03:12:33</td>
<td>&quot;GET A.html HTTP/1.0&quot;</td>
<td>200</td>
<td>7220</td>
<td>B.html</td>
<td>Mozilla/3.04 (Win98, I)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-0500]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>209.654.76.9</td>
<td>-</td>
<td>[25/Apr/2006:05:02:23</td>
<td>&quot;GET A.html HTTP/1.0&quot;</td>
<td>200</td>
<td>3290</td>
<td>-</td>
<td>Mozilla/3.04 (Win98, I)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-0500]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>209.654.76.9</td>
<td>-</td>
<td>[25/Apr/2006:05:06:03</td>
<td>&quot;GET A.html HTTP/1.0&quot;</td>
<td>200</td>
<td>1680</td>
<td>A.html</td>
<td>Mozilla/3.04 (Win98, I)</td>
</tr>
</tbody>
</table>

### c. Path completion

Path is the access path of every user recorded in the user session identified from the web log. Path incompletion may occur because of many reasons. For instance, local cache, agent cache, ‘post’ technique, browser’s ‘back’ button and use of proxy server can result in some important accesses not recorded in the access log file. As a result of this, the number of Uniform Resource Locators (URLs) recorded in log may be less than the real one. Thus, the user access paths are incompletely preserved in the web access log. To discover user’s travel pattern, the missing pages in the user access path should be appended. The purpose of the path completion is to accomplish this task.
2. Pattern Discovery

Pattern Discovery is the key component of the Web mining which is an amalgamation of the algorithms and techniques from data mining, machine learning, statistics and pattern recognition etc research categories. A brief discussion on pattern discovery is given below:

a. Log file analysis

It is the most widely used technique to obtain structured information out of the unstructured information present in server logs. There are number of tools in the market that accept the most common log file formats as input to provide [71] information such as the number of hits and page views, the number of unique and returning users, the average length of a page view, an overview of the browsers and operating systems that were used etc. In fact, this type of knowledge can be potentially useful for improving the system performance, enhancing the security of the system, facilitating the site modification task, and providing support for marketing decisions

b. Association rules

In web usage mining, association rules [72, 73, and 74] are used to find out which pages are frequently visited together in a single server session with a view to discover which websites and which sectors are frequently visited together. These pages may not be directly linked to one another via hyperlinks.

An association rule is usually presented in the following syntax:

GoogleCom <= GmailCom & OrkutCom (15: 2.788%, 0.27)

The above rule indicates that the association of the websites ‘www.gmail.com’ and ‘www.orkut.com’ with ‘www.google.com’ has support of 15 and confidence of 27%. In simple words, we can say that 27% of the users who visited ‘www.gmail.com’ and ‘www.orkut.com’ also visited the website called ‘www.google.com’.
Early systems used collaborative filtering for user recommendation and personalization. Bamshad Mobasher and colleagues [75] used association-rule mining based on frequent item sets and introduced a data structure to store the item sets. They split Web logs into user sessions and then mined these sessions using their suggested association rule algorithm. They argue that other techniques based on association rules for usage data do not satisfy the real-time constraints of recommender systems because they consider all association rules prior to making a recommendation. Ming-Syan Chen and colleagues [76] proposed a somewhat similar approach that uses a different frequent itemset counting algorithm.

c. Sequential patterns

This technique is similar to the association rules with the difference that it also takes the time dimension into account. It tries to find sequences in which a certain page (or Web site) usually comes before or after another page (or Web site). In other words, it “attempts to find inter-session patterns such that the presence of a set of items is followed by another item in a time-ordered set of sessions or episodes” [77].

d. Clustering

In general, clustering is a process of creating a partition such that all the members of each set of the partition are similar according to some metric [78]. In relation to web usage mining, the definition can be narrowed to a technique to group users in clusters based on their common characteristics like browsing pattern, keyword selection etc. Clustering algorithms learn in an unsupervised way wherein their own classes and subsets of related objects in the training set are discovered. Clustering of pages helps in grouping web pages with related content, information much useful for internet search engines and web assistance providers.

e. Classification

The main objective of classification in web domain is to develop a profile of users belonging to a particular class or category. Contrary to clustering, classification is a
supervised way of learning wherein the data items are mapped into one of the several predefined classes [79]. It can be done by using supervised inductive learning algorithms such as decision tree classifiers, naïve Bayesian classifiers, k-nearest neighbor classifiers, support vector machines etc. For example, classification on server logs may lead to the discovery of interesting rules such as: 48% of the users, who visit two or more sites of television stations in a single session, are younger than 22 years.

3. Pattern Analysis

Pattern analysis is the last step of the web usage mining process as shown in Fig. 2.6. The motivation behind pattern analysis is to filter out uninteresting rules or patterns from the set found in the pattern discovery phase. The most common form of pattern analysis consists of knowledge query mechanism such as SQL. Another method is to load the usage data into a data cube in order to perform OLAP (OnLine Analytical Processing) operations. Visualization techniques, such as graphing patterns or assigning colors to different values, can often highlight the overall patterns or trends in the data. Content and structure information can be used to filter out patterns containing pages of a certain usage type, content type, or pages that match a certain hyperlink structure. Yannis Manolopoulos and colleagues [80] provide a comprehensive discussion of Web logs for usage mining and suggest novel ideas for Web log indexing.

It can be concluded that web mining is an area which has always been on the hit list of researchers. They have been consistently working on applying web mining techniques for:

- **E-businesses** [81, 82].
- **Web personalization** [83] which is the task of making web-based information system adaptive to the needs and interests of individual users or groups of users.
- **Site modification and improvement** [84] which is an attempt to attract the users and
- **Usage characterization** [85] is another technique that basically deals with analyzing users for their interest patterns on the web.

The next section is an attempt to survey the various recent web mining applications.
2.4.1 RECENT WEB MINING APPLICATIONS

2.4.1.1 Robot Detection

Web Robot is software that navigates the hyperlink structure of the WWW for fetching and finding information. Christian Bomhardt and colleagues [86] have focused their research for differentiating robot behaviour from human behaviour by determining the navigational patterns from the click-stream data. E-commerce retailers are particularly concerned about the unauthorized deployment of web-robots for finding business intelligence at their Web sites. Also, Web robots tend to consume significant amount of network bandwidth at the expense of other users. Even the sessions generated by web robots make it more difficult to perform click-stream analysis effectively on the Web data. Identifying the IP address and user agent of the web clients are two prevalent methods for identifying web-robots. While these techniques are applicable to many well-known robots, they may not be enough to detect previously unknown robots.

2.4.1.2 Extracting User Profiles

User profiles can be defined as the representation of the knowledge about the user’s interesting information. This information is important as it is used to improve e-commerce. In an e-commerce site, the complete click-stream information of the user is recorded which gives the detailed record of every single action performed by him/her. This greatly improves the decision making process carried by the web site owners.

Adding behavioural information to the demographic and psychographic information about the user contributes in building more comprehensive user profile which can be used for many different applications as suggested by B. Masand and colleagues in [87]. Many organizations build web user profiles based on visits to their own sites, e.g. Alexa Research and DoubleClick are successful examples of building WWW user behaviour profiles. These approaches need browser cookies of some sort, which provides a fairly detailed view of a user’s browsing behaviour across the Web.
2.4.1.3 Finding Significant Pages on the Web

Two or more pages can be significant if their contents are similar or closely related to ones of higher-ranked pages. Since as a user, we are usually careless of pages with lower ranks, they are unconditionally discarded even if their contents are similar to some pages with high ranks. Such hidden pages thus must be extracted and bound with the significant higher ranked pages. This again applies web usage mining techniques like clustering. In order to obtain such clusters, Yoshiaki Okubo and colleagues [88] proposed a mechanism to find the semantic correlations among terms by applying Singular Value Decomposition (SVD) to the term-document matrix generated from a corpus with respect to a specific topic. Based on the correlations, potential similarities are evaluated among web pages from which clusters are obtained. The set of web pages is represented as a weighted graph $G$ based on the similarities and their ranks.

2.4.1.4 Google: A Search Engine Application

Today, Google is one of the most popular and widely used search engines. Google provides web users with information from more than 2.5 billion web pages that it has indexed on its server. Compared to other search engines, google has a simple and quick search facility. This property makes it the most widely used search engine. Previous search engines based their searching methodology on the web content in order to fetch a web page as a result of submitted query. However, Google is first engine to reflect importance of web structure namely link analysis from the web. The key method of google called PageRank, measures importance of a page, and is the underlying technology in all search products. PageRank method makes use of the information about link structure of the Web graph. This method plays significant role for returning relevant results to simple query.

Web content, especially text and hypertext data is the core data source that google uses. Web graphs contributed to improve google’s search capability and provide better results for web users. Google improved its search techniques such that it provides customized search to retrieve information for a specific web site. Google use the information about
usage statistics. This enhances the quality of Google’s results. It provides better search options to reach images and look for pages that have been updated dynamically. Google’s web directory provides a fast and easy way to search within a certain topic or related topics.

It can be concluded that the web usage mining is the area of research which is continuously evolving and is affecting every experience of the user on the web.

In 1999, a ‘Zona Research Report’ popularized the “8-second-rule” [89]. The 8-second rule is an old way of measuring the adequate response time of a web server through different bandwidth connections. It specified that if the load-time of a web page exceeds eight seconds, users are unlikely to wait, or "stick around", for its completion. In order to increase the "stickiness" of a website, faster ways to deliver the content to the user needed to be devised. Web Prefetching is an ideal way of doing so and has recently captured the eyes of the researchers. The next section is an attempt to survey what web prefetching is, its types and the different ways in which the web prefetching can be applied to the WWW domain.

### 2.5 WEB PREFETCHING

The increase in demand for WWW resources has exacerbated the response time as perceived by users in retrieving web objects, also referred to as access latency. The WWW became the “World Wide Wait”. The users started experiencing long delays for their requests over the web. The growth of Internet and the increase in number of clients using the Internet have challenged scientists to improve web performance. Even with the availability of high bandwidth Internet connections, fast processors and large amount of storage, the access latency problem has remained a challenge for researchers.

- One solution for improving the performance of the web or reducing the user perceived latency is to use *web caching*, which provides space for local storage and management of previously accessed web documents. Web caching is recognized as one of the effective techniques to alleviate the server bottleneck and reduce network traffic, thereby reducing network
latency. The basic idea is to cache requested pages at the server so that they don’t have to be fetched again. A request for a document already present in the web cache can be directly serviced instead of being forwarded to the intended web server. This reduces the latency experienced by the clients as well as the Internet traffic. Abrams [90] et al. has shown that the maximum achievable hit rate of caching proxies is 30% to 50%.

Although web cache schemes reduce the network and I/O bandwidth consumption, they still suffer from a low hit rate, stale data and inefficient resource management. The caching cannot prove beneficial if the web pages were not visited in the past.

- To improve cache performance, researches have introduced web prefetching [91, 92], to work in conjunction with web caching, which means prefetching web documents from web servers, even before the user requests them. The intuition is that “if predictions could be made about the pages that a user is most likely to be accessed, possibly in the near future by analyzing his/her historical click behaviour in advance, it would effectively reduce the responding time”.

Nevertheless, its potential positive effects may cause excessive network traffic and, without a carefully designed prefetching algorithm, several transferred pages may not be used by the client at all.

Prefetching techniques rely on predictive approaches to speculatively retrieve and store web objects into the cache for future use. Predictions on what to prefetch are made based on different criteria such as history, popularity and content. Much work has already been done in the field of prefetching, which has shown to effectively reduce web latencies by utilizing the user idle time. The idle time is the time elapsed between the current request and the next request of the user. Web prefetching is based on the similar idea of prefetching used in memory management of computer machines. Although analogous, the techniques applied in web prefetching are very different from the ones applied in
memory management of computer hardware. Researchers take advantage of the facts that the loading and displaying of pages requires a few seconds and that there is a substantial time gap between two consecutive requests from the user in different ways. Thus, different strategies need to be developed for web prefetching.

2.5.1 PREFETCHING: TYPES AND LOCATION

Prefetching is not a new concept. It is still evolving depending upon the user’s ever increasing need of reducing the latency time. In a web context, the prefetching can be of following types [93]:

1. **Client-based Prefetching:** In this, the prediction model is built at the client side, using a navigational pattern of the individual(s) using this client. Thus the model covers the behaviour of a single or a few users across different web servers.

2. **Server-based Prefetching:** In this, the model is built at the server, thus covering the usage pattern of all users accessing this specific server thereby offering the advantage of collecting much more data concerning individual web pages. This enables more exact predictions.

3. **Combined Client and Server Prefetching:** In this, the above mentioned both approaches are merged by the exchange of prediction data between server and client.

4. **Proxy-based Prefetching:** In this, the proxy can prefetch documents in its own cache. It can also give hints to the client about documents that may be worth prefetching.

The different architectures of prefetching the web documents at the various levels is shown in Fig. 2.7 (a), (b), (c), (d).
(a) Client side

(b) Server Side

(c) Combined Model
Prefetching can be done in various ways. The various prefetching strategies adopted by researchers in the recent past have been discussed in the next subsection.

2.5.2 PREFETCHING STRATEGIES

The purpose of prefetching strategies is to predict which documents are likely to be requested next in a session, depending on the documents requested so far. There are three main approaches to do this as described below:

1. **Popularity based Strategies**: These approaches make predictions based on the popularity of the web pages.

2. **Semantic Prefetching Strategies**: They analyze the content of documents or metadata attached to them to decide which documents will probably be requested. Typical input values of semantic strategies are:
   i. Hyperlinks contained in the document.
   ii. Keywords attached to or extracted from the document.
3. **Statistic Prefetching Strategies**: They base their decisions on statistics built from previous sessions. These statistics are based on certain input values which may include:

   i. First-order transition probabilities between a document and the next one.
   
   ii. Higher-order transition probabilities, depending on several previous documents.
   
   iii. Time passed between requests.

The next subsection looks into the various types of prefetching strategies and their advantages and disadvantages.

### 2.5.2.1 Popularity Based Prefetching Strategies

In 1998, Markatos et al. proposed a ‘top ten approach’ for prefetching [94]. Their idea is to keep the top ten popular documents for each web server. By this it meant that clients or proxy servers can prefetch only these popular documents without significantly increasing network traffic. Their result showed that this approach expects more than 40% of client requests and achieves close to a 60% hit ratio at the cost of increasing network traffic by no more than 10% in most cases. Their experiment is close to the prefetching by *Popularity* algorithm. The algorithm keeps copies of *n* most popular objects in the cache and updates them immediately whenever these objects are modified. From the Zipf-like distribution, we know that popular objects are responsible for majority of requests from users. If in cache, copies of popular objects are kept which are most likely to be requested, this will definitely achieve the highest possible hit ratio. On the other hand, its bandwidth consumption is high. Other popularity based strategies may include ‘prefetching by lifetime’ and ‘prefetching by good fetch’ as explained below:

1. **Prefetching by Lifetime**

The lifetime of an object is the interval between two consecutive modifications of object. The longer mean lifetime an object has, the less frequently it changes. The bandwidth
cost to update stale objects hence decreases. If \( n \) objects with the longest lifetime are selected to replicate in the local cache, the least network traffic usage can be envisioned.

2. Prefetching by Good Fetch

This algorithm was proposed by Venkataramani [95]. It balanced object access frequency and object update frequency. One of the problems with prefetching is that the prefetched object may not end up being used or being used before it gets stale. If an object ends up being referenced before it goes stale, it is considered as a good fetch. This algorithm calculates the probability of a good fetch of an object. It only prefetched objects whose probability of being accessed before being modified is above a given value. As for object ‘\( i \)’, assume the object’s lifetime is ‘\( l_i \)’, its probability of being accessed be ‘\( p_i \)’, and user request arrival rate denoting how many requests arrive per second is ‘\( a \)’. Then the probability of object ‘\( i \)’ to be accessed before it dies is

\[
P_{\text{goodFetch}} = 1 - (1 - p_i)^{a \times l_i} \quad \ldots \quad (2.1)
\]

Where \( a \times l_i \) is the number of requests arriving during the lifetime of object ‘\( i \)’, \( (1 - p_i)^{a \times l_i} \) represents the probability that none of requests arriving during the lifetime of object ‘\( i \)’ are to object ‘\( i \)’. Thus, \( 1 - (1 - p_i)^{a \times l_i} \) denotes the probability of object \( i \) to be accessed before it dies.

2.5.2.2 SEMANTIC PREFETCHING STRATEGIES

These strategies can be further of two types as:

1. Hyperlink based Strategies

A naive way of prefetching is called informed prefetching [96]. This requires the system to know which requests will take place in the near future and to disclose this knowledge to allow prefetching these requests. While this is useful for batch applications, it cannot be used in interactive applications, where the user chooses what will happen next. Such strategies have been followed in various client-based applications. For example, ‘Web accelerators’ use prefetching to reduce latencies in web browsers [97]. However, the algorithms employed are usually very simplistic, such as using the idle time of the web
connection to prefetch all links of the page currently viewed. While this reduces latencies, it also results in a huge increase in network traffic [90]. One such simple approach, called interactive prefetching scheme (IPS) [98, 99], analyzes HTML pages and prefetches some or all of the linked pages. This strategy is implemented in the WWWCollector proxy server.

2. Keyword-based Strategies

The WebWatcher system uses interests specified by a user to guess which hyperlinks in the web page have currently been viewed and this particular user is most likely to follow. Each user reveals their interests in several keywords. Every page is annotated with the keywords of all the users that have viewed it. The system compares the annotations on the hyperlinks with the keywords of the user to find the closest match [100].

GOTO Software [101] offers a client-side system that prefetches pages linked from the current page by comparing the link text with areas of interest (i.e., keywords) stated by the user. A similar approach is chosen by Naviscope Software [102]. By default the software prefetches links that contains the word “next”, but it can be configured to look for other keywords.

2.5.2.3 STATISTIC PREFETCHING STRATEGIES

These strategies can follow the following two methodologies:

1. Dependency Graphs

A simple prefetching algorithm using dependency graphs is proposed by [103]. The graph is constructed on a web server. Each file on the server is a node of the graph. If a client requests a document B during some “look-ahead window time w” after it requested a document A, an arc leads from A to B. Multiple requests of the same document during w by a client are counted only once. The weight of the arc is the number of accesses to B after A divided by the number of accesses to A. A document is considered worth of prefetching if its weight on the arc from the document currently viewed is higher than a prefetch threshold ‘P’. In this prefetching strategy, the server sends the client a
prefetching hint. The decision whether to actually prefetch this document is left to the client. This is a simple variation of the Prediction-by-Partial-Match approach, using a past history of length 1. Both these methods suffer from two major drawbacks i.e. they both consider only single past visit of the user and they don’t consider patterns corresponding to higher dependencies [104].


Palpanas [93] proposes an algorithm called Prediction by Partial Match (PPM). This prediction model is adapted from data compression algorithms, where predictions of the next elements of sequences are used to choose effective compression schemes. Based on a probability distribution of the data, frequent sequences are encoded using few bits. The less frequently sequences occur - the more bits are used to encode them. Such probability distribution is useful to prefetching as well, where the pages that are most likely to be required should be prefetched.

The compression algorithms which give best results use context modeling to calculate a prediction model. For text compression, the probability is calculated using Higher-Order Markov Models, i.e., the probability of a character is calculated depending on the last n characters preceding it where n is called the order of the model. The predictive precision of the model increases with n, but so does the amount of data that must be stored in the prediction model. Also, for larger n’s (models of higher order), most contexts are seen seldom or never. In such cases, the model cannot be used reliably. A solution to this problem is to keep probability models of different orders and switch to the next lowest order if no data or not enough data is available in the higher order model. Even better is blending the different order models, e.g., combining their predictions into a single probability. This algorithm can easily be adapted to Web prefetching. Instead of sequences of characters, sequences of Web page requests are measured. Thus a probability model of the way users navigate in the web is build. For building the model, an order-m prefetcher contains m + 1 Markov predictors of order 0 to m. An order-k Markov predictor calculates the probability \( p(P|P_k, P_{k-1}, \ldots, P_1) \), i.e., the conditional probability \( p \) that a page \( P \) is accessed after sequential accesses to pages \( P_1, P_2, \ldots, P_k \).
Thus a k-order markov predictor represents a context of length \( k \). The context 0 does not capture a conditional probability, but the unconditional probability that a document is fetched. Considering the huge number of documents on a web site, this probability is of little or no real use and may be omitted. Predictions can be made in different variations, either only events of the highest available order are considered, or predictions from several orders can be combined (blended). Another possibility would be to report the results of several orders, assigning a certain level of trust to each of them.

The prefetched documents are put in the same cache as normally fetched pages, thus replacing the least valuable documents. Palpanas [93] employs an LRU cache, but the algorithm would work with other replacement strategies without changes. This prefetching algorithm could be enhanced by enabling predictions for a longer time span, thus predicting several sequential requests. Another extension would be to make the algorithm adapt to server load, by reducing the amount of prefetching when load is high.

Another study using a PPM-based algorithm found a decrease in perceived latency by 23.4% [105]. They use a variation of the algorithm that can be configured to predict several steps in the future instead of just one. Documents are prefetched only if their probability of being requested surpasses a threshold value \( t \). Their algorithm reached a prediction accuracy of 40\% (with \( n, l, t = 2, 4, 0.125 \)) to 73\% (1, 1, 0.5), depending on the selected parameters, while causing an increase of 1\% to 15\% in traffic.

The next section discusses the role of agents in web mining domain.

### 2.6 ROLE OF AGENTS IN WEB MINING

Agent may refer to one who acts for, or in the place of another, by authority from him. An intelligent agent is an autonomous entity which observes and acts upon an environment and directs its activity towards achieving goals. These intelligent search agents may be mobile [106, 107]. The term mobile agent is often context-dependent and has two separate and distinct concepts: mobility and agency. The term agency implies having the same characteristics as that of an agent. These are self contained and identifiable computer programs that can move within the network from node to node and
act on behalf of a user or other entity. These can halt execution from a host without human interruption ([108]). The current network environment is based on the traditional client/server paradigm as shown in Fig. 2.8

![Client server architecture](image)

Fig 2.8 Client server architecture

However, in the case of mobile agents employed in a network, the service provision/utilization can be distributed in nature and is dynamically configured according to changing network performance metrics like congestion and user demand for service provision [109]. Mobile agents are typically suited to applications requiring structuring and coordinating wide area network and distributed services involving a large number of remote real time interactions.

They can decide how best a user’s request can be handled based on past data or history of a similar request. These programs are therefore capable of learning from user behavior to some extent. Fig. 2.9 shows how agents can act as the intermediary between the clients and the proxy to meet the client’s needs.

![Mobile Agent Communication](image)

Fig. 2.9 Mobile Agent Communication
2.7 CONCLUDING OBSERVATIONS

Recent advancements in the areas of internet technology and information retrieval have opened up exciting possibilities for mining WWW and thus help users in best possible manner to access the desired piece of information. Nevertheless, the existing web mining approaches suffer form various limitations and drawbacks. A critical look at the above literature review reveals the following problem areas with the current web mining approaches:

1. Relevancy is a very subjective term and keeps changing from user to user. To dig out the relevant web pages from WWW is quite a challenge.
2. How effectively the knowledge can be formed out of the vast information scattered on the web is another area for consideration.
3. The next challenging area comes from the lack in the capability of the existing web to personalize the information according to the users’ demand.
4. Page ranking is another problem area as current search engines give importance only to the link structure of the web page rather than its content.
5. To employ web prefetching in order to reduce the latency time for users is not important. What's important is how effectively it is used so that it does not add to the bandwidth requirements.

A framework for extracting relevant web pages from WWW using web mining is proposed in the succeeding chapters with the aim to handle the above said problems. To begin with, chapter three addresses the issue of extracting the relevant information from WWW for its users and comes out with the robust system that brings the most relevant and popular pages to the users.