Contents

X-XXXX-XXXX-X/XX/$X.XX+$X.XX

1

Web Usage Mining and Personalization

Bamshad Mobasher, DePaul University

contents

Abstract In this chapter we present a comprehensive overview of the personalization process based on Web usage mining. In this context we discuss a host of Web usage mining activities required for this process, including the preprocessing and integration of data from multiple sources, and common pattern discovery techniques that are applied to the integrated usage data. We also presented a number of specific recommendation algorithms for combining the discovered knowledge with the current status of a user's activity in a Web site to provide personalized content. The goal of this chapter is to show how pattern discovery techniques such as clustering, association rule mining, and sequential pattern discovery, performed on Web usage data, can be leveraged effectively as an integrated part of a Web personalization system.

X-XXXX-XXXX-X/XX/$X.XX+$X.XX

1.1 Introduction and Background

The tremendous growth in the number and the complexity of information resources and services on the Web has made Web personalization an indispensable tool for both Web-based organizations and for the end users. The ability of a site to engage visitors at a deeper level, and to successfully guide them to useful and pertinent information, is now viewed as one of the key factors in the site's ultimate success. Web personalization can be described as any action that makes the Web experience of a user customized to the user's taste or preferences. Principal elements of Web personalization include modeling of Web objects (such as pages or products) and subjects (such as users or customers), categorization of objects and subjects, matching between and across objects and/or subjects, and determination of the set of actions to be recommended for personalization.

To-date, the approaches and techniques used in Web personalization can be categorized into three general groups: manual decision rule systems, content-based filtering agents, and collaborative filtering systems. Manual decision rule systems, such as Broadvision (www.broadvision.com), allow Web site administrators to specify rules based on user demographics or static profiles (collected through a registration process). The rules are used to affect the content served to a particular user. Collaborative filtering systems such as Net Perceptions (www.netperceptions.com) typically take explicit information in the form of user ratings or preferences, and through a correlation engine, return information that is predicted to closely match the user's preferences. Content-based filtering systems such as those used by WebWatcher [Joachims et al., 1997] and client-side agent Letizia [Lieberman, 1995] generally rely on personal profiles and the content similarity of Web documents to these profiles for generating recommendations.

There are several well-known drawbacks to content-based or rule-based filtering techniques for personalization. The type of input is often a subjective description of the users by the users themselves, and thus is prone to biases. The profiles are often static, obtained through user registration, and thus the system performance degrades over time as the profiles age. Furthermore, using content similarity alone may result in missing important "pragmatic" relationships among Web objects based on how they are accessed by users. Collaborative filtering [Herlocker et al., 1999; Konstan et al., 1997; Shardanand and Maes, 1995] has tried to address some of these issues, and, in fact, has become the predominant commercial approach in most successful e-commerce systems. These techniques generally involve matching the ratings of a current user for objects (e.g., movies or products) with those of similar users (nearest neighbors) in order to produce recommendations for objects not yet rated by the user. The primary technique used to accomplish this task is the k-Nearest-Neighbor (kNN) classification approach which compares a target user's record with the historical records of other users in order to find the top k users who have similar tastes or interests.

However, collaborative filtering techniques have their own potentially serious limitations. The most important of these limitations is its lack of scalability. Essentially, kNN requires that the neighborhood formation phase be performed as an online process, and for very large data sets this may lead to unacceptable latency for providing recommendations. Another limitation of kNN-based techniques emanates from the sparce nature of the dataset. As the number of items in the database increases, the density of each user record with respect to these items will decrease. This, in turn, will decrease the likelihood of a significant overlap of visited or rated items among pairs of users resulting in less reliable computed correlations. Furthermore, collaborative filtering usually performs best when explicit non-binary user ratings for similar objects are available. In many Web sites, however, it may be desirable to integrate the personalization actions throughout the site involving different types of objects, including navigational and content pages, as well as implicit product-oriented user events

such as shopping cart changes, or product information requests.

A number of optimization strategies have been proposed and employed to remedy these shortcomings [Aggarwal et al., 1999; O'Conner and Herlocker, 1999; Sarwar et al., 2000a; Ungar and Foster, 1998; Yu, 1999]. These strategies include similarity indexing and dimensionality reduction to reduce real-time search costs, as well as offline clustering of user records, allowing the online component of the system to search only within a matching cluster. There has also been a growing body of work in enhancing collaborative filtering by integrating data from other sources such as content and user demographics [Claypool et al., 1999; Pazzani, 1999].

More recently, Web usage mining [Srivastava et al., 2000] has been proposed as an underlying approach for Web personalization [Mobasher et al., 2000a]. The goal of Web usage mining is to capture and model the behavioral patterns and profiles of users interacting with a Web site. The discovered patterns are usually represented as collections of pages or items that are frequently accessed by groups of users with common needs or interests. Such patterns can be used to better understand behavioral characteristics of visitors or user segments, improve the organization and structure of the site, and create a personalized experience for visitors by providing dynamic recommendations. The flexibility provided by Web usage mining can help enhance many of the approaches discussed above and remedy many of their shortcomings. In particular, Web usage mining techniques, such as clustering, association rule mining, and navigational pattern mining, that rely on offline pattern discovery from user transactions can be used to improve the scalability of collaborative filtering when dealing with clickstream and e-commerce data.

The goal of personalization based on Web usage mining is to recommend a set of objects to the current (active) user, possibly consisting of links, ads, text, products, or services, tailored to the user's perceived preferences as determined by the matching usage patterns. This task is accomplished by matching the active user session (possibly in conjunction with previously stored profiles for that user) with the usage patterns discovered through Web usage mining. We call the usage patterns used in this context aggregate usage profiles since they provide an aggregate representation of the common activities or interests of groups of users. This process is performed by the recommendation engine which is the online component of the personalization system. If the data collection procedures in the system include the capability to track users across visits, then the recommendations can represent a longer term view of user's potential interests based on the user's activity history within the site. If, on the other hand, aggregate profiles are derived only from user sessions (single visits) contained in log files, then the recommendations provide a "short-term" view of user's navigational interests. These recommended objects are added to the last page in the active session accessed by the user before that page is sent to the browser.

The overall process of Web personalization based on Web usage mining consists of three phases: data preparation and transformation, pattern discovery, and recommendation. Of these, only the latter phase is performed in real-time. The data preparation phase transforms raw Web log files into transaction data that can be processed by data mining tasks. This phase also includes data integration from multiple sources, such as backend databases, application servers, and site content. A variety of data mining techniques can be applied to this transaction data in the pattern discovery phase, such as clustering, association rule mining, and sequential pattern discovery. The results of the mining phase are transformed into aggregate usage profiles, suitable for use in the recommendation phase. The recommendation engine considers the active user session in conjunction with the discovered patterns to provide personalized content.

In this chapter we present a comprehensive view of the personalization process based on Web usage mining. A generalized framework for this process is depicted in Figures 1.1 and 1.2. We use this framework as our guide in the remainder of this chapter. We provide

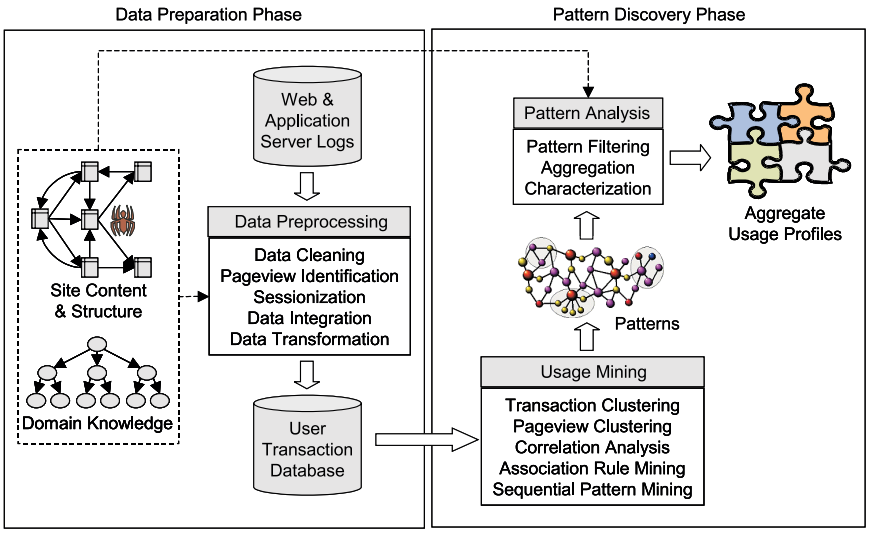

Figure 1.1: The offline data preparation and pattern discovery components.

a detailed discussion of a host of Web usage mining activities necessary for this process, including the preprocessing and integration of data from multiple sources (Section 1) and common pattern discovery techniques that are applied to the integrated usage data (Section 1.2.4). We then present a number of specific recommendation algorithms for combining the discovered knowledge with the current status of a user's activity in a Web site to provide personalized content to a user. This discussion shows how pattern discovery techniques such as clustering, association rule mining, and sequential pattern discovery, performed on Web usage data, can be leveraged effectively as an integrated part of a Web personalization system (Section 1.3.2.3).

1.2 Data Preparation and Modeling

An important task in any data mining application is the creation of a suitable target data set to which data mining algorithms are applied. This process may involve preprocessing the original data, integrating data from multiple sources, and transforming the integrated data into a form suitable for input into specific data mining operations. Collectively, we refer to this process as data preparation.

The data preparation process is often the most time consuming and computationally intensive step in the knowledge discovery process. Web usage mining is no exception: in fact, the data preparation process in Web usage mining, often requires the use of especial algorithms and heuristics not commonly employed in other domains. This process is critical to the successful extraction of useful patterns from the data. In this section we discuss some of the issues and concepts related to data modeling and preparation in Web usage mining. While this discussion is in the general context of Web usage analysis, we are focused especially on the factors that have been shown to greatly affect the quality and usability of the discovered usage patterns for their application in Web personalization.

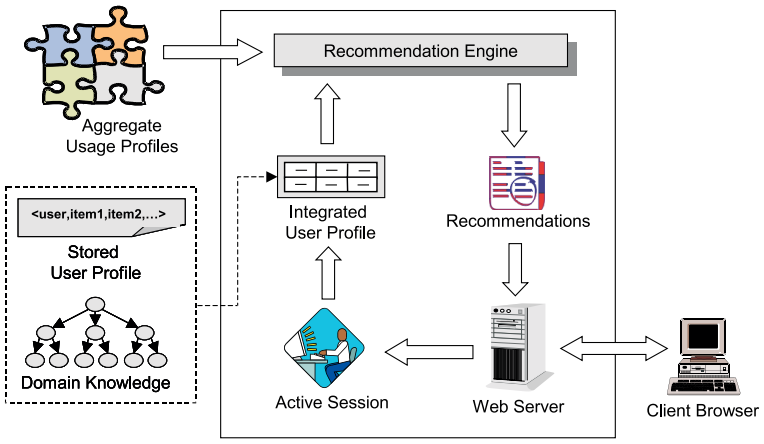

Figure 1.2: The online personalization component.

1.2.1 Sources and Types of Data

The primary data sources used in Web usage mining are the server log files, which include Web server access logs and application server logs. Additional data sources that are also essential for both data preparation and pattern discovery include the site files and meta data, operational databases, application templates, and domain knowledge. Generally speaking, the data obtained through these sources can be categorized into four groups ([Cooley et al., 1999; Srivastava et al., 2000]):

1.2.1.1 Usage data

The log data collected automatically by the Web and application servers represents the fine-grained navigational behavior of visitors. Depending on the goals of the analysis, this data needs to be transformed and aggregated at different levels of abstraction.

In Web usage mining, the most basic level of data abstraction is that of a pageview. Physically, a pageview is an aggregate representation of a collection of Web objects contributing to the display on a user's browser resulting from a single user action (such as a clickthrough). These Web objects may include multiple pages (such as in a frame-based site), images, embedded components, or script and database queries that populate portions of the displayed page (in dynamically generated sites). Conceptually, each pageview represents a specific "type" of user activity on the site, e.g., reading a news article, browsing the results of a search query, viewing a product page, adding a product to the shopping cart, and so on.

On the other hand, at the user level, the most basic level if behavioral abstraction is that of a server session (or simply a session). A session (also commonly referred to as a "visit") is a sequence of pageviews by a single user during a single visit. The notion of a session can be further abstracted by selecting a subset of pageviews in the session that are significant or relevant for the analysis tasks at hand. We shall refer to such a semantically meaningful subset of pageviews as a transaction (also referred to as an episode according to the W3C Web Characterization Activity [W3C]). It is important to note that a transaction does not refer simply to product purchases, but it can include a variety of types of user actions as captured by different pageviews in a session.

1.2.1.2 Content data

The content data in a site is the collection of objects and relationships that are conveyed to the user. For the most part, this data is comprised of combinations of textual material and images. The data sources used to deliver or generate this data include static HTML/XML pages, images, video clips, sound files, dynamically generated page segments from scripts or other applications, and collections of records from the operational database(s). The site content data also includes semantic or structural meta-data embedded within the site or individual pages, such as descriptive keywords, document attributes, semantic tags, or HTTP variables.

Finally, the underlying domain ontology for the site is also considered part of the content data. The domain ontology may be captured implicitly within the site, or it may exist in some explicit form. The explicit representations of domain ontologies may include conceptual hierarchies over page contents, such as product categories, structural hierarchies represented by the underlying file and directory structure in which the site content is stored, explicit representations of semantic content and relationships via an ontology language such as RDF, or a database schema over the data contained in the operational databases.

1.2.1.3 Structure data

The structure data represents the designer's view of the content organization within the site. This organization is captured via the inter-page linkage structure among pages, as reflected through hyperlinks. The structure data also includes the intra-page structure of the content represented in the arrangement of HTML or XML tags within a page. For example, both HTML and XML documents can be represented as tree structures over the space of tags in the page.

The structure data for a site is normally captured by an automatically generated "site map" which represents the hyperlink structure of the site. A site mapping tool must have the capability to capture and represent the inter- and intra-pageview relationships. This necessity becomes most evident in a frame-based site where portions of distinct pageviews may represent the same physical page. For dynamically generated pages, the site mapping tools must either incorporate intrinsic knowledge of the underlying applications and scripts, or must have the ability to generate content segments using a sampling of parameters passed to such applications or scripts.

1.2.1.4 User data

The operational database(s) for the site may include additional user profile information. Such data may include demographic or other identifying information on registered users, user ratings on various objects such as pages, products, or movies, past purchase or visit histories of users, as well as other explicit or implicit representations of a users' interests.

Obviously, capturing such data would require explicit interactions with the users of the site. Some of this data can be captured anonymously, without any identifying user information, so long as there is the ability to distinguish among different users. For example, anonymous information contained in client-side cookies can be considered a part of the users' profile information, and can be used to identify repeat visitors to a site. Many personalization applications require the storage of prior user profile information. For example, collaborative filtering applications usually store prior ratings of objects by users, though, such information can be obtained anonymously, as well.

1.2.2 Usage Data Preparation

The required high-level tasks in usage data preprocessing include data cleaning, pageview identification, user identification, session identification (or sessionization), the inference of missing references due to caching, and transaction (episode) identification. We provide a brief discussion of some of these tasks below; for a more detailed discussion see [Cooley, 2000; Cooley et al., 1999].

Data cleaning is usually site-specific, and involves tasks such as, removing extraneous references to embedded objects, graphics, or sound files, and removing references due to spider navigations. The latter task can be performed by maintaining a list of known spiders, and through heuristic identification of spiders and Web robots [Tan and Kumar, 2002]. It may also be necessary to merge log files from several Web and application servers. This may require global synchronization across these servers. In the absence of shared embedded session ids, heuristic methods based on the "referrer" field in server logs along with various sessionization and user identification methods (see below) can be used to perform the merging.

Client- or proxy-side caching can often result in missing access references to those pages or objects that have been cached. Missing references due to caching can be heuristically inferred through path completion which relies on the knowledge of site structure and referrer information from server logs [Cooley et al., 1999]. In the case of dynamically generated pages, form-based applications using the HTTP POST method result in all or part of the user input parameter not being appended to the URL accessed by the user (though, in the latter case, it is possible to re-capture the user input through packet sniffers on the server side).

Identification of pageviews is heavily dependent on the intra-page structure of the site, as well as on the page contents and the underlying site domain knowledge. For a single frame site, each HTML file has a one-to-one correlation with a pageview. However, for multi-framed sites, several files make up a given pageview. Without detailed site structure information, it is very difficult to infer pageviews from a Web server log. In addition, it may be desirable to consider pageviews at a higher level of aggregation, where each pageview represents a collection of pages or objects, for examples, pages related to the same concept category.

Not all pageviews are relevant for specific mining tasks, and among the relevant pageviews some may be more significant than others. The significance of a pageview may depend on usage, content and structural characteristics of the site, as well as on prior domain knowledge (possibly specified by the site designer and the data analyst). For example, in an e-commerce site pageviews corresponding to product-oriented events (e.g., shopping cart changes or product information views) may be considered more significant than others. Similarly, in a site designed to provide content, content pages may be weighted higher than navigational pages. In order to provide a flexible framework for a variety of data mining activities a number of attributes must be recorded with each pageview. These attributes include the pageview id (normally a URL uniquely representing the pageview), duration, static pageview type (e.g., information page, product view, or index page), and other metadata, such as content attributes.

The analysis of Web usage does not require knowledge about a user's identity. However, it is necessary to distinguish among different users. In the absence of registration and authentication mechanisms, the most wide spread approach to distinguishing among unique is the use of client-side cookies. Not all sites, however, employ cookies, and due to abuse by some organizations and because of privacy concerns on the part of many users, client-side cookies are sometimes disabled. IP addresses, alone, are not generally sufficient for mapping log entries onto the set of unique users. This is mainly due the proliferation of

ISP proxy servers which assign rotating IP addresses to clients as they browse the Web. It is not uncommon, for instance, to find a substantial percentage of IP addresses recorded in server logs of a high-traffic site as belonging to America Online proxy server or other major ISPs. In such cases, it is possible to more accurately identify unique users through combinations IP addresses and other information such as the user agents, operating systems, and referrers [Cooley et al., 1999].

Since a user may visit a site more than once, the server logs record multiple sessions for each user. We use the phrase user activity log to refer to the sequence of logged activities belonging to the same user. Thus, sessionization is the process of segmenting the user activity log of each user into sessions. Web sites without the benefit of additional authentication information from users and without mechanisms such as embedded session ids must rely on heuristics methods for sessionization. A sessionization heuristic is a method for performing such a segmentation on the basis of assumptions about users' behavior or the site characteristics.

The goal of a heuristic is the reconstruction of the real sessions, where a real session is the actual sequence of activities performed by one user during one visit to the site. We denote the "conceptual" set of real sessions by $\mathcal{R}$. A sessionization heuristic h attempts to map $\mathcal{R}$ into a set of constructed sessions, which we denote as $\mathcal{C} \equiv \mathcal{C}_h$. For the ideal heuristic, $h*$, we have $\mathcal{C} \equiv \mathcal{C}_h* = \mathcal{R}$. Generally, sessionization heuristics fall into two basic categories: time-oriented or structure-oriented. Time-oriented heuristics apply either global or local time-out estimates to distinguish between consecutive sessions, while structure-oriented heuristics use either the static site structure or the implicit linkage structure captured in the referrer fields of the server logs. Various heuristics for sessionization have been identified and studied [Cooley et al., 1999]. More recently, s formal framework for measuring the effectiveness of such heuristics has been proposed [Spiliopoulou et al., 2003], and the impact of different heuristics on various Web usage mining tasks has been analyzed [Berendt et al., 2002b].

Finally, transaction (episode) identification can be performed as a final preprocessing step prior to pattern discovery in order to focus on the relevant subsets of pageviews in each user session. As noted earlier, this task may require the automatic or semi-automatic classification of pageviews into different functional types or into concept classes according to a domain ontology. In highly dynamic sites, it may also be necessary to map pageviews within each session into "service-base" classes according to a concept hierarchy over the space of possible parameters passed to script or database queries [Berendt and Spiliopoulou, 2000]. For example, the analysis may ignore the quantity and attributes of an item added to the shopping cart, and focus only on the action of adding the item to the cart.

The above preprocessing tasks ultimately result in a set of n pageviews, $P = \{p_1, p_2, \cdots, p_n\}$, and a set of m user transactions, $T = \{t_1, t_2, \cdots, t_m\}$, where each $t_i \in T$ is a subset of P. Conceptually, we can view each transaction t as an l-length sequence of ordered pairs:

$$t = \left\langle (p_1^t, w(p_1^t)), (p_2^t, w(p_2^t)), \cdots, (p_l^t, w(p_l^t)) \right\rangle,$$

where each $p_i^t = p_j$ for some $j \in \{1, \cdots, n\}$, and $w(p_i^t)$ is the weight associated with pageview p_i^t in the transaction t.

The weights can be determined in a number of ways, in part based on the type of analysis or the intended personalization framework. For example, in a collaborative filtering applications, such weights may be determined based on user ratings of items. In most Web usage mining tasks, the focus is generally on anonymous user navigational activity where the primary sources of data are server logs. This allows us to choose two types of weights for pageviews: weights can be binary, representing the existence or non-existence of a pageview in the transaction; or they can be a function of the duration of the pageview in the user's session. In the case of time durations, it should be noted that usually the time spent by a

user on the last pageview in the session is not available. One commonly used option is to set the weight for the last pageview to be the mean time duration for the page taken across all sessions in which the pageview does not occur as the last one.

Whether or not the user transactions are viewed as sequences or as sets (without taking ordering information into account) is also dependent on the goal of the analysis and the intended applications. For sequence analysis and the discovery of frequent navigational patterns, one must preserve the ordering information in the underlying transaction. On the other hand, for clustering tasks as well as for collaborative filtering based on kNN and association rule discovery, we can represent each user transaction as a vector over the n-dimensional space of pageviews, where dimension values are the weights of these pageviews in the corresponding transaction. Thus given the transaction t above, the n-dimensional transaction vector $\vec{t}$ is given by:

$$\vec{t} = \langle w^t_{p_1}, w^t_{p_2}, \cdots, w^t_{p_n} \rangle,$$

where each $w^t_{p_j} = w(p^t_i)$, for some $i \in \{1, \cdots, n\}$, in case p_j appears in the transaction t, and $w^t_{p_j} = 0$, otherwise. For example, consider a site with 6 pageviews A, B, C, D, E, and F. Assuming that the pageview weights associated with a user transaction are determined by the number of seconds spent on them, a typical transaction vector may look like: $\langle 11, 0, 22, 5, 127, 0 \rangle$. In this case, the vector indicates that the user spent 11 seconds on page A, 22 seconds on page C, 5 seconds on page D, and 127 seconds on page E. The vector also indicates that the user did not visit pages B and F during this transaction.

Given this representation, the set of all m user transactions can be conceptually viewed as an $m \times n$ transaction-pageview matrix which we shall denote by TP. This transaction-pageview matrix can then be used to perform various data mining tasks. For example, similarity computations can be performed among the transaction vectors (rows) for clustering and kNN neighborhood formation tasks, or an association rule discovery algorithm, such as Apriory, can be applied (with pageviews as items) to find frequent itemsets of pageviews.

1.2.3 Post-Processing of User Transactions Data

In addition to the aforementioned preprocessing steps leading to user transaction matrix, there are a variety of transformation tasks that can be performed on the transaction data. Here, we highlight some of data transformation tasks that are likely to have an impact on the quality and actionability of the discovered patterns resulting from mining algorithms. Indeed, such post-processing transformations on session or transaction data have been shown to result in improvements in the accuracy of recommendations produced by personalization systems based on Web usage mining [Mobasher et al., 2001b].

1.2.3.1 Significance Filtering

Using binary weights in the representation of user transactions is often desirable due to efficiency requirements in terms of storage and computation of similarity coefficients among transactions. However, in this context, it becomes more important to determine the significance of each pageview or item access. For example, a user may access an item p only to find that she is not interested in that item, subsequently backtracking to another section of the site. We would like to capture this behavior by discounting the access to p as an insignificant access. We refer to the processing of removing page or item requests that are deemed insignificant as significance filtering.

The significance of a page within a transaction can be determined manually or automatically. In the manual approach, the site owner or the analyst is responsible for assigning

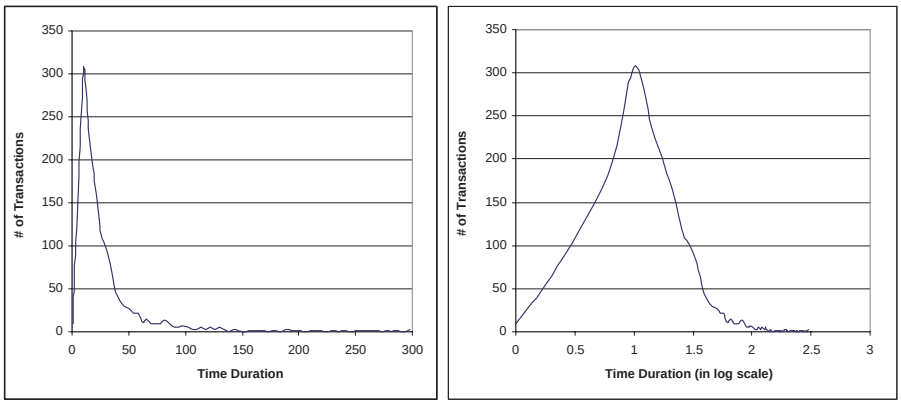

Figure 1.3: Distribution of pageview durations: raw-time scale (left), log-time scale (right).

significance weights to various pages or items. This is usually performed as a global mapping from items to weights, and thus the significance of the pageview is not dependent on a specific user or transaction. More commonly, a function of pageview duration is used to automatically assign signficance weights. In general, though, it is not sufficient to filter out pageviews with small durations since the amount of time spent by users on a page is not only based on the user's interest on the page. The page duration may also be dependent on the characteristics and the content of the page. For example, we would expect that users spend far less time on navigational pages than they do on content or product-oriented pages.

Statistical significance testing can help capture some of the semantics illustrated above. The goal of significance filtering is to eliminate irrelevant items with time duration significantly below a certain threshold in transactions. Typically, statistical measures such as mean and variance can be to systematically define the threshold for significance filtering.

In general, it can be observed that the distribution of access frequencies as a function of the amount of time spent on a given pageview is characterized by a log-normal distribution. For example, Figure 1.3 (left) shows the distribution of the number of transactions with respect to time duration for a particular pageview in a typical Web site. Figure 1.3 (right) shows the distribution plotted as a function of time in a log scale. The log normalization can be observed to produce a Gaussian distribution. After this transformation, we can proceed with standard significance testing: the weight associated with an item in a transaction will be considered to be 0, if the amount of time spent on that item is significantly below the mean time duration of the item across all user transactions. The significance of variation from the mean is usually measured in terms of multiples of standard deviation. For example, in a given transaction t, if the amount of time spent on a pageview p is 1.5 to 2 standard deviations lower than the mean duration for p across all transactions, then, the weight of p in transaction t might be set to 0. In such as case, it is likely that the user was either not interested in the contents of p, or mistakenly navigated to p and quickly left the page.

1.2.3.2 Normalization

There are also some advantages in using the fully weighted representation of transaction vectors (based on time durations). One advantage is that for many distance- or similarity-based clustering algorithms, more granularity in feature weights usually leads to more accurate results. Another advantage is that, since relative time durations are taken into account, the

need for performing other types of transformations, such as significance filtering, is greatly reduced.

However, raw time durations may not be an appropriate measure for the significance of a pageview. This is because a variety of factors, such as structure, length, and the type of pageview, as well as the user's interests in a particular item, may affect the amount of time spent on that item. Appropriate weight normalization can play an essential role in correcting for these factors.

Generally two types of weight normalization are applied to user transaction vectors: normalization across pageviews in a single transaction and normalization of a pageview weights across all transactions. We call these transformations transaction normalization and pageview normalization, respectively. Pageview normalization is useful in capturing the relative weight of a pageview for a user with respect to the weights of the same pageview for all other users. On the other hand, transaction normalization captures the importance of a pageview to a particular user relative to the other items visited by that user in the same transaction. The latter is particularly useful in focusing on the "target" pages in the context of short user histories.

1.2.4 Data Integration from Multiple Sources

In order to provide the most effective framework for pattern discovery and analysis, data from a variety of sources must be integrated. Our earlier discussion already alluded to the necessity of considering the content and structure data in a variety of preprocessing tasks such as pageview identification, sessionization, and the inference of missing data. The integration of content, structure, and user data in other phases of the Web usage mining and personalization processes may also be essential in providing the ability to further analyze and reason about the discovered patterns, derive more actionable knowledge, and create more effective personalization tools.

For example, in e-commerce applications, the integration of both user data (e.g., demographics, ratings, purchase histories) and product attributes from operational databases is critical. Such data, used in conjunction with usage data, in the mining process can allow for the discovery of important business intelligence metrics such as customer conversion ratios and lifetime values. On the other hand, the integration of semantic knowledge from the site content or domain ontologies can be used by personalization systems to provide more useful recommendations. For instance, consider a hypothetical site containing information about movies which employs collaborative filtering on movie ratings or pageview transactions to give recommendations. The integration of semantic knowledge about movies (possibly extracted from site content), can allow the system to recommend movies, not just based on similar ratings or navigation patterns, but also perhaps based on similarities in attributes such as movie genres or commonalities in casts or directors.

One direct source of semantic knowledge that can be integrated into the mining and personalization processes is the collection of content features associated with items or pageviews on a Web site. These features include keywords, phrases, category names, or other textual content embedded as meta-information. Content preprocessing involves the extraction of relevant features from text and meta-data. Meta-data extraction becomes particularly important when dealing with product-oriented pageviews or those involving non-textual content. In order to use features in similarity computations, appropriate weights must be associated with them. Generally, for features extracted from text we can use standard techniques from information retrieval and filtering to determine feature weights [Frakes and Baeza-Yates, 1992]. For instance, a commonly used feature-weighting scheme is tf.idf which is a function of the term frequency and inverse document frequency.

More formally,, each pageview p can be represented as a k-dimensional feature vector,

where k is the total number of extracted features from the site in a global dictionary. Each dimension in a feature vector represents the corresponding feature weight within the pageview. Thus, the feature vector for a pageview p is given by:

$$p = \langle fw(p, f_1), fw(p, f_2), \cdots, fw(p, f_k) \rangle ,$$

where $fw(p, f_j)$, is the weight of the jth feature in pageview $p \in P$, for $1 \leq j \leq k$. For features extracted from textual content of pages, the feature weight is usually the normalized tf.idf value for the term. In order to combine feature weights from meta-data (specified externally) and feature weights from the text content, proper normalization of those weights must be performed as part of preprocessing.

Conceptually, the collection of these vectors can be viewed as a $n \times k$ pageview-feature matrix in which each row is a feature vector corresponding to one of the n pageviews in P. We shall call this matrix PF. The feature vectors obtained in this way are usually organized into an inverted file structure containing a dictionary of all extracted features and posting files for each feature specifying the pageviews in which the feature occurs along with its weight. This inverted file structure corresponds to the transpose of the matrix PF.

Further preprocessing on content features can be performed by applying text mining techniques. This would provide the ability to filter the input to, or the output from, usage mining algorithms. For example, classification of content features based on a concept hierarchy can be used to limit the discovered usage patterns to those containing pageviews about a certain subject or class of products. Similarly, performing clustering or association rule mining on the feature space can lead to composite features representing concept categories. The mapping of features onto a set of concept labels allows for the transformation of the feature vectors representing pageviews into concept vectors. The concept vectors represent the semantic concept categories to which a pageview belongs, and they can be viewed at different levels of abstraction according to a concept hierarchy (either pre-existing or learned through machine learning techniques). This transformation can be useful both in the semantic analysis on the data, and as a method for dimensionality reduction in some data mining tasks, such as clustering.

A direct approach for the integration of content and usage data for Web usage mining tasks is to transform user transactions, as described earlier, into "content-enhanced" transactions containing the semantic features of the underlying pageviews. This process, performed as part of data preparation, involves mapping each pageview in a transaction to one or more content features. The range of this mapping can be the full feature space, or the concept space obtained as described above. Conceptually, the transformation can be viewed as the multiplication of the transaction-pageview matrix TP (described in Section 1.2.2) with the pageview-feature matrix PF. The result is a new matrix TF $= \{t'_1, t'_2, \ldots, t'_m\}$, where each t'_i is a k-dimensional vector over the feature space. Thus, a user transaction can be represented as a content feature vector, reflecting that user's interests in particular concepts or topics. A variety of data mining algorithms can then be applied to this transformed transaction data.

The above discussion focused primarily on the integration of content and usage data for Web usage mining. However, as noted earlier, data from other sources must also be considered as part of an integrated framework. Figure 1.4 shows the basic elements of such a framework.

The content analysis module in this framework is responsible for extracting and processing linkage and semantic information from pages. The processing of semantic information includes the steps described above for feature extraction and concept mapping. Analysis of dynamic pages may involve (partial) generation of pages based on templates, specified parameters, or database queries based on the information captured from log records. The

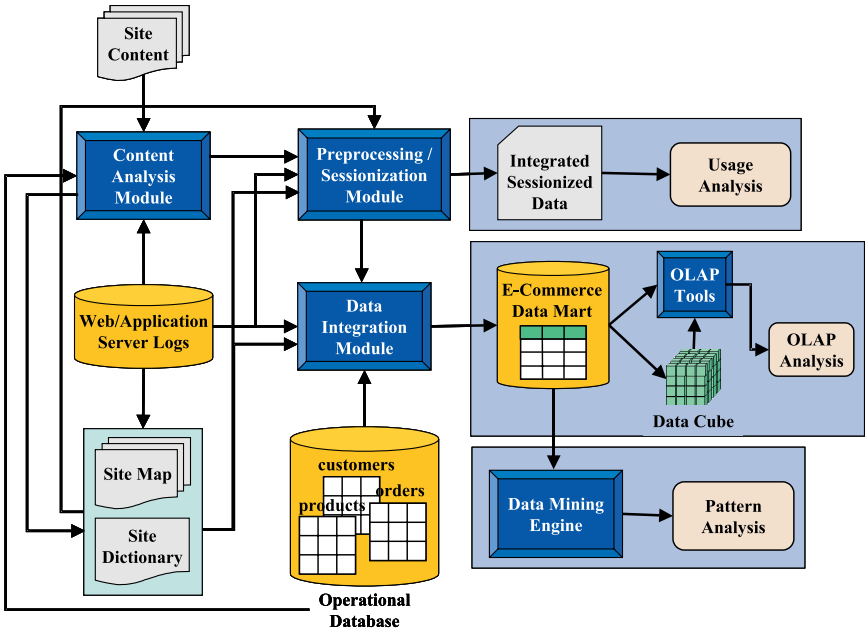

Figure 1.4: An integrated framework for Web usage analysis.

outputs from this module may include the site map capturing the site topology as well as the site dictionary and the inverted file structure used for content analysis and integration.

The site map is used primarily in data preparation (e.g., in pageview identification and path completion). It may be constructed through content analysis or the analysis of usage data (using the referrer information in log records). Site dictionary provides a mapping between pageview identifiers (for example URLs) and content or structural information on pages; it is used primarily for "content labeling" both in sessionized usage data as well as the integrated e-commerce data. Content labels may represent conceptual categories based on sets of features associated with pageviews.

The data integration module is used to integrate sessionized usage data, e-commerce data (from application servers), and product or user data from databases. User data may include user profiles, demographic information, and individual purchase activity. E-commerce data includes various product-oriented events, including shopping cart changes, purchase information, impressions, clickthroughs, and other basic metrics primarily used for data transformation and loading mechanism of the Data Mart. The successful integration of this type of e-commerce data requires the creation of a site-specific "event model" based on which subsets of a user's clickstream are aggregated and mapped to specific events such as the addition of a product to the shopping cart. Product attributes and product categories, stored in operational databases, can also be used to enhance or expand content features extracted from site files.

The e-commerce data mart is a multi-dimensional database integrating data from a variety of sources, and at different levels of aggregation. It can provide pre-computed e-metrics along multiple dimensions, and is used as the primary data source in OLAP analysis, as well as in data selection for a variety of data mining tasks (performed by the data mining engine).

We discuss different types and levels of analysis that can be performed based on this basic framework in the next section.

1.3 Pattern Discovery From Web Usage Data

1.3.1 Levels and Types of Analysis

As shown in Figure 1.4 different kinds of analysis can be performed on the integrated usage data at different levels of aggregation or abstraction. The types and levels of analysis, naturally, depend on the ultimate goals of the analyst and the desired outcomes.

For instance, even without the benefit of an integrated e-commerce datamart, statistical analysis can be performed on the preprocessed session or transaction data. Indeed, static aggregation (reports) constitutes the most common form of analysis. In this case, data is aggregated by predetermined units such as days, sessions, visitors, or domains. Standard statistical techniques can be used on this data to gain knowledge about visitor behavior. This is the approach taken by most commercial tools available for Web log analysis (however, most such tools do not perform all of the necessary preprocessing tasks described earlier, this resulting in erroneous or misleading outcomes). Reports based on this type of analysis may include information about most frequently accessed pages, average view time of a page, average length of a path through a site, common entry and exit points, and other aggregate measure.

The drawback of this type of analysis is the inability to "dig deeper" into the data or find hidden patterns and relationships. Despite a lack of depth in the analysis, the resulting knowledge can be potentially useful for improving the system performance, and providing support for marketing decisions. The reports give quick overviews of how a site is being used and require minimal disk space or processing power. Furthermore, in the past few years, many commercial products for log analysis have incorporated a variety of data mining tools to discover deeper relationships and hidden patterns in the usage data.

Another form of analysis on integrated usage data is Online Analytical Processing (OLAP). OLAP provides a more integrated framework for analysis with a higher degree of flexibility. As indicated in Figure 1.4, the data source for OLAP analysis is a multi-dimensional data warehouse which integrates usage, content, and e-commerce data at different levels of aggregation for each dimension. OLAP tools allow changes in aggregation levels along each dimensions during the analysis.

Indeed, the server log data, itself, can be stored in a multi-dimensional data structure for OLAP analysis [Zaiane et al., 1998]. Analysis dimensions in such a structure can be based on various fields available in the log files, and may include time duration, domain, requested resource, user agent, referrers, and so on. This allows the analysis to be performed, for example, on portions of the log related to a specific time interval, or at a higher level of abstraction with respect to the URL path structure. The integration of e-commerce data in the data warehouse can enhance the ability of OLAP tools to derive important business intelligence metrics. For example, in [Buchner and Mulvenna, 1999], an integrated Web log data cube was proposed which incorporates customer and product data, as well as domain knowledge such as navigational templates and site topology.

OLAP tools, by themselves, do not automatically discover usage patterns in the data. In fact, the ability to find patterns or relationships in the data depends solely on the effectiveness of the OLAP queries performed against the data warehouse. However, the output from this process can be used as the input for a variety of data mining algorithms. In the following sections we focus specifically on various data mining and pattern discovery techniques that are commonly performed on Web usage data, and we will discuss some approaches for using the discovered patterns for Web personalization.

1.3.2 Data Mining Tasks for Web Usage Data

We now focus on specific data mining and pattern discovery tasks that are often employed when dealing with Web usage data. Our goal is not to give detailed descriptions of all applicable data mining techniques, but to provide some relevant background information and to illustrate how some of these techniques can be applied to Web usage data. In the next section, we present several approaches to leverage the discovered patterns for predictive Web usage mining applications such as personalization.

As noted earlier, preprocessing and data transformation tasks ultimately result in a set of n pageviews, $P = \{p_1, p_2, \cdots, p_n\}$, and a set of m user transactions, $T = \{t_1, t_2, \cdots, t_m\}$, where each $t_i \in T$ is a subset of P. Each transaction t as an l-length sequence of ordered pairs: $t = \langle (p_1^t, w(p_1^t)), (p_2^t, w(p_2^t)), \cdots, (p_l^t, w(p_l^t)) \rangle$, where each $p_i^t = p_j$ for some $j \in \{1, \cdots, n\}$, and $w(p_i^t)$ is the weight associated with pageview p_i^t in the transaction t.

Given a set of transactions as described above, a variety of unsupervised knowledge discovery techniques can be applied to obtain patterns. Techniques such as clustering of transactions (or sessions) can lead to the discovery of important user or visitor segments. Other techniques such as item (e.g., pageview) clustering, association rule mining [Agarwal et al., 1999; Agrawal and Srikant, 1994], or sequential pattern discovery [Agrawal and Srikant, 1995] can be used to find important relationships among items based on the navigational patterns of users in the site. In the cases of clustering and association rule discovery, generally, the ordering relation among the pageviews is not taken into account, thus a transaction is viewed as a set (or, more generally, as a bag) of pageviews $s_t = \{p_i^t \mid 1 \leq i \leq l \text{ and } w(p_i^t) = 1\}$. In the case of sequential patterns, however, we need to preserve the ordering relationship among the pageviews within transactions, in order to effectively model users' navigational patterns.

1.3.2.1 Association Rules

Association rules capture the relationships among items based on their patterns of co-occurrence across transactions (without considering the ordering of items). In the case of Web transactions, association rules capture relationships among pageviews based on the navigational patterns of users. Most common approaches to association discovery are based on the Apriori algorithm [Agrawal and Srikant, 1994, 1995] that follows a generate-and-test methodology. This algorithm finds groups of items (pageviews appearing in the preprocessed log) occurring frequently together in many transactions (i.e., satisfying a user specified minimum support threshold). Such groups of items are referred to as frequent itemsets.

Given a transaction T and a set $I = \{I_1, I_2, \ldots, I_k\}$ of frequent itemsets over T. The *support* of an itemset $I_i \in I$ is defined as

$$\sigma(I_i) = \frac{|\{t \in T : I_i \subseteq t\}|}{|T|}$$

An important property of support in the Apriori algorithm is its downward closure: if an itemset does not satisfy the minimum support criteria, then neither do any of its supersets. This property is essential for pruning the search space during each iteration of the Apriori algorithm. Association rules which satisfy a minimum confidence threshold are then generated from the frequent itemsets. An association rule r is an expression of the form $X \Rightarrow Y$ (σ_r, α_r), where X and Y are itemsets, $\sigma_r = \sigma(X \cup Y)$ is the support of $X \cup Y$ representing the probability that X and Y occur together in a transaction. The confidence for the rule r, α_r, is given by $\sigma(X \cup Y)/\sigma(X)$ and represents the conditional probability that Y occurs in a transaction given that X has occured in that transaction.

The discovery of association rules in Web transaction data has many advantages. For example, a high-confidence rule such as {special-offers/, /products/software/} ⇒ {shopping-cart/} might provide some indication that a promotional campaign on software products is positively affecting online sales. Such rules can also be used to optimize the structure of the site. For example, if a site does not provide direct linkage between two pages A and B, the discovery of a rule {A} ⇒ {B} would indicate that providing a direct hyperlink might aid users in finding the intended information.

The result of association rule mining can be used in order to produce a model for recommendation or personalization systems [Fu et al., 2000; Lin et al., 2002; Mobasher et al., 2001a; Sarwar et al., 2000b]. The top-N recommender systems proposed in [Sarwar et al., 2000b] uses the association rules for making recommendations. First all association rules are discovered on the purchase information. Customer's historical purchase information then is matched against the left-hand-side of the rule in order to find all rules supported by a customer. All right-hand side items from the supported rules are sorted by confidence and the first N highest ranked items are selected as recommendation set. One problem for association rule recommendation systems is that a system cannot give any recommendations when the dataset is sparse. In [Fu et al., 2000] two potential solutions to this problem were proposed. The first solution is to rank all discovered rules calculated by the degree of intersection between the left-hand-side of rule and a user's active session and then to generate the top k recommendations. The second solution is to utilize collaborative filtering technique: the system finds "close neighbors" who have similar interest to a target user and makes recommendations based on the close neighbor's history. In [Lin et al., 2002] a collaborative recommendation system was presented using association rules. The proposed mining algorithm finds an appropriate number of rules for each target user by automatically selecting the minimum support. The recommendation engine generates association rules for each user among both users and items. Then it gives recommendations based on user association if a user minimum support is greater than a threshold. Otherwise, it uses article association.

In [Mobasher et al., 2001a] a scalable framework for recommender systems using association rule mining was proposed. The recommendation algorithm uses an efficient data structure for storing frequent itemsets, and produces recommendations in real-time, without the need to generate all association rules from frequent itemsets. We discuss this recommendation algorithm based on association rule mining in more detail in Section 1.3.2.3.

A problem with using a global minimum support threshold in association rule mining is that the discovered patterns will not include "rare" but important items which may not occur frequently in the transaction data. This is particularly important when dealing with Web usage data, it is often the case that references to deeper content or product-oriented pages occur far less frequently that those of top level navigation-oriented pages. Yet, for effective Web personalization, it is important to capture patterns and generate recommendations that contain these items. Liu et al. [Liu et al., 1999] proposed a mining method with multiple minimum supports that allows users to specify different support values for different items. In this method, the support of an itemset is defined as the minimum support of all items contained in the itemset. The specification of multiple minimum supports allows frequent itemsets to potentially contain rare items which are nevertheless deemed important. It has been shown that the use of multiple support association rules in the context of Web personalization can be useful in dramatically increasing the coverage (recall) of recommendations while maintaining a reasonable precision [Mobasher et al., 2001a] .

1.3.2.2 Sequential and Navigational Patterns

Sequential patterns (SP's) in Web usage data capture the Web page trails that are often visited by users, in the order that they were visited. Sequential patterns are those sequences of items that frequently occur in a sufficiently large proportion of transactions. A sequence $\langle s_1, s_2, \ldots, s_n \rangle$ occurs in a transaction $t = \langle p_1, p_2, \ldots, p_m \rangle$ (where $n \leq m$) if there exist n positive integers $1 \leq a_1 < a_2 < \ldots < a_n \leq m$, and $s_i = p_{a_i}$ for all i. We say that $\langle cs_1, cs_2, \ldots, cs_n \rangle$ is a contiguous sequence in t if there exists an integer $0 \leq b \leq m - n$, and $cs_i = p_{b+i}$ for all $i = 1$ to n. In a contiguous sequential pattern (CSP), each pair of adjacent elements, s_i and s_{i+1}, must appear consecutively in a transaction t which supports the pattern, while a sequential pattern can represent non-contiguous frequent sequences in the underlying set of transactions.

Given a transaction set T and a set $S = \{S_1, S_2, \ldots, S_n\}$ of frequent sequential (respectively, contiguous sequential) pattern over T, the support of each S_i is defined as follows:

$$\sigma(S_i) = \frac{|\{t \in T : S_i \text{ is (contiguous) subsequence of } t\}|}{|T|}.$$

The confidence of the rule $X \Rightarrow Y$, where X and Y are (contiguous) sequential patterns, is defined as

$$\alpha(X \Rightarrow Y) = \frac{\sigma(X \circ Y)}{\sigma(X)},$$

where $\circ$ denotes the concatenation operator. Note that the support thresholds for SP's and CSP's also satisfy downward closure property, i.e., if a (contiguous) sequence of items, S, has any subsequence that does not satisfy the minimum support criteria, then S does not have minimum support. The Apriori algorithm used in association rule mining can also be adopted to discover sequential and contiguous sequential patterns. This is normally accomplished by changing the definition of support to be based on the frequency of occurrences of subsequences of items rather than subsets of items [Agrawal and Srikant, 1995].

In the context of Web usage data, CSP's can be used to capture frequent navigational paths among user trails [Spiliopoulou and Faulstich, 1999; Schechter et al., 1998]. In contrast, items appearing in SP's, while preserving the underlying ordering, need not be adjacent, and thus they represent more general navigational patterns within the site. Frequent item sets, discovered as part of association rule mining, represent the least restrictive type of navigational patterns, since they focus on the presence of items rather than the order in which they occur within user session.

The view of Web transactions as sequences of pageviews allows us to employ a number of useful and well-studied models which can be used to discover or analyze user navigation patterns. On such approach is to model the navigational activity in the Web site as a Markov chain. In general, a Markov model is characterized by a set of states $\{s_1, s_2, \ldots, s_n\}$ and a transition probability matrix

$$\{p_{1,1}, \ldots, p_{1,n}, \ldots, p_{2,1}, \ldots, p_{2,n}, \ldots, p_{n,1}, \ldots, p_{n,n}\},$$

where $p_{i,j}$ represents the probability of a transition from state s_i to state s_j.

Markov models are especially suited for predictive modeling based on contiguous sequences of events. Each state represents a contiguous subsequence of prior events. The order of the Markov model corresponds to the number of prior events used in predicting a future event. So, a kth-order Markov model predicts the probability of next event by looking the past k events. Given a set of all paths R, the probability of reaching a state s_j from a state s_i via a (non-cyclic) path $r \in R$ is given by: $p(r) = \prod p_{k,k+1}$, where k

Transactions	Frequency
A, B, E	10
B, D, B, C	4
B, C, E	10
A, B, E, F	6
A, D, B	12
B, D, B, E	8

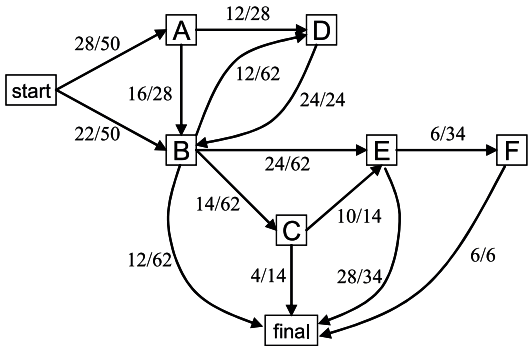

Figure 1.5: An example of modeling navigational trails as a Markov chain.

ranges from i to $j-1$. The probability of reaching s_j from s_i is the sum over all paths: $p(j|i) = \sum_{r \in R} p(r)$.

In the context of Web transactions, Markov chains can be used to model transition probabilities between pageviews. In Web usage analysis, they have been proposed as the underlying modeling machinery for Web prefetching applications or to minimize system latencies [Deshpande and Karypis, 2001; Palpanas and Mendelzon, 1999; Pitkow and Pirolli, 1999; Sarukkai, 2000]. Such systems are designed to predict the *next* user action based on a user's previous surfing behavior. In the case of first-order Markov models, only user's current action is considered in predicting the next action, thus each state represents a single pageview in the user's transaction. Markov models can also be used to discover high-probability user navigational trails in a Web site. For example, in [Borges and Levene, 1999] the user sessions are modeled as a hypertext probabilistic grammar (or alternatively, an absorbing Markov chain) whose higher probability paths correspond to the user's preferred trails. An algorithm is provided to efficiently mine such trails from the model.

As an example of how Web transactions can be modeled as a Markov model, consider the set of Web transaction given in Figure 1.5 (left). The Web transactions involve pageviews A, B, C, D, and E. For each transaction the frequency of occurrences of that transaction in the data is given in table's second column (thus there are a total of 50 transactions in the data set). The (absorbing) Markov model for this data is also given in Figure 1.5 (right). The transitions from the "start" state represent the prior probabilities for transactions starting with pageviews A and B. The transitions into the "final" state represent the probabilities that the paths end with the specified originating pageviews. For example, the transition probability from the state A to B is $16/28 = 0.57$ since out of the 28 occurences of A in transactions, in 16 cases, B occurs immediately after A.

Higher-order Markov models generally provide a higher prediction accuracy. However, this is usually at the cost of lower coverage and much higher model complexity due to the larger number of states. In order to remedy the coverage and space complexity problems, Pitkow and Pirolli [1999] proposed all-kth-order Markov models (for coverage improvement) and a new state reduction technique, called longest repeating subsequences (LRS) (for reducing model size). The use of all-kth-order Markov models generally requires the generation of separate models for each of the k orders: if the model cannot make a prediction using the kth order, it will attempt to make a prediction by incrementally decreasing the model order. This scheme can easily lead to even higher space complexity since it requires the representation of all possible states for each k. Deshpande and Karypis [2001], propose selective Markov models, introducing several schemes in order to tackle the model complexity problems with all-kth-order Markov models. The proposed schemes involve pruning

Transactions	Frequency
A, B, E	10
B, D, B, C	4
B, C, E	10
A, B, E, F	6
A, D, B	12
B, D, B, E	8

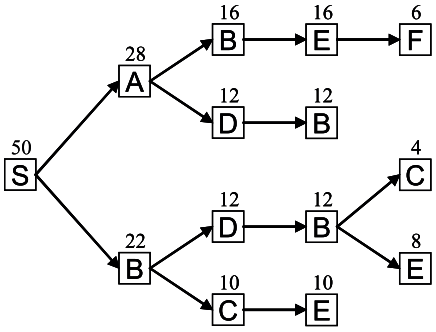

Figure 1.6: An example of modeling navigational trails in an aggregate tree.

the model based on criteria such as support, confidence, and error rate. In particular, The support-pruned Markov models eliminate all states with low support determined by a minimum frequency threshold.

Another way of efficiently representing navigational trails is by inserting each trail into a trie structure [Spiliopoulou and Faulstich, 1999]. It is also possible to insert frequent sequences (after or during sequential pattern mining) into a trie structure [Pei et al., 2000]. A well-known example of this approach is the notion of aggregate tree introduced as part of the WUM (Web Utilization Miner) system [Spiliopoulou and Faulstich, 1999]. The aggregation service of WUM extracts the transactions from a collection of Web logs, transforms them into sequences, and merges those sequences with the same prefix into the aggregate tree (a trie structure). Each node in the tree represents a navigational subsequence from the root (an empty node) to a page and is annotated by the frequency of occurrences of that subsequence in the transaction data (and possibly other information such as markers to distinguish among repeat occurences of the corresponding page in the subsequence). WUM uses a powerful mining query language, called MINT, to discover generalized navigational patterns from this trie structure. MINT includes mechanism to specify sophisticated constraints on pattern templates, such as wildcards with user-specified boundaries, as well as other statistical thresholds such as support and confidence.

As an example, again consider the set of Web transaction given in the previous example. Figure 1.6 shows a simplified version of WUM's aggregate tree structure derived from these transactions. The advantage of this approach is that the search for navigational patterns can be performed very efficiently and the confidence and support for the sequential patterns can be readily obtained from the node annotations in the tree. For example, consider the navigational sequence ⟨A, B, E, F⟩. The support for this sequence can be computed as the support of F divided by the support of first pageview in the sequence, A, which is 6/28 = 0.21, and the confidence of the sequence is the support of F divided by support of its parent, E, or 6/16 = 0.375. The disadvantage of this approach is the possibly high space complexity, especially in a site with many dynamically generated pages.

1.3.2.3 Clustering Approaches

In general, there are two types of clustering that can be performed on usage transaction data: clustering the transactions (or users), themselves, or clustering pageviews. Each of these approaches are useful in different applications, and in particular, both approaches can be used for Web personalization. There has been a significant amount of work on the applications of clustering in Web usage mining, e-marketing, personalization, and collaborative filtering.

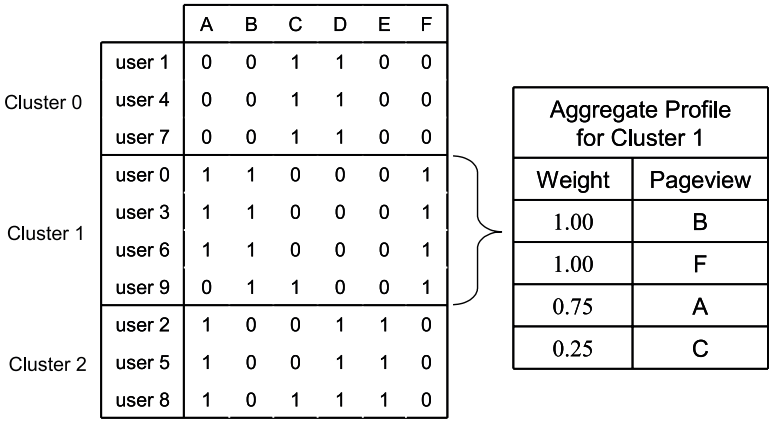

Figure 1.7: An example of deriving aggregate usage profiles from transaction clusters.

For example, an algorithm called PageGather has been used to discover significant groups of pages based on user access patterns [Perkowitz and Etzioni, 1998]. This algorithm uses, as its basis, clustering of pages based the Clique (complete link) clustering technique. The resulting clusters are used to automatically synthesize alternative static index pages for a site, each reflecting possible interests of one user segment. Clustering of user rating records has also been used as a prior step to collaborative filtering in order to remedy the scalability problems of the k-nearest-neighbor algorithm [O'Conner and Herlocker, 1999]. Both transaction clustering and pageview clustering have been used as an integrated part of a Web personalization framework based on Web usage mining [Mobasher et al., 2002b].

Given the mapping of user transactions into a multi-dimensional space as vectors of pageviews (i.e., the matrix TP in Section 1.2.2), standard clustering algorithms, such as k-means, generally partition this space into groups of transactions that are close to each other based on a measure of distance or similarity among the vectors. Transaction clusters obtained in this way can represent user or visitor segments based on their navigational behavior or other attributes that have been captured in the transaction file. However, transaction clusters by themselves are not an effective means of capturing an aggregated view of common user patterns. Each transaction cluster may potentially contain thousands of user transactions involving hundreds of pageview references. The ultimate goal in clustering user transactions is to provide the ability to analyze each segment for deriving business intelligence, or to use them for tasks such as personalization.

One straight forward approach in creating an aggregate view of each cluster is to compute the centroid (or the mean vector) of each cluster. The dimension value for each pageview in the mean vector is computed by finding the ratio of the sum of the pageview's weights across transactions to the total number of transactions in the cluster. If pageview weights in the original transactions are binary, then the dimension value of a pageview p in a cluster centroid represents the percentage of transactions in the cluster in which p occurs. Thus, the centroid dimension value of p provides a measure of its significance in the cluster. Pageviews in the centroid can be sorted according to these weights and lower weight pageviews can be filtered out. The resulting set of pageview-weight pairs can be viewed as an "aggregate usage profile" representing the interests or behavior of a significant group of users. We discuss how such aggregate profiles can be used for personalization in the next section.

As an example, consider the transaction data depicted in Figure 1.7 (left). In this case, the feature (pageview) weights in each transaction vector is binary. We assume that the data has already been clustered using a standard clustering algorithm such as k-means, resulting

in three clusters of user transactions. The table in the right portion of Figure 1.7 shows the aggregate profile corresponding to cluster 1. As indicated by the pageview weights, pageviews B and F are the most significant pages characterizing common interests of users in this segment. Pageview C, however only appears in one transaction and might be removed given a filtering threshold greater than 0.25.

Note that it is possible to apply a similar procedure to the transpose of the matrix TP, resulting a collection of pageview clusters. However, traditional clustering techniques, such as distance-based methods generally cannot handle this type clustering. The reason is that instead of using pageviews as dimensions, the transactions must be used as dimensions, whose number is in tens to hundreds of thousands in a typical application. Furthermore, dimensionality reduction in this context may not be appropriate, as removing a significant number of transactions may result in losing too much information. Similarly, the clique-based clustering approach of PageGather algorithm [Perkowitz and Etzioni, 1998] discussed above can be problematic, since finding all maximal cliques in very large graphs is not, in general, computationally feasible.

One approach that has been shown to be effective in this type (i.e. item-based) clustering is Association Rule Hypergraph partitioning (ARHP) [Han et al., 1998]. ARHP can efficiently cluster high-dimensional data sets and provides automatic filtering capabilities. In the ARHP, first association rule mining is used to discover a set I of frequent itemsets among the pageviews in P. These itemsets are used as hyperedges to form a hypergraph $H = \langle V, E \rangle$, where $V \subseteq P$ and $E \subseteq I$. A hypergraph is an extension of a graph in the sense that each hyperedge can connect more than two vertices. The weights associated with each hyperedge can be computed based on a variety of criteria such as the confidence of the association rules involving the items in the frequent itemset, the support of the itemset, or the "interest" of the itemset.

The hypergraph H is recursively partitioned until a stopping criterion for each partition is reached resulting in a set of clusters C. Each partition is examined to filter out vertices that are not highly connected to the rest of the vertices of the partition. The connectivity of vertex v (a pageview appearing in the frequent itemset) with respect to a cluster c is defined as:

$$conn(v, c) = \frac{\sum_{e \subseteq c, v \in e} weight(e)}{\sum_{e \subseteq c} weight(e)}.$$

A high connectivity value suggests that the vertex has strong edges connecting it to other vertices in the partition. The vertices with connectivity measure greater than a given threshold value are considered to belong to the partition, and the remaining vertices are dropped from the partition. The connectivity value of an item (pageviews) defined above is important also because it is used as the primary factor in determining the weight associated with that item within the resulting aggregate profile. This approach has also been used in the context of Web personalization [Mobasher et al., 2002b], and its performance in terms of recommendation effectiveness has been compared to the transaction clustering approach discussed above.

Clustering can also be applied to Web transactions viewed as sequences rather than as vectors. For example in [Banerjee and Ghosh, 2001] a graph-based algorithm was introduced to cluster Web transactions based on a function of longest common subsequences. The novel similarity metric used for clustering takes into account both the time spent on pages as well as a significance weight assigned to pages.

Finally, we also observe that the clustering approaches such as those discussed in this section can also be applied to content data, or to the integrated content-enhanced transactions described in Section 1.2.2. For example, the results of clustering user transactions can be combined with "content profiles" derived from the clustering of text features (terms

or concepts) in pages [Mobasher et al., 2000b]. The feature clustering is accomplished by applying a clustering algorithm to the transpose of the pageview-feature matrix PF, defined earlier. This approach treats each feature as a vector over the space of pageviews. Thus the centroid of a feature cluster can be viewed as a set (or vector) of pageviews with associated weights. This representation is similar to that of usage profiles discussed above, however, in this case the weight of a pageview in a profile represents the prominence of the features in that pageview that are associated with the corresponding cluster. The combined set of content and usage profiles can then be used seamlessly for more effective Web personalization. One advantage of this approach is that it solves the "new item" problem which often plagues purely usage-based or collaborative approaches: when a new item (e.g., page or product) is recently added to the site, it is not likely to appear in usage profiles due to the lack of user ratings or access to that page, but it may still be recommended according to its semantic attributes captured by the content profiles.

1.4 Using the Discovered Patterns for Personalization

As noted in the Introduction section, the goal of the recommendation engine is to match the active user session with the aggregate profiles discovered through Web usage mining, and to recommend a set of objects to the user. We refer to the set of recommended object (represented by pageviews) as the recommendation set. In this section we explore the recommendation procedures to perform the matching between the discovered aggregate profiles and an active user's session. Specifically, we present several effective recommendation algorithms based on clustering (which can be seen as an extension of standard kNN-based collaborative filtering), association rule mining (AR), and sequential pattern (SP) or contiguous sequential pattern (CSP) discovery. In the cases of AR, SP, and CSP, we consider efficient and scalable data structures for storing frequent itemset and sequential patterns, as well as a recommendation generation algorithms that use these data structures to directly produce real-time recommendations (without the apriori generation of rule).

Generally, only a portion of current user's activity is used in the recommendation process. Maintaining a history depth is necessary because most users navigate several paths leading to independent pieces of information within a session. In many cases these sub-sessions have a length of no more than three or four references. In such a situation, it may not be appropriate to use references a user made in a previous sub-session to make recommendations during the current sub-session. We can capture the user history depth within a sliding window over the current session. The sliding window of size n over the active session allows only the last n visited pages to influence the recommendation value of items in the recommendation set. For example, if the current session (with a window size of 3) is $\langle A, B, C\rangle$, and the user accesses the pageview D, then the new active session becomes $\langle B, C, D\rangle$. We call this sliding window, the user's active session window.

Structural characteristics of the site or prior domain knowledge can also be used to associate an additional measure of significance with each pageview in the user's active session. For instance, the site owner or the site designer may wish to consider certain page types (e.g., content versus navigational) or product categories as having more significance in terms of their recommendation value. In this case, significance weights can be specified as part of the domain knowledge.

1.4.1 The kNN-Based Approach

Collaborative filtering based on the k-Nearest-Neighbor (kNN) approach involves comparing the activity record for a target user with the historical records of other users in order to find the top k users who have similar tastes or interests. The mapping of a visitor record to its neighborhood could be based on similarity in ratings of items, access to similar content or pages, or purchase of similar items. The identified neighborhood is then used to recommend items not already accessed or purchased by the active user. Thus, there are two primary phases in collaborative filtering: the neighborhood formation phase and the recommendation phase.

In the context of personalization based on Web usage mining, kNN involves measuring the similarity or correlation between the active session $\vec{s}$ and each transaction vector $\vec{t}$ (where $t \in T$). The top k most similar transactions to $\vec{s}$ are considered to be the neighborhood for the session s, which we denote by $NB(s)$ (taking the size k of the neighborhood to be implicit):

$$NB(s) = \{\vec{t}_{s_1}, \vec{t}_{s_2}, \cdots, \vec{t}_{s_k}\}.$$

A variety of similarity measures can be used to find the nearest neighbors. In traditional collaborative filtering domains (where feature weights are item ratings on a discrete scale), the Pearson r correlation coefficient is commonly used. This measure is based on the deviations of users' ratings on various items from their mean ratings on all rated items. However, this measure is may not be appropriate when the primary data source is clickstream data (particularly in the case of binary weights). Instead we use the cosine coefficient, commonly used in information retrieval, which measures the cosine of the angle between two vectors. The cosine coefficient can be computed by normalizing the dot product of two vectors with respect to their vector norms. Given the active session $\vec{s}$ and a transaction $\vec{t}$, the similarity between them is obtained by:

$$sim(\vec{t}, \vec{s}) = \frac{\vec{t} \cdot \vec{s}}{|\vec{t}| \times |\vec{s}|}.$$

In order to determine which items (not already visited by the user in the active session) are to be recommended, a recommendation score is computed for each pageview $p_i \in P$ based on the neighborhood for the active session. Two factors are used in determining this recommendation score: the overall similarity of the active session to the neighborhood as a whole, and the average weight of each item in the neighborhood.

First we compute the mean vector (centroid) of $NB(s)$. Recall that, the dimension value for each pageview in the mean vector is computed by finding the ratio of the sum of the pageview's weights across transactions to the total number of transactions in the neighborhood. We denote this vector by $cent(NB(s))$. For each pageview p in the neighborhood centroid, we can now obtain a recommendation score as a function of the similarity of the active session to the centroid vector and the weight of that item in this centroid. Here we have chosen to use the following function, denoted by $rec(\vec{s}, p)$:

$$rec(\vec{s}, p) = \sqrt{weight(p, NB(s)) \times sim(\vec{s}, cent(NB(s)))},$$

where $weight(p, NB(s))$ is the mean weight for pageview p in the neighborhood as expressed in the centroid vector. If the pageview p is in the current active session, then its recommendation value is set to zero.

If a fixed number N of recommendations are desired, then the top N items with the highest recommendation scores are considered to be part of the recommendation set. In our

implementation, we normalize the recommendation scores for all pageviews in the neighborhood (so that the maximum recommendation score is 1), and return only those which satisfy a threshold test. In this way, we can compare the performance of kNN across different recommendation thresholds.

1.4.2 Using Clustering For Personalization

The transaction clustering approach discussed in Section 1.3.2 will result in a set $TC = \{c_1, c_2, \cdots, c_k\}$ of transaction clusters, where each c_i is a subset of the set of transactions T. As noted in that section, from each transaction cluster we can derive and aggregate usage profile by computing the centroid vectors for that cluster. We call this method PACT (Profile Aggregation Based on Clustering Transactions) [Mobasher et al., 2002b].

In general, PACT can consider a number of other factors in determining the item weights within each profile, and in determining the recommendation scores. These additional factors may include the link distance of pageviews to the current user location within the site or the rank of the profile in terms of its significance. However, to be able to consistently compare the performance of the clustering-based approach to that of kNN, we restrict the item weights to be the mean feature values of the transaction cluster centroids. In this context, the only difference between PACT and the kNN-based approach is that we discover transaction clusters offline and independent of a particular target user session.

To summarize the PACT method, given a transaction cluster c, we construct an aggregate usage profile pr_c as a set of pageview-weight pairs:

$$pr_c = \{\langle p, weight(p, pr_c)\rangle \mid p \in P, weight(p, pr_c) \geq \mu\},$$

where the significance weight, $weight(p, pr_c)$, of the pageview p within the usage profile pr_c is:

$$weight(p, pr_c) = \frac{1}{|c|} \cdot \sum_{t \in c} w_p^t,$$

and w_p^t is the weight of pageview p in transaction $t \in c$. The threshold parameter μ is used to prune out very low support pageviews in the profile. An example of deriving aggregate profiles from transaction clusters was given in the previous section (see Figure 1.7).

This process results in a number of aggregate profiles each of which can, in turn, be represented as a vector in the original n-dimensional space of pageviews. The recommendation engine can compute the similarity of an active session $\vec{s}$ with each of the discovered aggregate profiles. The top matching profile is used to produce a recommendation set in a manner similar to that for the kNN approach discussed above. If $\vec{pr}$ is the vector representation of the top matching profile pr, we compute the recommendation score for the pageview p by

$$rec(\vec{s}, p) = \sqrt{weight(p, pr) \times sim(\vec{s}, \vec{pr})},$$

where $weight(p, pr)$ is the weight for pageview p in the profile pr. As in the case of kNN, if the pageview p is in the current active session, then its recommendation value is set to zero.

Clearly, PACT will result in dramatic improvement in scalability and computational performance, since most of the computational cost is incurred during the offline clustering phase. We would expect, however, that this decrease in computational costs be accompanied also by a decrease in recommendation effectiveness. Experimental results [Mobasher et al., 2001b] have shown that through proper data preprocessing and using some of the data transformation steps discussed earlier, we can dramatically improve the recommendation effectiveness when compared to kNN.

T1: $\{ABDE\}$
T2: $\{ABECD\}$
T3: $\{ABEC\}$
T4: $\{BEBAC\}$
T5: $\{DABEC\}$

Figure 1.8: Sample Web Transactions involving pageviews A, B, C, D and E.

Size 1	Size 2	Size 3	Size 4
$\{A\}(5)$	$\{A,B\}(5)$	$\{A,B,C\}(4)$	$\{A,B,C,E\}(4)$
$\{B\}(6)$	$\{A,C\}(4)$	$\{A,B,E\}(5)$	
$\{C\}(4)$	$\{A,E\}(5)$	$\{A,C,E\}(4)$	
$\{E\}(5)$	$\{B,C\}(4)$	$\{B,C,E\}(4)$	
	$\{B,E\}(5)$		
	$\{C,E\}(4)$		

Figure 1.9: Example of discovered frequent itemsets.

It should be noted that the pageview clustering approach discussed in Section 1.3.2 can also be used with the recommendation procedure detailed above. In that case, also, the aggregate profiles are represented as collections of pageview-weight pairs, and thus can be viewed as vectors over the space of pageviews in the data.

1.4.3 Using Association Rules for Personalization

The recommendation engine based on association rules matches the current user session window with frequent itemsets to find candidate pageviews for giving recommendations. Given an active session window w and a group of frequent itemsets, we only consider all the frequent itemsets of size $|w|+1$ containing the current session window. The recommendation value of each candidate pageview is based on the confidence of the corresponding association rule whose consequent is the singleton containing the pageview to be recommended.

In order to facilitate the search for itemsets (of size $|w| + 1$) containing the current session window w, the frequent itemsets are stored in a directed acyclic graph, here called a Frequent Itemset Graph. The Frequent Itemset Graph is an extension of the lexicographic tree used in the "tree projection algorithm" [Agarwal et al., 1999]. The graph is organized into levels from 0 to k, where k is the maximum size among all frequent itemsets. Each node at depth d in the graph corresponds to an itemset, I, of size d and is linked to itemsets of size $d+1$ that contain I at level $d+1$. The single root node at level 0 corresponds to the empty itemset. To be able to match different orderings of an active session with frequent itemsets, all itemsets are sorted in lexicographic order before being inserted into the graph. The user's active session is also sorted in the same manner before matching with patterns.

Given an active user session window w, sorted in lexicographic order, a depth-first search of the Frequent Itemset Graph is performed to level $|w|$. If a match is found, then the children of the matching node n containing w are used to generate candidate recommendations. Each child node of n corresponds to a frequent itemset $w \cup \{p\}$. In each case, the pageview p is added to the recommendation set if the support ratio $\sigma(w \cup \{p\})/\sigma(w)$ is greater than or equal to α, where α is a minimum confidence threshold. Note that $\sigma(w \cup \{p\})/\sigma(w)$ is the confidence of the association rule $w \Rightarrow \{p\}$. The confidence of this rule is also used as the recommendation score for pageview p. It is easy to observe that in this algorithm the

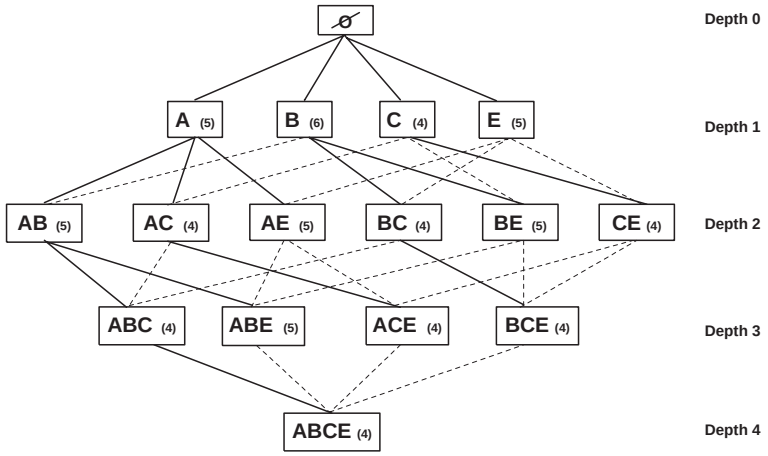

Figure 1.10: An example of a Frequent Itemsets Graph.

search process requires only $O(|w|)$ time given active session window w.

To illustrate the process, consider the example transaction set given in Figure 1.8. Using these transactions, the Apriori algorithm with a frequency threshold of 4 (minimum support of 0.8) generates the itemsets given in Figure 1.9. Figure 1.10 shows the Frequent Itemsets Graph constructed based on the frequent itemsets in Figure 1.9. Now, given user active session window $\langle B, E\rangle$, the recommendation generation algorithm finds items A and C as candidate recommendations. The recommendation scores of item A and C are 1 and 4/5, corresponding to the confidences of the rules $\{B, E\} \rightarrow \{A\}$ and $\{B, E\} \rightarrow \{C\}$, respectively.

1.4.4 Using Sequential Patterns for Personalization

The recommendation algorithm based on association rules can be adopted to work also with sequential (respectively, contiguous sequential) patterns. In this case, we focus on frequent (contiguous) sequences of size $|w| + 1$ whose prefix contains an active user session w. The candidate pageviews to be recommended are the last items in all such sequences. The recommendation values are based on the confidence of the patterns. If the confidence satisfies a threshold requirement, then the candidate pageviews are added to the recommendation set.

A simple trie structure, which we call Frequent Sequence Trie (FST), can be used to store both the sequential and contiguous sequential patterns discovered during the pattern discovery phase. The FST is organized into levels from 0 to k, where k is the maximal size among all sequential (respectively, contiguous sequential) patterns. There is the single root node at depth 0 containing the empty sequence. Each non-root node N at depth d contains an item s_d and representing a frequent sequence $\langle s_1, s_2, \ldots, s_{d-1}, s_d\rangle$ whose prefix $\langle s_1, s_2, \ldots, s_{d-1}\rangle$ is the pattern represented by the parent node of N at depth $d - 1$. Furthermore, along with each node we store the support (or frequency) value of the corresponding pattern. The confidence of each pattern (represented by a non-root node in the FST) is obtained by dividing the support of the current node by the support of its parent node.

The recommendation algorithm based on sequential and contiguous sequential patterns has a similar structure as the algorithm based on association rules. For each active session window $w = \langle w_1, w_2, \ldots, w_n\rangle$, we perform a depth-first search of the FST to level n. If a

Size 1	Size 2	Size 3
$\langle A\rangle(5)$	$\langle A,B\rangle(4)$	$\langle A,B,E\rangle(4)$
$\langle B\rangle(6)$	$\langle A,C\rangle(4)$	$\langle A,E,C\rangle(4)$
$\langle C\rangle(4)$	$\langle A,E\rangle(4)$	
$\langle E\rangle(5)$	$\langle B,C\rangle(4)$	
	$\langle B,E\rangle(5)$	
	$\langle C,E\rangle(4)$	

Figure 1.11: Example of discovered sequential patterns.

match is found, then the children of the matching node N are used to generate candidate recommendations. Given a sequence $S = \langle w_1, w_2, \ldots, w_n, p\rangle$ represented by a child node of N, the item p is then added to the recommendation set as long as the confidence of S is greater than or equal to the confidence threshold. As in the case of frequent itemset graph, the search process requires $O(|w|)$ time given active session window size $|w|$.

Size 1	Size 2
$\langle A\rangle(5)$	$\langle A,B\rangle(4)$
$\langle B\rangle(6)$	$\langle B,E\rangle(4)$
$\langle C\rangle(4)$	
$\langle E\rangle(5)$	

Figure 1.12: Example of discovered contiguous sequential patterns.

To continue our example, Figures 1.11 and 1.12 show the frequent sequential patterns and frequent contiguous sequential patterns with a frequency threshold of 4 over the example transaction set given in Figure 1.8. Figures 1.13 and 1.14 show the trie representation of the sequential and contiguous sequential patterns listed in the Figure 1.11 and 1.12, respectively. The sequential pattern $\langle A, B, E\rangle$ appears in the Figure 1.13 because it is the subsequence of 4 transactions: T_1, T_2, T_3 and T_5. However, $\langle A, B, E\rangle$ is not a frequent contiguous sequential pattern since only three transactions (T_2, T_3 and T_5) contain the contiguous sequence $\langle A, B, E\rangle$. Give a user's active session window $\langle A, B\rangle$, the recommendation engine using sequential patterns finds item E as a candidate recommendation. The recommendation score of item E is 1, corresponding to the rule $\langle A, B\rangle \Rightarrow \langle E\rangle$. On the other hand, the recommendation engine using contiguous sequential patterns will, in this case, fails to give any recommendations.

It should be noted that, depending on the specified support threshold, it might be difficult to find large enough itemsets or sequential patterns that could be used for providing recommendations, leading to reduced coverage. This is particularly true for sites with very small average session sizes. An alternative to reducing the support threshold in such cases would be to reduce the session window size. This latter choice may itself lead to some undesired effects since we may not be taking enough of the user's activity history into account. Generally, in the context of recommendation systems, using a larger window size over the active session can achieve better prediction accuracy. But, as in the case of higher support threshold, larger window sizes also lead to lower recommendation coverage.

In order to overcome this problem, we can use the all-kth-order approach discussed in the previous section in the context of Markov chain models. The above recommendation framework for contiguous sequential patterns is essentially equivalent to kth-order Markov models, however, rather than storing all navigational sequences, only frequent sequences

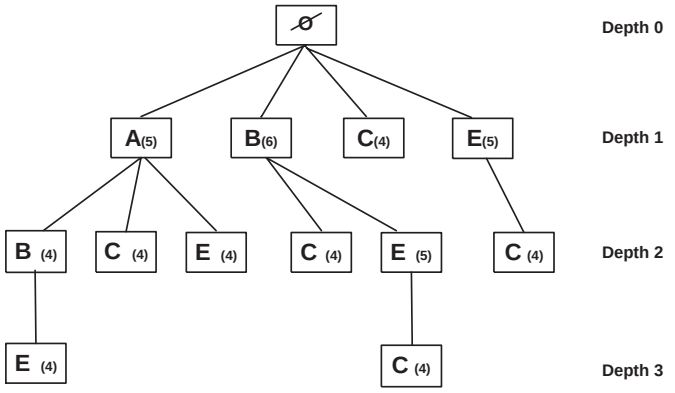

Figure 1.13: Example of a Frequent Sequence Trie (FST).

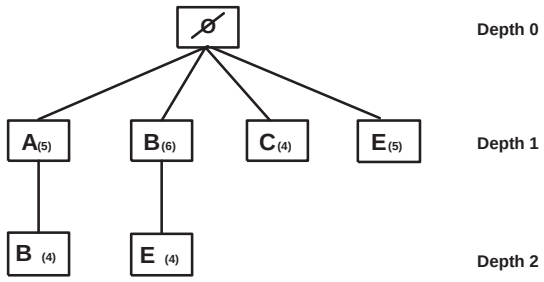

Figure 1.14: Example of an FST for contiguous sequences.

resulting from the sequential pattern mining process are stored. In this sense, the above method is similar to support pruned models described in the previous section [Deshpande and Karypis, 2001], except that the support pruning is performed by the Apriori algorithm in the mining phase. Furthermore, in contrast to standard all-kth-order Markov models, this framework does not require additional storage since all the necessary information (for all values of k) is captured by Frequent Sequence Trie structure described above.

The notion of all-kth-order models can also be easily extended to the context of general sequential patterns and association rule. We extend these recommendation algorithms to generate all-kth-order recommendations as follows. First, the recommendation engine uses the largest possible active session window as an input for recommendation engine. If the engine cannot generate any recommendations, the size of active session window is iteratively decreased until a recommendation is generated or the window size becomes 0.

1.5 Conclusions and Outlook

In this chapter we have attempted to present a comprehensive view of the personalization process based on Web usage mining. The overall framework for this process was depicted in Figures 1.1 and 1.2. In the context of this framework, we have discussed a host of Web usage mining activities necessary for this process, including the preprocessing and integration of data from multiple sources, and pattern discovery techniques that are applied

to the integrated usage data. We have also presented a number of specific recommendation algorithms for combining the discovered knowledge with the current status of a user's activity in a Web site to provide personalized content to a user. The approaches we have detailed show how pattern discovery techniques such as clustering, association rule mining, and sequential pattern discovery, performed on Web usage data, can be leveraged effectively as an integrated part of a Web personalization system.

In this concluding section, we provide a brief discussion of the circumstances under which some of the approaches discussed might provide a more effective alternative to the others. We also identify the primary problems the solutions of which may lead to the creation of the next-generation of more effective and useful Web personalization and Web mining tools.

1.5.1 Which Approach?

Personalization systems are often evaluated based on two statistical measures, namely precision and coverage (also known as recall). These measures are adaptations of similarly named measures often used in evaluating the effectiveness of information retrieval systems. In the context of personalization, precision measures the degree to which the recommendation engine produces accurate recommendations (i.e., the proportion of relevant recommendations to the total number of recommendations), while coverage (or recall) measures the ability of the recommendation engine to produce all of the pageviews that are likely to be visited by the user (i.e., proportion of relevant recommendations to all pageviews that will be visited - according to some evaluation data set). Neither of these measures individually are sufficient to evaluate the performance of the recommendation engine, however, they are both critical. A low precision in this context will likely result in angry customers or visitors who are not interested in the recommended items, while low coverage will result in the inability of the site to produce relevant cross-sell recommendations at critical points in user's interaction with the site. In previous work [Mobasher et al., 2001a, 2002b,a] many of the approaches presented in this chapter have been evaluated based on these measures using real usage data. Here we present a summary of the findings.

In the case of clustering approaches, we have compared the performance of transaction clustering method, PACT, with the pageview clustering approach based on hypergraph partitioning (ARHP) [Mobasher et al., 2002a]. In general, the ARHP approach performs better when the data set is filtered to focus on more "interesting" objects (e.g., content-oriented pages that are situated more deeply within the site). It seems to produce a smaller set of high quality, and more specialized, recommendations, even when a small portion of the user's clickstream is used by the recommendation engine. On the other hand, PACT provides a clear performance advantage when dealing with all the relevant pageviews in the site, particularly as the session window size is increased. Thus, if the goal is to provide a smaller number of highly focused recommendations, then the ARHP approach may be a more appropriate method. This is particularly the case if only specific portions of the site (such as product-related or content pages) are to be personalized. On the other hand, if the goal is to provide a more generalized personalization solution integrating both content and navigational pages throughout the whole site, then using PACT as the underlying aggregate profile generation method seems to provide clear advantages.

More generally, clustering, in contrast to association rule or sequential pattern mining, provides a more flexible mechanism for personalization, even though it does not always lead to highest recommendation accuracy. The flexibility comes from the fact that many inherent attributes of pageviews can be taken into account in the mining process, such as time durations and possibly relational attributes of the underlying objects.

The association rule (AR) models also performs well in the context of personalization. In general, the precision of AR models are lower than the models based on sequential

patterns (SP) and contiguous sequential patterns (CSP), but they often provide much better coverage. Comparison to kNN have shown that all of these techniques outperform kNN in terms of precision. In general, kNN provides better coverage (usually in par with the AR model), but the difference in coverage is diminished if we insist on higher recommendation thresholds (and thus more accurate recommendations).

In general, the SP and the AR models provide the best choices for personalization applications. The CSP model can do better in terms of precision, but the coverage levels are often too low when the goal is to generate as many good recommendations as possible. This last observation about the CSP models, however, does not extend to other predictive applications such as pre-fetching, where the goal is to predict the immediate next action of the user (rather than providing a broader set of recommendations). In this case, the goal is not usually to maximize coverage, and the high precision of CSP makes it an ideal choice for this type of application.

The structure and the dynamic nature of a Web site can also have an impact on the choice between sequential and non-sequential models. For example, in a highly connected site reliance on fine-grained sequential information in user trails is less meaningful. On the other hand, a site with many dynamically generated pages, where often a contiguous navigational path represents a semantically meaningful sequence of user actions each depending on the previous actions, the sequential models are better suited in providing useful recommendations.

1.5.2 The Future: Personalization Based on Semantic Web Mining

Usage patterns discovered through Web usage mining are effective in capturing item-to-item and user-to-user relationships and similarities at the level of user sessions. However, without the benefit of deeper domain knowledge, such patterns provide little insight into the underlying reasons for which such items or users are grouped together. It is possible to capture some of the site semantics by integrating keyword-based content-filtering approaches with collaborative filtering and usage mining techniques. These approaches, however, are incapable of capturing more complex relationships at a deeper semantic level based on the attributes associated with structured objects.

Indeed, with the growing interest in the notion of semantic Web, an increasing number of sites use structured semantics and domain ontologies as part of the site design, creation, and content delivery. The primary challenge for the next-generation of personalization systems is to effectively integrate semantic knowledge from domain ontologies into the various parts of the process, including the data preparation, pattern discovery, and recommendation phases. Such a process must involve some or all of the following tasks and activities.

1. Ontology learning, extraction, and preprocessing: Given a page in the Web site, we must be able extract domain-level structured objects as semantic entities contained within this page. This task may involve the automatic extraction and classification of objects of different types into classes based on the underlying domain ontologies. The domain ontologies, themselves, may be pre-specified, or may be learned automatically from available training data [Craven et al., 2000]. Given this capability, the transaction data can be transformed into a representation which incorporates complex semantic entities accessed by users during a visit to the site.

2. Semantic data mining: In the pattern discovery phase, data mining algorithms must be able to deal with complex semantic objects. A substantial body of work in this area already exist. These include extensions of data mining algorithms (such as association rule mining and clustering) to take into account a concept hierarchy over the space of items. Techniques developed in the context of "relational" data mining

are also relevant in this context. Indeed, domain ontologies are often expressed as relational schema consisting of multiple relations. Relational data mining techniques have focused on precisely this type of data.

3. Domain-level aggregation and representation: Given a set of structured objects representing a discovered pattern, we must then be able to create an aggregated representation as a set of pseudo objects each characterizing objects of different types occurring commonly across the user sessions. Let us call such a set of aggregate pseudo objects a Domain-level Aggregate Profile. Thus, a domain-level aggregate profile characterizes the activity of a group of users based on the common properties of objects as expressed in the domain ontology. This process will require both general and domain-specific techniques for comparison and aggregation of complex objects, including ontology-based semantic similarity measures.

4. Ontology-based recommendations: Finally, the recommendation process must also incorporate semantic knowledge from the domain ontologies. This requires further processing of users activity record according to the ontological structure of the objects accessed, and the comparison of the transformed "semantic transactions" to the discovered domain-level aggregate profiles. To produce useful recommendations for users, the results of this process must be instantiated to a set of real objects or pages that exist in the site.

The notion of "Semantic Web Mining" was introduced in [Berendt et al., 2002a]. Furthermore, a general framework was proposed for the extraction of a concept hierarchy from the site content and the application of data mining techniques to find frequently occurring combinations of concepts. An approach to integrate domain ontologies into the personalization process based on Web usage mining was proposed in [Dai and Mobasher, 2002], including an algorithm to construct domain-level aggregate profiles from a collection of semantic objects extracted from user transactions.

Efforts in this direction are likely to be the most fruitful in the creation of much more effective Web usage mining and personalization systems that are consistent with emergence and proliferation of the semantic Web.

1.6 References

References

R. Agarwal, C. Aggarwal, and V. Prasad. A Tree Projection Algorithm for Generation of Frequent Itemsets. In Proceedings of the High Performance Data Mining Workshop, Puerto Rico, April 1999.

C. C. Aggarwal, J. L. Wolf, and P. S. Yu. A New Method for Similarity Indexing for Market Data. In Proceedings of the 1999 ACM SIGMOD Conference, Philadelphia, PA, June 1999.

R. Agrawal and R. Srikant. Fast Algorithms for Mining Association Rules. In Proceedings of the 20th International Conference on Very Large Data Bases (VLDB'94), Santiago, Chile, Sept 1994.

R. Agrawal and R. Srikant. Mining Sequential Patterns. In Proceedings of the International Conference on Data Engineering (ICDE'95), Taipei, Taiwan, March 1995.

A. Banerjee and J. Ghosh. Clickstream Clustering Using Weighted Longest Common Subsequences. In Proceedings of the Web Mining Workshop at the 1st SIAM Conference on Data Mining, Chicago, Illinois, April 2001.

B. Berendt, A. Hotho, and G. Stumme. Towards Semantic Web Mining. In Proceedings of the First International Semantic Web Conference (ISWC02), Sardinia, Italy, June 2002a.

B. Berendt, B. Mobasher, M. Nakagawa, and M. Spiliopoulou. The Impact of Site Structure and User Environment on Session Reconstruction in Web Usage Analysis. In Proceedings of the 4th WebKDD 2002 Workshop, at the ACM-SIGKDD Conference on Knowledge Discovery in Databases (KDD'2000), Edmonton, Alberta, Canada, July 2002b.

B. Berendt and M. Spiliopoulou. Analysing Navigation Behaviour in Web Sites Integrating Multiple Information Systems. VLDB Journal, Special Issue on Databases and the Web, 9(1):56–75, 2000.

J. Borges and M. Levene. Data Mining of User Navigation Patterns. In B. Masand and M. Spiliopoulou, editors, Web Usage Analysis and User Profiling: Proceedings of the WEBKDD'99 Workshop, LNAI 1836, pages 92–111. Springer-Verlag, 1999.

A. Buchner and M. D. Mulvenna. Discovering Internet Marketing Intelligence through Online Analytical Web Usage Mining. SIGMOD Record, 4(27):54–61, 1999.

M. Claypool, A. Gokhale, T. Miranda, P. Murnikov, D. Netes, and M. Sartin. Combining Content-based and Collaborative Filters in an Online Newspaper. In Proceedings of the ACM SIGIR '99 Workshop on Recommender Systems: Algorithms and Evaluation, Berkeley, California, August 1999.

R. Cooley. Web Usage Mining: Discovery and Application of Interesting Patterns from Web Data. Ph. d. dissertation, Department of Computer Science, University of Minnesota, Minneapolis, Minnesota, 2000.

R. Cooley, B. Mobasher, and J. Srivastava. Data Preparation for Mining World Wide Web Browsing Patterns. Journal of Knowledge and Information Systems, 1(1):5–32, 1999.

M. Craven, D. DiPasquo, D. Freitag, A. McCallum, T. Mitchell, K. Nigam, and S. Slattery. Learning to Construct Knowledge Bases from the World Wide Web. Artificial Intelligence, 118(1-2):69–113, 2000.

H. Dai and B. Mobasher. Using Ontologies to Discover Domain-Level Web Usage Profiles. In Proceedings of the 2nd Semantic Web Mining Workshop at ECML/PKDD 2002, Helsinki, Finland, August 2002.

M. Deshpande and G. Karypis. Selective Markov Models for Predicting Web-Page Accesses. In Proceedings of the First International SIAM Conference on Data Mining, Chicago, April 2001.

W. B. Frakes and R. Baeza-Yates. Information Retrieval: Data Structures and Algorithms. Prentice Hall, Englewood Cliffs, NJ, 1992.

X. Fu, J. Budzik, and K. J. Hammond. Mining Navigation History for Recommendation. In Proceedings of the 2000 International Conference on Intelligent User Interfaces, New Orleans, LA, January 2000. ACM Press.

E. Han, G. Karypis, V. Kumar, and B. Mobasher. Hypergraph Based Clustering in High-Dimensional Data Sets: A Summary of Results. IEEE Data Engineering Bulletin, 21(1): 15–22, March 1998.

J. Herlocker, J. Konstan, A. Borchers, and J. Riedl. An Algorithmic Framework for Performing Collaborative Filtering. In Proceedings of the 22nd ACM Conference on Research and Development in Information Retrieval (SIGIR'99), Berkeley, CA, August 1999.

T. Joachims, D. Freitag, and T. Mitchell. WebWatcher: A Tour Guide for the World Wide Web. In Proceedings of the International Joint Conference in AI (IJCAI97), Los Angeles, August 1997.

J. Konstan, B. Miller, D. Maltz, J. Herlocker, L. Gordon, and J. Riedl. Grouplens: Applying Collaborative Filtering to Usenet News. Communications of the ACM, 40(3):77–87, 1997.

H. Lieberman. Letizia: An Agent that Assists Web Browsing. In Proceedings of the 1995 International Joint Conference on Artificial Intelligence, IJCAI'95, Montreal, Canada, August 1995.

W. Lin, S. A. Alvarez, and C. Ruiz. Efficient Adaptive-Support Association Rule Mining for Recommender Systems. Data Mining and Knowledge Discovery, 6:83–105, 2002.

B. Liu, W. Hsu, and Y. Ma. Association Rules with Multiple Minimum Supports. In Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'99, poster), San Diego, CA, August 1999.

B. Mobasher, R. Cooley, and J. Srivastava. Automatic Personalization Based on Web Usage Mining. Communications of the ACM, 43(8):142–151, 2000a.

B. Mobasher, H. Dai, T. Luo, and M. Nakagawa. Effective Personalization Based on Association Rule Discovery from Web Usage Data. In Proceedings of the 3rd ACM Workshop on Web Information and Data Management (WIDM01), Atlanta, Georgia, November 2001a.

B. Mobasher, H. Dai, T. Luo, and M. Nakagawa. Improving the Effectiveness of Collaborative Filtering on Anonymous Web Usage Data. In Proceedings of the IJCAI 2001 Workshop on Intelligent Techniques for Web Personalization (ITWP01), Seattle, WA, August 2001b.

B. Mobasher, H. Dai, T. Luo, and M. Nakagawa. Using Sequential and Non-Sequential Patterns in Predictive Web Usage Mining Tasks. In Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM02), Maebashi City, Japan, December 2002a.

B. Mobasher, H. Dai, T. Luo, Y. Sun, and J. Zhu. Integrating Web Usage and Content Mining for More Effective Personalization. In E-Commerce and Web Technologies: Proceedings of the EC-WEB 2000 Conference, Lecture Notes in Computer Science (LNCS) 1875, pages 165–176. Springer, September 2000b.

B. Mobasher, H. Dai, and M. Nakagawa T. Luo. Discovery and Evaluation of Aggregate Usage Profiles for Web Personalization. Data Mining and Knowledge Discovery, 6:61–82, 2002b.

M. O'Conner and J. Herlocker. Clustering Items for Collaborative Filtering. In Proceedings of the ACM SIGIR Workshop on Recommender Systems, Berkeley, CA, August 1999.

T. Palpanas and A. Mendelzon. Web Prefetching Using Partial Match Prediction. In Proceedings of the 4th International Web Caching Workshop (WCW99), San Diego, CA, March 1999.

M. Pazzani. A Framework for Collaborative, Content-Based and Demographic Filtering. Artificial Intelligence Review, 13(5-6):393–408, 1999.

J. Pei, J. Han, B. Mortazavi-Asl, and H. Zhu. Mining Access Patterns Efficiently from Web Logs. In Proceedings of the 4th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'00), Kyoto, Japan, April 2000.

M. Perkowitz and O. Etzioni. Adaptive Web Sites: Automatically Synthesizing Web Pages. In Proceedings of the 15th National Conference on Artificial Intelligence, Madison, WI, July 1998.

J. Pitkow and P. Pirolli. Mining Longest Repeating Subsequences to Predict WWW Surfing. In Proceedings of the 2nd USENIX Symposium on Internet Technologies and Systems, Boulder, Colorado, October 1999.

R.R. Sarukkai. Link Prediction and Path Analysis Using Markov Chains. In Proceedings of the 9th International World Wide Web Conference, Amsterdam, May 2000.

B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Application of Dimensionality Reduction in Recommender Systems–A Case Study. In Proceedings of the WebKDD 2000 Workshop at the ACM-SIGKDD Conference on Knowledge Discovery in Databases (KDD'2000), August 2000a.

B. M. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Analysis of Recommender Algorithms for E-Commerce. In Proceedings of the 2nd ACM E-Commerce Conference (EC'00), Minneapolis, MN, October 2000b.

S. Schechter, M. Krishnan, and M. D. Smith. Using Path Profiles to Predict HTTP Requests. In Proceedings of the 7th International World Wide Web Conference, Brisbane, Australia, April 1998.

U. Shardanand and P. Maes. Social Information Filtering: Algorithms for Automating 'Word of Mouth'. In Proceedings of the Computer-Human Interaction Conference (CHI95), Denver, CO, May 1995.

M. Spiliopoulou and L. Faulstich. WUM: A Tool for Web Utilization Analysis. In Proceedings of EDBT Workshop at WebDB'98, LNCS 1590, pages 184–203. Springer Verlag, 1999.

M. Spiliopoulou, B. Mobasher, B. Berendt, and M. Nakagawa. A Framework for the Evaluation of Session Reconstruction Heuristics in Web Usage Analysis. INFORMS Journal of Computing - Special Issue on Mining Web-Based Data for E-Business Applications, 15 (2):171–190, 2003.

J. Srivastava, R. Cooley, M. Deshpande, and P. Tan. Web Usage Mining: Discovery and Applications of Usage Patterns from Web Data. SIGKDD Explorations, 1(2):12–23, 2000.

P. Tan and V. Kumar. Discovery of Web Robot Sessions Based on Their Navigational Patterns. Data Mining and Knowledge Discovery, 6:9–35, 2002.

L. H. Ungar and D. P. Foster. Clustering Methods For Collaborative Filtering. In Proceedings of the Workshop on Recommendation Systems at the 15th National Conference on Artificial Intelligence, Madison, Wisconsin, July 1998.

World Wide Web Committee, Web Usage Characterization Activity. http://www.w3.org/WCA.

P. S. Yu. Data Mining and Personalization Technologies. In Proceedings of the International Conference on Database Systems for Advanced Applications (DASFAA99), Hsinchu, Taiwan, April 1999.

O. Zaiane, M. Xin, and J. Han. Discovering Web Access Patterns and Trends by Applying OLAP and Data Mining Technology on Web Logs. In Proceedings of the IEEE Conference on Advances in Digital Libraries (ADL'98), Santa Barbara, CA, April 1998.