University of Alberta

Library Release Form

Name of Author: Jun Luo

Title of Thesis: Towards a Flexible Interactive Web Usage Mining System

Degree: Master of Science

Year this Degree Granted: 2001

Jun Luo
Department of Computing Science
211 Athabasca Hall
University of Alberta
Edmonton, Alberta
Canada T6G 2E1

Date: ________________

University of Alberta

Towards a Flexible Interactive Web Usage Mining System

by

Jun Luo

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of **Master of Science**.

Department of Computing Science

Edmonton, Alberta
Fall 2001

University of Alberta

Faculty of Graduate Studies and Research

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled **Towards a Flexible Interactive Web Usage Mining System** submitted by Jun Luo in partial fulfillment of the requirements for the degree of **Master of Science**.

Dr. Osmar R. Zaïane
Supervisor

Dr. Russ Greiner
Internal Examiner

Dr. T. Craig Montgomerie
External Examiner

Date:

To my parents,
my friends
and my wife

Abstract

With the rapid development of Internet technology, the World Wide Web is becoming one of the most important media for disseminating, collecting and sharing information. The enormous amount of web usage data, which has been collected and accumulated over years, has provided a very exciting arena for data mining.

Web usage mining generally refers to applying data mining techniques to web usage data and can provide in-depth analyses of users' navigation behaviours. There have been many sophisticated web usage mining systems which are dedicated to analyzing web usage data in the e-commerce field. However, their utilities are greatly restricted by the rigid pre-defined web usage mining process. In this thesis, we present a prototype of an open-structure interactive web usage mining system. With this system, not only useful models can be easily integrated by developers to expand the functionalities, but also various data mining experiments can be designed by decision makers interactively to discover patterns of interest.

Acknowledgements

First of all, my thanks go to my supervisor Dr. Osmar R. Zaïane for his guidance and funding.

I would like to thank the members of my examining committee, Dr. Russ Greiner and Dr. T. Craig Montgomerie, who reviewed this work and offered constructive criticisms.

Special thanks to Weinan Wang and Behzad Mortazavi-Asl for sharing their work.

Finally, I would like to express my sincere gratitude to my parents for their support all these years.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

With the rapid development of Internet technology, the World Wide Web (WWW) is becoming one of the most important media for disseminating, collecting and sharing information. The enormous amount of web usage data, which has been collected and accumulated over years by web servers for each single access, has provided a very exciting arena for data mining.

Web usage mining generally refers to applying data mining techniques to web usage data. In this thesis, we present a prototype of an open-structure interactive web usage mining system. With this system, not only useful models can be easily integrated by developers to expand the functionalities, but also various data mining experiments can be designed by decision makers interactively to discover patterns of interest in many application domains, such as e-commerce, on-line learning, or any web-based application.

1.1 Motivation

The fact that communications between organizations and their clients are becoming easier, faster, and more economical through the use of Internet, has motivated more and more organizations — including schools, companies and governments — to design and construct WWW-based virtual environments to provide services. The Internet has also dramatically changed the interaction patterns between organizations and their customers, and the success of these organizations relies on understanding and providing good supports for these patterns.

This gives rise to emerging intensive demands to understand web site users, determine the effectiveness of web site designs, and enhance on-line information distribution. An e-commerce web site, for instance, can make greater profits from developing better marketing strategies and personalized services to convert random visitors into regular consumers. This can be done by identifying certain characteristics of users and their shopping patterns. Similarly, with an e-learning web site, it is beneficial for instructors to understand students' on-line learning behaviour. Knowledge about the students' navigation patterns can provide ideas to improve the quality of instruction or help assess on-line learning models.

In addition, it is quite common for users to complain that they are unable to easily locate desired information. The reason for this is that there is always some discrepancy between the opinions of web designers and users about navigation or accessibility of information on web sites: the method of organizing on-line documents reflects the designers' points of view, which would not necessarily be shared by the users. Web site administrators, therefore, are in great need of a non-intrusive and automatic mechanism through which they can objectively evaluate the effectiveness of web site designs by investigating the frequently visited, as well as less visited, zones of the sites, and comparing the actual and expected navigation patterns.

Web usage mining can provide such in-depth analyses of users' navigation behaviours. There have been many sophisticated web usage mining systems which are dedicated to analyzing web usage data in the e-commerce field. However, their utilities are greatly restricted by **the rigid pre-defined web usage mining process.** In many cases, the decision makers who use these systems have various decision problems. To better solve these problems, they need the flexibilities to design the preferred mining process.

1.2 Scope and Organization of the Thesis

This thesis addresses **the design and implementation of a flexible interactive web usage mining system**, which is distinguished from other existing web usage mining systems in that it allows constraint specification in all phases of the web usage mining process: data preprocessing, pattern discovery, and pattern analysis. With this system, the decision makers are able to flexibly and interactively design the desired mining process according to a specified decision problem.

This thesis demonstrates the advantages of our system mainly from two aspects:

- Data preprocessing is a very critical phase in web usage mining, which transforms the raw web log data into reliable and formatted data. It often consumes

80% to 95% of the effort and resources needed [Ede2001]. Users visiting different web sites often have very diversified navigation behaviours. It is, therefore, not helpful to apply certain simple standards in the data preprocessing phase, even though such approaches have been proven to be practical in analyzing some particular web sites. This situation poses a challenge to our research.

  We present several new techniques in order to obtain reliable and desirable data for in-depth analysis. Specifically, we propose a novel way, using the fuzzy set concept, to model users' on-line behaviours. With the fuzzy membership functions, we are able to integrate multiple criteria to transform raw web log data. **In the data preprocessing phase, our system offers the flexibilities to select and combine some of these approaches in order to properly model users' navigation behaviours.**

- In the pattern discovery phase, three generic data mining algorithms — association rule mining, intra-transaction sequential pattern analysis, and inter-transaction sequential pattern analysis — have been integrated in our system to extract patterns from web usage data. However, most of these algorithms are inclined to extract the statistical dominant patterns which are not necessarily related to the decision problems.

  In this thesis, we present a constraint-based sequential pattern analysis algorithm. With this data mining algorithm, decision makers can specify the preferred patterns or constraints on the patterns to discover by interacting with our system. **Our system also makes it possible to push these constraints into different phases of the web usage mining process.**

Our research was initially motivated by an interest in applying data mining techniques to analysis of students' on-line learning behaviours. Most of the examples used in this thesis are related to analyzing learners' activities in the context of web-based learning environments. The discovery of patterns from navigation history by web usage mining sheds light on learners' navigation behaviour and also on the efficiency of the models used in the on-line learning process. The discovered patterns may be used to evaluate learners' activities, as well as for adapting and customizing resource delivery, providing automatic recommendations for activities, etc. [ZLu2001]

However, it should be noted that at the current stage, our research doesn't address the issues on pattern analysis, which aims at analyzing the discovered patterns and providing advanced decision support. So we mainly evaluate our work using these two criteria:

- Whether our system can effectively extract patterns from web log files (How many patterns can be discovered)?
- Whether the interactive web usage mining process can help find the desired knowledge according to the specified requirements effectively and efficiently?

This thesis is organized as follows: the following chapter introduces the different research directions of web mining and, in particular, it gives a survey of a number of web usage mining projects. Chapter 3 presents our open-structure interactive web usage mining system. Chapter 4 discusses issues related to the data gathering and preprocessing, and proposes a novel way to extract the fuzzy boundary transactions. Chapter 5 focuses on a constraint-based sequential pattern analysis algorithm and extends its ability to handle the fuzzy boundary transactions. Chapter 6 presents the case study of our system and Chapter 7 concludes our work and proposes future directions.

Chapter 2

Related Works

Data mining has traditionally been applied to databases and/or data warehouses. Web mining, which takes advantage of the powerful data mining techniques to extract useful knowledge from the WWW, has been an active research area. Different from simple resource discovery on the Internet, web mining aims at finding knowledge that is not explicitly expressed by resources. There are three major domains in web mining research, namely, web content mining, web structure mining and web usage mining.

This chapter begins with a brief summary of the taxonomy of web mining, followed by a survey of current commercial web analysis tools and sophisticated web usage mining applications.

2.1 Taxonomy of Web Mining

Traditional data mining, also often called knowledge discovery from data, is a non-trivial process through which decision makers can extract the implicit, novel, and potentially useful patterns from large datasets [HKa2000]. Recent research has made efforts to apply data mining techniques on the web and achieved significant progress in discovering useful knowledge. According to discussions in [Zai1998], these efforts can be classified into the following three domains:

2.1.1 Web Content Mining

With the explosive growth of documents, such as articles, pictures, and videos, available on the WWW, ironically people often feel lost when faced with these superabundant resources. Although current powerful search engines have provided support in aiding users to locate desired web documents easily, seldom do they explore the extent of the information hidden on these web documents.

In fact, in most cases these documents are too dynamic and anarchical to be tamed in a well structured manner. At first, most of them are represented in natural language, from which there are currently few effective methods to extract the full meaning. Secondly the HTML tags reveal only the display structure of a web page in the browsers; but cannot give insight into the content.

Despite these difficulties, it is still possible to capture the semantic features from HTML files. This encourages researchers to delve into the problems related to extracting knowledge from web content. Previous efforts have been made into exploring primarily three sub-issues: **resource discovery**, which aids users in locating desired and unfamiliar on-line documents; **information extraction**, which summarizes, filters and/or interprets the web documents; and **generalization**, which categorizes those documents into several groups according to the similarities/distances between their contents [Etz1996, Zai1998].

2.1.2 Web Structure Mining

The hyperlinks between web documents reveal some interesting information beyond the actual content. In bibliographic citations, it is generally believed that a paper being cited by many other papers is a popular paper, while a paper citing a large number of popular papers can be a valuable survey. A web page, analogously, if highly referenced by other web pages, may contain very popular topics, and a page having many links to other popular web pages may be a valuable site that users can utilize to locate related web pages.

J. Kleinberg in [Kle1997] named the former type of web page, **authorities** and the latter type, **hubs**, and also uncovered the strong mutually reinforcing relationship **between** authorities and hubs: a sound authority is probably pointed to by many good hubs, and a good hub should point to a number of sound authorities. This observation has been widely used in the field of web information retrieval.

2.1.3 Web Usage Mining

J. Srivastava et al., in an up-to-date survey on web usage mining [SCD+2000], listed three possible resources of web usage data: web server, client terminal, and proxy server. These three different levels of web usage data correspond to **multi-user and single-site**, **single-user and multi-site**, and **multi-user and multi-site** navigation patterns.

Research into web usage mining can be generally divided into two approaches. The first tries to find the general access patterns and trends of the overall web site, without regard to the individual user. The other approach attempts to discover the important patterns of users by identifying the individual user and user sessions [Zai1998].

This thesis focuses on the analysis of multi-user and single-site usage patterns regarding the individual behaviour. In the rest of this chapter, we survey the current commercial web log analysis tools and the variety of web usage mining research projects.

2.2 Web Log Analysis Tools

The more visitors a web site attracts, the greater the potential profits that can be made from it. This is a key motivator in encouraging the owners of web sites to spend a great deal of money placing advertisement anchors on other web sites. However, it is essential that they are able to determine whether their investments are worthwhile or not. Most of the existing web log analysis tools provide mechanisms examining web traffic, and reporting on web site effectiveness and usage trends in order to provide this information.

In addition, by using these log analysis tools, we can determine the most popular web pages during a specified period, detect unauthorized traffic and erroneous visiting, estimate the downloading burden of the web server, assess web server performance, and so on. SurfAid supports OLAP (On-Line Analytical Processing) operations over the web usage data, and provides means for the decision makers to derive the desired format and content of reports [Sur2000]. NetTracker summarizes the usage information related to generating repeat visits, purchases, and web traffic growth [Net2000]. WebTrends models the customers' on-line behaviour into several categories, and generates various reports for different decision makers [Web1999].

All these functionalities, together with other customized conceptual manipulations and filtering modules, have contributed a great deal to supporting better marketing decisions, as well as improvement of web services. However, the power of these tools is still very limited in providing a thorough analysis of access patterns. Most of the reports indicate only statistical significance, rather than conceptual importance. A typical report entry, such as "During the time period T, there were N

hits on the particular web page(s) P" cannot offer insight into hidden usage patterns and trends. People therefore rely on web usage mining systems to perform more sophisticated and complex analytic tasks.

2.3 Web Usage Mining Projects

There are many web usage mining projects. According to their different research focuses and adopted approaches, we classify them into several main research directions: general web usage mining framework, sequential pattern analysis, clustering and on-line recommendation, characterization and visualization, and other efforts.

2.3.1 General Framework

WEBMINER, proposed by B. Mobasher et al. [MJH+1996, CMS1997] in 1996, presents a framework for web usage mining. It is a modular architecture consisting of six models for data cleaning, transaction identification, data integration, transformation, pattern discovery, and pattern analysis. Their extensive research on data preprocessing [CMS1999] is cited by many other later works. The generic data mining models are able to discover association rules, sequential patterns, and classification rules from web usage data.

WEBSIFT, which is the abbreviation of WEB Site Information Filter System, evolves from the WEBMINER prototype, and considers more details in pattern evaluation. R. Cooley et al. [CTS1999], in the WEBSIFT project, assume the web topology and content data is valuable domain knowledge to construct the belief set: the web pages are "related" if they are linked together and/or they share some similarities in content. The discovered patterns which are inconsistent or unknown to the belief set, are considered to be more interesting patterns because they are potentially unexpected to decision makers.

WebLogMiner uses data mining and data warehousing techniques to analyze web log records. A general description of the four steps involved in processing the web log data is presented in [ZXH1998]. First, the irrelevant and noisy data are cleaned and filtered out, and the remaining data are stored in the relational database, where the multi-dimensional data model of web usage information is also constructed. As a consequence, OLAP operations can be performed in web log data cubes — including drill-down, roll-up, slice and dice. Finally data mining algorithms are carried out on these data cubes to discover interesting patterns. WebLogMiner demonstrates its potential on time-series analysis, including network traffic analysis, event sequence and user behaviour pattern analysis, transition analysis, and trend analysis.

2.3.2 Sequential Pattern Analysis

WUM (Web Utilization Miner) is another interesting web usage mining tool [SFW1999, SPF1999, BSp2000]. It also has three modules performing preparation of the web usage data, discovery of interesting navigation patterns, and visualization of the discovered patterns. The main problem this approach attempts to address is: given a number of traversed paths, discover sub-paths with structural or statistical properties of interest. In WUM, M. Spiliopoulou et al. proposed a data structure, the aggregated tree, to merge the transaction-like usage information. By traversing the branches in the aggregated tree with a certain support threshold, the important navigation patterns can be found. In addition, a mining query language, MINT, is designed to express "pattern descriptor" which specifies the features of the interesting patterns so as to confine the mining results.

HPG (Hypertext Probabilistic Grammars) is a probabilistic regular grammar which models user navigation sessions [BLe1998, BLe1999]. Using HPG, a directed graph is generated to summarize the transactions, where each vertex represents one web page, and the arc between two vertices is associated with the weight to estimate the possibility of transition. Some deduction algorithms can then be carried out on this directed graph to find the composite sequences. However, this approach has to work under the assumption that a sequence will not form a cycle. This limitation eventually leads to the loss of some important patterns.

F. Masseglia et al. [MPC1999] apply the GSP (Generalized Sequential Pattern) algorithm to discover sequential patterns from web log data. This GSP algorithm was originally proposed by R. Srikant and R. Agrawal [ASr1995, ASr1996] to improve their own previous works on sequential pattern analysis. GSP has three important features: users can specify the upperbound of time interval between two consecutive events in a sequential pattern; it is able to discover the inter-transaction sequential patterns; and it introduces the taxonomy, i.e. concept hierarchies into the mining process to perform multi-level conceptual sequential pattern analysis. Another work of theirs is related to the WebTool System which uses an efficient approach, called ISEWUM (Incremental Sequence Extraction for Web Usage Mining), to incrementally update the sequential patterns with the new transactions or new clients

[MPT2000].

J. Pei et al. [PHM+2000] investigate the issues related to efficiently extracting sequential patterns from large repositories of web log data. They employ a concise, highly-compressed Web Access Pattern tree, WAP-tree in short, to generalize the web usage data and then perform a special algorithm, WAP-mine, to discover sequential patterns from the WAP-tree. They demonstrate that their algorithm outperforms the GSP algorithm in speed.

Most of these researchers have achieved significant results in analyzing various web usage data, yet few of them have investigated the interactions through the process of web usage mining. In Chapter 6, we will present our research on this issue and investigate the varying impacts of pushing these interactions into the different phases of web usage mining.

2.3.3 Characterization and Visualization

The goal of visualizing web usage information is to intuitively depict the ways the web sites are being used and capture some important features.

J. Pitkow et al. [CPi1995, Pit1997, Pit1998] investigate the problem of characterization of web usage data. Many widely-used heuristics to process the web log data are developed based on their research results. For example, the typical time-out threshold to break the click streams into transactions is originally proposed in their research in [CPi1995].

N. Minar and J. Donath [MDo1999] attempt to visualize the crowds at a web site. They use icons to represent the users, and the animation of these icons to simulate web traffic. From the visualization results, it is quite easy to tell the popular parts of the web site, and the important paths that users often take.

E. H. Chi et al. [CPM+1998] use the Web Ecology and Evolution Visualization(WEEV) techniques to intuitively represent the relationships among web content, topology, and usage through time. Distinct from N. Minar and J. Donath's work, they introduce a new visualization technique called Time Tube, instead of animation, to capture the changing nature of the web site.

2.3.4 Clustering and Recommendation

Basically, the efforts to cluster the web usage data can be classified into three main domains: clustering the users, clustering the web pages, and clustering the user sessions.

O. Nasraoui et al. in their paper [NFJ+1999] propose a typical clustering approach to build the user profiles. Their work is based on their definitions on the similarities between the user sessions. With the estimations of the similarity between user sessions, the CARD (Competitive Agglomeration for Relational Data) algorithm is used to group these sessions. In addition, they modified the algorithm for robust fuzzy clustering.

Another similar approach is proposed by Y.Fu et al. [FSS1999]. However before utilizing the clustering algorithm, BIRCH, they applied attribute-oriented induction on the web pages to summarize user sessions. By doing this, the similarity between sessions can be drawn from both the "semantic" similarities of the web pages.

A very recent work from A. Foss, W. Wang and O. R. Zaïane presents a novel clustering algorithm, TURN [FWZ2001]. This algorithm clusters the user sessions without any input parameters from the decision makers.

Differing from the above projects, which investigate the issues of clustering the user sessions, M. Perkowitz and O. Etzioni's approach attempts to group web pages [PEt1998, PEt1999]. Their goal is to create an adaptive web site which automatically improves its organization and presentation by learning from visitor access patterns. The idea is to construct some index pages each of which consists of links to a set of web pages sharing the same topic. Here they assume that if the web pages are frequently visited together then these pages represent coherent topics in most users' minds. The PageGather and SCML algorithms are used to group the web pages into clusters and derive the topics from each cluster.

Collaborative Filtering, which is also a recommender algorithm, provides a way to recommend items to a particular user by searching for items of interest to a group of users who have similar interests. However, some shortcomings of collaborative filtering — including requiring user explicit ratings, lack of scalability, and poor performance when handling high-dimensional and sparse data — restrict its usability in the field of web usage mining [UFo1998].

B. Mobasher et al. use clustering techniques to improve the scalability and performance of collaborative filtering techniques. It is found to be difficult to apply the distance-based clustering techniques on web usage mining. This is partly because it is hard to define the distance between user sessions, which are often high dimensional data. It is, therefore, necessary to find other clustering techniques for partitioning, rather than the distance-based clustering. ARHP (Association Rule Hypergraph Partitioning), which is a partitioning technique based on the frequent itemsets generated by association rule mining [MCS1999a, MCS1999b], and PACT (Profile Aggregations based on Clustering Transactions), which groups and summarizes the user transactions, are proposed to build up aggregate usage profiles. With the aggregate usage profiles, some strategies are designed to perform on-line recommendation [MDL+2000].

2.3.5 Others

P. K. Chan uses an innovative way to consider the user profiles as a combination of two components: WAG (Web Access Graph) and PIE (Page Interest Estimator) [Cha1999]. WAG is extracted from the web access log, while PIE is determined by integrating several factors which reflect the user interest, such as the frequency of visited web pages, bookmark, and so on. Several classification algorithms, including C4.5, CART, BAYES, and RIPPER, have been performed on user profiles to predict if a web page will be of interest to a user. Similar efforts include the work of D. Murray et al., which uses neural network to predict the demographic information of users based on their navigation behaviours [MDu1999].

2.4 Summary

Web usage mining has become one of the hottest fields for data mining research. Various techniques have been invented or borrowed to extract meaningful and useful patterns from web access data. The critical issues about these applications include how to extract well-formatted and reliable data from the raw web log, how to extract meaningful users' navigation patterns, and how to interpret and represent the discovered patterns.

Chapter 3

System Framework

A web usage mining system basically is a data mining system for the particular domain of web usage data. Similar to other data mining systems, most of the web usage mining applications can be divided into three generic modules which perform three consecutive tasks on the web usage data. A data gathering and preprocessing module collects the web usage data and meta-data as well as cleans, filters and transforms the log entries. A module of pattern discovery applies a variety of data mining algorithms, such as association rule mining, sequential pattern analysis, clustering, and classification on the formatted usage data, in order to discover significant and potentially useful patterns. Finally, the pattern analysis module provides functions to help decision makers to retrieve and interpret the discovered patterns.

This chapter begins with a detailed description of a three-tier system structure for web usage mining applications. We then propose an open-structure interactive web usage mining system. With this system, decision makers can flexibly design the web usage mining process according to a specified decision problem.

3.1 Three-tier System Structure

There are several essential steps in the process of Knowledge Discovery in Databases, KDD in short. These are: data cleaning, data integration, data selection, data transformation, data mining, pattern evaluation, and knowledge presentation. A complete knowledge discovery process consists of all these steps performed in sequence [HKa2000].

These steps also can be generalized into three phases. During the data cleaning, integration, selection, and transformation, we process the raw log entries into well-structured and reliable data. These four steps are primary components in the data gathering and preprocessing phase. The pattern discovery phase can also be referred to as the data mining phase. The two final steps — pattern evaluation and knowledge presentation — are main tasks in the pattern analysis phase. This three-tier structure is widely adopted in other web usage mining research [CMS1997, BSp2000]. Figure 3.1 shows the data flow, the modules of three phases, and their functions in the web usage mining process. In the rest of this section, we describe the functions of each of the three phases.

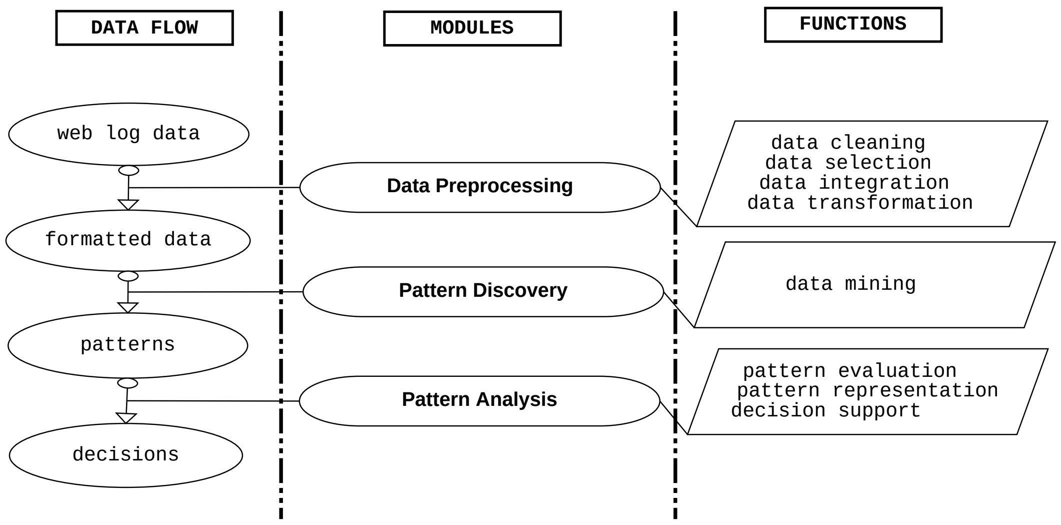

Figure 3.1: Three-tier Architecture of Web Usage Mining

3.1.1 Data Gathering and Data Preprocessing

Web usage mining systems distinguish themselves from other data mining applications primarily in that special and laborious efforts have to be made in the phase of data preprocessing. Data gathering and preprocessing is arguably the most important and difficult phase in web usage mining for the following reasons: First of all, the web log entries are semi-structured and chaotic in the sense that these entries are simply ordered chronologically and the data about different users are mixed together. In addition, there is erroneous, irrelevant, and incomplete information existing in the raw web log data. Since the quality of the formatted data obviously affects the quality of the patterns discovered in the subsequent phases, carefully adopting strategies to generate reliable data according to the different decision problems is very important.

In general, there are three subtasks in the data gathering and preprocessing phase:

- First, the web usage data and the relevant meta-data are collected;
- The second step consists of cleaning and filtering out misleading, incomplete and irrelevant data;
- Finally, different strategies are used to transform these cleaned data from a long sequence of clicks into a set of more meaningful units to depict users' navigation behaviours.

3.1.2 Pattern Discovery

In the phase of pattern discovery, a variety of generic data mining algorithms can be used to extract interesting patterns from web usage information. In Chapter 2, we showed that the following algorithms are typically used to find meaningful patterns from web usage data: characterization, association rule mining, sequential pattern analysis, clustering, and classification. Here, we summarize these algorithms in the context of web usage mining.

Characterization Characterization gives a concise generalization over the given data. Most current web log analysis tools provide the statistical reports to summarize some features of the web usage information. One widely used statistical report provides information such as: during time T, N users visit the web page P. Some other interesting statistical reports contain information such as: the average time consumption of one visit, clicks during a single visit, most frequent entry web pages. Not only does such an analysis provide a foundation for further in-depth research, it also sheds light on ways to improve the system performance and support marketing decision making.

Association Rule Mining Association rule mining is a classical data mining algorithm. It discovers the correlations among items within the transactions. In the context of web usage mining, a typical association rule looks like this: 55.6% of the learners who did the Exercise 1 also visited Chapter 1 during one visit. Such information is very valuable to web site administrators in evaluating their design of web site as it enables them to examine the disparity between their expectations and actual usage patterns.

Sequential Pattern Analysis Sequential pattern analysis aims at discovering the frequent sequences that happen iteratively. Based on the discovered sequential patterns, it is possible to develop advanced strategies to predict the subsequent activities of a user, and evaluate the web site design. Moreover, decision makers are able to understand users' navigation behaviours by inspecting discovered sequential patterns.

Clustering Clustering is a data mining algorithm which groups the data into a number of clusters, such that the data in the same cluster are similar according to certain pre-defined distance functions, while data in different clusters are dissimilar to each other. There are three clustering domains in the context of web usage mining. One finds the clusters of users such that the users in the same cluster have similar navigation patterns and/or interests. Another groups web pages based on their usage patterns: it is supposed that web pages belonging to the same cluster share some related topics. Finally, clustering the sessions provides a mechanism for summarizing the navigation patterns.

Classification Classification is another grouping algorithm that categorizes the data into a number of pre-defined classes. It is different from clustering algorithms in that the clustering algorithms separate the data into groups based on the distance measures, without pre-defining class labels. A classification algorithm can be used to predict the unknown attributes for navigation behaviours. For example, a typical classification rule looks like this: if a learner succeeds in Exercise 1 and participates in all the on-line discussions, then he/she would finally succeed in Exercise 2 as well. This rule has an accuracy of 83.5%.

3.1.3 Pattern Analysis

There are two sub-tasks in the pattern analysis phase: retrieving the interesting rules from the discovered rule set and representing these rules in a way that is easy to understand and interpret.

In certain situations, data mining algorithms can discover a considerable number of rules, hence extracting knowledge from these rules can be another laborious mining task. It is therefore necessary to design mechanisms to enable decision makers to get their most desired knowledge with the least effort. On the other hand, a well-designed user interface exploring the rules definitely helps decision makers to clarify the decision problems, as well as determine and explain the solutions.

Some recent trends of pattern analysis show that developing the learning algorithms or decision strategies based on the validated patterns achieves better results [LAR2000, UFo1998, MCS1999b]. Since the patterns summarize the important features of the data and are much smaller in size, instead of using

$$Decision = function(Data),$$

the researchers seek solutions based on

$$Decision = function(Patterns).$$

3.2 An Open-structure Interactive Web Usage Mining System

With an open-structure system, developers are able to integrate new functions and algorithms with relative ease. In addition, the enabled interactions through the web usage mining process give the decision makers the power to express different requirements according to different decision problems. In the rest of this section, we discuss the motivations and means for devising an open-structure web usage mining system.

3.2.1 Open-structure Architecture

An open-structure system makes extending and upgrading the system functions easier. Specifically, in the context of web usage mining for web-based learning environments, designing an open-structure system has the following advantages.

- Due to the diversity of the web server specifications, web log files of different web sites usually have different formats. The web servers are also different in their abilities to provide the meta-data, which are useful to interpret the usage data and further data mining results. With regard to web-based learning environments, different web sites often use different Internet techniques, such as CGI, ASP, Javascript, etc., to enable on-line interactions between instructors and learners. As a consequence, the approaches to process these log data are quite different. However, to process web usage data from a new web site in an open-structure web usage mining system, what is needed is to merely develop a new parser for the new web log data without dramatic modifications in the other modules.

- Thanks to the exciting growth of research into data mining applications, more and more data mining algorithms, as well as pattern analysis and visualization tools have been invented and developed. An open-structure web usage mining systems helps reduce the cost of integrating these new modules.

As shown in the previous sections, the data flow in web usage mining is clear in the three-tier system architecture from the high level perspective. In the data preprocessing phase, the raw web log data are transformed into formatted data. Thereafter, a variety of data mining algorithms can be used on these formatted data to extract significant patterns. Finally, decision makers are able to analyze the discovered patterns.

In order to make our system flexible enough to allow plug-and-play of new data preprocessing modules, data mining algorithms, and pattern analysis tools with relative ease, we emphasize the modular and generic design of important functions. In addition, we adopt the following two methods:

- With generic modules, the first method is to rigidly confine the format of the intermediate data, such as the formatted dataset and discovered patterns. It is also appropriate to explore the possibility of the XML representations of this intermediate information.

- The modules' I/O functionalities are extended to handle different input/output data formats. This is one of the simple methods for parsing web log data with different formats.

3.2.2 Interaction Design

Most current web usage mining systems limit user interactions in the pattern analysis phase, i.e., the users can only express a few simple requests when the whole mining process has been done. Because of this, for the decision makers, the whole process is like a black box, to which they provide the data and from which they get the results "by magic".

However, for web-based learning environments, enabling interactions through the web usage mining process has many advantages. First of all, instructors usually have different focuses in their decision problems. For instance, an instructor would only be interested in learners' activities which are related to his/her course. In addition, there are different kinds of domain knowledge that are valuable in web usage mining. Some of them can be explicitly expressed in typical data format. For example, the web topology can be expressed in a tree-like data structure. However it is almost certainly impossible to explicitly express all the domain knowledge in the desired data structure. For example, an instructor would determine the progress of the course content delivery and the relationships among the different parts of the course content. Since this kind of knowledge is very specific and volatile, it is obviously not reasonable to encode it in some pre-defined data format. With well-designed interactions, it is possible for the decision makers to implicitly integrate some of this domain knowledge in the mining process.

In addition, there are many heuristics that are based on various assumptions in the web usage mining process. It should be noted that not all these assumptions are valid in any specified web site. For example, the time-out method to identify transactions is based on the assumption that if a user does not send any request to a web server for a long period, he/she probably has left the web site. This assumption is applicable to many web sites. However in an e-learning web site, learners often spend a great deal of time in downloading large documents or software, or merely reading the course content. In another example, to measure the similarity of two web pages, some clustering algorithms [NFJ+1999, JJo1999, FSS1999] take the hierarchical organization of the web site into consideration. However, many e-learning web sites dynamically generate the web pages. They are unable to provide such information. It is necessary, therefore, to offer flexibility to the decision makers in selecting appropriate modules of the web usage mining process according to different web sites and decision problems. [1]

We can also examine the advantages of interactive web usage mining from the perspective of the data mining process. There are some important patterns that are attribute-related. If we try to investigate these patterns in the whole dataset, they

[1]It should be noted that there are side effects of providing many parameters to instructors, since most of them are not knowledgeable about data mining. The flexibilities in this case often turn into obstacles that keep instructors from using our system with comfort. To address this issue, there is another research effort in the overall project to design new algorithms that need minimum input from the user[FWZ2001]. Also, in Chapter 4, we present some approaches that automatically adjust the input from instructors to the targeted web usage data. Other approaches include research on the human computer interface and psychology and are beyond the scope of this thesis.

become statistically trivial and difficult to distinguish. However if we concentrate on examining these patterns with a subset with certain restrictions on some of the attributes; or we use additional descriptors of these patterns to search the dataset, these patterns can become prominent. For example (Figure 3.2), we assume that there are some patterns in the whole search space: Pattern#1, Pattern#2, Pattern#3, and Pattern#4. Pattern#1 is statistically dominant in the whole search space. Most data mining algorithms are more inclined to extract Pattern#1 but omit Pattern#2, Pattern#3, and Pattern#4. However, these normally omitted patterns can be valuable for decision making.

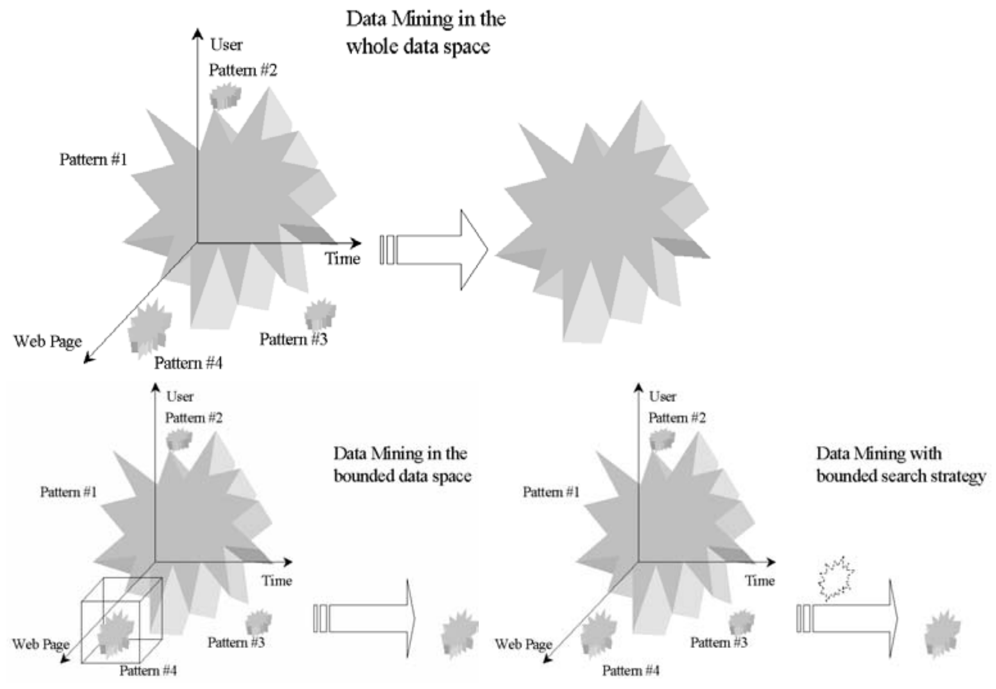
Data Mining in the whole data space
User
Pattern #2
Pattern #1
Time
Web Page
Pattern #3
Pattern #4
Data Mining in the bounded data space
Data Mining with bounded search strategy

Figure 3.2: Bounded Search Space and Bounded Search Strategy

This problem illustrated in this example is very common in the context of web usage mining for web-based learning environments since learners often show interest in some particular set of web pages at just one point of time. For example, they would probably visit repeatedly the course content of Chapter 3 when there is an upcoming exam on Chapter 3, while after that, seldom would they revisit it frequently. Interactive data mining makes it possible to uncover these statistically trivial but valuable patterns. Furthermore, by using this mechanism, we are able to significantly reduce the search cost for important patterns for two key reasons: at first, by filtering out the irrelevant or less-relevant data, we reduce the size of the search space considerably; secondly, we can push some constraints into the data mining algorithms to improve the efficiency of the searching for desired patterns.

Based on the above discussions, we are able to draw a conclusion that it is critical to the design of different interactions, through the whole web usage mining process, for the decision makers, as illustrated in Figure 3.3.

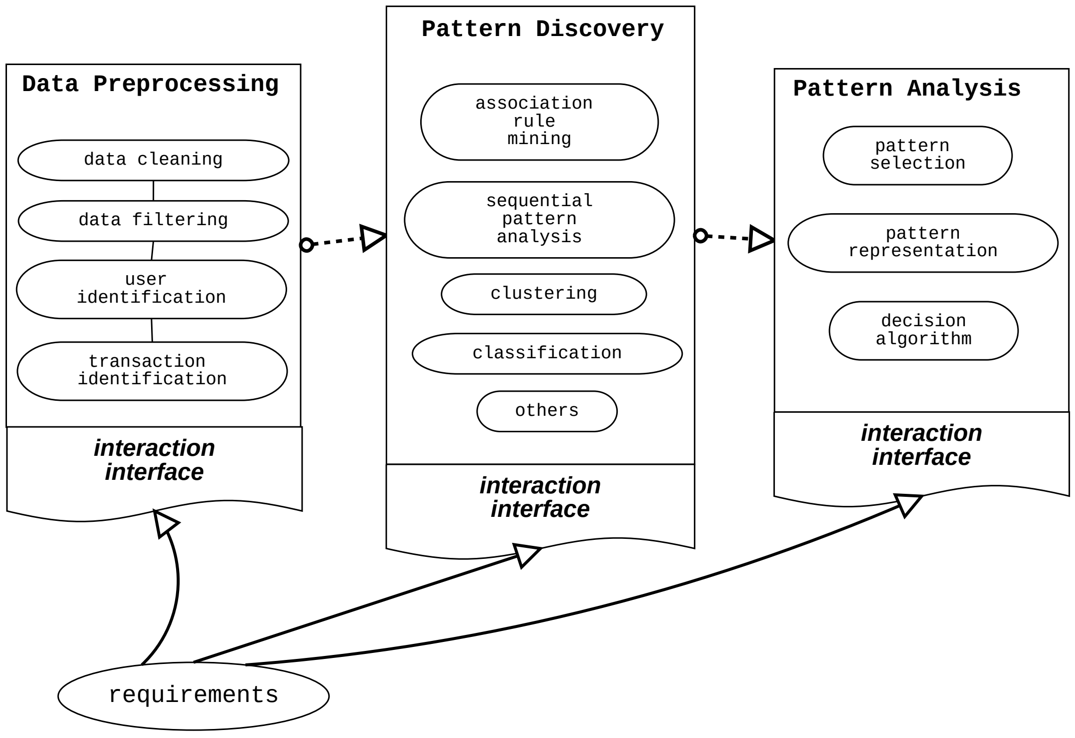

Figure 3.3: Interactive Web Usage Mining

We introduce two kinds of interactions in our web usage mining system. The instructors can express their requirements by using either the **specifications** or the **constraints**:

specifications There are various choices existing in each phase of web usage mining. As we know, there are many strategies that can be used for data preprocessing, and there are also various data mining algorithms that can be chosen for different decision problems. All these different options can be combined to produce a variety of different web usage mining experiments. For example, we can choose the time-out method to generate the transaction dataset, and then employ association rule mining to discover association rules. Or we can generate the transaction dataset by other criteria, and then use a sequential pattern analysis algorithm to obtain frequent sequences.

In our system, instructors are able to design their web usage mining experiments by indicating the specifications, whenever appropriate, according to different data environments and decision problems.

constraints The interactive web usage mining system enables instructors to impose different constraints throughout the process of knowledge discovery. These constraints can be further classified into the following two categories.

data-related constraints: These constraints are related to the different attributes and their domain in the dataset. In the context of web usage mining, the commonly used attributes include users, time period, and web pages. These constraints can be expressed by an SQL-like grammar. In our system, this kind of constraint is often specified in the phase of data preprocessing to filter out irrelevant data, in order to reduce the search space.

pattern-related constraints: These constraints are closely related to data mining algorithms. For an association rule, for example, there are two typical constraints: the support threshold and the confidence threshold. In addition, there can be other constraints to specify the special actions in the rule antecedent or consequent. These constraints are used mostly in the phases of patterns discovery and pattern analysis. Different data mining algorithms have different types of parameters to describe the desired patterns. Integrating the constraints in the pattern discovery phase can improve the efficiency in one single search process. However, if there is a requirement to repeatedly search some patterns with different constraints, it is preferable to impose the constraints in the phase of pattern analysis.

3.3 Summary

The web usage mining process can be divided into three generic phases: data preprocessing, pattern discovery, and pattern analysis. With an open-structure web usage mining system, developers can integrate new functions and algorithms with relative ease. To better analyze and assess the learners' navigation behaviours in web-based learning environments, we design an interactive web usage mining system, with which instructors are able to flexibly express different requirements through the web usage mining process.

Chapter 4

Data Gathering and Preprocessing

Web usage mining applies data mining techniques to web usage data. Normally, web usage mining applications extract usage information from the web log files and meta data. These meta data are often used to interpret the web log data in a meaningful manner, as well as to analyze and evaluate discovered patterns. The task of a data preprocessing module is to obtain reliable and desirable datasets from raw web log files, which, in most cases, contain a considerable amount of noisy and irrelevant information. In general, data preprocessing in web usage mining system consists of the following steps:

Data cleaning to remove the noisy log entries;

Data integration to map the log entries to semantically meaningful activities;

Data selection to remove the irrelevant log entries;

Data transformation to generate transactions from the list of log entries to depict users' navigation behaviours;

Of these steps, the task of data transformation is the most difficult and challenging. It can be divided into two sub-tasks: identifying the user click streams and identifying the transactions. By identifying the user click streams, we can group the web log entries into a number of sequences, each of which is labeled with one unique user. The entries inside these sequences are maintained in the order of time. In the second step, efforts are made to break these sequences further into a number of subsequences, called transactions which semantically depict the navigation behaviours.

This chapter begins with introduction on gathering useful data for web usage mining (Section 4.1). Section 4.2 presents approaches to data cleaning and selection. Section 4.3 introduces some terms widely used in data transformation and discusses the tasks of data transformation. Sections 4.4 and 4.5 overview the existing approaches to identifying user click streams and transactions. In addition, Section 4.5 gives a general definition of transaction and presents some novel approaches that are meaningful to extract transactions from web log files of e-learning web sites. Section 4.6 proposes a novel approach to generate fuzzy boundary transactions which possess many advantages over the traditional transactions.

4.1 Data Gathering

Two types of data are especially useful in web usage mining: web log data and meta data. Web log data are the main sources of usage information. Meta data are needed in the process of web usage mining to interpret web usage information and discovered patterns. Meta data include web topology, web content, mapping information, and user profiles.

4.1.1 Web Log Data

Typically, when a user visits a specific web page on one particular web site, the web server of the site automatically records the visiting information of this user. Table 4.1 shows an example of a common entry in the web log file from the web site of the Department of Computing Science, University of Alberta. [1]

Attribute	Value
remote hostname	*edm-ab00-000.netcom.ca*
rfc931 and authuser	- xyz
date/time	*[01/Feb/2001:00:00:06 -0700]*
method, filename, and protocol	*"GET /~somebody/cmput100/chapter1.pdf HTTP/1.1"*
status code	*200*
size	*14882*
referrer	*"http://www.cs.ualberta.ca/~somebody/cmput100/"*
user agent	*"Mozilla/4.0 (compatible; MSIE 5.0; Windows 98; DigExt)"*
cookie	*"uabcs=edm-ab00-000.netcom.ca.XXXXXXXX4"*

Table 4.1: A Common Web Log Entry

This entry contains the following information:

[1]Some contents have been modified on purpose

- The user's **remote hostname** is "`edm-ab00-000.netcom.ca`". This attribute is also called IP address. In some other cases, this attribute looks like "`61.137.172.177`". The value of this attribute indicates the "location" of the user in the world of Internet.

- The value for attribute **rfc931** is not available in this entry, therefore, the minus sign is typically placed in this field. The value for **authuser** is "xyz". These two attributes show the log-in name or identity of the user.

- The **date/time** of the request is "[01/Feb/2001:00:00:06 -0700]".

- The **method of the request** is "GET" and the **filename of the request** is "`/~somebody/cmput100/chapter1.pdf`". In addition, the **protocol of the request** is "HTTP/1.1".

- The **status code of the request** is "200", which indicates the request is successful. Other frequently seen status codes include "404" for the error message that the web page was not found.

- The **size** of the information for the request is 14882 bytes.

- The **referrer** is "`http://www.cs.ualberta.ca/~somebody/cmput101/`", meaning that before the user sent the request, the user was at the URL specified in the field of referrer.

- The **user agent** contains information about the operating system and browser software being used by the current user. In this case, they are "Windows 98" and "MSIE 5.0".

- The **cookie** of the user is "`uabcs=edm-ab00-000.netcom.ca.XXXXXXXX4`". The following is the definition from Netscape about cookies [Net1999]:

  > *"Cookies are a general mechanism which server side connections (such as CGI scripts) can use to both store and retrieve information on the client side of the connection. The addition of a simple, persistent, client-side state significantly extends the capabilities of Web-based client/server applications."*

  The cookies are issued by a server and stored in a user client when the server returns an HTTP object to the user client. These codes are long enough to be unique to one single user client, i.e., the user's computer. This mechanism not only provides web developers with ways to develop many on-line applications, but also sheds light on how to identify a user from web log data.

A web log file normally consists of thousands of entries in the order of time. Most web log files have a similar format. The two prevailing log formats are the CLF (Common Log Format) and ECLF (Extended Common Log Format). A CLF file has fields including: remotehost, rfc931, authuser, time/date, request, status, and bytes. An ECLF file has all the attributes in a CLF file, plus two additional attributes: referrer and user agent. Nonetheless, the server can define additional fields or omit some fields if necessary. For instance, the entry cited above contains the additional field, **cookies**.

4.1.2 Web Topology

Normally, a web site can be described as a graph where each node is matching with one web page, and two nodes are adjacent if there is a hypertext link between them. Web topology, also called web structure, depicts the connections between web pages.

However, it is quite a different task to represent the web topology of a dynamical web site, such as a CGI-based web site, where the web pages are dynamically generated according to the request from users. Yet, it is still possible to find out the "linkage" information which indicates the relationships among requests. For well-designed CGI-based web sites, for example, it is possible to figure out the underlying "invoking" and "invoked" relationships among CGI scripts which are analogous to the "referring" and "referred" relationships among static web pages.

The web structure reflects the web site designers' views on the way to organize information; unfortunately it does not always appear intuitive to the users. In web usage mining, knowledge about web structure not only is useful in the data preprocessing [CMS1999, SCD+2000], but is also valuable in evaluating the discovered patterns [PEt1999, Spi1999]. Moreover, combining this structure knowledge and some data mining results leads to more intuitive introspect about web re-structuring [CMS1997].

4.1.3 Web Content

Web content includes the text, graphics, and other embedded files in the web pages. It is useful to categorize the information domains of the web site based on the web content. Two general ways are used to capture the content of a web page. In one, content providers can summarize the web contents and specify them using HTML meta tags. Another approach makes use of information retrieval techniques to automatically extract keywords to represent the documents.

Considering the web page content within the learning process of web usage mining is significant in its ability to describe users' behaviours and interests in a semantically meaningful manner [JFM1997, FSS1999].

4.1.4 Mapping Information

Mapping information maps web pages into semantically meaningful actions. For e-learning web sites, it is helpful to map the learners' clicks of web pages into more meaningful activities, such as downloading course content, participating in discussions, and taking on-line exam.

Mapping information also helps reduce redundant information in web log files. For example, web pages with the URL `http://www.cs.ualberta.ca/~jun/wum_webpage/main.html` and `http://www.cs.ualberta.ca/~jun/wum_webpage/index.html`, are actually two web pages that appear together on a single screen. This display is enabled by the HTML frames. When analyzing activities related to these two web pages, it is necessary to map these two web pages into one single action since they are indeed perceived as one web page by users.

Moreover, mapping information is necessary to analyze some CGI-based web sites. One particular CGI script can perform different functions for different parameters. On the other hand, several CGI scripts can work together to accomplish a certain task. Therefore, mapping these CGI scripts into the actions can reduce the size of the data mining search space and improve the meaningfulness of the subsequent data mining process.

4.1.5 User Profiles

Some web sites require their users to provide personal information such as gender, age, address, and preferences. The user profiles are valuable in web usage mining for identifying user groups. In the case of e-learning web sites, the user profiles can also contain information such as learners' grades and fields of study. Integrating the users' personal information with the usage information can lead to generating some interesting rules, such as " 81.5% female students finished and submitted the exercises in two weeks after distributing the exercises. Only 47.6% male students did so."

4.2 Data Cleaning and Selection

As discussed before, web log files contain a large amount of erroneous, misleading, and incomplete information. Generally, in order to filter out irrelevant data and noisy data, we specifically inspect the following types of log entries:

error requests or requests reset by users When users send the "reset"/"stop" request to web servers, it means that their former requests are of little use to them. In most cases, these requests do not reflect users' interests so they are irrelevant to many decision problems.

requests using methods other than GET For example, when using the "POST" method in the HTML form to trigger the execution of the CGI script, web servers would not explicitly record the arguments input by users in the log files. While in most cases they are indispensable in identifying users' behaviours, the entries corresponding to this type of request are often incomplete and inaccurate in depicting those behaviours.

requests on the image, audio, video and other embedded files For instance, the web page "a.html" contains five embedded image files. Each time there is a request for the "a.html", the web server records six entries in the web log, with one entry for the "a.html" and five other entries for the five image files respectively. However, it should be noted that those five entries actually come with the visiting "a.html" and, as a matter of fact, users usually perceive the "a.html" and the other five images as a whole web page. So the entries recording requests on embedded files usually are of little value in capturing the users' interests.

executions of CGI Script, Applet, and other Script codes In cases without enough meta data to map these requests into semantically meaningful actions, these records are often too dynamic and insufficiently informative to make sense to decision makers.

requests generated by web agents There are many web agents on the Internet whose function is to pre-fetch pages for caching or indexing. These robots traverse web sites and summarize their content for search engines. Such requests are automatically generated and do not reflect users' interests.

requests generated by proxies It is difficult to identify users for these requests, so it is often meaningless to carry out further data mining process on these data.

others There are various of types of data that can be cleaned out, depending on the goals of decision problems and the settings of web sites.

So far, we have discussed the types of entries which may need to be removed because they are limited in their abilities to provide enough reliable information or of little value to reflect users' interest. However, it should be noted that sometimes these types of data are valuable for some decision problems. We have to carefully select from the above options according to different decision problems. For example, for web sites in which most web pages use multimedia as important components, cleaning out the requests of the embedded files causes too much loss of important usage information.

In Chapter 3, we pointed out that there are also some data that are less or not at all relevant to particular decision problems. Here, we provide the filter mechanism with which instructors can specify constraints to filter out irrelevant data. This filter currently has three dimensions:

users Instructors can specify a group of learners which they are interested in. These specified learners often share some similarities in user profiles.

periods Some web pages exist only for some specified period and the users' navigation patterns are often time-related. For example, instructors can specify the period of interest to be from mid-term exam to one week after.

web pages Decision makers may want to investigate navigation behaviours in only a subset of web pages. For example, some instructors may be interested only in learners' behaviours concerning the on-line discussions.

4.3 Tasks of Data Transformation

Most current researchers in this area make use of **a series of clicks** to model the users' navigation behaviours. Given a specified period, each user has one delimited series of clicks in the order of time. We call this series of clicks a user click stream. Moreover, some researchers further break each user click stream into a number of smaller sets of clicks. In [W3C1999], there are two related definitions, as quoted below:

> **User Session** A delimited series of user clicks across one or more Web servers.
>
> **Sever Session** A collection of user clicks to a single Web server during a user session, also called a visit.

Server session is the one unit usually adopted, although it is sometimes called "session" or "user session". In addition, a server session may be broken into several, more semantically meaningful, units called "transactions" [CMS1997, CMS1999, CPY1998]. A "transaction" in these cases is a subset of related user clicks that occur within a server session. As mentioned in [W3C1999], there is another similar definition — episode:

> An **episode** is a subset of related user clicks that occur within a user session.

Following the convention of user session and server session, we create a new term, **Server Episode**, to match the term "transaction" quoted above. The following example is used to illustrate these concepts.

> *Mr. X activated the Netscape window. Before he shut down the Netscape window, he visited MSN.com and Yahoo.com. In MSN.com, he first read some of the headline news, and then checked email. He logged off MSN.com after chatting on the latest progress in NHL playoff.*

All the browsing behaviours happening after Mr. X opened the Netscape window, and before closing it, are a **user session**. The subset of the navigations which happened in MSN.com are a **server session for MSN.com** in which there are three **server episodes**: reading news, checking emails and chatting.

In this thesis, we use "server episode" instead of "transaction" to name a subset of a server session. "Transaction", as used in this thesis, is a more general concept whose definition will be given in Section 5.4. At this point, we need to know only that a "transaction" is a series of clicks. Therefore, a server session is a special kind of transaction. A server episode is also a type of transaction.

In data transformation, we try to generalize a number of transactions from the user clicks. A transaction implies that there potentially exist relationships between clicks which this transaction contains. That is to say, if some clicks are contained in the same transaction, we initially take a guess that the occurrence of these clicks together is not purely by accident. The subsequent data mining process rejects or validates these guesses by examining them along with all the other transactions.

We can, therefore, estimate approximately the "contribution" of a transaction by counting the number of relationships implied by it. So, given a transaction

$$T = \{Click_1, Click_2, \cdots, Click_n\},$$

we assume that there exist relationships in any subset of this transaction. As shown in Table 4.2, the total number of these relationships indicated by this transaction is approximately $O(2^n)$.

Number of relationships of 2 clicks:	$\binom{n}{2}$
Number of relationships of 3 clicks:	$\binom{n}{3}$
...	...
Number of relationships of n clicks:	$\binom{n}{n}$
Total number of relationships:	$\sum_{m=2}^{n} \binom{n}{m} = 2^n - n - 1.$

Table 4.2: Estimate of the Number of Relationships Inferred by a Transaction

Note that not all these relationships are supported by all the transactions. Data mining algorithms search the patterns and trends which are supported by the majority of the dataset. To extract semantically meaningful transactions, the first step is to identify user click streams. These click streams are then broken into transactions, according to different criteria.

4.4 User Click Stream Identification

4.4.1 Types of Users

According to [Pit1997], users of a web site can be separated into these categories:

unidentified users An unidentified user is a user whose information cannot be traced by a server. On the WWW, a user visiting a web site is required to provide its IP address, such that the server can satisfy the request by knowing where to send the information. In this sense, unidentified users do not truly exist on the Web. However, if most users visit a web site via a proxy, and no additional information is available to the web site, these users are thought to be unidentified because most of the log entries have the same IP address.

session users A session user is a user who can be **approximately** identified using cookies or other heuristics.

tracked users The tracked users can be uniquely and reliably identified across multiple visits to a site. These users are often asked to register before visiting in the web site.

identified users The identified users are tracked users who provide additional personal data.

4.4.2 Methods

Ideally, we can distinguish users by their log-in name. However, the log-in mechanism is not available for all web sites. In this case, to identify users, an initial thought is to use the IP address: the same IP address implies the same user. However, this method is inaccurate for two reasons: first, many users share the same IP address, especially for those who visit via proxy. Second, the same user might not keep using the same IP address. To adjust this method, the assumption is applied that one user only uses one combination of the browser software and operating system. For example, users with same IP address but using different agents[1] — say, Internet Explorer in Windows 98 and Netscape in Linux — are treated as two distinct users under this assumption.

Cookies generation provides a way to uniquely identify the user machine. By using cookies, not surprisingly, we can identify a user more precisely than by using the IP address only. However, it is still important to be aware that the correspondence between cookies and users is still a many-to-many correspondence. One user can use different machines with different cookies; as well different users may use the same machine with one cookie address.

There are several other methods to identify user click streams. Most of them have been discussed in [Pit1997, CMS1999]. Table 4.3 shows to what extent these

[1]According to the previous definition, a combination of the browser software and operating system is called an agent.

METHOD	IDENTIFIED USER TYPE
IP/Agent	session user
IP/Web topology	session user
cookies	session user
registration id	tracked user, identified user
client side tracing software	tracked user, identified user

Table 4.3: Methods to Identify User Click Streams

methods are able to identify users. In web-based learning environments, learners are usually asked to register at the beginning of their visit. In this sense, the learners are all identified.

4.5 Transaction Identification

After identifying users, for each user we can get a sequence of clicks with time stamp, called a user click stream. Usually a click stream contains hundreds of clicks, depending on the time span of the analyzed log. However, these click streams have to be further broken into a number of transactions before data mining algorithms can be carried out. This section provides insight into the motivation for identifying transactions, and then gives a formal definition of transaction. Based on this definition, we review the existing methods and propose our approaches.

4.5.1 Motivation

The following reasons make transaction identification indispensable for web usage mining.

- First of all, transaction is a basic conceptual unit to depict user navigation behaviours. The underlying relationships between clicks in one transaction are the mining resources of most web usage mining algorithms.

- It is widely known that users' interest in a web site, as well as the structure and content of web sites, keep changing. Therefore, if there are two clicks, for example, belonging to one user, one happening 15 seconds ago, and the other 5 months ago, in most cases it is reasonable to assume that there is no strong connection between the two clicks. On the other hand, if that same user visits a second web page immediately after the first, we can assume that there may be a strong connection between the clicks.

- Sometimes, breaking click streams down into several transactions can reduce the error risks in user identification. As we have discussed, the time span of click streams can be so long that there may be many clicks belonging to other users. Transactions, on the contrary, are the subsequences of click streams. Since a transaction lasts a much shorter time than a click stream, the assumptions of methods for identifying users probably hold inside a transaction. Thus, we have more confidence in saying that clicks in one transaction probably belong to one single user.

- Even when the user click streams have been correctly identified, transactions provide more reliable information for most data mining algorithms. Many data mining algorithms, such as association rule mining, sequential pattern analysis, and some of the clustering algorithms, require a transaction-like dataset. Although click streams basically fit in this format, we use the transactions instead because the click streams provide too much irrelevant information, which potentially misguides data mining algorithms.

4.5.2 Definition of Transaction

Based on previous discussions, we give a formal definition of transaction:

> **A transaction is a sequence of clicks that are "close" to one another according to some criteria.**

In other words, we believe that there probably exist some strong connections between the clicks inside a transaction. In order to estimate the "closeness" between clicks, the following criteria are normally used:

server session and server episode Users of a web site begin visiting the web site from some specific web pages and, after getting information from several other web pages, they leave the site. Most researchers regard a server session as a transaction. However, it is almost impossible to tell exactly when a user leaves a web site, unless there are some particular mechanisms requiring this user to provide additional information when navigating — such as registering. As previously mentioned, a server episode is a subset of the server session and it is a semantically meaningful concept unit. A user generally has one particular information need during a server episode.

Clicks in a server session or a server episode are analogous to the products bought by a customer in the supermarket during one shopping trip. They are "close" to each other together because they satisfied a user's need for information during one visit.

time Not all the web sites provide mechanisms that help identify server sessions and server episodes. In these cases, the timestamp associated with each click sheds light on estimating the "closeness" between the clicks. If clicks do not happen a long time apart, there possibly exists some connection between them.

web topology If the web pages visited in two clicks can be linked together through only a few hyperlinks, we assume these two clicks are "close" to each other.

web content Being related in content can provide other evidence to support the relationship between clicks.

domain knowledge Sometimes, domain knowledge can help to determine whether there is a relationship between two clicks or not.

4.5.3 Identifying Transactions by Time Out

This approach has been widely utilized in most of the existing web usage mining systems because it is simple and does not require additional information, other than the web log. It takes the time interval between two consecutive clicks into consideration: given an undetermined click, if the time interval between this click and the immediately previous one is less than the pre-defined threshold, then it should be included in the same transaction as the one to which the previous one belongs. Otherwise, this click should be the first click in a new transaction.

In [CPi1995], L.D.Catledge and J. Pitkow calculated the time-out between every two clicks of one user. Their results indicate that the mean value of those time-outs was 9.3 minutes. By adding 1.5 deviations from the mean, 25.5 minutes was used as the maximum time-out of two adjacent clicks in one transaction. R. Cooley et al. in [SCD+2000, CMS1999] also adopted this approach. Their WebSIFT used 30 minutes as the threshold of the time-out. In other applications, [NFJ+1999, JJY+1999] followed same approach but used 45 minutes as the threshold.

Obviously the most critical issue about this method is to find an appropriate threshold. This is usually determined by the different settings of web sites. For example, for a search engine web site, this time threshold can be very short. On e-learning web sites, however, where learners would probably spend a relatively long time in downloading and reading content, this time threshold would probably be much longer.

4.5.4 Identifying Transactions by Time Duration

Assuming that meaningful transactions have an average length associated with them, we can determine transactions by setting a time window, such that the time between any two clicks in a transaction will not exceed a specified threshold [CMS1999]. For example, we can consider the clicks occurring within one day to belong to one transaction. This approach is useful in cases where decision makers want to investigate user behaviours over a specified period.

This approach is not very commonly used because it is possible to divide two consecutive clicks which happened close in time into different transactions. For example, if 60 minutes is chosen as the length of time window and there are three clicks in a user click stream. The idle time between them are 59 minutes and 3 minutes subsequently. By this method, the first two clicks are grouped into a transaction, and the third one belongs to another transaction. It is not reasonable because obviously the second click is much closer to the third one than to the first click.

4.5.5 Identifying Transactions by Other Criteria

In some cases, if meta data describing the features of web sites is available, we can adopt other methods to identify transactions.

- Transactions can also be identified by utilizing web topology, based on the observation that a user normally visits a new web page from previously visited pages during a single visit. Hence, within a user click stream, a click which cannot be linked to by visited pages is considered to belong to another transaction.

  Another observation is that instead of writing a new URL, visitors are inclined to use the back, then forward, selections to browse the sibling web pages from the current page. Based on this assumption, a forward reference sequence, which indicates that the visitor is looking for some specific information, terminates when a backward reference occurs. Transactions are obtained by finding the maximum forward reference sequences. For example, if a visitor browsed the web pages in sequence: (A, B, C, A, D), this sequence can be looked as two transactions, (A, B, C) and (A, D) — because from page C to page A, the visitor is supposed to press the back button in browser to begin another topic search [CPY1998, CMS1999]. This approach is initially used to identify server episodes.

For web sites where frames are used, the methods utilizing web topology information often fail because with frames, most web pages can be linked together easily.

- The content of web pages can also provide hints for determining whether to group two clicks in a transaction. For example, in a search engine web site, the query terms specified by users shed light on understanding the users' information needs. When a user searches for two different but related terms, such as "Amazon" and "River in South America", he/she is probably looking for information resources related to the Amazon river in South America. In this case, this user apparently cannot get the desired answer set in the first query because most search engines will generate a long answer list related to the amazon.com bookstore. The second trial is trying to clarify his/her information need. These two queries should be considered as one transaction, since they are both express the same information need of that user.

4.5.6 Customizing Transactions

Decision makers can integrate their domain knowledge by customizing transactions. In an e-learning web site, for example, instructors might want to know how learners work after knowing the date of an on-line exam, and before taking that exam. Instructors expect some patterns and trends in this period. Therefore, it is very useful to provide a mechanism to enable decision makers to define different transactions according to different decision problems. The following parameters are used to specify customized transactions:

page Wherever there is a click of this **page**, we start a new transaction from this click.

(page:time) We generate transactions beginning with the **page** and lasting for the length of **time**.

(page A:page B) We generate transactions beginning with **page A** and ending with **page B**.

In Chapter 5, we integrate this mechanism and the sequential pattern analysis algorithm to find patterns of interest.

4.5.7 Adjusting the Pre-defined Time-out Threshold

Of above approaches, identifying transactions by time-out is the most widely used, due to its simplicity. However, given a time-out threshold, it is very difficult to evaluate its validity. The smaller the threshold is, the more transactions it will generate and the fewer are the clicks in one transaction. This affects data mining results, especially for association rule mining and sequential pattern analysis.

It is not realistic to expect instructors to know what value is proper for the threshold, although in our system they are free to choose any threshold. It would be easier for instructors to just specify an initial value, from which our approach can automatically search for an appropriate value for the threshold for the targeted web log data.

Figure 4.1 shows a typical distribution of interval lengths. As we can see, there is a large exponential component in this distribution when lengths of interval are relatively short, while with increasing length, the distribution becomes even. We interpret this phenomenon along with users' on-site and off-site behaviours. When a user is visiting a web site and reading a web page, the idle time between the current request and the next one, is influenced by the type of the visited web page. If the web page is rich in content, the idle time should be long. If the web page is just an index page, the idle time should be short. As a consequence, the distribution of intra-transaction intervals[1] is related to the types of visited web pages and does not appear in uniform trends. On the contrary, the length of inter-transaction intervals[2] is quite random. With the assumption that an off-site interval is longer than an on-site interval, given a cutoff time as the threshold, we expect to see an exponential distribution of intervals in [0, cutoff time], and a uniform distribution after that.

In addition, ideally if a time interval is larger than the threshold, then it would be **significantly** larger. Otherwise, it is hard to distinguish inter-transaction intervals

[1]The interval between two consecutive clicks which belong to one transaction.

[2]The interval between two consecutive clicks which belong to two adjacent transactions, respectively.

from intra-transaction intervals — since the small inter-transaction intervals are very close to the large intra-transaction intervals.

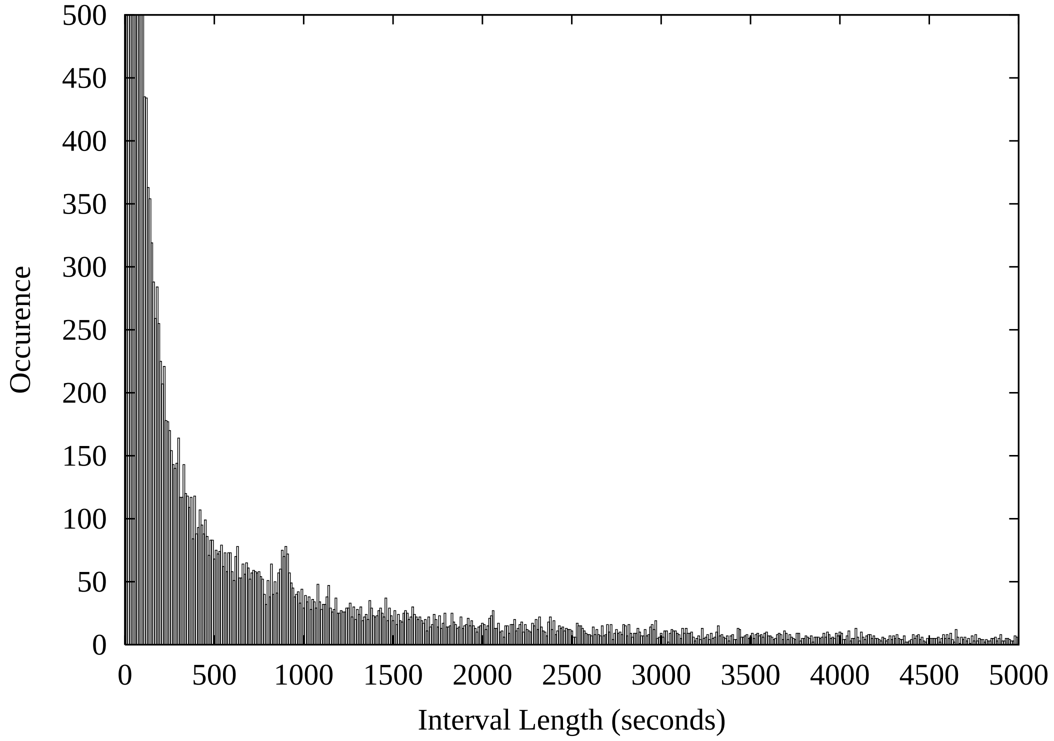

Figure 4.1: Distribution of Interval Lengths

Based on previous discussion, although a perfect cutoff time for the threshold does not really exist, we can still estimate the quality of a given threshold using some function. Let $|I[t_1, t_2]|$ ($t_1 < t_2$) denote the number of intervals whose lengths are between t_1 and t_2:

$$MeasureFunction(threshold) = \frac{|I[threshold, threshold + \alpha \cdot \Delta t]|}{|I[threshold - \alpha \cdot \Delta t, threshold]|}$$

where α is a smooth factor to reduce influences of noise and Δt is a specified range of time. These two factors can be modified if necessary. If the value of $MeasureFunction(threshold)$ is small enough, it means that there are fewer inter-transaction intervals whose lengths are close to the threshold is small, than intra-transaction intervals whose lengths are close to the threshold. We believe that this threshold is a proper cutoff time since most inter-transaction intervals are larger than $(threshold + \Delta t)$.

By using the algorithms shown in Table 4.4, given an initial value and an increasing pace, we can adjust the value of the threshold from an initial value to a more appropriate one. Actually, the initial value is used to eliminate unnecessary consideration of the small values for the threshold[1]. The modified value will not be too large, since as the length of interval increases, the distribution becomes uniform, and the *MeasureFunction* is close to 1 in this case. If a better value cannot be found, we reset the threshold to the initial value.

Require: $initial\_value, \Delta t, smallenough \in (0,1), \epsilon$ is a value close to 0
Ensure: threshold (modified)

```
  $threshold = initial\_value$
  while $MeasureFunction(threshold) \geq smallenough$ do
    if $MeasureFunction(threshold) \geq 1 - \epsilon$ then
      $threshold = initial\_value$
      break;
    end if
    $threshold = initial\_value + \Delta t$;
  end while
return $threshold$.
```

Table 4.4: Algorithm: Adjusting the Time-out Threshold

4.5.8 Modifying Transactions

Each transaction groups a number of clicks that are "close" to one another according to some criteria. With this definition, we expect two kinds of errors in identifying transactions:

- The first type of error separates a set of clicks into several transactions when they should be in the same transaction. In this case, we lost information of the relationships between clicks separated in the different transactions.
- The second type, by contrast, groups a set of clicks in one transaction when they should be separated into several transactions. In this case, this transaction indicates some relationships which do not really exist.

In a practical sense, it is not easy to develop formulae to estimate errors. However, it is possible to list some detectable situations in which transactions are **probably**, not absolutely, wrongly identified [HGk2000]. We list four typical situations. The first three situations correspond to the former type of error, while the last one infers the latter type of error. We illustrate these situations using a sequence as

[1]*MeasureFunction*(*threshold*) is inclined to have a small value when the threshold is small. Hence, an initial value can prevent generating a too small threshold.

an example: $(T_0, I_0, T_1, I_1, T_2)$, where T_0,T_1,and T_2 are three consecutive identified transactions and I_0 and I_1 are the intervals between them.

Situation 1: The last click of one transaction is very closely related to the first click of the next transaction according to some evidence. For example, the web pages visited by the last click of T_1 and the first click of T_2 are both related to the same topic.

Situation 2: There are only a few, for example, less than 3, clicks in a transaction. As previously discussed in this chapter, the transaction with fewer clicks provides less useful information. In addition, in many web sites, visits with only one or two clicks are very rare.

Situation 3: The time length of one transaction is significantly larger than the length of the following interval. For example, the length of T_0 is 2 hours, while the length of I_0 is only 20 minutes.

Situation 4: There are too many clicks in a transaction. The number is arbitrarily chosen.

In these situations, there are probably some errors in identifying transactions. It is feasible to develop some approaches to modify the identified transactions based on the above observations. In the first three situations, we can easily merge the transaction to its neighbour transaction. In the last situation, we need additional strong evidence to break a transaction into a number of subsequences.

4.6 Fuzzy Boundary Transaction

We can always list some abnormal situations in transaction identification. However, it is not practical to reduce errors by merely relying on observation. Of the two types of errors in identifying transactions, the first type would eventually cause the loss of some interesting patterns.

In addition, most of the traditional methods for identifying transactions imply transitivity between clicks in a transaction. That is, if click A and click B are believed to be in a transaction, and click B and click C are also supposed to be in one transaction, then so are click A and click C. It is a very strong assumption which is not necessarily valid for depicting the navigation behaviours.

These problems are caused mainly by the clear boundaries of traditional transactions. Since, in many cases, there is not sufficient information supporting whether a click should be included in a transaction or not, the fuzzy set concept is well suited to solve this problem by providing a way to depict a smooth transition between a member and a non-member of a transaction. Moreover, when we break a click stream into transactions, we only apply one simple standard to summarize complex and uncertain navigation behaviours in the traditional transaction identification. However, with the fuzzy membership function, we can involve multiple criteria in identifying transactions.

4.6.1 Fuzzy Logic, Fuzzy Set, and Logic Operations

Fuzzy logic introduces the concept of partial truth to handle the widely existing uncertainty in the real world [Zad1965]. Given a statement, instead of evaluating it with "completely true" and "completely false", fuzzy logic uses a truth value in the range of $[0, 1]$ to tell to what extent you can trust this judgment. For example, a statement with truth value 0.8 is more reliable than one with truth value 0.4.

Each element in a fuzzy set F has two values: one is the element itself; the other is the degree of membership of this element in F. For example, in a fuzzy set $FS = \{(a, 0.4), (b, 1), (c, 0.8)\}$, the degrees of membership of a,b and c in FS are 0.4, 1, and 0.8, respectively.

In addition, we can perform logic operations in the fuzzy set. Following the last example, if we have two statements, X and Y:

$$X : a \in FS \text{ and } Y : c \in FS;$$

then

$$\begin{array}{lclcl} truth(notX) & = & 1.0 - truth(X) & = & 1.0 - 0.4 = 0.6 \\ truth(XandY) & = & min(truth(X), truth(Y)) & = & min(0.4, 0.8) = 0.4 \\ truth(XorY) & = & max(truth(X), truth(Y)) & = & max(0.4, 0.8) = 0.8 \end{array}$$

4.6.2 Generating Fuzzy Boundary Transactions

The basic idea of our approach is to examine how the clicks in a transaction are related to its neighbour transaction. As shown in Table 4.5, this approach consists of two major steps: First we identify transactions using previously introduced methods. The degrees of membership of clicks in current transactions are all set to 1, meaning that these clicks totally belong to these transactions. The fuzzy membership function is then applied to estimate the degrees of membership of clicks in their neighbour transactions. If the degrees are large enough, we append these clicks to their neighbour transactions, with the values of the degrees.

Input: Identified Transaction T
Output: Fuzzy Boundary Transaction FBT

```
  for all $t \in T$ do
    $fbt \Leftarrow \{\}$;
    for all $click \in t$ do
      $fbt = fbt \cup \{(click, 1)\}$ ;
    end for
    $FBT = FBT \cup fbt$
  end for
  for all $fbt \in FBT$ do
    for all $click \in fbt$ do
      Let $nbt$ be the neighbour transaction of $fbt$;
      $fv = fuzzyMembershipFunction(click, nbt)$;
      if $fv$ is large enough then
        $append(click, fv, nbt)$; {append the $click$ into the $nbt$ with $fv$}
      end if
    end for
  end for
return $FBT$.
```

Table 4.5: Algorithm: Generating the Fuzzy Boundary Transactions

We use the following example to illustrate our idea of generating fuzzy boundary transaction:

In the first step, we break the click stream (a b c d e g c d f a g) into three transactions, namely (a b c d e), (g c d), and (f a g). At the beginning of "fuzzification", we assign 1 to the degree of membership of all the elements. Then we carry out a process of forward fuzzification or backward fuzzification on the three initial transactions. The forward fuzzification appends clicks with the degrees of membership to their subsequent transaction. In this example, to "fuzzify" transaction 1, we find that (g,1) in transaction 2 should be included in transaction 1 with degree of membership 0.6, and then we append (g,0.6) to transaction 1. This procedure is iteratively executed until we cannot find any click in transaction 2 that can be appended to transaction 1. We then apply the same strategy to fuzzify transaction 2, and so on so forth.

By contrast, the backward fuzzification inserts clicks into a transaction from its last neighbour. Either the forward fuzzification or the backward fuzzification

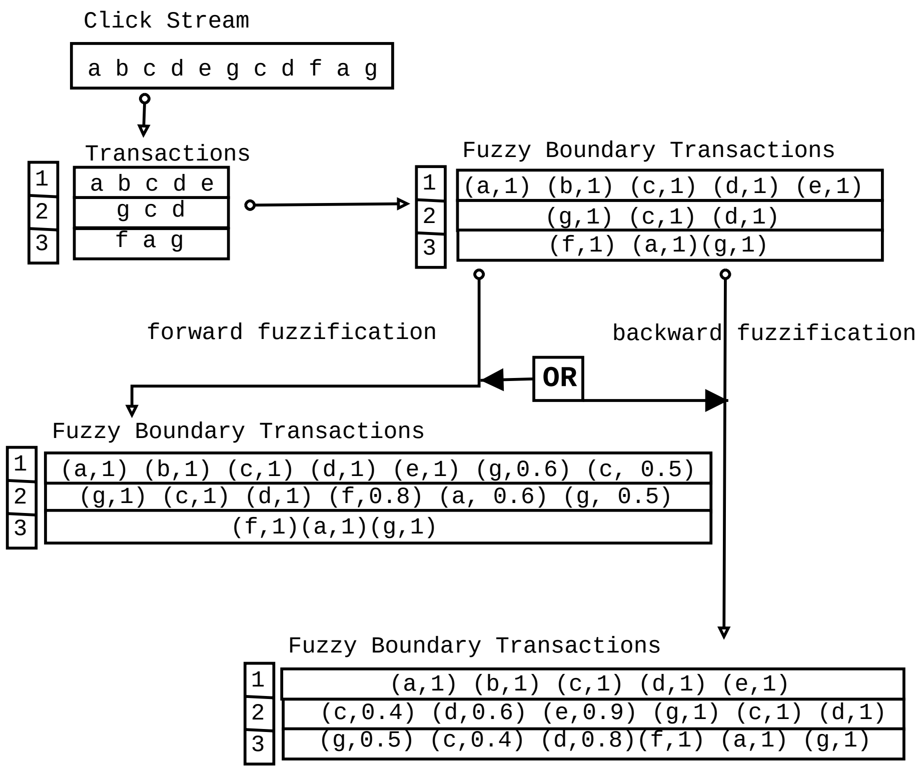
Click Stream
a b c d e g c d f a g
Transactions
1 a b c d e
2 g c d
3 f a g
Fuzzy Boundary Transactions
1 (a,1) (b,1) (c,1) (d,1) (e,1)
2 (g,1) (c,1) (d,1)
3 (f,1) (a,1)(g,1)
forward fuzzification
backward fuzzification
OR
Fuzzy Boundary Transactions
1 (a,1) (b,1) (c,1) (d,1) (e,1) (g,0.6) (c, 0.5)
2 (g,1) (c,1) (d,1) (f,0.8) (a, 0.6) (g, 0.5)
3 (f,1)(a,1)(g,1)
Fuzzy Boundary Transactions
1 (a,1) (b,1) (c,1) (d,1) (e,1)
2 (c,0.4) (d,0.6) (e,0.9) (g,1) (c,1) (d,1)
3 (g,0.5) (c,0.4) (d,0.8)(f,1) (a,1) (g,1)

Figure 4.2: Fuzzy Boundary Transaction

fuzzifies only one end of the two boundaries of a transaction. The reason we do not fuzzify both sides of a transaction is that if we do so, there would be duplicate information. As is shown in Figure 4.3, the pair [(e,1) (g,0.6)] in transaction 1 and the pair [(e, 0.9) (g,1)] in transaction 2 actually correspond to the same pair in the original transactions. However, it should be noted that there are still some other redundancies in a one-direction fuzzification. However, it is easy to eliminate these redundancies when performing pattern discovery algorithms. Related issues will be discussed in Chapter 5.

4.6.3 Designing Fuzzy Membership Function

To generate fuzzy boundary transactions, adopting appropriate fuzzy membership function is critical. In the current stage, we simplify the problem of estimating the degree of membership of a click to its neighbour transactions by measuring the "closeness" of this click with the last/first click in the neighbour transaction. In other words, if a click is believed to be closely related to another click, which is in the boundary of the neighbour transaction, then the estimated value of their relationship

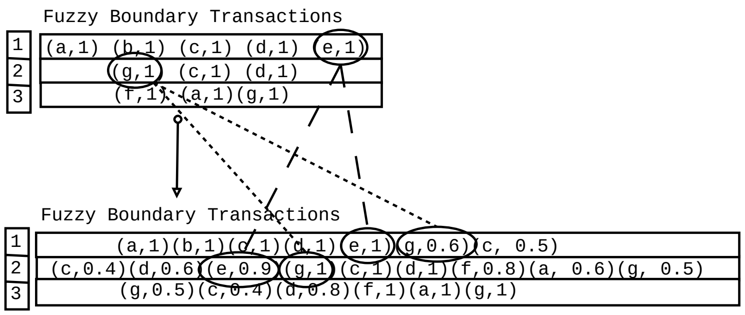
Fuzzy Boundary Transactions
1 (a,1) (b,1) (c,1) (d,1) (e,1)
2 (g,1) (c,1) (d,1)
3 (f,1) (a,1)(g,1)
Fuzzy Boundary Transactions
1 (a,1)(b,1)(c,1)(d,1)(e,1)(g,0.6)(c, 0.5)
2 (c,0.4)(d,0.6)(e,0.9)(g,1)(c,1)(d,1)(f,0.8)(a, 0.6)(g, 0.5)
3 (g,0.5)(c,0.4)(d,0.8)(f,1)(a,1)(g,1)

Figure 4.3: Duplicate Information in Two-direction Fuzzification

is used as the degree of membership of this click in the neighbour transaction. Following the previous example with the forward fuzzification, in transaction 2, if we find the (g,1) is closely related to the last item in transaction 1 — (e,1), we measure this relationship with value in [0,1] and use this value as the degree of membership of (g,1) in transaction 1.

To estimate the closeness of one click to another, as addressed in the previous discussion, there are many useful methods. Here, we mainly take into consideration three factors: the number of other clicks between the two clicks, the time span between them, the similarity between the URLs. Actually, other factors can be easily added into this computing framework if desired.

- The fewer the number of clicks between the two targeted clicks, the more likely that these two clicks are related. In our research, we use the following formula to measure the value of the closeness of two clicks according to their positions. Let $\Delta Click$ denote the number of clicks between two clicks. The closeness of these two clicks is evaluated as following:

$$simClk(\Delta Click) = \frac{4}{5 + e^{2 \cdot \Delta Click - 8}}$$

  As we can tell from Figure 4.4, if $\Delta Click$ is less than 4, the value of closeness is around 0.8. If $\Delta Click$ is more than 6, then we consider there is no such closeness existing. Thus, the value is very near 0.

- If two clicks happen close to each other in time, they are supposed to be related. Let $\Delta Time$ denote the idle time between these two clicks, then the formula we use to estimate the similarity according to the time distance is:

$$simTime(\Delta Time) = \frac{\text{cutoff time}}{\Delta Time}$$

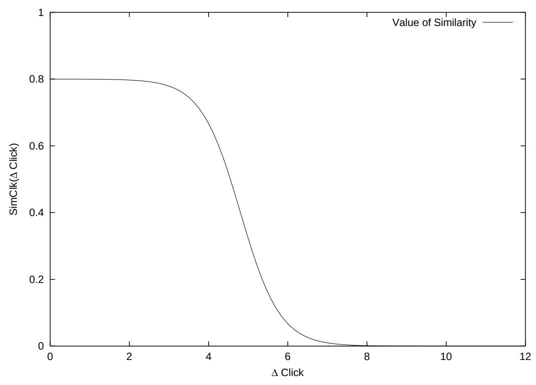

Figure 4.4: Value Curve of $simClk(\Delta Click)$

The cutoff time can be specified by decision makers, or can be directly taken as the mean length of the inter-transaction intervals.

- If the two URLs of the web pages visited by the two clicks are very similar, we assume they are related. The formula we adopt here is:

$$simURL(URL_1, URL_2) = \frac{\text{The length of similar URL}}{\text{The length of the longest URL}}$$

The length of a URL is equal to the number of directories in the URL. For example, there are three URLs:

A: `http://www.cs.ualberta.ca/research/`,

B: `http://www.cs.ualberta.ca/programs/graduate/faq.html`

C: `http://www.cs.ualberta.ca/programs/graduate/gapsinfo.html`.

Then the length of A is 2, while the lengths of B and C both are 4. Using previous function, we can get:

$$\begin{array}{lll} simURL(A,B) & = 1/4 & = 0.25 \\ simURL(B,C) & = 3/4 & = 0.75 \\ simURL(C,C) & = 4/4 & = 1 \end{array}$$

Finally, we design our membership function based on these three values.

$$\begin{aligned} & fuzzyMembershipFunction(click, nbt) \\ = \; & distance(click, \text{the } boundaryClick \text{ in } nbt) \\ = \; & min(max(simURL, simTime), simClk) \end{aligned}$$

where

nbt	:	neighbour transaction;
boundaryClick	:	The first or the last click in the neighbour transaction;

This equation can be translated into the following rule:

```
If     { (the click and boundaryClick happen in a short period) OR
       (these two clicks are similar in the URL) } AND
       (there are only a few clicks between these two clicks)
Then   include this click in nbt;
```

4.7 Summary

Web log entries typically contain information such as the identity of the user, the time of the request, the file of the request, etc. The useful meta data in the context of web usage mining include web topology, web content, mapping information, and user profiles. The modules of data cleaning and data selection let instructors flexibly remove irrelevant data according to different decision problems.

Given a cleaned web log, we begin data transformation with identifying user click streams and then break these click streams into transactions. A transaction is a sequence of clicks that are “close” to one another according to some criteria. The criteria can be time, web topology, web content, or domain knowledge. In order to avoid distracting instructors with issues related to details of web usage mining techniques, we propose an approach to automatically adjust the pre-defined time-out threshold to the targeted dataset.

By introducing the concept of fuzzy boundary transactions, we manage to preserve more information in each transaction which hopefully would lead to more meaningful rules and patterns, reduce the limitations brought by the transitivity assumption of traditional transactions, and integrate multiple criteria into generating fuzzy boundary transactions.

Chapter 5

Constraint-based Sequential Pattern Analysis

In Chapters 2 and 3, we introduced various useful patterns in web usage mining. Among these, sequential pattern analysis is one of the most widely used data mining algorithms, due to its strength in demonstrating web usage trends and patterns. There are also many variants of sequential patterns analysis. In this thesis, our definition of sequential pattern analysis is similar to the definition proposed in [SFW1999], with some slight differences. Unlike [ASr1995, ASr1996, MPC1999], our algorithm does not search for sequential patterns beyond boundaries of transactions. Instead, we focus on discussing how to push the specified constraints into the searching process. In addition, we demonstrate how to make our algorithm work in searching for patterns of interest, by combining interactions in other phases of web usage mining.

This chapter focuses on sequential pattern analysis, which aims to discover the sequences of actions that happen frequently in a time-ordered transaction dataset. The first two sections (Section 5.1 and 5.2) present a formal definition of sequential pattern analysis problem, and analyze an *apriori*-like algorithm along with the properties of sequential pattern. We then present in Section 5.3 several kinds of interactions applicable in the whole web usage mining process regarding sequential pattern analysis. Finally, in Section 5.4, we extend the algorithms to handle fuzzy boundary transaction dataset.[1]

[1]Our approach is different from C. M. Kuok el al.'s algorithm [KWW1998] for mining fuzzy association rules in the database. Their approach deals with the transactions with quantitative data, and the fuzzy concept is used to discretize the continuous values.

5.1 Problem Definition

A typical sequential pattern indicates important sequences that are repeatedly followed by users. The following are some important definitions to describe sequential patterns.

action An action represents that a user browses one particular web page or one particular group of web pages. We use a to represent one specified action.

sequence A sequence is a time-ordered series of actions. Let s denote one sequence such that $s = a_1 a_2 \cdots a_n$, where a_i happens before a_j if and only if $i < j$. n is the length of s in this case.

k-sequence A sequence of length k.

supersequence and subsequence Given two sequences $s = a_1 a_2 \cdots a_n$ and $s^{'} = a^{'}_1 a^{'}_2 \cdots a^{'}_m$, we say that s is a subsequence of $s^{'}$ and $s^{'}$ is a supersequence of s, denoted as $s \subseteq s^{'}$, if and only if $\exists$ a series of numbers $i_1, i_2, \cdots, i_n$ and $1 \leq i_1 < i_2 < \cdots < i_n < m$, such that $a_k = a^{'}_{i_k}$ for $\forall k \in [1, n]$.

order of sequences Given two sequences $s = a_1 a_2 \cdots a_n$ and $s^{'} = a^{'}_1 a^{'}_2 \cdots a^{'}_m$, if a_n happens before $a^{'}_1$, we define this kind of relationship as $s \prec s^{'}$.

concatenate two sequences Given two sequences $s = a_1 a_2 \cdots a_n$ and $s^{'} = a^{'}_1 a^{'}_2 \cdots a^{'}_m$, if $s \prec s^{'}$, we can append $s^{'}$ to s to generate a new sequence. $s^{''} = s \cdot s^{'} = a_1 a_2 \cdots a_n a^{'}_1 a^{'}_2 \cdots a^{'}_m$.

contain If a sequence s contains another sequence $s^{'}$, then s is the supersequence of the $s^{'}$, vice versa.

sequential pattern Given a sequence dataset $\mathcal{S} = \{s_1, s_2, \cdots, s_n\}$, $\mathcal{X}_s$ and $\mathcal{Y}_s$ are two sequences, then $\mathcal{X}_s \Rightarrow \mathcal{Y}_s$ is a sequential pattern whose significance is measured by *support* and $confidence$: [1]

$$\begin{aligned} support(\mathcal{X}_s \cdot \mathcal{Y}_s) &= \frac{|\{s | s \in \mathcal{S} \text{ and } \mathcal{X}_s \cdot \mathcal{Y}_s \subseteq s\}|}{|\mathcal{S}|} \\ confidence(\mathcal{X}_s \Rightarrow \mathcal{Y}_s) &= \frac{|\{s | s \in \mathcal{S} \text{ and } \mathcal{X}_s \cdot \mathcal{Y}_s \subseteq s\}|}{|\{s^{'} | s^{'} \in \mathcal{S} \text{ and } \mathcal{X}_s \subseteq s^{'}\}|} \\ &= \frac{support(\mathcal{X}_s \cdot \mathcal{Y}_s)}{support(\mathcal{X}_s)} \end{aligned}$$

[1] |a finite set| indicates the cardinality, i.e., the size of this set.

The value of *support* shows the statistical dominance of the sequential patterns while the value of *confidence* depicts the conditional probability of $\mathcal{Y}_s$ given $\mathcal{X}_s$

Based on above definitions, **a sequential pattern analysis algorithm aims to find sequential patterns in the form of $\mathcal{X}_s \Rightarrow \mathcal{Y}_s$, such that**

$$\begin{aligned} support(\mathcal{X}_s \cdot \mathcal{Y}_s) &\geq min\_sup, \\ confidence(\mathcal{X}_s \Rightarrow \mathcal{Y}_s) &\geq min\_conf. \end{aligned}$$

$$(min\_sup \in (0,1), min\_conf \in (0,1])$$

5.2 Algorithm

In this section, we introduce a level-wise algorithm, i.e., an *apriori*-like algorithm to search sequential patterns [ASr1994]. Two important properties of sequential pattern make it feasible to extract the sequential patterns effectively and efficiently.

5.2.1 Properties

In the two properties of a sequential pattern described here, one is related to the *support* of the sequential pattern, the other is concerned with *confidence*.

Property of support If the *support* of one sequence s, $support(s) \geq min\_sup$ ($min\_sup \in (0,1)$), then $\forall s^{'} \subseteq s$, $support(s^{'}) \geq min\_sup$. This property is also called downward closure property in [SRC1997].

Proof It is evident that $\forall s_i \in \mathcal{S}$, if $s \subseteq s_i$, then $s^{'} \subseteq s_i$. However, the fact that $s^{'} \subseteq s_i$ does not infer that $s \subseteq s_i$. In other words, providing $s^{'} \subseteq s$,

$$|\{s_i | s_i \in \mathcal{S} \text{ and } s \subseteq s_i\}| \leq |\{s_i | s_i \in \mathcal{S} \text{ and } s^{'} \subseteq s_i\}|$$

Then

$$\begin{aligned} support(s^{'}) &= \frac{|\{s_i | s_i \in \mathcal{S} \text{ and } s^{'} \subseteq s_i\}|}{|\mathcal{S}|} \\ &\geq \frac{|\{s_i | s_i \in \mathcal{S} \text{ and } s \subseteq s_i\}|}{|\mathcal{S}|} \\ &= support(s) \\ &\geq min\_sup, \text{ as required.} \quad \blacksquare \end{aligned}$$

Property of confidence Let $s_1 = a_1 a_2 \cdots a_i$ and $s_2 = a_{i+1} a_{i+2} \cdots a_n$, $s_1^{'} = a_1 a_2 \cdots a_{i-1}$ and $s_2^{'} = a_i a_{i+1} a_{i+2} \cdots a_n$, then $confidence(s_1 \Rightarrow s_2) \geq confidence(s_1^{'} \Rightarrow s_2^{'})$

Proof Following the definitions, we can state $s_1 \cdot s_2 = s_1^{'} \cdot s_2^{'}$.

$$\begin{aligned} confidence(s_1 \Rightarrow s_2) &= \frac{support(s_1 \cdot s_2)}{support(s_1)} \\ &= \frac{support(s_1^{'} \cdot s_2^{'})}{support(s_1)} \\ &\geq \frac{support(s_1^{'} \cdot s_2^{'})}{support(s_1^{'})} \\ &= confidence(s_1^{'} \Rightarrow s_2^{'}), \text{ as required.} \quad \blacksquare \end{aligned}$$

5.2.2 Algorithm Analysis

Our algorithm is a two-step approach for searching and generating sequential patterns. At first, the algorithm searches frequent sequences whose supports are larger than the given threshold. It then generates sequential patterns with enough confidence from the discovered frequent sequences.

The property of *support* sheds light on the level-wise searching strategy for frequent sequences. This strategy, as shown in Table 5.1, begins with finding the frequent actions as frequent 1-sequences, and then uses these sequences to generate the sequences of size 2, size 3 and so on until it cannot find longer frequent sequences. In each level of searching, the property of support effectively reduces the efforts to generate candidates of the longer frequent sequences since according to this property, any subsequence of a candidate is a frequent sequence itself. These candidate frequent sequences are then tested to see if their supports are large enough.[1] Thereafter, the justified frequent sequences are used to construct longer frequent sequences in the next level of searching. In short, the task of the first step is accomplished by level-wise candidate generating and justifying.

In the second step (Table 5.2), the property of *confidence* is a valuable aid in finally generating sequential patterns. For a frequent sequence, if a pattern with the first k actions in the predicate set satisfies the confidence requirement, then the patterns with more than k actions in the predicate set all satisfy the confi-

[1]Scanning the dataset is an obvious but inefficient approach.

dence requirement. Therefore, we need to find only the smallest k for each frequent sequence.

```
Input: Sequence Dataset $S_D$,
  minimum support $min\_sup$, minimum confidence $min\_conf$.
Output: Sequential Patterns $SP$
  with $support(SP) \geq min\_sup$ and $confidence(SP) \geq min\_conf$.

  Find the frequent actions, {i.e. $freq\_1\_seq$};
  $k \Leftarrow 1$;
  $Freq\_Seq \Leftarrow \phi$;
  repeat
    $k \Leftarrow k + 1$;
    $freq\_k\_seq = gen\_freq\_k\_seq(freq\_(k-1)\_seq, freq\_1\_seq)$;
    $Freq\_Seq \Leftarrow Freq\_Seq \cup freq\_k\_seq$;
  until no more $freq\_k\_seq$ found;
  $SP \Leftarrow \phi$;
  for all $fs \in Freq\_Seq$ do
    $SP \Leftarrow SP \cup gen\_pattern(fs)$;
  end for
return $SP$.
```

Table 5.1: Algorithm: Sequential Pattern Analysis

```
  $freq\_k\_seq \Leftarrow \phi$;
  for all $seq \in freq\_(k-1)\_seq$ do
    for all $action \in freq\_1\_seq$ do
      $candidate = seq \cup action$;
      if check($candidate, min\_sup$) then
        { check if the $candidate$ has enough support}
        $freq\_k\_seq \Leftarrow freq\_k\_seq \cup candidate$;
      end if
    end for
  end for
return $freq\_k\_seq$.
```

Table 5.2: Function:$gen\_freq\_k\_seq(freq\_(k-1)\_seq, freq\_1\_seq)$

```
  for $k = 1$ to $lengthOf(freq\_seq)$ do
    $rule = splitSeq(freq\_seq, k)$;
    {generate a rule with first k actions in $freq\_seq$ as in the predicate set}
    if $confidence(rule) \geq min\_conf$ then
      return $rule$;
    end if
  end for
return $null$;
```

Table 5.3: Function:$gen\_pattern(freq\_seq)$

5.3 Constraint-based Approach

In this section, five special sequential patterns are first presented. We then propose our different approaches for searching these patterns, pinpoint the modifications in the algorithm, and discuss their impacts.

5.3.1 Constraint-based Sequential Patterns

1. **Sequential patterns beginning with specified action(s)** attempt to answer questions such as " If users visited graduate program information, what would they probably visit next in order?" Following our previous definitions, in this case we are searching sequential patterns $\mathcal{X}_s \Rightarrow \mathcal{Y}_s$, where there exists a specified action, a, such that $\mathcal{X}_s$ begins with a.

2. **Sequential patterns ending with specified action(s)** Here, we are looking for knowledge such as "What are the navigation patterns which finally lead learners to participate in the discussions on course content?". This kind of pattern can be formally expressed as $\mathcal{X}_s \Rightarrow \mathcal{Y}_s$, where there exists a specified action, a, such that a is the final action in $\mathcal{Y}_s$.

3. **Sequential patterns containing specified action(s)** are basically very similar to the former two patterns, except that they do not require the specified actions to appear in a particular position. These patterns shed light on understanding rules such as, " What would learners probably visit in order, before and after downloading the demonstration software?". We formally describe these patterns as $\mathcal{X}_s \Rightarrow \mathcal{Y}_s$, where there exists a specified action, a, such that $a \in \mathcal{X}_s \cdot \mathcal{Y}_s$.

4. **Sequential patterns** which introduce two constraints: the **starting action** and **the time length of the sequential pattern**. Such a pattern is mean-

ingful because it can help instructors understand problems such as "After learners took Exam 1, what would they normally do in the next month?"

5. In the search for **the sequential patterns beginning and ending with specified actions**, decision makers have to specify two delimited actions for these patterns. "After learners downloaded Chapter 1 course content, and before they took Exam 1, what activities would most likely happen?" is a typical question that can be answered by this kind of pattern.

5.3.2 Solutions

Basically, two sorts of solutions are presented here. In the first type, modifications are limited in the pattern discovery phase. By contrast, the second type of solution takes the whole web usage mining process into consideration.

In order to search the first three kinds of special sequential patterns, two functions in previously introduced algorithm are modified. The modifications in constraint-based function $gen\_freq\_k\_seq(freq\_(k-1)\_seq, freq\_1\_seq)$ reduce the number of candidates in each of the level-wise searches, and the modifications in function $gen\_pattern(freq\_seq)$ speed up the process of generating the sequential patterns with special requirements.

```
  $freq\_k\_seq \Leftarrow \phi$;
  **for all** $seq \in freq\_(k-1)\_seq$ **do**
    **for all** $action \in freq\_1\_seq$ **do**
      $candidate = seq \cup action$;
      **if** $candidate$ conforms to $PATTERN\_FILTER$ **then**
        **if** check($candidate, min\_sup$) **then**
          { check if the $candidate$ has enough support}
          $freq\_k\_seq \Leftarrow freq\_k\_seq \cup candidate$;
        **end if**
      **end if**
    **end for**
  **end for**
**return** $freq\_k\_seq$.
```

Table 5.4: Constraint-based Function:$gen\_freq\_k\_seq(freq\_(k-1)\_seq, freq\_1\_seq)$

```
  $rule = splitSeq(freq\_seq, PATTERN\_FILTER)$;
  {generate a rule using $PATTERN\_FILTER$}
  **if** $confidence(rule) \geq min\_conf$ **then**
    **return** $rule$;
  **end if**
**return** $null$;
```

Table 5.5: Constraint-based Function:$gen\_pattern(freq\_seq)$

The above solution (Tables 5.4 and 5.5) is limited in the phase of pattern discovery by modifying the sequential pattern analysis algorithm. There is another solution for searching the first three types of sequential patterns :

1. The algorithm shown in Table 5.1 is used to search sequential patterns without any constraint.
2. The features of the special sequential patterns are used to filter out uninteresting patterns.

In addition, with the flexibility to express constraints in the different phases of web usage mining, we can solve additional problems. The last two types of sequential patterns actually impose some special requirements on the transaction identification. As indicated in Chapter 4, we are able to customize transactions with the delimited descriptors: (page : time), (page A : page B). With these two descriptors, we can generate special transactions datasets for searching the last two types of sequential patterns, respectively.

1. In the data preprocessing phase, we customize a number of transactions using corresponding descriptors.
2. The sequential pattern analysis algorithm introduced in the Table 5.1 is then applied to extract sequential patterns.

5.3.3 Discussion

As we can see, interactions throughout the web usage mining process render the ability to extract various types of special patterns. These patterns, not surprisingly, are often very valuable to specific decision problems.

Generally speaking, interactions in the phase of data preprocessing aim at reducing the search space. By doing so, we are able to find interesting patterns which are statistically dominant only in the bounded search spaces. The constraints in the pattern discovery phase help make the process of searching faster because, with them, it is feasible to conduct purposeful searching for interesting patterns. And, with interactions in the phase of pattern analysis, we can highlight the rules of interest.

It is important to note that the constraints in different phases of web usage mining sometimes are transferable, i.e., some constraints can be replaced by other constraints in the subsequent phases of web usage mining.

- In order to find the local significant patterns, we can loosen some thresholds in the phase of pattern discovery, instead of imposing constraints in the phase of data preprocessing. However, by doing this, not only will the time to search for rules dramatically increase, but also the number of patterns will grow exponentially, making it difficult for decision makers to sift through and analyze the patterns.

- We can also push the constraints from the pattern discovery phase to the pattern analysis phase. It usually leads to a considerable increase of time needed to search for patterns of interest.

5.4 Mining on the Fuzzy Boundary Transactions

A fuzzy boundary transaction dataset is supposed to preserve more information than the traditional transaction dataset. Before modifying our algorithm, we investigate the properties of fuzzy boundary transactions in terms of sequential pattern analysis.

5.4.1 Observation

Along with the following observations, three questions are asked concerning sequential pattern analysis. We illustrate these observations using the example shown in Figure 5.1. In this example, we have two fuzzy boundary transactions.

1. There are some redundancies existing in the fuzzy boundary transactions since we duplicate some clicks in their neighbour transactions. In the example, to match the pattern (g,c), there is a sequence ((g,1),(c,1)) in transaction 2, and a sequence ((g,0.6),(c,0.5)) in transaction 1. However, as indicated in the Figure, these two sequences in fact originate from one sequence in the original transaction 2.

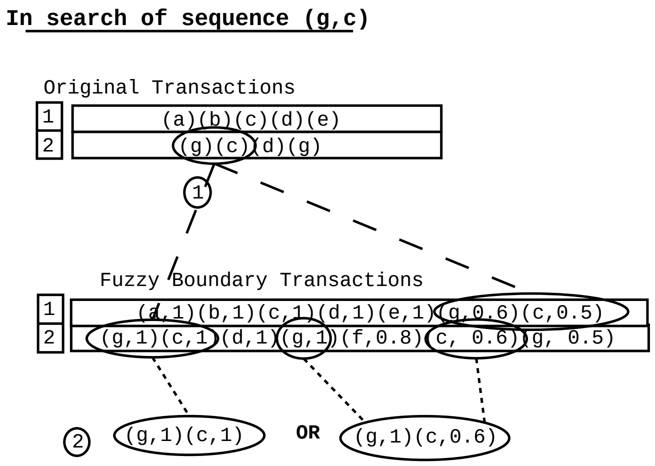
In search of sequence (g,c)
Original Transactions
1
(a)(b)(c)(d)(e)
2
(g)(c)(d)(g)
1
Fuzzy Boundary Transactions
1
(a,1)(b,1)(c,1)(d,1)(e,1)(g,0.6)(c,0.5)
2
(g,1)(c,1)(d,1)(g,1)(f,0.8)(c, 0.6)(g, 0.5)
2
(g,1)(c,1)
OR
(g,1)(c,0.6)

Figure 5.1: Properties of Fuzzy Boundary Transactions

The question is: Should we count the redundant sequences as well? Since they actually correspond to one original sequence, obviously it is not appropriate to count them repeatedly. But, how do we prevent the repeated counting?

2. Sometimes we can find more than one sequence in a transaction to match a pattern. In fuzzy boundary transaction 2, we find two sequences that satisfy the pattern (g,c): ((g,1),(c,1)) and ((g,1),(c,0.6))

   This leads to the question: since these two sequences are different, in that they are associated with different fuzzy membership degrees, which sequences should be chosen to match the pattern?

3. After discovering a sequence in a transaction which matches the pattern, we can find that these clicks are associated with different degrees of membership. In our example, the sequence ((d,1),(c,0.5)) in transaction 1, and the sequence ((d,1),(c,0.6)) in transaction 2 both match the pattern (d,c).

   The question is: how do we evaluate this "matching"?

5.4.2 Modification

To answer the above three questions, we need to change the previous definitions of "**subsequence and supersequence**" and "**contains**". Fortunately, the general framework of the previous algorithms are still applicable. To extend them to handle fuzzy boundary transactions, we need only design new functions to estimate the support and confidence of a sequential pattern in the fuzzy boundary transactions dataset.

In the rest of this section, we adjust the definitions of the two terms, and provide answers to the above three questions using the new definitions.

subsequence and supersequence Given two fuzzy boundary sequences, $fs = \{(click_i, fv_i)|i \in [1,n], \forall i : fv_i \in (0,1]\}$ and $fs^{'} = \{(click^{'}_j, 1)|j \in [1,m]\}$, we say that fs is the supersequence of $fs^{'}$ and $fs^{'}$ is the subsequence of the fs, denoted as $fs^{'} \subseteq fs$, if and only if $\exists$ a series of number $i_1, i_2, \cdots, i_m$ and $1 \leq i_1 < i_2 < \cdots < i_m < n$ satisfies that:

- $click_{i_k} = click^{'}_k$ for $\forall k \in [1,m]$;
- $\exists l \in \{1, \cdots, n\}, fv_l = 1$.

We can prevent the counting of duplicate information by specifying the requirement in the supersequence: among the matching elements, there must be at least one element with the degree of membership equal to 1. So ((g,0.6),(c,0.5)) in transaction 1 is not a legal matching, and thus transaction 1 is not a supersequence of (g,c).

contain In the traditional approach, the concept of "**contain**" is a boolean concept, that is, the value of the fact that fs contains $fs^{'}$ is either 0 or 1. Here, we change this concept into a fuzzy concept.

First, we define the fuzzy value of "**matching**". If a matching between fs and $fs^{'}$ is found with a series of numbers $i_k (k \in [1,m])$, then

$$truth(\text{matching between } fs \text{ and } fs^{'}) = min_{i_k}(fv_{i_k});$$

In our example, the matching value of pattern (d,c) for transaction 1 is equal to $min(1, 0.5)$, which is 0.5. This matching value is equal to 0.6 for transaction 2.

Now we can give a definition of "**contain**". For the two previously defined sequences fs and $fs^{'}$, there exists a number, t to be specific, of matchings, denoted as $\{match_j | j \in [1,t]\}$. Thus, given a statement:

$$X : fs \text{ contains } fs^{'},$$

then

$$truth(X) = \begin{cases} max_{j \in [1,t]}(match_j) & \text{if } fs \text{ is supersequence of } fs^{'}; \\ 0 & \text{if } fs \text{ is NOT supersequence of } fs^{'}. \end{cases}$$

According to this definition, for the two matchings between transaction 2 and pattern (g,c), the truth value of the first matching ((g,1),(c,1)) is 1 and the truth value of the second matching ((g,1),(c,0.6)) is 0.6. Therefore, the truth value of the statement that transaction 2 contains pattern (g,c) is $max(1, 0.6)$, which is 1.

Hence, the previous definitions on the support and confidence should be modified as following:

$$\begin{aligned} support(\mathcal{X}_s \cdot \mathcal{Y}_s) &= \frac{\sum_{s \in \mathcal{S}} truth(s \text{ contains } \mathcal{X}_s \cdot \mathcal{Y}_s)}{|\mathcal{S}|} \\ confidence(\mathcal{X}_s \Rightarrow \mathcal{Y}_s) &= \frac{support(\mathcal{X}_s \cdot \mathcal{Y}_s)}{support(\mathcal{X}_s)} \end{aligned}$$

Replacing these definitions in the previous algorithm, we can perform constraint-based sequential pattern analysis on the fuzzy boundary transaction dataset.

5.5 Summary

Using constraints in the phase of data preprocessing results in reducing the search space, and therefore makes searching for local significant patterns easier. The constraints in the pattern discovery can be pushed into the process of searching, and thereby make the searching faster. Combining interactions through the web usage mining process can help decision makers to find more patterns of interest. In order to extend our algorithm to handle fuzzy boundary transactions, we give new definitions for the support and confidence of a sequential pattern.

Chapter 6

Case Study

In this chapter, we present some experimental results from our research on data preprocessing, and compare the mining results between fuzzy boundary transactions and traditional transactions. Finally, we use some examples to demonstrate the functions of our system.

6.1 Data source

Currently we are testing our method on web logs from two systems: an in-house built system at the Technical University of British Columbia (TechBC), a university that delivers most of its courses on-line, and Virtual-U, a web-based learning environment built in the context of the TeleLearning Canadian Centres of Excellence. The examples we use for illustration in this thesis come from a TechBC web log with records of 100 students' on-line activities in two courses, TECH 142 and TECH 150 from September 14th, 1999 to December 17th, 1999. There are 200,433 entries in this web log file of a size of 109 Megabytes.

One typical entry in the original log file looks as follows:

```
1, 1999-09-14 22:02:13,200, "/TECH150.1/Unit.2/Presentation.1/FAQ/
index.html","-"
```

This entry shows that the user with ID "1" successfully visited at 1999-09-14 22:02:13 the web page `/TECH150.1/Unit.2/Presentation.1/FAQ/index.html`.

6.2 Data Preprocessing

6.2.1 Mapping Web Pages

Since the URL syntax of this web site encodes the structure of the site, when preprocessing the web log we provide a way to generalize the log entries. For example, if a student visits these following web pages successively:

`/TECH142.1/Unit.1/LearningPath.1/ActivitySequence1/External1.html`,

`/TECH142.1/Unit.1/LearningPath.1/ActivitySequence1/favicon.ico`,

`/TECH142.1/Unit.1/LearningPath.1/ActivitySequence1/index.html`,

`/TECH142.1/Unit.1/LearningPath.1/ActivitySequence1/tooltip.htc`,

we can automatically generalize these four clicks into one action, say, "Tech142.1, unit 1, Learning Path 1, and Activity Sequence 1". We might even generalize it at a higher-level such as "Tech142.1, unit 1". These drill-down and roll-up functionalities are provided to instructors to enable them to manipulate the dataset and impose a concept level during the constraint-based mining process.

6.2.2 Extracting the Transactions

Since there are no other meta data available, we provide two options to the decision makers: either customizing the transactions or identifying the transactions using the time-out method.

In the latter case, instructors are asked to input an initial value — if the idle time between two consecutive clicks is shorter than the initial value, instructors believe these two clicks should be in the same transaction with strong confidence. In fact, it is much easier to specify such a value than to determine the cutoff time directly for a targeted web log data. [1] From this value, the algorithm shown in Table 4.4 is used to search for a proper time-out threshold for the web log in hand. Table 6.1 shows how the suggested threshold changes with different initial values. The results are typical of what is expected.

- From Figure 6.1, we can find that the number of intervals in the range 800-960 seconds (13-16 min.) is significantly larger than the number of those around 960-1000 seconds (16-17 min.). We can also discover that the trend of the distribution after 1000 seconds is more smooth than that before it. In this

[1] In practice, we can set the default initial value as 10 minutes for various web log data, since most thresholds widely used in different web sites are larger than 10 minutes.

case, the value of $MeasureFunction(threshold)$ presented in Chapter 4 is close to 1. Therefore, given an initial value larger than 1000 seconds, typically a better value for the threshold can not be found for this web log, so the suggested threshold is set the same as the initial value.

Initial value	Suggested threshold
600 seconds(10 min.)	16 min.
900 seconds(15 min.)	16 min.
1020 seconds(17 min.)	17 min.
1500 seconds(25 min.)	25 min.
2000 seconds(33.3 min.)	33.3 min.

Table 6.1: Suggested Thresholds with Different Initial Values

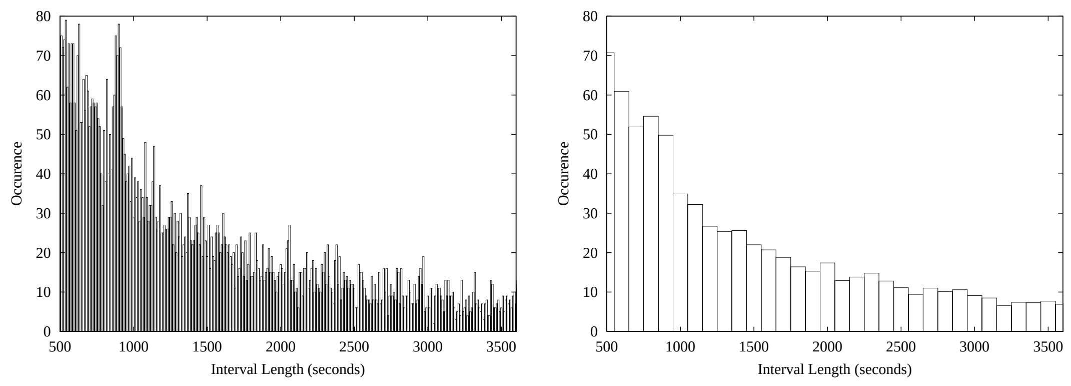

Figure 6.1: Distribution of Interval Length (Range from 10 min. to 60 min.)

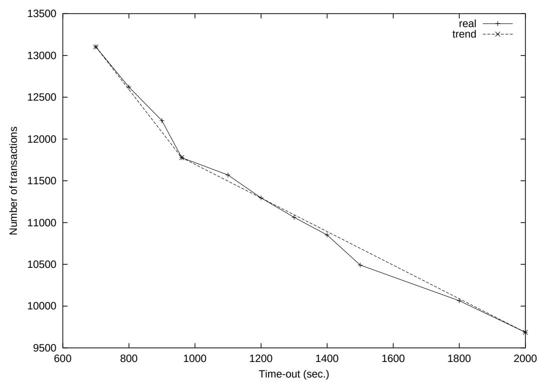

Figure 6.2: Number of Transactions with Different Time-out Thresholds

- In addition, Figure 6.2 shows how the number of the alleged transactions changes with the different time-out thresholds. In this Figure, we can tell that, generally, the number of transactions decreases linearly with the increasing of the time-out threshold. It is noteworthy to find that the point with the time-out threshold of 960 seconds (16 min.) is a turning point in the curve. The rate of decrease before this point is obviously larger than that after the point.

Both phenomena imply that, with this threshold (16 min.), most of inter-transaction intervals would be significantly longer than intra-transaction intervals. The algorithm works as expected. In the following discussion, we use 960 seconds (16 min.) as the time-out threshold to identify transactions unless indicate otherwise.

However, since this method employs one single standard — time-out threshold — to generalize complex usage behaviours, there must be a large number of errors in the transaction identification. Especially in web-based learning environments, reading is a very common activity and may be very time consuming, it is likely that we set cut to break transactions in a place where a learner was spending time reading. We list two abnormal situations which conflict with our expectation and probably indicate a such mistake.

Situation 1: The number of clicks in a transaction is less than three. Because most on-line activities consist of more than two clicks in this environment, we expect that there are at least three clicks happening during one visit.

Situation 2: The length of a transaction is larger than the following interval. For example, if a learner has already spent 4 hours on site learning and is still going to spend some extra time on site, a 20-30 minutes idle time becomes acceptable in a transaction since the length of it becomes fairly small, compared to the length of the transaction.

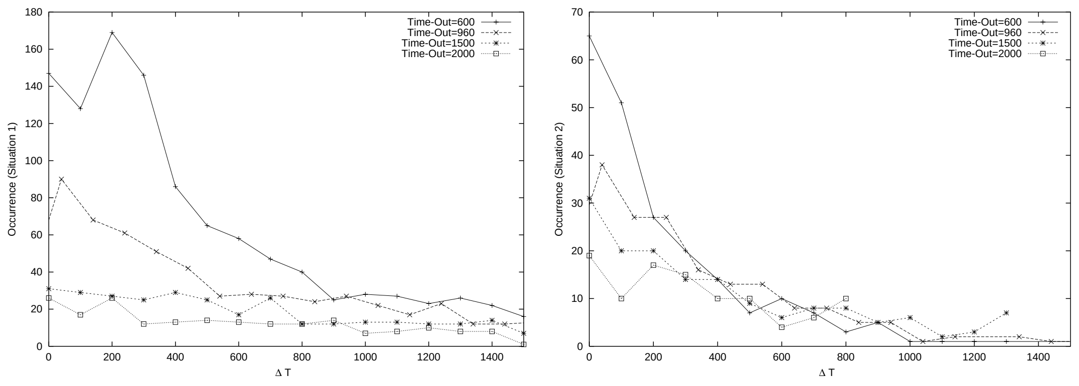

Figure 6.3: The Occurrence of Intervals with Length (Time-Out+ΔT)

Time-out Threshold	Num. of Merged Transactions	Num. of Transactions	Percentage
600	1022	13367	7.65%
960	**742**	**11777**	**6.30%**
1500	524	10491	4.99%
2000	417	9686	4.31%

Table 6.2: Modification Rates with Different Thresholds

Figure 6.3 shows, given a pre-defined threshold, most of the lengths of the inter-transaction intervals in these two situations are in the range of [threshold, threshold+ ΔT] when ΔT is small. Therefore, we merge these transactions with their nearest neighbours. Table 6.2 shows the results of our modifications. In general, we can merge 5% - 8% of the transactions, depending on the length of defined time-out threshold: the larger is the threshold, the fewer transactions do we merger. It makes sense because these two situations correspond to a type of error, with which we separate clicks that should be in one transaction into different transactions. On the other hand, we can not simply specify a large threshold instead, since it often causes another type of error, in which we group the clicks that should belong to different transactions into one transaction. The typical consequence of making this type of error is generating many transactions whose length is very long, as illustrated in Figure 6.4.

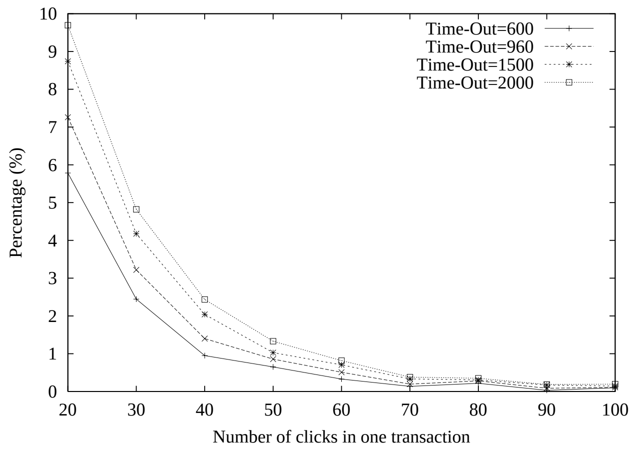

Figure 6.4: Percentages of Long Transactions according to Different Thresholds

6.3 Mining on Fuzzy Boundary Transactions

With the current fuzzy membership function, some of the clicks in click streams are duplicated into more than one transaction. To support our expectation that fuzzy boundary transactions preserve more information than traditional transactions, we perform our experiments using four datasets:

Dataset 1: All the log entries;

Dataset 2: All the log entries concerning TECH142 and TECH150;

Dataset 3: All the log entries concerning TECH142;

Dataset 4: All the log entries concerning TECH150;

In Table 6.3, we compare fuzzy boundary transactions with traditional transactions in the average number of clicks per transaction, using four datasets. As we can see, with the current fuzzy membership function, the number of clicks per transaction increases by average of 3 to 4.

Dataset	aver. clicks per fuzzy boundary transaction	aver. clicks per traditional transaction	Rate of increase
1	13.44	10.45	28.61%
2	12.16	7.94	53.15%
3	11.50	7.22	59.28%
4	12.19	7.96	53.14%

Table 6.3: Comparisons between Fuzzy Boundary Transactions and Traditional Transactions

In addition, we perform two data mining algorithms — association rule mining and sequential pattern analysis — on these four datasets and compare the results with the number of discovered frequent itemsets and frequent sequences.

The results shown in Table 6.4 support our previous expectation that the fuzzy boundary transactions preserve more information than do the traditional transactions. Typically, compared to the 30%-60% increase in the size of transactions, we can get a 70%-200% increase in the number of discovered patterns. This evidence validates the effectiveness of our approach to mining fuzzy boundary transactions.

6.4 Interactive Web Usage Mining

Table 6.5 summarizes the general interactions between the users and our web usage mining systems.

Dataset 1 (support = 30%)			
	number of frequent itemsets/sequences		
	using fuzzy concept	traditional	rate of increase
Association Rule Mining	11	11	0%
Sequential Pattern Analysis	17	10	70%
Dataset 2 (support = 5%)			
	number of frequent itemsets/sequences		
	using fuzzy concept	traditional	rate of increase
Association Rule Mining	25	12	108%
Sequential Pattern Analysis	19	8	137.5%
Dataset 3 (support = 10%)			
	number of frequent itemsets/sequences		
	using fuzzy concept	traditional	rate of increase
Association Rule Mining	20	14	42.86%
Sequential Pattern Analysis	14	8	75%
Dataset 4 (support = 8%)			
	number of frequent itemsets/sequences		
	using fuzzy concept	traditional	rate of increase
Association Rule Mining	26	8	225%
Sequential Pattern Analysis	16	5	220%

Table 6.4: Comparisons of the Mining Results using Fuzzy Boundary Transactions and Traditional Transactions

6.4.1 Bounded Search Space

Users often change their navigation behaviours. Especially in an e-learning web site, the course content changes from time to time, and therefore so does the focus of users' behaviours. In addition, different users are inclined to have different navigation behaviours, and the successful learners probably have more effective learning patterns.

A lot of interesting patterns can therefore be discovered under particular circumstances, such as the particular group of users, a particular period of time, and/or particular web pages. If we investigate these patterns over all the search space, they are usually, not surprisingly, no longer statistically significant and thus become less likely to be discovered.

In this experiment, Dataset 3 (all the log entries concerning TECH142) is used. A subset, called Dataset 3a, is used by contrast as an example of bounded search space. Dataset 3a consists of the entries concerning TECH142 and which happen in the period from October 1st to October 31st. With the same method, and the same fuzzy membership function, we generate 5441 transactions from Dataset 3, and 1833 transactions from Dataset 3a.

Data Preprocessing	
specify filters	users of interest
	period of interest
	web pages of interest
select mapping level	
select method for identifying transaction	by time-out (with/without adjustment)
	by time-duration
	customizing transactions
generate fuzzy boundary transactions (yes or no)	
Pattern Discovery	
select data mining algorithms	Association Rule Mining
	Intra-transaction Sequential Pattern Analysis
	Inter-transaction Sequential Pattern Analysis (***)
specify constraints for Sequential Pattern Analysis	
mining on fuzzy boundary transactions (yes or no)	
Pattern Analysis	
specify interesting patterns	
***: PrefixSpan algorithm is provided by the authors of [PHM+2001]	

Table 6.5: System Interactions Summary

At first, we perform the sequential pattern analysis in both datasets with the same significant level (support=10%, confidence=40%). From Dataset 3, we obtain 2 frequent sequences in total, while with Dataset 3a, 19 frequent sequences are discovered, most of which are concerned with TECH142.1 and TECH142.2. These activities are only typical in October, rather than in all the three months.

Still we can decrease the significant level when mining on Dataset 3 in order to get similar results to what we have obtained from Dataset 3a. For example, similar to the rule discovered within Dataset 3a:

page (216)[1] $\Rightarrow$ *page (212)*[2] *confidence = 73.19%, support = 11.87%*[3]

a rule can be discovered in Dataset 3:

page (216) $\Rightarrow$ *page (212) confidence = 73.28%, support = 5.40%*

However, in order to discover this rule, we have to reduce the threshold of support from the previous 10% to around 5%. As a consequence, 106 sequential patterns will be retrieved from Dataset 3. However, this not only increases the difficulties for instructors to find the desired knowledge, it also increases the time needed to discover these rules. The amount of time needed to discover the rules from Dataset

[1] `/Data/TECH142.2/Unit.1/LearningPath.1/PriorKnowledgeAssessment/index.html`

[2] `/Data/TECH142.2/Unit.1/LearningPath.1/index.html`

[3] This rule means before visiting "TECH142.2 Unit.1 LearningPath.1", 73.19% of learners visited "PriorKnowledgeAssessment" of "TECH142.2 Unit.1 LearningPath.1". This rule is supported by 11.87% samples.

3 with 5% support, is about 40 times greater than the time needed to discover them from Dataset 3 with 10% support, and 20 times greater than the amount of time needed to search rules from Dataset 3a with 10% support. [1]

6.4.2 Bounded Search Strategy

As indicated in the last experiment, there exist some tradeoffs between the desire to discover more patterns and the cost of searching and analyzing these rules. In some cases, if decision makers have clear ideas about the usage of their web sites, they can then focus their search on the most relevant data. In the TECHBC web sites, for instance, the instructor of TECH142 probably knows that the “TECH142.2 Unit.1” was offered on-line in October. So, if the instructor wants to find patterns related to “TECH142.2 Unit.1”, he/she can confine the search on Dataset 3a instead of Dataset 3. However, decision makers do not always have this kind of prior knowledge, which means that they often have to reduce the thresholds of the pattern discovery algorithms in order to obtain more patterns. However, this results in more time required to search all these patterns, and further efforts needed for instructors to filter out the patterns which are not useful to them.

We provide two approaches to alleviate these difficulties. Normally, the decision makers will have their preferences for the patterns, according to the different decision problems. These preferences can be valuable in reducing the cost of searching the desired patterns, and fortunately, most of these preferences can be explicitly expressed by the constraints.

A brute force method is to use these constraints to filter out uninteresting patterns in the pattern analysis phase. However, by using this method, the time to search these patterns is still fairly long. So in the second approach, we use these constraints to refine the search. Based on our previous experiments, we assume that instructors are attempting to find the interesting rules from Dataset 3, with the support threshold as 5%. Without any constraint, it takes about 1858 seconds to finally generate 106 sequential patterns. If the instructors are interested in the rules related to page(212), then this condition, “the discovered patterns should contain page(212)” can be used to improve the efficiency of the search. It will take 41 seconds to generate 2 rules in the latter case. In another instance, the instructors may be interested in knowing “what did the students normally visit after visiting ‘TECH142.1

[1] All the results are obtained with the environment, Linux 6.22, Intel Celeron 533, 128M memory.

Unit.2 LearningPath.1'?". By integrating this constraint in the sequential pattern, we can discover 15 relevant patterns within 147 seconds.

6.4.3 Other

In Chapter 5, we presented a way to integrate interactions in the data preprocessing phase with our sequential pattern analysis algorithm to search special patterns of interest. For example, an instructor is interested in learners' navigation patterns after visiting page(137)[1] and before visiting page(187)[2]. To find such patterns, we first generate a customized transaction dataset in which each transaction begins with a click of page(137) and ends with a click of page(187). About 74 transactions are then obtained, with approximate 166 clicks per transaction in average.

From this dataset, the sequential pattern analysis algorithm generates 877 sequential patterns in total with a significant level (support=70%, confidence=70%).

The following are two typical discovered patterns:

page(139) ⇒ page(142) page(148) page(212) confidence = 88.41% support = 82.43% ;

page(139) ⇒ page(142) page(169) page(174) confidence = 86.96% support = 81.08%.

page(139) :	`/Data/TECH142.1/Unit.4/LearningPath.1/ActivitySequence2/index.html`
page(142) :	`/Data/TECH142.1/Unit.4/LearningPath.1/ActivitySequence3/index.html`
page(148) :	`/Data/TECH142.1/Unit.4/LearningPath.1/ActivitySequence5/index.html`
page(169) :	`/Data/TECH142.1/Unit.5/TechWeek.1/Overview/index.html`
page(174) :	`/Data/TECH142.1/Unit.5/TechWeek.1/index.html`
page(212) :	`/Data/TECH142.2/Unit.1/LearningPath.1/PriorKnowledgeAssessment/`

6.5 Summary

With the experiments with TechBC web log data, we verify that our proposed approaches:

- can search for an appropriate time-out threshold for a web log data, and can modify identified transactions in abnormal situations;
- can generate fuzzy boundary transactions and effectively perform data mining algorithms on these transactions;
- can design various interactions through the web usage mining process.

[1] `/Data/TECH142.1/Unit.4/LearningPath.1/ActivitySequence1/index.html`
[2] `/Data/TECH142.2/Unit.1/LearningPath.1/ActivitySequence1/index.html`

Chapter 7

Conclusions and Future Work

7.1 Conclusions

Performing human interactions in web-based environments is becoming a common activity. The web log data provide a raw trace of users' activities on web sites. Applying data mining techniques to extract useful patterns from the web log data is an interesting topic which has prompted many research efforts. A web usage mining system usually consists of three major steps: data preprocessing, pattern discovery, and pattern analysis.

Our research is motivated by the emerging demands to analyze users' navigation behaviours in web-based environments. The traditional web usage mining systems restrict their utilities with the rigid pre-defined web usage mining process. We present a flexible interactive web usage mining system in this thesis and show its advantages in two major aspects:

- Decision makers are offered the flexibilities to select and combine many techniques to perform data preprocessing to the web log data. In particular, we explain the task of identifying transaction from a more general aspect. A transaction is a series of clicks that are "close" to one another, according to some criteria. With this definition, we have developed several modifications over the existing approaches. A method to automatically adjust pre-defined time-out threshold to the targeted dataset is presented. This approach prevents decision makers becoming involved in too many details concerning web usage mining techniques. Moreover, we propose an effective and novel way to extract fuzzy boundary transactions. Fuzzy boundary transactions provide a more intuitive way to depict the on-line activities and they preserve more information than traditional transactions. In addition, we have modified the typical data mining algorithms, such as association rule mining and sequential pattern analysis algorithm to extend their functionalities to handle the fuzzy boundary transactions.

- Our system allows interactions in all phases of the web usage mining process. It provides decision makers with greater power and flexibility to extract desired knowledge from web usage data. We also investigate the issues related to pushing constraints into different phases of web usage mining. A constraint-based sequential pattern analysis algorithm is presented in this thesis, from which we have made the following conclusions:
  - Using constraints in the phase of data preprocessing results in reducing the search space. It makes searching for local significant patterns easier.
  - The constraints in the pattern discovery can be pushed into the process of searching and make it faster.
  - Combining interactions through the web usage mining process can help decision makers to find more patterns of interest.

7.2 Future Work

Web usage mining is still a very open area and significant research efforts have been made into various issues. In terms of our research, there are many feasible advances that can be made in the predictable future:

- Our research has mainly focused on the phase of data preprocessing, due to its importance in web usage mining. Additional efforts have been made to investigate interactions in pattern discovery and perform different algorithms on the fuzzy boundary transaction dataset. However, our system is in need of a strong pattern analysis model with which the discovered patterns can be presented intuitively to instructors. Also, some machine learning and artificial intelligence features can be added to our system to solve clearly defined decision problems. For example, many current research from these fields can be applied to solve problems such as: knowing a student's learning pattern, how to recommend other students' good experiences to this student?

- At current stage, the fuzzy set theory is only applied to fuzzify the boundaries of transactions. However, we can extend this fuzzy computing to the whole process of generating transactions. Exploring other fuzzy membership functions and investigating their performance would also be an interesting topic.

- Although there has been a graphic interface that helps educators to interact with the web usage mining process, a simple mining and query language is required to make our system more powerful. It is important to simplify interactions to make them easy for instructors to follow. In general, what is required is fewer interactions concerning details of the data mining process, since decision makers in our case are not necessarily knowledge about data mining. Well-defined interactions enable instructors to more easily clarify their decision problems.

- Research on automatically distributing constraints to the whole process of web usage mining is another possible research direction. Ideally, instructors do not need to follow the three phases of web usage mining. In the current decision support model, we separate the interactions into three phases, which requires instructors to understand the process of web usage mining. However, they are usually not concerned much with the process, instead they are more interested in discovered patterns. It would, therefore, be preferable if instructors were able to identify their problems by specifying interactions all at once, and be able to rely on an intelligent model that can distribute all these constraints and specifications effectively and efficiently.

Bibliography

[ASr1994] R. Agrawal, R. Srikant, *Fast algorithms for mining association rules*, In Proc. of the 20th VLDB Conference, pages 487–499, Santiago, Chile, 1994.

[ASr1995] R. Agrawal, R. Srikant, *Mining Sequential Patterns"*, Proc. of the Int'l Conference on Data Engineering (ICDE), Taipei, Taiwan, March 1995.

[ASr1996] R. Agrawal, R. Srikant *Mining Sequential Patterns: Generalizations and Performance Improvements*, appear in Proc. of the Fifth Int'l Conference on Extending Database Technulogy (EDBT), Avignon, France, March 1996.

[BGG+1999] Daniel Boley, Maria Gini, Robert Gross, Eui-Hong (Sam) Han, Kyle Hastings, George Karypis, Vipin Kumar, Bamshad Mobasher, Jerome Moore *Document Categorization and Query Generation on the World Wide Web Using WebACE*, Journal of Artificial Intelligence Review, Vol. 13, No. 5-6, pp. 365-391, 1999.

[BHK1998] J. S. Breese, D. Heckerman, C. Kadie, *Empirical Analysis of Predictive Algorithms for Collaborative Filtering*, Proceedings of 14th Conference on Uncertainty in AI, Madison, WI, July, 1998.

[BLe1998] J. Borges, M. Levene, *Mining Association Rules in Hypertext Databases*, Proceedings of 4th international conference on Knowledge Discoveryand Data Mining, pp 149-153, Aug., 1998.

[BLe1999] J. Borges, M. Levene, *Data Mining of User Navigation Patterns*, Proceedings of 5th ACM SIGKDD internation conference on Knowledge Discoveryand Data Mining, San Diego, U.S.A., Aug., 1999.

[BSp2000] B. Berendt, M. Spiliopoulou *Analysis of navigation behaviour in web sites integrating multiple information systems*, The VLDB Journal(2000) 9, pp56-75

[Cha1999] P. K. Chan, *A non-invasive learning approach to building web user profiles*, Proceedings of 5th ACM SIGKDD internation conference on Knowledge Discoveryand Data Mining, San Diego, U.S.A., Aug., 1999.

[CMS1997] R. Cooley, B. Mobasher, J. Srivastava, *Web Mining: Information and Pattern Discovery on the World Wide Web*, Proceedings of the ninth IEEE international conference on Tools with AI, 1997.

[CMS1999] R. Cooley, B. Mobasher, J. Srivastava, *Data Preparation for Mining World Wide Web Browsing Patterns*, Journal of Knowledge and Information System, Vol.1, No.1,1999.

[CPi1995] L. D. Catledge, J. Pitkow, *Characterizing Browsing Strategies in the World-Wide Web*, Computer Networks and ISDN Systems 27(6): 1065-1073, 1995.

[CPM+1998] E. H. Chi, J. Pitkow, J. Mackinlay, P. Pirolli, R. Gossweiler, and S. Card, *Visualizing the Evolution of Web Ecologies*, In Proceedings of ACM CHI Conference on Human Factors in Computing Systems. pp. 400-407 1998.

[CPY1998] M. -S. Chen, J. S. Park, P. S. Yu, *Efficeint Data Mining for Path Traversal Patterns*, IEEE Transaction on Knowledge and Data Engineering, Vol.10, No.2,pp 209-221, April, 1998.

[CTS1999] R. Cooley, P.-N. Tan, J. Srivastava, *WebSIFT: The Web Site Information Filter System*, Proceedings of the Web Usage Analysis and User Profiling Workshop, August 1999.

[CKL1999] D. W. Cheung, B. Kao, J. Lee, *Discovering User Access Patterns on the World Wide Web* Proceedings of 5th ACM SIGKDD internation conference on Knowledge Discoveryand Data Mining, San Diego, U.S.A., Aug., 1999.

[Ede2001] H. A. Edelstein, *Pan for Gold in the Clickstream*, Informationweek, March 2001, http://www.informationweek.com/828/mining.html

[Etz1996] O. Etzioni, *The World Wide Web: quagmire or gold mine?*, Communications of the ACM, Vol.39, No.11, Nov,1996

[FSS1999] Y. Fu, K. Sandhu, M. -Y. Shih, *Clustering of Web Users Based on Access Patterns*, Proceedings of 5th ACM SIGKDD internation conference on Knowledge Discoveryand Data Mining, San Diego, U.S.A., Aug., 1999.

[FWZ2001] A. Foss, W. Wang, O. R. Zaïane, *A Non-Parametric Approach to Web Log Analysis*, Proc. Web Mining Workshop, in conjunction with the SIAM International Conference on Data Mining, Chicago, IL, USA, April 7, 2001.

[GRS+1999] M. N. Garofalakis, R. Rastogi, S. Seshadri, K. Shim, *Data Mining and the Web: Past, Present and Future*, Proceedings of WIDM99, Kansas City, U.S.A., 1999.

[HGk2000] Daqing He, Ayse Gker, *Detecting Session Boundaries from Web User Logs*, In Proceedingsof of the BCS-IRSG 22nd Annual Colloquium on Information Retrieval Research, 2000.

[HKa2000] J. Han, M. Kamber, *Data Mining: Concepts and Techniques*, Chapter1, pp 5-10 Morgan Kaufmann Publishers, 2000.

[HZF1994] J. Han, O. R. Zaïane, Y. Fu, *Resource and Knowledge Discovery in Global Information Systems: A Multiple Layered Database Approach*, Technical Report CMPT94-10, Simon Fraser University, November 1994.

[JJo1999] A. Joshi and K. Joshi, *On mining web access logs*, Technical report, CSEE Department, UMBC, 1999. `http://www.csee.umbc.edu/~ajoshi/web-mine/`

[JJY+1999] K. P. Joshi, A. Joshi, Y. Yesha, R. Krishnapuram, *Warehousing and Mining the Web Logs*, Proceedings of WIDM99, Kansas City, U.S.A., 1999.

[JFM1997] J. Joachims, D. Freitag, T. Mitchell, *WebWatcher: A Tour Guide for the World Wide Web*, Proceedings of IJCAI97, August 1997.

[Kle1997] J. Kleinberg, *Authoritative sources in a hyperlinked environment. Proc. 9th ACM-SIAM Symposium on Discrete Algorithms*, IBM Research Report RJ 10076, May 1997.

[KKP+1996] E. Kranakis, D. Krizanc, A. Pelc, D. Peleg, *The Complexity of Data Mining on the Web*, Proceedings of the fifteenth annual ACM symposium on Principles of distributed computing , 1996, Page 153.

[KWW1998] C. M. Kuok, A. Fu, and M. H. Wong. *Mining fuzzy association rules in databases*, SIGMOD Record, 27(1):41–46, 1998.

[LAR2000] W. Lin, S. A. Alvarez, C. Ruiz *Collaborative Recommendation via Adaptive Association Rule Mining*, WEBKDD2000, August 20, 2000, Boston, MA, USA.

[LBO1999] B. Lan, S. Bressan, B. C. Ooi, *Making Web Servers Pushier*, Proceedings of 5th ACM SIGKDD internation conference on Knowledge Discoveryand Data Mining, San Diego, U.S.A., Aug., 1999.

[Men1999] J. Mena, *Web Mining: Creating, Enhancing, Analyzing and Acting on Web Data*, Digital Press "Data Mining Your Website", July, 1999.

[MJH+1996] B. Mobasher, N. Jain, E. Han and J. Srivastava, *WEBMINER: Pattern Discovery from World Wide Web Transactions* Technical Report TR96-050, Department of Computer Science, University of Minnesota, February, 1996.

[MCS1999a] B. Mobasher, R. Cooley, J. Srivastava, *Automatic Personalization Through Web Usage Mining*, Technical Report TR99-010, Department of Computer Science,Depaul University, 1999.

[MCS1999b] B. Mobasher,R. Cooley and J. Srivastava, *Creating Adaptive Web Sites Through Usage-Based Clustering of URLs*, Proceedings of the 1999 IEEE Knowledge and Data Engineering Exchange Workshop (KDEX'99), November 1999.

[MDo1999] N. Minar and J. Donath, *Visualizing the crowds at a web site*, In Proceedings of CHI 1999.

[MDu1999] D. Murray, K. Durrell, *Inferring Demographic Attributes of Anonymous Internet Users* Proceedings of 5th ACM SIGKDD internation conference on Knowledge Discoveryand Data Mining, San Diego, U.S.A., Aug., 1999.

[MDL+2000] B. Mobasher, H. Dai, T. Luo, M. Nakagawa, Y. Sun, J. Wiltshire *Discovery of Aggregate Usage Profiles for Web Personalization* in Proceedings of the Web Mining for E-Commerce Workshop (WebKDD'2000), held in conjunction with the ACM-SIGKDD Conference on Knowledge Discovery in Databases (KDD'2000), August 2000, Boston.

[MPC1999] F. Masseglia, P. Poncelet, and R. Cicchetti, *An Efficient Algorithm for Web Usage Mining*, Networking and Information Systems Journal (NIS), Vol. 2, No. 5-6, pp. 571-603, 1999.

[MPT2000] F. Masseglia, P. Poncelet, and M. Teisseire, *Web Usage Mining: How to Efficiently Manage New transactions and New Customers*, Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery (PKDD'00), Lyon, France, September 2000.

[Net1999] Netscape support documents *PERSISTENT CLIENT STATE HTTP COOKIES Preliminary Specification - Use with caution* `http://home.netscape.com/newsref/std/cookie_spec.html`,1999.

[Net2000] NetTracker whitepapters *Analyzing Web Site Traffic* `http://www.sane.com/products/NetTracker/whitepapers.html`, 2000

[NFJ+1999] O. Nasraoui, H. Frigui, A. Joshi, R. Krishnapuram, *Mining Web Access logs Using Relational Competitive Fuzzy Clustering* Proceedings of 8th international Fuzzy System Association World Congress, Aug., 1999.

[Pit1997] J. Pitkow, *In Search of Reliable Usage Data on the WWW*, Proceedings of 6th international WWW conference, pp451-463, Santa Clara, U.S.A., 1997.

[Pit1998] J. Pitkow, *Summary of WWW characterizations*, Computer Networks And ISDN Systems 30 (1998), no. 17, 551-558.

[PEt1999] M. Perkowitz, O. Etzioni, *Adaptive Web Sites: Conceptual Cluster Mining*, Proceedings of IJCAI99, 1999.

[PEt1998] M. Perkowitz, O. Etzioni, *Adaptive Web Sites: Automatically Synthesizing Web Pages*, Proceedings of 15th national conference on AI, 1998.

[PHM+2000] J. Pei, J. Han, B. Mortazavi-Asl, and H. Zhu, *Mining Access Patterns efficiently from Web logs*, Proc. 2000 Pacific-Asia Conf. on Knowledge Discovery and Data Mining (PAKDD'00), Kyoto, Japan, April 2000.

[PHM+2001] J. Pei, J. Han, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal, and M.-C. Hsu, *PrefixSpan: Mining Sequential Patterns Efficiently by Prefix-Projected Pattern Growth*, Proc. 2001 Int. Conf. on Data Engineering (ICDE'01), Heidelberg, Germany, April 2001.

[Spi1999] M. Spiliopoulou, *Managing Interesting Rules in Sequence Mining*, Proceedings of the 3rd European Conf. on Principles and Practice of Knowledge Discovery in Databases PKDD'99, number 1704 in LNAI, pp 554-560, Prague, Czech Republic, Sept. 1999.

[Sur2000] IBM SurfAid whitepapers, *Measuring Web Site Visits* `http://surfaid.dfw.ibm.com/web/whitepapers.html`, August 14, 2000

[SCD+2000] J. Srivastava, R. Cooley, M. Deshpande, P.-N. Tan, *Web Usage Mining: Discovery and Applications of Usage Patterns from Web Data*, To appear in SIGKDD Explorations, Vol. 1, Issue 2, 2000.

[SFW1999] M. Spiliopoulou, L. C. Faulstich, K. Winkler, *A Data Miner analyzing the Navigational Behaviour of Web Users*, Proceedings of workshop on Machine Learning in User Modelling of the ACAI'99, Creta, Greece, July, 1999.

[SPF1999] M. Spiliopoulou, C. Pohle, L. C. Faulstich, *Improving the Effectiveness of a Web Site with Web Usage Mining*, Proceedings of 5th ACM SIGKDD internation conference on Knowledge Discoveryand Data Mining, San Diego, U.S.A., Aug., 1999.

[SRC1997] B. Sergey, M. Rajeev, S. Craig *Beyond Market Basket: Generalizing Association Rules to Correlations*, SIGMOD'97, pp. 265-276, Tucson, Arizona, 1997.

[UFo1998] L. H. Ungar, D. P. Foster, *Clustering Methods for Collaborative Filtering*, Proceedings of the Workshop on Recommendation Systems. AAAI Press, Menlo Park California 1998.

[W3C1999] *Web Characterization Terminology and Definition Sheet* W3C Working Draft 24-May-1999
http://www.w3.org/1999/05/WCA-terms/

[Was1999] A. M. Wasfi, *Collecting user access patterns for building user profiles and collaborative filtering*, Proceedings of the 1999 international conference on Intelligent user interfaces, Redondo Beach, U.S.A., Jan, 1999.

[Web1999] WebTrends sample reports, `http://www.webtrends.com/reports/reports.asp` 1999

[WDH+1995] K. Wittenburg, D. Das, W. Hill, L. Stead, *Group Asynchronous Browsing on the World Wide Web* Proceedings of the 4th International World Wide Web Conference, Boston, 1995.

[YJM1996] T. W. Yan, M. Jacobsen, H. G. Molina, U. Dayal, *From User Access Patterns to Dynamic Hypertext Linking*, Proceedings of 5th international World Wide Web Conference, May 1996.

[Zad1965] L. A. Zadeh, *Fuzzy sets*, Information and Control 8: 338-353, 1965.

[Zai1998] O. R. Zaïane, *From Resource Discovery to Knowledge Discovery on the Internet*, Technical Report TR 1998-13, Simon Fraser University, August, 1998.

[ZXH1998] O. R. Zaïane, M. Xin, J. Han, *Discovering Web Access Patterns and Trends by Applying OLAP and Data Mining Technology on Web Logs*, Proceedings of ADL98, Santa Barbara, April 1998.

[ZLu2001] O. R. Zaïane, J. Luo, *Towards evaluating learners' behaviour in a web-based distance learning environment*, in Proc. of IEEE International Conference on Advanced Learning Technologies (ICALT01), Madison, WI, August 6-8, 2001.