System for Knowledge Mining in Data from Interactions between User and Application

Ilona Bluemke and Agnieszka Orlewicz

Abstract. The problem of knowledge extraction from the data left by web users during their interactions is a very attractive research task. The extracted knowledge can be used for different goals such as service personalization, site structure simplification, web server performance improvement or even for studying the human behavior. The objective of this paper is to present a system, called ELM (Event Logger Manager), able to register and analyze data from different applications. The registered data can be specified in an experiment. Currently ELM system provides several knowledge mining algorithms, i.e., apriori, ID3, C4.5 but easily other mining algorithms can be added.

Keywords: data mining, service personalization, knowledge discovery algorithms.

1 Introduction

The time spent by people in front of computers still increases so the problem of knowledge extraction from the enormous amount of data left by web users during their interactions is a research task that has increasingly gained attention in the last years. The analysis of such data can be used to understand users preferences and behaviors in a process commonly referred to as Web Usage Mining (WUM). The extracted knowledge can be used for different goals such as service personalization, site structure simplification, web server performance improvement or even for studying the human behavior. In the past, several WUM systems have been made, some of them are presented in Sect. 2. The objective of this paper is to present a WUM system, called *ELM* (Event Logger Manager), designed and implemented at the Department of Electronics and Information Systems Warsaw University of Technology. Our

Ilona Bluemke
Institute of Computer Science, Warsaw University of Technology,
Warsaw, Poland
e-mail: I.Bluemke@ii.pw.edu.pl

K.A. Cyran et al. (Eds.): Man-Machine Interactions, AISC 59, pp. 103–110.
springerlink.com

system significantly differs from existing WUM systems. ELM is flexible and easy to integrate with any Java application. We use aspect modification, as proposed in [2], to add code responsible for registering required data. ELM is able to collect and store data from users interactions from many applications in one data base, and analyze them using knowledge discovery algorithms. ELM system contains several knowledge mining algorithms, i.e., apriori, ID3 and C4.5 taken from weka library [13], but without any difficulty other algorithms can be added. In our system a user can also decide on the kind of data aquisited and filtered. In Sect. 3 the architecture of ELM and its main modules are presented. Conclusions are given in Sect. 4.

2 Web Usage Mining Systems

The information placed in Internet is still increasing. During navigation web users also leave many records of their activity. This enormous amount of data can be a useful source of knowledge but sophisticated processes are need for the analysis of these data. Data mining algorithms can be applied to extract, understood and use knowledge from these data and all these activities are called web mining. Depending on the source of input data web mining can be divided into three types:

1. contents of internet documents are analyzed in Web Content Mining,
2. structure of internet portals are analyzed in Web Structure Mining,
3. the analysis of data left by users can be used to understand users preferences and behavior in a process commonly referred to as Web Usage Mining (WUM).

The web usage mining process as described in [11] consists of following phases:

- data acquisition,
- data preprocessing,
- pattern discovery,
- pattern analysis.

Often the results of pattern analysis are feedback to pattern discovery activity. Effective data acquisition phase is crucial for web usage mining. Data from users interactions with internet application can be stored on server side, proxy servers, client-side. Data can be stored in browser caches or in cookies at client level, and in access log files at server or proxy level. The analysis of such data can be used to understand users preferences and behavior in a Web Usage Mining (WUM) [7, 15]. The knowledge extracted can be used for different goals such as service personalization, site structure simplification, and web server performance improvement, e.g., [8, 10].

There are different WUM systems: small systems performing fixed number of analysis and there are also systems dedicated to large internet services. In the past, several WUM projects have been proposed, e.g., [1, 4, 5, 6, 9, 14]. In Analog system [14] users activity is recorded in server log files and processed to form clusters of user sessions. The online component builds active user sessions which are then classified into one of the clusters found by the offline component. The classification allows to identify pages related to the ones in the active session and to return the requested page with a list of related documents. Analog was one of the

first project of WUM. The geometrical approach used for clustering is affected by several limitations, related to scalability and to the effectiveness of the results found. Nevertheless, the architectural solution introduced was maintained in several other more recent projects. Web Usage Mining (WUM) system, called SUGGEST [1], was designed to efficiently integrate the WUM process with the ordinary web server functionalities. It can provide useful information to make easier the web user navigation and to optimize the web server performance.

Many Web usage mining systems are collaborating with an internet portal or service. The goal is to refine the service, modify its structure or make it more efficient. In this group are systems SUGGEST [1], WUM [7], WebMe [9], OLAM [6]. Data are extracted from www server's logs. So far the only system we were able to find using data from client side is Archcollect [5]. This system is registering data by modified explorer. Archcollect is not able to perform complex analysis. We have not found a system registering data on both client and server sides.

3 ELM Web Usage Mining System

In *Event Logger Manager* (ELM) system all web usage mining processes (mentioned in Sect. 2) are performed. ELM is responsible for data storage and also data analysis by some knowledge discovery algorithms. System is able to store data in local or remote data base. ELM user is able to define what events should be stored and when. Some basic types of events are predefined but user can also specify its own logical events. In ELM system data are preprocessed, data mining algorithms can be executed and its results may be observed. Without any difficulties new algorithms can be added.

3.1 ELM Architecture

In Fig. 1 the main parts of ELM system are presented. Data acquisition is performed by EventLogger and analysis by EventManager. Both modules are using eventDB, relational data base, containing stored events. EventLogger API is written to facilitate

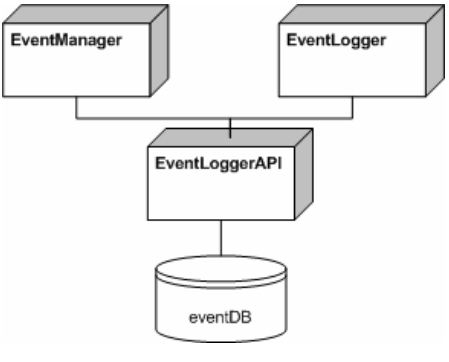


Fig. 1 ELM architecture

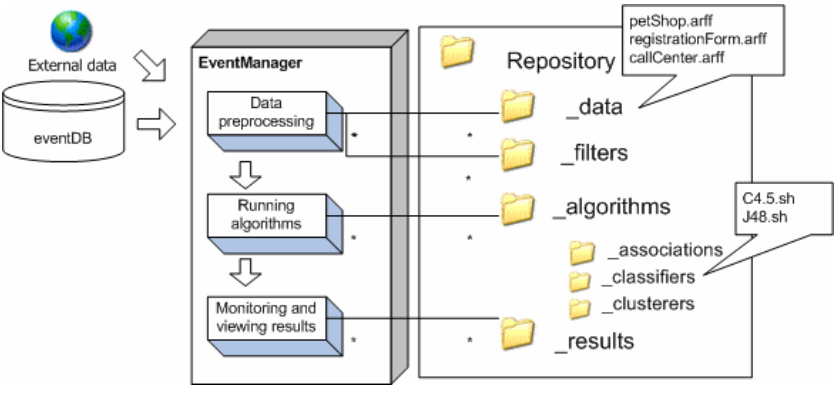


Fig. 2 ELM repository

the access to data base for data registering and data analyzing modules. In this module logical types of events may be defined. For defined events also the parameters may be created. EventLogger API writes or reads events with parameters, in accessing the data base only JDBC (Java DataBase Connectivity) is used. JDBC provides methods for quering and updating data in a relational database.

EventLogger is able to register events handled at server side. We use aspect modification proposed in [2]. An aspect, written in AspectJ [12] is responsible for storing appropriate information describing the event. This aspect can be woven to the code of Java application, even if only compiled code is available.

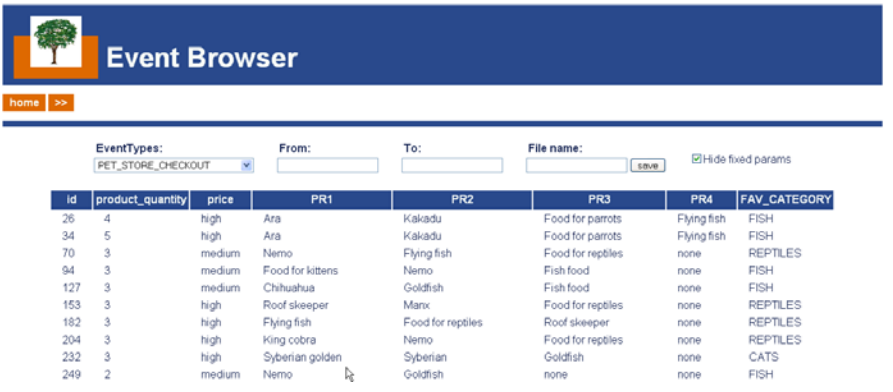
3.2 ELM Repository

ELM system is a tool which can be used by internet application owners and expert analyzing users behavior. Data registered in different applications are kept in eventDB (Fig. 2). Application EventManager reads the registered data from eventDB, presents and transforms them, runs mining algorithms and presents the results. In Fig. 2 the ELM repository is shown. Directory `_data` contains input data for mining algorithms, extracted from eventDB. These data are in `ARFF` (Attribute Relation File Format) [15] format. In directory `_filters` there are scripts processing data, e.g., merging data from several files. Directory `_algorithms` contains scripts running mining algorithms. Currently following algorithms: apriori, ID3, and C4.5 from weka library [13] are available. In this directory scripts running others algorithms can also be inserted. The implementation of newly added algorithm should work on input data in `ARFF` format. More implementation details are given in [3].

3.3 EventManager

EventManager presents events registered in data base, is able to browse and process algorithm's input data, runs mining algorithms and also presents its results. The

Fig. 3 EventManager initial screen



id	product_quantity	price	PR1	PR2	PR3	PR4	FAV_CATEGORY
26	4	high	Ara	Kakadu	Food for parrots	Flying fish	FISH
34	5	high	Ara	Kakadu	Food for parrots	Flying fish	FISH
70	3	medium	Nemo	Flying fish	Food for reptiles	none	REPTILES
94	3	medium	Food for kittens	Nemo	Fish food	none	FISH
127	3	medium	Chihuahua	Goldfish	Fish food	none	FISH
153	3	high	Roof skeeper	Manx	Food for reptiles	none	REPTILES
182	3	high	Flying fish	Food for reptiles	Roof skeeper	none	REPTILES
204	3	high	King cobra	Nemo	Food for reptiles	none	REPTILES
232	3	high	Syberian golden	Syberian	Goldfish	none	CATS
249	2	medium	Nemo	Goldfish	none	none	FISH

Fig. 4 EventBrowser screen

initial screen of EventManager is shown in Fig. 3. From this screen following modules can be called:

- EventBrowser (screen shown in Fig. 4),
- ArffBrowser (screen shown in Fig. 5),
- AlgorithmRunner (screen shown in Fig. 6),
- AlgorithmMonitor (screen shown in Fig. 7).

The web mining process requires successive calls of all four modules.

EventBrowser presents events defined in EventLogger module and stored in data base eventDB (Fig. 1). User may choose interesting type of event from a pull down list and events only of this type are shown on a screen (Fig. 4). In columns the parameters describing events are shown. Common for all events parameters are e.g.: identifier, time, user name and session identifier. In EventLogger module user may define specific for an event parameters, these parameters also are displayed. In EventBrowser user is able to choose events from determined time period and observe only some events parameters. The selected from data base events can be saved in `_data` directory (Fig. 2) as `ARFF` file for future analysis.

ArffBrowser presents `ARFF` files saved in `_data` directory (Fig. 2). From a pull down list an `ARFF` file can be chosen. This file is displayed in a tabular form. In Fig. 5 an example of a file saved by EventBrowser is seen. Usually, files saved by EventBrowser are inadequate for a mining algorithm. ArffBrowser is also responsible for the `ARFF` file adaptation. The typical adaptations are e.g. removing

Fig. 5 ArffBrowser screen

Fig. 6 AlgorithmRunner screen

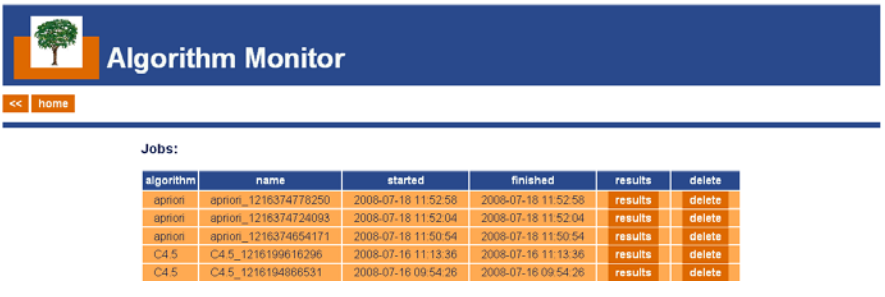


Fig. 7 AlgorithmMonitor screen

parameters, grouping, merging data from many rows into one row. The modified file can be stored in `_data` directory as a new file or replace the old one. From this module filters, performing operations on files can also be started.

The mining algorithms are started by AlgorithmRunner module. From a pull down list (Fig. 6) one of following algorithm's types may be selected: classification, clustering, discovering sequential patterns and associations. After the selection

of algorithm type a list of algorithms available in this category is presented e.g: for classification algorithms currently ID3 and C4.5 [15] can be chosen. The implementation of algorithms are taken from weka library [13]. The algorithms are kept as scripts in `_algorithms` directory (Fig. 2). The script running algorithm is also displayed in a text window. This window can be edited by a user, so some arguments can be changed. User should also select the `ARFF` file containing input data for the algorithm. After pressing the `View` button (Fig. 6), AlgorithmRunner checks, if the selected file contains data appropriate for the chosen type of algorithm, e.g. for classification algorithm class attribute is necessary. The `Run` button starts the execution of mining algorithm.

AlgorithmMonitor module monitors started algorithms and manages `_results` directory in ELM repository (Fig. 2). For each started algorithm appropriate subdirectory is created. The name of this directory is composed of the name of algorithm and date of execution. In this subdirectory the results of algorithm are stored in `results.txt` file. On the screen of AlgorithmMonitor module (Fig. 7) a list of started algorithms is presented, still running algorithms have different color than the finished ones. User may read the results or delete them.

4 Conclusions

Knowledge about users and understanding user needs is essential for may web applications. Registration of user interactions with internet application can be the basis for different analysis. Using algorithms for knowledge discovery it is possible to find many interesting trends, obtain pattern of user behavior, better understand user needs. We present a general system for web usage mining and business intelligence reporting. Our system is flexible and easy to integrate with web applications. To integrate ELM with any web application only an aspect should be written and woven. ELM is able to collect and store data from users interactions from many applications in one data base, and analyze them using knowledge discovery algorithms. A user of our system may add own algorithms or tune implemented ones, as well as decide what information will be stored, filtered and analyzed.

References

1. Baraglia, R., Palmerini, P.: Suggest: A web usage mining system. In: Proceedings of the International Conference on Information Technology: Coding and Computing, pp. 282–287 (2002)
2. Bluemke, I., Billewicz, K.: Aspects in the maintenance of compiled programs. In: Proceedings of the 3rd International Conference on Dependability of Computer Systems, pp. 253–260 (2008)
3. Bluemke, I., Chabrowska, A.: The design of a system for knowledge discovery from user interactions. Zeszyty Naukowe Wydziału ETI Politechniki Gdańskiej 6, 287–292 (2008) (in Polish)

4. Botia, J.A., Hernansaez, J.M., Gomez-Skarmeta, A.: METALA: a distributed system for web usage mining. In: Mira, J., Álvarez, J.R. (eds.) IWANN 2003. LNCS, vol. 2687, pp. 703–710. Springer, Heidelberg (2003)
5. de Castro Lima, J., et al.: Archcollect front-end: A web usage data mining knowledge acquisition mechanism focused on static or dynamic contenting applications. In: Proceedings of the International Conference on Enterprise Information Systems, vol. 4, pp. 258–262 (2004)
6. Cercone, X.H.: An OLAM framework for web usage mining and business intelligence reporting. In: Proceedings of the IEEE International Conference on Fuzzy Systems, vol. 2, pp. 950–955 (2002)
7. Chen, J., Liu, W.: Research for web usage mining model. In: Proceedings of the International Conference on Computational Intelligence for Modelling, Control and Automation, p. 8 (2006)
8. Clark, L., et al.: Combining ethnographic and clickstream data to identify user web browsing strategies. Information Research 11(2), 249 (2006)
9. Lu, M., Pang, S., Wang, Y., Zhou, L.: WebME—web mining environment. In: Proceedings of the International Conference on Systems, Man and Cybernetics, vol. 7 (2002)
10. Nasraoui, O.: World wide web personalization. In: Wang, J. (ed.) Encyclopedia of Data Mining and Data Warehousing. Idea Group (2005)
11. Srivastava, J., et al.: Web usage mining: Discovery and applications of usage patterns from web data. SIGKDD Explorations 1(2), 12–23 (2000)
12. The Eclipse Foundation: AspectJ page, `http://www.eclipse.org/aspectj/`
13. The University of Waikato: Weka home page, `http://www.cs.waikato.ac.nz/ml/weka`
14. Turner, S.: Analog page, `http://www.analog.cx`
15. Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools and techniques, 2nd edn. Morgan Kaufmann, San Francisco (2005)