# Selecting the Best Tailored Algorithm for Personalizing a Web Site

Federico Botella[1], Enrique Lazcorreta[1], Antonio Fernandez-Caballero[2], Pascual González[2], José A. Gallud[2] and Alejandro Bia[1]

[1] Operations Research Center University Institute and Department of Statistics, Mathematics and Computer Science, University Miguel Hernández of Elche
03202 Elche, Spain
{federico, enrique, abia}@umh.es
[2]Computer Science Research Institute of Albacete and Department of Computer Science Systems, University of Castilla-La Mancha
02071 Albacete, Spain
{caballer, pgonzalez, jgallud}@dsi.uclm.es

**Abstract.** Automatic personalization for dynamic web systems has been carried out by several methods and techniques, like data mining or Web usage mining. The Apriori algorithm is one of the most used for web personalization. We have designed a web recommender system based on this algorithm to select the best user-tailored links. We needed an efficient algorithm to preserve updated our system in real time. We tested several implementations of Apriori algorithm but none of them looked to run properly with our data. We tested these implementations with different access log files of public domain (BMS-WebView.dat, BMS-POS) and with our log data. We decided to develop a new implementation that better adapts to our data.

**Keywords:** personalization, recommender systems, data mining, apriori algorithm.

## 1 Introduction

Automatic personalization for dynamic web systems has been carried out by several methods and techniques, like data mining or Web usage mining [5]. There are some techniques for web personalization that relies on the interaction with humans to collect personal information about users of a website, but this lead to problems with the quality of gathered data due to subjectiveness. Moreover, data about users is changing constantly as user preferences changes in short time. In some cases user preferences are automatically learned form web usage data by using data mining techniques [8].

Web Mining is an area of Data Mining oriented to the discovery of knowledge into web data [7]. There are three main areas inside Web Mining related with the structure of the Web, the contents of the Web and the usage of the Web, that are called Web Structure Mining, Web Contents Mining and Web Usage Mining, respectively. The

last one is the type of Web mining that let us to study the behavior of users in a website and to personalize or to adapt a website to the preferences of users or to offer them some recommendations.

In most current researches there is not easy to find a formal procedure to find the preferences of user or the associations between products or services of a web by means of web usage mining. Given a set of transactions where each transaction is a set of items (itemset), an association rule implies the form $X \rightarrow Y$, where X and Y are itemsets; X and Y are called the body and the head, respectively. The support for the association rule $X \rightarrow Y$ is the percentage of transactions that contain both itemset X and Y among all transactions. The confidence for the rule $X \rightarrow Y$ is the percentage of transactions that contain itemset Y among transaction that contain itemset X. The support represents the usefulness of the discovered rule and the confidence represents certainty of the rule. Association rule mining is the discovery of all association rules that are above a user-specified minimum support minsup and minimum confidence minconf. Apriori algorithm is one of the prevalent techniques used to find association rules [1], [2] Apriori operates in two phases. In the first phase, all itemsets with minimum support (frequent itemsets) are generated. This phase utilizes the downward closure property of support. In other words, if an itemset of size k is a frequent itemset, then all the itemsets below (k−1) size must also be frequent itemsets. Using this property, candidate itemsets of size k are generated from the set of frequent itemsets of size (k−1) by imposing the constraint that all subsets of size (k − 1) of any candidate itemset must be present in the set of frequent itemsets of size (k − 1). The second phase of the algorithm generates rules from the set of all frequent itemsets.

## 2 Selection the Algorithm for Web Personalization

We have designed a web recommender system based on the Apriori algorithm to select the best user-tailored links according to his or her preferences. First of all, we extract the user sessions of the log file of our web server, so we obtain data about the users that acceded to our web site, the pages they requested, the sequence of pages they visited and the time of each visit. To facilitate the extraction of user sessions, the logon/logoff activity was registered in a database of our web platform. Next, we apply our modified Apriori algorithm to discover the association rules. Finally we run again the algorithm to discover which rules are interrelated. Our target is to classify our users not by which pages they visit but how they visit it. We will run the algorithm again each time one user just finishes his session, so we will maintain updated our system in order to give an updated next suggestion.

Thus we needed an efficient algorithm to preserve updated our model in real time. We tested several implementations of Apriori algorithm as we have selected this algorithm for our recommender system. The implementations of several authors like Bodon [3], Goethals [6] and Borglet [4] were selected and we perform a preliminary test to select one of them for integrating in our server: an Intel Pentium 4 2.53 GHz processor (family 15, model 2) with 512 KB L2 cache and 1,5 GB RAM. The system runs over Windows 2003 Server SP2 for 32-bit. We selected different datasets for testing these implementations of the Apriori algorithm from the FIMI'03 website (at

http://fimi.cs.helsinki.fi/data): the dataset BMS-POS (35,3MB with 3,360,020 transactions) and the dataset BMS-View1 (1,97MB with 149,369 transactions). We also use a dataset of transactions constructed with the sessions extracted from the weblogs of one server of our Department (5,37MB with 585,149 transactions).

The implementations of Bodon and Goethals have similar conducts for all datasets, whilst the implementation of Borgelt presents the better execution times of the three (see Fig. 1, Fig. 2 and Fig. 3).

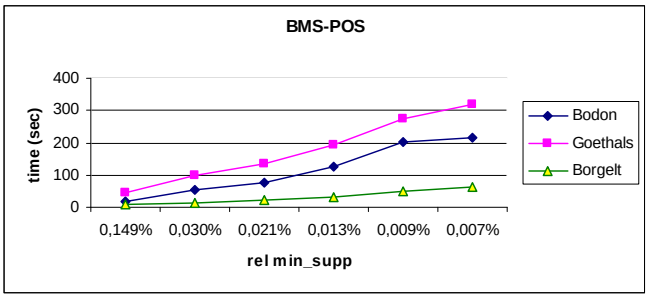


Fig. 1 Execution times results for dataset BMS-POS

The output file produced by the execution of these implementations varies form 18 KB to 58 MB. The largest output files were of size 58MB (BMS-View1, for implementation of Borgelt, minsup = 23%, having 1,177,608 frequent sets with 6 frequent 16-itemsets), 47 MB (OurData, for implementation of Borgelt, minsup = 0,51% having 898,822 frequent sets with 3 frequent 16-itemsets) and 13 MB (BMS-POS, for implementation of Borgelt, minsup = 0,007% having 626,779 frequent sets with 114 frequent 10-itemsets).

We have to note that the 1,177,608 sets for the BMS-View1 dataset with minsup = 23% require 19 seconds to end with the implementation of Borgelt, whereas the implementations of Bodon and Goethals require 67 and 64 seconds to end respectively. On the other hand, the 626,779 sets for the BMS-POS dataset with minsup = 0,007% require 62 seconds to end with for the implementation of Borgelt, whereas the implementations of Bodon and Goethals require 215 and 320 seconds to end respectively. Finally, the 898,822 sets for the OurData dataset with minsup = 0,51% require 270 seconds to end with for the implementation of Borgelt, whereas the implementations of Bodon requires 457 seconds to end and it seems that the implementation of Goethals has some problem to process our dataset as it needs around 700 sec to finish.

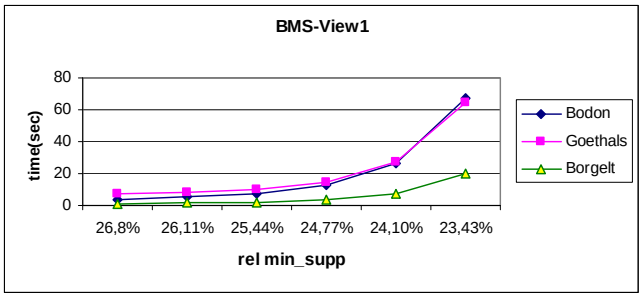


Fig. 2 Execution times results for dataset BMS-View1

The better times has always been achieved with the implementation of Borgelt, whereas the implementations of Bodon and Goethals have similar runtimes for the datasets of BMS-View1, but with the dataset BMS-POS the implementation of Bodon has nearly the half runtime than the implementation of Goethals.

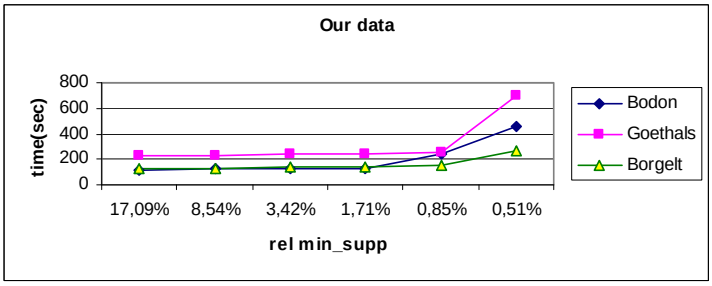


Fig. 3 Execution times results for our dataset

The results of these implementations with our dataset have not been all satisfactory we expected, as all algorithms are over the 4 minutes and Goethals even overcomes the 11 minutes. So we decided to develop our particular implementation of the Apriori algorithm to try to reduce these results and to offer suggestions to users of our website in real time.

## 3 Conclusions

In this work we have presented the results obtained for processing particular datasets using different implementations of the Apriori algorithm. As we can have seen depending on the type of the dataset and the selection of the minsup value we can get important differences in the runtime values for each implementation. Depending on the nature of the dataset we can obtain great differences in the runtime values. So we decided to develop a new implementation of the Apriori algorithm that better adapts to our specific datasets.

**Acknowledgments.** This work is supported in part by the Spanish Junta de Comunidades de Castilla-La Mancha PAI06-0093 grant.

## References

1. Agrawal, R., Imielinski, T., Swami, A.: Mining association between sets of items in massive database. International Proceedings of the ACM-SIGMOD International Conference on Management of Data (1993) 207–216.
2. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules. Proceedings of the International Conference on very large Data Bases (1994) 407–419
3. Bodon, F.: A trie-based APRIORI implementation for mining frequent item sequences. In Proc. Of the 1st Int. Workshop on open source datamining. Chicago (2005) 56-65
4. Borgelt , B.: Efficient Implementations of Apriori and Eclat. Workshop of Frequent Item Set Mining Implementations. Melbourne, FL, USA (2003)
5. De Bra, P., Aroyo, L., Chepegin, V.: The next big thing: Adaptive Web-based systems. Journal of Digital Information, Vol. 5-1 (2004)
6. Goethals, B.: Survey on frequent pattern mining. Technical report, Helsinki Institute for Information Technology, (2003)
7. Kosala, R., Blockeel, H.: Web Mining Research: A survey. ACM SIGKDD, Vol. 2 (2000) 1-15
8. Mobasher, B., Cooley, R., Srivastava, J.: Automatic personalization based on web usage mining. Communications of the ACM, Vol. 43 (2000) 142–151