# Evaluation of the Prediction Capability of a User Behaviour Mining Approach for Adaptive Web Sites.

**Brigitte Trousse**

Inria - AID Research Group, B.P. 93 - 06902 Sophia Antipolis Cedex, France
(Brigitte.Trousse)@inria.fr

**Abstract**

This article addresses the problem to predict user behaviours on a Web site which is very known to be an important way for improving the effectiveness of Web sites. Our objective is to evaluate our collaborative filtering approach of recommendation computation called the BROADWAY approach, according to the prediction feature. The main assumption of your approach is to take the sequence of user actions of a group into account for recommendation computation. For the prediction evaluation, we use the log files containing two months of usage from the site http://www.hyperreald.,org/music/machines/. We develop a prediction system based on our approach for that site and implement two others Web mining algorithms (statistics, Markov) in order to compare our work with them. This paper reports the main results issued from our different experiments.

## 1 Introduction

In IJCAI'97, Perkowitz and Etzioni (1998) challenged the AI international community to address the problem of creating what they call *adaptive Web sites*[1], i.e. Web sites which automatically *improve* their organisation and their presentation by learning from user access patterns. An important way to allow such an auto-adaptivity and to improve the effectiveness of Web sites is to predict user behaviour. So the purpose of this work is to answer the above cited challenge and to evaluate our collaborative filtering approach of on-line recommendation computation called the BROADWAY approach (Trousse, Jaczynski, and Kanawati, 1998; Trousse, Jaczynski, and Kanawati, 1999), according only to the prediction feature. To achieve this goal, we apply our approach of recommendation computation to support users during their navigation in a Web site by providing recommended Web pages. Our approach is more general and could be applied for different purposes (query formulation (Kanawati, Jaczynski, and Trousse, 1999), ranking, etc.).

The BROADWAY approach is based on the following assumption: *two (similar) users went through a similar sequence of Web-pages, they might have similar browsing intend, so that we can recommend one user the documents evaluated (or interpreted by the system) as relevant by the other one.* Modelling user behaviours with our approach is based on a bottom-up[2] approach which consists in observing some chosen variables (eg visited page, page content) and recording their evolution.

In this paper, our goal is to evaluate the predictive feature of our approach and validate the main assumption: *the next user action in the user session depends on the previous user actions in the user session.*. Indeed, as others (Spiliopoulou, Pohle, and Faulstich, 1999; Aumann et

[1]See http://www.cs.washington.edu/homes/map/adaptive/

[2]A top-down approach would consist in matching user actions to a pre-specified user behaviour model.

al., 1998), we think that sequence mining is the appropriate paradigm for analysing the behaviour of users in a Web site. To validate such an assumption, we compare our method with a classical Markovian one taking only the current user action into account and,then a statistical one (Aumann et al., 1998) taking temporal aspects into account as we did. Our experiments use log files proposed by Perkowitz and Etzioni (1998) containing two months of usage from the MusicMachines site (cf. Section 5.1). So we develop a BROADWAY-based system for that site and implement the two cited Web mining algorithms. Then we made different experiments whose the main results are reported in this paper.

The rest of the paper proceeds as following. Section 2 describes the prediction problem then Section 3 introduces the BROADWAY approach of on-line recommendation computation. In Section 4, we then explain how Case based Reasoning (CBR) techniques are used to implement such an approach, in particular how to deal with cases with temporal indices for on-line Web-page recommendation service for a Web site. Section 5 describes our experiments with the evaluation method and the main results. Finally Section 7 reprots on the current states of our work and on future activities.

## 2 Predicting User Behaviour from Web Server Logs

In this section, we describe the problem of predicting user behaviours from Web server logs which is extracted from (Aumann et al., 1998). User navigates through a Web site and we would like to predict which page he will access next. To achieve it, we have the past Web server logs. We consider the case where events (URL request) occur in sequence. Formally, let $E = \{E_1, \ldots, E_n\}$ be the set of possible events (all the URLs the user has accessed since he entered the site). A sequence is a tuple $S = (e_1, \ldots, e_k) \in E^*$. Having a training set X of sequences, $X = \{S_1, \ldots, S_n\}$ (past logs), we seek an algorithm A, such that when we have a sequence S' of a testing set Y, the algorithm outputs a prediction $e = A(S')$. A is said correct if, for a given sequence $S = (e_1, \ldots, e_k, e_{k+1})$, $A(e_1, \ldots, e_k) = e_{k+1}$. Two main metrics are classically used:

- The **accuracy** of A on a testing set Y of sequences, is classically defined by the fraction of sequences in Y for which A is correct. That means.when the first recommendation is equal to the next visited page (i.e. what we call "next is next"). Such an accuracy is computed according two ways: 1) on the visited pages where there is at least one prediction (cf. the *accuracy* or the precision of predictions) or 2) on the total number of visited pages where are requested predictions (cf. the *global prediction capability* of the system). In addition, another notion is often introduced, that is the *confidence* of the prediction which is dependent to the adopted method (cf. section 5.1).
- The **coverage** of the algorithm A is the ratio of the number of predictions on the number of queries (the algorithm does not always make a prediction). This metric is complementary to the accuracy metric i.e the precision of predictions.

## 3 The BROADWAY approach

This section describes our general approach of recommendation computation. It consists in recommending to an user what other (similar) users that have behaved similarly to that user, had positively evaluated[3]. Other systems in IR field rely on using user behaviour similarity as a basis for recommendation computation (Aumann et al., 1998).

CBR techniques are used to implement the BROADWAY approach (Jaczynski and Trousse, 1998;

[3]The evaluation could be explicitly expressed by an user in the context of intranet, user community, project team or could be issued from the interpretation of user actions by the system as successful actions (for example, buying a product, ordering a book, etc.)

Trousse, Jaczynski, and Kanawati, 1999). CBR is a problem solving methodology where, in order to find the solution to a current problem, one looks for a similar problem in an experience base, takes the solution from the past and uses it for a starting point to find a solution to the current problem. A *case* is a contextualised piece of knowledge representing an experience. It is generally composed of two main parts: 1) *the problem* that describes the state of the world also called the situation) when the experience occurred. and 2) *the solution* that states the derived solution to that problem. In this approach, the situation part of a case is composed of the user behaviour while the solution gives the result of that behaviour i.e. next relevant user actions according to his implicit goal. The use of CBR techniques to implement such an approach needs to deal with *cases with time-extended situations* (Jaczynski, 1997). CBR*TOOLS, an object-oriented framework (Jaczynski and Trousse, 1998) for CBR applications development allows the manipulation of such cases by implementing a *time-extended situation indexing model*. Following this indexing scheme, user behaviour is modelled by a set of *time series* that hold observations of the evolution over time of variables that are said to be relevant to describe the user behaviour. The choice of these variables depends obviously on the application domain and the type of recommendation to compute. Time series are grouped into records that have an application-dependent semantic. The use of the *time-extended situation indexing model* involves hence the following phases:

1. Identification of observation variables related to user actions and definition of records and associated semantics.

2. Definition of case structure to use: the situation part and the solution part.

3. Definition of reasoning steps: how to retrieve cases, how to reuse solutions, how to revise and how to learn new experiences.

Two main recommender systems based on this approach were already developed for customised web services: BROADWAY V1, a browsing advisor for the Web (Jaczynski and Trousse, 1999) and BROADWAY-QR, a recommender system for supporting the query formulation in the context of meta-search engines (Kanawati, Jaczynski, and Trousse, 1999).

Next, such an approach is based on two potential capabilities: prediction capability by providing earlier as possible shortcuts inside the sequence of user actions but also an emergence capability by providing new actions based on reusing experiences of a group of users. Next, as our goal is to evaluate the prediction capability of our Broadway approach compared with two others ones, we apply it for a Page recommendation service of a web site in order to support users during their navigation in the chosen Web site and then to improve the effectiveness of such a site..

## 4 Supporting Users During their Navigation in a Web Site

The application[4] of our approach for supporting users during their navigation inside a Web site is based on our first experience related to the Web browsing advisor BROADWAY V1 (Jaczynski and Trousse, 1999) (a part of Jaczynski's thesis work) which could be easily adapted to this new context.

### 4.1 Watching User's Navigations behind the Scene

Applying our approach in such a context relies on identifying a set of observation variables relevant for recommendation computation and issued from raw observed data. In order to build online adaptive recommender service based on our approach, we need to select a technology

[4]To implement BROADWAY-based recommender systems for supporting browsing on the Web or inside a Web site, we use two Java-based software developed by our team: BROADWAY*TOOLS and CBR*TOOLS (http://www.inria.fr/aid/cbrtools/manual/).

that makes possible such an observation (proxy as in Broadway V1, servlet, cookies, etc.). The system observes the users navigating inside the Web site and summarises their behaviour according to a set of variables that evolve over time for each user. Figure 1 shows an example (extracted from the Broadway V1 application) of a navigation having a behavioural situation observed via four variables: 1) page address, 2) page content, 3) user evaluation and 4) the display ratio corresponding to the display time relatively to its size (number of characters). The evolution of each variable is represented by sampled time series since the unit of the chosen model of time is a change of page. A navigation groups together the times series of an user for a given time interval.

### 4.2 Navigational Cases Indexed By a Time-Extended Situation

A navigational case references a navigation at a specific instant (time reference), making the separation between the past and the future. The list of advised pages represents the future pages of the case, which have been evaluated relevant by the system or highly relevant by the user. So cases are composed of three parts: 1) a time-extended situation from a navigation, 2) a list of pages which can be advised in that situation and 3) a set of data used to manage the case such as the date of creation. The time-extended situation has an instantaneous part and a behavioural part. The instantaneous part contains any information which is shared by all the cases based on the same navigation (for example, the navigation context, user profile, etc.). The behavioural part defines which pages or sequence of pages are relevant in the description of this situation. We need to define a case template that can be applied at a precise instant of the navigation to build potential[5] cases.

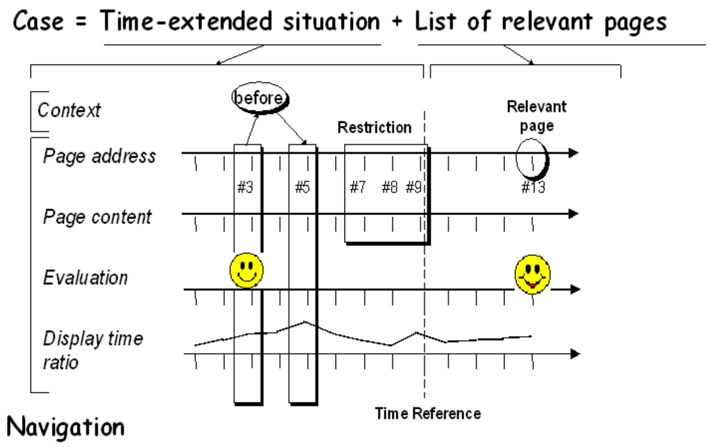


Figure 1: Example of a Broadway V1 Case

Figure 1 illustrates a potential case obtained with the following template where the case situation (cf. black boxes) is defined as follows: 1) the last three pages and their keywords (called the restriction), 2) a set of past relevant pages (user evaluation and display time ratio) and, 3) a set of before constraints which are used to relate selected pages. According to some conditions, potential cases may be stored in the case base as *concrete cases*.

### 4.3 Reusing Past Navigation Sequences

The recommender module of a Broadway-based system follows the classical CBR cycle composed of the four following steps:

[5]We use the term *potential case* obtained from raw data at the difference of the usual meaning of *cases*, which supposes cases validated by experts.

The **case retrieval** in a BROADWAY-based system uses a complex strategy with two alternatives: concrete case retrieval and potential case retrieval. The first alternative only considers concrete cases. If no concrete case with enough similarity is found, the second alternative will be executed to retrieve matching potential cases. This strategy encourages the reuse of existing concrete cases rather than identifying potential cases. In addition, this strategy leads to a better efficiency of the retrieval step because the first alternative requires less computations.Both the alternatives may require different sequential filtering steps. They use different local similarity measures for each variable and for the temporal constraints comparison. Then a standard global similarity measure (weighted average) is used to aggregate local similarities, allowing the comparison between navigational cases. We describe here the main similarity measure used to compare the page addresses (i.e URLs). A page address (ie. URL) is identified by its last node and we use the following similarity measure, if $p$ and $q$ are two addresses of pages:

$$S_{ad}(x,y) = 1 - \frac{h(x, MSCA(x,y) + h(y, MSCA(x,y))}{h(x, root) + h(y\ , root)}$$

where h() gives the number of links between two nodes and MSCA() gives the most specific common abstraction of two nodes.

The **retrieval step** has identified a list of relevant cases and each case gives a list of recommended pages. In the reuse step, the k-most reusable pages are selected and ordered according to their *re-usability* (i.e. their *confidence* ). The re-usability of each page is computed by a weighted average of different features, for example, based on the number of cases that recommend this page, the average similarity of the cases that recommend this page, the average loading time of the page. In some group-oriented applications (cf the system BROADWAY V1), we can also add, if possible, the average user's evaluation of the page in the retrieved cases and the attachment of annotations given by users

The goal of this **revision step** is to revise the solution of navigational cases i.e the suggested list of advised pages issued from the previous step, the reuse step. During the **learning step**, different aspects are addressed: when adding/deleting navigations and cases and how to re-organise the case base and the navigation base, etc.

For more details on the specification of these four steps in the context of supporting browsing, we recommend to have a look on BROADWAY V1 (Jaczynski and Trousse, 1999).

## 5 Preliminary Experiments

In the experiments[6] related to the prediction capability of our method, we have three main objectives: (i) to analyse the influence of the main configuration parameters of a Broadway-based system on the two metrics, (ii) to validate the assumption based on taking the sequence of user actions into account (cf. Section 1) and (iii) to compare our Broadway method with the work of Aumann et al. (1998) based on the same assumption.

### 5.1 Method

To achieve it, we implement a Broadway-based system called BROADWAY-PREDICT and two others algorithms described further which are applied on Web server logs for predicting the next page visited by an user during his navigation in a chosen web site. Our evaluation method

[6]More details could be found in an internal AID (INRIA) document (1999) (in French) on the "Evaluation of the Broadway Approach, for the Prediction of User Behaviours on the Web. Application on the MusicMachine Site", 52 pages

consists of running the considered algorithm twice times: one for the training phase and one for the testing one. Before detailing the pre-processing of logs and these two phases, we give some information about the log data and the system configuration.

**Log Data** Our experiments draw on logs collected from the Music Machine[7] Web site, a site dedicated to information about many kinds of electronic musical instruments. Logs are organised into two files: the first contains the server logs for the month of September 1997 while the second one contains logs for the month of October 1997. Each file contains all accesses to the MusicMachines site for a single day. The accesses are organised into *paths*. Each path is the series of URLs requested from a particular machine. Note that we do not distinguish among multiple users coming from the same source. A typical path will appear as below. The first line contains the originating machine (converted to unique numbers for the sake of anonymity). Each succeeding line corresponds to one URL requested from that machine. Each request contains the originating machine (O), the time of the request (T), the URL requested (U), and the referring URL (R). Fields are separated by "||" as shown below:
O:0000002560 || T:1997/09/12-22:43:00 || U:/ || R:http://www.hyperreal.org/

**Configuration of Broadway-Predict** Its configuration is minimal as it is only based on HTTP file and it only implements the first two CBR steps in order to facilitate our comparative evaluation with the two off-line methods. For *recording navigations*, we use a navigation context composed of the list of words used in the URL address without the two constant first ones (www.hypereal.org and machines) and two behavioural indices: the URL of the visited page and an implicit evaluation based on the display time of the visited page relatively to its file-type derived from its suffix (.gif, .html, etc.). The maximum time $d$ of visited URLs was first calculated on logs and the implicit evaluation takes its value inside (-1, 0, 1, 2) according the following rules: -1 if d $\in$ $[0,\ m/3]$, 0 if d $\in$ $]m/3,\ m]$ where the visited page is the first or the final one of the sequence, 1 if $\in\ ]m,\ m*5]$ and 2 if d $\in\ ]m*5,\ +[)$, where $m$ is the median of the time of the visited page. Such a median was calculated on the entire log file for each file-type (html pages and texts (13s), images (1s), audio files (18s), archives (48s). For *case retrieval*, we use two sequential filtering steps: navigation context filtering and restriction filtering where the restriction is composed only with the last three pages[8]. The case template is composed of the restriction and a set of relevant pages based on the implicit evaluation. After a first experiment, we fix the configuration as follows. The maximum number of recommendations at this step is 100 and the chosen similarity threshold is 0.99. For *case reuse*, we use only the maximum number of proposed pages equal to 10 and a minimum page relevance threshold equal to 0.95. Such a page relevance (cf confidence) is based on the aggregation of two features: the representativity i.e the ratio between the number of cases that recommend this page and the number of selected cases and and the average similarity of the cases that recommend this page.

**Log Pre-processing and Splitting Method** Our approach relies on mining standard HTTP[9] servers logs in order to recognise and analysis user access patterns. A first step in any path mining algorithm consists on identifying single user sessions from the log files. Two main obstacles for recognising user sessions from log file data are: how to identify users, and how to identify sessions. We make the assumption that one machine is used only by one user and a session is not interrupted more than 30mn, if such a case we split the connexions in different sessions issued from the same user. The paths files are pre-processed and merged in one file composed of 32000 navigation sessions. Then we split the paths into two sets: the training set

[7]htpp://machines.hyperreal.org

[8]In fact, it would be interesting to study the influence of the length of such a restriction on the two metrics.

[9]A typical HTTP-server log file contents the following information for each line: the client IP address, the HTTP request date and time, the requested URL and finally the request result.

and the testing one. First the memory of navigations is built by executing once the system. Then we evaluate the prediction power of the system by executing it on the testing set. We use a random method to split the file into two sets which is based on an chosen inclusion rate between the training and testing sets.

**Training and Testing Phases** The goal of the training phase is to record the sessions of users from the training set composed of a sequence of navigations. A testing set issued from the session logs is now used. The recommender module provides a list of recommended Web pages by reusing past experiences of the training set. For our experiments, we use two free parameters: the training set size (600, 1000, 2000, 4000 and 8000 navigations) and the inclusion rate between the training set and the testing set (0, 25, 50, 75, 100%). The testing set size is fixed to 60 navigations and the minimal (or maximal) length of the selected navigations (due to our restriction length equal to 3) is 4 (respectively 50). The experiments were conducted for 25 simulation configurations and, for each configuration, three random runs were performed i.e 75 runs in total.
The first experiment related to BROADWAY-PREDICT concerns the objective (i) and also aims to analyse the influence of the training set size and the inclusion rate on the two metrics. The second one which is related to the two objectives (ii) and (iii) addresses a comparative evaluation of three algorithms.

**Two Web Mining Algorithms** Two others algorithms[10] are implemented in Java language. The LxS (LXS = Length time Support) algorithm proposed by Aumann et al. (1998) for pre-fetching Web-pages and based on the same assumption is divided into two phases: training and testing. In the training phase, the algorithm extracts frequent[11] episodes (with a maximum length) from historical data which are then used in the testing phase. For the testing phase, given a sequence $S = (e_1, \ldots, e_m) \in E^*$, they wish to predict the next event e. To do so, they consider all the sequences $S' = (e_1, \ldots, e_m, e)$ $e \in E$ and all suffixes of those sequences and, then they compute their support[12]. For the sequence S, for each possible event e, and for a length i=1,...,m let $(e,i) - extension(S) = (e_{m-i+1}, \ldots, e_m, e)$, the extension of the last i events of S with the event e. For $S' = (e,i) - extension(S)$, they define the weight of S' as follows:

$$weight(s') = \begin{array}{ll} support(S').i & \text{if S' is frequent} \\ 0 & \text{oterwise} \end{array}$$

The strength of a prediction e is obtained by summing all the weights of extensions ending in e. Thus, for each possible extension e, they compute:

$$strenght(e) = \sum_{i=1}^{m-1} weight((e,i) - extension(S))$$

and make a prediction (only if $0 \neq max\{strength(g) : g \in E\}$), equal then to the extension with the highest strength. $predict(S') = e$ s.t. $strength(e) = max\{strength(g) : g \in E\}$

We compared also our work as Aumann et al. (1998) did to the natural Markov algorithm as a base line. This algorithm predicts the next event based on the current event alone. Thus, for each pair of events (e,g) we compute $support(e,g)$. Then, for each event e, we compute:

$$best(e) = g \text{ s.t } sup(e,g) = maxsup(e,g).$$

[10]More details on such algorithms could be found in an internal Draft AID (INRIA) document of S. Van Der Hoeven on "Studies of two Web Mining Algorithms for Web-pages Prediction", 20 pages, 1999.

[11]Aumann et al. (1998) say that a subsequence is frequent if it has support at least $\sigma$, for some threshold $\sigma$. The value of $\sigma$ is application dependent. After studying its influence on the accuracy and coverage metrics, we choose 5 as value. For the confidence and the length we choose respectively 1.5 and 5.

[12]The support of a subsequence S' in a sequence S is the number of times S' appears in S as a subsequence.

## 5.2 Results

**Results for BROADWAY-PREDICT** Results show that the number of retrieved cases increases when the training set size increases, but it doesn't increase when the inclusion rate between the training and the testing sets increases. Figure 2 shows that the accuracy (i.e. the prediction rate) increases (from 0.3 to 0.8) when the inclusion rate increases. The coverage increases in the same way than the accuracy but is relatively low (0.2) when the inclusion rate is equal to 0 with a small learning set of 600 navigations.

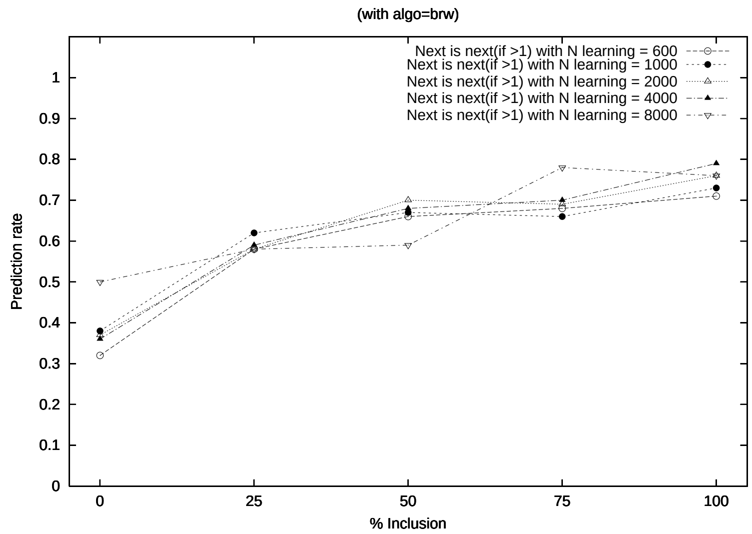


Figure 2: Influence of the inclusion rate on the accuracy

**Comparative Results** The first result concerns the influence of the inclusion rate (with a training set of 8000 navigations) on the two metrics. Figure 3 shows that the accuracy (respec-

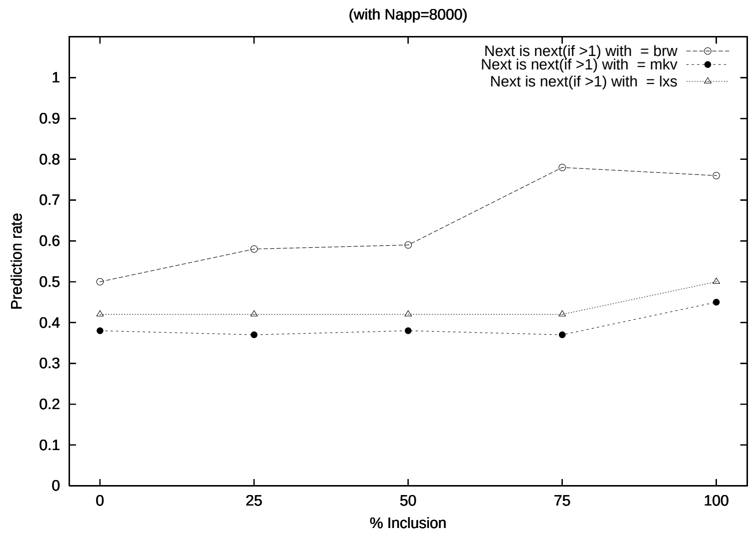


Figure 3: Influence of the inclusion rate on the accuracy

tively the coverage) seems to be stable around 0.4 (respectively between 0.5-0.6) for the two others algorithms when the inclusion rate increases, whereas the accuracy with our approach is up to 0.5. On the other hand, for BROADWAY-PREDICT, the coverage increases when the inclusion rate increases, staying inferior to the two others algorithms (from 0.2 to 0.45).

The second result concerns the influence of the training set size on the two metrics. With an inclusion equal to 0%, the coverage seems to raise a maximum value around 0.5 from a training set of 2000 navigations for the two others algorithm, whereas, for our BROADWAY-based system, it seems to be around 0.2 but for any training set size. Moreover Figure 4 shows that the accuracy increases slightly with the training size for the two others algorithms at the difference of our system. The accuracy is better than the two others algorithms when the training set overpasses 8000 navigations.

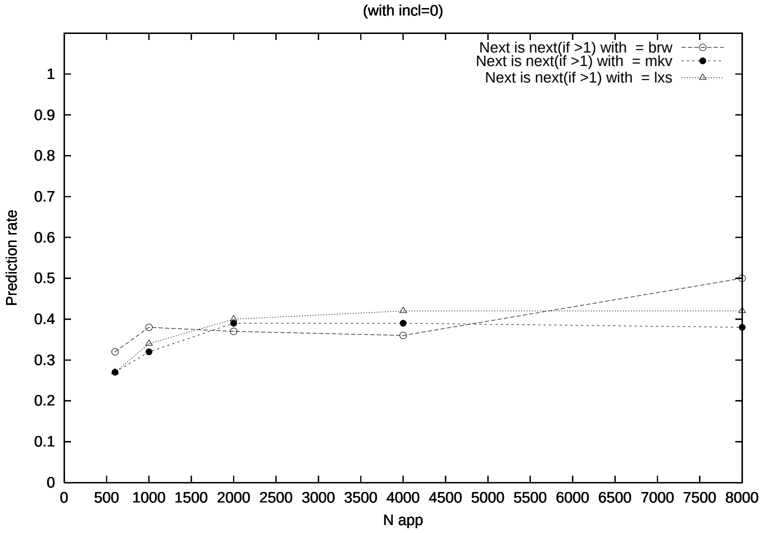


Figure 4: Influence of the training set size on the accuracy

To conclude, comparative results first validate our objective (ii) showing that the two methods taking into account previous user actions to predict the next one have a better accuracy than the Markovian one. Second results show a better accuracy for our approach compared with the work of (Aumann et al., 1998).

## 6 Conclusion and Future Work

In this paper, we proposed an original customised recommendation computation approach which takes the sequence of user actions into account, recording such sequences as a *set* of time series at the difference of existing web mining works. Such a computation is based on case-based reasoning to analyse and reuse user sessions. Next, we presented how to apply such an approach on improving a web site by dynamically adding customised Web-page recommendation. To make some experimental evaluation of our approach, we then implemented BROADWAY-PREDICT, a BROADWAY-based service for a specific musical Web site and two others approaches (statistical and Markovian), applying them on log data. Results on the accuracy for our approach are very promising. Concerning the low coverage (0.25 %) obtained with our approach, we explain this by the minimal system configuration (case structure,...) we have designed due to a very poor semantics of the chosen site

In future work, we seek to improve the efficiency of our method by using better case indexing and to test our approach on additional Web sites, more semantically structured than the used one. Indeed, in our opinion, results could be improved by taking more semantics of the chosen Web site in the definition of the navigational case structure (in particular, page content and user evaluation). In addition, we plan to follow our comparative analysis with others relevant path mining methods such as the clustering one (Yan et al., 1996; Perkowitz and Etzioni, 1999; Spiliopoulou, Pohle, and Faulstich, 1999). Finally, we think that the BROADWAY approach applying to customised recommendation Web services is an interesting step towards the long-term goal (expressed by (Perkowitz and Etzioni, 1998)) i.e to create adaptive sites that automatically suggest alternative organisations of their contents based on visitor access patterns. Such a goal is related to what Despeyroux and Trousse (2000) call the specification of *dynamic semantics of a Web site*.

**Acknowledgements** We want to thank three past AID members: two students F. Martineau and S. van Der Hoeven for their implementation work and specially R. Kanawati for his active participation in the initial steps of this work and his contribution.

## References

Aumann, Y., O. Etzioni, R. Feldman, M Perkowitz, and T Shmiel. 1998. Predicting Event Seqences : Data Mining for Prefetching Web-pages. Submitted paper to the Fourth International Conference on Knowledge Discovery and Data Mining (KDD'98).

Despeyroux, T. and B. Trousse. 2000. Semantic Verification of Web Sites Using Natural Semantcis. In *Proceedings of the 6th RIAO Conference - Content-Based Multimedia Information Access*, Paris, France, april.

Jaczynski, M. 1997. A Framework for the Management of Past Experiences with Time-Extended Situations. In *Proceedings of the 6th International Conference on Information and Knowledge Management (CIKM'97)*, pages 32–39, Las Vegas, Nevada, USA, november.

Jaczynski, M. and B. Trousse. 1998. An Object-Oriented Framework for the Design and the Implementation of Case-Based Reasoners. In *6Th German Workshop on Case-Based Reasoning*, Berlin, Germany.

Jaczynski, M. and B. Trousse, 1999. "Broadway : a Case-based System for Cooperative Information Browsing on the World-Wide-Web". In *Collaboration betwen Human and Artificial Societies. Coordination and Agent-based Distributed Computing*, pages 264–283. LNAI Series. J. A. Padget (editor).

Kanawati, R., M. Jaczynski, and B. Trousse. 1999. Applying the Broadway Recommandation Computation Approach for Implementing a Query Refinement service in the CBKB Metasearch Engine. In *Actes de RàPC'99, Raisonnement à Partir de Cas*, pages 17–26, Palaiseau, France, juin. AFIA.

Perkowitz, M. and O. Etzioni. 1998. Adaptive Web Sites: Automatically Synthesizing Web Pages. In *Proceedings of Fifteenth National Conference on Artificial Intelligence AAAI'98*, pages 727–732, Madison, Wisconsin, USA, July.

Perkowitz, M. and O. Etzioni. 1999. Adaptive Web Sites: Conceptual Clustering Mining. In *Proceedings of Sixteenth International Joint Conference on Artificial Intelligence (IJCAI'99)*, pages 727–732, Madison, Wisconsin, USA.

Spiliopoulou, M., C. Pohle, and L. Faulstich. 1999. Improving the Effectiveness of a Web Site with Web Usage Mining. In *Proceedings of the International Workshop on Web Usage Analysis and User Profiling (WEBKDD'99)*, august. http://www.wiwi.hu-berlin.de/ myra/WEBKDD99/index.html.

Trousse, B., M. Jaczynski, and R. Kanawati. 1998. Towards a Framework for Building Collaborative Information Searching systems. In *Proceedings of the second european conferenve on digital libraries (ECDL'98)*, LNCS. Springer, September. (Poster session).

Trousse, B., M. Jaczynski, and R. Kanawati. 1999. Using User Behavior Similarity for Recommandation Computation : The Broadway Approach. In H-J. Bullinger and J. Ziegler, editors, *Proceedings of 8th International Conference on Human Computer interaction (HCI'99)*, pages 85–89. Lawrence Erlbaum Associates, august. Munich.

Yan, T.W., M. Jacobsen, H. Garcia-Molina, and U. Dayal. 1996. From User Access Patterns to Dynamic Hypertext Linking. *Computer Network and ISDN systems*, 28:1007–1014, May. also in Proceedings of the 5th International WWW Conference WWW'96.