Link Recommender Systems: The *Suggestions by Cumulative Evidence* Approach

Jean-Yves Delort [a], Bernadette Bouchon-Meunier [a]
[a] *Computer Science Laboratory (LIP6)*
Paris 6 University
8, rue du Capitaine Scott
75015 Paris, France
E-mail: {Jean-Yves.Delort,Bernadette.Bouchon-Meunier}@lip6.fr

The increasing interest of website designers in customization technologies has stimulated recommender system researches. This paper deals with link recommender systems (LRS). LRS are tools intended to support and assist the users during their navigation on the Internet or on a specific website. LRS differ from object recommender systems (ORS) which are tools intended to make the user buy or access resources. If ORS efficiency depends on the number of times the users have followed the recommendation proposed by the system, LRS can neither be evaluated, nor compared, with respect to this only criteria. LRS suffer from the uncertainty of the available information about the user navigation behaviors and preferences. This paper introduces a new LRS intended to take this uncertainty into account in the recommendation process. Then, the issue of the evaluation of LRS is addressed and a some general features of LRS are put forward and discussed. Then, the proposed approach is compared with two other link recommendation algorithms. The proposed approach provides good results when evaluated with respect to ORS criteria and appears to be the best tested approach in the context of link recommendation.

Keywords: Recommender systems, Links, Uncertainty

1. INTRODUCTION

The increasing interest of website designers in customization technologies has stimulated recommender system researches (RS). Online recommenders can be distinguished into two categories: *Object recommender systems* (ORS) and *Link recommender systems* (LRS). ORS aim at providing suggestions on a set of resources such as products or news. They are intended to motivate or encourage users to follow them. Then, a set (or a list) of related objects matching the user's current interests is proposed. Preferences are often, but not necessarily, explicit votes. When behaviors are not ambiguous (i.e. the users intents or preferences can easily be understood) they will be interpreted as user votes in favor of, or against the objects. For instance, if a user buys a product, this is very likely to mean a user implicit vote in favor of the it. ORS are often used by retailers, such as Amazon.com, to suggest future purchases. The purposes of LRS are different. LRS aim at supporting or assisting users when they are browsing on a specific website or on the Internet. For instance, a LRS could be designed to help the users not to get lost when they are browsing in a large and depth website. In this paper, we focus on LRS intended to a support user while they are browsing on a given website.

The evaluation process of LRS is more complex than the ORS one. Indeed ORS, can be evaluated with respect to their ability to predict the next user clicks. This ability can be precisely and easily computed. Conversely, LRS are harder to compare: for instance, to which extent a RS helps the user to feel comfortable on the website and not to get lost on it?

In general, the making of user profiles and the extraction of user's preferences is based on a preprocessing step taking as an input the transaction

set recorded by the *Hypertext Transfer Protocol* (HTTP) server. This leads, because of the drawbacks of this protocol, to an uncertain representation of the user's behavior and of his choices.

This paper makes the following contributions: First *SCE*, an original link recommendation algorithm that handles uncertainty in user sessions is presented. It is based on Dempster-Shafer theory of evidence [15]. Then, general features of LRS are introduced and discussed. One of their advantage is that they can be expressed as numerical values which can be compared. Finally, the proposed algorithm is compared with two other link recommendation algorithms with respect to the ORS evaluation criteria s well as with the new ones. The proposed approach is shown to provide good results when evaluated with respect to ORS criteria and appears to be the best tested approach in the context of link recommendation.

2. RELATED WORKS

Recommender systems tend to be distinguished into two classes: content-based RS use the content of the items (or some information related to them) to suggest links or objects (for instance the textual content or metadata such as annotations). Collaborative RS are trying to find similarities between the users' behaviors and interests in order to decide which recommendations to make. Examples of content-based RS are search engines and examples of collaborative RS are the Alexa[1] system or the GroupLens system [10]. Alexa is also an example of LRS: it provides a "What's related" service that suggests Web pages or websites with respect to common points discovered between the user's behavior and the database of all the Alexa users' navigation behaviors. Another example of collaborative RS, but of the ORS kind, is the GroupLens Project that recommends messages in a newsgroup. The *Surflen* [7] project is a RS close to Alexa. Each client sends to a server its navigation history. Then a recommendation engine uses the whole user browsing history database to compute recommendations and send them back to the clients. Balabanovic [2] has proposed an ORS based on a collaborative algorithm aiming at taking advantage of the content as well.

[1]http://www.alexa.com

A frequent assumption with LRS is that a user access to a page implies an implicit vote in favor of this page. LRS approach is more general than the object one, and such RS can be deployed on any website because algorithms just need always available data. Mobasher *et al.* [12] pioneered LRS research. They proposed two LRS algorithms. The former uses frequent association rules between URLs within the same sessions, while the latter clusters similar user sessions. However the algorithms they proposed have not been evaluated nor any evaluation methodology has been proposed. Letizia [11] is an agent to assist users when they are browsing on the Internet. The system combines two different recommendation strategies: In one hand it tries to predict the next user clicks and on the other hand it takes into account heuristics inferring user interests from their browsing behavior.

Breese [3] and Spiekerman [16] have introduced alternate evaluation strategies for ORS. Yet, the proposed approaches were still highly connected to the abilities of the RS to make accurate predictions. On the contrary, the issue of the LRS evaluation implies cognitive factors. For instance to which extent a RS helps the user to feel comfortable on the website and not to get lost on it?

At the same time of RS researches, the outcomes of works on prefetching have highlighted the efficiency of user navigation models on a web site. Prefetcher systems try to predict the next resources that a visitor will request immediately after. This can be very efficient if the pages are dynamically made because they can be pregenerated. Prefetcher objectives are closed to ORS but surprisingly they have not been used for a recommendation purpose. In this paper an interest into one of the most used and efficient user navigation model is taken. The markovian approach has been extensively studied as a user navigation model[14,6,13]. It is based on a graph the states of which represent the last k resources accessed by a user called *active windows*. Transition matrix expresses the probability that a user will click on a link with respect to the k last pages he has seen. The main drawback of this statistical representation is its extreme memory complexity when k and the number of pages increases. However this paper will also demonstrated that this approach is efficient when used by an ORS.

3. UNCERTAIN DATA

3.1. Hypertext Transfer Protocol

One of the reasons of the success and popularity of the Internet is probably that the mainly used protocol, HTTP lets the users remain anonymous. As a counterpart of this, if we want to add personalization services to a website, we are bridled because available data on the users are scarce. Furthermore, as HTTP does not maintain persistent connection, it is neither possible to know what the page the user is currently viewing nor it is possible to be sure that the user is still viewing a page of the website.

With the introduction of client-side stored information (the cookies) user profiling has become possible. Nevertheless, due to privacy reasons the right of using cookies depends on the user acceptance. Yet, the users are often reluctant to be tracked while they are browsing on a website. A great deal of personalization services use the link structure (the dependency graph), the resources and meta-descriptions (if they exist), and above all, the HTTP user access log file. Each line of it contains the minimum pieces of information for a transaction between a client and the server to occur. It is described by, at least [1]:

- the client IP address,
- the requested URL,
- the protocol version,
- the transaction status[2],
- the date and time.

Depending on the HTTP protocol version additional information can be sent, such as the REFERRER which indicates the previous resource accessed by a user on his browser before his last request. The REFERRER is an interesting data which can be used to improve the retrieving process of the user accessed pages.

3.2. Uncertain data

Because of dynamic IP addressing, it is not always possible to recognize a user because he may have accessed the same website by means of different IP addresses. Besides, the unique knowledge of the IP address is not sufficient to know exactly if the user is a single person or not. Indeed, if the terminal is used by more than one person, or a set of users access the Internet *via* a gateway, the IP address recorded by the HTTP server will be the same. In the sequel, we will use indistinctly the terms *user* and *client* to refer to any terminal that makes requests to a server and which can be labeled with a unique identifier.

To relieve the load of the server, a request is made only if the resource is neither in the client cache nor in a proxy server or the document needs to be refreshed. Thus, nothing guarantees that some pages have been accessed more than once, or that some URLs not recorded in the log file have been accessed. The cache client keeps in memory the previously seen pages in order not to have to require resources that have recently been accessed. In the same vein, proxy servers act like a unique shared client cache among a set of users. Provided one of the users accessed the resources, they will be directly sent to any other users connected to the same proxy and who is willing to access these same documents. These transaction will not be reported to the Web server. Thanks to the local cache and the proxy servers, the number of requests is tremendously reduced. However this performance improvement has a cost: it goes with a loss of information since the moves and current positions in the website of the users are then uncertain.

Finally, another problem comes from the fact that the users never explicitly vote for or against the resources they access. The starting point is thus to choose a way to decide whether he liked them or not, with respect to his behavior. The generally accepted hypothesis is to consider that each time a document is accessed it means an implicit vote in favor of it.

4. USER SESSIONS

4.1. Notations

A *resource domain* referrers to a set of resources accessible by users and is denoted by $\mathcal{U}$. A *session* is a suite of transactions which have occurred for some time between a client and a server. A transaction can be represented by a tuple $t = (ID, URL, DATE, ...)$ where $t.ID$ is the user id,

[2] A number, for instance 200 for "OK 200" and 404 for "Not Found 404".

$t.URL$ is the requested URL and $t.DATE$ is the transaction time. These three data uniquely identify any possible transaction. In the sequel, the letters t and u will be used when dealing with transactions and resources respectively. $< \mathcal{U} >$ denotes the set of all possible sequences of consecutive transactions of $\mathcal{U}$. We call *length of the session* $\overrightarrow{s}$, and we note $|\overrightarrow{s}|$ the length of the sequence $< t_1, ..., t_n >$. The symbol $< s >$ is an abbreviation of the sequence $< t_1.URL, .., t_n.URL >$.

4.2. User sessions

The concept of user session is often used to refer to the relations between the users' information needs and their interactions: it represents a set of accessed pages related to the same search activity. Most session boundaries detection algorithms are based on a time constraint between the access time of the resources within the same session [9,4]. Content-based approach have also been proposed [5].

Generally, the cohesion between the elements within a session depends on four relations linking the transactions together:

1. all the transactions where request by the same user,
2. all URLs belong to the same resource domain,
3. the transactions occurred in a row and
4. there exists an additional temporal relation on the transactions.

Usually the last condition comes down to a maximum time span between the transactions. It can be τ_0-*static*, if the time span between the first t_0 and the last accessed document t_n cannot exceed a given number τ_0 of seconds:

$$t_n.DATE \leq t_0.DATE + \tau_0$$

It can also be chosen τ_1-*dynamic* which means that the time span between two consecutive transactions cannot exceed a given number τ_1 of seconds:

$$\forall i = 2..n, \quad t_{i+1}.DATE \leq t_i.DATE + \tau_1$$

Let $\overrightarrow{s} =< t_1, .., t_p >$ be a user session, suppose that a new request t_{p+1} occurres, then the concatenation of sequences of transactions $\overrightarrow{s}$ and $< t_{p+1} >$, denoted by $\overrightarrow{s}.< t_{p+1} >$ is still a session. In this situation, the user will be considered to be still present on the site. Then $\overrightarrow{s}$ is called an *active session* and the user will be said to be an *active user*.

5. RECOMMENDATION ALGORITHMS

We call *recommendation function* any process taking as an input a session $\overrightarrow{s}$ and giving as an output a subset (possibly empty) of $\mathcal{U}$.

DEFINITION 1 *A recommendation function on a resource domain $\mathcal{U}$ is a multivaluated function:*

$$\overrightarrow{\mathcal{S}} \longrightarrow \mathcal{P}(\mathcal{U})$$
$$\overrightarrow{s} \longmapsto Rec(\overrightarrow{s}) = \{u_1, .., u_p\}$$

The set $Rec(\overrightarrow{s})$ is called the recommendation of the session $\overrightarrow{s}$.

Recommendation processes are often based on a function that associates a weight with each resource of $\mathcal{U}$. This weight conveys the degree of likeliness that the resource is a good or a bad recommendation.

DEFINITION 2 *Given a domain $\mathcal{U}$, $\overrightarrow{\mathcal{S}}$ the set of all the sessions on $\mathcal{U}$, a* valuated recommendation function *is a function $RecVal$ defined by:*

$$RecVal : (\overrightarrow{\mathcal{S}} \times \mathcal{U}) \longrightarrow [0, 1]$$
$$(\overrightarrow{s}, u) \longmapsto RecVal(\overrightarrow{s}, u)$$

The value $RecVal(\overrightarrow{s}, u)$ is called weight of the recommendation *of u. A recommendation is a set of pages of $\mathcal{U}$ having the highest $RecVal$ value.*

When a recommender system cannot give a recommendation, then it outputs the empty set.

DEFINITION 3 *Given a domain $\mathcal{U}$, a recommendation function Rec and a session $\overrightarrow{s}$, we will say that a recommendation* succeeds *if:*

$$Rec(\overrightarrow{s}) \neq \emptyset$$

6. MANAGING UNCERTAINTY WITH EVIDENCE THEORY

In this section the fundamentals of evidence theory are outlined. Then, the *Suggestion by Cumulative Evidence* (SCE) algorithm is introduced.

6.1. BELIEF THEORY: BASIC CONCEPTS

Evidence theory is a powerful tool to handle problems with uncertain and imprecise data. It was introduced by Dempster and Shafer [15]. A reference set $\mathcal{U}$, called *universe of the Discourse* or equally *frame of discernment* is introduced. It represents a set of mutually exclusive alternatives, for instance all the possible values of an attribute.

DEFINITION 4 *Let $\mathcal{U}$ be a universe of the Discourse. A function $m : 2^{\mathcal{U}} \rightarrow [0, 1]$ is called a* basic probability assignement over $\mathcal{U}$ *if:*

$$(1)\ m(\emptyset) = 0$$

$$(2)\ \sum_{A \subseteq \mathcal{U}} m(A) = 1$$

The amount $m(A)$ is called *basic value* of the probability $m(A)$ associated with the event A. It measures the strength of the belief that A will occur. A *focal element* of a belief function Bel is any subset $A \subset \mathcal{U}$ such that $m(A) > 0$ and the *core* of Bel is the union set of all its focal elements. To represent the reasons to believe in A, all the quantities $m(B)$ such that $B \subset A$ must be added to $m(A)$. This leads to define:

DEFINITION 5 *Let $\mathcal{U}$ be a frame of discernment, and m be a basic probability assignement over $\mathcal{U}$. A* belief function over $\mathcal{U}$ *is a function $Bel : 2^{\mathcal{U}} \rightarrow [0, 1]$ defined by:*

$$Bel(A) = \sum_{B \subseteq A} m(B) \qquad (1)$$

6.2. Suggestions by Cumulative Evidence

The proposed method relaxes the consecutively hypothesis of the pages within a session. Thus it can take into account all the transactions previously occurred in the session. The proposed approach, called *Suggestions by Cumulative Evidence* (SCE) is based on the idea that all previously seen pages and their combinations must play a role in the link recommendation decision process. After a suitable aggregation of all the evidence suggesting that a resource is connected to others, the global information on each resource should increases. Thus, for each admissible resource, a degree of belief that this page may interest the user, with regard to his history is computed. Then, pages are ranked with respect to their degree of belief and a classical technique (support pruned criterion [6]) is used to prune the pages that have low support in the learning database. In other words, to each pair (A, u), a weight $p(A, u)$ is associated which gives the strength of the following assertion: "In a session $\overrightarrow{s}$, such that $< s >=< u_1, .., u_n >$, if $A \subseteq < s >$ then $u \in < s >$". $p(A, u)$ conveys the strength of the relation characterized by the simultaneous presence of the resources of A and u within a given session. Thus $p(A, u)$ is the conditional probability of u knowing A. If the confidence of the rule $A \Rightarrow u$ is not zero, $p(A, u)$ matches with the confidence in the rule $A \Rightarrow u$. Otherwise, $p(A, u)$ is set to zero. During a training phase, the values of $p(A, u)$ are pre-computed for given minimum support $minsup$ and minimum confidence $minconf$. To each pair $(u, w) \in (\mathcal{U}, \mathcal{P}(\mathcal{U}))$ the following weight is associated:

$$m'_u(w) = conf(w \Rightarrow \{u\})$$

In order to respect condition (2) of definition 4 it is necessary to normalize:

$$m_u(w) = \frac{m'_u(w)}{\sum_{z \subseteq \mathcal{U}} m'_u(z)}$$

Thus, coefficients m_u are basic probability assignments. The valuated recommendation function associated with SCE algorithm is the following belief function:

$$Rec_{SCE}(\overrightarrow{s}, u) = \sum_{w \subseteq < s >} m_u(w)$$
$$= 0 \text{ otherwise}$$

To each page a hashtable is associated. A key for this hashtable is a frequent set w and its corresponding value $m_u(w)$, is recorded if it is different from 0. As the model depends on minimum support and minimum confidence thresholds, it is easily tunable to reach a reasonable size.

7. EVALUATION METRICS

Recall, precision and coverage measures are generally used to assess the efficiency of ORS [3,16]. However, these measures do not necessarily characterize the intent of LRS. In this section, new criteria intended to represent LRS are introduced

and modelized.
In a sake of simplicity, assume that $\overrightarrow{s} =< t_1, .., t_n >$ is a session and $p \leq n$, then Rec_s^p will denote the set $Rec_s^p = Rec(< t_1, ..., t_p >)$.

7.1. RECALL, PRECISION AND COVERAGE

The following definition sums up the classical evaluation measures used when evaluating ORS.

DEFINITION 6 *Let $\mathcal{U}$ be a resource domain, Rec a recommendation function, $\overrightarrow{s}$ a session of size n and $l \leq n$.*

- *A recommendation Rec_s^l is* covered*, if:*

$$|Rec_s^l| > 0$$

- *The* validity *of Rec_s^l is given by:*

$$val(Rec_s^l) = |Rec_s^l \cap (t_{l+1}, .., t_n)|$$

- *A covered recommendation is said* precise *if:*

$$val(Rec_s^l) > 0$$

- *The* recall *of a covered recommendation is:*

$$\frac{val(Rec_s^l)}{|Rec_s^l|}$$

The coverage *(resp.* precision*) of a set of recommendations is the average number of recommendations which were covered (resp. precise). The* recall *of a set of recommendations is the mean of their recalls.*

7.2. STABILITY, MOST FREQUENT PAGES AND K.L.D.

The *degree of stability* between two consecutive recommendations is defined by the number of common suggestions between them. The lower this value, the more unstable the recommendation function is and the more abruptly suggestions evolve. Theoretically once the system has understood the user information needs and navigation style the recommendation sets should not vary too much. Thus this degree should stabilize to a value close to 1.

DEFINITION 7 *Let $\mathcal{U}$ be a resource domain, Rec a recommendation function and $\overrightarrow{s} =< t_1, ..., t_n >$ a session. Let us consider a prefix of $\overrightarrow{s}$, $< t_1, ..., t_l >$ with $l < n$, the so-called* degree of stability *of Rec on the session $\overrightarrow{s}$ between t_l and t_{l+1} is defined by the number:*

$$\frac{|Rec_s^l \cap Rec_s^{l+1}|}{max(|Rec_s^l|, |Rec_s^{l+1}|)}$$

Previous works on prefetching have discovered a relation between the frequency of requests to Web documents and their rank of popularity, it is known as the *Zipf's law* [8]. Let us denote by ζ the rank of popularity (with regard to its frequency) of a resource and P the frequency of occurrence, then: $P \sim \zeta^{-\beta}$ with β typically close to 1. Such distributions imply that, usually, few resources concentrate the interest of the users. It can be expected that the pages recommended by a LRS do not differ too much from this law. It is important that the pages well ranked by their frequency are also recommended in a high proportion. The *Most Frequent Pages* ratio is thus defined by the number of frequently recommended pages over the total number of recommended pages.
However, a RS cannot only focus on most frequent pages because useful information is not always contained in most frequent pages. The *Kullback-Leibler divergence* (KLD) is well suitable to assess the difference between the distributions of the accessed and recommended pages. To compute the KLD, the probabilities $P_{learn}(u)$ are computed over the learning database that expresses the probability that a page u will be requested. During the testing step, the corresponding distributions P_{test} are computed over the set of sessions. The *Kullback-Leibler divergence* is given by the following formula:

$$I(P_{learn}, P_{test}) = \sum_{u \in \mathcal{U}} \log\left(\frac{P_{learn}(u)}{P_{test}(u)}\right) \times P_{learn}(u)$$

The higher this number, the less representative of the user navigation behavior the recommendations are.

8. TESTS AND DISCUSSION

This section presents first the testing datasets. Then, the two others link recommendation algo-

rithms that will be compared with the SCE approach are presented. Finally, the results are given and discussed.

8.1. DATASETS

Three HTTP user access log files have been considered[3]. The interest of these logs is that they are include different users' information seeking behaviors and interests. The first one considered is the WorldCup Soccer 98 official website. Note that the content of these website was highly dynamically generated and thus the content of documents having the same URL was likely to change in a few hours. For instance, the content of the homepage changed more than once a day during the contest. The second log considered belongs to *Music Machines*. This website address people interested in professional audio devices. The last user access log file comes from the website of ClarkNet, an US internet access provider.

Each log file was split up into two subsets of transactions, the former used to train the models and the latter used to test them. Once the useless transactions filtered (made up audio and visual materials requested because they were linked the requested pages), the sessions were extracted with a τ_1-dynamic algorithm with $\tau_1 = 5$ minutes (see table 1). Because of the problems due to caches and proxy servers only the first instance of each URL in the sessions was kept. For training as well as for testing, the only sessions with a size between 4 and 30 were kept (which correspond to more than 99% of the total number of extracted sessions for any log file considered).

8.2. COMPARISON WITH OTHER LINK RECOMMENDER ALGORITHMS

We have compared the *SCE* algorithm with two others methods. The first one is adapted from a classical prefetching approach based on $N-GRAM$ model. It is an adaptation we made in order to show that even if prefetching algorithms have good precision and recall they do not necessarily fulfill conditions making a recommender system to be good. In $N-GRAM$ models, the only last N pages of an active session are taken into account in the prediction process. The valuated recommendation function is trained over a session set $\mathcal{D}$. To do so, the probability $P_{\mathcal{D}}(\overrightarrow{s}, u)$ that a link u will be accessed with regard to the last N transactions of the session $\overrightarrow{s} \in \mathcal{D}$ is considered. The valuated recommendation function given by $RecVal_{NGR}(\overrightarrow{s}, u) = P_{\mathcal{D}}(\overrightarrow{s}, u)$ is the expression of the final recommendation function. Given the number of pages on the websites we consider and because of the problem of representation for the $N-GRAM$ model, we have chosen $N = 1$.
In [12], Mobasher *et al.* propose an algorithm based on frequent association rules between the resources of sessions. Their algorithm is also based on a distance factor between the nodes of the dependency graph. This factor is particularly interesting because it prompts the recommendation function to favor pages which are the farthest from the last one accessed. Thus, recommendations allow to save clicks to access the relevant information. They introduce the *Physical distance* between two nodes u_1 and u_2, $d(u_1, u_2)$ which is the minimal length of the path between u_1 and u_2. Then they define a physical distance between an active session $\overrightarrow{s}$ and a resource u_i for the graph $G =< \mathcal{U}, V >$ by: $dist(u_i, \overrightarrow{s}, G) = \min_{u_j \in <s>} d(u_i, u_j)$. Eventually the distance factor between a link u and a session $\overrightarrow{s}$ is:

$$ldf(u, \overrightarrow{s}) = log(dist(u, \overrightarrow{s}, G)) + 1 \text{ if } u \notin < s >$$
$$= 0 \text{ otherwise}$$

Mobasher *et al.*'s algorithm can be expressed more easily with the following valuated recommendation function:

$$Rec_{MOB}(\overrightarrow{s}, u) = conf(< w > \Rightarrow \{u\}) * ldf(u, w)$$
$$= 0 \text{ otherwise}$$

where $conf(< w > \Rightarrow \{u\})$ depends on the thresholds $minsup$ and $minconf$.

8.3. RESULTS AND DISCUSSION

For each dataset, recommendation process starts when the tested active sessions have a size larger than 2. Table 2 summarizes the measures previously defined [4] and the three graphics in Table 3 show the average stability of each recommendation

[3]Log files available at http://repository.cs.vt.edu and http://www.cs.washington.edu/research/adaptive/.

	M. Machines	W.C. Soccer	ClarkNet
Total transactions	359.853	1.193.353	1.654.882
Total different IPs	18.037	17.835	54324
Total different pages	4.970	1.014	8845
Total sessions	8.859	12.994	16814
Total sessions for training	7.714	8.736	10334
Total sessions for testing	1.145	4.258	6480
Average size of sessions	8.4	6.9	6.1

Table 1
Features of log files

	WordCup Soccer			Music Machines			ClarkNet		
	NGR	MOB	SCE	NGR	MOB	SCE	NGR	MOB	SCE
Cov	95%	63%	99%	86 %	10%	84%	99%	54%	94%
Pre	61.85%	68.61%	58.05%	48.98%	40.92%	40.23%	66.4%	52.7%	48%
Cov x Pre	58.76 %	43.22 %	57.46 %	42.14 %	4.09 %	33.79%	65.73%	42%	39%
Rec	36.1%	46.44%	34.2%	36.4%	32.02%	28.28%	39.47%	42.4%	39.9%
ARS	4.62	4.71	5	4.54	2.75	4.75	4.33	3.12	4.73
MFP	62.4%	91.08%	92.17%	18.76%	80.2%	76.33%	54%	86.14%	82.4%
KL	2.78	0.84	1.2	2.5	0.41	1.08	2.64	0.75	1.12

Table 2
Results

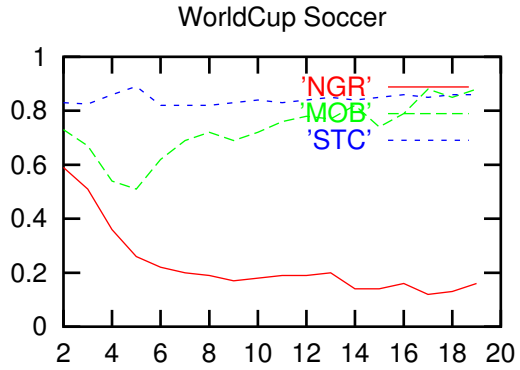
function between the second and twentieth pages in the sessions.

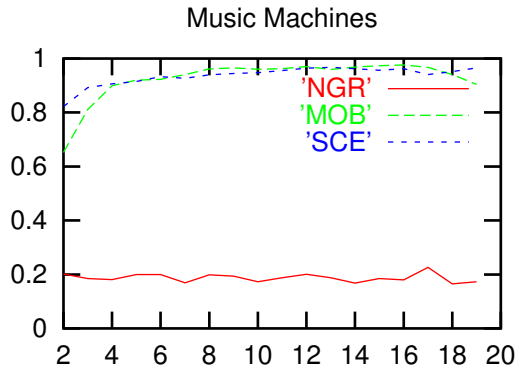
It can be seen that NGR always has the highest precision rate. This has the following explanation: NGR is a purely statistical approach which tries to predict the *following* pages of the active window. In contrast, MOB and SCE, two association rules based algorithms, are able to recommend pages that had great chances to be seen *before* the last one accessed. This can give the user alternate interesting ways to consider the documents he has accessed. However, as they are defined, precision and recall do not take it into account.

Secondly, it can be observed that association rule approaches are able to reach an excellent level of stability, whereas NGR stability is low. The first reason is that N-GRAM model relies on an active window which implies that recommendations it makes have been computed on short-term observations. Recommendations are computed without taking into account the whole information contained in the sessions. In addition, results show that MOB and SCE are more tolerant to uncertainty resulting from the order of the transactions inside active sessions but they also respect much better the user's preference given by the MFP ratio. It is worth noting that SCE is the most stable one. MOB's stability with WordCup is different from Music Machines and ClarkNet's ones. This is due to the ARS ratio differences. At each step, MOB recommends links among a small set of admissible resources (that is why ARS ratio is low), thus pages are frequently the same. With the WorldCup Soccer dataset, the amount of admissible pages is high (ARS ratio is high), and thus it can be seen that the behavior becomes less stable although still better than NGR's one.

With regard to coverage, SCE has the best KLD. It means that SCE is the recommendation function which loses the less information from the learning database. With its high MFP our algorithm often recommends frequently accessed pages, it is also close to the user's behavior thanks to KLD and eventually, it has rather good recall, precision and coverage.

[4] Cov=coverage, Pre=Precision, Rec=recall, ARS=Average Recommendation Set Size, MFP=Most Frequent Pages (at least $\geq 0.1\%$), KL = Kullback-Leibler divergence.





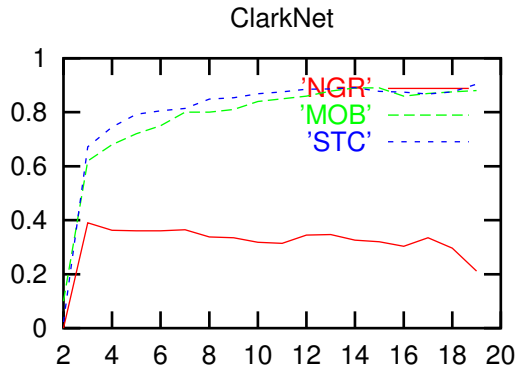


Table 3
Average stability degrees of N-GRAM, MOB and SCE

9. CONCLUSION

The uncertainty about the user navigation behaviors is the main drawback that prevents the deployment of LRS. Results show that the proposed algorithm, SCE as well as NGR are efficient as a ORS. Three new LRS evaluation criteria were also introduced. With respect to these criteria the best results are given by the SCE algorithm.

Working out the following multiobjective problem is a challenge we are currently working on: How to automatically tune the LRS with respect to expected values of the criteria that would be chosen *a priori* by the website designer?

References

[1] Hypertext Transfer Protocol – HTTP/1.1, RFC 2616, June, 1999.

[2] M. Balabanovic. An adaptive web page recommendation service. In W. L. Johnson and B. Hayes-Roth, editors, *Proceedings of the First International Conference on Autonomous Agents (Agents'97)*, pages 378–385, New York, 5–8, 1997. ACM Press.

[3] J. S. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence, Madison, WI*, pages 43–52, July 1998.

[4] R. Cooley, B. Mobasher, and J. Srivastava. Data preparation for mining world wide web browsing patterns. *Knowledge and Information Systems*, 1(1):5–32, 1999.

[5] J.-Y. Delort, B. Bouchon-Meunier, and M. Rifqi. A content-based approach of shift of focus detection in www navigation (under submission), 2003.

[6] M. Deshpande and G. Karypis. Selective markov models for predicting web-pages accesses. In *1st SIAM Data Mining Conference*, 2001.

[7] X. Fu, J. Budzik, and K. J. Hammond. Mining navigation history for recommendation. In *Proceedings of Intelligent User Interfaces, ACM Press*, pages 106–112, 2000.

[8] S. Glassman. A caching relay for the world wide web. *Proceedings of the First International World Wide Web Conference*, pages 69–76, 1994.

[9] D. He and A. Goker. Detecting session boundaries from web user logs. In *Proceedings of of the BCS-IRSG 22nd Annual Colloquium on Information Retrieval Research*, 2000.

[10] J. Konstan and al. Grouplens : Applying collaborative filtering to usenet news. *Communications of the ACM*, 40(3):77–87, 1997.

[11] H. Lieberman. Letizia: An agent that assists web browsing. In *Proceedings 14 th International Conference Artificial intelligence (IJCAI)*, 1995.

[12] B. Mobasher, R. Cooley, and J. Srivastave. Automatic personalization based on web usage mining, technical report tr99010. Technical report, Department of Computer Science, DePaul University, 1999.

[13] J. E. Pitkow and P. Pirolli. Mining longest repeated subsequences to predict world wide web surfing. In *Proceedings of the Second USENIX Symposium on Internet Technologies and Systems*, 1999.

[14] R. R. Sarukkai. Link prediction and path analysis using markov chains. In *Proceedings of the 9th International World Wide Web Conference*, 2000.

[15] G. Shafer. *A mathematical theory of evidence*. Princeton University Press, 1974.

[16] S. Spiekermann and M. Christ. Strategic design of interactive recommender systems. In *Third Berlin Internet Economics Workshop*, May 2000.