RR/LSI-98-4-R

BayesProfile: application of Bayesian Networks in to website user tracking

Ramón Sangüesa
Ulises Cortés

Report LSI–98–4–R

BayesProfile: application of Bayesian Networks to website user tracking

Ramón Sangüesa, Ulises Cortés, *
e-mail:sanguesa.lsi.upc.es, ia.lsi.upc.es
Mario Nicolás
Department of Software
Technical University of Catalonia
Jordi Girona, 1-3, Barcelona 08034
SPAIN
phone: 34-3-401 56 40, 34-3-4025697
FAX: 34-3-401 70 14

Abstract

Detecting the most probable *next* page a user is bound to visit inside a website has important practical consequences: it allows to suggest recommendations to the visitors as to which may be the pages of interest to them in a complex website; it is of help for website designers for deciding how to organize the site contents and it is also useful for pre-caching voluminous objects that the user will very probably need. In sum, it helps to customize web contents. In order to achieve that goal a classification, prediction an evaluation cycle has to be performed. Among the several possible alternative technologies we discuss a real use of Bayesian Network representations. The obtained results are commented, compared to other approaches and its applicability to other domains is also discussed.

Keywords: Learning Agents, Bayesian networks, Sequence detection, Adaptive Hypermedia, User Modeling, User tracking. Bayesian Networks, Data mining, World Wide Web.

1 Introduction

One of the most effective ways for making a website valuable comes from its ability to adapt itself to its users. This is in tune with a more general trend that permeates several

*This work has been supported by project CICYT-TIC960878 of the Spanish Science and Technology Commission fields where information becomes valuable because it is adapted in content or in form to the user's needs and points of view [5]. The burgeoning area of knowledge-based Internet information retrieval agents is an example of these efforts to give the user what he or she is looking for in a way that is closer to its interests and vision. Searching agents do exist that take into account user's preferences when looking for addresses in the net, Webert and Syskill being two of the most notable achievements in this individualization of searching styles [1]. Several other techniques for adapting information and content from material existing on the net or in hypermedia form have been developed in other areas related to the Web, most notably in Intelligent Tutoring Systems [5, 6]. There, there exists a pressing need to adapt contents to user's level of knowledge and competence in a given subject by means of a *user profile* or *user model.* In both cases it is necessary to detect and represent the user's preferences and to make use of this knowledge in order to search for information taylored to this *user profile* as well as to adapt the way in which the information is made available to the user. We think that a similar approach can be followed but taking the initiative not from the point of view of the user, as searching agents [9] do, but from the side of where information lies: that is, the website.

We can consider a wesite as being just a collection of pages connected through links. What is interesting for a web site designer is to be able to fullfill user's requests for information. For our purposes, anything that accesses a web site and navigates through its pages its a user, it doesn't matter whether the detected page accesses are the effect of a human behaviour or are the actions of an automatic information searching or web crawling agent. What is important is to try to get a model of the behaviour of such a web site visitor. Having a model helps in hypothesizing about the true interests revealed by the actions taken by the user. A model is a tool for further analysis and reasoning.

In modelling a user we have to make a choice as to which aspects should be put into the user model in order to make it really useful, compact, efficient and related to the goals for which that model is created in the first place. These are the goals of research in user modelling that also have spawned several interesting ways of detecting commonalities among users, obtaining general descriptions from individual profiles.

We will comment in the following how website visitors could be modelled, which are the alternative formulations for a user model; how this is related to issuing recommendations and how we have developed a whole system that is able to obtain individual and generic user models from actual hits on the pages of a web site. We will show, too, how this can be used to issue simple recommendations to the user.

The paper is organized as follows. Section §2 deals with the problem of web visitor modelling and discusses why Bayesian Networks can be a useful modelling tool. Section §4 evaluates alternative ways of relating user models to web contents, thus identifying possible methods for creating recommendations. Section §3 presents several measures for identifying similar interests from user's behaviour, introducing new ways for defining measuring similarity between actions. Section §4.1 defines a representation for similar groups of users. Section §5 describes the whole WebProfile system and shows some of the results obtained with it. Section §7 finally sums up the whole work, relates it to similar projects and comments on new extensions that are being developed and tested.

2 User models of web visitors

User models are of interest for many tasks but here we are only interested in models that are useful for issuing recommendations. So, the first thing to do is to clarify what a recommendation is.

The goal that we addressed is that of guiding a visitor through the pages of a complex website. A recommendation is meant to bring the attention of the user to a page or group of pages that bears some relationship to the ones that the user has already seen in a single visit or in several visits. A *visit* is a sequence of web page accesses that can be adscribed to an individual user. As such, it is composed of individual URLs that correspond to pages contained in the website. A visit has an initial time, and an initial URL or page and a final time and URL.

Given a visit v and with a time window (t_1, t_2) with $t_1 \leq t_2$ a *partial visit* is any subsequence s_1 such that $s_1 \subset v$, this implies that the corresponding time window of s_1 is also included in (t_1, t_2).

Alternatively, we can also see a visit as a sequence of *transitions*, that is, a sequences of navigation steps from one URL to the following one, which usually is equivalent to following a link that connects *directly* two pages.

The problem of giving a recommendation can be expressed in the following terms.

Definition 2.1
The recommendation problem
Given:

1. *A partial visit v*

2. *A set of pages $p_1 \dots p_n$*

Construct: *A list of pages $\lambda = r_1 \dots r_n$ such that $r_i \in p_1 \dots p_n$ and $\nexists r_i \in \lambda | r_i \in v$ and for each $r_i \in \lambda$ the relation $similar(r_i, l_j, \epsilon)$ holds for any l_j in λ, where $\epsilon \geq 0$*

Now, what does $similar(r_i, l_j)$ mean?. This is a predicate that establishes a *degree of similarity* between two objects. Similarity relationships are defined in many ways [10]. All of them have as a property, that its value reaches a maximum/minimum for identical objects and viceversa for completely incomparable objects. Usually the maximum value is 1 and the minimum is zero, with aproximately indetical objects receiving a similarity value in $[0, 1]$.

In our case, objects are web pages. So the recommendation problem amounts to constructing a list of pages that, given a previous set of pages (the ones appearing in a visit or set of visits) each page in the set has a similarity value that is higher than a preset threshold value, ϵ.

A similarity function can be defined so as to reflect our point of view of which pages are mutually related. Several alternatives are possible here. We will discuss them so as to show how the way the recommendation problem is formulated relates to the final form of the user model needed to give such a recommendation.

2.1 Similarity between pages

One possible way of establishing similarity between pages consists in using a series of common descriptors for describing the pages, tag each page with a *descriptor vector* indicating the level up to which the page reflects each one of the descriptors. Later bu, defining a measure in the space of descriptors, it is possible to assess the similarity between two pages as a measure of the proximity of their corresponding description vectors in the descriptors space. This is the typical approach of classification and clustering theory, see for example [11]. Values for the descriptors can be entered directly by the builder of the website or calculated by inspecting the pages contents or key descriptors (HTML Tags, for example). In the first case, the analysis of the texts appearing in web pages may give a hint as to which are the respective terms that appear with maximum frequency and so give a basis for detecting similar pages. Several ranking methods do exist that are borrowed from text analysis, natural language processing and information retrieval techniques [14, 26]. HTML tags [3] offer a similar basis for establishing similarity between pages.

However, descriptor-based methods seem to fail in one important aspect as is the fact that both of them understands similarity as an *static* property of pages (in general, of content) that is established once and for all at the moment of building the website and linking pages together. Note that in contrast to that, the way a user sees the same group of pages may change in time and it is very probable that, given the same group of pages, any two visitors, when requested to group them by their perceived similarity will propose a different way of grouping the pages. This is a well-known problem in classification that has thoroughly researched cognitive roots [25]. Similarity is a subjective measure, it is, so to speak, in the eye of the viewer, in this case, in the eye of the web visitor.

There are at least two reasons for looking for another way of defining similarity between pages. One is the just mentioned subjective view of similarity that is in accordance with the view of making website contents and recommendations individually taylored to each user. The other reason is of a practical nature. If a list of descriptors has to be defined by the web-designer, tagging of pages have to be mandatory for all pages in the website and be consistently maintained, which is not an easy task, although there is a strong trend towards standardizing web page description [15].

So, we will develop another view of similarity between pages. From our point of view, given that similarity must be referred to each individual user, it has to be defined in other terms. Again several alternatives are possible.

Let us suppose that we have identified a group of pages that are accessed over and over again by a given user. Then,an analyisis on the descriptros or contents of the pages could be done and a similarity degree could be calculated. Again this poses the problem of which descriptors to set aside as the relevant ones for calculating similarity. We would like to use a definition of similarity that introduced the minimum commitment as to which descriptors to use in describing pages.

Let us return to the concept of a visit.

Each visitor in a web site leaves a trace that reflects each one of the visits he/she/it has paid to the site. Log files store such traces. Usually, information in a log file does not allow precise individual identification of a user. However there are techniques to distinguish one user individual hit from another [24]. We say that pages are similar for a user if they are *close* in any visit the user paid to the site, that is if they are in close positions in a sequence. So, similarity becomes a relationship of proximity in a space of visits. This idea is based on an implicit use of tematic grouping as it is done for example in student modelling in intelligent tutoring systems. Closer pages contain also information that is very similar as to their corresponding subjects. Closer pages, then, are also tematically similar.

3 Sequence similarity criteria and detection

The sequence space is formed by all the possible sequences obtained by using the URLs of the pages in a web site. The first for solving the similarity problem will be to devise a set of criteria for identifying similar sequences. There is more than one alternative at that point.. We will introduce some new concepts that will help us in defining that criteria.

Definition 3.1
Sequence base
A finite set of tokens, $b_1 \ldots b_n$ upon which a sequence can be built its call the base of the sequence

Given a sequence base b, of lenght n the number of possible sequences to be built upon it is the number of combinations with repetition that is possible to built on it. Note that in a given website, this number will be usually lower than that theoretical maximum, due to navigation structure constraints.

Given two sequences S_1 and S_2 with respective bases b_1 and b_2 we say that they *share* a sequence base if $b_1 \subset b_2$ or viceversa.

We can start now to set conditions for sequence similarity.

In order for any two visits V_1 and V_2 to be similar they have to share their respective bases. Note that if respective bases for V_1 and V_2 are b_1 and b_2 if $b_1 \cap b_2 = \emptyset$ then we can ensure that V_1 and V_2 have no element in common and so, they cannot have but a similarity of 0.Remember that similarity is taken to be a *graded* relation with values lying in the interval $[0, 1]$. Similarity reaches a minimum when $b_1 \cup b_2 = \emptyset$ and a maximum when $b_1 = b_2$ and *all elements in the sequence appear in the same order.* That is $V_1 = p_5 p_{345} p_{56}$ and $V_2 = p_5 p_{345} p_{56}$ should be assigned the maximum similarity value whereas $V_1 = p_5 p_{345} p_{56}$ and $V_1 = p_5 p_{27} p_{34}$ should be assigned the minimum value, zero. We want to be able to represent intermediate degrees of similarity between sequences. For exaple $p_5 p_{345} p_{56}$ is more similar to $p_5 p_{345} p_{100}$ than $p_5 p_{24} p_{34}$. Let us review the possible criteria for similarity between sequences by taking into account several parameters.

3.1 Sequence similarity criteria

We put forth here some similarity criteria upon whith a similarity function can be built.

of these criteria.We note them S_1, S_2, S_3 and S_4. Each one of them tries to capture some of the possible ways of understanding similarity in an increasing complexity. Strict similarity would imply the same sequence base for any sequence to be measured.

Definition 3.2
Similarity criterion S_1: strict similarity
Given two visits V_1 and V_2 they are said to be similar with degree α, noted $Sim_{S1}(V_1, V_2)$, if for each if V_1 and V_2 share the same sequence base and if $|V_1| = |V_2| = n$ then similarity between V_1 and V_2 is:

$$Sim_{S1}(V_1, V_2) = \frac{\sum_{i=1}^{n} f(v_{1i}, v_{2i})}{n} = \alpha$$

where $f(v_{1i}, v_{2i})$ is 1 if $v_{1i} = v_{2i}$, that is if both pages have the same URL and 0 otherwise.

Clearly similarity is maximum when pages in the same position in both sequences are the same. We can relax this very strict criterion by forgetting about some of the elements of the sequence. The following criterion says that from a given point in the sequence both sequences are equal.

Definition 3.3
Similarity criterion S_2: sharing partial sequence order
Given two visits V_1 and V_2 they are said to be similar with degree α, noted $Sim_{S2}(V_1, V_2)$, if for each $v_{1i} \in V_1$, $v_{1i} \geq k$, $k > 1$ and there exists a v_{2j}, $1 \leq j \leq n$, $v_{2j} \in V_2$ such that from v_2j to the end of sequence V_2, $v_i1 = v_2j$. If the number of visits in V_1 exhibiting such propery is m and the total number of elements in V_1 is n then $Sim_{S2}(V_1, V_2) = \frac{m}{n} = \alpha$

This criterion is useful to eliminate information that is not very relevant. For example, the first pages in a webpage are very general pages with non very specific information. We can relax even more this characteristic by allowing that from a given point of the first sequence the same pages appear in the same order in the second sequence but this time without worrying about the number of intermediate pages in between. For example, if V_1 is $p_1p_2p_3$ and V_2 is $\mathrm{p}_1p_{27}p_{34}p_4\mathrm{p}_2p_{547}\mathrm{p}_2$ then similarity between V_1 and V_2 should be higher then, for example, V_1 and $V3$, $V3$ being $V3 = \mathrm{p}_1p_{27}p_{727}p_{90}p_{23}p_{28}p_{2745}p_{35}p_4\mathrm{p}_2p_{547}\mathrm{p}_3$

Definition 3.4
Similarity criterion S_3: approximate order
Given two visits V_1 and V_2 they are said to be similar with degree α, noted $Sim_{S3}(V_1, V_2)$, if for each $v_{i1}, v_{i+1} \in V_1$, if the number of intermediate pages in V_2 between pages v_i and v_{i+1} is $n(i1, i+1)$ then

$$\alpha = \frac{\sum_{i=1}^{m_1-1} n(v_i, v_{i+1})}{m_2}$$

where $m_1 = |V_1|$ and $m_2 = |V_2|$

This criterion is useful for comparing important pages in visits. It is interesting for grouping visits that go through these pages no matter which other pages they have to go in between. The last criterion is just a combination of the first and the third ones.

Definition 3.5
Similarity criterion S_4: Hybrid criterion
Given two visits V_1 and V_2 the degree of similarity α, noted $Sim_{S4}(V_1, V_2)$, is said to be:

$$Sim_{S4}(V_1, V_2) = \frac{Sim_{S1}(V_1, V_2) + Sim_{S3}(V_1, V_2)}{2}$$

Now that we are able to establish a similarity among visits (sequences of pages) we have a tool for grouping together visits that refer to a similar subset of pages, that is we are able to group together types of users.

4 Identifying common traits in visitors

Clustering methods exist for, upon the definition of a distance or similarity measure, grouping together objects that share same characteristics. Our problem is to build *aggregations of user visits*. User visits have been formalized as sequences built on a set of possible pages, a visit's sequence base.

As our goal is to identify, after some hits in a visit, which type of user accessing a website it is important to devise à methods that is able to cluster in the same class visitors that are interested in the same types of pages, i.e., clustering together visitors that access *similar* pages. We want to find a common description for users that access similar pages. Given several visits of the same user a *common description* of these visits will result in an *individual user profile*. By finding common traits of the visits of several users on a group of pages we will be able to describe a *generic user profile* for that group of pages.

In both cases it is important to aggregate similar objects into classes. Here similar objects are similar users and similarity is stablished by using information about visits.

In this type of clustering algorithms, it is important to devise a useful representation for eacb *class*. A class can be described simply by enumeration of all the objects that belong to it or by resorting to a simple representation that summarizes the common characteristics of the elements of the class.

We want to have a representation of a class that could be useful in two aspects: on one hand it has to help in deciding, during the clustering process to which class an object (i.e. visit) belong to and, more important, the information contained in the class representation [1] should be of help for the two important operations not related directly to classification but to the *practical use* of this classification, namely *prediction* of the next page a visitor may access and *recommendation* of new pages that may be of interest to him.

Usuallly prototypes are represented as descriptor vectors [8, 2], probabilistic descriptor vectors [7], decision trees [13] or rules [11].

We propose another way of representing classes that retains the sequential nature of the clustered objects, that is, visits. Given a class of visits we need to represent the structure of the most common followed paths, that is subsequences of visits. It is important to see that, given the similarity criteria defined in the previous section, a prototype is an entity

[1]what is usually called a *prototype* and we have called a *generic user profile*

that reflects *trends* in the sequences so, for similar reasons to the ones that advised the use of a probabilistic description of classes in other systems, as, for example **AUTOCLASS**, a probabilistic representation of sequences is also advisable here. In order to do so, we have introduced a new representation.

A previous definition will be of help here.

Definition 4.1
Access relationship
Given a set of sequences V^* on a sequence base V, we can define the access relationship, $\mathcal{A}$ between elements of the sequence base as follows:

1. $\mathcal{A} \subseteq \mathcal{V} \times \mathcal{V} \times [\prime, \infty]$

2. *Given $v_1, v_2 \in V, \alpha \in \mathcal{R}$, $(v_1, v_2, \alpha) \in \mathcal{A}$ if $(v_1, v_2) \in V^*$ and $\alpha = freq(v_1, v_2)$ where $freq(v_1, v_2)$ is hte normalized frequency of occurence of transition (v_1, v_2) in the set of sequences V*

The frequency of occurrence of a transition $freq(v_1, v_2)$ is the frequency of occurrece of v_2 *given that* the previous element in the same sequence is v_1. So $freq(v_i, v_j)$ is an estimation of the *conditional probability*, $P(v_j|v_i)$.

Now we can define the structure that reflects this relationship.

Definition 4.2
Trajectory tree
A trajectory tree for and accesibility relation $\mathcal{A}$ defined on a sequence base V, $T(V, \mathcal{R})$ is a directed acyclic graph such that:

1. *If $v_i \in V$ then there exists a node $v_i \in T(V, \mathcal{R})$*

2. *If $v_1, v_2 \in V, \alpha \in \mathcal{R}$ and $(v_1, v_2, \alpha) \in \mathcal{R}$, then there exists a directed link (v_i, v_j) in $T(V, \mathcal{R})$.*

Note that, as defined, a trajectory tree is nothing else but a *Bayesian Network* [12]. Bayesian Networks have very interesting properties that make them useful for prediction and diagnosis. Given such a representation we can answer queries of the type "Given that the user is in page345, which is the probability of him going to $p45$, "Given that the use is in page345, which is the probability that he visisted previously p89?" and, more importantly, "Do a user that visits pages 345 and 346 usually go thorugh p89 to reach pages 567 and 456?". This is a powerful representation in the sense that allows quite detailed and refined responses to complex queries.

There is a strong assumption here which is the one implied by this transformation: that the sequences can be represented by a regular grammar. See [24] for a discussion and also the comments in section §7.

Construction of a trajectory tree for a class of visits can be done by applying standard Bayesian Network construction algorithms. We use our own algorithm, **POSSCAUSE**, [20, 16, 19, 17, 22] that allows the efficient construction of several types of belief networks, ensuring that the resulting network is the most constrained one [2] and the most informative one [3].. See Chapters 5 and 6 in [16] for the details.

At this point, we have defined criteria for establishing similarity between sequences, so that we are able to group similar sequences together, and we have also put forth a representation that summarizes the prototypical sequence structure from a group of visits.

Note that a trajectory tree can be used as a very fine-grained user model at least in two different forms: as individual user model or as a description of a group of similar users. Note also that this user model is more powerful than a single belief vector description [4] in the sense that it allows performing powerful inferences on it, the same type of inferences a Bayesian Network allows. It is also important to remark that it can have the same reasoning power as a rule representation of classes but with a more compact representation.

Let us see first how a class can be built.

4.1 Building a classification based on sequences

The classification algorithm works with a previous transformation of information contained in the website logfiles. First of all a time window is used in order to identify possible accesses by the same user. Then, URLs are transformed into number sequences in such a way that the website root URL is assigned the first number and then the subsequent adresses are assigned succesive numbers.

From the log files a set of visits V is extracted. Each one of the member of V is a visit V_i consisting in $v_1 \dots v_n$ accesses to URLs.

The process for building a classification is very similar to the nearest neighbour algorithm [8], the difference here is that each object is in fact an structured object, a visit.

Each time a new visit V_i is taken into consideration it has to be assigned a class. Let us suppose that at a given moment the existing classes are, $C_1 \dots C_m$. In order to include V_i into a class, a comparison of its similarity with the members of each class has to be performed. V_i will be included as a member of the class C_max that exhibited the highest similarity.

The idea is to start with a single class consisting in the first visit found. Note that when the similarity between the elements of existing classes and any new incoming visit V_j is too low, the newcomer should create its own class, having itself as the only member. The process repeats until all visits have been processed. There are many possible variations. For example, when introducing a new visit induces a recalculation of several classes because members of one class "fall in the orbit" of other class. Note also that presently a comparison with all the members of the class is done. It would also be possible to compare a single visit against the trajectory tree of each class. This is something we are aiming at.

Now that a set of classes is created it is possible to issue a recommendation.

[2]the one with highest degree of association between variables

[3]the one whose underlying joint uncertainty distribution is closest to the one implied by the data

```
Input: A similarity function $S$;
  a similarity threshold $\epsilon$;
  a set of user visits $\mathcal{V} = V_1 \ldots V_k$
Output: A collection of classes $\mathcal{C} = C_1 \ldots C_m$, each one of them represented by a trajectory tree $\mathcal{T}_{C_m}$
  Let $\mathcal{C} = \emptyset$
  for each visit $V_i$ in $\mathcal{V}$ do
    if No $C_i \in \mathcal{C}$ such that $S'(c_i, V_i) \geq \epsilon$ then
      $\mathcal{C} = \mathcal{C} \cup V_i$
    else
      $C_i = C_i \cup V_i$
    end if
  end for
  for each resulting class $C_i \in \mathcal{C}$ do
    calculate its trajectory tree $\mathcal{T}_{C_i}$
  end for
```

Algorithm 1: The sequence classification algorithm

5 How BayesProfile works

We comment here very briefly the tasks that **BayesProfile** has to perform in order to issue a recommendation.

First we describe how a recommendation is constructed. Remember from §2.1 that a recommendation is a list of pages that are very similar to the ones appearing on another set. The idea is that by analysing the set of pages being visited by the user a selection of very probable classes is done from the set of classes $\mathcal{C}$. Once a subset of probable classes is identified, let us call it $\mathcal{C}'$ then the most probable *next* page in each $C_i \in \mathcal{C}'$ is selected. This is done by finding the highest conditional probability in the trajectory tree corresponding to C_i, $\mathcal{T}_{C_i}$ given that the user has last visited page was p_k. That is we indentify the link in each tree with the highest probability. In that way, a set of recommended pages is built, $\mathcal{R}$.

So the tasks to be done are the following ones:

1. Track a user accessing the website, registering the URLs of the pages he/she/it visits
2. Identify the most probable class, given the pages he/she/it has visited
3. Output identify the pages that the user will visit next with the highest probability

Note that the user tracking tasks can be used also to update conditional probability tables for each of the classes that contain connections between the pages he/she/it is visiting.

The process followed for recommendation is the one corresponding to algorithm§r̃efalgrec.

```
Input: A list of visited pages $\lambda$;
  A list of classes $\mathcal{C} = C_1 \dots C_n$ with trajectory trees, $\mathcal{T}_{C_1} \dots \mathcal{T}_{C_n}$
Output: A recommendation $\rho = \{p_1 \dots p_n\}$ where each $p_i$ is the highest probability page
  in $C_i$ given that visited pages are the ones appearing in $\lambda$.
  Let $\rho = \emptyset$
  Find $\mathcal{C}'$ the most probable classes given that visited pages are the ones in $\lambda$
  for each $C_i$ in $\mathcal{C}'$ do
    Find the page $p_{ij}$ such that $P(p_{ij}|\lambda)$ is maximum in $\mathcal{T}_{C_i}$
    Let $\rho = rhho \cup p_{ij}$
  end for
```

Algorithm 2: Recommendation algorithm

The several tasks have been organized as a reduced subset of cooperating agents: one for user tracking, one for probability updating, one for class identification and one for recommendation construction. See details of their workings in [18].

6 Results

BayesProfile has been tested on a commercial web site containing approximately 45000 pages corresponding to 967 website entries (homepages) It has also been used in our Department site for comparison with WebProfile, a previous recommendation algorithm, **WebProfile** that used **AUTOCLASS** for clustering analogous webpages, see [23].

In Table §6 a comparison about the number of classes obtained on the departmental website by **WebProfile** and **BayesProfile** can be seen. Note that in this previous system, the clustering of users is done by using and unstructured information, that is, individual page accesses, unrelated to other accesses by the same user. The idea of a visit or a trajectory is missing in **WebProfile**.

Table 1: Clustering behaviour: departamental website data

Criterion	**Number of classes**
Criterion 1	49
Criterion 2	43
Criterion 3	27
Criterion 4	33
WebProfile	1

As it can be deduced from the table the previous system tends to produce less specific clusters of pages. We are now making measurements about the predicting ability of **WebProfile** in comparison to **BayesProfile**.

Results of applying **BayesProfile** on data coming from a commercial website are shown in table§řefcommercialfor several different similarity criteria. It can be seen that results

Table 2: **BayesProfile** behaviour: commercial website log data

Criterion	**Number of classes**
Criterion 1	181
Criterion 2	133
Criterion 3	146
Criterion 4	96
WebProfile	1

vary widely depending on the strictness of the criteria. The more strict the criteria for similarity is, the greater the number of classes with fewer elements. In can be seen that with higher similarity thresholds, classes with only one member abound.

Table 3: Intra- and Interclass similarity: commercial website data

Criterion	**Intraclass similarity**	**Interclass similarity**
Criterion 1	0.66	0.20
Criterion 2	0.45	0.50
Criterion 3	0.87	0.15
Criterion 4	0.43	0.50

Intraclass and interclass similarity measures where also made by measuring the mean similarity between members of the same class and between prototypes of differents classes. A good classification algorithm should show a high intraclass similarity and a low interclass similarity. In both measures the different similarity used seem to perform well.

7 Discussion and future work

The problem of issuing recommedantions for helping website navigation has been cast in terms of the identification of the most probable page(s) a user may visit next *given that* he/she/it has visited a set of pages in the website, λ.

In order to identify this set of probable pages, similarity concepts have been introduced. Probable pages are those, that usually are visited in conjunction. So the notion of *visit* has been introduced in order to compare the sequential structure of several visits.

In order to avoid maintaining a separate *user model* for each user of the website, an aggregation method is used in order to create classes of similar users, or, *generic user models.* This has required the creation of several criterions for similarity between sequences.

The recommendation task amounts then to the identifying of the most probable generic user model(s) *given that* the user has visited a set of pages λ.

A generic use model is represented by means of a *trajectory tree.* So after identifying which are the more probable trees for a given visit, the next most probable pages for each tree is identified.

Results of classifications seem to indicate that the four criteria give good classifications. We want to establish experiments in order to snhow that the **BayesProfile** procedure does in fact return more predictive models. Moreover, it seems that classification quality is better than the one obtained by using clustering algorithms based on individual accesses instead of structured user visits, as it happened in a previous system **WebProfile**.

Currently **BayesProfile** is being used in two test environments. It will be also used as an alternative user modelling method in an architecture for Intelligent Tutoring Systems that we are currently building [21].

References

[1] M. Ackerman, D. Billsus, S. Graffney, S. Hettich, G. Khoo, J.K. Dong, R. Kleftstadt, C. Lowe, A. Ludeman, J. Muramatsu, K. Omori, M. Pazzani, D. Semler, P. Yap, and B. Starr. Learning probabilistic user profiles. *AI magazine*, 18(2):47–56, 1997.

[2] J. Béjar and U. Cortés. Linneo: una herramienta para la adquisición de conocimientos y generación de reglas de clasificación en dominios poco estructurados. In *Proceedings IBERAMIA-92*, Cuba, 1992.

[3] T. Berners-Lee and D. Connolly. Hypertext markup language. Internet Working Draft 5, November 1993.

[4] D. Billsus and M. Pazzani. Learning probabilistic user models. In *Proceedings of the Sixth International Conference on User Modelling*, Chia Laguna, Sardinia, Italy, 1996.

[5] P. Brusilovsky. Integrating hypermedia and intelligent tutoring systems. In *Proceedings of AIED-95*, 1995.

[6] P. Brusilovsky, P. Schwarz, and G. Weber. Elm-art: An intelligent tutoring system on world wide web. In *Proceedings of Intelligent Tutoring Systems-96*, pages 261–269, 1996.

[7] P. Cheeseman, J. Kelly, M. Self, J. Stutz, W. Taylor, and D. Freeman. Autoclass: A bayesian classification system. In *Proceedings of the Fifth International Conference on Machine Learning*, pages 51–61, 1988.

[8] T.M. Cover and P.E. Hart. Nearest neighbor pattern classification. *Information Theory*, 14(5):515–516, May 1968.

[9] P. Edwards. Exploiting learning technologies for world wide web agents. In *IEE Colloquium on Intelligent World Wide Web Agents*, 1997.

[10] J.G. Klir and T.A. Folger. *Fuzzy sets, Uncertainty and Information.* Prentice-Hall, Inc., 1988. (Chapter 3).

[11] R.S. Michalski. *A theory and methodology of inductive learning*, pages 83–134. Machine Learning: An Artificial Intelligence Approach. Morgan Kaufmann, 1983.

[12] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann, Los Altos, CA,USA, 1988.

[13] J.R. Quinlan. Induction of decision trees. *Machine Learning*, 1:81–106, 1986.

[14] G. Salton and M.J. McGill. *Introduction to Modern Information Retrieval.* McGraw-Hill, Inc., New York, NY, USA, 1983.

[15] R. Sangüesa. Learning possibilisstic causal networks from data. 1997.

[16] R. Sangüesa. *Learning Possibilistic Causal Networks from Data.* PhD thesis, Department of Software, Technical University of Catalonia (UPC), 1997.

[17] R. Sangüesa and U. Cortés. A parallel algorithm for building possibilistic causal networks. *International Journal of Approximate Reasoning*, 1988. (in press).

[18] R. Sangüesa and U. Cortés. The bayesian agent: an incremental approach for learning agents working under uncertainty. VIM Project Winter Seminar, Cooperation between artificial and human societies, Ravello (Italy), November 1996.

[19] R. Sangüesa and U. Cortés. Learning causal networks from data: a survey and a new algorithm for learning possibilistic causal networks from data. *AI Communications*, 4(19):1–31, 1997.

[20] R. Sangüesa, U. Cortés, and J. Béjar. Causal dependency discovery with posscause: an application to wastewater treatment plants. In *Proceedings of the First International Conference on the Practical Application of Knowledge Discovery and Data Mining*, Westminster Central Hall, London, 1997.

[21] R. Sangüesa, U. Cortés, J. Béjar, and T.P. Hall. Integrating learning tools by means of cooperative agents technology. In *ENABLE'97 Conference: enabling network-based learning*, Espoo-Vant "aa, Finland, 1997.

[22] R. Sangüesa, U. Cortés, and J. Cabós. Possibilistic conditional independence: a similarity-based measure and its application to causal network learning. *International Journal of Approximate Reasoning*, 1988. (in press, scheduled for the second issue of the year).

[23] S. Saxon. Webprofile. Master's thesis, Computer Science Department, University of Aberdeen and Department of Software, Technical University of Catalonia (UPC), 1997.

[24] H. Toivonen. *Discovery of frequent pattenrs in large data sets.* PhD thesis, Department of Computer Science, University of Helsinki, 1996.

[25] S. Vosniadou and A. Ortony. *Similarity and analogical reasoning.* Cambridge University Press, 1989.

[26] B. Yuwono and D.L. Lee. Search and ranking algorithms for locating resources on the world wide web. Technical report, Department of Computer Science, Hong Kong University of Science and Technology, 1996.

Departament de Llenguatges i Sistemes Informàtics
Universitat Politècnica de Catalunya

Research Reports – 1998

LSI-98-1-R	"Optimal Sampling Strategies in Quicksort and Quickselect", Conrado Martinez, Salvador Roura.
LSI-98-2-R	"Query, PACS and simple-PAC Learning", J. Castro and D. Guijarro.
LSI-98-3-R	"Interval Analysis Applied to Constraint Feasibility in Geometric Constraint Solving", R. Joan-Arinyo and N. Mata.
LSI-98-4-R	"BayesProfile: application of Bayesian Networks to website user tracking", Ramón Sangüesa and Ulises Cortés.

Hardcopies of reports can be ordered from:

Nuria Sánchez
Departament de Llenguatges i Sistemes Informàtics
Universitat Politècnica de Catalunya
Campus Nord, Mòdul C6
Jordi Girona Salgado, 1–3
08034 Barcelona, Spain
`secrelsi@lsi.upc.es`

See also the Department WWW pages, `http://www-lsi.upc.es/`