A logical framework for intelligent profiling in multicriteria web pages recommendations

Alessandra Mileo

Abstract.
The growth of the World Wide Web emphasized the need of providing some tools in charge of helping users in finding information on the Web, by determining which web pages on a particular topic would be interesting for them. Development of such tools is strictly related to categorization and user profiling. We want to focus on how user interests can be summarized and expressed in a profile, and how this profile can be specified to support a user browsing the Web.

Our approach is related to learning and it is based on representing and updating the profile through Answer Set Programming. The intuition is that of providing users with a framework to set and control the multicriteria recommendation policy followed by the system, through a combination of existing techniques, namely the application of similarity functions, clustering algorithms and utility functions associating weights to pages.

1 Introduction
The World Wide Web (WWW) is constituted by a large, distributed set of heterogeneous documents that is constantly changing. Almost anyone provided with an Internet connection may put documents on the Web, making it harder for a visitor to locate interesting pages exploring all possible paths where pertinent information can be found. This problem has been addressed by developing different methods and tools among which probabilistic approaches [6] and techniques for data analysis (data mining [4], collaborative filtering [5], clustering, learning).

Considering learning-based approaches to user profiling, some solutions have been proposed so far: Lieberman (1995) implemented the user interface agent Letizia, based on tracing past browsing behavior of the user to anticipate his/her browsing activity and provide suggestions on interesting links [7]; Mladenić, (1999) developed Personal Web Watcher, a system that uses machine learning techniques to deduce user interests and suggest interesting links on the requested web documents [8]; Paazani and Billsus (1997) proposed an algorithm for learning and revising profiles based on machine learning approach [9] where interestingness of unseen sites was determined on the basis of long-term information goals and statistical rating on previously visited sites provided by the user himself. These approaches were strictly domain-dependent and did not give the user any control in how data analysis techniques were applied to obtain recommendations.

We want to provide users with a tool to express such a control by the extraction of a profile of interests, and the specification and enforcement of recommendation policies.

In our proposal, a user profile can be seen as containing a set of user-specific interests (partially set by the user and partially learned by the system) on a set of heterogeneous topics, and a set of preferences on how to combine recommendation criteria. The main concept we based this work on is that of intelligent user profiling, i.e. automatic revision of user’s interests and preferences enforcement in page ranking, through an inference engine analyzing user’s browsing behavior.

The framework relies on continuance and coherence of a search/browsing session: a user who randomly browses the Web would not get any significant benefit from intelligent suggestions proposed by the system.

With this proposal we want to give an idea of how a declarative approach to user profile specification and maintenance could be a valid way to allow users to control and combine existing recommendation methods very intuitively.

Section 2 of this paper introduces the general principles and main features of the framework proposed, exploring details of the architecture components and how they relate each other. Section 3 presents our conclusions and hints for future developments.

2 Adaptive Logic Browsing Adviser (ALBA)
ALBA combines some of the advantages of previous learning-based attempts to profile composition and update. Differently from previous approaches, in this framework the user profile is related both to user’s browsing behavior and to a certain amount of initial information related to what user likes/dislikes, implicitly provided by the user himself. These additional details about user interests are context-independent and can be extracted almost automatically, so that a user is not forced to provide statistical information or select crucial features. A user profile also contains user’s preferences on how to apply recommendation criteria.

The formal logical framework for profile specification and maintenance is that of Answer Set Programming (ASP). ASP is based on the stable models semantics for Logic Programs proposed by Gelfond and Lifschitz [3, 2] and it can be seen as bringing together concepts and results from Logic Programming, Default Reasoning and Deductive Databases.

ALBA uses automated commonsense (non-monotonic) reasoning to implement the agent component in charge of reasoning on a dynamic profile. With the term dynamic we refer to the fact that a user profile should change and be updated

1 Dipartimento di Informatica e Comunicazione, Università degli Studi di Milano. Email: mileo@dico.unimi.it

2 The term browsing session is related to the browsing activities in between two subsequent requests of suggestions.

3 In ASP, a logic program is given a meaning in terms of alternative models that are i) compatible with the rules of the program and ii) truth-minimal.
according to different/new user’s preferences that may be discovered “on the way”. Such preferences can be deduced by user’s browsing behavior, i.e. user’s previous choices related to the browsing activity. As a consequence, profile learning mechanism is to be made dynamic too. To achieve this property, some of the parameters (thresholds) used in the inference process, are to be updated, as shown in Section 2.3.

In the following subsections we analyze the main components of ALBA Framework, and identify their interactions with the user during browsing activity.

A complete schema of ALBA is shown in Figure 1.

2.1 User’s Initial Profile

A user is supposed to provide an initial set of preferences that can be classified into preferences on i) sources and ii) reasons, stored in a local file which is loaded as soon as the system is run.

Preferences on sources are represented by a set of pages (in form of URL addresses) considered interesting by the user. These pages can be a subset of all pages that are included among the favorites links in the browser menu. Preferences on sources are written as a set $F$ of $m$ facts of the form:

$$F = \{ \text{fav}(\text{Url}_1), \ldots, \text{fav}(\text{Url}_m) \}$$

Facts in $F$ are part of the basic knowledge (Knowledge Base or KB) the system has about the user.

When a user asks ALBA for suggestions, the initial profile is enriched with new predicates and facts derived from user’s browsing behavior as detailed in Section 2.2, so that the profile is constantly updated according to evolution and specification of user’s interests during a browsing session.

Preferences on reasons can be defined by the user in form of ordering relations on the criteria according to which a page can be recommended by ALBA.

We associate each reason for recommendation to a function $f_i(P, P_j)^4$ representing a relation among the content of the current page the user is visiting, $P$, and the set $P_{\text{link}}$ of contents of each page linked by $P$, namely $P_1, \ldots, P_j$. Function $f_i$ can be formally defined as follows:

**Definition 1** $f_i : P \times P_{\text{link}} \rightarrow R$

In this first proposal, three reasons to recommend a link are considered: the first one refers to the application of a generic similarity function, the second one expresses how pages are correlated with interesting pages as a result of a clustering process, and the last one combines both criteria and is centered on utility evaluation. These set of methods is not exhaustive and can be extended to consider other interesting approaches to page ranking.

A user can specify an order among this functions by dragging and dropping their indicator names into ordered slots. As a result, each function $f_i$ is associated to a value $p(f_i) \in N$ that will be used by the Adviser component as detailed in Section 2.4.

Preferences on both sources and reasons are merged in a current search profile instance as soon as ALBA is run.

2.2 Log file filtering

Whenever the user asks ALBA for recommendations on interesting links of a page, a rule extractor analyzes information contained in the log file and extracts the meaningful ones to update the profile before ranking pages.

Which actions should be considered meaningful, and consequently used to update the profile, is a crucial issue in this context. Previously developed systems such as Letizia [7] did not have any filter: any of the user’s actions were considered important and used for profile updating. Anyway, a user may perform several kind of actions mostly irrelevant in determining user’s interests in a specific topic, and it is not always trivial to determine which of the user’s activities are related to a particular interest or not. That’s the reason why we decided to select a reduced set of actions among those traced by our logger tool$^5$. Such actions are then automatically rewritten as logic predicates and passed to the Learner for the inference process.

The filtering mechanism considers meaningful the following actions (this set can be extended):

i. visiting an URL with a frequency $N$ of times and with an average dwell time $D$;
ii. adding/removing an URL to/from the list of favorites at a given time $T$;
iii. following/not following a suggestion proposed by ALBA with a frequency $N$, respectively clicking/not clicking on link URL.

Such actions are represented by the following predicates, respectively:

i. \text{vis}(\text{Url}, \text{N}, \text{D})
ii. \text{add}(\text{Url}, \text{Time}) / \text{rem}(\text{Url}, \text{Time})
iii. \text{foll}(\text{Url}, \text{N}) / \text{disc}(\text{Url}, \text{N})

Any other action is filtered out.

---

$^4$ Besides that, each reason is also associated to a name and to a short description the user can refer to.

$^5$ There are several freeware and shareware tools for activity logging that may be found at http://www.sharewareconnection.com/titles/activity-logging.htm.
We call \( F \) the complete set of predicates representing meaningful actions extracted from the log file. The initial set of facts \( F_{\text{log}} \) representing the profile, is thus extended to

\[
F_{\text{log}} = F \cup A.
\]

### 2.3 Learner component

The Learner component is in charge of:

- **evaluating** \( F_{\text{log}} \) through the ASP solver Smodels;
- **updating the profile** by using information from the log file, filtered by the rule extractor;
- **updating threshold parameters** in deduction rules on the basis of user’s response to suggestions provided by applying some heuristics;

Parameters update makes the framework adaptive in the sense that it allows to adapt the learning process to different users as well as to a user that changes or specifies his/her topic of interests from one search session to the other.

When boolean preferences\(^6\) are specified, a page is classified in the profile to be either interesting or uninteresting. The page located by \( \text{Url}_i \) is in the list of favorites if \( \text{Url}_i \in F \), or if the log file traced that it has been added to the list of favorites during browsing; in the latter case, \( \text{fav}(\text{Url}_i) \) is inferred by applying a rule of the form:

\[
\text{fav}(\text{Url}_i) :- \text{add}((\text{Url}_i, T), \text{not \ rem}(\text{Url}_i, T1), T1 > T). \quad (1)
\]

which represents the fact that link \( \text{Url}_i \) is in the list of favorite for the current session if the log file reveals it has been added to favorites at time \( T \) and not subsequently removed\(^7\).

All facts in the extended profile \( F_{\text{log}} \), together with rule (1), are used to infer if a page is to be considered interesting, by applying a default rule of the form:

\[
\text{int}(\text{Url}) := \text{not n_int(Url), fav(Url)}. \quad (2)
\]

where not is to be intended as negation as failure. The fact that a page is considered interesting can be inferred also according to user’s behavior as follows:

i) if link \( \text{Url}_i \) has a frequency \( N \) of visits with an average dwell time \( D \) greater than a given minimum \( \epsilon \), or

ii) if the user reached \( \text{Url}_i \) by following a recommendation from ALBA with frequency \( N \) greater than a given minimum \( \text{Min}_i \).

then the page located by \( \text{Url}_i \) is supposed to be interesting to the user.

These reasons are represented in the Learner by the following rules, respectively:

\[
\begin{align*}
\text{int}(\text{Url}_i) := & \text{not n_int(Url)_i}, \text{vis}(\text{Url}_i, N, D), D > \epsilon. \\
\text{int}(\text{Url}_i) := & \text{not n_int(Url)_i}, \text{fav}(\text{Url}_i), N > \text{Min}_i.
\end{align*}
\]

Rules (1) and (2) above indicate that, by default, all pages located by links that are in the list of favorites are interesting. This holds unless uninterestingness of a page is inferred by user’s browsing behavior. When a page is inferred to be uninteresting, the default mechanism allows us to detect an exception to default by applying the rule:

\[
\text{n_int}(\text{Url}_i) :- \text{unint}(\text{Url}_i). \quad (3)
\]

The predicate \( \text{unint}(\text{Url}_i) \) can be derived in three cases:

i) \( \text{Url}_i \) has a frequency \( N \) of visits, with an average dwell time \( D \) lower than a given minimum \( \epsilon \);

ii) \( \text{Url}_i \) has been suggested by ALBA and the user discarded the recommendation with a frequency \( N \) greater than a given minimum \( \text{Min}_i \);

iii) \( \text{Url}_i \) has been removed by the list of favorites at time \( T \) and not added again later.

The correspondent rules in the Learner are as follows:

\[
\begin{align*}
\text{unint}(\text{Url}_i) := & \text{vis}(\text{Url}_i, N, D), D < \epsilon. \\
\text{unint}(\text{Url}_i) := & \text{disc}(\text{Url}_i, N), N > \text{Min}_i. \\
\text{unint}(\text{Url}_i) := & \text{rem}(\text{Url}_i, T), \text{not add}(\text{Url}_i, T1), T1 > T.
\end{align*}
\]

In addition, a consistency constraint of the form:

\[
:- \text{int}(\text{Url}_i), \text{not n_int}(\text{Url}_i). \quad (4)
\]

is added to assure that a page cannot be considered both interesting and uninteresting at the same time.

The Learner component uses the solver Smodels\(^8\) to compute a solution \( S \). This solution is a set of predicates where for each link \( \text{Url}_i \) either the predicate \( \text{int}(\text{Url}_i) \) or the predicate \( \text{n_int}(\text{Url}_i) \) holds, and it is used to update the initial profile accordingly for the following request of suggestions.

Maintenance and use of such a simple profile of interests is expected to allow more precise profiling compared to previous AI-based approaches mentioned so far, where either the unfiltered browsing activity or user profile specification was used.

Adaptivity of the system also lays in the Learner ability of changing thresholds on the basis of user’s responses (positive or negative) to recommendations proposed by ALBA. Values for parameters \( \epsilon \) (expressed in milliseconds), \( \text{Min}_i \) (integer) and \( \text{Min}_i \) (integer) are initially set by the system and then revised one time for each browsing session.

Parameters are updated according to some simple heuristics, among which:

- the less the user stays on a web page before accepting it, the smaller \( \epsilon \) should be;
- the more often the user discard a suggested page, the greater \( \text{Min}_i \) should be;
- ...

### 2.4 Adviser component

The Adviser component adds suggestions on links of a page the user asked recommendations for. This process consists of four phases that will be detailed below. We refer to \( \text{Url}_i \) as to an URL, while \( P(\text{Url}_i) \) refers to the content of the page located by \( \text{Url}_i \). \( P \) refers to the content of page \( P \), and \( \text{Url}(P) \) is the URL locating page \( P \).

\(^6\) In this preliminary prototype of the system, we only consider boolean preferences, but all general principles apply also to ranked preferences.

\(^7\) Information in the log file are deleted as soon as they have been used by the previous call to ALBA. So, if we name \( T' \) the instant in which ALBA loaded log file info the last time, possible values for \( T \) are such that \( T > T' \).

\(^8\) http://www.tcs.hut.fi/Software/smodels/
Clustering:

\( Url_i \) represents URLs considered interesting to the user; \( Url_i \) together with page \( P \), user asked suggestions for, and the \( P_j \) pages pointed by \( P \) at the first level of depth, are clusterized on the basis of a similarity function (e.g. cosine similarity) whose value, \( s(P, P_j) \), is stored in a table; a boolean function \( cl(P, P_j) \) indicates that pages \( P_i \) and \( P_j \) are in the same cluster.

Page Ranking:

current page the user is visiting \( (P) \), as well as all pages \( P_i \) the user can potentially browse from there, are given a weight \( W_j \). Let us consider two sets:

\[
WC_j = \{ URL_i \mid P(UR_l) \text{ is in the same cluster as } P_j \text{ and } int(UR_l) \in F_{log} \}
\]

\[
W_I = \{ URL_i \mid (P(UR_l) \text{ contains } URL(P_i) \text{ or } \text{and int(UR_l) } \in F_{log} \}
\]

\( WC_j \) and \( W_I \) represents two orthogonal evaluation of the interestingness of a page \( P_i \), w.r.t. the general set of user’s interests \( int(UR_l) \). \( W_j \) is now expressed by the sum of the cardinality of the two sets defined above:

\[
W_j = |WC_j| + |W_I|
\]

Computing similarities:

results of the precedent phases are used to compute correlation functions according to which suggestions are added to the original page \( P \), before returning it to the user.

In our first idea, there are three correlation functions representing reasons for recommendation:

1. \( f_1(P, P_j) \) is related to the similarity between the two pages, as returned by the clustering process:

\[
f_1(P, P_j) = \begin{cases} 
  s(P, P_j) & \text{if } cl(P, P_j) \\
  0 & \text{otherwise}
\end{cases}
\]

2. \( f_2(P, P_j) \) combines results of the clustering process and interestingness of the page:

\[
f_2(P, P_j) = \begin{cases} 
  WC_j & \text{if } cl(P, P_j) \\
  0 & \text{otherwise}
\end{cases}
\]

3. \( f_3(P, P_j) \) refers to the supposed increment of interests of page \( P_j \) w.r.t. \( P \):

\[
f_3(P, P_j) = \begin{cases} 
  W_j - W_0 & \text{if } W_j > W_0 \\
  0 & \text{otherwise}
\end{cases}
\]

Values of correlation functions are disposed in a similarity matrix \( S \in N^{n \times n} \) where the \( j \) rows correspond to linked pages \( P_j \) to be eventually recommended, the \( i \) columns correspond to correlation functions \( f_i \), and each element \( s_{ij} \) in the matrix is the result of \( f_i(P, P_j) \).

Once computed, matrix \( S \) is multiplied by a correlation matrix \( C \in N^{n \times n} \) which is a diagonal matrix where elements \( d_{ii} \) of the diagonal correspond to the weights of \( f_i \) according to preferences on sources expressed by the ordering relation on \( f_i \), \( i = 1, 2 \) defined in Section 2.1.

By this operation, a weighted similarity matrix \( W_i \in N^{n \times n} \) is obtained.

Each row \( j \) of \( W_i \) is associated a value \( r_j \) which is a linear combination of elements in the row. This value is then used to rank pages. Let us now clarify how the Adviser works by an example.

Example 1 User has asked ALBA agent suggestions on links of page \( P_0 \) containing \( URL(P_j) \), \( j = 1 \ldots 4 \). The Learner component has induced four interesting sources for which \( int(URL_i) \), \( j = 1 \ldots 4 \) holds:

\[
\begin{array}{c}
W_1 = 2 \\
W_2 = 1 \\
W_3 = 2 \\
W_4 = 4
\end{array}
\]

\[
\begin{array}{c}
\text{P} \\
\text{P} \\
\text{P} \\
\text{P}
\end{array}
\]

Figure 2. Example 1: page ranking

Given \( s(P, P_1) = 1 \), \( s(P, P_2) = 0.92 \), \( s(P, URL_1) = 0.87 \), \( s(P, URL_2) = 0.7 \), \( s(P, URL_3) = 0.8 \), \( s(P, URL_4) = 0.9 \), \( s(P, URL_2) = 0.78 \). Suppose the clustering algorithm gives four clusters: \( C_1 = \{ P_1, P(UR_1), P(UR_2) \}, C_2 = \{ P_4, P(UR_1), P(UR_4) \}, C_3 = \{ P_3, P(UR_2), C_4 = \{ P_2, P(UR_4) \} \}

As a result of the clustering process, only page \( P_0 \) is in the same cluster as the original page the user is visiting, \( P_0 \).

Correlation functions are now computed:

\[
\begin{array}{c}
f_1(P, P_j) = \begin{cases} 
  s(P, P_j) & \text{if } j = 3 \\
  0 & \text{otherwise}
\end{cases}
\end{array}
\]

\[
\begin{array}{c}
f_2(P, P_j) = \begin{cases} 
  WC_j & \text{if } j = 3 \\
  0 & \text{otherwise}
\end{cases}
\end{array}
\]

\[
\begin{array}{c}
f_3(P, P_j) = \begin{cases} 
  W_j - W_0 & \text{if } j = 4 \\
  0 & \text{otherwise}
\end{cases}
\end{array}
\]

and put in a similarity matrix \( S \) as described above.

9. Note that the original page user is visiting, \( P \), could be identified by index \( j = 0 \) while computing the weight to be assigned to it.

10. This is just a possibility, as the two evaluations could be combined in a different way or be included in different correlation functions.

11. In this case the number of pages linked by \( P \) is the same as the number of interesting links. This happens only in this examples; ranges of the two indexes can generally be different.

12. Similarity results for the remaining combination of pages are filtered out cause they are lower than a given minimum.
Preferences on reasons are given by \( p(f_j) \), \( i = 1..3 \) and disposed in a correlation matrix \( C \). Their value corresponds to their position in the \( n \) ordered slots (we consider \( p(f_1) \) placed in the \( k \)-th slot as corresponding to \( p(f_1) = n-k+1 \)). Suppose the user expressed the order \( f_1 \succ f_3 \succ f_2 \). The correspondent weighted similarity matrix \( S \cdot C = W_s \) is thus obtained as follows:

\[
W_s = \begin{pmatrix}
0 & 0 & 0 \\
0 & 0 & 0 \\
0.92 & 1 & 0 \\
0 & 0 & 2 \\
0 & 0 & 0 \\
0.92p(f_1) & p(f_2) & 0 \\
0 & 0 & 2p(f_3)
\end{pmatrix}
\]

We have that \( p(f_1) = 3, p(f_3) = 2 \) and \( p(f_2) = 1 \). To define the ranking value \( r_j \) for page \( P_j \) we use a linear combination of elements of the \( j \)-th row of \( W_s \) as follows:

\[
r_j = \sum_{i=1}^{3} w_{ji}
\]

so that \( r_1 = r_2 = 0, r_3 = 2.76 + 1 = 3.76, \) and \( r_4 = 4 \).

This final values are used to rank pages pointed by \( P \). Note that \( P_3 \) is less preferred than \( P_4 \) despite its being in the same cluster as \( P_3 \). This is due to the combination of two factors: first, correlation between \( P_3 \) and \( P_1 \) is not as substantial as the increment of interestingness the user could get in visiting page \( P_4 \); second, function \( f_3 \) combining interestingness of a page and similarity metric, is not much less preferred by the user than \( f_1 \), based on similarity only; as a consequence, due to the preferential nature of the multicriteria framework, page \( P_3 \) is slightly less preferred than \( P_4 \) because browsing through it results in a significant increment of interestingness for the user, although \( P_3 \) is more similar to \( P_0 \) than \( P_4 \). A different order on functions \( f \), would affect this result.

Adding suggestions:

Once all values \( r_j \) has been computed for each page \( P_j \), suggestions are added to \( P \) and a modified version of it, \( P' \), is returned to the user. ALBA returns a ranked list of URLs as a result of evaluating correlation functions. To provide a meaningful \( P' \), the idea is to associate the original page \( P \) with a pop-up window where all URLs in \( P \) are presented in an ordered list according to the results of ranking, so that the user can directly access to them by clicking on the correspondent link.\(^{13}\)

3 Conclusions and Future Work

Adaptive Logic Browsing Adviser (ALBA) is proposed as a framework for users to specify and update a profile of interests and non trivial recommendation policies.

Although no prototype is available for complete validation yet, we believe that a user profile combining general interests and recommendation policies can be useful to increase system performance especially when user is not confident with the search topic and with the recommendation methods applied. The behavior of the whole system, however, depends on the combination of techniques applied for clustering and computing similarity.

The system is expected to be very flexible as it can be integrated with modern recommendation techniques as soon as they are developed, not affecting the effectiveness of its intuitive use for profile specification.

Non intrusiveness is granted by the fact that initial user’s interests are extracted almost automatically: no complex statistical information, or features selection are needed.

Profiling activity performed by ALBA is user- and interaction-oriented, in that only recent browsing activity of the user is considered for profile extraction and update. It would be interesting to allow users to combine and compare results of the application of their preferences in recommendation policies: as an example, in a research group the interests of each member, represented by \( WC_j \) and \( WI_j \), could give the others an immediate and intuitive idea of the related topics that are being investigated for that specific research context.

Correlation functions could potentially be \( n \); this would generate a \( n \)-dimensional matrix expressing more complex recommendation policies. In this respect, we want to consider different sets of correlation functions (e.g. treating \( WC_j \) and \( WI_j \) separately), test more complex heuristics for parameters update and study non-linear combination of elements determining the ranking value \( r_j \).

Recent interest related to preference specification in AI and logic programming \cite{Schaub, Schaub01} suggests an extension of our solution to cases in which a more complex profile needs to be specified expressing a preferential order on the different rules that can be applied to infer interestingness of a page or quantify interestingness of a page to some degree \cite{Gelfond}.

All these aspects represent material for developments we want to investigate in a future paper, where more detailed experimental results will be also provided in order to evaluate effectiveness of this approach and provide significant empirical data.

References

\begin{thebibliography}{11}
\end{thebibliography}

\(^{13}\) Alternative solutions could be considered.

\(^{14}\) This number is limited by the available memory and may affect performances.