Using Ontologies to Discover Domain-Level Web Usage Profiles

Honghua (Kathy) Dai and Bamshad Mobasher
{hdai,mobasher}@cs.depaul.edu

School of Computer Science, Telecommunication, and Information Systems,
DePaul University, Chicago, Illinois, USA

Abstract. Usage patterns discovered through Web usage mining are effective in capturing item-to-item and user-to-user relationships and similarities at the level of user sessions. Without the benefit of deeper domain knowledge, such patterns provide little insight into the underlying reasons for which such items or users are grouped together. This can lead to a number of important shortcomings in personalization systems based on Web usage mining or collaborative filtering. For example, if a new item is recently added to the Web site, it is not likely that the pages associated with the item would be a part of any of the discovered patterns, and thus these pages cannot be recommended. Keyword-based content-filtering approaches have been used to enhance the effectiveness of collaborative filtering systems by focusing on content similarity among items or pages. These approaches, however, are incapable of capturing more complex relationships at a deeper semantic level based on different types of attributes associated with structured objects. This paper represents work-in-progress towards creating a general framework for using domain ontologies to automatically characterize usage profiles containing a set of structured Web objects. Our motivation is to use this framework in the context of Web personalization, going beyond page- or item-level constructs, and using the full semantic power of the underlying ontology.

1 Introduction and Problem Statement

The goal of Web usage mining [24] is to capture and model the behavioral patterns and profiles of users interacting with a Web site. The discovered patterns are usually represented as collections of pages that are frequently accessed by groups of users with common needs or interests. Such patterns can be used to better understand behavioral characteristics of visitor or user segments, improve the organization and structure of the site, and create a personalized experience for visitors by providing dynamic recommendations [25, 7, 3, 15, 9, 10, 22].

At a conceptual level, there may be many different kinds of objects within a given site that are accessible to users. At the physical level, these objects may be represented by one or more Web pages. For example, consider a site containing information about movies. This site may contain pages related to the movies themselves, actors appearing in the movies, directors, studios, etc.

Conceptually, each of these entities represents a different type of semantic object. During a visit to this site, a user may access several of these objects together during a session.

Given the session-based Web usage data, a variety of mining techniques can be used to discover patterns, including clustering, association-rule or sequential pattern discovery. For example, clustering of user sessions may result in the discovery of a group of similar sessions based on pages commonly accessed within those sessions. In order to find an aggregate representation of user interests captured by such a cluster, one might derive the cluster centroid containing a set (or a vector) of pages that are common among cluster elements. In [18, 17], we called these aggregate representations of session clusters *aggregate usage profiles* and used these profiles for Web personalization.

Specifically, each user session s can be viewed as an n-dimensional vector over the space of all pages, i.e.,

$$t = \langle w(p_1, s), w(p_2, s), \ldots, w(p_n, s) \rangle ,$$

where $w(p_i, t)$ is a weight, in session s, associated with the page p_i. The weights can be binary representing the existence or non-existence of a page access in the session, or they can be some value representing the user's interest on the page (e.g., the page stay time). Applying data mining techniques, such as clustering, to this space may result in a set $CL = \{cl_1, cl_2, \ldots, cl_k\}$ of session clusters, where each cl_i is a subset of the set of sessions.

Given a session cluster cl, we can construct a usage profile pr_{cl} as a set of pageview-weight pairs by computing the centroid of cl:

$$pr_{cl} = \{\langle p, weight(p, pr_{cl}) \rangle \mid weight(p, pr_{cl}) \geq \mu\}$$

where

- the significance weight, $weight(p, pr_{cl})$, of the page p within the usage profile pr_{cl} is given by
$$weight(p, pr_{cl}) = \frac{1}{|cl|} \cdot \sum_{s \in cl} w(p, s);$$
- $w(p, s)$ is the weight of page p in session $s \in cl$; and
- the threshold μ is used to focus only on those pages in the cluster that appear in a sufficient number of sessions in that cluster.

Each such profile, in turn, can be represented as a vector in the original n-dimensional space. The aggregate representation of common usage patterns as sets or vectors of pages makes such usage profiles quite useful for personalization and collaborative filtering [12]: given a new user, u who has accessed a set of pages, P_u, so far, we can measure the similarity of P_u to the discovered profiles, and recommend to the user those pages in matching profiles which have not yet been accessed by the user.

While in the above discussion we have focused on clustering as the primary data mining technique for the discovery of usage profiles, it should be noted that

a variety of other techniques can also be used. For example, frequent itemsets discovered as part of association rule mining [1] on the usage data, would also lead to sets of items or pages representing usage profiles [16].

One problem with the above approach is that the profiles only capture common usage patterns at the page level. They do not reflect the underlying reasons why the group of users represented by the profile are interested in the accessed pages. This can lead to a number of important shortcomings in personalization systems based on Web usage mining or collaborative filtering. For example, if a new item is recently added to the Web site, it is not likely that the pages associated with the item would be a part of any of the discovered patterns (due to a lack of sufficient usage level data). Yet, a user who fits one of the profiles may indeed be interested in the new item, because the underlying domain characteristics of the item might correspond to those of items in one or more of the profiles.

In the movie site example mentioned above, consider the situation where a discovered usage profile may contain pages related to a number of movies many of which are from the same genre and have been directed by the same director. Using the above standard approach for personalization, pages appearing in this profile may be recommended to a user who has accessed some of the other pages in that profile. However, if a new movie is added to the site with similar properties, it will not appear in any profile until a sufficient number of users have accessed this movie together with other similar movies. This problem is often coined as the "new item problem" in collaborative filtering.

A common approach to resolving this problem has been to integrate content characteristics of pages with the usage patterns [6, 20, 21, 19]. Generally, in these approaches, keywords are extracted from the content on the Web site and are used to either index pages by content or classify pages into various content categories. In the context of personalization, this approach would allow the system to recommend pages to a user, not only based on a matching usage profile, but also (or alternatively) based on the content similarity of these pages to the pages user has already visited.

Keyword-based approaches, however, are incapable to capturing more complex relationships among objects at a deeper semantic level based on the inherent properties associated with these objects. In our movie site example, the above content-based filtering approach would allow the system to recommend other movies based on similarities in their textual descriptions or other content characterisitcs. But, the system would have considerable difficulty in recommending, for example, unrelated movies from the same genre, having the same main actors as those already accessed by the user, etc. To be able recommend different types of complex objects using their underlying properties and attributes, the system must be able to rely on the characterization of user segments and objects, not just based on keywords, but at a deeper semantic level using the domain ontologies for the objects.

This paper represents work-in-progress towards creating a general framework for using domain ontologies to automatically characterize usage profiles contain-

ing a set of structured Web objects. Our motivation is to use this framework in the context of Web personalization, going beyond page-level constructs, and using the full semantic power of the underlying ontology. This effort involves the following tasks:

1. Given a page in the Web site, we must extract domain-level structured objects as semantic entities contained within this page. This task involves the automatic extraction and classification of objects of different types into classes based on the underlying domain ontologies. Our goal is to create a representation for a usage profile discovered through Web usage mining process (e.g., through clustering or association rule mining), not simply as a set of pages, but as a set of structured objects embedded in those pages.
2. Given a set of structured objects representing a usage profile, we must create an aggregated representation as a set of pseudo objects each characterizing objects of different types occurring commonly across the user sessions. We call such a set of aggregate pseudo objects a *Domain-level Aggregate Profile*. Thus, a domain-level aggregate profile characterizes a collection of similar users based on the common properties of objects in the domain ontology that were accessed by these users.

We begin by providing a formal framework for the representation of the domain ontology and for creating aggregate representations of domain-level objects. We then discuss how the resulting aggregate profiles can be used in the context of personalization.

2 Representing Domain Ontologies for Web Objects

There has been much recent work in designing ontology languages to formally represent knowledge on the Web, such as *RDFS* [2] and *DAML+OIL* [13]. The ontology language DAML+OIL has been extended to include formal semantics and reasoning support by mapping to the description logic $\mathcal{SHOQ}(D)$ [14].

Generally speaking, in our current work we adopt the syntax and semantics of $\mathcal{SHOQ}(D)$ to represent domain ontologies. In $\mathcal{SHOQ}(D)$, the notion of *concrete datatype* is used to specify literal values and *individuals* which represent real objects in the domain ontology. Moreover, *concepts* can be viewed as sets of individuals, and *roles* are binary relations between a pair of concepts or between concepts and datatypes. The detailed formal definitions for concepts and roles are given in [11, 14]. Because our current work does not focus on reasoning tasks such as deciding subsumption and membership, we do not focus our discussion on these operations. The reasoning apparatus in $\mathcal{SHOQ}(D)$ can be used to provide more intelligent data mining services as part of our future work.

In $\mathcal{SHOQ}(D)$, a concept has the meaning of a set of individuals, which in our context are called "objects". The notion of a concept is quite general and may encompass a heterogeneous set of objects with different properties (roles) and structures. In the present work, we are interested in deriving aggregate representations of a group of objects that have a homogenous concept structure

(i.e., have similar properties and data types). For example, we may be interested in a group of movie objects, each of which has specific values for properties such as "year", "genre", "actors", etc. For the purpose of presentation, we call such a group of objects a *class*. Thus, in our framework, the notion of a class represents a restriction of the notion of a concept in $\mathcal{SHOQ}(D)$.

More specifically, our notion of class can be defined in the context of $\mathcal{SHOQ}(D)$ as follows.

Definition 1 A *class* $C \sqsubseteq I$ (I is the set of all individuals in the domain ontology) is a set of objects where there exists a set of roles R, such that, $\forall r \in R, D_r = \{v_2 \mid (v_1, v_2) \in r\}$, and $C \sqsubseteq \forall R.D_r$.

We call the roles that characterize the class C *attributes*. These attributes together define the internal properties of the objects in C or relationships with other concepts that involve the objects in C. And we denote the domain of values of the attribute r as D_r. Furthermore, because we are specifically interested in aggregating objects at the attribute level, we extend the notion of a role to include a domain-specific combination function and an ordering relation.

More formally, a class C is characterized by a finite set of attributes A_C, where each attribute $a \in A_C$ is defined as follows.

Definition 2 Let C be a class in the domain ontology. An *attribute* $a \in A_C$ is a 4-tuple, $a = \langle T_a, D_a, \psi_a, \preceq_a \rangle$, where

- T_a is the *type* for the values for the attribute a.
- D_a is the domain of the values for a;
- $\preceq_a$ is an ordering relation among the values in D_a; and
- ψ_a is a *combination function* for the attribute a.

The "type" of an attribute in the above definition may be a concrete datatype or it may be a set of objects (individuals) belonging to another class. Given a type T_a for an attribute a, we have $D_a \sqsubseteq dom(T_a)$.

In the context of data mining, comparing and aggregating values are essential tasks. Therefore, ordering relations among values are necessary properties for attributes. We associate an ordering relation $\preceq_a$ with elements in D_a for each attribute a. The ordering relation $\preceq_a$ can be null (if no ordering is specified in the domain of values), or it can define a partial or a total order among the domain values. For standard types such as values from a continuous range, we assume the usual ordering. In cases when an attribute a represents a concept hierarchy, the domain values of a are a set of labels, and $\preceq_a$ is a partial order representing the "is-a" relation.

Furthermore, we associate a data mining operator, called the *combination function* ψ_a, with each attribute a. The combination function ψ_a defines an aggregation operation among the corresponding attribute values of a set of objects belonging to the same class. This function is essentially a generalization of the "mean" or "average" function applied to corresponding dimension values of set of vectors when computing the centroid vector. In this context, we assume that

the combination function is specified as part of the domain ontology for each attribute of a class. An interesting extension would be to automatically learn the combination function for each attribute based on a set of positive and negative examples.

Classes in the ontology define the structural and semantic properties of objects in the domain which are "instances" of that class. Specifically, each object o in the domain is also characterized by a set of attributes A_o corresponding to the attributes of a class in the ontology. In order to more precisely define the notion of an object as an instance of a class, we first define the notion of an instance of an attribute.

Definition 3 Given an attribute $a = \langle T_a, D_a, \psi_a, \preceq_a \rangle$ and an attribute $b = \langle T_b, D_b, \psi_b, \preceq_b \rangle$, b is an *instance* of a, if $D_b \subseteq D_a, T_b = T_a, \psi_b = \psi_a$, and $\preceq_b$ is a restriction of $\preceq_a$ to D_b. The attribute b is a *null instance* of a, if $D_b = \emptyset$.

Definition 4 Given a class C with attribute set $A_C = \{a_1^C, a_2^C, \ldots, a_n^C\}$, we say that an object o is *instance of class* C, if o has attributes $A_o = \{a_1^o, a_2^o, \ldots, a_n^o\}$ such that, a_i^o is a, possibly null, instance of a_i^C, for $1 \leq i \leq n$.

In the context of $\mathcal{SHOQ}(D)$, we can use the concept inclusion axiom $\{o\} \sqsubseteq C$ to denote the "instance-of" relation between o and C. In general, the domain ontology can be represented as a set of assertions on classes and attributes.

Based on the definitions of attribute and object instances, we can provide a more formal representation of the combination function ψ_a. Let c be a class and $\{o_1, o_2, \ldots, o_n\}$ a set of object instances of c. Let $a \in A_C$ be an attribute of class c. The combination function ψ_a can be represented by:

$$\psi_a(\{\langle a_{o_1}, w_1\rangle, \langle a_{o_2}, w_2\rangle, \ldots, \langle a_{o_n}, w_n\rangle\}) = \langle a_{agg}, w_{agg}\rangle,$$

where each a_{o_i} belonging to object o_i is an instance of the attribute a, and each w_i is a weight associated with the attribute instance a_{o_i} representing the significance of that attribute relative to the other instances. Furthermore, a_{agg} is a *pseudo instance* of a meaning that it is an instance of a which does not belong to a real object in the underlying domain. The weight w_{agg} of a_{agg} is a function of $w_1, w_2, \ldots, w_n$.

Given a set of object instances, $\{o_1, o_2, \ldots, o_n\}$, of class C, a *domain-level aggregate profile* for these instances is obtained by applying the combination function for each attribute in c to all of the corresponding attribute instances across all objects $o_1, o_2, \ldots, o_n$.

An Example

As an example, let us come back to the movie Web site discussed in the previous section. The Web site includes collections of pages containing information about movies, actors, directors, etc. A collection of pages describing a specific movies might include information such as the movie title, genre, starring actors, director, etc. An actor or director's information may include name, filmography (a set of

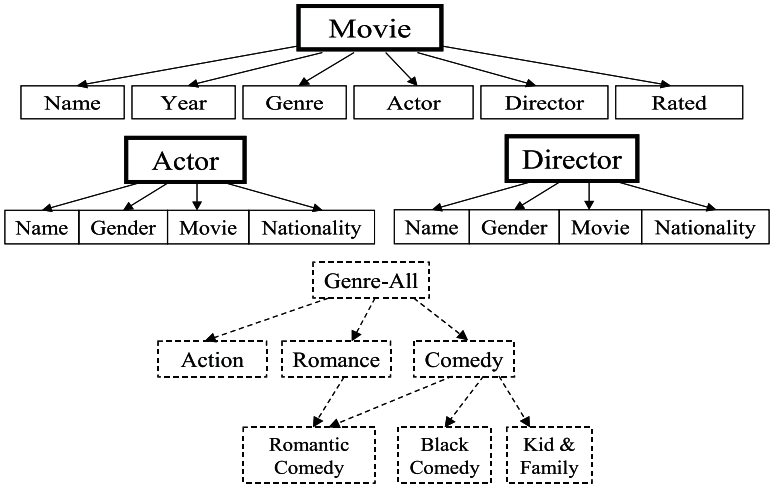

Fig. 1. The Ontology for a movie Web site

movies), gender, nationality, etc. The portion of domain ontology for this site, as described, contains the classes **Movie**, **Actor** and **Director** (see Figure 1). The collection of Web pages in the site represent a group of embedded objects that are the instances of these classes.

The class **Movie** has attributes *Year*, *Actor* (representing the role "acted by"), *Genre*, *Director*, etc. The *Actor*, and *Director* attributes have values that are other objects in the ontology, specifically, object instances of classes **Actor** and **Director**, respectively. The attribute *Year* is an example of an attribute whose datatype is positive integers with the usual ordering. The attribute *Genre* has a concrete datatype whose domain values in D_{Genre} are a set of labels (e.g., "Romance" and "Comedy"). The ordering relation for $\preceq_{Genre}$ defines a partial order based on the "is-a" relation among these labels (resulting in a concept hierarchy of Genre's a portion of which is shown in Figure 1).

An attribute a of an object o has a domain D_a. In cases when the attribute has unique value for an object, D_a is a singleton. For example, consider an object instance of class **Movie**, "About a Boy" (see Figure 2). The attribute *Actor* contains three objects (H. Grant, R. Weisz and T. Collette) that are instances of the class **Actor** (for the sake of presentation we use the Actor's name to stand for the object of Actor). Therefore, $D_{Actor} = \{$H. Grant, R. Weisz and T. Collette$\}$. Also, A real object may have values for only some of the attributes. In this case the other attributes have empty domains. For instance, the attribute *Director* in the example has an empty domain, and is thus not depicted in the figure.

We may, optionally, associate a weight with each value in the attribute domain D_a (usually in $[0, 1]$). This may be useful in capturing the relative importance of each attribute value. For example, in a given movie the main actors should have higher weights than other actors. In our example, the object actor *H.*

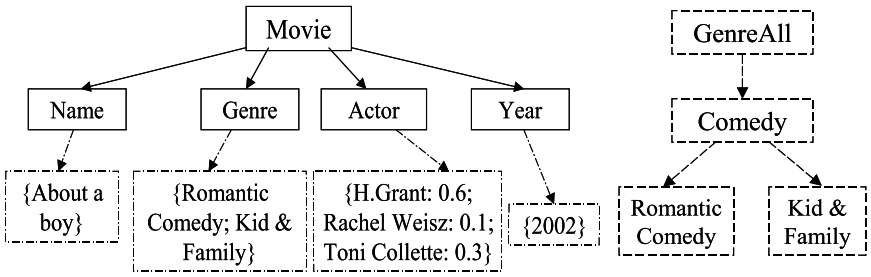

Fig. 2. An Example of an Object in Class Movie

Grant has weight 0.6 and the object Actor *Rachel Weisz* has weight 0.1. Unless otherwise specified, we assume that the weight associated with each attribute value is 1.

In the object o shown in Figure 2, the domain for the attribute *Genre* is the set {"Genre-All", "Comedy", "Romantic Comedy", "Kid & Family"}. The ordering relation $\preceq^o_{Genre}$ is a restriction of $\preceq_{Genre}$ to {Genre-All, Comedy, Romantic Comedy, Kid & Family}.

Let us now define the combination functions for some of the attributes in class **Movie**. Note that the combination functions are only applicable when creating an aggregate representation of a set of objects. For the attribute *Name*, we are interested in all the movie names appearing in the instances. Thus we can define ψ_{Name} to be the union operation performed on all the singleton *Name* attributes of all movie objects.

The attribute *Actor* contains a weighted set of objects in class **Actor**. In such cases we can use a vector-based weighted mean operation. The domain of the resulting aggregate attribute instance D'_{Actor} can be viewed as the union of the domains of the **Actor** attributes of the individual Movie objects: $D'_{Actor} = \cup_i D_{Actor_i}$, and the weight for an object o in D'_{Actor} is determined by

$$w'_o = \frac{\sum_i w_i \cdot w_o}{\sum_i w_i}.$$

For example, applying ψ_{Actor} to {⟨{S, 0.7; T, 0.2; U, 0.1}, 1⟩, ⟨{S, 0.5; T, 0.5}, 0.7⟩, ⟨{W, 0.6; S, 0.4}, 0.3⟩} will result in the aggregate domain D'_{Actor} of {⟨S, 0.58⟩, ⟨T, 0.27⟩, ⟨W, 0.09⟩, ⟨U, 0.05⟩}.

As for the attribute *Year*, the combination function may create a range of all the *Year* values appearing in the objects. Another possible solution is to discretize the full *Year* range into decades and find the most common decades that are in the domains of the attribute. For example, applying ψ_{Year} to {⟨{2002}, 1>}, ⟨{1999}, 0.7⟩, ⟨{2001}, 0.3⟩} may result in an aggregate instance $Year'$ of attribute *Year* with $D'_{Year} = [1999, 2002]$.

The attribute *Genre* of class **Movie** contains a partial ordering relation $\preceq_{Genre}$ which represents a concept hierarchy among different *Genre* labels. Thus, for each instance object p of *Genre*, the relation $\preceq^a_{Genre}$ specifies the restriction

of the partial ordering relation to D_p. The combination function, in this case, can perform tree (or graph) matching to extract the common parts of the conceptual hierarchies among all instances [23]. Given a set of objects $\{p_1, \ldots, p_n\}$ of *Genre*, we can define $\psi_{Genre}(\{\langle p_1, w_1\rangle, \langle p_2, w_2\rangle, \ldots, \langle p_n, w_n\rangle\}) = \langle \cap_i D_{p_i}, w'\rangle$, where w' is the average of the weights of instance values in the intersection. The relation $\preceq'_{Genre}$ is a restriction of $\preceq_{Genre}$ to $\cap_i D_{p_i}$. This function provides us the common segment of conceptual hierarchy for *Genre* in the set of movie instances. For example, applying ψ_{Genre} to {{"Romantic Comedy", "Kids&Family"}, {"Romance", "Comedy"}, {"Romance"}}, and assuming that all weights are 1, will result in an aggregate instance *Genre*$'$ of attribute *Genre* with the value {"Romance"}.

3 Creating an Aggregated Representation of a Usage Profile

We now return to our original problem: how to create an aggregate representation of a discovered usage profile at the domain-level. As noted earlier, a usage profile at the page or item level can be viewed as a weighted set (or vector) of pages. The problem of extracting instances of the ontology classes from these pages is an interesting problem in its own right and has been studied extensively (see, for example, [5]). Here we assume that, either using manual rules, or through supervised learning methods, we can extract various object instances represented by the pages in the original page- or item-level usage profile. Thus, the usage profile can be transformed into a weighted set of objects: $pr = \{\langle o_1, w_{o_1}\rangle, \langle o_2, w_{o_2}\rangle, \ldots, \langle o_n, w_{o_n}\rangle\}$ in which each o_i is an object in the underlying domain ontology and w_i represents o_i's significance in the profile pr. The profile represents a set of objects accessed together frequently by a group of users (as determined through Web usage mining). Our goal is to create an aggregate representation of this weighted set of objects to characterize the common interests of the user segment captures by the profile at the domain level.

Given the representation of a profile pr as a weighted set of objects, the objects in pr may be instances of different classes $c_1, c_2, \ldots, c_k$ in the ontology. The process of creating a domain-level aggregate profile begins by partitioning pr into collections of objects with each collection containing all objects that are instances of a specified class (in other words, the process of classifying the object instances in pr). Let $g_i = \{\langle o_1^{c_i}, w_{o_1^{c_i}}\rangle, \ldots, \langle o_m^{c_i}, w_{o_m^{c_i}}\rangle\}$ be the elements of pr that are instances of the class c_i.

Having partitioned pr into k groups of homogeneous objects, $g_1, \ldots, g_k$, the problem is reduced to creating aggregate representation of each partition g_i. This task is accomplished with the help of the combination functions for each of the attributes of c_i some of whose object instances are contained in g_i. Once the representatives for every partition of objects are created, we assign a significance weight to each representative to mark the importance of this group of objects in the profile. In our current implementation the significance weight for each representative is computed as the sum of all the object weights in the partition.

Input: A weighted set of objects: $O = \{\langle o_1, w_1\rangle, \ldots, \langle o_n, w_n\rangle\}$
Output: The domain-level aggregate of O: $\{\langle o'_1, w'_1\rangle, \ldots, \langle o'_k, w'_k\rangle\}$

Main Procedure:

$Result = \emptyset$
Partition objects in O into $g_1, g_2, \ldots, g_k$,
  such that $\forall g_i, \exists$ a class c_{g_i}, with objects in g_i being instances of c_{g_i}
For each $g_i (i = 1, \ldots, k)$ do
  Build a pseudo object o'_i to be an instance of c_{g_i}:
    For each attribute a in the attribute set of class c_{g_i} do
      Let $a^{o'_i}$ be an instance of a in the object o'_i:
        $a^{o'_i} = \psi_a(g_i)$
    Endfor
  $Result = Result \cup \{o'_i\}$
  $w'_i = \sum_{o_k \in g_i} w_k$
Endfor
Return *Result*

Fig. 3. The Algorithm DPA for Creating and Aggregate Representation of a Weighted Set of Objects

However, significance weight can be computed using other numeric aggregation functions. Figure 3 summarizes the algorithm DPA for creating domain-level aggregate profiles.

Examples Continued: Generating Domain-Level Aggregate Profiles

We present two examples of transforming usage profiles into domain-level aggregates using ontological information. The first example is based on our hypothetical movie Web site (Figure 1), and the second is based on the results of our experiments with a real Web site containing real estate property listings.

Suppose we have discovered a Web usage profile and transformed it into a weighted set of 3 movie objects in the class **Movie** (see Figure 4). We can generate a domain-level aggregate representation of these movies by applying the DPA algorithm to this profile. Because the objects are the instances of the same class, the DPA algorithm does not have to partition the objects. Thus, we can directly apply combination functions on each attribute in the class **Movie**. The details of combination functions used in this example were described in the previous section. Figure 5 shows the pseudo object instance representing the domain-level aggregation of these objects.

Note that the original item-level profile gives us little information about the reasons why these objects were commonly accesed together. However, after we characterize this profile at the domain-level, we find some interesting patterns: they all belong to *Genre* "Romance", and the actor S has a high score compared with other actors. This might tell us that this group of users are interested particularly in the movies belonging to "Romance" and are particularly fond of the actor S.

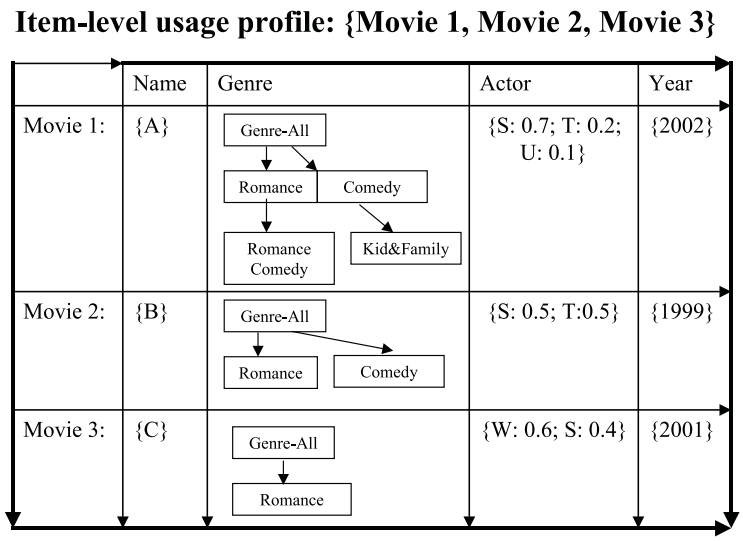

Fig. 4. A Weighted Set of Objects in a Usage Profile from a Movie Web Site

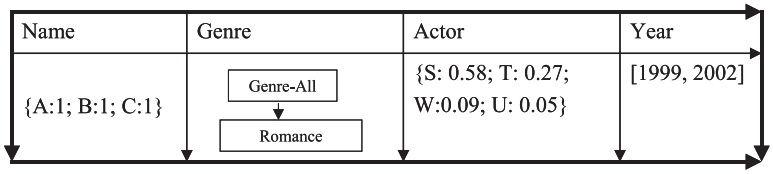

Fig. 5. An Example of Domain-Level Aggregate Profile from a Movie Web site

Our next example is based on a real usage profile discovered from the usage data belonging to a real estate Web site containing various property listings. The ontology of the Web site has a single class **property**. An object which is the instance of class **property** represents a real estate property that has been listed for sale. This ontology is depicted in Figure 6.

In this case, all attributes have atomic values. For example, the attribute *number of bedrooms* has a range of values {1, 2, 3, 4, 5, 6} and *style of building* has values among {*1 Story*, *2 Story*, *Town Home*, *Ranch*}. Figure 6 also shows an example of a property object instance with the value sets for each attribute in dashed boxes.

In our experiment, we began by clustering similar user sessions. Then we generated the usage profiles obtaining the cluster centroids in which each item's weight represents the percentage of cluster sessions containing that item. For example, one of the usage profiles obtained from our experiment is: {⟨Property 1,1⟩,⟨Property 2,0.7⟩,⟨Property 3,0.18⟩,⟨Property 4,0.18⟩}.

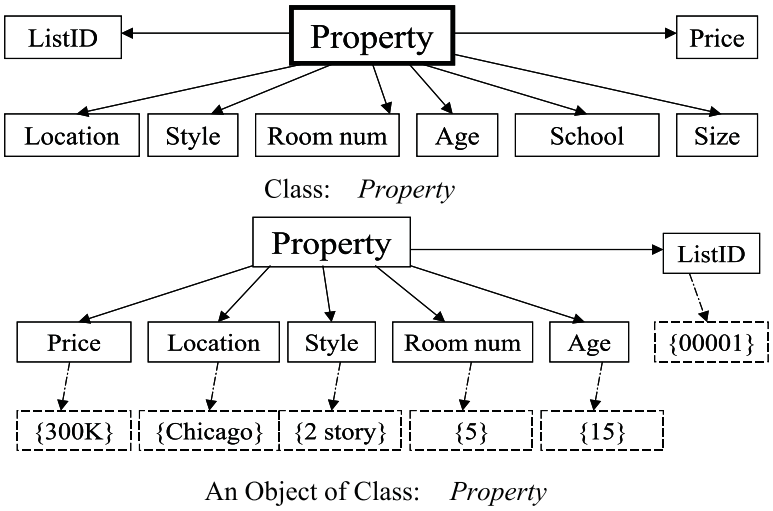

Fig. 6. The Ontology of a Real Estate Web site Containing Property Listings

We used the "weighted average" as our combination function for the attributes containing numeric data. Let w_{a_i} be the weight associated with instance a_i of attribute a. Because all the attribute instances contain singletons as their value sets, we can use v_{a_i} to represent the only value for the instance a_i . The combination function ψ_a for each of the numeric attributes (*Price*, *Size* and *Rooms*) gives the value set D'_a of the aggregate instance, as follows:

$$D'_a = \left\{ \frac{\sum_i v_{a_i} \times w_{a_i}}{\sum_i w_{a_i}} \right\}$$

For the attribute *Location*, the combination function computes the union of all the values in this attribute. In this example, this function returns {Chicago, Evanston}. For the attribute *Style*, the combination function computes the most frequent style values in this attribute. In this case, the function returns {2 Story}. Finally, we used the same combination function for the attribute *Year* as the one used in the class **movie** of the previous example. Figures 7 and 8 show the original usage profiles discovered through clustering and the resulting domain-level aggregate profile, respectively.

After generating domain-level usage profile, we have the information about the average price, size, number of rooms, as well as the locations, major styles, and year range. Such information not only enriches the profile with more knowledge about the listed items, but also provides us more possibilities for Web personalization. In the following section we discuss how we can leverage domain-level aggregate profiles for Web personalization.

Item-level usage profile: {Property 1, Property 2, Property 3, Property 4}

	Weight	Price ($)	Location	Size (ft²)	Rooms	Style	Year Built
Property 1	1	{475000}	{Chicago}	{5260}	{5}	{2 Story}	{1995}
Property 2	0.7	{299900}	{Chicago}	{4790}	{4}	{2 Story}	{1989}
Property 3	0.18	{272000}	{Evanston}	{3119}	{4}	{2 Story}	{1998}
Property 4	0.18	{99000}	{Chicago}	{2056}	{3}	{1 Story}	{1956}

Fig. 7. A Weighted Set of Objects in a Usage Profile from a Real Estate Web Site

Weight	Price ($)	Location	Size (ft²)	Rooms	Style	Year Built
1	{364907}	{Chicago, Evanston}	{4633}	{4}	{2 Story}	[1956-1995]

Fig. 8. An Example of Domain-Level Aggregate Profile from a Real Estate Web site

4 Web Personalization Based on Domain-Level Usage Profiles

Domain-level aggregate profiles present users' interests not just as a set of items or pages, but in terms of the common underlying properties and relationships among those items that are captured by object attributes. This fine-grained domain knowledge enables more powerful approaches to personalization.

We consider the browsing history of the current user to be a weighted set of items that the user has visited. The weight can be based on the amount of time the user spends on each item or other measures that represent the significance of the items in the current browsing session. The algorithm DPA can be applied to this weighted set to create an aggregate domain-level representation of the user browsing history. We call this aggregate representation the *current user model*. Given the current user model, there are two possible approaches to generating personalized recommendations for the user. These two approaches are depicted in Figure 9.

The first approach searches the domain ontologies to recommend the items that have similar features with the current user model. We call the recommendationed objects the *instantiations* of the current user model. This approach is essentially a generalization of the keyword-based content filtering approach and does not rely on any discovered usage profiles. While this approach does not encounter the "New Item" problem, it does not have the advantages of collaborative filtering systems, namely, considering item-to-item relationships that are based on how items are used or accessed together rather than based on content similarity. In addition, this approach requires the definition of possibly complex similarity or distance functions for each of the attributes of the underlying classes in the ontology.

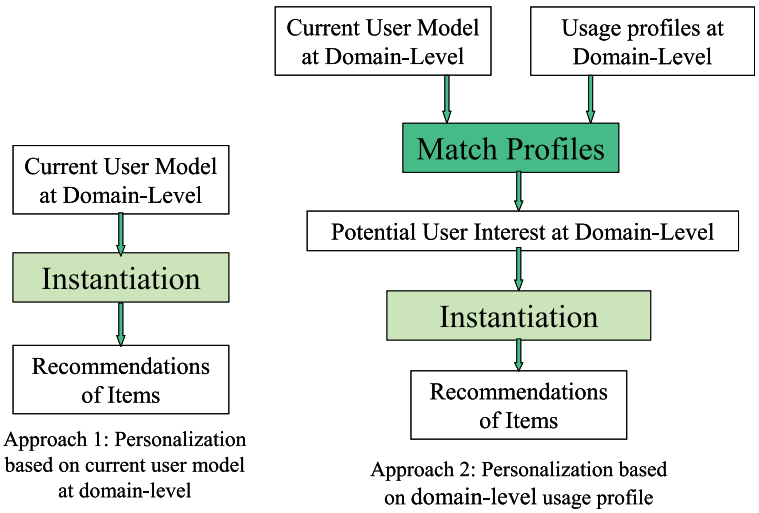

Fig. 9. Personalization Framework utilizing Domain Knowledge

The second method generates recommendations based on the discovered domain-level aggregate profiles. The recommendation engine matches the current user model with all the discovered usage profiles. The usage profiles with matching score greater than some pre-specified threshold are considered to represent this user's potential interests. A successful match implies that the current user shares common interests with the group of users this usage profile represents. The recommendation engine also matches the items in the domain with the user's potential interests and will recommend to the user the matching items. This approach is more complex than the pure content-based approach in that it involves two matching processes: matching the current user with the usage profiles and matching the item candidates with the user's potential interests.

The second approach has a number of advantages. First, it retains the user-to-user relationships that can be captured by the discovered usage profiles. Secondly, in contrast to standard collaborative filtering, it provides more flexibility in matching usage profiles with current user model because in this case matching involves comparison of features and relationships, not exact item identities. Furthermore, the items do not have to appear in any usage profiles in order to be recommended, since fine-grained domain relationships are considered during the instantiation process. The following example shows that this approach can also be used to solve the "New Item" problem.

Let us consider the real estate Web site discussed in the previous section. We again consider the real usage profile containing *property 1, property 2, property 3 and property 4* depicted in Figure 7. Suppose that there is a new item *property 5* (recently added to the site) which is not included in any usage profiles. Assume that *property 5* has attributes $\langle Price = \{370000\}, Location = \{Chicago\}, size = \{4500\}, rooms = \{4\}, style = \{2story\}, Yearbuilt = \{1993\}\rangle$.

Furthermore, assume that a user has browsed pages related to *property 1* and *property 2*. If the recommendation engine were based on item-level usage profiles, *property 5* would not be recommended. The recommendation engine would instead recommend *property 3, property 4*. However, the current user model (*property 1* and *property 2*) is indeed more similar to *property 5* than to *property 3* or *property 4* in terms of their domain-level attributes. If the recommendation engine uses domain-level aggregate profile of Figure 8, it would be able to recommend *property 5*.

5 Conclusion and Future Work

In this paper we have presented a general framework for using domain ontologies to automatically characterize usage profiles containing a set of structured Web objects. Our motivation has been to use this framework in the context of Web personalization, going beyond page- or item-level constrcuts, and using the full semantic power of the underlying ontology.

In our approach we consider a Web site as a collection of objects belonging to certain classes. Given a collection of similar user sessions (e.g., obtained through clustering) each containing a set of objects, we have shown how to create an aggregate representation of for the whole collection based on the attributes of each object as defined in the domain ontology. This aggregate representation is a set of pseudo objects each characterizing objects of different types commonly occurring across the user sessions. We have also presented a framework for Web personalization based on domain-level aggregate profiles.

Currently we assume that we have the predefined combination functions for each class attribute specified as part of the domain ontology. One area of future work involves the study of machine learning techniques in order to discover the best way to summarize the attribute automatically. Another area of future work involves the creation of general as well as domain-specific distance ro similarity functions allowing for the comparison of objects in different classes with the domain-level profiles. Finally another interesting area of work will be to explore use of discoverd domain-level aggregates from Web usage mining to enrich the existing domain ontology for a Web site.

References

1. R. Agrawal and R. Srikant. Fast Algorithms for Mining Association Rules. Proc. *20th Int. Conf. Very Large Data Bases, VLDB*, 1994.
2. D. Brickley, and R.V. Guha, Resource Description Framework (RDF) Schema Specification 1.0, *World Wide Web Consortium*, 2000. http://www.w3.org/TR/rdf-schema/
3. B. Berendt and M. Spiliopoulou. Analysing navigation behaviour in web sites integrating multiple information systems. In *VLDB Journal, Special Issue on Databases and the Web.* 9(1):56-75, 2000.

4. J. Broekstra, M. Klein, S. Decker, D. Fensel, F. van Harmelen, and I. Horrocks. Enabling knowledge representation on the web by extending RDF schema. In Proceedings of the 10th World Wide Web conference, Hong Kong, China, May 1–5, 2001.
5. M. Craven, D. DiPasquo, D. Freitag, A. McCallum, T. Mitchell, K. Nigam, S. Slattery. Learning to construct knowledge bases from the world wide web. In *Artificial Intelligence*, 118:69-113, 2000.
6. M. Claypool, A. Gokhale, T. Miranda, P. Murnikov, D. Netes, and M. Sartin. Combining Content-based and Collaborative Filters in an Online Newspaper. In *Proceedings of the ACM SIGIR '99 Workshop on Recommender Systems: Algorithms and Evaluation.* University of California, Berkeley, Aug. 1999.
7. R. Cooley, B. Mobasher and J. Srivastava. Data preparation for mining World Wide Web browsing patterns. *Journal of Knowledge and Information Systems*, (1) 1, 1999.
8. M. Deshpande and G. Karypis. Selective markov models for predicting web-page accesses. In *First International SIAM Conference on Data Mining*, 2001.
9. E. Damiani, B. Oliboni, E. Quintarelli, and L. Tanca. Modeling users' navigation history. In *Workshop on Intelligent Techniques for Web Personalisation at the 17th International Joint Conference on Articial Intelligence (IJCAI01)*, Seattle, Washington, (USA), 2001.
10. X. Fu, J. Budzik, and K. J. Hammond. Mining navigation history for recommendation. In *Proc. 2000 International Conf. Intelligent User Interfaces*, New Orleans, LA, January 2000. ACM.
11. R. Giugno and T. Lukasiewicz. P-SHOQ(D): A Probabilistic Extension of SHOQ(D) for Probabilistic Ontologies in the Semantic Web. Accepted for publication in Proceedings of *the 8th European Conference on Logics in Artificial Intelligence (JELIA'02)*, Cosenza, Italy, September 2002. Lecture Notes in Artificial Intelligence, Springer, 2002.
12. J. Herlocker, J. Konstan, A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In *Proceedings of ACM SIGIR'99*, 1999.
13. Ian Horrocks. DAML+OIL: a reason-able web ontology language. In *Proc. of EDBT 2002*, March 2002. To appear.
14. I. Horrocks and U. Sattler. Ontology reasoning in the SHOQ(D) description logic. In B. Nebel, editor, *Proc. of the 17th Int. Joint Conf. on Artificial Intelligence (IJCAI-01)*, pages 199-204. Morgan Kaufmann, 2001.
15. B. Mobasher, R. Cooley and J. Srivastava. Automatic personalization based on Web usage mining. In *Communications of the ACM*, (43) 8, August 2000.
16. B. Mobasher, H. Dai, T. Luo, and M. Nakagawa. Effective Personalization Based on Association Rule Discovery from Web Usage Data. *In Proceedings of the 3rd ACM Workshop on Web Information and Data Management (WIDM01), held in conjunction with the International Conference on Information and Knowledge Management (CIKM 2001)*, Atlanta, Georgia, November 2001.
17. B. Mobasher, H. Dai, T. Luo, and M. Nakagawa. Improving the effectiveness of Collaborative Filtering on Anonymous Web Usage Data. *In Proceedings of the IJCAI 2001 Workshop on Intelligent Techniques for Web Personalization (ITWP01)*, August 2001, Seattle.
18. B. Mobasher, H. Dai, T. Luo, and M. Nakagawa. Discovery of Aggregate Usage Profiles for Web Personalization. *In Proceedings of the Web Mining for E-Commerce Workshop (WebKDD'2000), held in conjunction with the ACM-SIGKDD Conference on Knowledge Discovery in Databases (KDD'2000)*, August 2000, Boston.

19. B. Mobasher, H. Dai, T. Luo, Y. Sun, and J. Zhu. Combining web usage and content mining for more effective personalization. In *Proceedings of the International Conference on ECommerce and Web Technologies (ECWeb)*, 2000.
20. D. Mladenic. Text learning and related intelligent agents: a survey. *IEEE Intelligent Systems*, 14(4):44–54, 1999.
21. M. Pazzani. A Framework for Collaborative, Content-Based and Demographic Filtering. *Artificial Intelligence Review*, Dec. 1999, pp. 393-408.
22. Pitkow J. and Pirolli P. Mining Longest Repeating Subsequences to Predict WWW Surfing. In *Proceedings of the 1999 USENIX Annual Technical Conference*, 1999.
23. R. Ramesh, L. V. Ramakrishnan. Nonlinear pattern matching in trees. In *Journal of the ACM*, 39(2):295-316, 1992.
24. J. Srivastava, R. Cooley, M. Deshpande, P-T. Tan. Web usage mining: discovery and applications of usage patterns from Web data. *SIGKDD Explorations*, (1) 2, 2000.
25. Myra Spiliopoulou and Lukas C. Faulstich. WUM: A Tool for Web Utilization Analysis. In *Proceedings of EDBT Workshop WebDB'98*, Valencia, Spain, Mar. 1998.