Usage-driven Evolution of Personal Ontologies

Peter Haase, York Sure

Institute AIFB
University of Karlsruhe
Germany

{haase,sure}@aifb.uni-karlsruhe.de

Andreas Hotho

KDE Group
University of Kassel
Germany

aho@cs.uni-kassel.de

Lars Schmidt-Thieme

Computer-based New Media
Institute for Computer Science
University of Freiburg
Germany

lst@informatik.uni-freiburg.de

Abstract

Large information repositories as digital libraries, online shops, etc. rely on a taxonomy of the objects under consideration to structure the vast contents and facilitate browsing and searching (e.g., ACM topic classification for computer science literature, Amazon product taxonomy, etc.). As in heterogeneous communities users typically will use different parts of such an ontology with varying intensity, customization and personalization of the ontologies is desirable. In this paper we adapt a collaborative filtering recommender system to assist users in the management and evolution of their personal ontology by providing detailed suggestions of ontology changes. Such a system has been implemented in the context of Bibster, a peer-to-peer based personal bibliography management tool. Finally, we report on an in-situ experiment with the Bibster community that shows the performance improvements over non-personalized recommendations.

1 Introduction and Related Work

Large information repositories as digital libraries, online shops, etc. rely on a taxonomy of the objects under consideration to structure the vast contents and facilitate browsing and searching (e.g., ACM Topic Hierarchy for computer science literature, Amazon product taxonomy). As in heterogeneous communities users typically will use different parts of such an ontology with varying intensity, customization and personalization of the ontologies is desirable.

Such personal ontologies reflect the interests of users at certain times. Interests might change as well as the available data; therefore the personalization requires support for the evolution of personal ontologies. The sheer size of e.g. the ACM Topic Hierarchy makes it quite difficult for users to easily locate topics which are relevant for them. Our approach benefits from having a community of users which allows for recommending relevant topics according to similar interests.

As our main contribution in this paper we adapt a collaborative filtering recommender system to assist users in the management and evolution of their personal ontology by providing detailed suggestions of ontology changes. The approach is implemented as an extension of the Bibster application and has been thoroughly evaluated with promising results.

Related work exists in three different aspects: (i) work in recommender systems, especially collaborative filtering in general, (ii) work in using taxonomies in recommender systems, and (iii) ontology evolution in general. Recommender systems have their roots in relevance feedback in information retrieval (Salton, 1971, p. 324-326) , broaden the domain from documents and link structure to arbitrary domains and typically combine knowledge about different users. User- and item-based collaborative filtering and content-based recommender systems have been introduced in (Goldberg, Nichols, Oki & Terry, 1992, Resnick et al., 1994, p. 175-186, Stojanovic et al., 2002, p. 285-300, Balabanović & Shoham., 1997). Taxonomies are used in recommender systems to improve recommendation quality for items, e.g., in (Middleton, Shadbolt & Roure., 2004, Ziegler et al., 2004). But, to the best of our knowledge there is no former approach for the inverse task, to use recommender systems for the personalization of the taxonomy or more generally of an ontology. Ontology evolution is a central task in ontology

management that has been addressed e.g. in (Klein & Noy. 2003, Stojanovic et al., 2002). One approach for *usage-driven change discovery* in ontology management systems has been explored in (Stojanovic, N. & Stojanovic, L., 2002), where the user's behavior during the knowledge providing and searching phase is analyzed. However, the existing work only addressed the evolution of a single ontology in a centralized scenario. In our work we are extending the idea of applying usage-information to a multi-ontology model by using collaborative filtering to recommend ontology changes based on the usage of the individual ontologies.

The paper is structured as follows. In Section 2 we present our underlying ontology model along with a set of change operations and the methods for recommending ontology change operations. We have implemented the recommender functionalities as extensions of the Bibster application, a semantics-based Peer-to-Peer application. We will describe this implementation in Section 3 and the results of an evaluation experiment in Section 4. Finally, we conclude in Section 5.

2 Recommending Ontology Change Operations

2.1 Ontology Model

In our work we adhere to the Karlsruhe Ontology Model (Stumme et al., 2003):
An *ontology* is a structure $O := (C, \leq_C, R, \sigma_R, \leq_R, I, \iota_C, \iota_R)$ consisting of

- three disjoint sets C, R, and I called *concept identifiers*, *relation identifiers*, and *instance identifiers*,
- a partial order $\leq_C$ on C *called concept hierarchy* or taxonomy,
- a function $\sigma_R : R \rightarrow C^2$ called *signature*,
- a partial order $\leq_R$ on R called *relation hierarchy*,
- a function $\iota_C : C \rightarrow 2^I$ called *concept instantiation*,
- a function $\iota_R : R \rightarrow 2^{I \times I}$ called *relation instantiation*.

2.2 Ontology Change Operations

Based on this ontology model, the smallest operations can be derived in a straightforward manner from its different components. We can define operations for adding and removing concepts (C^+, C^-), instances (I^+, I^-), or relations (R^+, R^-), as well as for asserting and retracting concept hierarchy relationships $(\leq_C^+, \leq_C^-)$, i.e., $c \leq d$ for $c, d \in C$, relation hierarchy relationships $(\leq_R^+, \leq_R^-)$, i.e., $s \leq t$ for $s, t \in R$, or concept instantiations, (ι_C^+, ι_C^-), i.e., $i \in \iota_C(c)$ for $i \in I$, $c \in C$ or relationship instantiations, (ι_R^+, ι_R^-), i.e., $(i, j) \in \iota_R(s)$ for $i, j \in I$, $s \in R$. In the following, we will denote this set of possible ontology change operations with *OCO*. More complex operations such as merge, move, etc. can be expressed using a sequence of change operations (Stojanovic et al., 2002). In the following however, we will focus on recommending atomic change operations, i.e. adding and removing elements of the ontology model.

2.3 Ontology Ratings

Our ontology model so far describes the actual state of an ontology for a user. Once we enter the more dynamic scenario of ontology evolution, it makes sense that a user (i) can express more fine-grained how important a certain element for him is and (ii) can express explicitly negative ratings for elements not part of his ontology.

We model this importance information by a rating annotation. Let S denote the set of all possible elements of the ontology that are allowed to be rated (e.g. $S = C$), then an *ontology rating annotation* is a partial function $r: S \rightarrow R$. High positive values denote the relative importance of an element, negative values that it is unwanted by the user. We will consider rating annotations as an additional ontology component in the following.

In particular, we define the following two ontology rating annotations:

1. We use an explicit rating, called the membership-rating r^m with taboos, for which (i) all elements actually part of the ontology have rating +1, (ii) all elements not actually part of the ontology can be explicitly marked taboo by the user and then get a rating -1.
2. We use an implicit, usage-based rating called r^u, which indicates the relevance of the elements based on how it has been used, e.g. counts the percentage of queries issued by the user and instances in his knowledge base that reference a given element.

We will consider rating annotations as an additional ontology component in the following.

2.4 Recommending Ontology Changes

A recommender system for ontology changes tries to suggest ontology changes to the user based on some information about him and eventual other users. Formally, an *ontology recommender* is a map

$$\rho : X \to 2^{OCO}$$

where X contains suitable descriptions of the target ontology and user.

For example, let recommendations depend only on the actual state of a user's ontology, i.e., $X = O$, where O denotes the set of all possible ontologies. A simple ontology evolution recommender can be built by just evaluating some heuristics on the actual state of the ontology, e.g., if the number of instances of a concept exceeds a given threshold, it recommends adding subconcepts to this concept. But without any additional information, this is hardly very useful, as we would not be able to give any semantics to these subconcepts: we could recommend to further subdivide the concept, but not how, i.e., neither be able to suggest a suitable label for these subconcepts nor assertions of instances to them. We will call such an approach *content-based* to distinguish it from more complex ones.

Recommendation quality eventually can be improved by taking into account other users' ontologies and thereby establishing some kind of collaborative ontology evolution scenario, where each user keeps his personal ontology but still profits from annotations of other users. The basic idea is as follows: Assume that for a target ontology we know similar ontologies called *neighbors* for short, then we would like to spot patterns in similar ontologies that are absent in our target ontology and recommend them to the target ontology. Another wording of the same idea is that we would like to extract ontology change operations that applied to the target ontology increase the similarity with its neighbors.
Let

$$sim : O \times O \to R$$

be such a similarity measure where (O,P) is large for similar ontologies O and P and small for dissimilar ontologies. Typically, these measures are symmetric and maximal for two same arguments. For further properties and examples of similarity functions for ontologies, we refer the reader to (Ehrig, Haase & Stojanovic, 2004).

Recall that ontologies in our context may have additional rating annotations that are valuable information to consider in similarity measures suitable for recommendation tasks.

We can choose a simple unnormalized correlation measure (vector similarity) to compute similarities between ontologies of two users based on their ratings of the elements in the ontology:

$$sim_r(O,P) := \frac{\sum_{s \in S} r_O(s) r_P(s)}{\sqrt{\sum_{s \in S} r_O(s)^2} \sqrt{\sum_{s \in S} r_P(s)^2}}$$

Similarities for the two different rating annotations r^m and r^u are computed separately and then linear combined with equal weights:

$$sim(O,P) := \frac{1}{2} sim_{r^m}(O,P) + \frac{1}{2} sim_{r^u}(O,P)$$

Finally, as in standard user-based collaborative filtering, ratings of all neighbors Ω are aggregated using the similarity-weighted sum of their membership ratings, allowing for a personalized recommender function for a given ontology element *c*:

$$r_{personalized}(O,\Omega,c) := \frac{\sum_{P\in\Omega} sim(O,P)\, r_P^m(c)}{\sum_{P\in\Omega} |sim(O,P)|}$$

The recommendations are obtained directly from the rating: Elements with a positive rating are recommended to be added to the ontology, elements with a negative rating are recommended to be removed. Disregarding the similarity measure between the users' ontologies, we can build a naive recommender that does not provide personalized recommendations, but instead simply recommends "most popular" operations based on an unweighted average of the membership ratings:

$$r_{personalized}(O,\Omega,c) := \frac{\sum_{P\in\Omega} sim(O,P)\, r_P^m(c)}{|\Omega|}$$

3 Case Study: Bibster

In this section we will first introduce the Bibster system (Haase et al., 2004) and the role of personalized ontologies in its application scenario. We will then describe how the recommender functionality is applied in the system to support the users in evolving their personalized ontologies.

3.1 Application Scenario: Sharing Bibliographic Metadata with Bibster

Bibster[1] is an award-winning semantics-based Peer-to-Peer application aiming at researchers who want to benefit from sharing bibliographic metadata. Many researchers in computer science keep lists of bibliographic metadata, preferably in BIBTEX format, which they must laboriously maintain manually. At the same time, many researchers are willing to share these resources, assuming they do not have to invest work in doing so.
Bibster enables the management of bibliographic metadata in a Peer-to-Peer fashion: it allows to import bibliographic metadata, e.g. from BIBTEX files, into a local knowledge repository, to share and search the knowledge in the Peer-to-Peer system, as well as to edit and export the bibliographic metadata.

Two ontologies are used to describe properties of bibliographic entries in Bibster, an application ontology and a domain ontology (Guarino, 1998). Bibster makes a rather strong commitment to the application ontology, but the domain ontology can be easily substituted to allow for the adaption to different domains.

Bibster uses the SWRC[2] ontology as application ontology, which describes different generic aspects of bibliographic metadata. The SWRC ontology has been used already in various projects, e.g. also in the semantic portal of the Institute AIFB[3] .

[1] http://bibster.semanticweb.org/
[2] http://ontoware.org/projects/swrc/
[3] http://www.aifb.uni-karlsruhe.de/about.html

In our scenario we use the ACM Topic Hierarchy[4] as the domain ontology. This topic hierarchy describes specific categories of literature for the Computer Science domain. It covers large areas of computer science, containing over 1287 topics ordered using taxonomic relations, e.g.:
SubTopic(Artificial_Intelligence, Knowledge_Representation_Formalisms).

The domain ontology is used for classification of metadata entries, e.g. *isAbout(someArticle, Artificial_Intelligence)*, therefore enabling advanced querying and browsing. The classification can be done automatically by the application or manually (by drag and drop).

3.2 Extensions for Evolution and Recommendations

In Bibster we initially assumed both ontologies to be globally shared and static. This basically holds for the application ontology, but users want to adapt the domain ontology continuously to their needs. This is largely motivated by the sheer size of the ACM Topic Hierarchy which makes browsing, and therefore also querying and manual classification, difficult for users.

As part of this work we implemented extensions as described in the previous section to Bibster which support the evolution – i.e. the continuous adaptation – of the domain ontology by the users. A basic assumption here is that all users agree in general on the ACM Topic Hierarchy as domain ontology, but each user is only interested in seeing those parts of it which are relevant for him at a certain point of time.

In the application, we have separated the interaction with the ontology in two modes: a *usage mode* and an *evolution mode*. The usage mode is active for the management of the bibliographic metadata itself, i.e. creating and searching for the bibliographic metadata. This mode only shows the current view on the ontology consisting of the topics that the user has explicitly included in his ontology. The evolution mode allows for the adaptation of the ontology. In this mode also the possible extensions along with the corresponding recommendations are shown.

Ontology Change Operations To keep things simple and trying to separate effects from eventually different sources as much as possible, we allow as change operations the addition and removal of topics from the personal ontology. More specifically, this addition/removal corresponds to the addition/removal of the individual assertion axiom (e.g. *Topic(Knowledge Representation Formalisms)*), and the role assertion axiom that fixes the position in the topic hierarchy (e.g. *SubTopic(Artificial Intelligence, Knowledge_Representation_Formalisms)*). The addition of topics is restricted to those topics that are predefined in the ACM Topic Hierarchy. Also, the position of the topics is fixed globally by the background ontology.

Ontology Ratings To elicit as much information as possible from users' work with the application, we gather various ontology rating annotations in the different modes.

We obtain the membership-rating r^m in the evolution mode from the explicit user actions (c.f. Figure 2): The user can either add a topic in the taxonomy, which will assign a rating 1 for the topic, or he can exclude (taboo) the topic from the taxonomy, which will assign -1 for the explicitly taboo-ed topic.

We obtain the usage-based rating r^u in the usage mode by counting the percentage of queries issued by the user and instances in his knowledge base that reference a given topic. (For this, references to all topics are retained, especially also to topics not contained in the ontology of the user.)

The ontology ratings of the individual users are propagated together with peer profile descriptions in the Peer-to-Peer network, such that every peer is informed about the usage of the ontology in the network. For the details of this process, we refer the reader to (Haase et al., 2004).

Recommending Ontology Changes For the recommendations of topics we rely on the rating function $r_{personalized}$ presented in the previous section. From the ratings of the topics, we can directly obtain the recommendations: Topics with a positive rating are recommended to be added to the ontology, topics with a negative rating are

[4] http://www.acm.org/class/1998/

recommended to be removed. (Please note that adding a topic actually means adding the corresponding axioms, as described above.)

Topics in the topic hierarchy are visualized depending on the current rating r^m of the topic and on the recommendation for the topic using the coding scheme shown in Figure 1. Figure 2 shows a screenshot of the ontology in the evolution mode.

Rating	Recommendation		
	Remove	**Neutral**	**Add**
Taboo-ed	X topicname	X topicname	+ topicname
Unrated	- topicname	? topicname	+ topicname
Accepted	- topicname	√, topicname	√, topicname

Figure 1: Visualization of topics in evolution mode

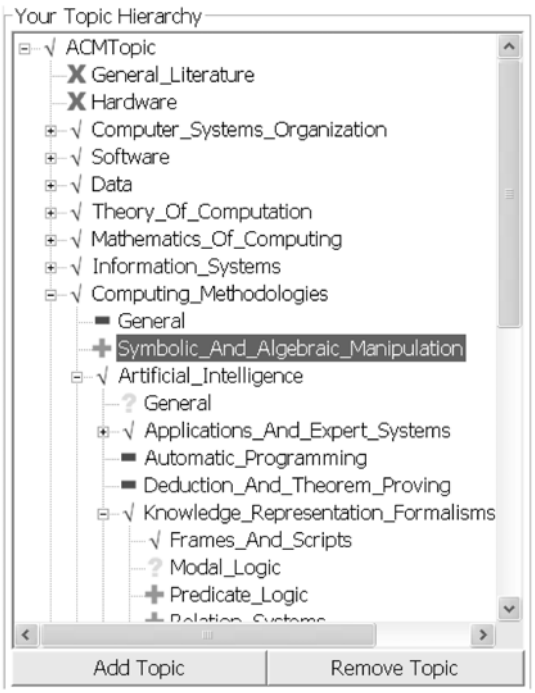

Figure 2: Screenshot

4 Evaluation

For our evaluation, we wanted to study two questions: (i) Do users accept recommendations for ontology changes at all? (ii) Is a personalized recommender better suited for the task than a naive, non-personalized recommender?

To answer these questions, we have performed a user experiment in an in-situ setting using the Bibster system, in which we compared the baseline (non-personalized) and the personalized recommender, as defined in the previous section. In the following we will describe the setup of the experiment, evaluation measures, and the results.

4.1 Design of the Experiment

The experiment was performed within three Computer Science departments at different locations. For a pre-arranged period of one hour, 23 users were actively using the system. The recommender strategy (baseline or personalized) was chosen randomly for each user at the first start of the Bibster application. The users were not aware of the existence of the different recommendation strategies.
During the experiment, the users performed the following activities (in no particular order), which are typical for the everyday use of the system:

- *Import data:* The users need to load their personal bibliography as initial dataset. This data should also reflect their research interest. As described before, the classification information of the bibliographic instances is part of the ontology rating and thus used to compute the similarity between the peers.
- *Perform queries:* The users were asked to search for bibliographic entries of their interest by performing queries in the Peer-to-Peer system. These queries may refer to specific topics in the ontology, and are thus again used as ontology ratings.
- *Adapt ontology:* Finally the users were asked to adapt their ontology to their personal needs and interests by adding or removing topics. This process was guided by the recommendations of the respective recommender function. The recommendations were updated (recalculated) after every ontology change operation.

The user actions were logged at every peer for later analysis. The logged information included: The type of the action (e.g. user query, ontology change operations), the provided recommendations, and a timestamp.

4.2 Evaluation Measures

We base our evaluation on the collected usage information in form of events consisting of the actual user action $e \in OCO$, i.e., the specific ontology change operation performed, and the set $\hat{E} \subseteq OCO$ of recommendations at that point in time, represented by a set $E \subseteq OCO \times P(OCO)$.

We observe a successful recommendation or a *hit*, when $e \in \hat{E}$. For non-hits, we distinguish two situations: (i) If the actual recommendation was exactly the opposite action, e.g., we recommended to add a topic but the user taboo-ed it, then we call this an *error*. (ii) If there was no recommendation for this action neither for its opposite, we call this *restraint*. Based on these counts, we can compute the following performance measures.

$$\text{recall}(\mathrm{E}) := \frac{\left|\left\{(e,\hat{E}) \in \mathrm{E} \mid e \in \hat{E}\right\}\right|}{|\mathrm{E}|}$$

$$\text{error}(\mathrm{E}) := \frac{\left|\left\{(e,\hat{E}) \in \mathrm{E} \mid opp(e) \in \hat{E}\right\}\right|}{|\mathrm{E}|}$$

$$\text{restraint}(\mathrm{E}) := \frac{\left|\left\{(e,\hat{E}) \in \mathrm{E} \mid opp(e) \notin \hat{E} \wedge e \notin \hat{E}\right\}\right|}{|\mathrm{E}|}$$

where *opp* denotes the respective opposite operation, e.g., $opp(e^+) := e^-$ and $opp(e^-) := e^+$. Higher recall and lower error and restraint are better.

For a higher level of detail, we do so not only for all user actions, but also for some classes $OCOC \subseteq OCO$ of user actions, such as all *add*- and all *remove/taboo*-operations.
As each of the measures alone can be optimized by a trivial strategy, we also computed the profit of the recommenders with respect to the profit matrix in Table 1:

$$\text{profit(E)} := \frac{\sum_{(e,\hat{E})\in \text{E}} \sum_{\hat{e}\in\hat{E}} \text{cost}(e,\hat{e})}{|\text{E}|} = recall(E) - error(E)$$

Table 1: Evaluation Profit Matrix

User Action	Recommendation		
	Remove	**None**	**Add**
Remove	1	0	-1
None	0	0	0
Add	-1	0	1

An intuitive reading of the profit is: The higher the profit, the better the performance of the recommender. In the best case (*profit*=1), all user actions were correctly recommended by the system, in the worst case (*profit*=-1), all user actions were opposite of the recommendation.

4.3 Evaluation Results

For the 23 participating users in the experiment, the baseline recommender was active for 10 users, the personalized recommender was active for the other 13 users. The participants performed a total of 669 user actions (452 add topic and 217 remove topic), 335 of these action were performed by users with the baseline strategy, 334 by users with the personalized recommender. Table 2 shows the number of add-topic-actions for the most popular topics.

Table 2: Most Popular Topics

ACM Topic	**# Add Actions**
Information_Systems	23
Computing_Methodologies	15
Data	14
Computing_Methodologies/Artificial_Intelligence	12
Information_Systems/Database_Management	12
Software	11
Mathematics_Of_Computing	10
Computer_Systems_Organization	10
Computer_Systems_Organization/Computer_Communication_Networks	10
Computing_Methodologies/Artificial_Intelligence/ Knowledge_Representation_Formalisms_And_Methods	10

Figure 3 shows the cumulative results of the performance measures defined above for the baseline and the personalized recommender. The diagrams show the results for *Add* and *Remove* operations separately, as well as combined for all change operations.

As we can see in Figure 3 (upper right), overall the personalized recommender correctly recommended more than 55% of the user actions, while the baseline achieved less than 30%. The error rate of the baseline algorithm is considerably higher: We observed an *error* = 17% and 9% for the baseline and the personalized approach, respectively. Further we observed a very large amount of restraint operations with *restraint* = 67% for users with the baseline strategy. Probably this is the result of a large number of recommendations irrelevant to the user given by the system with the baseline strategy. In such a case the user would not like to follow the system and constructs the ontology mainly by himself. Only from time to time he takes some of the recommendations into account.

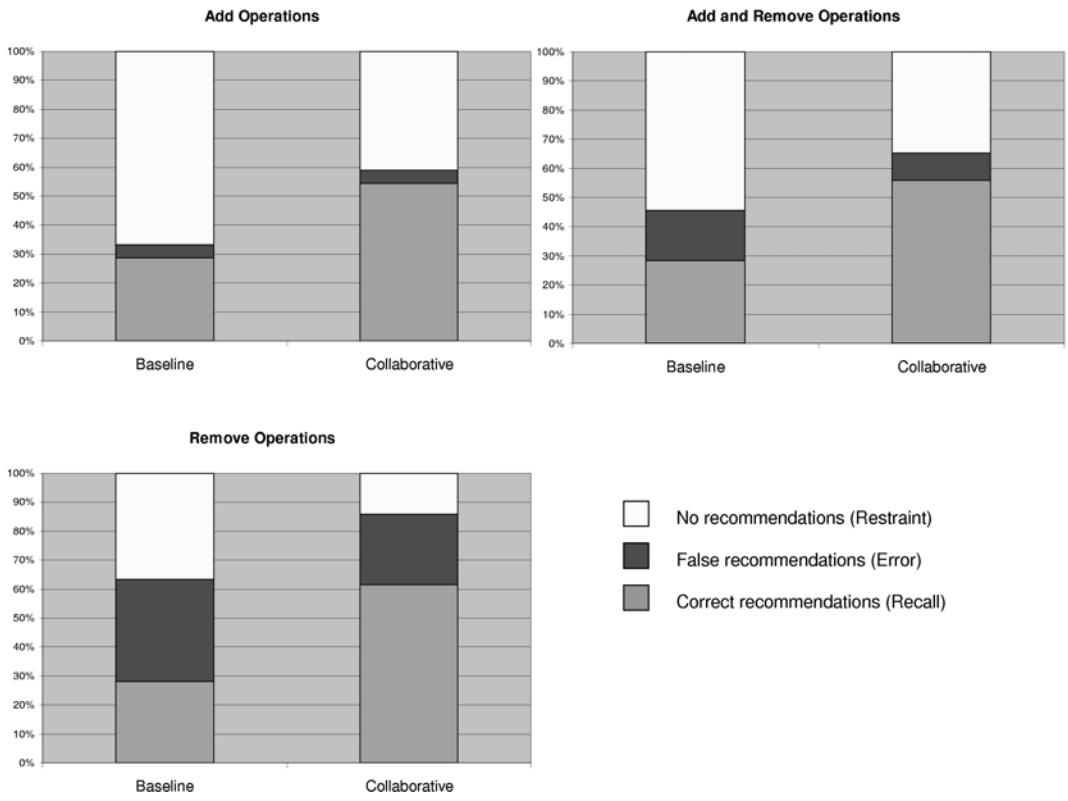

Figure 3: Performance measures of the recommender

By comparing add and remove operations we observe a higher amount of *error* recommendations for remove operations in comparison to the really small amount of it for the add recommendations while the correct recommendations are comparable for both operations (cf. Figure 3, left side). We think that this observation is based on the fact that a user is more likely to follow an add operation without a "substantiated" reason or explanation than a remove operation. While adding something to his "collection" and following the idea of having more the remove operation forces the feeling of "loosing" something, so typically users are more reluctant to remove topics.

Calculating the overall profit of the two recommender functions, we obtain $profit(E) = 0.11$ for the baseline recommender. For the collaborative recommender, we obtain a significantly better value of $profit(E) = 0.47$. Concluding we can state that the personalized recommender function provides substantially more useful recommendations.

5 Conclusion

We have presented an approach to recommend ontology change operations to a personalized ontology based on the usage information of the individual ontologies in a user community. In this approach we have adapted a collaborative filtering algorithm to determine the relevance of ontology change operations based on the similarity of the users' ontologies.
The results of our experimental evaluation with the Peer-to-Peer system Bibster show the benefit of exploiting the similarity between the users' ontologies in personalized recommender compared with a simple, non-personalized baseline recommender. As the recommendation of adding or removing concepts in a given concept hierarchy can only be a first step we focus for the next steps on recommendations of richer change operations.

Acknowledgments: Research reported in this paper has been partially financed by the EU in the IST project SEKT (IST-2003-506826) (http://www.sekt-project.com). We would like to thank our colleagues for fruitful discussions.

References

Balabanović, M. & Shoham, Y. (1997). Fab - content-based, collaborative recommendation. *CACM*, 40(3):66–72,.

Ehrig, M., Haase, P., & Stojanovic, N. (2004). Similarity for ontologies - a comprehensive framework. In *Workshop Enterprise Modelling and Ontology: Ingredients for Interoperability, at PAKM 2004*, DEC 2004.

Goldberg, D., Nichols, D., Oki, B, & Terry, D. (1992). Using collaborative filtering to weave an information tapestry. 35(12):61–70. *CACM*

Guarino, N. (1998). Formal ontology and information systems. In N. Guarino, editor, *Proc. 1st Int. Conf. on Formal Ont. in Inf. Sys. (FOIS)*, volume 46 of *Frontiers in AI and App.*, Trento, Italy,. IOS-Press.

Haase, P., Broekstra, J., Ehrig, M., Menken, M., Mika, P., Plechawski, M., Pyszlak, P., Schnizler. B., Siebes, R., Staab, S., Tempich, C. (2004). Bibster - a semantics-based bibliographic peer-to-peer system. In *Proc. of the Third Int. Semantic Web Conf., Hiroshima, Japan, 2004*, Nov. 2004.

Klein, M. & Noy, N. (2003). A component-based framework for ontology evolution. In *Proc. of the WS on Ont. and Distr. Sys., IJCAI '03*, Acapulco, Mexico, Aug.9, 2003.

Maes, P. & Shardanand, U. (1995). Social information filtering: algorithms for automating "'word of mouth"'. In *Proc. of the SIGCHI conf. on Human factors in computing systems*, pages 210–217. ACM Press/Addison-Wesley Publishing Co.

Middleton, S., Roure, D. D., & Shadbolt, N. (2004). Ontological user profiling in recommender systems. *ACM Trans. on Inf. Systems*, 22:54–88, 2004.

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., Riedl, J.(1994). Grouplens: An open architecture for collaborative filtering of netnews. In *Proc. of the Conf. on Comp. Sup. Coop. Work (CSCW'94)*, pages 175–186, Chapel Hill NC. Addison-Wesley.

Salton, G. (1971). Relevance feedback and the optimization of retrieval effectiveness. In Salton, G., editor, *The SMART system — experiments in automatic document processing*, pages 324–336. Prentice-Hall Inc., Englewood Cliffs, NJ.

Stojanovic, L., Mädche, A., Motik, B., Stojanovic, N. (2002). User-driven ontology evolution management. In *European Conf. Knowledge Eng. and Management (EKAW 2002)*, pages 285–300. Springer-Verlag.

Stojanovic, N. & Stojanovic, L. (2002). Usage-oriented evolution of ontology-based knowledge management systems. In *Int. Conf. on Ontologies, Databases and Applications of Semantics, (ODBASE 2002), Irvine, CA*, LNCS, pages 230–242.

Stumme, G., Ehrig, M., Handschuh, S., Hotho, A., Maedche, A., Motik, B., Oberle, D., Schmitz, C., Staab, S., Stojanovic, L., Stojanovic, N., Studer, R., Sure, Y., Volz, R. & Zacharias, V.(2003). The Karlsruhe view on ontologies. Technical report, University of Karlsruhe, Institute AIFB.

Ziegler, C., Schmidt-Thieme, L., Lausen, G. (2004). Exploiting semantic product descriptions for recommender systems. In *Proc. 2nd ACM SIGIR Semantic Web and IR WS (SWIR '04), July 25-29, 2004, Sheffield., UK*.