The current issue and full text archive of this journal is available at
www.emeraldinsight.com/1468-4527.htm

Received 2 September 2012
First revision accepted
26 February 2013

Web analytics of user path tracing and a novel algorithm for generating recommendations in Open Journal Systems

Behnam Taraghi, Martin Grossegger, Martin Ebner and Andreas Holzinger
Graz University of Technology, Graz, Austria

Abstract

Purpose – The use of articles from scientific journals is an important part of research-based teaching at universities. The selection of relevant work from among the increasing amount of scientific literature can be problematic; the challenge is to find relevant recommendations, especially when the related articles are not obviously linked. This paper seeks to discuss these issues.

Design/methodology/approach – This paper focuses on the analysis of user activity traces in journals using the open source software "Open Journal Systems" (OJS). The research questions to what extent end users follow a certain link structure given within OJS or immediately select the articles according to their interests. In the latter case, the recorded data sets are used for creating further recommendations. The analysis is based on an article matrix, displaying the usage frequency of articles and their user selected successive articles within the OJS. Furthermore, the navigation paths are analysed.

Findings – It was found that the users tend to follow a set navigation structure. Moreover, a hybrid recommendation system for OJS is described, which uses content based filtering as the basic system extended by the results of a collaborative filtering approach.

Originality/value – The paper presents two original contributions: the analysis of user path tracing and a novel algorithm that allows smooth integration of new articles into the existing recommendations, due to the fact that scientific journals are published in a frequent and regular time sequence.

Keywords Algorithms, Open Journal System, Path analysis, Recommender system, Reseach-based teaching, Web analytics

Paper type Research paper

Introduction and motivation for research

Scientific literature is an important knowledge source for research-based teaching activities in universities (Handelsman *et al.*, 2004; Holzinger, 2010, 2011). Established engineering education is mostly deductive, i.e. the teacher starts with basics and fundamentals, then moves to theories and consequently progresses to the applications of those theories. Alternative teaching approaches are more inductive, e.g. topics are introduced by presenting specific observations, case studies or problems and theories are taught using examples or the students are assisted to discover them. Student-centred and inductive teaching methods can be helpful in reaching such goals (Motschnig-Pitrik and Holzinger, 2002; Prince and Felder, 2006).

For all these approaches, the recommendation of appropriate literature is a major issue and Recommender Systems (RS) are an alternative method of presenting and

Online Information Review
Vol. 37 No. 5, 2013
pp. 672-691

1468-4527
DOI 10.1108/OIR-09-2012-0152

recovering knowledge from various resources and therefore a further step towards personalised e-learning systems (Khribi *et al.*, 2009). In contrast to search engines, which deliver results from a user query, RS are focused on delivering unexpected results (Kim *et al.*, 2004), which also support the pooling and sharing of information for informal learning (Linton and Schaefer, 2000). They even try to create recommendations for resources, whereas search engines are restricted in that regard due to limitations in automatic content analysis.

In this paper we introduce an RS for an online journal system. We focus on the analysis of user paths (the sequence of articles read by users known as user activity) at first. The influence of an indexing structure such as the table of contents in the navigation path is examined and visualised. We emphasise that the recorded data can be misleading when applied in the recommendation algorithms, as they do not reflect the interests of users appropriately. The analysis shows that users mainly follow the existing table of contents. Based on this analysis we introduce a hybrid RS consisting of a content-based technique and a collaborative filtering algorithm. The collaborative filtering algorithm is based on the weighted user paths. We introduce a method to reduce the undesirable effect of the navigation path (in our case a table of contents) so that the proposed collaborative filtering algorithm can provide relevant serendipitous and novel recommendations.

For our analysis a journal with an installation of Open Journal Systems (OJS) was used. OJS (Willinsky, 2005) is a journal management and publishing program that includes the possibility to generate peer reviews. To extend the functionality of OJS, the possibility of plug-in development is offered. Consequently a RS can easily be integrated and made available for all installations of OJS. The main indexing structure is the list of articles of an issue of one particular journal.

Background and related work

There are different types of RS that vary in terms of the addressed domains and user requirements. Content-based RS try to recommend items that are similar to those the users have liked before. They rely on matching the users' interests and preferences to the attributes of the items (Mladenic, 1999). Similar items are those that have the same attributes in common. For example in a movie recommendation application attributes such as actors, genres and subjects can be taken into consideration for each movie. Each item is described as a set of predefined attributes in a structured way. The attributes can have a known set of values in this case. Some similarity measures such as cosine similarity or machine learning algorithms can be used to learn the similar items and the associated user profiles (Pazzani and Billsus, 2007; Amatriain *et al.*, 2011). In social tagging RS the user tagging activity is taken into account. Tags are freely chosen keywords that are used by users to annotate and categorise the resources within a social content sharing system. Szomszor *et al.* (2007) introduced a movie RS. The algorithm used in their approach is based on the similarity between tags of a movie and the tags of movies the user has already rated. Diederich and Iofciu (2006) introduced a prototype that uses tag-based profiles to find and recommend people with similar interests. Instead of using objects directly, they used the tags associated with the objects to build the user's profile and enhance a user's current community of practice.

To generate content-based recommendations for text-based resources such as documents or articles, it is necessary to analyse and compare the textual contents.

Many algorithms exist for automatic text analysis such as latent semantic analysis (LSA) and term frequency/ inverse term frequency.

LSA (Landauer *et al.*, 1998; Hofmann, 2001; Kreuzthaler *et al.*, 2011; Evangelopoulos *et al.*, 2012) uses a singular value decomposition to reduce the dimensions of the word-document matrix and remove the sparsity of the matrix (Deerwester *et al.*, 1990). This form of content analysis tries to identify concepts, in such a way that text documents can be compared beyond the analysis of term frequencies such as used in the term frequency/ inverse term frequency method (Ramos, 2003). To reduce the size of the word-document matrix, a pre-processing step with the use of some standard natural language processing operations such as tokenisation, stop-words removal and stemming algorithms (Baeza-Yates and Ribeiro-Neto, 1999; Porter, 1980; Snowball, 2011) can be introduced.

For dynamic multimedia resources including music, images and movies there are no appropriate algorithms for automatic content analysis so far (Yoo *et al.*, 2004). To create recommendations for such dynamic resources, additional information is required such as metadata or a tagging system. However, such media should not be selected randomly, but by using learners' previous knowledge and expertise (Holzinger *et al.*, 2008). Therefore both approaches have the disadvantage of requiring careful manual editing (Lops *et al.*, 2011).

Consequently there is an urgent need for algorithms to recommend resources without knowing the content of the previously recommended items. Such algorithms, referred to as collaborative filtering (Schafer *et al.*, 2007) or people-to-people correlation (Schafer *et al.*, 2001), use the implicit data generated from the user's interaction with the resources and comparison with other users. The main idea is: if there are users who use similar items compared to the test-user, then there is a high probability that the test-user is interested in the items consumed by their neighbours (Desrosiers and Karypis, 2011). Another prediction task performed by a recommender system is the rating prediction (Liu *et al.*, 2011). Here the system analyses the ratings among the users to predict how the test-user will rate a new item. Explicit data can be used to perform this task, for example ranking a resource by a user. It is possible to use implicit data, such as the time a user has spent on an article; however this data cannot be treated as equally reliable as explicit data (Adomavicius and Tuzhilin, 2005). Lemire and Maclachlan (2005) introduced a collaborative filtering item-to-item recommendation technique that predicts ratings for items using a model-based scheme. This approach takes the number of ratings to perform a weighted prediction. Niemann *et al.* (2010) introduced a new technique of similarity calculations for item-based collaborative filtering. In their paper they stated the hypothesis that similarity of usage indicates content similarity. To prove their hypothesis they monitored the usage of learning objects and based on that created the usage context profiles for the objects.

Content-based RS have some assets and drawbacks when compared to RS based on collaborative filtering algorithms (Lops *et al.*, 2011). They are user independent in the sense that they rely solely on the similarities between items, whereas collaborative filtering methods need to find the users who have the same taste as the active user (nearest neighbours). By the same token, collaborative filtering RS suffer from the so-called cold-start problem: they fail to consider new items for recommendations, as users have not rated new items previously. Content-based algorithms tend to give recommendations with a low degree of novelty (Herlocker *et al.*, 2004). This problem, referred to as the serendipity problem (Beale, 2007), or over-specialisation, causes the

RS to recommend items that are too similar to the ones the user has already rated. Hence, it prevents the RS from giving surprising and serendipitous recommendations.

To improve the expected results not only in terms of accuracy (Adomavicius and Tuzhilin, 2005) but also in terms of serendipity (Beale, 2007) and novelty (Herlocker *et al.*, 2004) the underlying algorithms can be combined to form hybrid systems (Burke, 2007). This leads to different forms of hybrid systems such as "switching" (using system one or system two dependent on a computed threshold), or "cascading" (the systems are ordered like a cascade, and a secondary system improves the results of a primary system) (Burke, 2005).

As in any field, such as e-commerce, there is also a need for RS in the e-learning domain. There have been some studies on RS to support recommendation of learning resources (Manouselis *et al.*, 2011). Nadolski *et al.* (2009) created an environment to simulate different combinations of hybrid RS for informal learning networks. They compared different algorithms with regard to the impact on learners. Manouselis *et al.* (2007) introduced a neighbourhood-based collaborative algorithm approach to support recommendation of learning objects. They considered multi-dimensional ratings of learning resources provided by the users for the recommendation. Tang and McCalla (2004, 2005) proposed an evolving e-learning system that is used for sharing research papers among university students. They applied a hybrid RS using tags and user ratings. They clustered learners into groups of learners with similar interests and then used classic collaborative filtering techniques to identify learners with similar interests in each individual cluster.

The hybrid RS proposed in this paper consists of a content-based technique as the primary system, thus avoiding the cold-start problem. A collaborative filtering approach is used as the secondary system that extends the results from the primary system according to pre-defined criteria. The collaborative approach used in our RS is based on the weighted user paths.

User path tracing

One of the available online resources (L3t, 2011) provides a specific RS with the data for the analysis of user behaviour concerning the use of articles. L3T focuses on e-learning and has been issued as an online book consisting of 48 articles written by 115 authors and is an open educational resource. The data for the analysis was gathered with Piwik Web Analytics (Piwik, 2011), an open source tool for monitoring user actions within a web application (Marek, 2011). At the time of our data analysis (mid-2011), there were about 40,000 visits and 180,000 actions stored in the Piwik database. In our case the points of interest are the documents read by the users, each identified by a unique ID. For instance the first article in the table of contents (TOC) has the internal ID number 89 (see Table I).

A path is referred to as a sequence of articles read by the user. "Read by the user" implies that the user has downloaded the complete article and not only looked at the abstract. The shortest possible path contains two elements, for example $89 \Rightarrow 88$. This means that the user first downloaded article 89 and then article 88.

The idea behind the analysis of user paths was to find out whether users browse the OJS L3T according to a specific interest in particular topics or in the order of the TOC given by the journal. If a new user starts to read some articles he/she might follow an existing recommended path and items recorded on this path would probably be interesting for this user.

Article matrix

The first analysis of the extracted data aimed to determine the frequencies of any successive article. One article was chosen and all frequencies for each successive article were noted. This was done for every article, resulting in an article matrix (see Table I). All articles are treated as independent from the articles read before. The idea was to investigate whether some articles were more likely to be selected next.

The article matrix stores the current article in the row and the frequency for each following article in the columns. As an example, consider article 89. The algorithm counts all articles that are read after article 89 and computes their frequencies.

Table I shows a part of the resulting article matrix. Each row is the label of one particular article. The rows are in the same order as the TOC. The number in the columns represents the frequencies, how often the document in the column is accessed after reading the document in this row. For example look at row 6 with the label 41. All frequencies for the next articles are very low. The article which follows next according to the TOC is article 44. Its column has a value of 0.70, which indicates that 70 per cent of the articles that were read after article 41 had the label 44.

In the whole matrix the highest values are noted along the diagonal of the matrix and range from about 50 per cent to 70 per cent. Due to the fact that the article labels describing rows and columns of the matrix are in the same order as the TOC, these high values along the diagonal mean that the TOC has a very strong influence on the decisions of users as to which article will be read next. The values beside the diagonal are very low and almost distributed equally. Hence in the next step, the frequencies of the TOC are removed. Articles that follow next in the TOC are ignored and marked with the label TOC (see Table II).

The highest values now range up to 30 per cent and are again grouped around the diagonal. For example row 2 with the label 88 has 29 per cent in column 1 (label 89 marked in italic in Table II), which stands for the article back in the TOC. Column 3 with the label 49 is skipped (TOC) since it marks the successor in the TOC. Column 4 with the label 73 still reaches 13 per cent (marked in italics in Table II). The matrix again shows the influence of an indexing structure on the behaviour of users, for the successive articles are more likely to be selected out of the surrounding area of the current article in the TOC.

Path analysis

The next idea was to track the users' paths through the journal and store the extracted paths in a path database. In contrast to the article matrix, the articles are not

Table I.
Article matrix showing the influence of the TOC on the sequence of read articles

Label (TOC)	89 (1)	88 (2)	49 (3)	73 (4)	54 (5)	41 (6)	44 (7)	38 (8)	71 (9)
89 (1)		*0.62*	0.09	0.03	0.02	0.02	0.00	0.01	0.01
88 (2)	0.13		*0.55*	0.06	0.04	0.03	0.00	0.04	0.00
49 (3)	0.08	0.07		*0.53*	0.04	0.05	0.01	0.04	0.01
73 (4)	0.03	0.03	0.04		*0.71*	0.04	0.01	0.04	0.00
54 (5)	0.03	0.02	0.02	0.03		*0.64*	0.02	0.05	0.01
41 (6)	0.02	0.02	0.01	0.01	0.03		*0.70*	0.09	0.00
44 (7)	0.01			0.00		0.04		*0.75*	0.05
38 (8)	0.02	0.01	0.02	0.00	0.01	0.01	0.03		*0.52*
71 (9)	0.02		0.01	0.00		0.01	0.02	0.02	

independent in this case. If a new user follows a known path, some items on this or similar paths can be recommended. Each unique path is treated as one specific case. The treatment of recommendations as cases is described by Lorenzi and Ricci (2005) in detail.

To extract the paths from the Piwik dataset, the matrix algorithm was modified to create some path structure. Through this algorithm about 1,700 paths were discovered within the dataset gathered by the Piwik logging tool.

Figure 1 shows an example of the stored user paths.

The first row notes the path that is already traced, in this case 49|73| i.e. path 49 ⇒ 73. All paths that start with article 49 and go on with article 73 are examined. Next to this path, the article numbers to select the next article are listed. In the main frame, the statistical analysis of the examined paths is performed. All possible article numbers that follow the selected path in the database are listed in the square brackets. Additionally, the frequency of occurrence is shown. For example in Figure 1 article 54 has the highest frequency of occurrence. Seven out of 14 different paths have the form 49 ⇒ 73 ⇒ 54.

A formal description of this line can be expressed as follows:

[ID of next article] : #occurrence of #unique paths (percentage) #total paths.

The paths that do not continue are not displayed in the list. This can be noticed when the occurrences are summed up. In our example the paths which consist only of the articles 49 and 73 are not displayed, and the sum of the unique paths displayed is 13. The considered path consists of the first three articles: 89 ⇒ 88 ⇒ 49. The successive article with the highest frequency is number 73 (about 80 per cent) and it is the

Label (TOC)	89 (1)	88 (2)	49 (3)	73 (4)	54 (5)	41 (6)	44 (7)	38 (8)	71 (9)
89 (1)		TOC	0.24	0.08	0.06	0.05	0.01	0.02	0.02
88 (2)	*0.29*		TOC	*0.13*	0.08	0.07	0.01	0.08	0.01
49 (3)	0.16	0.15		TOC	0.09	0.10	0.03	0.09	0.01
73 (4)	0.10	0.09	0.15		TOC	0.14	0.05	0.15	0.02
54 (5)	0.10	0.06	0.05	0.08		TOC	0.05	0.13	0.02
41 (6)	0.06	0.06	0.04	0.05	0.09		TOC	0.31	0.02
44 (7)	0.04			0.01		0.17		TOC	0.18
38 (8)	0.04	0.02	0.04	0.01	0.03	0.03	0.06		TOC
71 (9)	0.09		0.03	0.01		0.03	0.09	0.11	

Table II.
Article matrix without considering the frequency of TOC

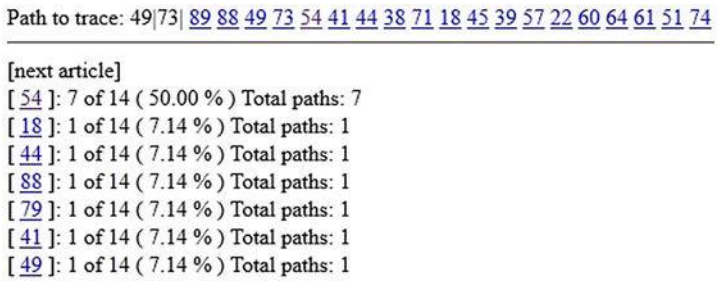

Figure 1.
Tracing paths of article 49 followed by article 73

successive article in the TOC. Additionally in this case the total amount of paths (175) is different from the unique amount (111). It implies that there are some identical paths, in that different users have taken the same path (starting with $89 \Rightarrow 88 \Rightarrow 49 \Rightarrow 73$). It seems logical that along the TOC the probability that users follow the same path is much higher.

Graphical analyses

In Figure 2 a comparison of all unique paths with their starting documents can be observed. The first three articles reach the highest values. These are the documents from where the most paths start. Articles which are situated lower in the TOC are less likely to be the beginning of a user path.

Figure 3 shows the comparison of the absolute path lengths: the number of documents read by the user when starting from one particular article. The very high value of the first article can be explained by the fact that there are a lot of paths which start from the first article of the TOC. Hence, these paths have a high average length (Figure 4). Furthermore, many users took the same path. As a result the first article in the TOC reaches an enormous value.

The comparison of the average length of the paths (the number of all read documents of all paths, starting from one particular article, divided by the number of all paths starting from that article) shows again a strong first article. Additionally, it can be seen that documents which are located lower in the TOC can generate some paths with a good average length. However, this has to be compared to Figure 5 where the total numbers of paths are listed.

Figure 6 shows the comparison of the total amount of paths starting from a particular article. In contrast to Figure 3, here all paths are counted (not only the unique ones). The Figures are not particularly different because articles placed lower in the TOC are less likely to generate identical paths. The first few articles are the most frequent beginnings for identical paths.

The recommendation system

As the analysis shows, it is necessary to generate recommendations which differ from an existing path order such as a TOC. Every RS that relies merely on the navigation paths cannot produce surprising recommendations in OJS as the behaviour of the users is influenced by the sequence defined in the TOC. The analysis of user paths can be helpful for developing an RS but it is important not to follow an existing structure. Datasets regarding users' navigation paths on OJS can be seen as additional resources for an RS. To use a collaborative filtering approach (Herlocker *et al.*, 2004; Desrosiers and Karypis, 2011), it is necessary to identify the users and build clusters, so that a new user can be compared and added to. Such a system would not work for new incoming users because users can only be added to a neighbourhood if they have already read some common articles. Another problem is the identification of the users in OJS. Normally no user account is needed to read an article in an OJS. Hence, the identification has to be performed in different ways, such as cookies. This can lead to imprecise collaborative filtering, as the users may not be identified correctly.

This fact underlines that the RS should not base its algorithms totally on a collaborative filtering approach. The primary RS for OJS can work with a content based filtering approach (Adomavicius and Tuzhilin, 2005) based on the abstract texts of the articles. To create better recommendations a secondary system can be implemented based on a collaborative filtering algorithm using clustering and the

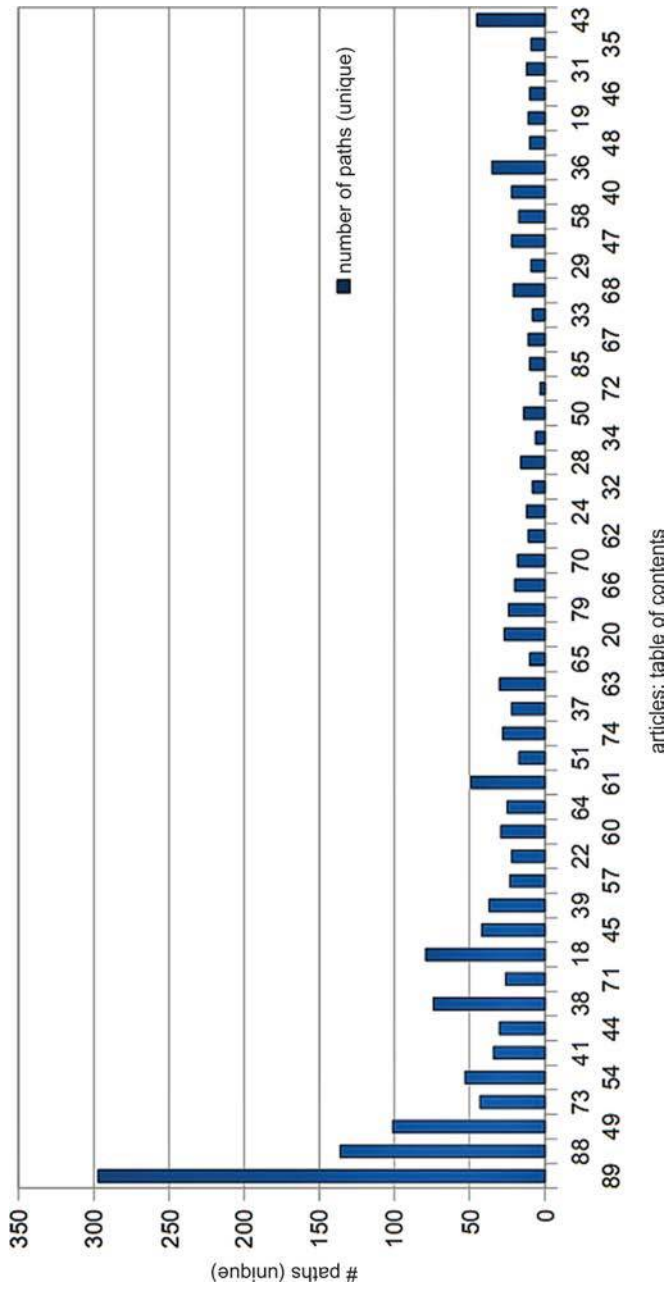

Figure 2.
Comparison of unique paths

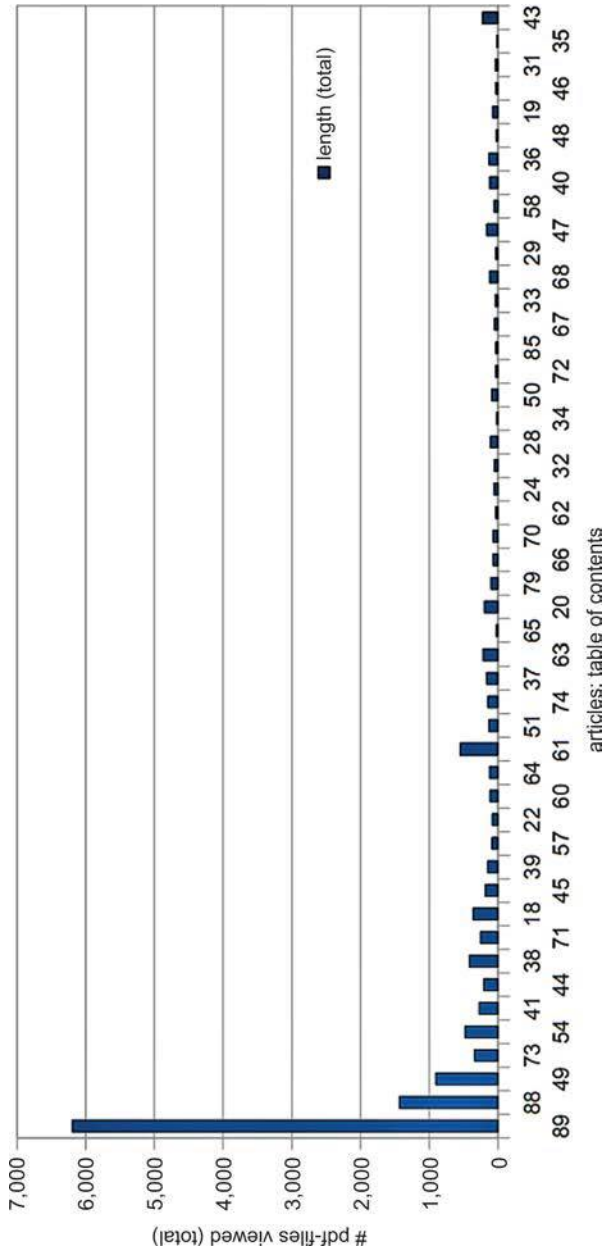

Figure 3.
Comparison of total lengths

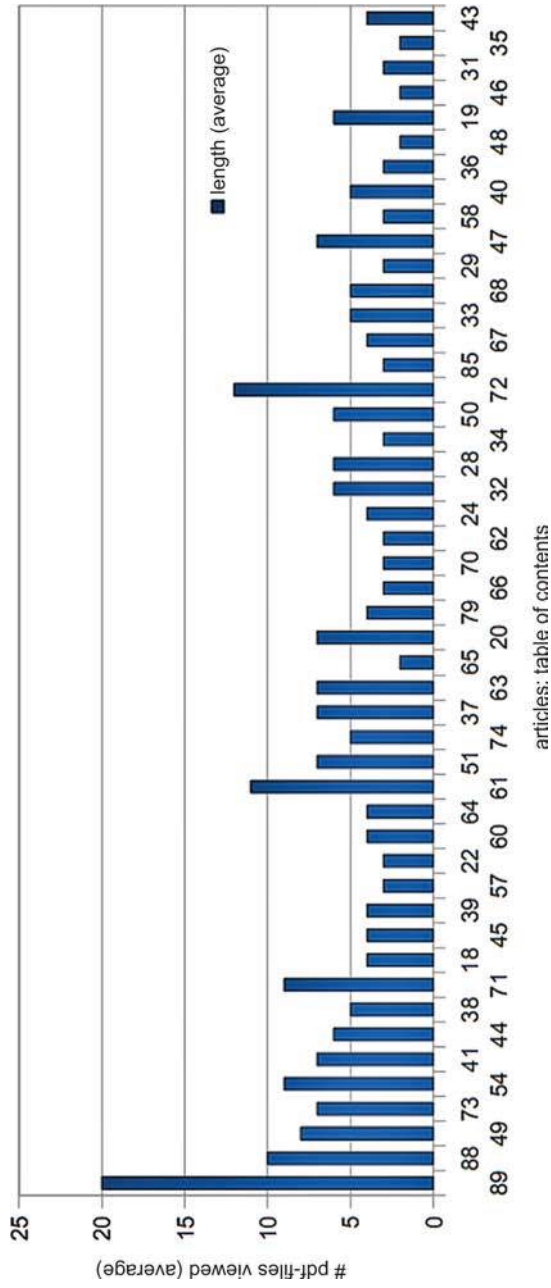

Figure 4.
Comparison of average lengths

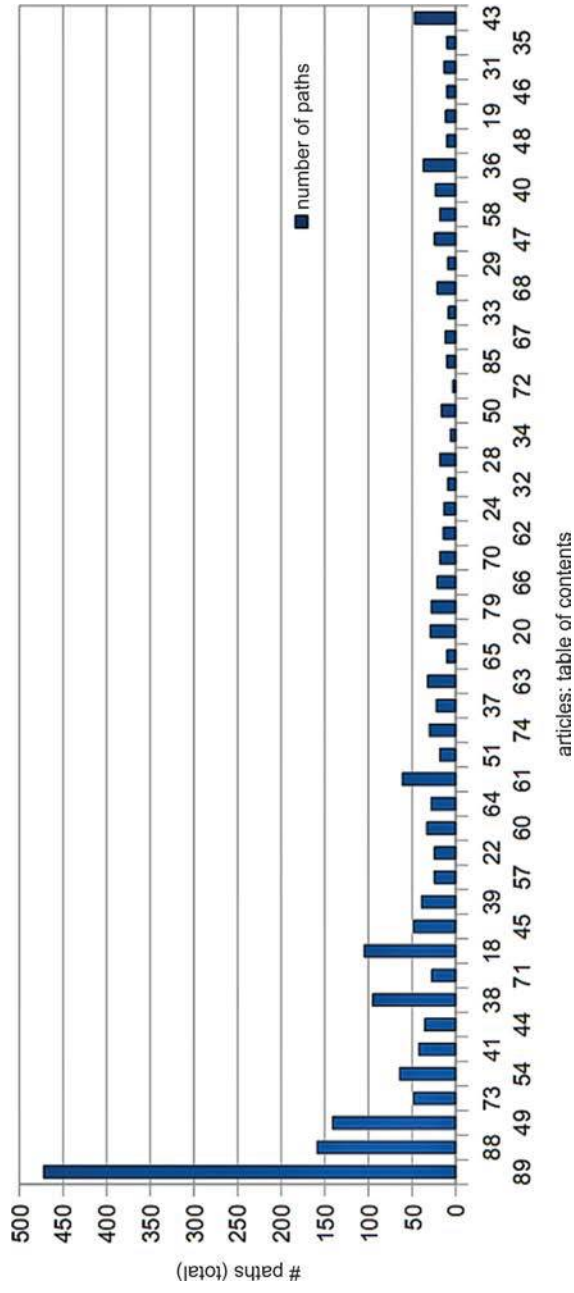

Figure 5.
Comparison of total paths

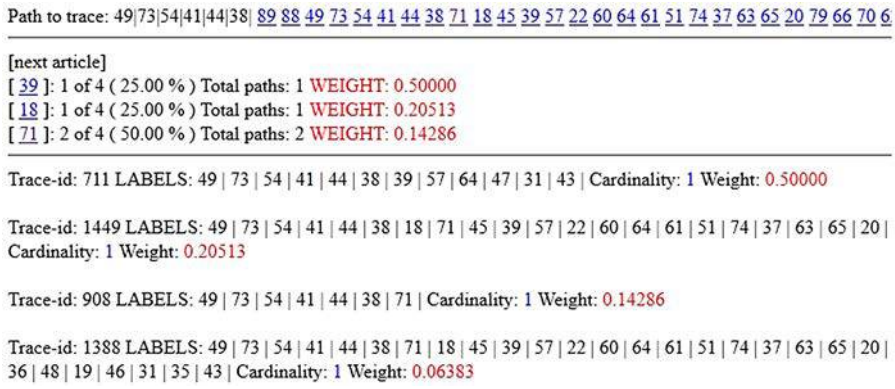

Figure 6.
Path tracing including weights

recorded user paths. This secondary system can improve the results obtained by the primary system.

Primary system: content based filtering

To generate recommendations without considering user behaviour, it is necessary to compare the content of the articles. One basic concept is "tagging" where the author(s) of an article add keywords describing the content or the categories that the article belongs to. Tagging depends strongly on the accuracy of the editor to enter meaningful tags. If every article in an e-learning journal is tagged with the term "e-learning", it will not be very helpful for finding related articles. This leads to the need for additional tools to find the similarity between articles, using some form of automatic content analysis such as LSA.

LSA is used to identify the concepts of the abstract texts. Stemming is used as a pre-processing step to reduce the size of the word-document matrix used in LSA.

By combining tagging and the use of stemming as well as LSA on the abstracts of the articles in OJS, basic recommendations can be created. Better results can be achieved if the whole article is analysed instead of just the abstract texts. In case of the RS for OJS, we considered only the abstract text to reduce the computation time and hence the overall server load.

Secondary system: collaborative filtering

To implement the secondary system, which uses a collaborative filtering approach, it is necessary to identify the users. It is possible to read the articles without being registered in OJS. As a consequence the identification of a user has to be done through cookies or by other means, for instance the user's static IP address.

The system records the user's path by storing each visited page as a path record, which consists of the path, the document-label and the position in the path.

Recommendations can be created by examining similar paths and recommending items that lie ahead on a similar path. Better results can be achieved by grouping the users according to the articles they have read and forming user groups. For this task it is necessary to store the documents read by each user in a user-document table. Users who have selected similar articles will have a better chance of producing paths that will meet the interests of the other members of their groups. This can be realised by a periodic clustering algorithm, such as nearest neighbour clustering (Kim and Yang,

2005). User groups are created in the background and the existing users are added to the created groups. Then the algorithm can select the paths for creating recommendations out of the paths which are related to a group of users.

As seen in the analysis of user paths it is necessary to treat frequent paths in a special way, especially if they represent an existing link structure like the TOC. Concerning the view of a collaborative recommender system this means that the elements of such a path do not represent the interests of the user. According to this observation similar paths do not represent similar taste and as a consequence articles in such paths cannot provide a reliable basis for the computation of user similarity.

A method to measure the relevance of a path is the computation of a specific weight for each path. The goal is to punish paths which follow a given link structure; in the case of the investigated system L3T, this is the TOC. The found relations with the highest frequencies are stored in a separate table and used for punishing paths which include some of these relations. A relation is defined as articleA ⇒ articleB; this means articleB is succeeding articleA. Additionally, the inverse direction articleB ⇒ articleA is added to create bi-directional relations. Within the dataset used in this paper, these relations represent the TOC, as the highest frequencies follow the TOC (Figure 1).

For each path, each article pair of the path is compared against the relation table and the found relations are counted and divided by the length of the path. Thus it is possible to find even small parts of a certain path which are identical to an existing link structure like the TOC. The punishing term for a complete path is computed as follows:

$$Pt_p = 1 - \frac{\#\text{relationsInPath}}{path\_length}$$

The frequency of a specific path plays an important role. It is not expedient that frequent paths be recommended because if they become even more frequent, a point can be reached where they will always be recommended. So a high frequency finally ranks down the path in the recommendation algorithm:

$$weight_p = pt_p^* \frac{1}{path\_frequency}$$

The weight of a path is formed by dividing the punishing term of the path (pt_p) by the frequency of the path. The maximum weight an article can achieve is 1. A weight less than 1 indicates that it must contain parts of a given structure or it must occur more than once or in both cases. This computation is a weakness of this algorithm, as paths that exist only once are rated better than similar paths, which exist for example three times. The adjustment of the weight of each path by the frequency of the path so that high frequency paths are filtered out must be examined in future work.

This algorithm was implemented within the tracing application to demonstrate the recommendation of articles based on the user traces. Even after following a long path 49 ⇒ 73 → 54 ⇒ 41 ⇒ 44 ⇒ 38 the system tries to leave the strongest path (which is the TOC). For example the computation for Trace-ID 711:

#relations in path = 6; (49 ⇒ 73, 73 ⇒ 54, 54 ⇒ 41, 41 ⇒ 44, 44 ⇒ 38, 39 ⇒ 57);

Length = 12;

6/12 = 0.5;

Cardinality = 1;

0.5*1/1 = 0.5

The next article according to the TOC is number 71. All with this article number succeeding the current path have a low weight. Paths which are different from the TOC are more likely to be recommended. The paths do not reach high weights because they contain many relations from the TOC.

RS for OJS: a hybrid system

To overcome the limitations of the content based and collaborative filtering approaches, a hybrid system combines the results of the primary and the secondary system. The primary system with the content-based approach has the benefit of being able to produce results regardless of the amount of data concerning the user's paths; these are the basic recommendations and must always be present in the final set. If there are enough data to form user groups and if the current user has read enough articles to be linked to a group, the secondary system starts with the generation of recommendations by creating user groups and investigating the paths linked to the users of such groups. These items are used to extend or replace some of the results of the primary system.

If the secondary system finds some recommendations, some articles can be replaced from the set obtained by the primary system. Criteria for replacement are: the user has already read an article found by the primary system, or the similarity value of an article found by the primary system is below a specific threshold. Some results from the secondary system should always be displayed (if available), as a collaborative filtering system is more likely to produce surprising results than a system with a content-based filtering approach (Desrosiers and Karypis, 2011). A proposal for displaying a final result set is using $m = 3$ elements of the content based filtering system and $n = 2$ elements of the collaborative system:

$$\mathrm{SET}_{\mathrm{recommendations}} = \mathrm{mResult}_{\mathrm{content\ based}} + \mathrm{nResult}_{\mathrm{collaborative}}$$

The question of how many of the results of the two systems should be displayed must be investigated by a field study by using different settings of weighting the two systems and comparing the results with some test users who rate the received recommendations.

Evaluation of RS for OJS

To qualify an algorithm, it is necessary to investigate the algorithm in terms of usefulness and relevance of its results. For this purpose a second algorithm was implemented which works with conditional probabilities. Using the article matrix (Figure 1), the computation of the conditional probabilities was done by using the Bayesian Theorem (Bretthorst, 1990):

$$P(B|A) = \frac{P(B \cap A)}{P(A)} = \frac{P(A|B)^{*}P(B)}{P(A)}$$

In the case of articles, this equation computes the probability that a user reads article B after reading article A. Additionally, a mind map was manually made to visualise the relevance of the articles to each other. It was used as a reference to measure the values

of precision and recall (Manning *et al.*, 2008). In the context of RS the precision is the fraction of recommended articles that are relevant to the actual article. Recall is the fraction of relevant articles that have been recommended by the RS.

To perform a valid system quality check, k-fold cross-validation was used (Kohavi, 1995). One of the most used types of this method is the 10-fold cross-validation that was applied to our test system.

In addition to the precision and recall measures, a third measurement value was introduced: the hit ratio. The hit ratio was computed by comparing the recommendations with the next article selected by the user. If the next article is in the set of recommended articles, it is considered a hit. The number of "hits" divided by the number of created recommendations is called the hit ratio (Jin *et al.*, 2005). Both algorithms were tested through the iterations and the values for precision and recall for each algorithm were computed and compared.

Validation results

For each of the two algorithms, the 10-fold cross-validation process was performed. Each validation step was performed three times with input data of different path lengths. The experiments demonstrated that paths containing more than five articles produce good results. If the paths used for analysis are shorter, too many possibilities exist and the quality of the predictions worsens. For the tests, different path lengths were used: 6 to 10, 6 to 20 and 6 to 50. These paths were read from the input data set and simulate the users' actions.

In Table III all tests are noted. One recommendation is defined as six recommended articles generated for the current article. The recommendations created in the smaller borders (e.g. $L < 10$) are included in the higher ones (e.g. $L < 20$). Overall, 13,547 recommendations were generated and analysed. To demonstrate the effect of the path length, each step of the cross-validation was performed within the three borders and evaluated for each border. The average results for precision are shown in Table IV, and the average results for recall in Table V. Our algorithm is called "Traces" in both tables and the algorithm with conditional probabilities is named "WSK".

The comparison of the precision values of the two algorithms shows a better value of 7 per cent for the "WSK" because it can always give a recommendation when a relation between the articles exists. In contrast our "Traces" algorithm works differently. It searches for existing paths in the database. If no identical or similar path

Table III.
Test results showing six recommended articles generated for a current article

#Test	$L < 10$	$L < 20$	$L < 50$
1	170	492	1,969
2	184	477	1,506
3	143	373	1,001
4	174	405	975
5	97	243	879
6	124	361	787
7	111	188	526
8	86	194	571
9	109	270	516
10	90	160	366
Sum	1,288	3,163	9,096

is found, the results from content based filtering are recommended. These results rely only on the abstract texts of the articles. The average recall values (see Table V) are smaller than the average precision values because more relevant items exist than the maximum number of recommended articles in one recommendation, namely six. The difference between the two algorithms concerning recall is only about 4 per cent.

The longer a path becomes, the more it might follow an existing structure and the more it will differ from the expert mind map. As a consequence the values for precision and recall decrease if the path length increases. In contrast the percentage of the hit ratio increases with the path length. Short paths are harder to predict, as they diverge more than longer paths, which tend to follow an existing link structure. There are only a few long paths stored in the database and so the prediction will become very accurate for paths that follow this stored path. The number of recommended articles for one recommendation also has a big influence on the hit ratio.

Considering the frequency-percentage of recommendations for each individual article and how long ago the articles were published, we could observe that the old articles, which were not read, or accessed recently, were recommended with less frequency by WSK than our approach.

This analysis has shown that an algorithm which works with weighted user generated paths can produce relevant results. The algorithm tries to produce recommendations which differ from an existing link structure and which better reflect a user's interests.

Conclusion

Our analysis of the user paths has shown that the users of the investigated system follow an existing link structure such as the TOC. This was performed with three tools: first an article matrix was built to observe the successive frequencies of each article, second an application to follow a user's path was created and third a graphical analysis of the gathered data of all paths was made. The article matrix showed the huge influence of the TOC. Additionally, this matrix can be used to identify the dominating link structure, because the successive articles with the highest frequencies are obvious. By analysing the users' paths the importance of the TOC was noticed again. The application with the ability to follow a user's path revealed that it is difficult to leave the strongest path, as it has such an enormous frequency. As a consequence it can be difficult for an RS to recommend articles away from this path, if based on such path data.

Table IV. Average test results: precision values

	Precision (WSK)	Precision (traces)
T < 10	0.42787	0.34364
T < 20	0.41939	0.33623
T < 50	0.39552	0.32961

Table V. Average test results: recall values

	Recall (WSK)	Recall (traces)
T < 10	0.24742	0.20515
T < 20	0.24406	0.20409
T < 50	0.23882	0.20517

During the graphical analysis more information about the structure and distribution of user paths in the system was illustrated: the dominance of the first articles in the TOC in terms of average path length, total path length and amount of paths starting from these articles was shown. This leads to the consequence that an existing link structure has to be taken into account when designing an RS using user paths. Such an RS suffers from the tendency to create recommendations identical to the strongest path (for example the TOC). To create useful recommendations this effect must be reduced by identifying and removing such misleading paths from the process of recommendation generating. Based on this analysis a combination of content based and collaborative filtering approaches was used to build a recommender algorithm for the articles within OJS.

The length of the paths plays a remarkable role in the results of the recommendation: the shorter the paths are, the more frequently they appear in the data set. For the sake of a realistic recommendation, a minimum path length must be taken into consideration. Our experiments demonstrate that paths containing more than five articles produce good results. If the paths used for the analysis are shorter, there are too many possibilities and the quality of the predictions decreases.

To validate the applied algorithm and ensure that it covers all the requirements within OJS, another approach was taken into consideration. The validation algorithm based on the well-known conditional probabilities was applied to all existing paths. Both algorithms resulted in approximately the same recall and precision. The validation algorithm resulted in slightly better values than our approach, however, the old articles that were not read or accessed recently were recommended with less probability than by our approach. In other words our algorithm considers the relevant articles independent of when they were published, whilst the validation algorithm would not meet our requirements in OJS (recommending all relevant articles, including the very old ones). This advantage could be of enormous interest for systems with the same requirements.

References

Adomavicius, G. and Tuzhilin, A. (2005), "Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 17 No. 6, pp. 734-749.

Amatriain, X., Jaimes, N., Oliver, N. and Pujol, J.M. (2011), "Data mining methods for recommender systems", in Ricci, F., Rokach, L., Shapira, B. and Kantor, P. (Eds), *Recommender Systems Handbook*, Springer, Berlin, pp. 39-71.

Baeza-Yates, R. and Ribeiro-Neto, B. (1999), *Modern Information Retrieval*, Pearson, Harlow.

Beale, R. (2007), "Supporting serendipity: using ambient intelligence to augment user exploration for data mining and web browsing", *International Journal of Human-Computer Studies*, Vol. 65 No. 5, pp. 421-433.

Bretthorst, G.L. (1990), "An introduction to parameter estimation using Bayesian probability theory", in Fougere, P.F. (Ed.), *Maximum Entropy and Bayesian Methods, Fundamental Theories of Physics*, Vol. 39, Springer, Amsterdam, pp. 53-79.

Burke, R. (2005), "Hybrid systems for personalized recommendations", in Mobasher, B. and Anand, S.S. (Eds), *Intelligent Techniques for Web Personalization*, Springer, Berlin, pp. 133-152.

Burke, R. (2007), "Hybrid web recommender systems", in Brusilovky, P., Kobsa, A. and Nejdl, W. (Eds), *The Adaptive Web*, Springer, Berlin, pp. 377-408.

Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K. and Harshman, R. (1990), "Indexing by latent semantic analysis", *Journal of the American Society for Information Science*, Vol. 41 No. 6, pp. 391-407.

Desrosiers, C. and Karypis, G. (2011), "A comprehensive survey of neighborhood-based recommendation methods", in Ricci, F., Rokach, L., Shapira, B. and Kantor, P. (Eds), *Recommender Systems Handbook*, Springer, Berlin, pp. 107-144.

Diederich, J. and Iofciu, T. (2006) in Tomadaki, E. and Scott, P. (Eds), "Finding communities of practice from user profiles based on folksonomies", *Innovative Approaches for Learning and Knowledge Sharing, EC-TEL Workshop Proceedings*, Open University, Milton Keynes, p. 288-97.

Evangelopoulos, N., Zhang, X.N. and Prybutok, V.R. (2012), "Latent semantic analysis: five methodological recommendations", *European Journal of Information Systems*, Vol. 21 No. 1, pp. 70-86.

Handelsman, J., Ebert-May, D., Beichner, R., Bruns, P., Chang, A., Dehaan, R., Gentile, J., Lauffer, S., Stewart, J., Tilghman, S.M. and Wood, W.B. (2004), "Scientific teaching", *Science*, Vol. 304 No. 5670, pp. 521-522.

Herlocker, J.L., Konstan, J.A., Terveen, L.G. and Riedl, J.T. (2004), "Evaluating collaborative filtering recommender systems", *ACM Transactions on Information Systems (TOIS)*, Vol. 22 No. 1, pp. 5-53.

Hofmann, T. (2001), "Unsupervised learning by probabilistic latent semantic analysis", *Machine Learning*, Vol. 42 Nos 1-2, pp. 177-196.

Holzinger, A. (2010), *Process Guide for Students for Interdisciplinary Work in Computer Science/Informatics*, 2nd ed., Books on Demand, Norderstedt.

Holzinger, A. (2011), *Successful Management of Research and Development*, Books on Demand, Norderstedt.

Holzinger, A., Kickmeier-Rust, M. and Albert, D. (2008), "Dynamic media in computer science education; content complexity and learning performance: is less more?", *Educational Technology & Society*, Vol. 11 No. 1, pp. 279-290.

Jin, X., Zhou, Y. and Mobasher, B. (2005), "A maximum entropy web recommendation system: combining collaborative and content features", in *ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, ACM, New York, NY, p. 612-617.

Khribi, M.K., Jemni, M. and Nasraoui, O. (2009), "Automatic recommendations for e-learning personalization based on web usage mining techniques and information retrieval", *Educational Technology & Society*, Vol. 12 No. 4, pp. 30-42.

Kim, T.-H. and Yang, S.-B. (2005), "An effective recommendation algorithm for clustering-based recommender systems", in Zhang, S. and Jarvis, R. (Eds), *AI 2005: Advances in Artificial Intelligence*, Springer, Berlin, pp. 1150-1153.

Kim, Y.S., Oh, J.S., Lee, J.Y. and Chang, J.H. (2004), "An intelligent grading system for descriptive examination papers based on probabilistic latent semantic analysis", in Webb, G.I. and Yu, X. (Eds), *AI 2004: Advances in Artificial Intelligence*, Springer, Berlin, pp. 1141-1146.

Kohavi, R. (1995), "A study of cross-validation and bootstrap for accuracy estimation and model selection", in *Proceedings of the 14th International Joint Conference on Artificial Intelligence, Morgan Kaufmann, San Francisco, CA*, p. 1137-1145.

Kreuzthaler, M., Bloice, M.D., Faulstich, L., Simonic, K.M. and Holzinger, A. (2011), "A comparison of different retrieval strategies working on medical free texts", *Journal of Universal Computer Science*, Vol. 17 No. 7, pp. 1109-1133.

L3t (2011), *L3T – Lehrbuch fuer Lernen und Lehren mit Technologien*, available at: http://l3t.eu (accessed 8 April 2013).

Landauer, T.K., Foltz, P.W. and Laham, D. (1998), "An introduction to latent semantic analysis", *Discourse Processes*, Vol. 25 Nos 2-3, pp. 259-284.

Lemire, D. and Maclachlan, A. (2005), "Slope one predictors for online rating-based collaborative filtering", *Society for Industrial Mathematics*, Vol. 5, pp. 471-480.

Linton, F. and Schaefer, H.P. (2000), "Recommender systems for learning: building user and expert models through long-term observation of application use", *User Modeling and User-Adapted Interaction*, Vol. 10 Nos 2-3, pp. 181-207.

Liu, B., Mobasher, B. and Nasraoui, O. (2011), *Web Usage Mining Web Data Mining*, Springer, Berlin.

Lops, P., Gemmis, M. and Semeraro, G. (2011), "Content-based recommender systems: state of the art and trends", in Ricci, F., Rokach, L., Shapira, B. and Kantor, P. (Eds), *Recommender Systems Handbook*, Springer, Berlin, pp. 73-105.

Lorenzi, F. and Ricci, F. (2005), "Case-based recommender systems: a unifying view", in Mobasher, B. and Anand, S.S. (Eds), *Intelligent Techniques for Web Personalization*, Springer, Berlin, pp. 89-113.

Manning, C.D., Raghavan, P. and Schutze, H. (2008), *Introduction to Information Retrieval*, Cambridge University Press, Cambridge.

Manouselis, N., Vuorikari, R. and Van Assche, F. (2007), "Simulated analysis of MAUT collaborative filtering for learning object recommendation", in *Proceedings of the Workshop on Social Information Retrieval in Technology Enhanced Learning (SIRTEL 2007)*, available at: http://infolab-dev.aua.gr/sirtel2007/papers.htm (accessed 8 April 2013).

Manouselis, N., Drachsler, H., Vuorikari, R., Hummel, H. and Koper, R. (2011), "Recommender systems in technology enhanced learning", in Ricci, F., Rokach, L., Shapira, B. and Kantor, P. (Eds), *Recommender Systems Handbook*, Springer, Berlin, pp. 387-415.

Marek, K. (2011), "Web analytics overview", *Library Technology Reports*, Vol. 47 No. 5, pp. 5-10.

Mladenic, D. (1999), "Text-learning and related intelligent agents: a survey", *IEEE Intelligent Systems*, Vol. 14 No. 4, pp. 44-54.

Motschnig-Pitrik, R. and Holzinger, A. (2002), "Student-centered teaching meets new media: concept and case study", *IEEE Journal of Educational Technology & Society*, Vol. 5 No. 4, pp. 160-172.

Nadolski, R.J., Van den Berg, B., Berlanga, A., Drachsler, H., Hummel, H., Korper, R. and Sloep, P. (2009), "Simulating light-weight personalised recommender systems in learning networks. A case for pedagogy-oriented and rating-based hybrid recommendation strategies", *Journal of Artificial Societies and Social Simulation*, Vol. 12 No. 1, pp. 1-3.

Niemann, K., Scheffel, M., Friedrich, M., Kirschenmann, U., Schmitz, H.C. and Wolpers, M. (2010), "Usage-based object similarity", *Journal of Universal Computer Science*, Vol. 16 No. 16, pp. 227-290.

Pazzani, M.J. and Billsus, D. (2007), "Content-based recommendation systems", in Brusilovsky, P., Kobsa, A. and Nejdl, W. (Eds), *The Adaptive Web*, Springer, Berlin, pp. 325-341.

Piwik (2011), "Piwik – web analytics", available at: http://de.piwik.org (accessed 8 April 2013).

Porter, M.F. (1980), "An algorithm for suffix stripping", *Program*, Vol. 14 No. 3, pp. 130-137.

Prince, M.J. and Felder, R.M. (2006), "Inductive teaching and learning methods: definitions, comparisons, and research bases", *Journal of Engineering Education*, Vol. 95 No. 2, pp. 123-138.

Ramos, J. (2003), "Using TF-IDF to determine word relevance in document queries", in *Proceedings of the First Instructional Conference on Machine Learning*, available at: www.cs.rutgers.edu/~mlittman/courses/ml03/iCML03/papers/ramos.pdf (accessed 9 April 2013).

Schafer, J.B., Konstan, J.A. and Riedl, J. (2001), "E-commerce recommendation applications", *Data Mining and Knowledge Discovery*, Vol. 5 Nos 1/2, pp. 115-153.

Schafer, J.B., Frankowski, D., Herlocker, J. and Sen, S. (2007), "Collaborative filtering recommender systems", in Brusilovky, P., Kobsa, A. and Nejdl, W. (Eds), *The Adaptive Web*, Springer, Berlin, pp. 291-324.

Snowball (2011), available at: http://snowball.tartarus.org (accessed 8 April 2013).

Szomszor, M., Cattuto, C., Alani, H., O'Hara, K., Baldassarri, A., Loreto, V. and Servedio, V.D.P. (2007), "Folksonomies, the semantic web, and movie recommendation", in *Proceedings of the Workshop on Bridging the Gap between Semantic Web and Web 2.0 at the 4th European Semantic Web Conference*, available at: http://eprints.soton.ac.uk/id/eprint/264007 (accessed 8 April 2013).

Tang, T. and McCalla, G. (2004), "On the pedagogically guided paper recommendation for an evolving web-based learning system", in *Proceedings of the 17th International FLAIRS Conference, American Association for Artificial Intelligence*, p. 86-91, available at: http://aaaipress.org/Papers/FLAIRS/2004/Flairs04-019.pdf (accessed 9 April 2013).

Tang, T. and McCalla, G. (2005), "Smart recommendation for an evolving e-learning system: architecture and experiment", *International Journal on E-Learning*, Vol. 4 No. 1, pp. 105-129.

Willinsky, J. (2005), "Open Journal Systems: an example of open source software for journal management and publishing", *Library Hi Tech*, Vol. 23 No. 4, pp. 504-519.

Yoo, J.-H., Ahn, K.-S., Jun, J. and Rhee, P.-K. (2004), "A recommendation system for intelligent user interface: collaborative filtering approach", in Negoita, M., Howlett, R. and Jain, L. (Eds), *Knowledge-Based Intelligent Information and Engineering Systems*, Springer, Berlin, pp. 869-879.

About the authors

Behnam Taraghi is a PhD student and Junior Researcher in the Department of Social Learning at Graz University of Technology in Austria. His thesis is related to recommender systems in e-learning systems, open educational resources and personal learning environments. He has developed personal learning environments for Graz University of Technology and has published several papers in international conferences and workshops in this regard.

Martin Grossegger is a Master's student at Graz University of Technology. His thesis addresses the development of recommendations for Open Journal Systems.

Martin Ebner is Head of the Department of Social Learning at Graz University of Technology and responsible for all university-wide e-learning activities. He is an Associate Professor of Media Informatics at the Institute for Information Systems and Computer Media. His research focuses on e-learning, mobile learning, learning analytics, social media and the use of Web 2.0 technologies for teaching and learning.

Andreas Holzinger is Head of the Human-Computer Interaction Research Unit at the Institute for Medical Informatics, Statistics and Documentation at the Medical University of Graz. He is also an Associate Professor of Informatics at the Institute for Information Systems and Computer Media at Graz University of Technology. His research interests include knowledge discovery, data mining and applications of computational intelligence. Dr Holzinger has been a Visiting Professor in Berlin, Innsbruck, London, Vienna and Aachen.

Andreas Holzinger is the corresponding author and can be contacted at: a.holzinger@tugraz.at