The Role of Semantic Relevance in Dynamic User Community Management and the Formulation of Recommendations*

Nick Papadopoulos[1] and Dimitris Plexousakis[1,2]

[1] Institute of Computer Science, Foundation for Research and Technology - Hellas
P.O. Box 1385, GR-71110, Heraklion, Greece
{npap,dp}@ics.forth.gr
[2] Department of Computer Science, University of Crete
P.O. Box 2208, GR-71409, Heraklion, Greece

Abstract. In recent years, an increasing interest in recommendation systems has emerged both from the research and the application point of view and in both academic and commercial domains. The majority of comparison techniques used for formulating recommendations are based on set-operations over user-supplied terms or internal product computations on vectors encoding user preferences. In both cases however, the "identical-ness" of terms is examined rather than their actual semantic relevance. This paper proposes a recommendation algorithm that is based on the maintenance of user profiles and their dynamic adjustment according to the users' behavior. Moreover, this algorithm relies on the dynamic management of communities, which contain "similar" and "relevant" users and which are created according to a classification algorithm. The algorithm is implemented on top of a community management mechanism. The comparison mechanism used in the context of this work is based on semantic relevance between terms, which is evaluated with the use of a glossary of terms.

1 Introduction

In recent years, an increasing interest in recommendation systems has emerged both from the research and the application points of view, and in both academic and commercial domains. Many online "e-shops" have adopted recommendation techniques to recommend new items to their customers based on a logged history of previous purchases or transactions. The majority of existing recommendation systems does not adequately address the information filtering needs of their users. A principal requirement main reason for such systems is the employment of a mechanism for filtering

* This research has been supported by project CYCLADES (IST-2000-25456): An Open Collaborative Virtual Archive Environment.

A. Banks Pidduck et al. (Eds.): CAISE 2002, LNCS 2348, pp. 200-215, 2002.

information that is available through the Web, and dynamically adjust it to the user's needs and interests [1]. Information filtering and its subsequent tailoring to the user's interests constitute the ultimate goals of recommendation systems. Modern recommendation systems mainly base their functionality on one of two approaches, in order to recommend a document or product to a user. These two approaches are *content-based filtering* and *collaborative-filtering* [2,3,6,7] respectively. The adoption of the former or the latter depends on the type of information that a system aims to provide to its users.

In the *content-based* approach, the system filters information (typically a document) and aims to provide recommendations based on the contents of this document. Hence, given their preferences, recommendations for documents that are relevant to their interests are forwarded to the users [2,3]. Such an approach has been mainly adopted by information retrieval [7,8] and machine learning systems [9,10,11]. For instance, in the case of text documents, recommendations are provided based on the degree of matching between the content of a document and a user's profile. The user's profile is built and maintained according to an analysis that is applied to the contents of the documents that the user has previously rated [11,12]. The user's ratings in combination with a mechanism for obtaining the feature terms of a document [2,3,4,5] are the only requirement of these systems.

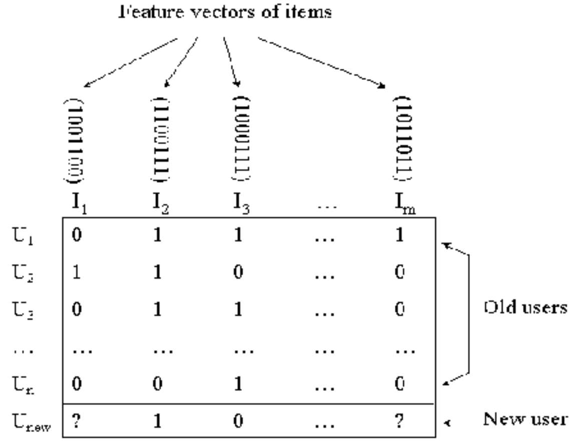
On the other hand, systems adopting the *collaborative-filtering* approach, aim to identify users that have relevant interests and preferences with a particular user. Thereafter, the documents that these users prefer are recommended to that particular user. The idea behind this approach is that, it may be of benefit to one's search for information to consult the behavior of other users who share the same or relevant interests and whose opinion can be trusted [4]. Such systems take advantage of the documents that other users have already "discovered" and rated. In order for these systems to be able to "detect" the relevance between users, there must be a comparison mechanism, which in this case too, is the usage of user profile information [11,12]. The requirement imposed by such systems is that new users have to rate some documents, so that profiles can be built for them [2,3,4,5].

If these two approaches are applied separately, they present crucial disadvantages and suffer from specific problems. In order for these problems to be dealt with, hybrid approaches need to be devised. Motivated by these issues, this paper presents such a hybrid approach in which the semantic (and not solely lexical) relevance of terms in dynamically maintained user profiles is exploited for the formulation of recommendations.

The rest of the paper is organized as follows. Section 2 presents a comparison between content-based filtering and collaborative filtering as well as the benefits of hybrid approaches. Section 3 describes the proposed algorithm for formulating recommendations. Section 4 presents preliminary experimental results. Section 5 summarizes our contribution and draws directions for further research.

2 Comparing Content-Based and Collaborative Filtering

A representation of the data available in a recommendation system is depicted in Figure 1. Data are represented by means of a matrix whose rows correspond to users and columns correspond to items. Each item is associated with a feature vector. A feature vector can represent an item in different ways. For example we can consider the case of text documents where the (binary) feature vector denotes whether a term is presented or not in a certain document. This means that '1' would denote the appearance of a certain term in the document while '0' would denote the opposite. The elements of this matrix are binary, indicating whether a user has rated an item.



	I_1	I_2	I_3	…	I_m
U_1	0	1	1	…	1
U_2	1	1	0	…	0
U_3	0	1	1	…	0
…	…	…	…	…	…
U_n	0	0	1	…	0
U_{new}	?	1	0	…	?

Fig. 1. Representation of data in recommendation systems

The main difference between content-based and collaborative-filtering systems is found in the way they use the available data. Content-based system use only the feature vector of the item that a user has rated (columns with '1') and ignore the items that other users have rated. Therefore, they utilize only the last row of the matrix and the feature vector of the items, ignoring the data in the first *n* rows [4,5].

Systems adopting collaborative filtering on the other hand, formulate their recommendations based solely on the ratings of other users that are considered relevant to a given user. Thus, in this case, only the data within the matrix and not the feature vectors are used. These systems identify the users, which are interested in the items that a given user is also interested in, by checking the columns with '1' in the last row and the preceding *n* rows with '1' in the same columns [2,4].

Both approaches have certain shortcomings. One problem with content-based systems is that, in some applications, it is often difficult, and in others even infeasible, to describe the items as vectors or as some other form (such items for example are images, movies and general multimedia items). As a result, we are not able to compare the relevance of such items; we can only check their identity. A second problem is over-specialization, meaning that recommendations can only be made for items that are similar to what a user has already rated [2,3,4,5]. Finally, a less severe problem[1] is that each new user starts with an empty profile. In order for the recommendation system to achieve a high degree of accuracy, the user must rate a number of items. The users' ratings constitute the only factor that influences the subsequent performance of the system [2,4].

Collaborative filtering systems avoid the aforementioned problems of content-based systems but suffer from others. Specifically, an item can be recommended to a user, even if it is not so similar to those the user has already rated. Recommendations for this particular item are based on the preferences of other "relevant" users. In these systems, rather than computing the similarity of items, the similarity of users is computed [2]. However, if a new item appears in the system's database, the system cannot consider it and thus it will not be able to provide recommendations for that until another user rates it. Furthermore, for a user whose preferences and, thus, profile, are "unusual" compared to the rest of the users, there will probably not be any good recommendations, since there will not be any other relevant users [2,4,5].

It is becoming obvious that techniques that base their functionality on either of these approaches or both but apply them separately present crucial disadvantages. In order for these problems to be dealt with, hybrid approaches combining techniques from both types of systems are adopted. Hence, content-based techniques are usually used in order to build and maintain users' profiles. Then, collaborative filtering techniques are applied on these profiles, so that the relevant users can be identified and can thereafter form a community. Furthermore, collaborative filtering techniques are applied on these communities in order for recommendations to be provided. Each such community can also be considered as a single user and it is feasible thereafter for recommendations to be provided to an entire community.

3 User Classification and Recommendations

In this section, a general recommendation algorithm that adopts a hybrid approach is presented. This algorithm is based on the maintenance of a user-profile for each user and on the dynamic adjustment of this profile to the user's behavior. Moreover, the algorithm is based on the dynamic management of communities that encompass "relevant" users [12,13]. The classification of users into communities and the identification of "relevant" users are based on the examination of users' profiles.

The goal is to devise techniques that are applicable for both content-based and collaborative filtering. The maintenance of the profiles is performed according to a user's behavior, always taking into account the user's ratings on documents [1,13,14]. Hence, techniques from content-based systems are used in order for the documents to be analyzed and the user profiles to be properly updated. Thereafter, the algorithm for user classification into communities is activated with the result of classifying relevant users into the same community. Subsequently, techniques from collaborative filtering are applied on the derived communities, in order for recommendations to be provided to their members.

[1] Many do not consider this as a problem at all.

We believe that the suggested method deals well with the problems of the recommendations systems. The only requirement imposed to users is that they must rate some documents, so that a profile can be build for them. The larger the number of documents a user rates, the better and more adequate are the recommendations that this user will receive. We also claim that, regardless of a user's distinctiveness, the classification algorithm will classify that user into the appropriate communities. A user must have a quite idiomorphic profile, so as to be left out of all the communities.

The above claim is founded on the fact that the comparison mechanism of the algorithm bases its functionality not on "identicalness" of the users but on *semantic relevance* of them. To be more specific, users' semantic relevance implies terms' semantic relevance and terms' semantic relevance is evaluated according to various criteria (e.g., synonymity, subsumption) of semantic relevance. Such criteria for evaluation can be provided by dictionary of terms such as WordNet [26]).

3.1 Characterizing User Interests: User Profiles

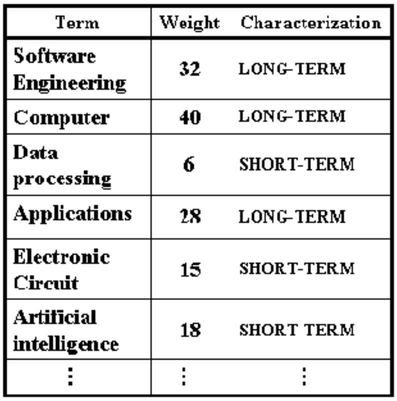
The interests and the preferences of each user are expressed through terms that participate into the user's profile that is built and dynamically maintained. The next paragraphs describe user profiles and the way the profiles are maintained.

Each term in a user profile expresses, in a certain degree, the likes of a particular user. A weight is associated to each term for denoting this degree. This weight indicates the importance of the term in the user's interests [1,15]. The weights dynamically change as a result of the user's behavior, and so the importance of that term in the user's interests changes as well [1,15,16,17,18,19,20,21,22]. Moreover, if the weight of a term becomes zero, that term is removed from the profile.

According to their associated weights, all terms in a profile are classified into two groups: one that contains "heavier" terms (called "LONG-TERMS"), and one containing the "lighter" ones (called "SHORT-TERMS"). This classification / characterization denotes whether a term expresses the user's interests at a high or low degree respectively [1,21,23]. As an example, Figure 2 depicts the weights and respective characterizations of terms relevant to Computer Science bibliography for a weight threshold of 20. This characterization is closely related to the weights, meaning that whenever the weight changes, according to the user's behavior, this characterization may also change accordingly. If the weight of a term exceeds a certain threshold then the characterization will automatically change to "LONG-TERM" and if it falls below that threshold it will change to "SHORT-TERM". The usage of that characterization is mainly adopted for practical reasons and is particularly helpful in order for the terms to be visibly distinguishable.

Every time a user rates a document, the user profile is properly adjusted according to the relevance the document has with her interests. This means that, depending on whether the rating is positive or negative, the weight of some terms (and consequently the characterization) will change and either increase or decrease accordingly. Moreover, if the rating is positive then some new terms might be inserted into profile.

For example, if the user rates positively a document, which contains the term 'Artificial Intelligence', then that term's weight will increase. If this happens again and again then the weight will exceed the threshold and thus the term will be considered as a "LONG-TERM". More details are presented in the following subsections.



Term	Weight	Characterization
Software Engineering	32	LONG-TERM
Computer	40	LONG-TERM
Data processing	6	SHORT-TERM
Applications	28	LONG-TERM
Electronic Circuit	15	SHORT-TERM
Artificial intelligence	18	SHORT TERM
⋮	⋮	⋮

Fig. 2. A user profile (with threshold of change equal to 20)

3.1.1 Automatic Profile Maintenance

The maintenance of profiles is carried out by applying a gradual decrement of all term weights in a profile. This decrement is performed without being noticed by the user and it is applied periodically at appropriate times. The exact time intervals between successive applications of weight modification may depend on the application in question. It is obvious that automatic profile maintenance cannot be applied, say every day, because a user may not carry out any action for the time span of a day and so it would be unwarranted to change the profile in any way. Thereafter, it is important to define precisely the appropriate time points, which in our case are considered to be the time points at which a user acts and which are expressed by the ratings of the documents.

In the previous subsection, we mentioned that whenever the user rates some documents, an adjustment of the profile is performed and either new terms are inserted into profile or the weights of some terms are appropriate updated. The terms inserted into the profile, are the “characteristic terms” of the rated documents. “Characteristic terms” are meant to be those terms of the documents, which are produced by a textual analysis that is performed so that the “stop words” (terms which do not have any special meaning for the document) are discarded [24,25].

In addition to the above procedure, which is directly related to the documents that a user has rated, a further adjustment of the profile is deemed necessary. This adjustment is based on continuous and gradual decrement of the weight of each term, independently of the rated documents. In this manner, all terms automatically lose some degree of importance as far as user interests are concerned. The main goal of this procedure is to achieve better and more “fair” distinction of the terms in the “LONG-TERM” and “SHORT-TERM” lists. Hence, as long as a term is not affected by the user’s ratings, its weight decreases and so this term will soon be considered as a “SHORT-TERM”, meaning that it loses some of the importance it has among the rest of the user’s interests.

On the other hand, a term that is usually affected positively by the user's ratings has a high weight value and thus, if the weight has exceeded a certain value, it is considered "LONG-TERM". This means that user's likes are well expressed by that term and due to its high weight it will not be affected immediately by the automatic profile maintenance. Consequently, the more important a term is the more it will take for it to lose that importance. Conversely, the less important a term is the less it will take for it to lose that importance.

By applying this procedure for automatic profile maintenance, an additional goal is achieved. The weights of terms which are not affected by the user's ratings, meaning that they probably do not have any importance in her interests, gradually decrease until they become zero and are removed from the profile. Hence, by applying automatic profile maintenance, terms which no longer express the likes of a user, are automatically and without that user's interference removed from her profile.

3.2 Description of the Algorithm

The algorithm, which is proposed in this paper, forms the basis for a dynamically adjustable to a user's behavior recommendation algorithm. The goal of this algorithm is the users' classification into communities according to their relevant interests.

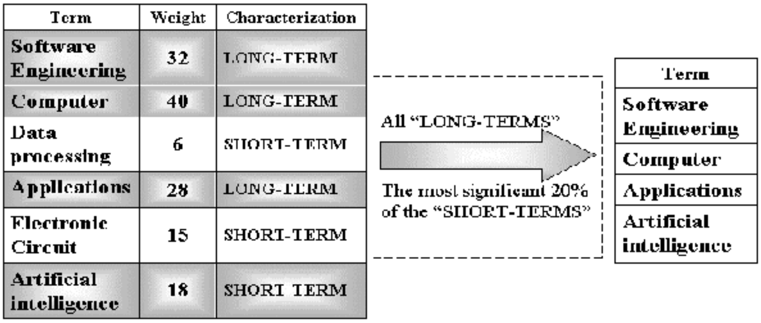
3.2.1 Algorithm's Requirements

The algorithm bases its functionality on user profiles. It expects that the terms in those profiles are associated with a weight, which denotes the importance of that term. Also, it expects that the terms in a profile can be distinguished into two groups, "LONG-TERMS" and "SHORT-TERMS", as described in section 3.1 above.

3.2.2 Algorithm's Operational Procedure

Given the fact that each user can be described by a set of terms, checking users' relevance is reduced to checking the relevance of their associated sets of terms. Hence, the algorithm must consider the creation of such a set per user. In order for these sets to be created, all the "LONG-TERMS" are selected at first and are inserted into that set. All "LONG-TERMS" are selected, without any discrimination, because they express the user's likes at a high degree and it is considered fair to select them all. In addition to this selection, a portion of the rest of terms, which are characterized as "SHORT-TERMS", is selected. This is a percentage value specified as a parameter for the algorithm *(selection percentage)* and thus it can vary. The portion of the "SHORT-TERMS" selected comprises the most important terms according to the weights. Eventually, a set of terms, which contains all the "LONG-TERMS" of the profile and a portion of the "SHORT-TERMS", is created for each user.

For instance, consider the user profile shown in the left part of Figure 3 where the terms that are eventually going to be selected have been emphasized. The right part of Figure 3 shows the set of terms that will be created by the algorithm. In this example the selection percentage equals 20% and one term is finally selected, since there are only 3 "SHORT-TERMS". The "SHORT-TERM" which is selected is the one with the greatest weight (in our example it is 'Artificial Intelligence' with weight 18).



Term	Weight	Characterization
Software Engineering	32	LONG-TERM
Computer	40	LONG-TERM
Data processing	6	SHORT-TERM
Applications	28	LONG-TERM
Electronic Circuit	15	SHORT-TERM
Artificial intelligence	18	SHORT-TERM

Term
Software Engineering
Computer
Applications
Artificial intelligence

Fig. 3. Selection of terms from a user profile with 20% selection percentage

The above procedure for term selection is defined so that the terms, which more accurately express the likes and the interests of each user be selected. Since the "LONG-TERMS" express the user's likes at a high degree, they are all selected without any distinction. Moreover, a portion of the most significant "SHORT-TERMS" is also selected. This is mainly done so that users with no "LONG-TERMS" in their profile can be taken into account as well.

Concerning the selection percentage, this is adopted so that the algorithm acts in a fair manner for all users, independently of the terms in each profile. That means that the more "SHORT-TERMS" are contained in a user profile, the more terms will be selected for the set, which will be created for that user by the algorithm. So, for the users who have a huge profile, which statistically implies many "SHORT-TERMS" in it as well, a lot of terms will finally be selected in addition to the "LONG-TERMS", which are selected in their totality in any case.

The previous distinction implies that users with big profiles are described in a more "detailed" manner as compared to users with small ones. Besides, big profiles usually reveal users with high activity and behavior, as accrued by their ratings. Hence, it would be fair for them to participate in the algorithm's operational procedure with more terms. On the other hand, in order for a user with a small profile, and thus low activity, not to be excluded at all, they can participate too but with fewer terms.

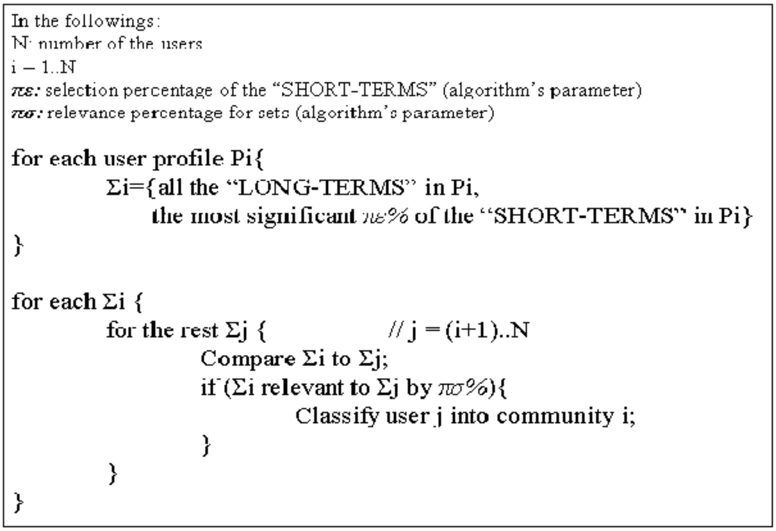
Obviously, the set of terms created for each user is a subset of that user's profile. Thereafter, these sets are examined and compared so that the ones with relevant terms can be identified. If two or more such sets are found, then the respective users are considered relevant and they are grouped together in the same community.

The following figures depict the algorithm in pseudocode (fig. 4) and a graphical representation of the algorithm's operational procedure (fig. 5).

3.2.3 User Comparison and Relevance

The comparison of users is performed according to the sets of terms created by the algorithm. Again, a percentage is used as a parameter of the algorithm *(relevance percentage)* to denote how similar two sets must be so as to be considered relevant. Specifically, given two sets, say A and B, and a relevance percentage, say π, A and B are considered relevant if one of the following two conditions hold:

- *π%* of the terms in set A is relevant to some of the terms in B
- *π%* of the terms in set B is relevant to some of the terms in A



```
In the followings:
N: number of the users
i = 1..N
πs: selection percentage of the "SHORT-TERMS" (algorithm's parameter)
πσ: relevance percentage for sets (algorithm's parameter)

for each user profile Pi{
        Σi={all the "LONG-TERMS" in Pi,
            the most significant πs% of the "SHORT-TERMS" in Pi}
}

for each Σi {
        for the rest Σj {              // j = (i+1)..N
                Compare Σi to Σj;
                if (Σi relevant to Σj by πσ%){
                        Classify user j into community i;
                }
        }
}
```

Fig. 4. Algorithm in pseudocode

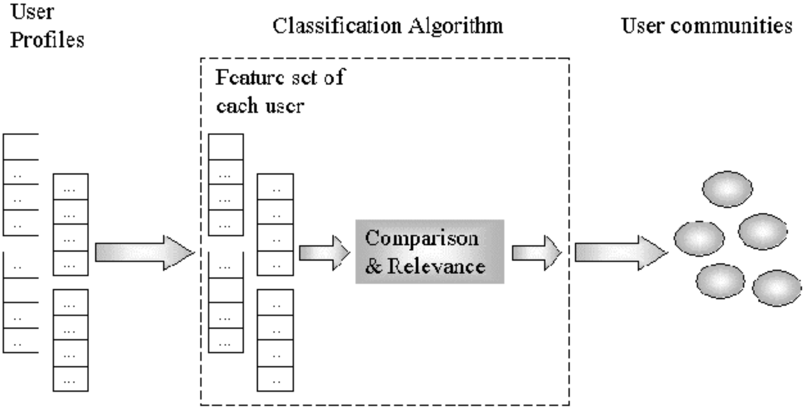


Fig. 5. Graphical representation of the algorithm's operational procedure

Consequently, in order for two sets to be considered relevant, it suffices for these sets to be relevant upon a percentage of their terms. In order for these conditions to be evaluated, each term in one set must be compared to each term of the other. Of course this induces a latency, which is partly avoided if set operations are used for checking the similarity of two sets. However, we believe that the degree of relevance that is finally achieved among sets, and thereafter among users, results in high-quality recommendations to the users. This quality of recommendations balances the drawback of the high cost of term comparisons. Furthermore, one can select the invocation times of the algorithm so that it does not become a burden on system performance. For example, it could be activated during certain periods of time (low user 'traffic-jam') and possibly on a snapshot of the system's data.

As it has become apparent from the preceding discussion, it is of great importance to be able to decide when two terms might be considered relevant and how someone could come to that conclusion. It is mentioned that the algorithm uses semantic relevance and in order for this semantic relevance to be examined, a hierarchical semantic relevance model is adopted. That means that, whenever a term is included in another term's thematic domain or it expresses a notion which is "equal to " or "subsumed by" the notion that the other term expresses, then these two terms may be considered relevant. The concept of "equality" can be defined in terms of the synonymity of terms or in terms of the same thematic domain, whereas the notion of "subsumption" can de defined in terms of the hyponyms of a term.[2] Such a hierarchical semantic model can be provided by thesauri or dictionaries of terms, such as WordNet [26]. It should be mentioned here that it is possible to specify whether or not semantic relevance will be applied by the algorithm; it is the third parameter of the algorithm. So the algorithm is capable of functioning in two ways, one that employs semantic relevance is applied and another that doesn't.

Let now examine these two functional ways by means of a simple example. Consider the "representative" sets for two users, which are the ones that were created during the algorithm's operational procedure and will form the base for user comparison and relevance. These sets are depicted in Figure 6 in which the bold line separates "LONG-TERMS" from "SHORT-TERMS" and the common terms are emphasized. If the relevance of these sets is examined by applying set intersection, then it is obvious that these sets are not highly relevant since the common terms are really few. Specifically, since there are only three common terms to these sets, this means that user 1 is relevant to user 2 by 3/18 = 16,67% and user 2 is relevant to user 1 by 3/12 = 25%. Alternatively, if semantic relevance is applied, then almost all the terms in the second user's set are related to some terms in first user's set (fig. 7). Now user 2 is relevant to user 1 by 10/12 = 83,33% (25% previously) and user 1 is relevant to user 2 by 9/18 = 50% (16,67% previously).

3.2.4 Algorithm's Complexity

In this section, we assess the complexity of the proposed algorithm. Assume that each profile contains l "LONG-TERMS" and s "SHORT-TERMS", hence it contains $l+s$ terms in total. Also assume that the total number of users is n.

It has already been mentioned that during the algorithm's operation some terms are selected from each user profile and a set of terms is created for each user. This set of terms represents the user in the rest of the procedure. These sets are created by including the l "LONG-TERMS", and a percentage, π, of the s "SHORT-TERMS". Eventually, each set contains $l+s*\pi$ terms. Hence, the term selection requires $n*(l+s*\pi)$ operations.

[2] A hyponym of a term is defined to be a term that expresses a more specific concept.

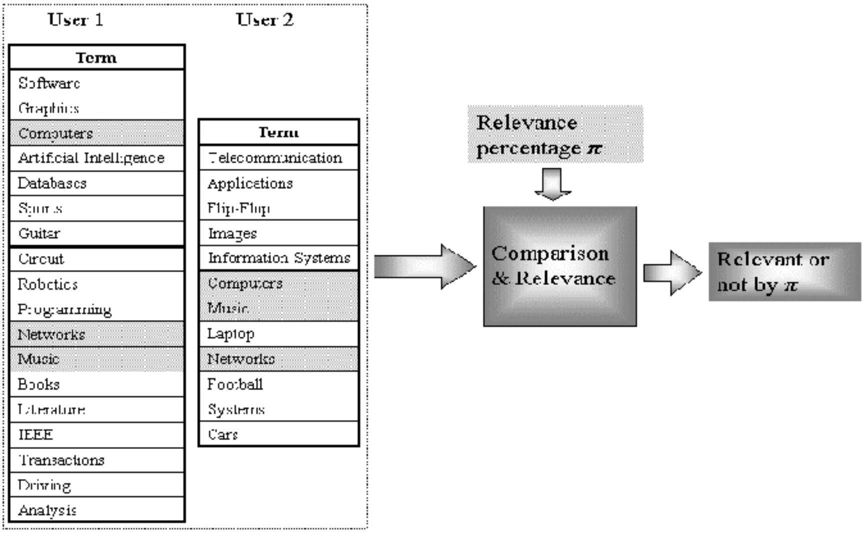


Fig. 6. Comparing two users

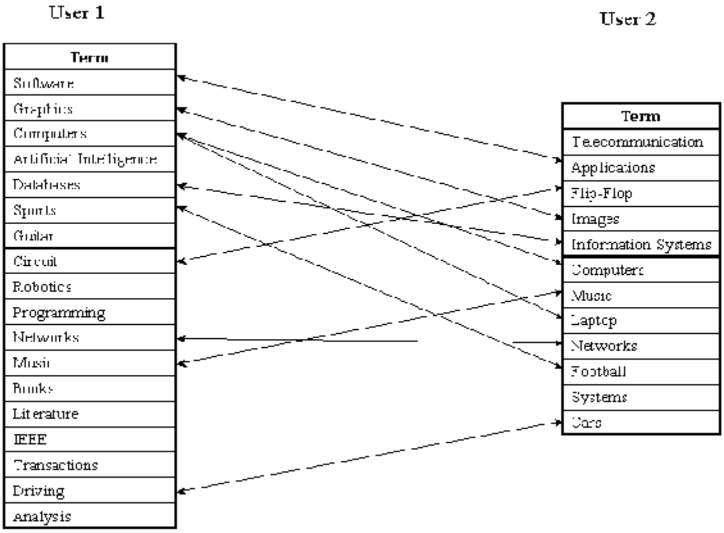


Fig. 7. Term relevance of two sets by applying semantic relevance

The algorithm examines all those sets, which on average contain $t=l+s*\pi$ terms. Specifically, all the terms of each set are compared to all the terms of the rest of the sets, where the total number of such sets is n. Each term in the first set is compared to all the terms in the second, requiring $t*t=t^2$ operations. After that, the first set is compared to the third, so an overall number of $t^2*t=t^3$ operations is required. The same procedure is performed for all n sets. As a result, in order for the first set to be examined, an overall number of t^n operations is required.

Similarly, the second set will be compared to all the remaining sets, but now the number of sets for comparison is $n-1$, since the first set has already been examined. Hence, a total number of t^{n-1} operations is required. For the third set, an overall number of t^{n-2} operations is required and so on. In total, the number of operations that are required equals: $t^n+t^{n-1}+...+t^2$ or $t^2+t^3+...+t^n$. This can be represented by the sum of a geometric sequence: $\boldsymbol{\Sigma=a_1*(L^n-1)/(L-1)}$, where $\boldsymbol{a_1=t^2}$ and $\boldsymbol{L=t}$. Hence, the total number of operations that are required equals $\boldsymbol{\Sigma=t^2*(t^n-1)/(t-1)}$.

Hence, that algorithm's complexity is $\mathbf{O}(t^n)$. This is of course quite a large figure for a large number of users. This is due to the exhaustive comparison that is being performed, in order to make sure that all the terms in all sets are examined. All the users must be exhaustively examined and compared to one another, in order for the relevant ones to be found and grouped together in communities. The following section presents preliminary performance results on the algorithm's implementation.

4 Experimental Results

We have performed some preliminary experiments so as to get a notion of the algorithm's performance. In all executions the parameters of the algorithm are the same, while the number of users increases. Particularly, the selection percentage is set to 20%, the relevance percentage to 30% and the function mode is set so as semantic relevance is applied.[3] The larger a profile is, the higher the relevance percentage should be, ranging between 20%-40% or maybe slightly more at times. Each set created by the algorithm contains about 60 terms.

As mentioned before, the algorithm's complexity is $\mathbf{O}(t^n)$, which means that adding a new user causes the execution time to rise geometrically. Figure 8 depicts a graph that shows the execution time required in respect of the users. As the number of users increases the execution time of the algorithm rises.

Remarkably, we notice that in a particular execution (for nine users) the execution time instead of rising, it falls off. This is quite noteworthy and it is worth explaining.

We have seen that the algorithm exhaustively compares all the users. Thus, the first user is compared to all the others, the second to the rest of them, and so on. If two or more users are considered to be relevant then they are grouped together into the same community. But, in that way, if the first user has many and various likes and interests, then there is a high possibility for her to be considered relevant to many other users. Therefore, all these users will be grouped together, and thus they will be excluded from the rest of the procedure. That leads to considerable reduction of the users, which must be furthermore compared by the algorithm.

The above analysis reveals a methodology, which could be followed in order for the execution time to be reduced. If the profiles were sorted in ascending order according to their size prior to the execution of the algorithm, this would result in the aforementioned situation where users with large profiles will be examined first.

[3] The selection of these values was based on experiments as well.

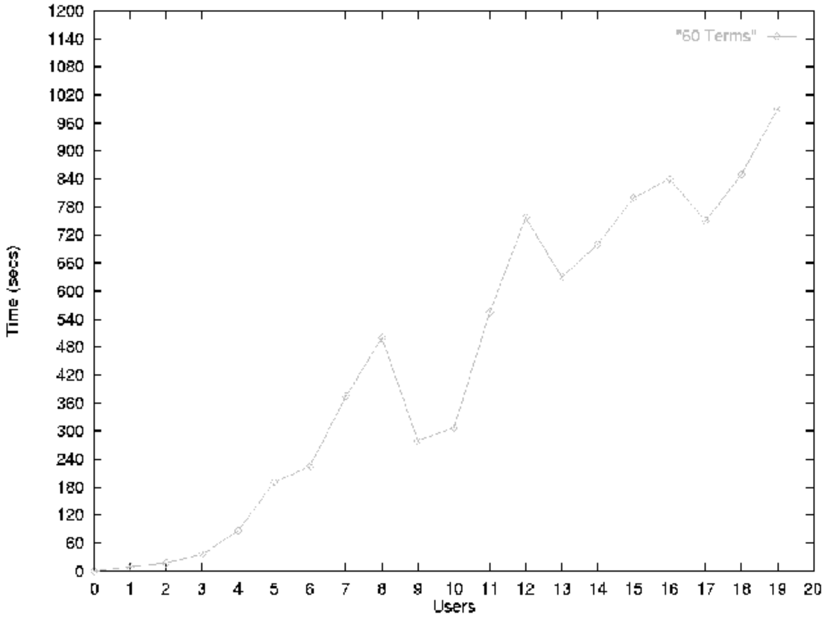


Fig. 8. Execution time for profiles with 60 terms

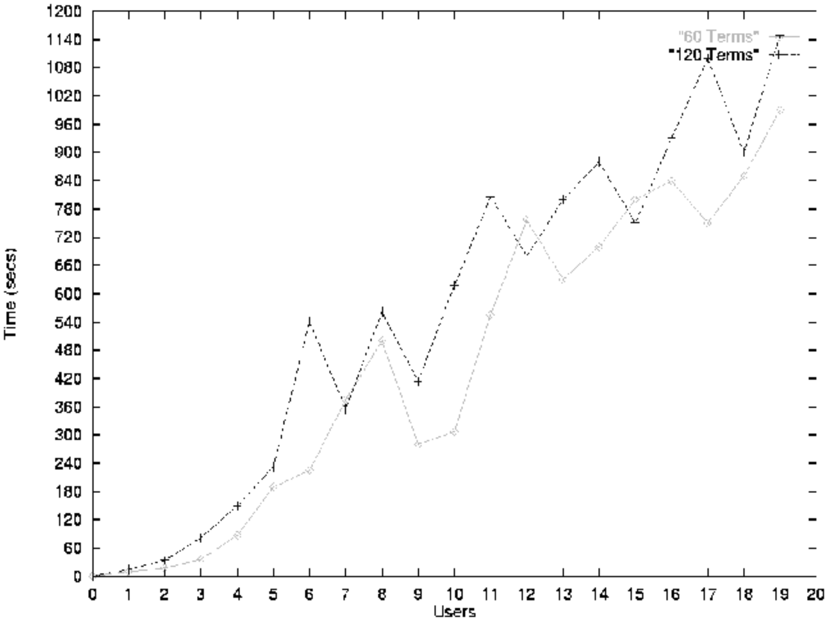


Fig. 9. Execution time for profiles with 60 and 120 terms

Figure 9 depicts the previous graph in comparison to a new one, which shows the execution time of the algorithm under the same circumstances as previously, but with twice the number of terms in the sets. Execution times are higher this time, but it is remarkable that the phenomenon, which was described above, is occurring more often. That is expected since user profiles are almost doubled and thus users have more and different likes.

5 Conclusions and Further Research

This paper proposes an algorithm that examines and compares all user profiles with the goal of classifying them into communities. In order for this to be achieved, all the sets that are created for each user are examined. The operation of the algorithm ensures that a user may participate in more than one community, which is really desirable especially for users with a great variety of interests and thus big profiles. Additionally, a user might not participate to any community, as is the case for new users, who haven't provided enough ratings so that a profile can be built for them, and for users with idiomorphic likes and interests, who are not relevant to any other user. In a nutshell, after the algorithm terminates, a user might belong to none, one or more communities.

It is also possible that two or more users participate in the same communities. But they are included in each one of the communities because different interests were considered for each of the users, or because of their relevance to different users. This is more probable to happen in the case of numerous users with big profiles and thus many terms in them. Since the algorithm is based on the selection of a percentage of terms, the larger the number of terms in the term sets, the higher is the possibility for these sets to be considered relevant concerning various thematic domains and, thus, more communities are likely to be created. Hence, the respective users will all be classified in these communities, which is of course desirable so that the users are able to receive recommendations from all of them.

The algorithm adequately addresses the case of idiomorphic users that may exist and classifies them in the appropriate communities. If there is a possibility for two users to be considered relevant, then this will be done so by the application of the algorithm, since user relevance is based on set relevance. Set relevance is in turn based on the semantic relevance of their terms. So, even though two particular terms have no apparent (i.e., lexical) similarity, these terms may be closely related and may sometimes express the same notion. For example, consider the terms 'car' and 'motor vehicle'. These terms semantically express the same concept since the term 'car' belongs to hyponyms of the term 'motor vehicle'.

As far as the complexity of the algorithm is concerned, we argue that it is balanced by the expected high quality and accuracy of the recommendations that are provided and are based on the user classification into communities. Our preliminary experimental results show that as new users are added the execution time of the algorithm rises, but also reveal a methodology, which could be followed in order for these execution times to be reduced.

We are currently investigating aspects of reducing even more the execution time both “directly”, by means of implementation, and “indirectly”. The algorithm creates the user communities from scratch and whenever it is applied the old communities are destroyed. This wouldn’t be desirable, especially if new users could be accommodated to the existing communities. Therefore, a “differential” classification algorithm could be applied only on new users, in order for them to be grouped to existing communities. This may be performed by comparing new users to existing communities and is feasible due to the fact that each community has its own profile, which is produced as the union of the profiles of all users in community. So each community could be treated as a single user and thus compared to other users. Such an algorithm is expected to have shorter execution time and could be executed more often than the algorithm presented in this paper, which could be executed rarely, so as to cover the cases in which the likes of some users already in a community have changed and are no longer relevant to those of other users in the community.

Finally, one could investigate the case where users with huge profiles (and thus with various and different likes) would act as anchors and organize communities around them. That would result in having incoherent sub-communities inside a community. To deal with these cases a pre-processing of all profiles would be performed so as to distinguish between the various sub-profiles of a user. The algorithm then could be applied on these sub-profiles.

References

1. D. H. Widyantoro, T. R. Ioerger, and J. Yen. “Learning User Interest Dynamics with, Three-Descriptor Representation”. *Journal of the American Society for Information Science (JASIS)*, 2000.
2. M. Balabanovic and Y. Sholam. “Combining Content-Based and Collaborative Recommendation”. *Communications of the ACM*, 1997.
3. G. Karypis. “Evaluation of Item-Based Top-N Recommendation Algorithms”, 2000. In *Proceedings of CIKM*, 2001, pp. 247-254, 2001.
4. M. Keim Condliff, D. D. Lewis, D. Madigan, and C. Posse. “Baeysian Mixed-Effects Models for Recommendation Systems”. In *Proceedings of ACM SIGIR Workshop on Recommendation Systems*, 1999.
5. M. Newman, and D. Fisher. “A News Article Recommendation System for Informavores”, http://www.cs.berkeley.edu/~danyelf/inf/.
6. J. Konstant, B. Miller, D. Maltz, J. Herlocker, L. Gordon, and J. Riedl. “Grouplens: applying collaborative filtering to Usenet news”. *Communications of the ACM*, 40(3):77-87, 1997.
7. P. Resnick, N. Iacovou, M. Suchak, P. Bergston, and J. Riedl. “Grouplens: An open architecture for collaborative filtering of netnews”. In *Proceedings of CSCW*, 1994.
8. R. Baeza-Yates, B. Ribeiro-Neto. “*Modern Information Retrieval*”. Addison Wesley, 1999.
9. T. Mitchell “*Machine Learning*”. NY:McGraw-Hill, 1997.

10. G. Widmer, and M. Kubat “Learning in the Presence of Concept Drift and Hidden Contexts”. *Machine Learning Journal, 1*, 69-101, 1996.
11. M. Pazzani and D. Billsus. “Learning and revising user profiles: The identification of interesting web sites”. *Machine Learning*, 27:313-331, 1997.
12. D.H. Widyantoro “*Learning User Profile in Personalized News Agent*”. Master’s thesis, Department of Computer Science, Texas A&M University, 1999.
13. B. Mobasher, H. Dai, T. Luo, M. Nakagawa, and J. Witshire. “Discovery of aggregate usage profiles for web personalization”. In *Proceedings of the WebKDD Worksho*p, 2000.
14. P. Chan. “A non-invasive learning approach to building web user profiles”. In *Proceedings of ACM SIGKDD International Conferenc*e, 1999.
15. C. Buckley and G. Salton. “Optimization of Relevance Feedback Weights”. In *Proceedings of the 18th Annual Intl ACM SIGIR Conference, Seattle*, 1995.
16. M.Balabanovic. “An Adaptive Web Page Recommendation Service”. In *Proceedings of First International Conference on Autonomous Agents*, pp. 378-385, 1997
17. G. Salton, and M.J. McGill. “*Introduction to Modern Information Retrieval*”.McGraw-Hill, 1983.
18. J. Allan. “Incremental Relevance Feedback for Information Filtering”. In Proceedings of the Nineteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 270-278, ACM, 1996.
19. J. J. Rocchio, "Relevance feedback in information retrieval". In G. Salton, The SMART Retrieval System: Experiments in Automatic Document Processing, pp. 313-323, Englewood Cliffs, NJ, 1971. Prentice Hall, Inc.
20. M. Balabanovic. “*Learning to Surf: Multi-agent Systems for Adaptive Web Page Recommendation*”. PhD thesis, Department of Computer Science, Stanford University, 1998.
21. D. Billsus and M. Pazzani. “A personal news agent that talks, learns and explains”. In *Proceedings of the third International Conference on Autonomous Agents (Agents '99)*, pages 268--275, Seattle, WA, 1999
22. B.D. Sheth, “*A Learning Approach to Personalized Information Filtering*”. Master's Thesis, Department of Electrical Engineering and Computer Science, MIT, 1994.
23. D.H.Widyantoro, T.R. Ioerger, and J. Yen. “An Adaptive Algorithm for Learning Changes in User Interests”. In *Proceedings of the Eight International Conference on Information and Knowledge Management*, pp. 405-412, 1999.
24. C. Fox, "*Lexical Analysis and Stoplists*", in *Information Retrieval: Data structures and Algorithms*, edited by W. B. Frakes and R. Baeza-Yates, 1992, Prentice-Hall, pp. 102-130.
25. E. Riloff, “Little words can make a big difference for text classification”. In Proceedings, 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'95), pp. 130-136, Seattle, 1995.
26. WordNet, a lexical database for the English language, http://www.cogsci.princeton.edu/~wn/.