Recommendation Based on Co-clustring Algorithm, Co-dissimilarity and Spanning Tree

Ouafae Baida, Najma Hamzaoui, Abdelfettah Sedqui, Abdelouahid Lyhyaoui
AbdelmalekEssaâdi University
LTI Lab, ENSA of Tangier
BP: 1818, Tanger Principal, Tanger. Morocco
{wafaebaida, nejmahamzaoui, abdelfettah.sedqui, lyhyaoui}@gmail.com

ABSTRACT: *Recommender system is a system that helps users to find interesting items. Actually, collaborative filtering technology is one of the most successful techniques in recommender system. In this article we propose two new approaches based on the co-clustering and co-dissimilarity between users. In the literature, we find a lot of approaches able to recommend items to the user. Aiming to offer a list of interesting items, we use a hybrid approach of collaborative filtering that performs better than others. Our collaborative filtering approach is partitioned in two steps, the first based a bond energy algorithm (BEA), it's one of group technology algorithm, its objective is realized the co-clustering or simultaneous clustering. The second is recommendation based on the graph theory, when we propose the use of kruskal algorithm; this one gives us a spanning tree with minimum weight from a connected graph. We define a group of criteria that help to determine the best items to recommend without computing the rating prediction.*

Keywords: Collaborative Filtering, Hybrid-based Collaborative Filtering, Graph Theory, Bond Energy Algorithm

Received: 26 June 2012, Revised 28 July 2012, Accepted 1 August 2012

1. Introduction

With the enormous information that exists in the web [1,2,3], recommendation systems offers users relevant information that will satisfy them and answer their real needs. As Stated, the recommendation system is a system that collects, filters, and recommends information. Automated CF is a technique that reduces information overload. According to [4], (CF) algorithms can be grouped into memory-based and model-based approaches. The collaborative filtering algorithm is considered as one of the most successful and widely used technical recommendation in the e-commerce recommendation system. A technique that recommends an item to a user based on users' interests, while trying not to disturb him too much.

There are two types of collaborative filtering algorithms: user-based and item-based collaborative filtering [5]. Both of them use the rating which represents the evaluation of the user on an item, and they rarely introduce hybrid approach. During the two latest decades, there are many advances in recommender systems research. An extensive review of the different approaches used in recommender systems is presented in [5,6]. CF still requires improvement to make good recommendations, in order to solve its shortcoming, such as sparsity, scalability.

Aiming this, we attempt in this paper to present two new approaches. The first one concerns the use of '*Bond Energy Algorithm* (BEA)'. [McCormick et al. 1972], where one algorithm of group technology that performs the co-clustering, clustering of rows and also of columns, to find the communities, and the second one present our solution to perform the best recommendation with Kruskal algorithm.

We use graph theory to predict the user future preferences without the need of computing the rating prediction. First, we form two matrices of dissimilarity items/items and users/users. The proposed approach is based on a Kruskal algorithm and the graph theory. The Kruskal algorithm allows us to find a spanning tree with minimum weight through a connected weighted graph. The proposed CF algorithm provides a direct way to calculate the recommendation .This is due to the fact of taking into account the use of the graph theory and the co-similarity to recommend a list of the best items.

The rest of the paper is structured as follows: Section 2 presents related works; Section 3 presents the proposed approaches in detail. Section 4includes dataset description and metrics, and finally we conclude in section 5.

2. Related Work

CF attempts to predict the rating of an item for a particular user based on how other users have previously given a note on the same item and recommended items they liked. According to [7], The CF can be classified into approaches memory-based and model-based. On the one hand, memory based algorithms exercise recommendations based on the entire collection of items previously rated by the users based on the totality of the rating matrix [8]. As an explanation, they use some kind of measurement aggregation, taking into account the ratings of other users, as being similar to a given item. Many similarity measures and a variety of aggregation criteria can occur in different models. On the other hand, in the models based algorithms, predictions are made by constructing a model. Finally, this model is then used for the recommendation. In this case, predictions are not based on ad hoc heuristics, but rather on a model learned from the underlying data using statistical techniques and machine learning.

Collaborative filtering generates personalized recommendations by aggregating the experiences of similar users in the system. One key aspect of collaborative filtering is the identification of consumers or users similar to the one which needs a recommendation. Cluster models, Bayesian Network models, and specialized association-rule algorithms, among other techniques, have been used for this identification purpose [9]. Based on similar consumers or neighbors, methods such as the most frequent item approach [10] can then be used to generate recommendations.

During the two last decades, there have been many advances in recommender systems research. An extensive review of the different approaches used in recommender systems is presented in [11].CF still requires improvement to make good recommendations, in order to solve its shortcomings such as sparsity, scalability, and cold start problem.

There are many unsolved problems in CF, one of them, the cold start of a user , in which situation , there is no information about this take, and recommendation seems to be impossible. To solve this problem, many works presented like NMF (nonnegative matrix factorization) k-means or other clustering algorithm. These ones explain methods to perform clustering and as result giving the communities. In [15], we find several definitions giving and some of them based in graph theory.

The second problem concerns a new element that has not been previously noted, this one cannot be recommended by users. On the other hand, sparsity, where the number of ratings specified the limited number of recommendations for the user whose tastes are unusual compared to the rest of the population.

To overcome the weaknesses of memory-based techniques, a line of research has focused on model-based techniques with the aim of seeking fore accurate. Based on ratings, these techniques are applied on both users and items, giving a new way to identify the neighborhood, instead of the use of the entire database.

A typical CF algorithm proceeds in three phases: calculating the similarity sim(i,j) between active user/item i and user/item j, neighborhood formation by selection the k similar users/items. Finally, we generate the top N items by weighted average of all the ratings of users/items in the neighborhood.

After the similarity computation, CF algorithms have to select the most similar users for the active user. This is an important step since the recommendations are generated using the ratings of neighbors and therefore neighborhood has an impact on the recommendation quality. Among the strategies that are remarkable in the literature for the selection of neighbors. According to [12],there is baseline strategy, baseline strategy with overlap threshold, similarity strategy that select the top nearest-neighbors purely according to their similarities with the active user. The combination of these strategies is preferable, but it could decrease the performance of the algorithms.

An important recent trend in information processing and organization of data and modeling many types of relationships and the dynamic processes in social systems is the use of graphs for two reasons. First, a graph or network-based model is easy to interpret and provides a natural and general framework for many different types of applications including recommender systems. Second, a rich set of graph-based algorithms is readily applicable when the recommendation task is formulated as a graph-theoretic problem. Typically the SR which is represented as a bipartite graph contains two sets of vertices: sets of users and other resources.

Below we briefly survey three representative graph-based approaches that explore relationships to improve the recommendations. For example, there are researchers who considered the problem of predicting links as a problem of machine learning [13]. They showed that taking the nature of the bipartite graph can improve the performance of forecasting models; it is obtained by projecting the bipartite graph to a graph unimodal and introducing new variants of topological measures to measure probability of two nodes to be connected. There are others in [14] who proposed an approach for smoothing votes. This is an algorithm based on a graph of resources. While every vote given by a user with a set of resources, must be sufficiently less.

Therefore, a coefficient of Smoothness is calculated based on a graph of resources while respecting the intrinsic structure of resources. This method can explore the geometric information of a data element and use this information to produce better recommendations. In [15] another method that uses aggregation graphs of preference for the prediction collaborative votes is presented.. The principle of this approach is based on the idea of forming a graph of preference for a target user based on the values of votes given to a set of resources in order to construct a graph of preferences, from graphs preferences users while minimizing the number of back-edge in the graph's overall preferences.

3. The Proposed Approach

The size of networks requires the need to find methods can make them easy to manage. This requirement involves finding ways to structure it as groups with common characteristics.

Nowadays, the most popular data that need co-clustering comes from bioinformatics [13], text mining, industry and collaborative filtering.

For recommender systems, we propose in this paper a new approach based on a technology to structure all users in the form of communities. We talk about the group technology.

In industry, group technology concept is based on the identification and exploitation of similarities and the similarity between the products and processes of design and manufacturing to streamline production and reduce manufacturing costs. [14] This technology will be useful in the case of recommender system for obtaining groups of users with a certain resemblance or votes with data similar to resource groups (for example movie). We will use one algorithm of this technology: Bond Energy Algorithm (BEA).

The proposed approaches in this article are based on a Bond energy algorithm and Kruskal algorithm of the graph theory. With these two algorithms, we can find the communities, users key and also a set of best items to recommended. The set of items recommended are the result of searching the relevant items in a connected weighted graph. But first how we can represent our system of recommendation with the graph theory?

First, in a recommendation-system application, there are two classes of entities, which we shall refer to as users and items. Users have preferences for certain items, and these preferences must be teased out of the data. The data itself is represented as a utility matrix, giving for each user-item pair, a value that represents what is known about the degree of preference of that user for that item. Values come from an ordered set, e.g., integers 1–5 representing the number of stars that the user gave as a rating for that item.

As a result, the System of recommendation is represented as a bipartite graph [16], it contains two groups of nodes G (U, I, E) where:

U: set of users

I: set of items

E: set of edges e_{ij} having the value as weight, the rating giving by one user U_i to item I_j.

The presentation of this graph on machines will be in the form of a utility matrix of degree N*M.(figure 1)

Where

N: number of the users

M: number of the items.

	u_1	u_2	$\dots$	u_M
I_1	R_{11}	R_{12}	$\dots$	R_{1M}
I_2	R_{21}	R_{22}	$\dots$	R_{2M}
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
I_N	R_{N1}	R_{N2}	$\dots$	R_{NM}

Figure 1. Utility matrix

3.1 Detection communities

Using the utility matrix and BE we can find easily the communities and the users key. It is one of the first algorithms to tackle rearrangement clustering [17]. BEA uses the measure of effectiveness (ME) in which the similarity measure for two rows, *i* and *j*, is:

$$\text{sim (i, j)} = \sum_{k=1}^{m} a_{ik}\, a_{jk} \qquad (1)$$

Where m is the number of features

It has several useful properties. First, it groups attributes with larger values together, and the ones with smaller values together (i.e., during the permutation of columns and rows, it shuffles the attributes towards those with which they have higher value and away from those with which they have lower value). Second, the composition and order of the final groups are insensitive to the order in which items are presented to the algorithm. Finally, it seeks to uncover and display the association and interrelationships of the clustered groups with one another.

All this explain that, BEA is a greedy algorithm that can give us the best detection for a set of users having the same interesting. Because the group found having the same values (higher or lower), so the indexes of rows present the users having evaluated these items (indexes of columns) by similar values of rating. (figure 1). And when we find one user belongs to several communities, we can consider him like a user key and we can recommended him to a new user because he presents several communities, that means several interesting and tastes.

3.2 Recommendation

Now, using the graph for SR, we adapt an algorithm of graph theory with recommendation systems, to make a recommendation without calculating the predictions to realize it. We will go through three steps, the first is the passage from matrix of rating to two dissimilarity matrices 'user-user' and 'item-item', of these two matrices, second we obtain two spannig trees using an algorithm of graph theory, and the third is based on the of nearest neighbors. Then we propose a list of items to be recommended.

o **Step 1: construct tow matrices of dissimilarity**

The first question we must deal with is how to measure similarity of users or items from their rows or columns in the utility matrix to construct two matrices of dissimilarity between items and between users. For calculating the similarity, we can use the Pearson Correlation Similarity or the Cosine/Vector Simirality [17].

The similarity between two elements *d* and *q* can be calculated using the following expression:

0	0	0	0	0	0	0	0	0	4	4	4	0
0	0	0	0	0	0	0	0	4	3	3	3	0
0	0	4	4	4	4	4	0	3	3	3	3	0
0	0	5	5	5	5	5	0	0	5	5	5	0
0	0	5	5	5	5	5	0	0	5	5	5	0
5	3	5	5	5	5	5	2	3	4	4	4	0
4	0	0	0	0	0	0	0	0	3	3	3	0
0	0	0	0	0	0	0	0	0	3	3	3	0
0	5	0	0	0	0	0	0	0	4	4	4	0
5	0	0	0	0	0	0	0	0	4	4	4	0
0	0	0	0	0	0	0	0	0	3	3	3	0
0	5	0	0	0	0	0	0	0	4	4	4	0
4	0	0	0	0	0	0	0	0	3	3	3	0
0	0	0	0	0	0	0	0	0	4	4	4	0
0	3	0	0	0	0	0	5	4	4	4	4	0

Figure 1. BEA applied in database Movielens

$$\text{cosine}\,(d, q) = \frac{\sum_{i=1}^{n} d_i * q_i}{\sqrt{\sum_{i=1}^{n} d_i^2}\,\sqrt{\sum_{i=1}^{n} q_i^2}} \tag{2}$$

To have the dissimilarity between two elements, we have the folowing:

$$\text{Dis}(d, q) = 1\text{- Cosine}\,(d, q) \tag{3}$$

As result, we are going to have two square matrices.

o **Step 2: construct tow spanning tree of dissimilarity**

The goal of a recommender system is to generate meaningful recommendations to a collection of users for items or products that might interest them. However, the recommended items should have a minimum dissimilarity with the items that are already evaluated by the user, and having also the best ratings given by the similar users.

To eliminate the edges having the important value of dissimilarity, we will use kruskal algorithm.

```
Kruskal(E, T : Sequence of Edge, P : UnionFind)
sort E by increasing edge weight
foreach {u, v} “ E do
if u, v are in different components of P then
add edge {u, v} to T
join the partitions of u and v in P
```

Figure 2. KRUSKAL algorithm

Then we can give a graphic modeling for both matrices ‘items-items’ and ‘users-users’ by taking as weight of edges the values of coefficient of dissimilarities. From these graphs, we can construct two spanning trees with minimum weight by using kruskal algorithm [18]

Figure 2 shows the algorithmic step of Kruskal algorithm. As a result, the represented tree (figure 3) will respect a hierarchy of the users or items being based on values minimum of the coefficients of dissimilarity as well as elimination of edges having a big weight.

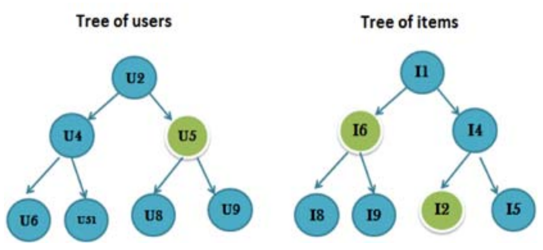

Figure 3. Spanning trees of users and items

a. User's tree:

The graph G (V, E) represents the matrix of dissimilarity users-users where:

V: the set of users.

E: the set of edges e_{ij} where the weights are the similarities values.

The spanning tree with minimum weight represents all the users classified according to dissimilarity minimum. (figure 2).

b. Item's tree:

By the same way, we are going to pass of a simple graph where nodes are the items and the edges represent the dissimilarities values towards a spanning tree with mimum weights (figure 2).

o **Step 3: Recommendation**

In this level, we provide details about the used terminology and recommendation process for the proposed CF approach. We summarize the symbols used for recommendation to the user.

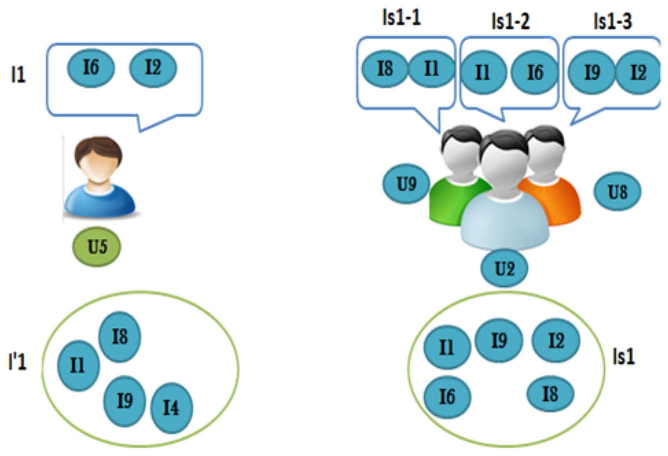

Figure 4. Set of items

For the active user U_5 we have (figure 3)

I_1: set of items already evaluated by U_5 and having the best ratings.

$I_1' \in$ I: Set of neighborhood and similar to I_1' given from tree of items.

I_{s1}: set of items for similar users having the best ratings.

To find this set, we first determine the best neighborhood of user active, and we searched all items having the best ratings Figure 4 illustrates the intersection of $I_1'$and I_{s1} which gives all the items which are similar to items of I_1', as well as them is already evaluated by the similar users closest of user U_1. Otherwise, we classified the union of I_1' and I_{s1} based in the order to have set of item having a maximum average rating.

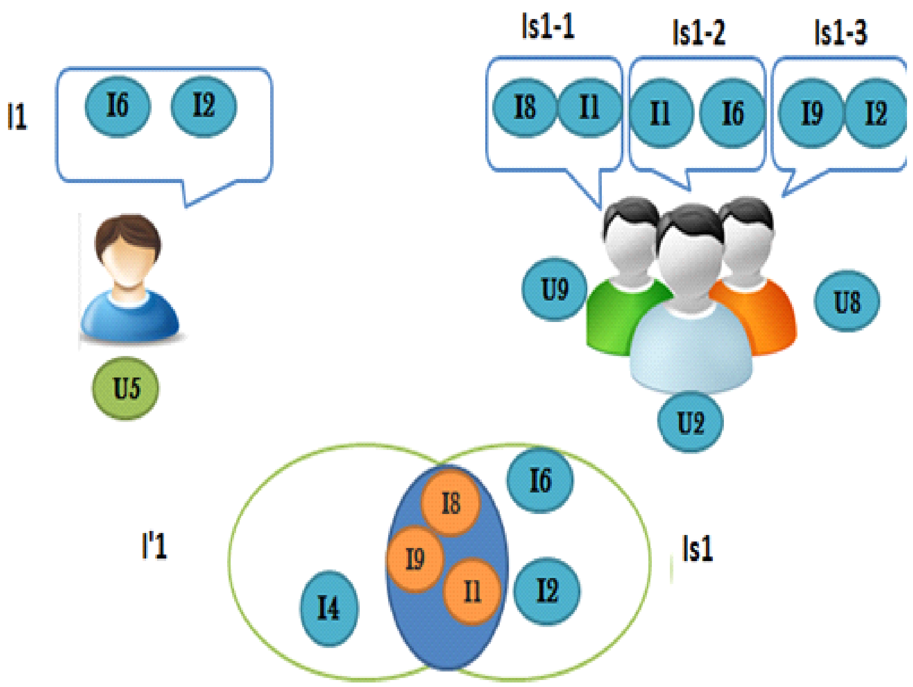

Figure 5. Set of items recommended

4. Data and Measurement

4.1 DataSet

In order to measure the prediction algorithm performance and to compare the results of different neighborhood based prediction algorithms, we use Dataset that comes from the Movielens dataset at Grouplens Research Project in Minnesota University. MovieLens is a movie recommending application users initially rate a subset of movies that they have already seen [19]. (http://MovieLens.umn.edu/) captures the user's rating for the movies and provides a list for movie recommendation. The historical dataset consists of 100,000 ratings from 943 users on 1682 movies with every user having at least 20 ratings and simple demographic information for the users is included. The site now has over 45000 users who have expressed opinions on 6600 different movies. The ratings are on a numeric five-point scale with 1 and 2 representing negative ratings, 4 and 5 representing positive ratings, and 3 indicating ambivalence. We need to extract the demographic information (gender of the movie for example) of the item existing in the movieLens data to construct demographic vectors of the 19 features for the item.

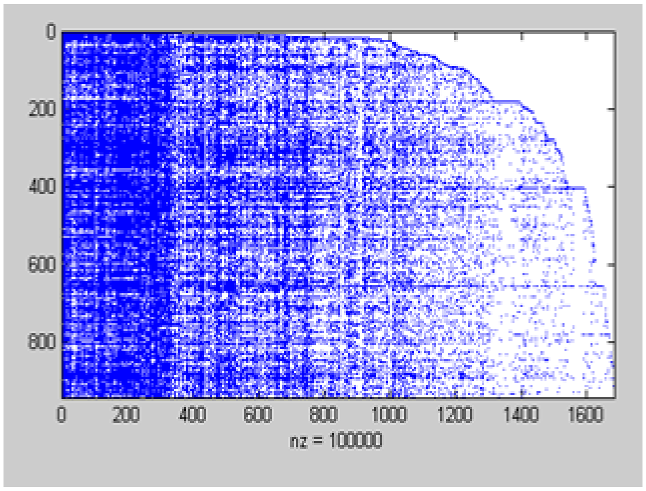

Figure 6. Data set Movielens

4.2 Evaluation Metric
In [20], evaluation measures for recommender systems are separated into three categories:

•Predictive Accuracy Measures. These measures evaluate how close the recommender system came to predicting actual rating/ utility values.

• Classification Accuracy Measures. These measures evaluate the frequency with which a recommender system makes correct/ incorrect decisions regarding items.

• Rank Accuracy Measures. These measures evaluate the correctness of the ordering of items performed by the recommendation system.

4.2.1 Precision and Recall
To use these metrics, recommender system must convert its ratings scale into a binary {Do not Recommend, Recommend} scale. Items for which the prediction is to recommend are shown to the user, other items are not shown. The transition mechanism is up to recommender systems.

$$\text{recall} = \frac{RR}{RR + RN}$$

$$\text{precision} = \frac{RR}{RR + RN}$$

Where:

RR: set of items relevant and recommend.

FP: set of items not relevant and recommend.

RN: set of items relevant and not recommend.

Our Approach

In our approach, we do not calculate values as predictions in conventional approaches, but based on graphical methods, we obtain a list of items to be recommended, which means we need to use already defined metrics to measure performance of our approach.

To compute them we must know which items are relevant and which ones are not relevant. We proposed to eliminate a set of items already evaluated by the target user (percentage of elimination), and we compare the list of items recommended by our approach to the list eliminated.

This proposal will also help us to easily determine a set of recommended data, not consumed by the user, which means the ability to calculate prediction and recall.

4.2.1.1 Percentage of elimination of data
By varying the value of percentage of eliminated data, we find results in the table 1. When we eliminate 25 % of data, we acquire results of good values for precision and recall, because the remaining 75 % of data containing several informations about the user, that helps to perform a good recommendation. Contrary, when we eliminate 75 % of information about active user, we do not acquire good results.

4.2.1.2 Neighborhood
Our approach is based on neighborhood (‘item / item’ and ‘user/user’) to perform the best recommendation. By varying this parameter we find that: when we exaggerate neighborhood we acquire the best results for recall.

4.2.1.3 Interpolated curve
We prefer calculating precision for predefined values of recall, from 0 % to 100 % by step of 10 %. In practice these values of recall can not be attained exactly: the values of precision must therefore be interpolated.

The rule of interpolation is as following: the value interpolated by precision for a level of recall *i* is the maximum precision acquired for an upper or equal recall in *i*.

Percentage of elimination	**Neighborhood level**	**Average of precision**	**Average of recall**
25%	1	0.9604	0.2501
	2	0.7610	0.4812
	3	0.5809	0.6324
50%	1	0.3416	0.1435
	2	0.2700	0.1863
	3	0.2031	0.2500
75%	1	0.1781	0.1166
	2	0.1592	0.0265
	3	0.1373	0.0085

Table 1. Precision/recall

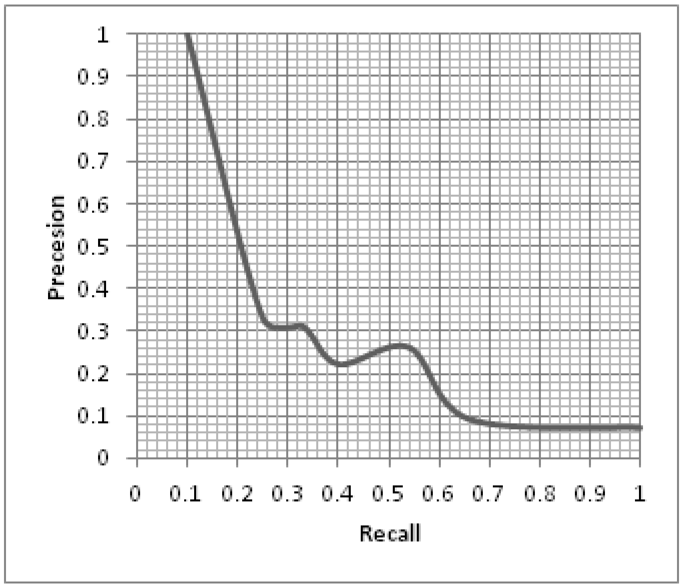

Figure 7. Precision and recall

5. Conclusion

In this article, a hybrid approach of collaborative filtering is presented. In comparison with other approaches mentioned in the section of related works, our approach gives a more accurate recommendation. Because we start first by using BEA to detect the communities and users key, then we try to apply our second approach at communities level to perform the recommendation, to have a spanning tree with minimum weight, we used the KRUSKAL algorithm which is considered one of the most competitive algorithms. This approach provides an interesting list of items by the fact of taking into account the use of the co-similarity matrices between items and users based on some criteria. This approach can be considered as the first work among our other ideas to realize and the future works will improve the recommender system application based on the graph theory.

References

[1] Janner, T., Schroth, C. (2007). Web 2.0 and SOA: converging concepts enabling the internet of services, *In*: IT Professional, p. 36–41.

[2] Knights, M. (2007). Web 2.0, *IET Communications Engineer*, p. 30–35.

[3] Lin, K. J. (2007). Building Web 2.0, *Computer,* p. 101–102.

[4] Breese, J. S., Heckerman, D., Kadie, C. (1998). EmpiricalAnalysis of Predictive Algorithms for Collaborative Flitering, *In*: Proc.14 th Conf. *Uncertainty in Artificial Intelligence*, July.

[5] Burke, R. (2002). Hybrid recommender systems: survey and experiments, User Model. *User-Adapted Interaction* 12 (4) 331–370.

[6] Adomavicius, G., 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* 17 (6) 734–749.

[7] Breese, J. S., Heckerman, D., Kadie, C. (1998). Empirical Analysis of Predictive Algorithms for Collaborative Flitering, *In*: Proc.14 th Conf. Uncertainty in Artificial Intelligence, July.

[8] Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., Riedl, J. (1994). 'Grouplens: an open architecture for collaborative of netnews _ltering, *In*: Proc. of CSCW '94, .

[9] Efficient adaptive-support association rule mining for recommender systems, *Data Mining and Knowledge Discovery*, 6 (1) 83-105

[10] Sarwar, B., Karypis, G., Konstan, J., Reidl, J. (2000a). Analysis of recommendation algorithms for e-commerce. *In*: Proceedings of the ACM Conference on Electronic Commerce, 158-167.

[11] Burke, R. (2002). Hybrid recommender systems: survey and experiments, User Model. User-Adapted Interaction 12 (4) 331–370., Adomavicius, G., Tuzhilin, A. (2005). Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions, *IEEE Trans Knowledge Data Eng.* 17 (6) 734–749.

[12] Zhang, J., Pu, P. (2007). A recursive prediction algorithm for collaborative filtering recommender systems. *In*: RecSys '07: Proceedings of the 2007 ACM conference on Recommender systems, p. 57–64, New York, NY, USA.ACM.

[13] Text Mining Biomedical Literature for Discovering Gene-to-Gene Relationships: A Comparative Study of Algorithms' Ying Liu, Shamkant B. Navathe, Civera, J., Dasigi, V., Ram, A., Ciliax, B. J., Dingledine, R. (2005). *IEEE/ACM Transactons on Computational Biology and Bioinformatics*, 2 (1) January-March.

[14] Walter Jean-Luc; Mutel Bernard (Directeur de thèse); Université de Mulhouse, Mulhouse, FRANCE (Université de soutenance) » Study of group technology implementation. Application in the Holeg international firm.

[15] Benchettara, N., Kanawati, R., Rouveirol, C. Supervised Machine Learning Applied to Link Prediction in Bipartite Social Networks.

[16] Fei Wang, Sheng Ma, Liuzhong Yang, Tao Li. Recommendation on Item Graphs.

[17] Ehsan KAZEMI LCA4, I&C, EPFL Community Structures in Recommender Systems.

[18] Maunendra, S. Sudeshna Sarkar Pabitra Mitra Desarkar, Aggregating Preference Graphs for Collaborative Rating Prediction.

[19] Climer, Sh., Zhang, W. (2006). Rearrangement Clustering: Pitfalls, Remedies, and Applications, *Journal of Machine Learning Research*, 7, 919–943

[20] Adomavicius, G., Tuzhilin, A. (2005). Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions, *IEEE Trans Knowledge Data Eng*. 17 (6) 734–749.

[21] Benchettara, N., Kanawati, R., Rouveirol, C. Supervised Machine Learning applied to Link Prediction in Bipartite Social Networks p. 4 .

[22] Vozalis, E., Konstantinos, Margaritis, G. Analysis of Recommender Systems' Algorithms.

[23] Osipov, V., Sanders, P., Singler, J. The Filter-Kruskal Minimum Spanning Tree Algorithm.

[24] Zheng Wen, (2008). Recommendation System Based on Collaborative Filtering December 12.