Matching Users with Groups in Social Networks

Domenico Rosaci and Giuseppe M.L. Sarné

Abstract Understanding structures and dynamics of social groups is a crucial issue for Social Network analysis. In the past, several studies about the relationships existing between users and groups in On-line Social Networks have been proposed. However, if the literature well covers the issue of computing individual recommendations, at the best of our knowledge any approach has been proposed that considers the evolution of on-line groups as a problem of matching the individual users' profiles with the profiles of the groups. In this paper we propose an algorithm that addresses this issue, exploiting a multi-agent system to suitably distribute the computation on all the user devices. Some preliminary results obtained on simulated On-line Social Networks data show both a good effectiveness and a promising efficiency of the approach.

1 Introduction

Nowadays, the Internet scenario has seen a significant growth in scale and richness of On-line Social Networks (OSNs), that are becoming very complex and internally structured realities, particularly in the largest communities as Facebook, Flickr, MySpace, Google+ and Twitter. In these networks we observe an increasing diffusion of social *groups*, that are sub-networks of users having similar interests [3, 7] and sharing opinions and media contents. In the past, some studies about the relationships existing between users and groups in OSNs have been proposed. For instance, in [8], a quantitative study is presented about the influence of neighbours on the probability of a particular node to join with a group, on four popular OSNs. Moreover, the proposal presented in [1] deals with the problem of the overwhelming number of groups, that causes difficulties for users to select a right group to join.

Domenico Rosaci, Giuseppe M.L. Sarné
University Mediterranea of Reggio Calabria, 89123 Reggio Calabria (Italy) e-mail: {domenico.rosaci, sarne}@unirc.it

To solve this problem, the authors introduce, in the context of Facebook, the Group Recommendation System using combination of hierarchical clustering technique and decision tree. Also the studies presented in [2, 10, 11] deal with the problem of giving recommendations to a group of users, instead of a single user. This is a key problem from the viewpoint of creating OSN groups that provide their users with a sufficient satisfaction. The problem is not simply to suggest to a user the best groups to join with, but also to suggest to a group the best candidates to be accepted as new members. If the existing research in OSN well covers the issue of computing individual recommendations, and the aforementioned issue begins to give attention to the issue of computing group satisfaction, however at the best of our knowledge any study has been proposed to consider the issue of managing the evolution of a OSN group as a problem of matching the individual users' profiles with the profiles of the groups. The notion itself of *group profile* is already unusual in OSN analysis, although the concept of *social profile* is not new in the research area of virtual communities. For instance, in [9], following some theories originated in sociological research on communities, a model of a virtual community is presented, defined as a set of characteristics of the community.

In this paper, we provide the following contribution:

- we introduce the notion of *group profile* in the context of OSNs, giving to this notion a particular meaning coherent with the concept of OSN group. In our perspective, an OSN group is not simply a set of categories of interests, but also a set of common rules to respect, a preferred behaviour of its members, a communication style and a set of facilities for sharing media contents. Our definition of group profile is coherent with the definition of a *user profile*, that contains information comparable with those of a group profile.
- We exploit the above notion of group profile to provide each group of an OSN with a *group agent*, capable of creating, managing and continuously updating the group profile. Similarly, we associate a *user agent* with each user of the OSN.
- Likewise to other approaches we proposed in the past to build recommender systems for virtual communities [12, 14, 15, 16], introducing efficiency via the use of a distribute agent system, here we propose to exploit the agents above to automatically and dynamically computing a matching between user profiles and group profiles in a distributed fashion. We note that the idea of associating an agent as a representative of a group is not completely new in a social network scenario, having been introduced also in [4, 5, 6]. However, in these past works this idea has been exploited to allow interoperability between different groups, without facing the problem of matching users and groups. We propose to provide the user agent with a matching algorithm able to determine the group profiles that best match with the user profile. This matching algorithm, named User-to-Groups (U2G) is based on the computation of a dissimilarity measure between user and group profiles. As a result of the U2G computation, the user agent will submit on behalf of its user some requests for joining with the best suitable groups. On the other hand, the agent of each group will execute the U2G algorithm to accept, among the users that requested to join with the group, only those users having profiles that sufficiently match with the group profile. This way, the dynamic

evolution of the groups should reasonably lead to a more homogeneous intra-group cohesion.
- Some experiments we have performed on a set of simulated users and groups confirms this intuition, and also show promising efficiency a scalability of the proposed algorithm.

The remaining of the paper is organized as follows. In Section 2 we introduce our reference scenario. Section 3 present the proposed U2G matching algorithm, while Section 4 describes the experiments we have performed to evaluate our approach. Finally, in Section 5 we draw our conclusions.

2 The Social Network Scenario

In our scenario, we deal with a Social Network S, represented by a pair $\mathcal{S} = \langle U, G \rangle$, where U is a set of *users*, G is a set of *groups* of users and each group of users $g \in G$ is a subset of U (i.e, $g \subseteq U \; \forall g \in G$).

We assume that a single user u (resp., a single group g), of $\mathcal{S}$ is characterized by the following properties:

- He/she (resp., it) deals with some categories of interest in the social network (e.g. *music*, *sport*, *movie*, etc.). We denote as C the set of all the possible categories of interests, where each element $c \in C$ is a string representing a given category. We denote as $INTERESTS_u$ (resp., $INTERESTS_g$) a mapping that, for each category $c \in C$, returns a real value $INTERESTS_u(c)$ (resp., $INTERESTS_g(c)$), ranging in [0..1], representing the level of interest of the user u (resp., of the users of the group g) with respect to discussions and multimedia content dealing with c. The values of this mapping are computed on the basis of the actual behavior of u (resp., of the users of g).
- He/she has a preference with respect the access mode of the groups (resp., it has adopted a particular access mode). The access mode is the policy regulating the access to a group (e.g., *open*, *closed*, *secret*, etc.). We denote as $ACCESS_u$ (resp., $ACCESS_g$) the access mode (represented by a string) associated with u (resp., g).
- He/she adopts or does not adopt (resp., it tolerates or does not tolerate) some possible *behaviour* available in the social network. A behaviour is a type of action that a user could perform, e.g. "publishing more than 2 posts per hour", or "publishing posts longer than 500 characters". We suppose each possible behaviour is represented by a boolean variable, that is equal to *true* if this behaviour is adopted, $false$ otherwise. We denote as $BEHAVIOURS_u$ (resp., $BEHAVIOURS_g$) a set of behaviours, representing how u (resp., g) behaves with respect to all the possible behaviours. For instance, if $BEHAVIOURS = \{b_1, b_2\}$, where b_1 represents the behaviour of publishing more than 2 posts per hour, and $b2$ represents the behaviour of publishing in most of the cases posts longer than 500 characters, then a property $BEHAVIOURS_u = \{true, false\}$ characterizes a

user u that publishes more than 2 posts per hour and that, in the most of cases, does not publish posts longer than 500 characters.
- He/she has (resp., its users have) a set of *friends*, that are users of the social network. We denote as $FRIENDS_u$ (resp., $FRIENDS_g$) the set of all the users that are friends of u (resp., the set of all the users that are friends of at least a member belonging to the group g).

Then, we define the profile p_u (resp., p_g) of a user u (resp., a group g) as a tuple $\langle INTERESTS_u, ACCESS_u, BEHAVIOURS_u, FRIENDS_u \rangle$ (resp., $\langle INTERESTS_g, ACCESS_g, BEHAVIOURS_g, FRIENDS_g \rangle$). Moreover, we assume that a software agent a_u (resp., a_g) is associated with each user u (resp., group g). The agent a_u (resp., a_g) automatically performs the following tasks:

- it updates the profile p_u (resp., p_g) of its user (resp., group) each way the user u (resp., a user of the group g) performs an action involving some information represented in p_u (resp., p_g). In particular, each time the user u publishes a post, or comments an already published post, dealing with a category c, the new value $INTERESTS_u(c)$ is updated as follows:

$$INTERESTS_u(c) = \alpha \cdot INTERESTS_u(c) + (1 - \alpha) \cdot \delta$$

that is a weighted mean between the previous interest value and the new value, where α and δ are real values (ranging in [0..1]). More in detail, δ represents the increment we want to give to the u's interest in c consequently of the u's action, while α is the importance we want to assign to the past values of the interest value with respect to the new contribution. The values α and δ are arbitrarily assigned by the user itself. Similarly, the $INTERESTS_g(c)$ value of a group g is updated by the agent a_g each time the $INTERESTS_u(c)$ value of any user $u \in g$ changes. The new value of $INTERESTS_g(c)$ is computed as the mean of all the $INTERESTS_u(c)$ values $\forall c \in g$. Moreover, each way the user u performs an action in the social network (e.g. publishing a post, a comment, etc.) its agent a_u analyses the action and consequently sets the appropriate boolean values for all the variables contained in $BEHAVIOURS_u$ (e.g., if $BEHAVIOURS_u$ contains a variable representing the fact of publishing more than 2 posts per hour, then a_u checks if the action currently performed by u makes true or false this fact and, consequently, sets the variable).
Analogously, the agent a_g, associated with a group g, updates the variables contained in $BEHAVIOURS_g$ each time the administrator of g decides to change the correspondent rules (e.g., e.g. if $BEHAVIOURS_g$ contains a variable representing the fact of publishing more than 2 posts per hour, and the administrator of g decides that this fact is not tolerated in the group, then a_g sets the correspondent variable to the value $false$).
Furthermore, if the user u (resp., the administrator of the group g) decides to change his/her preference with respect to the access mode, the agent a_u (resp., a_g) consequently updates $ACCESS_u$ (resp., $ACCESS_g$). Finally, if the user u (resp., a user of the group g) adds a new friend in his/her friends list, or deletes

an assisting friend from his/her friends list, the agent a_u (resp., a_g) consequently updates $FRIENDS_u$ (resp., $FRIENDS_g$). Note that the agent a_g computes the property $FRIENDS_g$ as the union of the sets $FRIENDS_u$ of all the users $u \in g$.
- Periodically, the agent a_u (resp., a_g) executes the user agent task (resp., group agent task) described in Section 3, in order to contribute to the *User-to-Group* (*U2G*) *Matching* global activity of the social network.

In order to perform the above tasks, each agent can interact with each other, sending and receiving messages. This possibility is assured by the presence of a *Directory Facilitator* agent (DF), associated with the whole social network, that provides a service of Yellow Pages. More in particular, the names of all the users and groups of the social network are listed in an internal repository of the DF, associating with each user and group the corresponding agent name. A *Communication Layer* allows an agent x to send a message to another agent y simply by using the name of y in the *receiver* field of the message. Note that maintaining the DF naming repository is the only centralized activity in our social network scenario, while the algorithm computing the U2G matching is completely distributed on the whole agent network.

3 The U2G Matching Algorithm

In our scenario, the U2G matching is a global activity distributed on the user and group agents belonging to the agent network. More in particular, each user agent a_u (resp. group agent a_g) periodically executes the following *user agent task* (resp. *group agent task*), where we call *epoch* each time the task is executed, and we denote as T the (constant) period between two consecutive epochs.

3.1 The user agent task

Let X be the set of the n groups the user u is joined with, where $n \leq n_{NMAX}$, being $NMAX$ the maximum number of groups which a user can join with. We suppose that a_u: (*i*) records into an internal cache the profiles of the groups $g \in X$ obtained in the past by the associated group agents; (*ii*) associates with each profile p_g the date of acquisition, denoted as $date_g$. Let also m be the number of the group agents that at each epoch must be contacted by a_u. In such a situation, a_u behaves as follows:

- In the DF repository, it randomly selects m groups that are not present in the set X. Let Y be the set of these selected groups, and let $Z = X \bigcup Y$ a set containing all the groups present in X or in Y.
- For each group $g \in Y$, and for each group $g \in X$, such that the date of acquisition $date_g$ is higher than a fixed threshold ψ, it sends a message to the group agent a_g, whose name has obtained by the DF, requesting it the profile p_g associated with the group.

- For each received p_g, it computes a *dissimilarity measure* between the profile of the user u and the profile of the group g, defined as a weighted mean of four contributions c_I, c_A, c_B and c_F, associated with the properties $INTERESTS$, $ACCESS$, $BEHAVIOURS$ and $FRIENDS$, respectively. More in particular, each of this contribution measures how much are different the values of the corresponding property in p_u and in p_g. To this purpose:
  - c_I is computed as the average of the differences (in the absolute value) of the interests values of u and g for all the categories present in the social network, that is:
$$c_I = \frac{\sum_{c \in C} |INTERESTS_u(c) - INTERESTS_g(c)|}{|C|}$$
  - c_A is set equal to 0 (resp.,1) if $ACCESS_u$ is equal (resp., not equal) to $ACCESS_g$.
  - c_B is computed as the average of all the differences between the boolean variables contained in $BEHAVIOURS_u$ and $BEHAVIOURS_g$, respectively, where this difference is equal to 0 (resp., 1) if the two corresponding variables are equal (resp., different).
  - c_F is computed as the percentage of common friends of u and g, with respect to the total numer of friends of u or g. That is:
$$c_F = \frac{FRIENDS_u \bigcap FRIENDS_g}{|FRIENDS_u \bigcup FRIENDS_g|}$$

  Note that each contribution has been normalized in the interval [0..1], for making comparable all the contributions. The dissimilarity d_{ug} of a group g with respect to the user u is then computed as:
$$d_{ug} = \frac{w_I \cdot c_I + w_A \cdot c_A + w_B \cdot c_B + w_F \cdot c_F}{w_I + w_A + w_B + w_F}$$
- Now, let τ a real value, ranging in [0..1], representing a threshold for the dissimilarity, such that each group $g \in Z$ is considered as a good candidate to join if (*ii*) $d_{ug} > \tau$ and (*ii*) it is inserted by a_u in the set $GOOD$. Note that if there exist more than $NMAX$ groups satisfying this condition, the $NMAX$ groups having the highest values of global difference are selected. For each selected group $g \in GOOD$, when $g \notin X$, the agent a_u sends a join request to the agent a_g, that also contains the profile p_u associated with u. Otherwise, for each group $g \in X$, when $g \notin GOOD$, the agent a_u leaves the group g.

3.2 The group agent task

Let K be the set of the k users joined with the group g, where $k \leq n_{KMAX}$, being $KMAX$ the maximum number of members allowed by the group administrator. We suppose that a_g stores into an internal cache the profiles of the users $u \in K$ obtained

in the past by the associated user agents, and also associates with each profile p_u the date of acquisition, denoted as $date_u$. Each time a_g receives a join request by a user agent r, that also contains the profile p_r associated with r, it behaves as follows:

- For each user $u \in K$ such that the date of acquisition $date_u$ is higher than a fixed threshold η, it sends a message to the user agent a_u, whose name has obtained by the DF, requesting it the profile p_u associated with the user.
- After the reception of the responses from the contacted user agents, it computes the *dissimilarity measure* d_{ug} between the profile of each user $u \in K \bigcup \{r\}$ and the profile of the group g, following the definitions given in Section 3.1.
- Now, let π a real value, ranging in [0..1], representing a threshold for the dissimilarity, such that a user u is considered as acceptable to join if $d_{ug} > \pi$. Then, the agent a_g stores in a set $GOOD$ those users $u \in K \bigcup \{r\}$ such that $d_{ug} > \tau$ (if there exist more than $KMAX$ users satisfying this condition, the $KMAX$ users having the highest values of global difference are selected). If $r \in GOOD$, a_g accepts its request to join with the group. Moreover, for any user $u \in K$, with $u \notin GOOD$, a_g deletes u from the group.

4 Experiments

In this section, we describe some preliminary experiments we performed to evaluate the effectiveness of the U2G matching activity in making more homogeneous the groups of an OSN. To this purpose, we have built an OSN simulator, written in JAVA and based on the JADE platform, capable of simulating the users' behaviours and the activities of their associated agents, as well as the group administrators' behaviour and the activities of the group agents, in a social network. In our simulation, we considered an OSN with 150.000 users and 1200 groups. The simulator provided each user and each group with a user profile, having the structure described in Section 2. More in detail, the profile p_u of a user u is generated as follows:

- each values $INTERESTS_u(c)$ is a random value from a uniform distribution of values in [0..1];
- $ACCESS_u$ is assigned from three possible values, namely $OPEN$, $CLOSED$ and $SECRET$, such that the probability of assigning the value $OPEN$ (resp., $CLOSED$, $SECRET$) is set to 0.7 (resp., 0.2, 0.1). This distribution values appear reasonable in a realistic OSN scenario;
- $BEHAVIOURS_u$ contains six boolean variables, representing the user's attitude to: (*i*) publish more than 1 post per day; (*ii*) publish posts longer than 200 characters in most of the cases; (*iii*) publish at least two comments per day to posts of other users; (*iv*) respond to comments associated with its posts in most the cases; (*v*) leave at least 2 rates "I like it "per day; (*vi*) respond to a message of another user in most the cases. The values of these variables are randomly generated from a uniform distribution on the possible value-set {*true*, *false*};

Table 1 Values used for the U2G parameters.

	$alpha$	δ	ψ (days)	τ	η (days)	π
Value	0.5	0.1	10	0.7	10	0.7

- in $FRIENDS_u$, the simulator inserts a set of other users, randomly choosing one of the following distributions: (*i*) users that have a dissimilarity with u smaller than 0.5 (the dissimilarity is computed in the same way of d_{ug} in Section 2); (*ii*) users randomly chosen from the set of the OSN users; (*iii*) 50 percent of the users generated as in (*i*) and the remaining 50 percent generated as in (*ii*). These distributions represent three realistic types of users, namely those that select their friends based on similar preferences and behaviour, those that randomly accept any friendship and those that behaves in an intermediate fashion with respect to the first two attitudes.

Users are then randomly assigned to the available groups, in such a way that a user is joined at least with 2 groups and at most with 10 groups. The profile p_g of each group g is assigned generating completely random values for the properties $ACCESS_g$ and $BEHAVIOURS_g$, while the properties $INTERESTS_g$ and $FRIENDS_g$ are computed based on the corresponding members' values, following the formulas described in Section 2. The values of the parameters introduced in Section 3 are shown in Table 1.

As a measure of the internal *cohesion* of a group, we use the concept of *average dissimilarity*, commonly exploited in Clustering Analysis [13], defined as the average of the dissimilarities between each pair of objects in a cluster. In our scenario, a group g is the equivalent of a cluster of users, and the average dissimilarity of g, denoted as AD_g is computed as $\frac{\sum_{x,y \in g, x \neq y} d_{xy}}{|g|}$.

In order to measure the global cohesion of the groups of the social network, we compute the mean MAD and standard deviation DAD of all the AD_g, by the formulas: $MAD = \frac{\sum_{g \in G} AD_g}{|G|}$; $\quad DAD = \sqrt{\frac{(AD_g - MAD)^2}{|G|}}$

In our simulation, after the random generation of the profiles of users and groups, the initial values for the above measures were $MAD = 0.512$ and $DAD = 0.43$, indicating a population with a very low homogeneity, due to the completely random generation of the groups. Starting from this initial configuration, we have applied the U2G algorithm described in Section 3, simulating a number of 150 epochs of execution per each user, where each epoch simulated a time period of a day. The results of the simulation, in terms of MAD and DAD with respect to the epochs, are shown in Figure 1-A and 1-B. The results clearly show that the U2G algorithm introduces a significant increment of the cohesion in social network groups, that after a period of about 110 epochs achieves a stable configuration represented by $MAD = 0.163$ and $DAD = 0.095$. Moreover, we repeated the experiments above, changing the number of simulated users and groups. In particular, in Figure 1-(C) we have plotted the stable MAD/DAD for different values of the users' number, having fixed to 1200 the number of the groups, while in Figure 1-(D) the stable

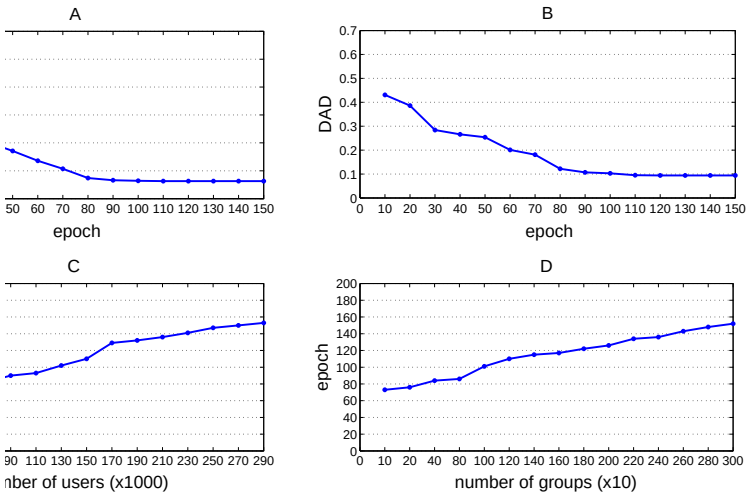

Fig. 1 The variation of (A) MAD and (B) DAD vs epochs and the variation of the epoch corresponding to stable MAD/DAD for (C) different number of users and 1200 groups and (D) different number of groups and 150.000 users.

MAD/DAD values are reported for different values of the groups' number, having fixed to 150.000 the number of the users. The results show that the number of the necessary epochs for achieving a stable configuration increases almost linearly with respect to the number of the groups, confirming a good scalability of the approach.

5 Conclusion

The problem of making possible a suitable evolution in OSN groups, dynamically increasing the intra-group cohesion, is emerging as a key issue in the OSN research field. If the notion of homogeneity is becoming already more complex with the introduction of high-structured user profiles, leading to design sophisticated approaches for computing similarity measures, on the other hand the large dimensions of current OSNs, as well as the continuously increasing number of groups, introduce the necessity of facing efficiency and scalability problems. In this paper, we present a User-to-Group matching algorithm, that allows a set of software agents associated with the OSN users to dynamically and autonomously manage the evolution of the groups, detecting for each user the most suitable groups to join with based on a dissimilarity measure. Moreover, the agents operate on behalf of the group administrators, such that a group agent accepts only those join requests that come from users profile compatible with the profile of the group. Some preliminary experiments clearly show that the execution of the matching algorithm increases in time the internal cohesion of the groups composing the social network.

Acknowledge
This work has been partially supported by the TENACE PRIN Project (n. 20103 P34XC) funded by the Italian Ministry of Education, University and Research.

References

1. E.A. Baatarjav, S. Phithakkitnukoon, and R. Dantu. Group Recommendation System for Facebook. In *On the Move to Meaningful Internet Systems: OTM 2008 Work.*, pages 211–219. Springer, 2008.
2. S. Basu Roy, S. Amer-Yahia, A. Chawla, G. Das, and C. Yu. Space Efficiency in Group Recommendation. *The VLDB Journal*, 19(6):877–900, 2010.
3. F. Buccafurri, D. Rosaci, G.M.L. Sarné, and L. Palopoli. Modeling Cooperation in Multi-Agent Communities. *Cognitive Systems Research*, 5(3):171–190, 2004.
4. P. De Meo, A. Nocera, G. Quattrone, D. Rosaci, and D. Ursino. Finding Reliable Users and Social Networks in a Social Internetworking System. In *Proc. of the 2009 Int. Database Engineering & Applications Symp.*, pages 173–181. ACM, 2009.
5. P. De Meo, A. Nocera, D. Rosaci, and D. Ursino. Recommendation of Reliable Users, Social Networks and High-Quality Resources in a Social Internetworking System. *AI Communications*, 24(1):31–50, 2011.
6. P. De Meo, G. Quattrone, D. Rosaci, and D. Ursino. Dependable Recommendations in Social Internetworking. In *Web Intelligence and Intelligent Agent Technologies, 2009*, pages 519–522. IET, 2009.
7. S. Gauch, M. Speretta, A. Chandramouli, and A. Micarelli. User Profiles for Personalized Information Access. In *The Adaptive Web*, volume 4321 of *LNCS*, pages 54–89, 2007.
8. P. Hui and S. Buchegger. Groupthink and Peer Pressure: Social Influence in Online Social Network Groups. In *Social Network Analysis and Mining, 2009. ASONAM'09. Int. Conf. on Advances in*, pages 53–59. IEEE, 2009.
9. J. Hummel and U. Lechner. Social Profiles of Virtual Communities. In *System Sciences, 2002. HICSS. Proc. of the 35th Annual Hawaii Int. Conf. on*, pages 2245–2254. IEEE, 2002.
10. M.L. Kasavana, K. Nusair, and K. Teodosic. Online Social Networking: Redefining the Human Web. *Journal of Hospitality and Tourism Technology*, 1(1):68–82, 2010.
11. J.K. Kim, H.K. Kim, H.Y. Oh, and Y.U. Ryu. A Group Recommendation System for Online Communities. *Int. Journal of Inf. Management*, 30(3):212–219, 2010.
12. L. Palopoli, D. Rosaci, and G. Sarné. A Multi-tiered Recommender System Architecture for Supporting E-Commerce. *Studies in Computational Intelligence 446, Intelligent Distributed Computing VI*, pages 71–81, 2013.
13. Ronald K Pearson, Tom Zylkin, James S Schwaber, and Gregory E Gonye. Quantitative Evaluation of Clustering Results Using Computational Negative Controls. In *Proc. 2004 SIAM Int. Conf. on Data Mining*, pages 188–199, 2004.
14. D. Rosaci and GML Sarné. Recommending Multimedia Web Services in a Multi-device Environment. *Information Systems*, 38(2):198–212.
15. D. Rosaci and G.M.L. Sarné. Efficient Personalization of e-Learning Activities Using a Multi-Device Decentralized Recommender System. *Computational Intelligence*, 26(2):121–141, 2010.
16. D. Rosaci and G.M.L. Sarné. A multi-agent recommender system for supporting device adaptivity in e-Commerce. *Journal of Intelligent Information Systems*, 38(2):393–418, 2012.