Trust and Compactness in Social Network Groups

Pasquale De Meo , Emilio Ferrara , Domenico Rosaci and Giuseppe M.L. Sarné

***Abstract*—Understanding the dynamics behind group formation and evolution in social networks is considered an instrumental milestone to better describe how individuals gather and form communities, how they enjoy and share the platform contents, how they are driven by their preferences/tastes and how their behaviors are influenced by peers. In this context, particularly relevant is the notion of *compactness* of a social group. While the literature usually refers to compactness as a measure to merely determine how much members of a group are similar among each other, we argue that the mutual trustworthiness between the members should be considered as an important factor in defining such a term. Trust has in fact profound effects on the dynamics of group formation and their evolution: individuals are more likely to join with and stay in a group if they can trust other group members. In this paper, we propose a quantitative measure of group compactness that takes into account both the similarity and the trustworthiness among users, and we present an algorithm to optimize such a measure. We provide empirical results, obtained on the real social networks EPINIONS and CIAO, that compare our notion of compactness versus the traditional notion of user similarity, clearly proving the advantages of our approach.**

***Index Terms*—Decision Support Systems, Machine Learning, Multi-agent systems, Social network services, Social Trust.**

I. Introduction

ONLINE Social Networks (OSNs) allow people to easily connect with each other as well as to share, discuss and comment opinions and multimedia content.

In such a context, a relevant role is played by social *groups*, that are sub-networks of users [1], [2]. Some recent studies investigated on existing relationships between users and groups in OSNs [3]–[5]. One of the problems recently put into evidence in this context is the overwhelming number of groups in real-world platforms. This causes difficulties for users to select the right group(s) to join with and often lowers their degree of satisfaction [6]–[8].

The existing research in OSNs covers the issue of computing individual recommendations, and the aforementioned examples give attention to the issue of computing group satisfaction. However, to the best of our knowledge, no study has been proposed to consider the issue of managing the evolution of an OSN group as a problem of improving the *compactness* of the group, i.e. the stability in time of the group configuration.

But why do we need to define the group compactness and how it could be used? The users of an OSN would desire to be included in groups in which they can perform satisfactory activities, such as commenting posts and sharing media with the other members. The satisfaction derives from the existence of affinities among the group members, that lead the members themselves to remain into the group and to not leave it, thus making the group *compact*. Interestingly, the notion of *compactness* in this context of OSN groups has been differently addressed in recent literature [9]–[11]. The most common approach is to consider group compactness coinciding with similarity among its users. Unfortunately, the concept of *similarity* itself is quite subjective and different notions have been used in different works, resulting in a lack of consensus in the research community [12]–[14].

Another common viewpoint is to consider communities as groups of users more densely connected among each other than with the rest of the network: this leads to conceive the group formation as a network clustering problem [15], [16]. A third recent approach aims at merging the two previous strategies, by suggesting that the formation of a group should be based on some group compactness model that considers both structural and semantic similarities (representing commonalities of relations, interests and preferences) [17], [18].

However, we argue that one important aspect has been overlooked so far: the mutual *trustworthiness* among the group members. The main reason underlying our proposal is the evidence that individuals are motivated to stay in groups with other members whom they trust. This evidence has been highlighted by several studies on real OSNs [19], [20]. Moreover, the need to integrate a measure of trust with a measure of similarity between group members has already been widely discussed in the research community of multi-agent systems, pointing out that the introduction of trust measures leads to significantly improve the effectiveness of the agents [21]–[23].

Our definition of compactness takes into account such a need. We introduce a measure that integrates both similarity and trust, by using a weight coefficient that each user can arbitrarily set. Based on this definition, we propose an algorithm for automatically improving the compactness of the groups in an OSN scenario. Our algorithm, called *U2G* (users-to-groups), solves the problem of optimal matching between the individual users' profiles and the profiles of the groups. Our algorithm can be used by a recommender agent, that will act as a counsellor for an associated human user, helping him to join with the most suitable groups. In words, the recommender agent can use our algorithm to make a prediction of the satisfaction the user will obtain by joining with a given group, based on the computation of the new compactness that the group would assume if the user will join with it. On the other hand, the algorithm can be implemented also by a recommender agent associated with a given group, which will act as a counsellor for the group administrator. In this case, the

P. De Meo is with the Department of Ancient and Modern Civilizations, University of Messina, 98166 Messina, Italy, e-mail: pdemeo@unime.it

E. Ferrara is with the Center for Complex Networks and Systems Research, School of Informatics and Computing, Indiana University, Bloomington (IN), USA, e-mail: ferrarae@indiana.edu

D. Rosaci is with the Department DIIES, University of Reggio Calabria, Loc. Feo di Vito, 89122 Reggio Cal., Italy, e-mail: domenico.rosaci@unirc.it

G.M.L. Sarné is with the Department DICEAM, University of Reggio Calabria, Loc. Feo di Vito, 89122 Reggio Cal., Italy, e-mail: sarne@unirc.it

Manuscript received September 30th, 2013, revised February 3rd,2014.

recommender agent is able to suggest to the administrator if the acceptance of a given requester user would be profitable or not. Therefore, the set of recommender agents can be viewed as a typical multi-agent recommender system [24]–[26]. We highlight that our definition of compactness simply merges the two most widely recognized factors influencing the cohesion of the groups, i.e. similarity and trust, by leaving to each user (and to each group administrator) the possibility to personally weight the importance of the similarity with respect to the trust. Our algorithm is the way by which we demonstrate the possibility to approximately optimize such a compactness measure by using a distributed (and thus practicable) approach. We also observe that our experiments with the version of the algorithm that uses only the similarity are useful to quantify the approximation introduced by this simplification.

The remainder of the paper is as follows: In Section II we discuss related literature and the novelties provided by this work; Section III introduces our reference scenario and Section IV presents the proposed U2G matching algorithm. Section V and VI describes the experiments we performed to evaluate our method and the advantages and limitations introduced by this approach. Finally, in Section VI we draw our conclusions.

II. Related Work

The research covered in this paper lies at the intersections of many fields like *group modelling*, *matching of user and group profiles* and *trust* in OSNs. In this section we describe some recent research results achieved in each of these fields and illustrate the main novelties brought in by our approach.

An increasing number of authors focused on the problem of suggesting items to the member of a group (*group recommendation*). Some of such approaches follow a *score aggregation* strategy [27], [28]. More formally, let $\mathcal{U} = \{u_1, \ldots, u_n\}$ be the user population, $\mathcal{I} = \{i_1, \ldots, i_m\}$ be a collection of items and suppose that a *rating function* $r : \mathcal{U} \times \mathcal{I} \rightarrow R$ is available, with R (*rating space*) a discrete set. Typically R ranges in $\{0, \ldots 5\} \cup \{\perp\}$, where $\perp$ specifies a not yet rated item. The function r takes a user $u_i \in \mathcal{U}$ and an item $i_k \in \mathcal{I}$ as input and generates an element $r_{ik} \in R$ as output. Let $G \subseteq \mathcal{U}$ be a group of users. The task of building a profile of G is equivalent to compute a function $f_G : \mathcal{I} \rightarrow R$ receives an item i_k as input to return how much i_k satisfies the members of G. In this context, perhaps, the two most popular strategies to compute $f_G(\ )$ are the *Average* [27] and the *Least Misery* [28].

A second category of approaches follow the *preference aggregation* paradigm [29], [30]. In [29] a technique based on stereotypes is described, where each stereotype considers some agents features (extracted by agents' profile and/or their observed behaviors) and an expected transaction outcome. In such a way, agents and strangers can be aggregated in groups by matching their profiles with the defined stereotypes also to derive their associated trust. By using strategies based on social theory, the authors of [30] suggest to build ontology-based user profiles to merge into a group profile; thanks to group profile, content recommendations can be generated for the group members. Our framework allows to model the policies followed to access groups and previous user behaviours like the posts she/he generated or if she/he liked/disliked an item. Moreover, the management of both group and user profiles is carried out by means of a multi-agent architecture and agents are in charge of updating user and group profiles as well as of finding groups a user could join. This features are only partially considered by some of the cited papers.

In the literature there are few papers dealing with the task of matching user and group profiles and, often, they only suggest groups a user can join. Such a problem is also called *affiliation recommendation* [31]. This differs from the group recommendation problem, where the objects to recommend are items like books or movies whereas in the affiliation recommendation problem they are groups. An early contribution is due to Spertus *et al.* [32] and describes an empirical comparison on six measures to compute the user-community similarity degree to recommend communities (that are equivalent to our groups). Most of these measures give relevance to the users' community memberships as implicit indicators of the users' behaviors; by contrast, our approach explicitly consider both individual and group interests, behavioral attitudes and their reciprocal perceived trust to recommend groups.

The task of recommending communities is also considered in [33], describing the CCF (Combinational Collaborative Filtering) algorithm to suggest new friendship relationships to users as well as the communities they could join. With a probabilistic point of view, CCF considers a community from two perspectives: a *bag of users* (formed by its members) and a *bag of words* describing community interests. By fusing these information sources it is possible to alleviate the data sparsity arising when only information about users (resp. words) is used.

As a further example of different strategies, Vasuki *et al.* [31] show that the co-evolution of the user's friendship relationships joint with the knowledge of group affiliations form a good predictor of the next groups she/he will join with.

Our algorithm differs from all the cited studies for many significant aspects. In particular, as first difference, interests and habits of users and groups are more accurately described in our agent knowledge representation (see Section III-A) than in other proposals. It allows the matching between users and groups profiles to be fruitfully performed by our algorithm without the need of probabilistic or machine learning techniques, as in [34]. Finally, our approach is different because we provide an algorithm to match users and groups which is distributed among group and user agents. It relies on a greedy heuristic because, at each stage, it computes how good a group is for a given user and selects, uniformly at random, $NMAX$ of these groups, being $NMAX$ a suitable threshold. In this way, the algorithm can efficiently manage large networks having a large number of groups. The U2G algorithm is also flexible because it can be driven by similarity or by trust or a their combination as the cost function.

Approaches to computing trust in OSNs model a user community as a graph G whose vertexes represent users and an edge joining two vertexes specifies that the user associated with the vertex v trusts that associated with the vertex u. Since users typically provide few trust values, the graph G is usually sparse and suitable techniques are required to infer new trust

and reputation values starting from available ones.

In [35] a maximum network flow algorithm on G infers trust and in [36], a modified version of the Breadth First Search algorithm on G infers multiple values of reputation for each user that aggregated by applying a voting algorithm give a unique user's reputation value. The approach of [37] considers paths up to a fixed length k in G and propagates the explicit trust values on them to obtain the implicit ones. Our approach significantly differ from the approaches cited above. First of all, we model trust as a linear combination of two factors related to the *reliability* and to the *reputation*. As a consequence we do not assume that a user provides trust declaration about other users. Reliability, in fact, can be inferred by a wide range of signals regarding user behaviors.

III. The Reference Scenario

Our scenario deals with an OSN $\mathcal{S}$ described as a tuple $\mathcal{S} = \langle \mathcal{U}, \mathcal{G} \rangle$, where $\mathcal{U}$ is a set of *users* and $\mathcal{G}$ is a set of *groups* of users, such that each group $g \in \mathcal{G}$ is a subset of $\mathcal{U}$.

In such a context, we associate a multi-agent system with $\mathcal{S}$, containing a software agent for each user $u \in \mathcal{U}$ and for each group $g \in \mathcal{G}$. Indeed, in our perspective, each user u is assisted by her/his personal agent, denoted by a_u, during the activities involving the participation to groups, and each group g is assisted by an administrator agent a_g managing the join requests. In Section III-A we describe the knowledge representation associated with the agents above, while Section III-B deals with the agents' tasks. Sections III-C and III-D discuss, respectively, our definitions of trust and compactness.

A. The agents' knowledge representation

In order to characterize the interests and the preferences of each user u (resp. group g), we associate u (resp. g) with a *profile* p_u (resp. p_g). Such a profile contains five *properties* to incorporate in our model. The first four properties are called *interests*, *access preferences*, *behaviors* and *friends* and they are called *preference properties*; they represent the preferences expressed by u (resp. the users of g) with respect to (*i*) topics of interest, (*ii*) mode to accessing groups, (*iii*) ways of performing activities and (*iv*) friends. A fifth property, called *trust*, is a trustworthiness property, and represents how much u (resp. g) trusts the users of the social network. We have chosen to adopt a uniform representation for both users and groups, and we have defined the profile of u (resp. g) as a 5-tuple $\langle I_u, A_u, B_u, F_u, T_u \rangle$ (resp. $\langle I_g, A_g, B_g, F_g, T_g \rangle$), where I_u, A_u, B_u, F_u, T_u (resp. I_g, A_g, B_g, F_g, T_g) are the properties of u (resp. g).

The property I_u (resp. I_g) represents the interests of u (resp., g) for the different categories of interest available in the OSN. We denote as C the set of all the possible categories, where each element $c \in C$ is an identifier of a given category and as I_u (resp. I_g) a mapping that, for each category $c \in C$, returns a real value $I_u(c)$ (resp. $I_g(c)$), ranging in $[0..1]$ and representing the level of interest of u (resp. the users of g) with respect to discussions and multimedia content dealing with c.

The property A_u represents the preference of u with respect to the mode of accessing to groups (e.g., *open*, *closed*, *secret*, etc.). We denote as A_u (resp. A_g) the access mode preferred by u (resp. set by the administrator of g). The property B_u represents the types of behavior adopted by u in her/his social activities. A behavior is a type of action that a user could perform. We suppose that a set of possible behaviors, denoted as $\mathcal{B}$, is associated with the OSN and represents the attitude of the user u with respect to a given behavior b belonging to $\mathcal{B}$ by a boolean variable, that is set to $true$ if b is adopted, $false$ otherwise. Therefore, B_u is a mapping that, for each behavior $b \in \mathcal{B}$, returns a boolean value $B_u(b)$. The property F_u (resp. F_g) represents 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 of g); it is included to reflect the users social structure, i.e. the social ties existing among users of the platform.

The property T_u represents the trust perceived by u with respect to each other user of the social network. In detail, T_u is a mapping that, for each user $v \in \mathcal{U}$, returns a real value $T_u(v)$, ranging in $[0..1]$; it represents the trust perceived by u with respect to v (see Section III-C). Analogously, the property T_g represents the trust that the group g, considered as a whole, perceives about each user of the OSN. Similarly to above, T_g is a mapping that, for each user $u \in \mathcal{U}$, returns a real value $T_g(u)$, ranging in $[0..1]$, representing the trust perceived by g with respect to u. More in particular, $T_g(u)$ is computed as the mean of all the individual trust measures $T_v(u)$ for each $v \in g$. It is important to highlight that our definition of trust for a group is a statistical simplification, reasonable only if the standard deviation from T_g is sufficiently small. This definition can have a little meaning in the cases that assumption is not valid. However, these cases correspond to groups having a small compactness; because of our algorithm has the goal of increasing the compactness, this yields a decrease in the standard deviation of T_g, thus making our definition reasonable.

We remark that both categories, access mode and behaviours are considered as common ontology elements for all the users of the network (similarly to the situation happening in Facebook, where the possible choices for categories of interest and access modes are pre-defined), thus we suppose a homogeneous semantic scenario. We also suppose that the computation of the interest values can be automatically performed by software agents that operate over the shoulders of the user, analysing the categories associated with the contents posted by the user himself.

B. The agents' tasks

According to the profiles and properties defined above, the agent a_u (resp. a_g) automatically performs the following tasks: *Task 1*. It updates the profile p_u (resp. p_g) of u (resp. g) every time u (resp. a user of g) performs an action involving any information represented in p_u (resp. p_g). Every time u publishes a post, or comments an already published post, dealing with a category c, the value $I_u(c)$ is updated as follows:

$$I_u(c) = \theta \cdot I_u(c) + (1 - \theta) \cdot \delta$$

that is a weighted mean between the previous interest value and the new value. Here θ 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 θ is the relevance we want to assign to the past values of $I_u(c)$ with respect to the new contribution. The values θ and δ can be either arbitrarily assigned by the user itself, or optimized via a learning algorithm. This formula has been largely used in many trust-based approaches for multi-agent system, showing good results in practical situations in term of effectiveness when the weights θ and δ are correctly set [38]–[42]. This way, it is possible for an OSN analyst to experimentally determine how much the interest increases when the user select the associated category, and how much important is the influence of the past choices in the updating. In our experiments, we refer to values of θ and δ commonly used in some multi-agent systems, as in [43].

Similarly, the $I_g(c)$ value of a group g is updated by the agent a_g every time the $I_u(c)$ value of any user $u \in g$ changes. The new value of $I_g(c)$ is computed as the mean of all the $I_u(c)$ values $\forall c \in g$. Moreover, every time u performs an action in the OSN (e.g. publishing a post, writing a comment, etc.) its agent a_u analyses the action and appropriately sets the boolean values for all the variables contained in B_u. Analogously, the agent a_g, associated with a group g, updates the variables contained in B_g every time the administrator of g decides to change the correspondent rules. Furthermore, if u (resp. the administrator of g) decides to change her/his preference with respect to the access mode, the agent a_u (resp. a_g) appropriately updates A_u (resp. A_g). Moreover, if u (resp. a user of g) adds or deletes a person in her/his friend list, the agent a_u (resp. a_g) consequently updates F_u (resp. F_g). Note that a_g computes the property F_g as the union of the sets F_u of all the users $u \in g$. Finally, if u expresses an evaluation of a post authored by another user v, the agent a_u re-computes the trust measure $t_{u,v}$, as described in Section III-C.

Task 2. Periodically, the agent a_u (resp. a_g) executes the user agent task (resp. group agent task) described in Section IV, in order to contribute to the *User-to-Group* (*U2G*) *Matching* global activity of the social network. To perform Tasks 1 and 2, agents 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 an indexing service. 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. The whole architecture described above is synthetically represented in Figure 1.

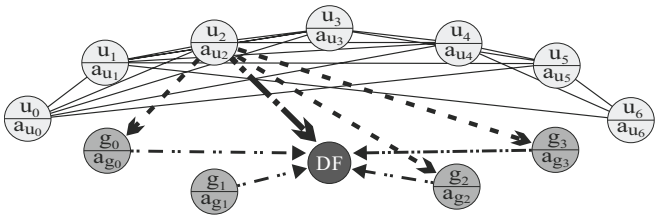

Fig. 1. The U2G multi-agent architecture.

C. Trust

The users of a social network mutually interact during their social activities, and each user u can express her/his level of satisfaction about the interactions with another user v.

In *Trust theory*, this satisfaction level is generally called *reliability* of v as perceived by u. Generally, in OSNs, u expresses her/his satisfaction about v by simply clicking on buttons such as "I Like It" (for example, on Facebook) or $+1$ (on Google+, YouTube, etc.), or vice-versa on the button "I Do Not Like It" or -1 (on YouTube, Yiid, etc.), without possibility of better refining her/his evaluation. However in our approach, for being as more general as possible, we represent the reliability of v as perceived by u with a real value denoted as $rel_{u\to v}$, ranging in the interval $[0..1]$, where 0 (resp. 1) is the least (resp. largest) value of reliability. Note that the reliability is an asymmetric measure: this implies that $rel_{u\to v}$ is not necessarily equal to $rel_{v\to u}$, and for such a reason in our notation we introduce the symbol $\to$ to clarify the direction of the trust relation.

The computation of $rel_{u\to v}$ strictly depends on the particular system of evaluation adopted in the given OSN. For instance, if the evaluation of posted contents can be performed via the buttons "I Like It" and "I Do Not Like It", $rel_{u\to v}$ could be computed as the ratio of the positive evaluations to the total evaluations expressed by u about the contents posted by v [44], [45]. Besides this measure of direct trust, we also define a global measure of the trust that the whole OSN perceives about each user u. We call this measure *reputation* of u, denoting it as rep_u, and we compute it by averaging all the reliability values $rel_{u\to v}$, for each $v \in U$.

Each user u, based on the aforementioned measures, can derive a synthetic measure of the trust about another user v, by integrating both reliability $rel_{u\to v}$ and reputation rep_v from her/his personal viewpoint, depending on the importance she/he gives to the reliability with respect to the reputation. We define the *trust* of u about v, denoted by $t_{u\to v}$, the following weighted mean:

$$t_{u\to v} = \alpha_u \cdot rel_{u\to v} + (1 - \alpha_u) \cdot rep_v$$

where α_u is a real coefficient, ranging in $[0..1]$, representing how much the reliability is considered important with respect to the reputation by the user u: if α_u is set to 0 by u, this means that u does not assign any relevance to the reliability in determine the v's trustworthiness, while $\alpha_u = 1$ means that u considers only the reliability for evaluating the trustworthiness of v. It's noteworthy that, following this intuition, trust is an asymmetric measure since in its formulation it accounts for reliability. The measure $rel_{u\to v}$ is updated by the agent a_u each time the user u expresses an evaluation of a post authored by u. Moreover, the agent a_u contacts the DF agent each time a reliability value is updated, sending the new value to it.

The agent a_u obtains the measure rep_u from the DF, when necessary to compute the trust. We highlight that the reliability $rel_{u\to v}$ is used when available, i.e. when a direct interaction between u and v happens. Otherwise, only the reputation value associated with v is used, derived from the whole community. We also specify that some cold start values are used for the reliability, that in our experiments are all set to 0, meaning no trust for a unknown user. Besides, we make the assumption that in the social network there is the possibility to express trust in a given user; for example, as in the case of the EPINIONS and CIAO OSNs we have used in our experiment, where a user directly expresses his trust for another user. Obviously, in those OSNs that do not make available direct mechanisms to express trust, our approach presents some limitations, and can be applied only by introducing some indirect trust measures, as considering the *I like it* mechanism, that could only partially capture a real trust evidence.

Finally, we define a measure $t_{u\to g}$ to determine the trustworthiness of a group g as perceived by u. This measure is determined by averaging all the values $t_{u\to v}$ for all the users v belonging to g. We also define the measure $t_{g\to u}$, representing a synthetic evaluation of the trust that the whole group g perceives versus the user u. It is computed by averaging all the trust values $t_{v\to u}$, where v is a member of g. Formally, $t_{g\to u} = \sum_{v\in g} t_{v\to u}/|g|$, $\forall v \in g$ and $|g|$ is the size of g.

D. Similarity and Compactness

The compactness of a pair of users (resp. a user and a group) depends on the degree of similarity of the two users and on the trust associated with the two users (resp, the user and the group). As for trust, we shall use the computational framework described in Section III-C. The similarity $\sigma_{u,v}$ between the profile of users u and v is defined as a weighted mean of four contributions c_I, c_A, c_B and c_F, associated with the properties I, A, B and F, respectively. Each contribution measures how much the values of the corresponding property in p_u and in p_v are similar. To this purpose:

- c_I is computed as the complement (with respect to 1) of the average difference between the interests values of u and v for all the categories present in the social network. Formally: $c_I = 1 - \frac{\sum_{c\in C} |I_u(c) - I_v(c)|}{|C|}$
- c_A is set equal to 1 (resp. 0) if A_u is equal (resp. not equal) to A_v.
- c_B is computed as the complement (with respect to 1) of the average difference between the boolean variables contained in B_u and B_v, 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 Jaccard similarity between the set of friends of u and v, i.e. $c_F = 1 - \frac{|F_u \bigcap F_v|}{|F_u \bigcup F_v|}$

Note that each contribution has been normalized in the interval $[0..1]$, to make comparable all the contributions. The similarity $\sigma_{u,v}$ is then computed as

$$\sigma_{u,v} = \frac{w_I \cdot c_I + w_A \cdot c_A + w_B \cdot c_B + w_F \cdot c_F}{w_I + w_A + w_B + w_F}$$

The similarity $\sigma_{u,g}$ between a user u and a group g is computed in the same manner described above, simply substituting the user v with the group g.

In order to compute the compactness between a user u and a user v, denoted by $\gamma_{u\to v}$, it is necessary to consider both the similarity $\sigma_{u,v}$ and the trust $t_{u\to v}$. Therefore, the compactness $\gamma_{u\to v}$ is usually an asymmetric measure, i.e. $\gamma_{u\to v}$ is generally different from $\gamma_{v\to u}$, because, in general $t_{u\to v} \neq t_{v\to u}$. Moreover, the computation of the compactness $\gamma_{u\to v}$ depends on how much importance the user u gives to the similarity with v with respect to the trust he has in v. We model the level of importance given to the similarity by a real coefficient WS_u, ranging in $[0..1]$. Consequently, we define the compactness $\gamma_{u\to v}$ as the following weighted mean:

$$\gamma_{u\to v} = WS_u \cdot \sigma_{u,v} + (1 - WS_u) \cdot t_{u\to v}$$

and the compactness $\gamma_{u\to g}$ between the user u and the group g as the analogous weighted mean:

$$\gamma_{u\to g} = WS_u \cdot \sigma_{u,g} + (1 - WS_u) \cdot t_{u\to g}$$

The asymmetric nature of this measure leads us to define also the compactness $\gamma_{g\to u}$ that a group g, considered as a whole, perceives versus a user u. This measure is defined as:

$$\gamma_{g\to u} = WS_g \cdot \sigma_{g,u} + (1 - WS_g) \cdot t_{g\to u}$$

where $\sigma_{g,u} = \sigma_{u,g}$ (because similarity is symmetric), while $t_{g\to u}$ is computed as described in Section III-C. The coefficient WS_g is associated with the given group g.

IV. The U2G Matching Algorithm

In this section we describe the *U2G* matching algorithm which has been designed to match users with groups with the goal of optimizing compactness. The U2G matching is a global activity distributed on the user and group agents belonging to the agent network. 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) time period elapsing between two consecutive epochs.

In the following, we first provide a theoretical justification of the U2G matching algorithm (Section IV-A) and, after that, we provide a complete description of the user agent task (see Section IV-B) and of the group agent task (Section IV-C).

A. Theoretical Justification

This section targets at illustrating the theoretical pillars on which the U2G algorithm is based on. The U2G algorithm is completely distributed and can be executed by user and group agents; due to the similarities between the algorithm run by user agents and that run by group agents we will focus only on the task of matching users with groups. The extension to the stage performed by group agents is straightforward and omitted for the sake of briefness.

The task of selecting the best suitable group(s) to recommend to each user can be conveniently formulated as an *optimization problem*. Let assume that each user u is willing to join with the group(s) g from which she/he will get highest

payoff: in our scenario, the payoff is given by the compactness function $\gamma_{u\to g}$. In our formulation it is convenient to add two further constraints. The former requires that a user joins with no more than $NMAX$ groups. Although we do not pose any limitation on the value of $NMAX$, it has been observed that $NMAX$ in real online platforms is much less than the overall number of existing groups. This results in that each user agent a_u is in charge of selecting no more than $NMAX$ groups a user can join with. The latter constraint is to fix a certain threshold $\tau > 0$ so that $\gamma_{u\to g} \geq \tau$. This constraints encodes the intuition that a user joins with a group only if this action yields an actual benefit (that, in our case, consists of an increase of compactness). Otherwise stated, we determine a priori that a user does not join with a given group if the expected payoff she/he can get back is less than a fixed threshold τ. If we define the variables $\delta_{u\to g} = 1$ if u joined with g and 0 otherwise, we obtain the following optimization problem:

$$\begin{aligned} &\max \sum_{u\in\mathcal{U}, g\in\mathcal{G}} \gamma_{u\to g}\delta_{u\to g} \\ &\text{s.t.} \quad \gamma_{u\to g} \geq \tau \qquad \forall u \in \mathcal{U}, \forall g \in \mathcal{G} \\ &\quad \sum_{u\in\mathcal{U}, g\in\mathcal{G}} \gamma_{u\to g}\delta_{u\to g} \leq NMAX \end{aligned} \tag{1}$$

Since we can assume that two arbitrary users u_1 and u_2 may join with the same group $\overline{g}$, we can suppose that for each group $g \in \mathcal{G}$ there are as many copies of g as the number of users in $\mathcal{U}$. This observation leads us to define $|\mathcal{U}|$ *optimization problems*, each of them as specified by Equation 1, and solve each of these problems independently one another. A further observation derives from the second constraint: since we can suppose that all the coefficients $\gamma_{u\to g}$ are known in advance, we can filter out all the groups $g^* \in \mathcal{G}$ such that $\gamma_{u\to g^*} < \tau$ and this yields some computational advantages. Therefore, if we define $\mathcal{G}_u^* = \{g \in \mathcal{G} : \gamma_{u\to g} \geq \tau\}$, the previous problem can be rewritten as

$$\begin{aligned} &\max \sum_{u\in\mathcal{U}, g\in\mathcal{G}_u^*} \gamma_{u\to g}\delta_{u\to g} \\ &\text{s.t.} \quad \sum_{u\in\mathcal{U}, g\in\mathcal{G}_u^*} \gamma_{u\to g}\delta_{u\to g} \leq NMAX \end{aligned} \tag{2}$$

In the following we shall denote as ϕ the objective function to optimize, i.e.:

$$\phi = \sum_{g\in\mathcal{G}} \gamma_{u\to g}\delta_{u\to g} \tag{3}$$

Under this formulation it is easy to recognize that the optimization problem in Equation 2 is a variant of the well-known Knapsack problem in which the weight of the items to manage are all equal to 1 and the profits are equal to $\gamma_{u\to g}$. In such a case, there is an optimal solution that can be generated by taking the best $NMAX$ groups, i.e. the $NMAX$ groups showing the highest values of compactness. In the following we shall focus on two cases, namely:

1) *Complete Knowledge*. We suppose that each user agent is aware of the compactness of all of the existing groups. In this case, the U2G algorithm generates the exact solution by means of a polynomial-time algorithm.
2) *Incomplete Knowledge*. In this case we suppose that each user agent knows just a subset of the available groups and, for each of them, also the compactness value is known.

The Incomplete Knowledge hypothesis is more realistic than the Complete Knowledge one because, in large OSNs, we expect a huge number of groups, which quickly grows over time. Therefore, the assumption that each agent can manage the list of all available groups and it can run on them some computation does not appear feasible. To model such a scenario we assume that each user agent can query the DF directory and it can extract a list $\mathcal{L}$ of groups and, based on the received results, it can decide which group(s) is relevant to a user. From an algorithmic standpoint, the formulation in case of incomplete knowledge is more challenging than that of complete knowledge: each agent knows just a fraction of the available groups and, in principle, we would not expect that it is able to generate the optimal solution. However, in the following we will show that under the reasonable hypothesis that groups in $\mathcal{L}$ are sampled uniformly at random from $\mathcal{G}$, then the solution produced by the U2G algorithm differs from the optimal one only by a constant factor. This desirable property of our algorithms makes it potentially very appealing for real applications.

B. The user agent task

In this section we describe the part of U2G algorithm about the matching of users and groups. In detail, we provide two versions of this algorithm: the former is called U2G-C and it handles the Complete Knowledge configuration; the latter, called U2G-I is designed to handle the Incomplete Knowledge configuration. We start discussing the U2G-C algorithm and subsequently we present the U2G-I algorithm.

In detail, let X be the set of the n groups the user u is affiliated to, where $n \leq NMAX$. As observed in Section IV-A, the coefficient $NMAX$ specifies the maximum number of groups which an arbitrary user can affiliate to.

We suppose that the user agent 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 each profile p_g with 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. The user agent a_u behaves as follows (see Figure 2): *Step 1*. In the DF repository, it randomly selects m groups that are not present in X. Let Y be the set of these selected groups, and let $Z = X \bigcup Y$ be a set containing all the groups present in X or in Y. *Step 2*. 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 (action 1 of Figure 2), whose name has obtained by the DF, requesting it the profile p_g associated with the group. *Step 3*. For each received p_g (action 2 of Figure 2), it computes the compactness $\gamma_{u\to g}$ between the profile of u and the profile of g (action 3 of Figure 2). *Step 4*. Now, let τ be a real value, ranging in $[0..1]$, representing

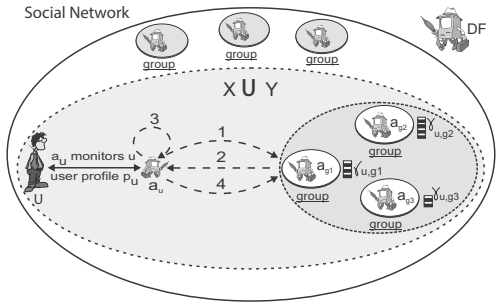

Fig. 2. User agent task schema

a threshold for the compactness, such that each group $g \in Z$ is considered as a good candidate to join with if $\gamma_{u \to g} > \tau$. All the good candidates are inserted by a_u in the set $GOOD$. Note that if more than $NMAX$ groups satisfy the condition for being inserted in $GOOD$, than the $NMAX$ groups having the highest values of compactness 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 (action 4 of Figure 2). Otherwise, for each group $g \in X$, when $g \notin GOOD$, the agent a_u deletes u from g. Observe that the usage of a date of acquisition has not been considered in Section IV-A; however this is a minor modification which has not any practical impact on the algorithm and on its performance. The U2G-C algorithm works in an iterative fashion and it is convergent, as proven in the following Theorem:

Theorem 4.1: Let $\phi^{(k)}$ be the value of the optimization function associated by the U2G-C algorithm at the k-th iteration and let $\hat{\phi}$ be the optimal solution associated with the optimization problem reported in Equation 2. Then the sequence $\{\phi^{(k)}\}$ is convergent to $\hat{\phi}$.

Proof To prove our result let us define $\delta_{u \to g}^{(k)}$ the value of the variable $\delta_{u \to g}$ generated by the U2G-C algorithm at the k-th iteration; let $\phi^{(k)}$ be the corresponding value of the objective function, i.e., $\phi^{(k)} = \sum_{g \in \mathcal{G}} \gamma_{u \to g} \delta_{u \to g}^{(k)}$. Let $\hat{\phi}$ be the optimal solution. We first observe that the sequence $\phi^{(k)}$ is *bounded*. In fact, by contradiction, suppose that $\phi^{(k)}$ is not bounded, i.e., for any arbitrary $L > 0$, there exists an index $\overline{k}$ such that $\phi^{(\overline{k})} > L$. Since L is arbitrary, we can fix $L > \hat{\phi}$, and, therefore, $\phi^{(\overline{k})} > \hat{\phi}$; this is a contradiction because, by definition, $\hat{\phi}$ is the maximum of ϕ, and, then, for any assignment of the variables $\delta_{u \to g}$, the corresponding value of ϕ must be less than $\hat{\phi}$. Secondly, observe that the sequence $\phi^{(\overline{k})}$ is monotonically non-decreasing because, at each iteration the U2G-C algorithm adds an element to $GOOD$ if and only if $\gamma_{u \to g} > \tau$ and this yields an increase in the value of ϕ. Since the sequence $\phi^{(\overline{k})}$ is bounded and monotonically non-decreasing it is also convergent and it converges to the actual optimal solution $\hat{\phi}$. □

Since the U2G-C algorithm is iterative we could stop computation prior to generate the optimal solution. In Section V-A we briefly discuss how quickly the U2G algorithm converges.

A slight change is required to implement the U2G-I algorithm, i.e. to handle the *Incomplete Knowledge* configuration. In this case, in fact, we assume that the algorithm knows only the largest number $NMAX$ of groups a user is allowed to join with. The user agent queries the DF and it receives as answer a code identifying a group g and the compactness $\gamma_{u \to g}$ of u with respect to g. On the basis of this information, the U2G-I algorithm has to decide if g has to be suggested to u or not. The U2G-I algorithm is similar to the U2G-C algorithm but it uses a different policy to build the $GOOD$ set. Prior to providing a technical description of the U2G-I algorithm we aim at illustrating the main intuitions behind it.

The U2G-I algorithm strongly resembles the well-known *secretary problem* [46]: in it we suppose that some persons apply for a position (e.g., secretary) and our goal is to select the best candidate. Candidates are interviewed in a random order and the hiring manager does not know in advance the curricula of all available candidates. For each candidate, a decision has to be taken immediately after the end of the interview. The secretary problem requires to find a policy to maximize the probability of selecting the best candidate [46].

The best known strategy to solve the secretary problem is quite simply but it produces surprisingly good results. In fact, suppose that n secretaries have to be interviewed and let us fix an integer r, such that $1 \leq r \leq n$. The first r candidates will form a *reference set* denoted as $\mathcal{R}$. After this we will explore the remaining $n-r$ candidates and if we find a candidate which is better than all the candidates in $\mathcal{R}$ then this candidate will be hired. The choice of r is the trickiest part of the algorithm: in fact if $r = 1$, then the algorithm incur in poor performances because it could select the second worst candidate. If $r = n - 1$ the algorithm generates the optimal solution but it has to view all n candidates. Theoretical analysis show that if we fix $r = \left\lceil \frac{n}{e} \right\rceil$ the probability of making the best choice is at least $\frac{1}{e}$ [46]. The U2G-I algorithm aims at finding the best $NMAX$ groups assuming that the algorithm can not know all available groups. If we suppose $NMAX = 1$ the problem to solve reduces to the secretary problem. [46] considers some extensions of the traditional secretary problem in which more than one candidates can be selected. However, the problem considered in this paper (and the associated algorithm) is not covered in [46]; in addition, as a further and novel research contribution, we provide a bound on the correctness of the results yielded by the U2G-I algorithm. The rationale behind the algorithm is to build a *reference set* of size r, being r a threshold whose value will be specified later and to select a group if its compactness against u is better than the compactness of the worst group currently stored in the reference set. The algorithm uses two sets G_1 and G_2 to generate the set $GOOD$ (which, at the beginning, is empty). The sets G_1 and G_2 are built as follows:

1) The user agent issues r queries with $NMAX \leq r \leq |\mathcal{G}|$ and it gets as answer r groups. The set G_1 will contain the received groups and it plays the role of the reference set.
2) The G_2 set will contain the best $NMAX$ groups.
3) The U2G-I performs other $|\mathcal{G}| - r$ iterations and, at each iteration, it queries the DF and gets a group g^* with a compactness value equal to $\gamma_{u \to g^*}$. The group g^*

is *picked uniformly at random* among all the available groups.

4) Let $\overline{g}$ be the $NMAX$-best group in G_2, i.e., the group in G_2 showing the lowest level of compactness with u. If $\gamma_{u\to g^*} > \gamma_{u\to\overline{g}}$ we say that g^* *beats* $\overline{g}$. In this case, $\overline{g}$ is deleted from G_2 and it is replaced by g^*. In addition, if $\overline{g}$ *belongs also to* G_1, then g^* will be inserted in $GOOD$.

In other words, the set $GOOD$ is progressively built as follows: for each retrieved group g^*, the U2G-I algorithm checks if there exists at least one group $\overline{g}$ in G_2 such that g^* beats $\overline{g}$. In the affirmative case, a further check is performed: the U2G-I algorithm verifies whether $\overline{g}$ belongs also to G_1. This resembles the secretary problem in which a candidate is selected if she/he beats all of the candidates in the reference set. Of course, in our setting, we do not want that g^* beats all the groups in the reference set but we are satisfied if it beats at least one of the groups in the reference set and, to this purpose, we compare g^* with the worst group in G_2. If this check ends positively, then g^* is inserted in $GOOD$.

We are now in the position of analyzing the quality of the solution generated by the U2G-I algorithm. To perform our analysis, let $\widetilde{\phi}$ be the value of the objective function returned by U2G-I algorithm. Since groups are selected uniformly at random, we have that $\widetilde{\phi}$ is a random variable and its value equals $\widetilde{\phi} = \sum_{g\in\mathcal{G}} \gamma_{u\to g} x_{u\to g}$, being $x_{u\to g}$ a random variable equal to 1 if and only if g is selected, 0 otherwise. Let $OPT = \{\hat{g}_1, \ldots, \hat{g}_{NMAX}\}$ be the *optimal solution* where $\hat{g}_i$ is the i-th best group, i.e. the group having the i-the best value of compactness with respect to the user u. Finally, let $\hat{\phi}$ be the value of the objective function ϕ associated with the optimal solution OPT. As in the secretary problem, we can assess the quality of the produced solution by comparing the expected value of $\widetilde{\phi}$ with $\hat{\phi}$. In particular, we wish to prove the following result:

Theorem 4.2: Let $\widetilde{\phi}$ be the value of the solution found by executing the U2G-I algorithm and let $\hat{\phi}$ be the optimal solution. We have that: $E[\widetilde{\phi}] > \frac{1}{e}\hat{\phi}$

Theorem 4.2 informs us that the solutions produced by the U2G-I algorithm, *on average*, differ from the optimal one by a constant factor, independently of the number $|\mathcal{G}|$ of available groups. This is therefore a tight bound on the correctness of the results produced by the U2G-I algorithm. To prove Theorem 4.2 we need the following preliminary result:

Lemma 4.3: Suppose to run the U2G-I algorithm in such a way as to $|G_1| = r$ and let $GOOD$ be the solution that the U2G-I algorithm produces. Finally, let $\hat{g}_i$ be the i-th best group. The probability that $\hat{g}_i$ belongs to $GOOD$ is at least $\frac{r}{|\mathcal{G}|}\ln\left(\frac{|\mathcal{G}|}{r}\right)$, i.e.: $\Pr[\hat{g}_i \in GOOD] \geq \frac{r}{|\mathcal{G}|}\ln\left(\frac{|\mathcal{G}|}{r}\right)$

Proof Suppose that the U2G-I algorithm selects a group, say g^*, which belongs to the solution $GOOD$ generated by the U2G-I algorithm. We wish to compute the probability that g^* coincides with the i-th best group $\hat{g}_i$. This means that there exists an iteration j in which g^* has been selected, with $r < j \leq |\mathcal{G}|$ and $g^* = \hat{g}_i$. Therefore, the probability that $\Pr[\hat{g}_i \in GOOD]$ is equal to:

$$\Pr\left(\bigcup_{j=r+1}^{|\mathcal{G}|} \{g^* \text{ selected in the } j\text{-th iteration and } g^* = \hat{g}_i\}\right) \tag{4}$$

Since the events above are mutually disjoint, we have that $\Pr[\hat{g}_i \in GOOD]$ is equal to:

$$\Pr[\hat{g}_i \in GOOD] = \\ = \sum_{j=r+1}^{|\mathcal{G}|} \Pr\left(g^* \text{ selected in the } j\text{-th iteration and } g^* = \hat{g}_i\right) \tag{5}$$

By applying the definition of conditional probability we have that

$$\Pr\left(g^* \text{ selected in the } j\text{-th iteration and } g^* = \hat{g}_i\right) = \\ \Pr\left(\hat{g}_i \text{ selected in the } j\text{-th iteration} | g^* = \hat{g}_i\right) \times \Pr\left(g^* = \hat{g}_i\right) = \\ = \frac{r}{j} \times \frac{1}{|\mathcal{G}|} \tag{6}$$

The previous equality is explained as follows: since all groups are selected uniformly at random, then the probability of selecting g^* is at least equal to $\frac{1}{|\mathcal{G}|}$. In addition, according to Step 4 in the U2G-I algorithm, g^* will be part of the $GOOD$ set if and only if there exists at least one group $\overline{g} \in G_1$ such that g^* beats $\overline{g}$. Since $|G_1| = r$ by hypothesis, there are r chances that the group $\overline{g}$ exists. Therefore, the probability that g^* has been selected in the j-th iteration is equal to $\frac{r}{j}$. We get:

$$\Pr[\hat{g}_i \in GOOD] = \sum_{j=r+1}^{|\mathcal{G}|} \frac{r}{j} \times \frac{1}{|\mathcal{G}|} = \frac{r}{|\mathcal{G}|} \sum_{j=r+1}^{|\mathcal{G}|} \frac{1}{j}$$

Since

$$\sum_{j=r+1}^{|\mathcal{G}|} \frac{1}{j} \geq \int_r^{|\mathcal{G}|} \frac{1}{x} dx = \ln\left(\frac{|\mathcal{G}|}{r}\right)$$

we get

$$\Pr[\hat{g}_i \in GOOD] \geq \frac{r}{|\mathcal{G}|}\ln\left(\frac{|\mathcal{G}|}{r}\right)$$

which ends the proof. □

By means of Lemma 4.3, we have the following result:

Theorem 4.4: For any value of r, the U2G-I algorithm generates a solution such that the expected value of $\widetilde{\phi}$ satisfies the following inequality:

$$E[\widetilde{\phi}] > \frac{r}{|\mathcal{G}|}\ln\left(\frac{|\mathcal{G}|}{r}\right)\hat{\phi}$$

being $\hat{\phi}$ the optimal solution.

Proof By definition of $\widetilde{\phi}$ and expected value we get:

$$E[\widetilde{\phi}] = E\left[\sum_{g\in\mathcal{G}} \gamma_{u\to g} x_{u\to g}\right] = \sum_{g\in\mathcal{G}} E[\gamma_{u\to g} x_{u\to g}] = \\ \sum_{g\in\mathcal{G}} \Pr[g \in GOOD]\gamma_{u\to g} \tag{7}$$

because $E[x_{u\to g}] = \Pr[x_{u\to g}=1]\times 1 + \Pr[x_{u\to g}=0]\times 0 = \Pr[x_{u\to g}=1] = \Pr[g \in GOOD]$. We can bound the left side of Equation 7 by focusing only on the $NMAX$ optimal groups belonging to OPT, i.e.:

$$E[\widetilde{\phi}] = \sum_{g\in\mathcal{G}} \Pr[g \in GOOD]\gamma_{u\to g} \geq \sum_{i=1}^{NMAX} \Pr\left[\hat{g}_i \in GOOD\right]\gamma_{u\to \hat{g}_i}. \tag{8}$$

By Lemma 4.3 we get $\Pr\left[\hat{g}_i \in GOOD\right] \geq \frac{r}{|\mathcal{G}|}\ln\left(\frac{|\mathcal{G}|}{r}\right)$, which allows us to write:

$$E[\widetilde{\phi}] \geq \sum_{i=1}^{NMAX} \frac{r}{|\mathcal{G}|}\ln\left(\frac{|\mathcal{G}|}{r}\right)\gamma_{u\to\hat{g}_i} = \frac{r}{|\mathcal{G}|}\ln\left(\frac{|\mathcal{G}|}{r}\right)\sum_{i=1}^{NMAX}\gamma_{u\to\hat{g}_i} = \frac{r}{|\mathcal{G}|}\ln\left(\frac{|\mathcal{G}|}{r}\right)\hat{\phi} \tag{9}$$

and this ends the proof. □

We are now able to prove Theorem 4.2. In particular, observe that, according to Theorem 4.4, for any choice of r we get $E[\widetilde{\phi}] \geq \frac{r}{|\mathcal{G}|}\ln\left(\frac{|\mathcal{G}|}{r}\right)\hat{\phi}$. In the previous inequality, the worst case occurs when the function $f(r) = \frac{r}{|\mathcal{G}|}\ln\left(\frac{|\mathcal{G}|}{r}\right)$ achieves its maximum value. Such a value can be easily computed by means of calculus. In particular, set $y = \frac{r}{|\mathcal{G}|}$ and observe that $f(y) = y\ln\frac{1}{y} = -y\ln y$. We have:

$$f'(y) = -\ln y - 1 \qquad f''(y) = -\frac{1}{y}$$

Observe that $f'(y) = 0$ if $y = y^* = \frac{1}{e}$ and $f''(y^*) < 0$. In addition, we have that $\lim_{y\to 0} f(y) = 0$ and $\lim_{y\to+\infty} f(y) = -\infty$. Therefore $f(y)$ achieves its maximum at $y = y^*$, i.e., if $\frac{|\mathcal{G}|}{r} = \frac{1}{e}$ which implies $r = \frac{|\mathcal{G}|}{e}$. The maximum of $f(r)$ is then equal to $\frac{1}{e}$. This also implies that $E[\widetilde{\phi}] > \frac{1}{e}\hat{\phi}$, i.e., the expected value of the solution found by the U2G-I algorithm differs from the optimal one by a constant factor and this proves Theorem 4.2.

C. The group agent task

In this section we describe the algorithm implemented by the group agent. Let K be the set of the k users joined with the group g, where $k \leq n_{KMAX}$, being n_{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 (see Figure 3): *Step 1*. 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 (action 1 of Figure 3). *Step 2*. After the reception of the responses from the contacted user agents (action 2 of Figure 3), it computes the compactness measure

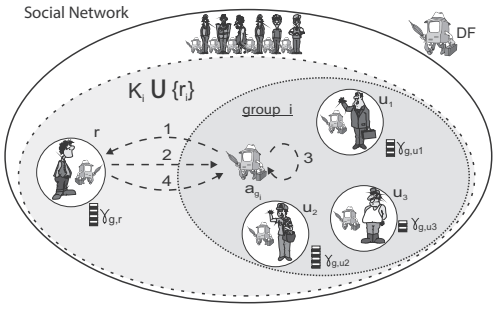

Fig. 3. Group agent task schema

$\gamma_{g\to u}$ between the profile of each user $u \in K\bigcup\{r\}$ and the profile of the group g (action 3 of Figure 3). *Step 3*. Now, let π a real value, ranging in $[0..1]$, representing a threshold for the compactness, such that a user u is considered as acceptable to join with if $\gamma_{g\to u} > \pi$. Then, the agent a_g stores in a set $GOOD$ those users $u \in K\bigcup\{r\}$ such that $\gamma_{g\to u} > \pi$ (if there exist more than n_{KMAX} users satisfying this condition, the n_{KMAX} users having the highest values of compactness are selected). If $r \in GOOD$, a_g accepts its request to join with the group (action 4 of Figure 3). Moreover, for any user $u \in K$, with $u \notin GOOD$, a_g deletes u from the group.

V. Evaluation

In this section we describe some experiments we performed to evaluate the effectiveness of the U2G matching algorithm. To this purpose, we have implemented an OSN simulator, called U2G-Sim, written in JAVA, capable of simulating our algorithm on a given OSN.

As a measure of the internal *compactness* of a group, we use the concept of *average compactness*, an extension of the average dissimilarity commonly exploited in Clustering Analysis [47]; it is defined as the average of the compactness values 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 compactness of g, denoted as AC_g is computed as $\sum_{x,y\in g, x\neq y}\gamma_{x\to y}/|g|$.

In order to measure the global compactness of the groups of the social network, we compute the mean MAC and standard deviation DAC of all the AC_g. We evaluate the performances of our algorithm on real data extracted from the well-known OSNs EPINIONS and CIAO. The two datasets have been crawled in the context of the research described in [48], and are publicly available at http://www.public.asu.edu/~jtang20/datasetcode/truststudy.htm. EPINIONS and CIAO are product review sites providing a trust network among users. Such sites provide a sensible platform to study trust in an online world.

In both these datasets we have, for each user, her/his profile, her/his ratings and her/his trust relations. For each rating, we have the product name and its category, the rating score and the helpfulness of this rating. The EPINIONS dataset consists of $22,166$ users, while CIAO contains only $12,375$ users, but it has more close-knit trust relationships. In a first experiment, we apply our matching algorithm to the CIAO dataset, assuming a number of 75 groups; users were randomly distributed in these groups. As we can see in Table I, we

TABLE I
VALUES OF THE PARAMETERS USED IN THE U2G-SIM SIMULATOR FOR CIAO AND EPINIONS DATASET.

	θ	τ
Value	0.8	0.29
	$KMAX$	$NMAX$
Value	250	20

have set τ and π to a value of 0.29 because this value produces the best results in the simulation. Moreover, we set $KMAX = 250$, $WS_u = 0.5$ and $WS_g = 0.5$ for each user u and group g; in this way we assume that all the users and all the groups give the same importance to similarity and trust. The profile p_u of a user u has been generated as follows:

- each values $I_u(c)$ is the percentage of reviews in the category c provided by u;
- A_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); we made an analysis on a set of 200 Facebook groups, obtaining the percentages above.
- B_u contains two boolean variables, representing the user's attitude to: (*i*) give to the products an average ranking higher than 3; (*ii*) obtain a helpfulness of their reviews higher than 3;
- in F_u, we insert the actual friends of u, i.e. the users in which he trusts.

We have called U2G-comp the version of our algorithm described in Section IV. In order to analyse the role of the trust, we have also used a version of our algorithm, called U2G-diff, that does not consider the trust component in the compactness, giving importance only to the similarity (i.e., $WS_u = 1$ for all the users u). We have repeated the experiment above on the EPINONS dataset. In this case, we have assumed the existence of 100 groups, and we have used $WS = 0.5$.

Figure 4 shows the results of the simulations in terms of MAC. We highlight that U2G-comp algorithm improves the initial MAC of the OSN for the CIAO dataset of about 14%, while by using the EPINIONS dataset the difference of the final improvement is equal to 15 %. Interestingly, in this situation for the two datasets the U2G-diff algorithm improves the MAC only in the first steps, finally converging to a value that is smaller than the initial one. Also in terms of DAC, the results illustrated in Figure 5 show that U2G-diff performs significantly worse with respect to U2G-comp, achieving the values of 0.21 and 0.24 for the CIAO and EPINIONS dataset, respectively. These results indicate that the use of the trust measure is essential to produce MAC improvements in real OSNs using our approach.

We repeated the simulations above, by varying the value of WS and we considered 7 different values of WS, namely 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8 and 1. For the CIAO dataset, Figure 6 shows that the values 0.5 is the best choice for WS, while the performances decreases if WS increases, i.e. if we progressively increase the role of similarity. Also the analysis performed varying the WS coefficient for the EPINIONS dataset shows results similar to those produced for CIAO, confirming that the best choice for WS is 0.5. It is interestingly to highlight that the choice $WS = 1$ corresponds to disregard trust information. In this case it is equivalent to use U2G-comp or U2G-diff algorithm.

Finally, we studied the stability of the groups after a period of application of our matching algorithm. In particular, using the timestamps associated with each tuple present in the EPINIONS dataset, we have trained the formation of the groups by using as training set the first 40,000 tuples. We applies our algorithm and obtained a MAC equal to 0.45. A subsequent application of the U2G-comp algorithm to the test set formed by the remaining 10,000 tuples of the dataset shows that the stability of the groups is guaranteed, as reported in Figure 7. In the same figure, we also reported the results of the analogous experiment performed by using the U2G-diff algorithm. This latter experiment demonstrates that the use of the sole similarity as a compactness measure not only leads to a MAC significantly lower than that obtained using a trust-based compactness, but also the groups so formed do not show a real compacteness when analyzed on the basis of the test-set.

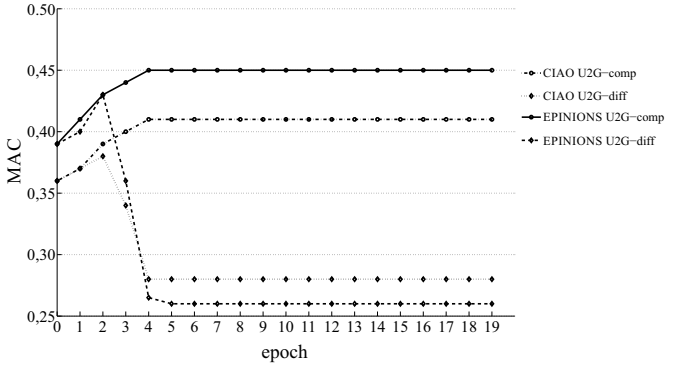

Fig. 4. Variation of MAC obtained with the U2G-comp and U2G-diff algorithms, for the CIAO and EPINIONS OSNs.

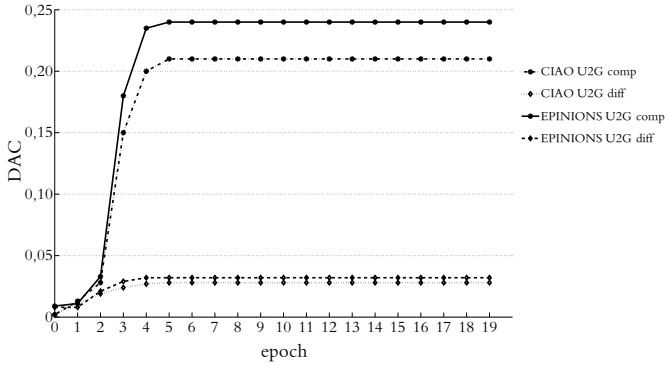

Fig. 5. Variation of DAC obtained with the U2G-comp and U2G-diff algorithms, for the CIAO and EPINIONS OSNs.

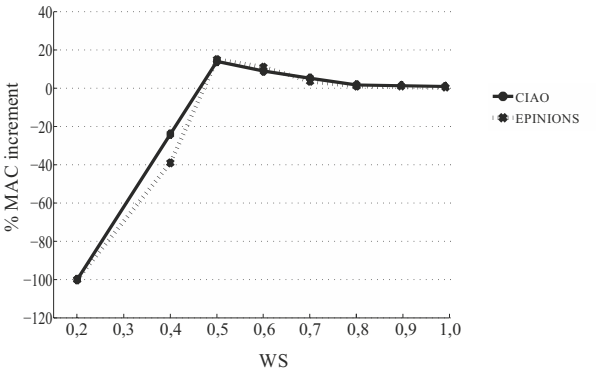

Fig. 6. Variation of MAC vs WS obtained with the U2G-comp algorithm, for the CIAO and the EPINIONS OSN.

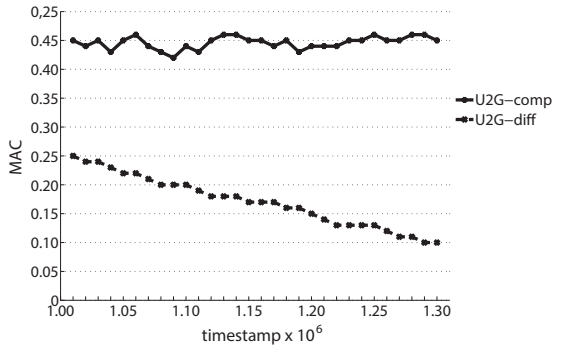

Fig. 7. Variation of MAC vs timestamp obtained with the U2G-comp and U2G-diff algorithm, for the EPINIONS OSN.

A. Computational Performance

We conclude the discussion of experimental results by illustrating on the computational performance of our approach. All the experiments presented in this paper were executed on a 2Ghz I7 Intel Processor equipped with 8GB RAM. The time required for all the experiments presented in this section roughly amounts to 3 seconds per epoch. In addition, previous experiment show that 3-4 epochs are usually needed for producing the optimal solution and, therefore, we can conclude that our approach seems efficient on real life OSNs.

VI. Discussion and Conclusion

In this paper, we presented a User-to-Group matching algorithm, that allows a set of software agents associated with OSN users to dynamically and autonomously manage the evolution of the groups, by detecting for each user the most suitable groups to join with based on two measures, namely compactness and similarity. Moreover, the agents operate on behalf of the group administrators, such that a group agent accepts only those join requests that come from users whose profiles are compatible with the profile of the group. Our experiments on real social network data clearly showed that the execution of the matching algorithm increases the internal compactness of the groups composing the social network. In detail, we observe that:

1) The U2G algorithm is easy to implement and it quickly converges. Both these two features are particularly relevant in a real OSN (consisting of millions of users and thousands of groups).
2) The U2G algorithm is also *flexible* because, with some simple modifications, it can implement different cost functions (e.g., similarity, trust or a combination of them) to associate users with groups.
3) In line with some results from Recommender System literature [49], trust and similarity can be profitably combined to yield more accurate results. Our experiments provide evidence that when the compactness measure is used, we achieve an increase of MAC of about 23% with respect to a merely random assignment of users to groups. The usage of similarity alone yields an improvement of MAC in the order of about 14%.

Roughly speaking, the experimental results support our hypothesis that forming groups on the basis of the only user similarity is not the best choice to obtain intra-group compactness, and that a significant improvement of such a compactness can be easily achieved by considering trust measures.

The experiments prove that the use of our algorithm, which leads the users to form aggregation based on our notion of compactness, actually creates compact aggregations. In fact, we experimentally proved that the aggregation formed on the basis of a training data set remains stable also in the subsequent periods of time, showing that the users in a group with high compactness maintain both high similarity and high trust. We have also shown the this property is valid only by introducing trust in the definition of the compactness measure.

References

[1] F. Buccafurri, D. Rosaci, G. M. L. Sarné, and L. Palopoli, "Modeling cooperation in multi-agent communities," *Cognitive Systems Research*, vol. 5, no. 3, pp. 171–190, 2004.
[2] S. Gauch, M. Speretta, A. Chandramouli, and A. Micarelli, "User profiles for personalized information access," in *The Adaptive Web*, ser. LNCS, vol. 4321. Springer, 2007, pp. 54–89.
[3] E. Ferrara, "A large-scale community structure analysis in Facebook," *EPJ Data Science*, vol. 1, no. 9, pp. 1–30, 2012.
[4] J. Zhan, "Secure collaborative social networks," *Trans. Sys. Man Cyber Part C*, vol. 40, no. 6, pp. 682–689, Nov. 2010.
[5] Z. Wang, D. Zhang, X. Zhou, D. Yang, and Z. Yu, "Discovering and profiling overlapping communities in location-based social networks," *Trans. Sys. Man Cyber: Systems*, vol. 44, no. 4, pp. 499–509, 2014.
[6] S. Basu Roy, S. Amer-Yahia, A. Chawla, G. Das, and C. Yu, "Space efficiency in group recommendation," *The VLDB J.*, vol. 19, no. 6, pp. 877–900, 2010.
[7] M. Kasavana, K. Nusair, and K. Teodosic, "Online social networking: Redefining the human Web," *J. of Hospitality and Tourism Technology*, vol. 1, no. 1, pp. 68–82, 2010.
[8] J. Kim, H. Kim, H. Oh, and Y. Ryu, "A group recommendation system for online communities," *Int. J. of Information Management*, vol. 30, no. 3, pp. 212–219, 2010.
[9] U. Brandes, J. Pfeffer, and I. Mergel, *Studying Social Networks: A Guide to Empirical Research*. Campus, 2013.
[10] L. Egghe and R. Rousseau, "Brs-compactness in networks: Theoretical considerations related to cohesion in citation graphs, collaboration networks and the Internet," *Mathematical and Computer Modelling*, vol. 37, no. 78, pp. 879 – 899, 2003.
[11] X. Xiong, G. Zhou, X. Niu, Y. Huang, and K. Xu, "Remodeling the network for microgroup detection on microblog," *Knowledge and Information Systems*, pp. 1–23, 2013.
[12] M. Lv, L. Chen, and G. Chen, "Mining user similarity based on routine activities," *Information Sciences*, vol. 236, no. 0, pp. 17 – 32, 2013.
[13] J. Kleinberg, "Analysis of large-scale social and information networks," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 371, no. 1987, 2013.

[14] M. E. Newman, "Detecting community structure in networks," *The European Physical J. B-Condensed Matter and Complex Systems*, vol. 38, no. 2, pp. 321–330, 2004.

[15] E. B. Ahmed, A. Nabli, and F. Gargouri, "Group extraction from professional social network using a new semi-supervised hierarchical clustering," *Knowledge and Information Systems*, pp. 1–19, 2013.

[16] F. Messina, G. Pappalardo, D. Rosaci, C. Santoro, and G. M. L. Sarné, "Hyson: A distributed agent-based protocol for group formation in online social networks," in *Multiagent System Technologies*, ser. Lecture Notes in Computer Science. Springer, 2013, vol. 8076, pp. 320–333.

[17] J. Doodson, J. Gavin, and R. Joiner, "Getting acquainted with groups and individuals: Information seeking, social uncertainty and social network sites," in *7th Int. AAAI Conf. on Weblogs and Social Media*, 2013.

[18] W. Zeng and L. Chen, "Recommending interest groups to social media users by incorporating heterogeneous resources," in *Recent Trends in Applied Artificial Intelligence*, ser. LNCS. Springer, 2013, vol. 7906, pp. 361–371.

[19] C. Dwyer, S. R. Hiltz, and K. Passerini, "Trust and privacy concern within social networking sites: A comparison of Facebook and MySpace." in *AMCIS'07*, 2007, p. 339.

[20] J. Fogel and E. Nehmad, "Internet social network communities: Risk taking, trust, and privacy concerns," *Computers in Human Behavior*, vol. 25, no. 1, pp. 153–160, 2009.

[21] D. Rosaci, "Trust measures for competitive agents," in *Knowledge-based Systems*, vol. 28. Elsevier, 2012.

[22] 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*, vol. 24, no. 1, pp. 31–50, 2011.

[23] P. De Meo, G. Quattrone, D. Rosaci, and D. Ursino, "Dependable recommendations in social internetworking," in *Web Intelligence and Intelligent Agent Technologies, 2009*. IET, 2009, pp. 519–522.

[24] D. Rosaci and G. M. L. Sarné, "Recommending multimedia Web services in a multi-device environment," *Information Systems*, vol. 38, no. 2, pp. 198–212, 2013.

[25] L. Palopoli, D. Rosaci, and G. M. L. Sarné, "A multi-tiered recommender system architecture for supporting e-commerce," *Studies in Computational Intelligence 446, Intelligent Distributed Computing VI*, pp. 71–81, 2013.

[26] L. Palopoli, D. Rosaci, and G. M. L. Sarné, "Introducing specialization in e-commerce recommender systems," *Concurrent Engineering: Research and Applications*, vol. 21, no. 3, pp. 187–196, 2013.

[27] S. Amer-Yahia, S. Roy, A. Chawlat, G. Das, and C. Yu, "Group recommendation: Semantics and efficiency," *Proc. of the VLDB Endowment*, vol. 2, no. 1, pp. 754–765, 2009.

[28] L. Baltrunas, T. Makcinskas, and F. Ricci, "Group recommendations with rank aggregation and collaborative filtering," in *Proc. of the ACM Conf. on Recommender Systems 2010*. ACM Press, 2010, pp. 119–126.

[29] X. Liu, A. Datta, K. Rzadca, and E.-P. Lim, "Stereotrust: a group based personalized trust model," in *Proc. of the 18th ACM conf. on Information and knowledge management*. ACM, 2009, pp. 7–16.

[30] I. Cantador and P. Castells, "Building emergent social networks and group profiles by semantic user preference clustering," in *2nd Int. Works. on Semantic Network Analysis, at the 3rd European Semantic Web Conf.)*, 2006, pp. 40–53.

[31] V. Vasuki, N. Natarajan, Z. Lu, B. Savas, and I. Dhillon, "Scalable affiliation recommendation using auxiliary networks," *ACM Transactions on Intelligent Systems and Technology*, vol. 3, no. 1, p. 3, 2011.

[32] E. Spertus, M. Sahami, and O. Buyukkokten, "Evaluating similarity measures: a large-scale study in the orkut social network," in *Proc. of the ACM Int. Conf. on Knowledge Discovery and Data Mining (SIGKDD'05)*. ACM, 2005, pp. 678–684.

[33] W. Chen, D. Zhang, and E. Chang, "Combinational collaborative filtering for personalized community rrecommendation," in *Proc. of the ACM Int. Conf. on Knowledge Discovery and Data mining (SIGKDD'08)*. ACM, 2008, pp. 115–123.

[34] W.-Y. Chen, J.-C. Chu, J. Luan, H. Bai, Y. Wang, and E. Y. Chang, "Collaborative filtering for orkut communities: Discovery of user latent behavior," in *Proc. of the 18th Int. Conf. on World wide web*. ACM, 2009, pp. 681–690.

[35] A.A.V.V., "Advogato's trust metric," `http://www.advogato.org/trust-metric.html`, 2013.

[36] J. Golbeck and J. Hendler, "Inferring binary trust relationships in web-based social networks," *ACM Transactions on Internet Technology*, vol. 6, no. 4, pp. 497–529, 2006.

[37] R. Guha, R. Kumar, P. Raghavan, and A. Tomkins, "Propagation of trust and distrust," in *Proc. of the Int. Conf. on World Wide Web (WWW'04)*. ACM, 2004, pp. 403–412.

[38] S. Sen and N. Sajja, "Robustness of reputation-based trust: Boolean case," in *Proc of the 1st Int. Joint Conf. on Autonomous Agents and Multiagent Systems*, ser. AAMAS '02. ACM, 2002, pp. 288–293.

[39] S. D. Ramchurn, N. R. Jennings, C. Sierra, and L. Godo, "Devising a trust model for multi-agent interactions using confidence and reputation," *Applied Artificial Intelligence*, vol. 18, no. 9-10, pp. 833–852, 2004.

[40] G. Lax and G. Sarné, "CellTrust: a reputation model for C2C commerce," *Electronic Commerce Research*, vol. 8, no. 4, pp. 193–216, 2006.

[41] L. Vercouter and G. Muller, "Liar: achieving social control in open and decentralized multiagent systems," *Applied Artificial Intelligence*, vol. 24, no. 8, pp. 723–768, 2010.

[42] D. Rosaci, G. Sarnè, and S. Garruzzo, "Integrating trust measures in multiagent systems," *International Journal of Intelligent Systems*, vol. 27, no. 1, pp. 1–15, 2012.

[43] D. Rosaci and G. Sarnè, "Multi-agent technology and ontologies to support personalization in B2C e-Commerce," *Electronic Commerce Research and Applications*, vol. 13, no. 1, pp. 13–23, 2014.

[44] T. DuBois, J. Golbeck, and A. Srinivasan, "Predicting trust and distrust in social networks," in *Privacy, Security, Risk and Trust, 4rd Int. Conf. on and 3rd Int. Conf. on Social Computing*. IEEE, 2011, pp. 418–424.

[45] H. Liu, E.-P. Lim, H. W. Lauw, M.-T. Le, A. Sun, J. Srivastava, and Y. Kim, "Predicting trusts among users of online communities: an epinions case study," in *Proc. of the 9th ACM conf. on Electronic commerce*. ACM, 2008, pp. 310–319.

[46] P. Freeman, "The secretary problem and its extensions: A review," *Int. Statistical Review*, pp. 189–206, 1983.

[47] R. K. Pearson, T. Zylkin, J. S. Schwaber, and G. E. Gonye, "Quantitative evaluation of clustering results using computational negative controls," in *Proc. of 2004 SIAM Int. Conf. on Data Mining*, 2004, pp. 188–199.

[48] J. Tang, X. Hu, H. Gao, and H. Liu, "Exploiting local and global social context for recommendation," in *Proc. of the Int. Joint con. on Artificial Intelligence (IJCAI 2013)*. AAAI Press, 2013, pp. 2712–2718.

[49] C. Ziegler and J. Golbeck, "Investigating interactions of trust and interest similarity," *Decision Support Systems*, vol. 43, no. 2, pp. 460–475, 2007.