Università degli Studi Mediterranea di Reggio Calabria
Archivio Istituzionale dei prodotti della ricerca

Using local trust measures to form agent CoT groups1

This is the peer reviewd version of the followng article:

Original
Using local trust measures to form agent CoT groups1 / Fortino, G., Fotia, L., Messina, F., Rosaci, D., Sarne, G.M.L.. - In: INTELLIGENZA ARTIFICIALE. - ISSN 1724-8035. - 14:1(2020), pp. 77-88. [10.3233/IA-190039]

Availability:
This version is available at: https://hdl.handle.net/20.500.12318/74274 since: 2020-12-29T12:27:44Z

Published
DOI: http://doi.org/10.3233/IA-190039
The final published version is available online at:https://content.iospress.com/articles/intelligenza-

Terms of use:
The terms and conditions for the reuse of this version of the manuscript are specified in the publishing policy. For all terms of use and more information see the publisher's website

Publisher copyright

This item was downloaded from IRIS Università Mediterranea di Reggio Calabria (https://iris.unirc.it/) When citing, please refer to the published version.

(Article begins on next page)

25 October 2026

This is postprint version of the paper published in Intelligenza Artificiale 14(1): 33-44. 2020.
The final version is available at DOI: 10.3233/IA-190039

Using Local Trust Measures to Form Agent CoT Groups [1]

Giancarlo Fortino [a,*], Lidia Fotia [b], Fabrizio Messina [c], Domenico Rosaci [d] and Giuseppe M. L. Sarné [b]
[a] *Department DIMES, University of Calabria, Rende (CS)*
E-mail: giancarlo.fortino@unical.it
[b] *DICEAM, University Mediterranea of Reggio Calabria, Italy*
E-mail: {lidia.fotia,sarne}@unirc.it
[c] *DMI, University of Catania, Italy*
E-mail: messina@dmi.unict.it
[d] *DIIES, University Mediterranea of Reggio Calabria, Italy*
E-mail: domenico.rosaci@unirc.it

Abstract. IoT devices dealing with complex tasks usually require powerful hardware capabilities or, as a possible alternative, to get on the Cloud those resources they need. When an IoT device is "virtualized" on the Cloud, it can take benefit from relying on one or more software agents and their social skills to mutually interact and cooperate. In particular, in a Cloud of Things scenario, where agents cooperate to perform complex tasks, the choice of a partner is a sensitive question. In such a context, when an agent is not capable to perform a reliable choice then, like real social communities, it can ask information to other agents it considers as trustworthy. In order to support agents in their partner choices, we conceived a local trust model, based on reliability and reputation measures coming from its ego-network, adopted to partition the agents in groups by exploiting trust relationships to allow agents to be associated with the most reliable partners. To this aim, we designed an algorithm to form agent groups by exploiting available local trust measures and the results obtained in a simulated scenario confirmed the potential advantages of this approach.

Keywords: Cloud of Things; Internet of Things; Multiagent System; Reputation; Trust; Voting

1. Introduction

Today, the "Internet of Things" (IoT) world performs complex tasks requiring increasing hardware and power capabilities to IoT devices. These device requirements become particularly crucial in presence of small and low-cost devices. In the meantime, Cloud Computing (CC) was introduced as a main Internet information technology addressed to share services, processes and data stored to form knowledge accessible in distributed environments. In this scenario, both IoT and CC converged into the so called Cloud-of-Things (CoT) [2,3]. This integration is motivated from the necessity of supporting the computational and storing requirements [4] coming from a wide number of heterogeneous, small and low-cost IoT devices [5], to create new services also available in nomadic scenarios [6]. Furthermore, cooperation among IoT devices, for instance to realize complex tasks, can be encouraged by giving them the opportunity to be associated with software agents to exploit their social attitudes [7,8,9,10]. In this scenario, agents have the problem of selecting the most reliable partners for cooperation. Unfortunately, in some cases, it could happen that agents do not have enough information about other peers. Indeed, the choice of a reliable partner needs of suitable

[1] A preliminary version of this study was presented at the 20th Workshop "from Objects to Agents" (WOA 2019), 26-28 June 2019, Parma (Italy) [1].

[*] Corresponding author. E-mail: sarne@unirc.it

0000-0000/14/$00.00 © 2014 – IOS Press and the authors. All rights reserved

information that could be also required as recommendations to trustworthy agents.

We propose of supporting this process by encouraging agents to form groups of reliable recommenders agents. Even though a common viewpoint considers that groups should be formed on the basis of some criterion representing commonalities of relations, interests and/or preferences [11], it is known that a high level of mutual trustworthiness existing among the group members is an important social property within a community [12,13,14].

Consequently, to maximize the benefits that an agent can receive in joining a group (and vice versa), the adoption of trust measures, usually derived by combining reliability and reputation measures, could improve both individual and global satisfaction [15,16,17,18]. Therefore, we regard the trust-based processes to form agent groups of reliable recommenders over a CoT context. This approach significantly improves the activities also of those IoT devices poorly equipped. The synergy deriving by integrating CoT and software agents allows to boost social activities (i.e., interacting, forming groups, federating groups or CCs and so on) having place therein e.g. it could be meaningless with respect to devices only providing a measure on request.

However, as it happens in real user communities, in place of the global reputation, it is possible to adopt a *local reputation* [19] approach where the reputation value is derived by the opinions coming from the friends (or friends of friends and so on) of an agent, i.e. its ego-network (see Figure 1). It is usual when a user can not reach a reliable decision so that commonly he/she will adopt a local approach requesting an opinion to his/her friends. This local approach gives important benefits, among which *i*) avoiding heavy computational tasks, communication overloads in collecting opinions as well as in evaluating the sources trustworthiness and *ii*) significantly increasing the system reactivity.

To further improve the effectiveness of such a process we suggest to drive the formation of agent groups by exploiting a measure obtained by combining the local trust (formed by reliability and local reputation) and helpfulness. In order to verify the performance of this process, we considered a competitive CoT environment where heterogeneous devices consume/produce services and/or extract/exchange knowledge assisted by personal software agents working over the CC. Each associated agent will be able to support the group changes of its device in the CoT environment performed based on the temporary device convenience. Note that in the following each IoT device and its associated agent will be considered as the same entity.

The basic idea is that the generic consumer agent when chooses a data service (s) supplied by a provider agent, should consider its past experiences. If the agent is unable to make a reliable choice, then it could exploit the recommendation (i.e., a value belonging to the interval $[0, 1]$) provided by the community [20]. Moreover, we suppose that among agents belonging to the same group recommendations/opinions will be provided for free, otherwise a fee has to be paid for the recommendation/opinion. Therefore, groups/agents will be interested in accepting/belonging to those agents/groups having a high reliability and helpfulness by leading this approach to realize the desired competitive scenario. To evaluate the helpfulness of an agent we consider the relevance of its recommendations, while for a group we assume that it is the average of the helpfulness of its members.

Like human societies [21], we adopt a voting mechanism to carry out the group affiliation process. To this purpose, we designed a distributed algorithm for group formation (see Section 5) that we verified, in terms of efficiency and effectiveness, by means of some experiments on a simulated agent CoT scenario, which confirmed our expectations.

The rest of the paper is organized as follows. Section 2 refers to the related literature, Section 4 describes the adopted local trust model and voting mechanism, while Section 5 presents an algorithm to form groups. The experimental results are dealt in Section 6 and, finally, in Section 7 some conclusions and future development are drawn.

2. Related Work

Trust systems can be profitably exploited in open, competitive and distributed scenario to limit the risks to be engaged with unreliable partners [22,23,24,25].

In particular, in social communities, the aggregation rules [26], informative sources [27] and the modalities for inferring trust [28] are the factors that must be considered in computing trust measures.

In this respect, to solve the problem of suggesting to a group/member of a community if accepting/joining with a candidate/group, several trust-based approaches have been proposed. For instance, in [29] the authors verify that trust-based groups are more stable over time

with respect to groups formed without taking into account the contribute given by trust. Indeed, the expectations of receiving benefits is higher among the members of trust-formed groups. But, they not consider the local trust to obtain the group formation. In large communities where each member usually interacts only with a narrowest share of the community members, the adoption of local trust mechanisms is predominant with respect to the global one. Indeed, some studies found that the accuracy of local trust is greater when a personal viewpoint is used [30,31], while its narrower horizon depth contributes to reduce the computational costs [32].

In social contexts, it is conveniently to represent trust processes by means of a graph, named *trust network*, where nodes and oriented edges represent the members and their trust relationships, respectively. In such a way, the topological properties of the trust network help to study trust properties. In particular, Golbeck et al. [33] adopt a variant of the Breadth First Search to gather the reputation scores and, by using a voting mechanism, to compute an updated reputation rate for each user. To avoid overloading the system, in [34] updated trust scores are propagated only by using fixed length paths. Examples of local trust approaches are adopted by the TidalTrust [35] and MoleTrust [36] algorithms. The first one exploits the closer neighbors to compute its trust predictions, also by ignoring part of the neighbors if the trust network is too sparse. The second one performs a backward exploration by fixing a maximum depth in the search-tree of the trust network to calculate trust scores by using at depth x only the trust scores at depth $x-1$. All these approaches, unlike our work, do not use the advantages introduced by the concept of ego-network.

Another characteristic of our proposal is that of adopting a voting to reach a decision within a group. The adoption of a voting mechanism [37] is able to optimize the social utility and limiting conflicts [38], although the risks of manipulations always exist and, particularly, in software agent communities where agents can easily realize different malicious manipulations [39]. However, in huge communities a local voting might represent the best solution with respect to the difficulties of a global voting [40]. Similarly to a voting process also trust-based decisions place a bet on the basis of the expectations to receive some benefits [41] by one or more future events or behaviors [42]. Therefore, local trust and local voting have some intrinsic characteristics in common that can be usefully exploited in real or virtual communities denoted by a great (sparse) population in presence of poor communication infrastructure or storage and power constraints (like some IoT devices).

To this aim, [43] presents a local trust-based voting system, working in a mobile wireless scenario, where a node is admitted in a transmission path on the basis of its trustworthiness as perceived by the other nodes. The actual trust of a node is propagated among neighbors placed at one hop of distance on an oriented trust network by combining their confidence values assumed as trust measures. A node will be trusted/distrusted by using a local voting scheme. *SocialTrust* [44] is a framework to realize users' social trust groups by adopting a friends-of-friends relationship model, where by limiting the radius of this approach it is also possible to limit the impact of malicious users. In *SocialTrust* each user rates each other user he/she interacted with and converts such rates in a binary vote. In particular, a vote is a pair (user, vote), where user is a unique user identifier (the profile number) and vote is either "good" or "bad". Three voting mechanisms characterized by increasing levels of security and resilience can be adopted: (i) open voting; (ii) restricted voting; and (iii) trust-aware restricted voting. In particular, open voting is subject to ballot stuffing. By restricting the total size of vote allocated to each user, this restricted voting scheme avoids the problem of vote stuffing by a single user. They have no assurances that a malicious user will choose to vote truthfully for other users it has actually interacted with, but we do know that the total amount of voter fraud is constrained. Unfortunately, such a voting scheme is subject to collusive vote stuffing, in which many malicious users collectively decide to boost or demote the feedback rating of a selected user. Finally, to handle the problem of collusive vote stuffing, the authors advocate a weighted voting scheme in which users are allocated voting points based on how trustworthy they are.

Unfortunately, any ideal global or local voting procedure exists because all of them can be affected by manipulation, like strategic vote. This aspect is very critical for software agent communities, indeed agents can efficiently and effectively examine manifold manipulation opportunities, but this problem is assumed as orthogonal with respect to the focus of our proposal where trust drives voting.

Finally, some trust systems have been conceived for IoT and CC contexts. In literature, the approaches consider only neighbors for the calculation of the trust and they does not allow to obtain a correct value of local trust, as demonstrated in our work.

For instance, in [45] a *word of mouth* approach is used to propagate IoT devices' trust evaluations to the other nodes and in [46] each node evaluates the trustworthiness of its friend nodes and the opinions of the common friends (by adopting local trust measures). In the same social context, Chen et al. proposed in [47] a trust system to take into account the dynamic evolution of social relationships and self-adapting to trust fluctuations. In [48], for improving the performance of a grid of agent-based sensors, monitoring traffic flows on the roads by analyzing acoustical signals generated by vehicles in their motion, a distributed trust-system is built where each sensor-agent interacts only with its neighbor agents.

In a cooperative context, [49] gives attention to evaluate the skills of heterogeneous IoT devices in different cooperative tasks with a distributed approach. First and second-hand information and observations coming from the neighboring, are exploited to gather trustworthiness information, matching demand and offer for services, learning from past experiences and generating trust suggestion about other devices. BETaaS [50] is a system, also dealing with Big Data, which includes a trust model to esteem the reliability of monitored things and behaviors. Its trust model considers different aspects among which security, QoS, scalability, availability and gateways reputation. A Trust Management system for a CC marketplace in [51] evaluates a multidimensional trustworthiness of the CC providers by exploiting different sources and trust information. In [52], the authors designed a trust management architecture for CC marketplaces supporting customers in identifying trustworthy CC providers by verifying suspicious feedback arising by system and social threats. Unlike the previously mentioned approaches, a fully decentralized trust-based model for large-scale CC federations is described in [53] to allow any node to efficiently find the best collaborators in a set of candidate nodes without the need to explore the whole node space.

3. Scenario

We introduce a CoT environment where devices exchange services and/or extract/exchange knowledge supported by their associated software agents.

More formally, let A be the set of software agents living in the Cloud and let $G = \langle N, L \rangle$ be a directed graph that we, for convenience, adopt to represent the agents and their trust relationships, where N is the set of nodes (i.e., agents belonging to A), while L is the set of links (i.e., relationships occurring between two agents). For detail, see Section 4.

Moreover, we suppose that a generic agent is trying to join with one or more groups based on its real or perceived potential advantages. Now, we define the role of the agent administrator. It manages a group and can contact/remove those other agents it considers as useful/ineffective for joining with/removing from its group. The main objective of an agent administrator is that of maximizing the effectiveness of its group. In other words, the adopted mechanism implies that groups are interested to accept those agents having a high reliability; at the same time, agents are interested in being affiliated with those groups formed by agents denoted by a high reliability.

From an operative viewpoint, an agent (i.e., requester) can require a service to another agent (i.e., provider). To perform this task the requester can take benefit from its past experiences, but if the experiences are not sufficient to perform a good choice it can also require the opinions of other agents. In other words, if a_i (i.e., a generic i-th agent) has not a appropriate direct past experience about a provider agent a_j, it can ask a recommendation $\xi_{r,j} \in [0, 1]$ to another agent a_r. The selection of a_r is established by the algorithm described in the Section 5. Therefore, given that services are provided only for payment, while recommendations can also be provided for free only if a_r is in the same group of a_j, then the proposed scenario has a competitive nature. Each agent tries to introduce in each group other agents that are capable of delivering multiple services. At the same time, it requires to pay a fee for a service when an agent does not belong to its group.

4. The Local Trust Model and the Voting mechanism

The Local Trust Model. Let the oriented edges linking two nodes (i.e., agents) of the graph G be associated with the trust level ranging $[0, 1] \in \mathbb{R}$ that an agent has in another agent, where 0 means the minimum value and 1 is the maximum value. Therefore, the ego-network E_i of an agent $a_i \in A$ can be defined as a sub-graph $E_i \subseteq G$ including those nodes (i.e., agents) connected to a_i in a fixed depth. For definition, the ego-networks of a generic agent a including all the nodes of the virtual community for which a direct link (continue link) to a there exists and some

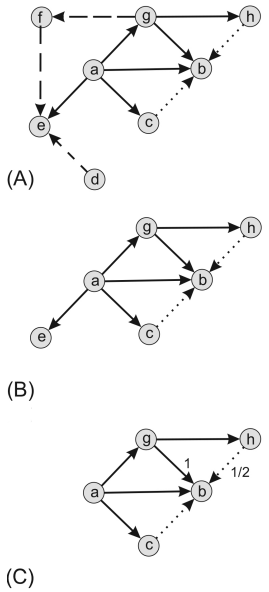

Fig. 1. The construction of ego-networks of the agent a: (A) the network of a; (B) the ego-networks of the agent a including all the nodes of the virtual community (nodes from a to f) for which a direct link to a there exists and (C) some other agents indirectly connected to a by a path of length 2 (e.g., all the agents connected to a by continue and dash links).

other agents indirectly connected to a by a path of fixed length (e.g., all the agents connected to a by continue and dash links). For example, the Figure 1 shows the three-step construction of the ego network of a. Consequently, given two generic nodes $i, j \in G$ (i.e., the associated agents a_i and a_j), we define the reliability $\rho_{i,j}$ that is a direct measure representing the confidence that a_i has about the capability of a_j to provide good suggestions. $\rho_{i,j}$ is an asymmetric measure (i.e., $\rho_{i,j} \neq \rho_{j,i}$) because the opinion of a_i about a_j may not coincide with the opinion of a_j about a_i. Also, we introduce $\phi_{i,j} \in [0,1] \in \mathbb{R}$ that is the feedback given by a_i to a_j after their interactions. Finally, $\rho_{i,j}$ is computed as:

$$\rho_{i,j} = \frac{1}{t} \cdot \sum_{k=1}^{t} \phi_{i,j}^{(k)}$$

on the basis of all the feedback $\phi_{i,j}^{(k)}$ given by a_i to a_j, where $\phi_{i,j}^{(k)}$ is the value of $\phi_{i,j}$ in the k-th interaction.

To this aim, let $\xi_{r,j}^{(s)} \in [0,1]$ be the s-th suggestion given by $a_r \in E_i$ about a_j, let $\epsilon_{i,r} \in [0,1]$ be the (average) helpfulness perceived by a_i about the capability of a_r to provide good suggestions[1], whilst $\phi_{i,j}^{(s)}$ is defined as the s-th value of $\phi_{i,j}$ defined before. In other words, the helpfulness $\epsilon_{i,r}$ of a_r perceived by a_i is computed, with respect to the feedback released by a_i for each of the m accepted suggestions provided by a_r to a_i about other agents or, more formally, as:

$$\epsilon_{i,r} = \frac{1}{m} \cdot \sum_{s=1}^{m} |\phi_{i,j}^{(s)} - \xi_{r,j}^{(s)}|$$

where the value m can be chosen by the system administrator due to a proper sensitivity analysis or a real time tuning.

To give an appropriate relevance to the recommender agents that in E_i are the closer to a_i, a parameter ω is introduced, computed as:

$$\omega_{i,r} = 2^{-(\widehat{l}_{(i,r)}-1)}$$

where $\widehat{l}_{(i,r)}$ is the shortest path, in terms of hops, between a_i and the recommender agent a_r.

Now, we define the *local reputation* $\sigma_{i,j}$ that is an indirect measures by taking into account how much, on average, the agents of E_i estimate the capability of a_j of having good interactions. By assuming that a_i, in its ego-network, is able to receive recommendations about a_j by a number z of recommenders, then $\sigma_{i,j}$ can be calculated as:

$$\sigma_{i,j} = \frac{1}{z} \cdot \sum_{r=1}^{z} \left(\epsilon_{i,r} \cdot \omega_{i,r} \cdot \xi_{r,j}\right)$$

where the suggestion $\xi_{r,j}$ is weighed by the path $\omega_{i,r}$ and the helpfulness $\epsilon_{i,r}$. In this way, suggestions from closer and more reliable users are enhanced.

The local trust measure $\tau_{i,j}$ that an agent a_i has about an agent a_j can be computed by combining reliability and local reputation (which also includes the helpfulness) as:

$$\tau_{i,j} = \alpha_i \cdot \rho_{i,j} + (1 - \alpha_i) \cdot \beta_i \cdot \sigma_{i,j}$$

where α_i and β_i are two parameters ranging in $[0,1] \in \mathbb{R}$. More specifically, the parameter α_i simply weights reliability and local reputation, while the parameter β_i for each agent a_i is computed as $\beta_i = p/\|E_i(x)\|$ where p is the number of nodes belonging to E_i. This highlights the dependability of $\sigma_{i,j}$ on the number of p nodes belonging to E_i (this because if the number of

[1] If no recommendation was provided by a_r to a_i, then the helpfulness of a_r perceived by a_i will be $\epsilon_{i,r} = 0$.

these nodes is too small then a_i will not receive a sufficient amount of information about a_j from its ego-network and the local reputation measure loses of relevance).

Note that a newcomer agent will receive suitable "cold start" values of reliability, reputation and helpfulness (see Section 6).

In this context, the local trust of a g group perceived by a_i denoted as $T_{g,i}$ is the average of all local trust measures calculated by a_i for all agents belonging to g. Similarly, the local trust of an agent a_i perceived by a group g (i.e., $T_{g,i}$), is assumed to be the average of all the local trust measures about a_i computed by all the agents belonging to g. Note that an agent can belong to many groups, and we denote by D the maximum number of groups that an agent can join with.

4.1. The voting mechanism

To take a decision about a new affiliation with a group g, we adopt a voting mechanism, which is based on the computation of a local trust defined in the previous Section. The voting process is carried out by all the agents belonging to g (i.e., each agent gives a vote $v \in \{0, 1\}$ to accept or refuse this agent into g, where 0/1 to mean "refuse"/"accept") [54].

In particular, the vote is influenced by the local trust measure that the voter computed about the potential new member, also exploiting the recommendations coming from its ego-network. Also, we introduce a suitable threshold $\Gamma_g \in [0, 1]$. The vote is equal to 0 (i.e., 1) if $\tau < \Gamma_g$ (i.e., $\tau \geq \Gamma_g$). In the following, the voting criterion v referred to a group g for a potential new member y will be assumed as the output of a function $V(g, v, y)$. In this respect, a reasonable strategy may be that of adopting a simple majority criterion to accept a requester into a group.

5. The Distributed Agent Grouping Algorithm

The proposed distributed agent grouping algorithm is described below, it consists of two procedures that are executed by:

1. each CoT agent that desires to find the "best" groups to join with, on the basis of the value of $T_{i,g}$ (where g identifies a generic group);
2. each group *administrator* that must evaluate if affiliating a new member with its administrated group based on the mutual trust existing among the group members and the potential new member.

The list of the experimental parameters adopted in the description of the algorithm is shown Table 1.

Algorithm 1 The agent a_i executes this algorithm for improving its group configuration with respect to the mutual trust with the related group members. To this aim, we define Gr that is a set of groups formed in a random way. Let $H_i \subset Gr$ be the set of the groups which a_i is affiliated to and for each group $g \in H_i \subset Gr$, contacted in the past, a_i stores its local trust measure $T_{i,g}$ and let $\hat{t}_g$ be the time elapsed from the corresponding last updating. Moreover, let D be a parameter denoting the maximum number of groups that an agent can join, let C be the maximum number of groups the generic agent can analyze, let π_i be a time threshold set by a_i and, finally, let $\theta_i \in [0, 1]$ be a threshold on the trust value between a_i and the generic group $g \in H_i$. In Section 6, we set the threshold θ_i.

Initially, the values of $T_{i,g}$ are calculated whenever the values previously stored are older than the threshold π_i (lines 1-3). Then a set of potential candidate groups P is built, with $\|P\| < D$, and ordered in decreasing manner based on the values $T_{i,g}$ of the groups, while Y is a set of groups randomly chosen and with the set $Z = Y \bigcup H$. The sets Y, Z and P might hold the groups already present into H_i, while some others might be new groups selected in a random way and inserted into the set Y. Based on the groups in P not into H_i, the agent a_i could improve the quality of its choices by joining with such groups. The two loops in lines 6-16 are the kernel of the procedure, after that $H_i = S_c$.

Algorithm 2 This algorithm is executed by the administrator a_g of a CoT group g when an agent, e.g. a_i, sends a join request to a_g. Let $K_g \subset Gr$ be the set of the agents affiliated to g, with $\|K_g\| \leq Q$ (where Q is the maximum number of agents that can be affiliated with g), let the set X be $X = K_g \bigcup a_i$, where a_i is the new, potential member of g and let μ be a time threshold set by a_g. Moreover, the administrator of a group g, i.e. a_g, stores the values of the local trust computed by all the members of its group for the agent which desires to join with, i.e. a_i, and the timestamp $\tilde{t}_i$ of its retrieval.

Initially, the administrator a_g requires to the members of its group to update their local trust measures about a_i (lines $1 - 5$), then if:

1. $||X|| < Q$ (line 6), then all the agents in g provide their vote about a_i. The function $V(\cdot)$, see Section 4, combines all the votes to determine if the agent a_i is accepted or not in g.

Table 1
Table of the main symbols

Symbol	Description	Symbol	Description
A	set of agents associated to the IoT devices	Y	set of groups randomly chosen, with $\|Y\| \leq M$
G	graph representing the agents and their relationships $G = \langle N, L \rangle$	a	agent
E_i	set of agents belonging to the ego-network of a_i, with $E_i \subseteq G$	a_g	agent administrator of the group g
Gr	set of groups formed in a random way	g	generic group
H_i	set of the groups which a_i is affiliated, with $H_i = \bigcup g_i \subseteq A$	$\tilde{t}$	time elapsed from the last execution of the procedure for an agent
K_g	set of agents affiliated with a group g	$\hat{t}$	time elapsed from the last execution of the procedure for a group
D	maximum number of new groups the single agent can analyze	θ	threshold on the level of trust between an agent and a generic group
C	maximum number of groups that an agent can join	μ	time threshold fixed by the agent administrator of a group
Q	maximum number of agents belonging to a group	π	time threshold fixed by an agent
P	set of candidate groups	τ	trust
$V(\cdot)$	voting function	T	trust about a group

Algorithm 1 The procedure executed by an agent.

Input: $H_i \subset Gr, D, \pi_i, \theta_i; Y = \{g \in G\}$ a set of groups randomly selected : $\|Y\| = C \leq D$, $H_i \bigcap Y = \{\ \}$, $Z = (H_i \bigcup Y)$

1: **for** $g \in Z : \hat{t}_g > \pi_i$ **do**
2: Compute $T_{i,g}$ by exploiting the agents $\in g$.
3: **end for**
4: $m \leftarrow 0$
5: Let be $P = \{g \in Z : T_{i,g} \geq \theta_i\}$, with $\|P\| = C$
6: **for all** $g \in P : g \notin H_i$ **do**
7: send a join request to the agent administrator of g
8: **if** g accepts the request **then** $m \leftarrow m + 1$
9: **end if**
10: **end for**
11: **for all** $g \in H_i : g \notin P$ **do**
12: Sends a leave message to g
13: $m \leftarrow m - 1$
14: **if** (m==0) **then break**
15: **end if**
16: **end for**

2. $||X|| = Q$ and the agent a_i is accepted into the group but in place of another agent. To make comparable agents can be used the measure of the trust that the group has about them, which is computed as explained in Section 4 (line 16). In particular, $T_{g,n}$ denotes the current value of trust between the group g and the agent $k_n \in X \bigcup \{a_i\}$.

Lines $6 - 11$ deal with the first scenario, while lines $12 - 18$ with the second one of Algorithm 2.

Algorithm 2 The procedure executed by a group administrator.

Input: $K_g, Q, a_i, \mu, X = K_g \bigcup \{a_i\}$;

1: **for all** $k \in K_g$ **do**
2: **if** $\tilde{t}_i \geq \mu$ **then** ask to k for updating local trust values of a_i
3: **end if**
4: **end for**
5: **if** $\|X\| < Q$ **then**
6: **if** $V(g, v, a_i) == 1$ **then** Send an accept message to a_i
7: **else** Send a reject message to a_i
8: **end if**
9: **else**
10: **for all** $k \in X$ **do** compute τ_{k,a_i}
11: **end for**
12: Let $X' = \{k_1, k_2, \ldots, k_{\|K_g\|+1}\}$ with $k_i \in X \bigcup \{a_i\}$, ordered by trust with $T_{g,m} \geq T_{g,n}$ iff $m < n$
13: **if** $X[\|K_g\| + 1] == a_i$ **then** Send a reject message to a_i
14: **else**
15: Send a leave message to the node $X[\|K_g\| + 1]$
16: Send an accept message to a_i
17: **end if**
18: **end if**

6. Experiments

In this section we present and discuss the results of a few experiments aimed at verifying the effectiveness of the approach. The experiment were performed by means of an ad hoc simulator written in the scientific programming language Octave [55].

More in detail, the ability of our algorithm to form groups denoted by a higher, in average, mutual trust among their members with respect to different compositions has been tested. The list of the parameters adopted in the experiments is reported in Table 2. For convenience, we will refer to each parameter with a parameter ID, reported in the first column of the same table.

We simulated a network of 1000 different CoT agents (each one associated with an IoT device), 1000

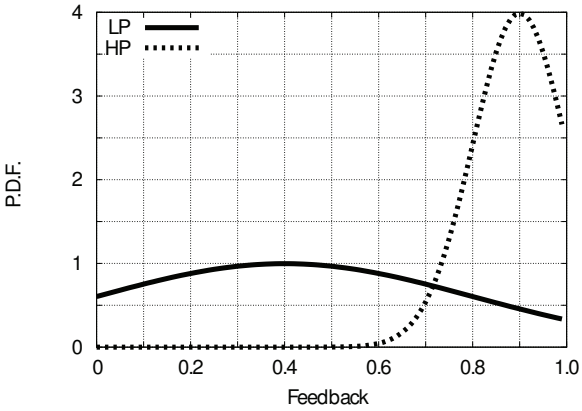

Fig. 2. a) Generated feedback values.

initial trust relationships and $|Gr|$ groups formed in a random way (par. ID 9). Trust values were set by a normal distribution and two different profiles: "low performance" (par. ID 3) and "high performance" (par. ID 4). In this way we could model two different users profiles; in the first case user show, in average, low performances as the feedback for services assume values around the value 0.2 with a given standard deviation, while in the second case ("high performance") the mean is 0.9. The values of feedback that have been generated based on the basis of the two normal distributions are depicted in see Figure 2. Moreover the trusted/distrusted agents ratio (par ID 6) has been set to 0.5 – the same value was set for the cold start trust value – and the initial sparsity of the trust network decreased along the simulation thanks to the availability of new reliability information.

The interactions among agents have been simulated by means of the Poisson distribution with expected value $\lambda = 50$ (par ID 2). The remaining simulation parameters are listed in the third part of Table 2, (par ID 7 – 12). In particular, as we discuss later in this section, we consider a variable parameter D (par ID 8), while the remaining parameters have been fixed on the basis of a preliminary sensitivity analysis aimed at obtaining a minimum benefit from the algorithm execution. For instance, the selected value of $\theta = 0.2$ represents the minimum value for the algorithm to avoid a trivial selection of all the groups as candidates.

More in detail, for each simulation iteration:

1. a number of agents mutual interactions is simulated; this number is not fixed it is generated on the bases of the Poisson distribution, as indicated in Table 2 (par ID 2). An interaction is simulated by registering a feedback given by an agent.
2. 100 execution of the algorithm have been simulated by triggering the Algorithm 1 on 100 different agents randomly chosen and each time an agent affiliation request to a group is simulated, the administrator-side of the Algorithm 2 is executed to decide whether or not admitting the requiring agent into its group;
3. some statistics are computed.

To evaluate the results, we define the measure *Average Mutual Trust* among the components of a group g as:

$$AMT_g = \frac{1}{2\|g\|} \sum_{\substack{i,j=1 \\ i \neq j}}^{\|g\|} (\tau_{i,j} + \tau_{j,i})$$

and the *Mean Average Mutual Trust*, with respect to a certain configuration at a certain time-step, as:

$$MAMT(Gr) = \frac{1}{\|Gr\|} \sum_{i=1}^{\|Gr\|} AMT_{g_i}$$

.

Figure 3 shows the median value of MAMT measured after each single iteration of the simulation for the different values of $D = [5 \div 10]$ for the first 30 iterations of the simulation. For $D = 5$ can be observed a slow convergence of the MAMT values, while for $D \geq 6$ there exist a radical change. In fact, D is the number of new groups the can be analyzed by a_i at each iteration of the Algorithm 1 execution, which are then mixed in the new set P with groups already into the set H_i. Therefore, we deduce the higher D, the higher the number of new groups analyzed in the Algorithm 1, the higher the probability to join with a new group where are present distrusted agents and replacing those having the worst trust value (by increasing in this way the MAMT value because, sooner or later, distrusted agents must leave groups). Moreover, in Figure 4 it is shown the presence into the all groups of distrusted agents at different simulation iterations per different values of D. Results confirmed that almost distrusted agents are replaced by trusted agents into the groups.

Therefore, the execution of the distributed algorithm allows a configuration of groups with a high level of (average) mutual trust among its members to be reached. More specifically, in a simulated environment, the convergence of the algorithm towards a

Table 2
Experiment Setting

Par. ID	**Parameter**	**Value**
–	*General*	
1	No. of Agents ($\|\|A\|\|$)	1000
2	No. of Feedback per iteration (Poisson distrib.)	$\lambda = 50$
	Agents Performance (Local Trust)	
3	Low Performance (Normal Distribution)	$mean = 0.9$; $stdDev = 0.1$
4	High Performance (Normal Distribution)	$mean = 0.2$; $stdDev = 0.1$
5	Cold start value of trust	0.5
6	Ratio of reliable/unreliable agents	0.5
	Group formation	
7	Q (Max no. of agents per group)	20
8	D (Max no. of groups an agent analyzes)	$\{5, 10, 15, 20\}$
9	$\|\|Gr\|\|$ (No. of groups)	50
10	l_{Max} (Maximum recommender distance)	{1,2}
11	θ (Minimum value of trust for a group to be selected as candidate for group formation)	0.2
12	Γ_g (threshold for trust-based voting)	0.5

group configuration with trusted agents is reached very rapidly (when the algorithm parameters are properly set) by improving the group composition very quickly. Conversely, a different choice, e.g. a lower D, will allow a few untrusted agents to be affiliated to some groups, so they can get benefit from the services of the trusted agents present within the group.

7. Conclusions and Future Development

In this paper, a CoT scenario supporting the IoT devices virtualization over the Cloud Computing in a multi-agent context has been presented. The social attitude of software agents to cooperate has been exploited to form groups for promoting satisfactory agents interactions which tightly depends on the choice of the partner. However, in absence of suitable information to perform a good choice, some suggestions can be asked

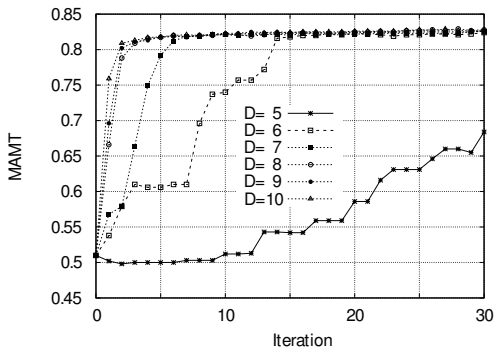

Fig. 3. MAMT - results until 30 Iterations

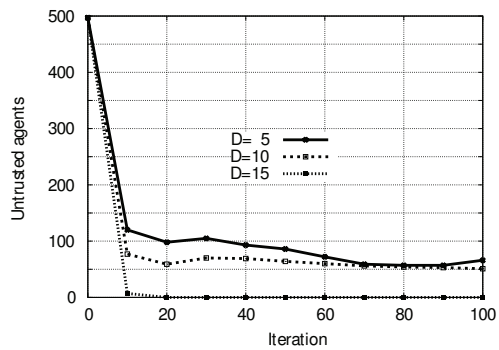

Fig. 4. Sum of untrusted agents vs number of iterations of the simulation

to those agents perceived as the mostly trustworthy in the community.

To promote the formation of agent groups of reliable recommenders we designed a distributed algorithm that, in a competitive and cooperative scenario, adopts a voting procedure based on the agent capability to provide useful recommendation exploiting local trust and helpfulness measures. In particular, the adoption of *local trust* measures avoids heavy computational tasks and communication overheads because only a little share of the agent community is involved in this process. Some experiments, in a simulated agent CoT scenario, confirmed the potential advantages given by our proposal to improve individual and group satisfaction in terms of mutual trust.

In our ongoing researches, we are studying to improve the effectiveness of the group formation process by adopting in the voting procedure a new measure (denoted ζ) combining the local trust measure with an

utility measure of the devices (i.e., agent) for the group itself.

In particular, the utility measure should take into account all those cases where the presence of agents provided of some skills could give significant benefits to the groups. Therefore, a group might decide the affiliation of these agents regardless of their trust values. In other words, let $S_g = \{k_1, k_2, \cdots, k_n\}$ be the n skills needed to a group and let m be the skills present in a group, with $m < n$; if this group is interested to maximize its effectiveness, it could be interested in accepting those agents having one or more of the $n-m$ skills for which the group is lacking.

More in detail, we are supposing that the contribution given by the utility term (i.e., υ) should increase with respect to the number of skills owned by the agent a_i and absent in the group g. More formally, $\upsilon_{g,i} = f(S_g, S_i)$, where $\upsilon_{i,g}$ is the utility of a_i for g and $f(\cdot)$ is a function returning a value (ranging in $[0, 1]$) and receiving in input the skills owned by a_i (i.e., S_i) and needed to the group g (i.e., S_g), respectively.

$\zeta_{g,i} = \lambda_g \cdot \upsilon_{g,i} + (1 - \lambda_g) \cdot \tau_{g,i}$ is a new measure, where ζ_g is a real value belonging to $[0, 1]$ and λ is a parameter (a real value ranging in $[0, 1]$) to weight the relevance assigned by the administrator of g to the utility with respect to the local trust (taking into account reliability, local reputation and helpfulness) of a_i. In particular, $\lambda = 1$ denotes that the administrator accepts to risk inserting in its group agents having a higher percentage of skills regardless of their values of trust; conversely, $\lambda_g = 0$ indicates that the administrator of g only considers the local trust of the agents.

Acknowledgment

This work has been partially supported by the Networks and Complex Systems (NeCS) Laboratory - Department of Engineering Civil, Energy, Environment and Materials (DICEAM) - University Mediterranea of Reggio Calabria, by the Italian MIUR, PRIN 2017 Project "Fluidware" (CUP H24I17000070001), and by the University of Catania, Piano per la Ricerca 2016-2018 - Linea di intervento 1 (Chance), prot. 2019-UNCTCLE-0343614.

References

[1] Giancarlo Fortino, Lidia Fotia, Fabrizio Messina, Domenico Rosaci, and Giuseppe M. L. Sarné. Supporting agent CoT groups formation by trust. In *Proceedings of the 20th Workshop "from Objects to Agents", WOA 2019*, volume 2024 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2019.

[2] Mohammad Aazam, Imran Khan, Aymen Abdullah Alsaffar, and Eui-Nam Huh. Cloud of things: Integrating internet of things and cloud computing and the issues involved. In *Applied Sciences and Technology (IBCAST), 2014 11th International Bhurban Conference on*, pages 414–419. IEEE, 2014.

[3] P. Parwekar. From internet of things towards cloud of things. In *2011 2nd International Conference on Computer and Communication Technology (ICCCT-2011)*, pages 329–333, Sept 2011. doi: 10.1109/ICCCT.2011.6075156.

[4] Alessio Botta, Walter De Donato, Valerio Persico, and Antonio Pescapé. Integration of cloud computing and internet of things: a survey. *Future Generation Computer Systems*, 56:684–700, 2016.

[5] G. Fortino, W. Russo, C. Savaglio, W. Shen, and M. Zhou. Agent-oriented cooperative smart objects: From iot system design to implementation. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 48(11):1939–1956, 2018.

[6] Gianluca Aloi, Giuseppe Caliciuri, Giancarlo Fortino, Raffaele Gravina, P Pace, Wilma Russo, and Claudio Savaglio. Enabling iot interoperability through opportunistic smartphone-based mobile gateways. *Journal of Network and Computer Applications*, 81:74–84, 2017.

[7] Dieter Uckelmann, Mark Harrison, and Florian Michahelles. An architectural approach towards the future internet of things. In *Architecting the internet of things*, pages 1–24. Springer, 2011.

[8] Pasquale De Meo, Fabrizio Messina, Maria Nadia Postorino, Domenico Rosaci, and Giuseppe M. L. Sarné. A reputation framework to share resources into iot-based environments. In *2017 IEEE 14th International Conference on Networking, Sensing and Control (ICNSC)*, pages 513–518. IEEE, 2017.

[9] G Fortino, F Messina, D Rosaci, and Giuseppe M. L. Sarné. Using trust and local reputation for group formation in the cloud of things. *Future Generation Computer Systems*, 89: 804–815, 2018.

[10] Giancarlo Fortino, Wilma Russo, Claudio Savaglio, Weiming Shen, and Mengchu Zhou. Agent-oriented cooperative smart objects: From iot system design to implementation. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 48 (11):1939–1956, 2017.

[11] James Doodson, Jeff Gavin, and Richard Joiner. Getting acquainted with groups and individuals: Information seeking, social uncertainty and social network sites. In *ICWSM*, 2013.

[12] Chao-Min Chiu, Meng-Hsiang Hsu, and Eric TG Wang. Understanding knowledge sharing in virtual communities: An integration of social capital and social cognitive theories. *Decision support systems*, 42(3):1872–1888, 2006.

[13] A. Comi, L. Fotia, F. Messina, G. Pappalardo, D. Rosaci, and G. M. L. Sarné. Forming homogeneous classes for e-learning in a social network scenario. In *IDC IX*, pages 131–141. Springer, 2016.

[14] Antonello Comi, Lidia Fotia, Fabrizio Messina, Domenico Rosaci, and Giuseppe ML Sarné. Grouptrust: Finding trust-based group structures in social communities. In *International Symposium on Intelligent and Distributed Computing*, pages 143–152. Springer, 2016.

[15] Anita Blanchard and Tom Horan. Virtual communities and social capital. In *Knowledge and social capital*, pages 159–

178. Elsevier, 2000.

[16] Pasquale De Meo, Lidia Fotia, Fabrizio Messina, Domenico Rosaci, and Giuseppe ML Sarné. Providing recommendations in social networks by integrating local and global reputation. *Information Systems*, 78:58–67, 2018.

[17] Fabrizio Messina, Giuseppe Pappalardo, Antonello Comi, Lidia Fotia, Domenico Rosaci, and Giuseppe ML Sarné. Combining reputation and qos measures to improve cloud service composition. *International Journal of Grid and Utility Computing*, 8(2):142–151, 2017.

[18] PeiYun Zhang, MengChu Zhou, and Giancarlo Fortino. Security and trust issues in fog computing: A survey. *Future Generation Computer Systems*, 88:16–27, 2018.

[19] P. De Meo, F. Messina, D. Rosaci, and Giuseppe M. L. Sarné. Recommending users in social networks by integrating local and global reputation. In *Proc. of the 7th Int. Conf. on Internet and Distributed Information Systems*, volume 8729 of *LNCS*, pages 437–446. Springer, 2014.

[20] Luigi Palopoli, Domenico Rosaci, and Giuseppe M. L. Sarné. A multi-tiered recommender system architecture for supporting e-commerce. In *Intelligent Distributed Computing VI*, pages 71–81. Springer, 2013.

[21] D Marc Kilgour and Colin Eden. *Handbook of group decision and negotiation*, volume 4. Springer Science & Business Media, 2010.

[22] G. Lax and G. M. L. Sarné. CellTrust: a reputation model for C2C commerce. *Electronic Commerce Research*, 8(4):193–216, 2006.

[23] P Raghu Vamsi and Krishna Kant. Systematic design of trust management systems for wireless sensor networks: A review. In *Advanced Computing & Communication Technologies (ACCT), 2014 4th Int. Conf. on*, pages 208–215. IEEE, 2014.

[24] Giancarlo Fortino and Paolo Trunfio. *Internet of things based on smart objects: Technology, middleware and applications*. Springer, 2014.

[25] PeiYun Zhang, MengChu Zhou, and Giancarlo Fortino. Security and trust issues in fog computing: A survey. *Future Generation Computer Systems*, 88:16 – 27, 2018. ISSN 0167-739X. doi: https://doi.org/10.1016/j.future.2018.05.008. URL `http://www.sciencedirect.com/science/article/pii/S0167739X17329722`.

[26] C. Dellarocas. Designing reputation systems for the social web. *SSRN Electronic Journal*, 2010.

[27] T.D. Huynh, N.R. Jennings, and N.R. Shadbolt. An integrated trust and reputation model for open multi-agent systems. *Autonomous Agents and Multi-Agent Systems*, 13(2):119–154, 2006.

[28] Y. Kim and H.S. Song. Strategies for predicting local trust based on trust propagation in social networks. *Knowledge-Based Systems*, 24(8):1360–1371, 2011.

[29] Pasquale De Meo, Fabrizio Messina, Domenico Rosaci, and Giuseppe M. L. Sarné. Forming time-stable homogeneous groups into online social networks. *Information Sciences*, 414: 117–132, 2017.

[30] P. Massa and P. Avesani. Trust metrics on controversial users: Balancing between tyranny of the majority. *International Journal on Semantic Web and Information Systems (IJSWIS)*, 3(1): 39–64, 2007.

[31] Antonello Comi, Lidia Fotia, Fabrizio Messina, Giuseppe Pappalardo, Domenico Rosaci, and Giuseppe ML Sarné. Using semantic negotiation for ontology enrichment in e-learning multi-agent systems. In *2015 Ninth International Conference on Complex, Intelligent, and Software Intensive Systems*, pages 474–479. IEEE, 2015.

[32] C.N. Ziegler and G. Lausen. Spreading activation models for trust propagation. In *e-Technology, e-Commerce and e-Service, EEE'04. 2004 IEEE Int. Conf. on*, pages 83–97. IEEE, 2004.

[33] J. Golbeck and J.A. Hendler. Inferring binary trust relationships in web-based social networks. *ACM Transactions on Internet Technology*, 6(4):497–529, 2006.

[34] R. Guha, R. Kumar, P. Raghavan, and A. Tomkins. Propagation of trust and distrust. In *Proc. of the 13th International Conference on World Wide Web*, pages 403–412. ACM, 2004.

[35] J.A. Golbeck. Computing and applying trust in web-based social networks. In *PhD Thesis*. University of Maryland, Department of Computer Science, 2005.

[36] P. Massa and P. Avesani. Trust-aware recommender systems. In *Proc. of the 2007 ACM Conference on Recommender systems*, pages 17–24. ACM, 2007.

[37] Steven J Brams and Peter C Fishburn. Voting procedures. *Handbook of social choice and welfare*, 1:173–236, 2002.

[38] Thomas C Beierle and Jerry Cayford. *Democracy in practice: Public participation in environmental decisions*. Resources for the Future, 2002.

[39] Jeremy Pitt, Lloyd Kamara, Marek Sergot, and Alexander Artikis. Formalization of a voting protocol for virtual organizations. In *Proc. of the 4th Int joint Conf. on Autonomous Agents and Multiagent Systems*, pages 373–380. ACM, 2005.

[40] Yann Chevaleyre, Ulle Endriss, Jérôme Lang, and Nicolas Maudet. A short introduction to computational social choice. *SOFSEM 2007: Theory and Practice of Computer Science*, pages 51–69, 2007.

[41] Margaret Levi. A state of trust. *Trust and governance*, 1:77–101, 1998.

[42] Paul Dumouchel. Trust as an action. *European J. of Sociology/Archives Européennes de Sociologie*, 46(3):417–428, 2005.

[43] Tao Jiang and John S Baras. Trust evaluation in anarchy: A case study on autonomous networks. In *INFOCOM*, 2006.

[44] James Caverlee, Ling Liu, and Steve Webb. The socialtrust framework for trusted social information management: Architecture and algorithms. *Information Sciences*, 180(1):95–112, 2010.

[45] Fenye Bao and Ing-Ray Chen. Dynamic trust management for internet of things applications. In *Proc. of the 2012 int. work. on Self-aware internet of things*, pages 1–6. ACM, 2012.

[46] Michele Nitti, Roberto Girau, Luigi Atzori, Antonio Iera, and Giacomo Morabito. A subjective model for trustworthiness evaluation in the social internet of things. In *Personal Indoor and Mobile Radio Communications (PIMRC), 2012 IEEE 23rd International Symposium on*, pages 18–23. IEEE, 2012.

[47] Ray Chen, Fenye Bao, and Jia Guo. Trust-based service management for social internet of things systems. *IEEE trans. on dependable and secure computing*, 13(6):684–696, 2016.

[48] M. N. Postorino and G. M. L. Sarné. An agent-based sensor grid to monitor urban traffic. In *Proceedings of the 15th Workshop dagli Oggetti agli Agenti, WOA 2014*, volume 1260 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2014.

[49] Yosra Ben Saied, Alexis Olivereau, Djamal Zeghlache, and Maryline Laurent. Trust management system design for the internet of things: A context-aware and multi-service approach. *Computers & Security*, 39:351–365, 2013.

[50] Carlo Vallati, Enzo Mingozzi, Giacomo Tanganelli, Novella Buonaccorsi, Nicola Valdambrini, Nikolaos Zonidis, Belén Martínez, Alessandro Mamelli, Davide Sommacampagna, Bayu Anggorojati, et al. Betaas: A platform for development and execution of machine-to-machine applications in the internet of things. *Wireless Personal Comm.*, 87(3):1071–1091, 2016.

[51] Sheikh Mahbub Habib, Sebastian Ries, and Max Muhlhauser. Towards a trust management system for cloud computing. In *Trust, Security and Privacy in Computing and Communications (TrustCom), 2011 IEEE 10th International Conference on*, pages 933–939. IEEE, 2011.

[52] Soon-Keow Chong, Jemal Abawajy, Masitah Ahmad, and Isredza Rahmi A Hamid. Enhancing trust management in cloud environment. *Procedia-Social and Behavioral Sciences*, 129:314–321, 2014.

[53] Fabrizio Messina, Giuseppe Pappalardo, Domenico Rosaci, Corrado Santoro, and Giuseppe M. L. Sarné. A trust-aware, self-organizing system for large-scale federations of utility computing infrastructures. *Future Generation Computer Systems*, 56:77–94, 2016.

[54] Linda SL Lai and Efraim Turban. Groups formation and operations in the web 2.0 environment and social networks. *Group Decision and negotiation*, 17(5):387–402, 2008.

[55] John Wesley Eaton, David Bateman, and Søren Hauberg. Gnu octave, 1997.