To appear in *Enterprise Information Systems*
Vol. 00, No. 00, Month 20XX, 1–20

Research Paper

Trust Models for Efficient Communication in Mobile Cloud Computing and their Applications to e-Commerce

Florin Pop[a], Ciprian Dobre[a*], Bogdan-Costel Mocanu[a], Oana-Maria Citoteanu[a], and Fatos Xhafa[b]

[a] *Computer Science Department, University Politehnica of Bucharest, Romania*;
[b] *Universitat Politècnica de Catalunya, Barcelona, Spain*

(*v4.0 released February 2015*)

Managing the large dimensions of data processed in distributed systems that are formed by datacenters and mobile devices has become a challenging issue with an important impact on the end-user. Therefore, the management process of such systems can be achieved efficiently by using uniform overlay networks, interconnected through secure and efficient routing protocols. The aim of this paper is to advance our previous work with a novel trust model based on a reputation metric that actively uses the social links between users and the model of interaction between them. We present and evaluate an adaptive model for the trust management in structured overlay networks, based on a Mobile Cloud architecture and considering a honeycomb overlay. Such a model can be useful for supporting advanced mobile market-share e-Commerce platforms, where users collaborate and exchange reliable information about, for example, products of interest and supporting ad-hoc business campaigns

Keywords: Overlay Networks; Trust Management; Reputation; Mobile Cloud; Peer-to-Peer Systems; Resource Management; e-Commerce.

1. Introduction

Recent years have shown an increasing interest by researchers in smart devices and technologies such as smartphones and tablets. Such devices incorporate ever more advanced communication, sensing and processing capabilities, thus creating new opportunities for businesses. Clients have access to information and are connected everywhere anytime with their businesses, many times through the Cloud as the infrastructure facilitating this ubiquitous connectivity. Thus, it is no wonder that, in recent years, we have witnessed increased research on new models to facilitate the business interactions made possible by mobile and Cloud platforms. Among this, Mobile Cloud Computing (MCC) can create interesting new opportunities for extending the connectivity of any single portable device, not only with the Cloud but also to the social layer made of other similar devices being carried around by "friends" (Samad, Loke, and Reed 2015). The Cloud enables access to information regarding a certain business, but the new advertisement models today are based on recommendations being made by other sibling devices. Recommending a certain product today is the result of a complex process involving trust and information being exchanged between people over social media channels.

MCC extends the cloud computing model with the mobility aspect offered by the

*Corresponding author. Email: ciprian.dobre@cs.pub.ro

mobile devices. Such a paradigm supports novel market techniques to bring more reliable information to the users in a faster way and is more reliable to the users, as is the case of mobile e-Commerce campaigns, novel advertising techniques and others (Zhang et al. 2014). As the number of mobile devices, apps and services is increasing exponentially, new businesses have arrived (Li et al. 2013). The application markets offer the end users services in Mobile Commerce, Mobile Learning, Mobile Healthcare and Mobile Gaming (De and De 2013). Another applicability of MCC is that it can cope with large-scale data-sets, where meeting the new features of data transmission among datacenters represents the main research challenge (Song et al. 2015).

For such new models of business interactions facilitated by MCC, we are interested in the concept of a mobile Peer-to-Peer network, a group of devices which is able to communicate directly with each other in a decentralized manner. The communication in this case is facilitated by wireless communication technologies, such as IEEE 802.11, Bluetooth, Ultra-Wide Band, or Wi-Fi Direct (Pyattaev et al. 2013).

Structured in an organized manner like an auto-adaptive overlay network for MCC, we previously proposed an adaptive Peer-to-Peer business model based on a bio-inspired topology, *the honeycomb*. This model has been chosen because it offers a fixed structure that can scale exceptionally well and support a large number of users and because it is auto-adaptive. Moreover, we would like to find a way to form the honeycomb structure and to study how well it behaves when content is distributed (Chen et al. 2013). Another objective is to find a way of computing and storing the trust values of active users in a network and to see how these values evolve according to the behavior of the users that join a mobile network.

The main subject of interest in MCC is the interaction between real mobile devices and the Cloud. In (Barbera et al. 2013), a feasibility study for mobile computation offloading and for on the fly mobile data and software back-up is presented. The proposed architecture of this paper is based on the existence of a pair consisting of a real mobile device and an associated clone in the cloud. The main observations highlighted in this paper were the facts that, in 50% of the cases, mobile devices are connected to Wi-Fi networks and that the average time for using mobile devices in an area with no Wi-Fi coverage is approximately 2 hours.

It would be worth considering the early experiments for human mobility coupled with small- and medium-range technologies, such as Bluetooth, Wi-Fi, and ZigBee. This approach is feasible only for close proximity network devices. Alternately, by considering environment and energy usage patterns, a scheme for energy usage offloading reduction in mobile cloud networks was presented in (Papanikolaou et al. 2014). The fundamental idea of these papers is to present how users might increase their reliability by taking advantage of opportunistic mobile clouds. The proposed schema was validated through simulation.

e-Commerce creates new services and facilities for all types of users, and now we are faced with mobile behavior, so special types of services are required. In (Opara and Gupta 2015), a study on the impact of web service opportunity, which offers a platform for integrated applications in a standard manner rather than in a proprietary way, is presented. The migration of an entity from the standard system to a real-time platform based on peer-to-peer systems is the most important feature of Web services utilization. In e-Commerce, reputation, which is a proper metric for trust, is one of the most important factors for service selection.

Even though the trust metric's definition has not yet been standardized, there are various researchers who have come up with a definition. Therefore, in (Mekouar, Iraqi, and Boutaba 2010), trust is defined as "the belief the trusting agent has in the trusted agent's willingness and capability to deliver a mutually agreed service in a given context and in

a given time slot." In the same paper, the authors proposed a definition for reputation as well. Thus, according to their opinion, reputation is defined as "an aggregation of the recommendations from all of the third-party recommendations agents and their first, second and third hand opinions as well as the trustworthiness of the recommendation agent in giving correct recommendations to the trusting agent about the quality of the trusted agent."

One model for trust management is presented in (Yu et al. 2014) based on a recommendation model entitled MeTrust. In this system, the recommendation criteria is based on an analytic hierarchy process. Additionally, an algorithm for trust calculation that was validated in simulation is presented in this paper. The results presented emphasize the fact that the proposed trust model can identify malicious peers with small overhead.

The research group from Distributed Systems in University Politehnica of Bucharest developed a platform for feasibility evaluation of resource sharing for collocated smartphones called HYCCUPS (Marin and Dobre 2013). The main purpose of this platform is to minimize the power consumption of the interconnected devices by taking both the availability and mobility of nodes into consideration. Beginning with the positive results, the group experimented with further models where the decentralized interactions between users carrying mobile devices become support for novel apps and business opportunities (Papanikolaou et al. 2014), (Ciobanu et al. 2015). The current work being presented in this paper is another step in this direction.

The main contributions of this paper can be summarized as follows:

- we have extended our previous research, which is related to a large-scale honeycomb overlay (Pop et al. 2014) with fault tolerance scenarios;
- we present a model for new resource discovery and request based on trust and introduce an algorithm that establishes 8 managers for one peer in the overlay. In this model, each peer can be a manager for other peers;
- we have adapted the computation of trust value for nodes proposed in our previous research (Vişan, Pop, and Cristea 2011) for the honeycomb overlay and introduce a new model to compute the links' trust value;
- we present and evaluate an adaptive trust model for the nodes and links trust management in structured overlay networks (honeycomb) based on a Mobile Cloud architecture. Such a model can prove useful for supporting advanced mobile market-share e-Commerce platforms where users collaborate and exchange trustful information about, for example, products of interest, supporting ad-hoc business campaigns.

The paper has the following structure. Section 2 presents several related works. In Section 3, we provide a brief description of the used honeycomb overlay and its properties. Section 4 describes the adaptive trust model adapted for the honeycomb overlay. Section 5 presents the simulation and the experimental results. Finally, Section 6 highlights the conclusions drawn by this paper.

2. Related Work

A good overview regarding reputation in Peer-to-Peer systems has been created in (Marti and Garcia-Molina 2006). The authors present a useful taxonomy of Peer-to-Peer reputation models composed by the gathering of information and ranking and rewarding the peers. This paper motivates the adaptive behavior of our proposed trust model. Having a good trust metric based on Social-Networks is a major challenge because the reputation ranking depends on the peers' social position. Therefore, the most dangerous peers are the front ones. In (Tian and Yang 2011), the authors proposed a better approach entitled

Poisonwater for Social-Network-based trust metrics that is more resistant to front peer attack than EigenTrust and PowerTrust. In comparison, the Poisonwater can mitigate front peer attacks by 20%. Another interesting approach regarding building good trust relationships in decentralized Peer-to-Peer systems is presented in (Wang and Nakao 2010), where the authors proposed a new model of trust based on reputation and risk evaluation. This model is suitable for defending against simple malicious attacks, collusive attacks and strategic attacks. The applicability of reputation was studied in (Şerbanescu et al. 2012) and (Achim, Pop, and Cristea 2011), where a reputation-based selection mechanism for a web service replica is proposed.

Reputation management in Peer-to-Peer networks based on Distributed Hash Tables (DHT) is the main idea presented by (Fedotova and Veltri 2009). The authors proposed an algorithm that permits the peers in the network to obtain individual ranking values for each node based on knowledge exchange.

Due to the sharp increase of mobile devices in the market, mobile Peer-to-Peer networks can be considered an important category of decentralized systems. Creating trust and managing reputation in this type of network is a challenge. Therefore, in (Qureshi, Min, and Kouvatsos 2012), a brief overview of the four trust management schemes for Mobile Peer-to-Peer systems is realized, and a new reputation trust management schema entitled M-trust is further proposed. Concerning the hybrid Peer-to-Peer networks, (Tian et al. 2014) suggests a SuperPeer trust model called SuperTrust to determine the peers to cooperate based on common interests. Therefore, peers that share less things in common with others are likely to interact with others more frequently. This model has a higher success rate for withstanding several attacks, such as simple malicious, denigrating peers, collusive peers and strategic peers attacks.

Trust is a very complex measurement value because it can be computed locally or globally based on the past interactions of the node or a single transaction. Additionally, it is dynamic and asymmetric because the trusted value for a node is computed based on the previous interaction of that node with other peers. The dynamic behavior of the trust value is based on the fact that the trust value for every node can rapidly decrease for instance, if the node starts to share malicious date in the system. In this case, a solution to establish a proper bound is to use entropy computed with interaction probabilities (Ţăpuş and Popescu 2012).

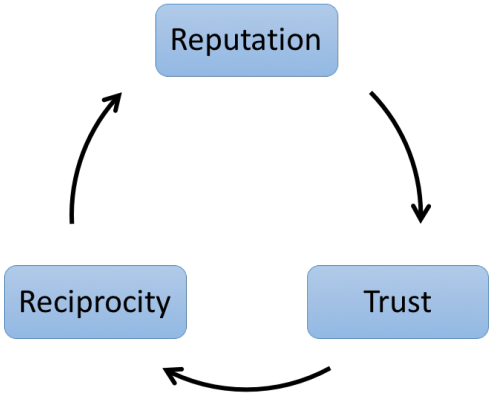

Figure 1. Relationship model of Trust.

Another property for trust is *transitivity*. Let us consider three nodes in the system, X, Y and Z, and suppose that X trusts Y and that Y trusts Z. This means that X also trusts Z, but the trust value for this relation is computed taking in consideration both the trust for Y and Y's trust for Z.

In (Ding, Yueguo, and Weiwei 2004), a computation model is presented that involves the interactions of a node with another node in a specific context (see Figure 1). Reputation is defined as the perception formed through past actions about the intentions and norms of a peer. Trust is defined as a subjective expectation that a peer has towards another based on the previous behavior and history. The flow of the system is as follows. An exchange takes place based on the trust value. After it is completed, the reputation is updated, the trust value is recomputed, and so on. The computation process for reputation may be affected by false recommendations.

A new trust model (Jiang et al. 2012), which is an extension of the Dempster-Shafer theory, is presented in (Shafer et al. 1976). The model improves the filtering of false recommendations and the dynamic adaptivity to strategic behaviors.

A description of how trust is computed and used within some of the most popular e-commerce systems is given below.

- **eBay**. When joining eBay, users are required to provide a valid email address and are allowed to choose any pseudonym that they want. Because their real identity is not checked when they create an email account, they can remain anonymous. Giving feedback is optional. In the beginning, any user could give feedback about any other user without directly interacting with him (Resnick and Zeckhauser 2002). Now, users can give ratings only on transactions that take place. There are two types of feedback. A general one, where the ratings are "positive," "negative" or "neutral," and a more detailed one via which a buyer can give comments about the seller (called DSR, or detailed seller rating). DSRs are anonymous and can be given as soon as the transaction takes place, or up to 60 days after. They consist of rating more aspects of the transaction by choosing the number of stars for each criterion. One star is the lowest score, while five stars is the highest ranking. A user's rating is not shown until there are at least 10 feedbacks. Average ratings are computed over a period of 12 months. Only one transaction between two users is considered within a week interval for the overall Feedback score.
- **Amazon**. As in the case of eBay, registering with Amazon's service requires an email address. The users can give feedback on the sellers and on the available books. Each can be rated from one star (worst review) to 5 (best review). The trust is computed as the average score of the feedback (Silaghi, Arenas, and Silva 2007). Unlike eBay, only the sellers are rated. They cannot give feedback concerning the buyers. The users similarly have 90 days to rate or comment on the transactions. An unlimited number of accounts can be created under the same email address, but with different passwords.

3. Auto-Adaptive Honeycomb Overlay Network

Peer-to-Peer systems are characterized by availability, massive scalability, robustness and flexibility. They are highly adaptive, capable of self-organization and can offer load balancing and fault tolerance. Some of their features are: efficient data search, redundant storage, hierarchical naming, anonymity, trust and authentication. Because the nodes inside a Peer-to-Peer system can join the system and leave at any moment without announcing their exit, the structure on which the system is built has to be able to respond in a quick and efficient manner to these changes (Pop et al. 2014).

The number of neighbors a node has differs for each structure. For example, in a ring, a node has only two neighbors; in a honeycomb, it has three; in a square mesh, four; and in a hexagonal mesh, six. Although a larger number of neighbors would seem a better choice because a node has direct contact with more neighbors and thus the probability

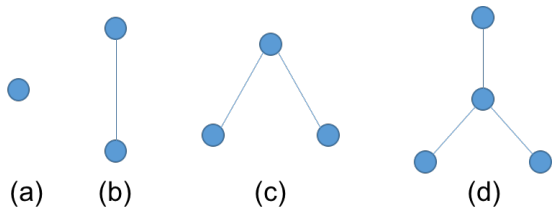

Figure 2. Nodes connection types in a Honeycomb Structure during the construction phase.

of finding the information it needs in a quicker manner (by traversing as few other nodes as possible) is higher, it can actually reduce the efficiency of the system. This happens because of the dynamics of the network. If nodes often change their availability, the structure needs to be reorganized. The smaller the number of nodes that are influenced, the less time is required to rebuild the system. In a honeycomb construction phase, a node may have from 0 to 3 neighbors (see Figure 2 (a)-(d)).

The honeycomb structure has been chosen because each peer has a maximum of three direct neighbors. The nodes can be placed so that they form chains that are connected through links. This means that a peer has two neighbors on its chain and a third one on an adjacent chain. Depending on its position, a peer can be linked to an exterior chain or an interior one.

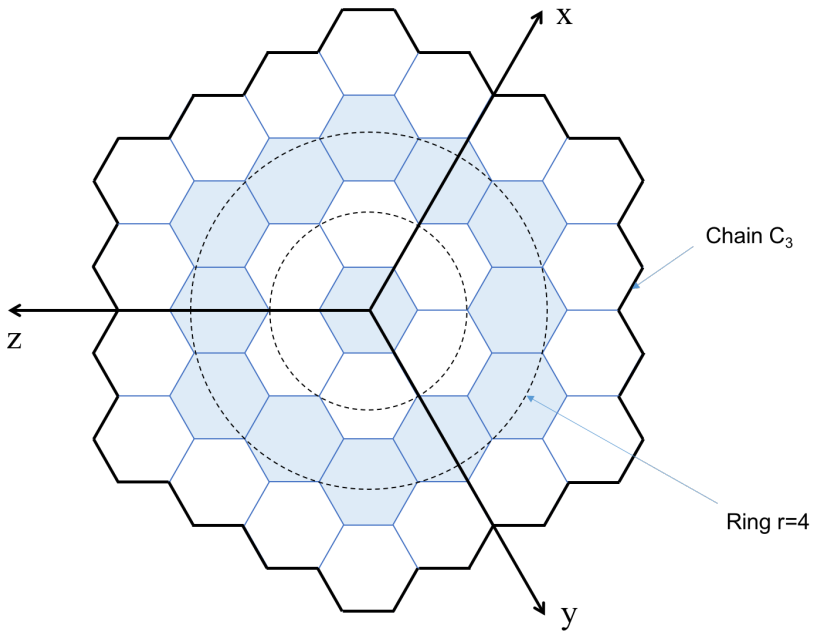

Figure 3. The Honeycomb Structure: the nodes are linked to form a hexagonal structure. The naming is done by node coordinates (x, y, z), but a model to aggregate the nodes in chains may also be used (Ghit, Pop, and Cristea 2011). The rings with $r = 2$ and $r = 4$ are represented. The Honeycomb Overlay has 4 full chains: $C_0, \ldots C_3$.

3.1. *Overlay Network Construction*

The solution for overlay network construction is similar to the approach presented in (Ghit, Pop, and Cristea 2011) for peer-to-peer networks. The authors describe a manner of constructing a honeycomb overlay by splitting the nodes of a chain into 2 different layers, called rings. Each node of the system is uniquely identified by computing a general

index that is constructed from the values of the chain, ring and index within each ring. The indexes from the interior chains are assigned so that they are identical to the ones from the exterior ring of the previous chain. Each new node is added inside a chain in a clockwise direction until this is completed, and then the next chain is formed.

A new construction method was proposed in (Pop et al. 2014). It makes use of the chains of the honeycomb, the indexes within each chain, which are set in a clockwise direction, and the coordinates from the three-axis system. The connection algorithm is constructed by using the lemma given in (Stojmenovic 1997): nodes of a honeycomb of size k can be coded by integer triples (x, y, z), such that

$$-k + 1 \leq x, y, z \leq k \quad and \quad 1 \leq x + y + z \leq 2.$$

Two nodes (x_1, y_1, z_1) and (x_2, y_2, z_2) are connected by an edge if:

$$|x_2 - x_1| + |y_2 - y_1| + |z_2 - z_1| = 1.$$

The coordinates of the neighbors of any node can be obtained by traveling across edges: $(\pm 1, 0, 0)$, $(0, \pm 1, 0)$ and $(0, 0, \pm 1)$. Considering a node (x, y, z), we can compute a list of possible coordinates for its neighbors: $(x + 1, y, z)$, $(x, y + 1, z)$, $(x, v, z + 1)$, $(x - 1, y, z)$, $(x, y - 1, z)$, and $(x, y, z - 1)$. Out of these six triplets, only three define real neighbors. The correct coordinates are the ones that respect the relation $1 \leq x + y + z \leq 2$. The overview of the honeycomb structure is presented in Figure 3.

The nodes are connected as follows. The first chain of the system is C_0. Each new node is added on the current chain in the clockwise direction until the chain is completed. The index of the first peer in each chain is 0. The last value depends on the total number of nodes of the chain. The first peer of the chain is placed on the Ox axe. It takes on the role of a SuperPeer and is responsible for computing the peer's coordinates inside its chain.

To find the next available position, the neighbors of the current peer are computed. From these three neighbors, the value of the current peer is removed. From the remaining peers, the ones that are closer to the origin of the system than the maximum distance of the last chain are removed. This is needed because they have already been assigned, and they form links between the two chains. If there are still more peers, then the one closest to the origin is chosen. As peers are added inside the system, new links between chains are formed. A link is set only when the exterior peer is available.

The position of the SuperPeer is computed by incrementing the value of the x coordinate for each chain. Every two chains, the coordinates for the Oy and Oz axes are decremented. For example, the SuperPeer of $C2$ has the following coordinates $(3, 1, 1)$, and the one on chain $C3$ has $(4, 1, 1)$.

In Figure 3, 30 peers have joined the system in 4 full chains $(C_0 \dots C_3)$. The first two chains are completely formed. On chain $C2$, the last peer is the one with index 5. The first link formed between $C2$ and $C1$ is between the peer with index 1 from the second chain $(2, 0, 1)$ and the one with index 1 on the third chain $(3, 0, 1)$. The second link is between the peer with index 2 from the second chain $(2, 1, 1)$ and the one with index 4 from the third chain $(2, 1, 2)$.

In general, for a full honeycomb structure, the total number of nodes on chain $k = 0, 1, \dots$ is $\overline{C_k} = 6(2k + 1)$, the total number of nodes in the overlay (with k_{max} chains) is

$$N_{k_{max}} = \sum_{k=0}^{k_{max}} \overline{C_k} = \sum_{k=0}^{k_{max}} 6(2k + 1) = 6(k_{max} + 1)^2,$$

the total number of links on the chain C_k is $l(C_k) = 6(2k + 1)$, the number of links between chain C_k and C_{k+1} is $l(C_k, C_{k+1}) = 6(k + 1)$, and the total number of links is

$$L_{k_{max}} = \sum_{k=0}^{k_{max}} l(C_k) + \sum_{k=0}^{k_{max}-1} l(C_k, C_{k+1}) = 9(k_{max} + 1)^2 - 3(k_{max} + 1).$$

3.2. *Fault Tolerance of the Honeycomb Overlay Network*

To be fault tolerant, the peers that leave the system have to be replaced. When a peer detects that one of its neighbors is down, it must send a message to its SP. All of the SuperPeers must keep a list with nodes detected to have exited the system. When a SuperPeer receives a message that a peer is down, it has to forward the information to the SuperPeer of the chain to which the peer is connected. If two neighbors announce that the same neighbor is down, then the SuperPeer from its chain must find a replacement. To modify the existent system as little as possible, the trust values of the peers from the last chain are checked. If there is a peer whose trust value exceeds the minimum required value, it will be moved to the position that was empty. If no such peer is found, then a new peer will be searched for on the $k - 1$ chain and so on until a candidate is found. If a position from the $k - 1$ chain must be filled, then the peer with the greatest value from the last chain will be chosen. For the last chain, the SuperPeer will simply wait to receive a new connection request, and it will fill the empty coordinate.

When a trust manager detects that one of the peers it keeps track of has a bad trust value for a long period of time, it makes a request to the SuperPeer from the peer's chain to remove it. When the SuperPeer receives such requests from the majority of the trust managers of that peer, it will search for a replacement. The same algorithm as in the case of peers leaving the system is then applied.

There are cases when several peers are disconnected from the system, and the honeycomb ends up being split in sections. A few such scenarios are presented in Figure 4.

The first scenario is the simplest one: only 2 peers are disconnected. It can only occur on the last chain of the system. The main part of the system can easily replace the missing 4 peers with new peers. The remaining 2 that are connected to each other can simply disconnect and then reconnect to the initial system or serve as a starting point of another honeycomb. In the latter case, one of them will be chosen as the SP. Because there is no way of knowing their trust values, the choice must be made based on other criteria, such as which one can support routing the largest number of messages.

The second scenario is that in which an entire line of peers leaves the system. It can occur when multiple peers that have sequentially connected to the system are disconnected along with their links to the exterior. Depending on which chain this has occurred for, the largest part of the system can choose to redistribute the nodes from the exterior chains to fill in the gap, or it can simply wait until new peers connect to the system and place them at the empty coordinates. The few peers that remain connected can choose as SuperPeer the peer with the best trust value because now there is still information about them in at least one trust manager. This holds true if there are at least four peers.

The third scenario shows the case when the line of peers forms a U structure. In this case, the hole formed must be filled with existing peers because the system's performance can be greatly reduced. The remaining peers can keep the structure, but reassign their coordinates, and choose the SuperPeer from within the new central chain based on the trust values.

The fourth case is the one that requires the least amount of peers to be disconnected to

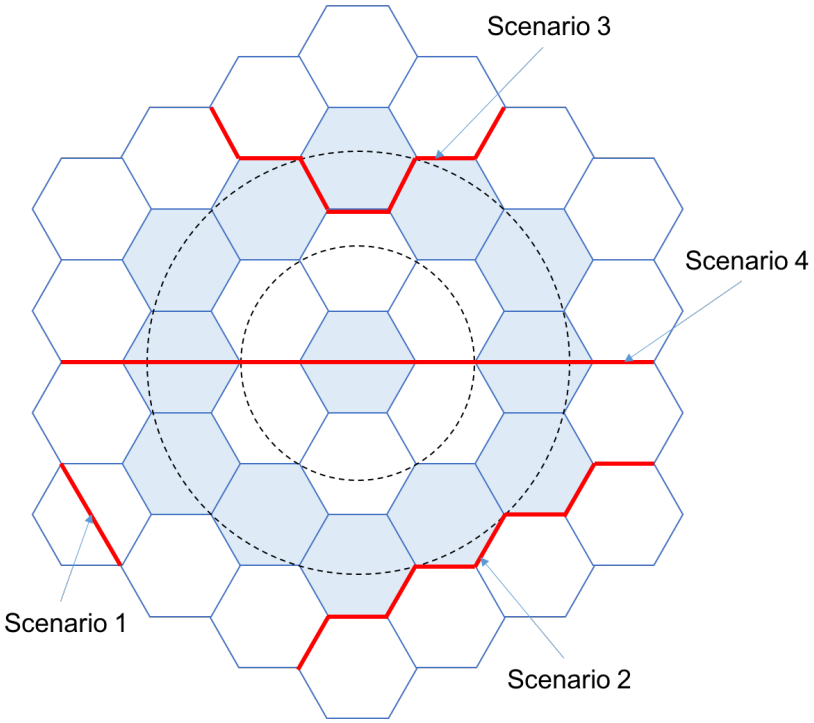

Figure 4. Honeycomb structure splitting: a fault tolerant approach.

break the structure. If the peers are disconnected along any of the axes, then all the chains are destroyed. In this case, 2 new honeycombs will be formed. The part that contains the SuperPeers can choose to redistribute the peers based on their trust value to fill the empty half. The peers that are most trusted will be placed inside the structure, and the least trusted are placed on the exterior chains. A more efficient algorithm is to compute which nodes can be considered as the central chain and to choose the first SuperPeer based on their trust values. The rest of the peers can be redistributed according to their trust values as previously mentioned. In this case, the least number of chains will be broken.

4. Trust Models for Efficient Communication

Peer-to-peer systems are currently widely known and used. Their common usage is for sharing information: audio, video, images or simple text files. Lately, they have been used in e-commerce applications such as e-Bay, Amazon, Yahoo! Auction and Edeal.

As long as users trust that the system is reliable, they will contribute with new interactions in the overlay and will continue to help distribute the existing content. Correct peers upload authentic files, send honest feedback and are available to share a file without modifying its content or forward/reply to queries by giving real information. However, as in the case of any system, there are some peers that will display malicious behavior. The trust value assigned to each peer is used to manage the transaction and reduce the quantity of corrupted content within the system.

Before defining the trust models for efficient communication, we will discuss the construction of the honeycomb overlay over centralized or decentralized systems.

One way for a centralized system to be constructed over a honeycomb overlay is to make the first peer of the system take the role of a SuperPeer. He would be responsible both for assigning new peers their location within the system and with computing and

storing their trust values. The advantage would be that the network would not be flooded with messages when peers make request to know the trust values of others nor when they report the feedback for the transactions. However, the SuperPeer can be overloaded with requests. It not only has to receive the feedback and compute the new trust values and to respond with the current values, but it also has to route simple messages and handle the logic of adding new peers into the system.

Another method would be to assign a SuperPeer to each chain of the system. In this case, the load would be significantly reduced. However, the load for the SuperPeers would be unequal. The one of the interior chains would have a small number of peers they would need to track, but as the chain number increases, the peer number would have a linear growth. The SuperPeer on the first chain would only track 6, the second chain 18. Because the total number of peers of the system is $6(k+1)^2$, where k isthe number of the chain, the SuperPeer on the k chain would have $6(k+2)^2 - 6(k+1)^2 = 6(2k+3)$ peers that it needs to manage.

The fixed structure of the honeycomb allows efficient message routing and assigning of coordinates to peers so that their anonymity can be protected. By routing messages to direct neighbors, only they know the identities of the peers, which increases the system's reliability and security and reduces the probability of attacks from malicious peers.

This scheme also permits choosing the managers such that all nodes can compute their individual position within the system. This reduces the number of messages that are routed through the system because peers can now only communicate through direct neighbors. Managers can be protected from malicious peers by assigning them so that they are not directly connected to the peers whose trust values they store and compute.

4.1. *Trust Model of Nodes in the Honeycomb Overlay*

When a new node joins the system, it receives a set of coordinates. All messages that are sent through the system are routed based on these coordinates. This increases the system's security because the only nodes who know the identities of others are the direct neighbors. However, it has a downside: nodes need to route messages that are not related to their requests. The total number of routed messages needs to be kept as low as possible so that nodes are not overloaded with messages that do not bring direct value. This requires employing a routing algorithm that offers the shortest path between 2 nodes when routing the messages.

The node that made the request for a certain resource rates each interaction. It then sends its feedback to the manager peers (MPs) of the one that replied with the resource. The feedback is given based on the time interval in which the request is completed and on the validity of the data received. If the message is corrupted in any way, the lowest possible trust value is given. By considering the response time for the message in which a node announces it has the resource and the number of hops between the nodes, the trust value is computed. If the distance is short, then the time interval in which the transaction is expected to take place is smaller. If the distance is larger, the time interval is longer. If the node sends the needed resource in a shorter or equal time to the minimum computed time interval, then the trust value is the largest allowed. As more time passes over the computed limit, the trust value decreases.

The trust value (TV) of a peer is given in (Vişan, Pop, and Cristea 2011), where a solution for peer-to-peer trust management is described. The trust value is computed in an adaptive way and is based on the feedback received for the last transaction, but it also takes into account all the feedback values previously received:

$$[TV^{new}] = [TV^{old}] + 1 \quad and \quad \{TV^{new}\} = \frac{[TV^{old}]\{TV\}^{old} + F}{[TV^{new}]},$$

where the $[TV]$ represents the number of instances of feedback received for the node. The feedback value is F, and $\{TV^{new}\}$ is the new trust value. This manner of computing the trust does not account very well for traitor nodes. After behaving correctly for a certain period of time, they are able to make a large number of transactions before their trust value is considerably lowered, as is shown in the experimental results section of this paper.

4.2. *Choosing the Manager Peers*

Each peer is assigned 9 managers. The distance between managers and the peer whose trust value is stored and computed by them is 3 vertices. Their position is computed in the following 2 cases (see Figure 5):

- Case 1: A node with coordinates (x, y, z) (white) has the following trust managers (black):

$$\begin{array}{l} (x+1, y-2, \quad z) \\ (x+1, y-1, z-1) \\ (x+1, y, \quad z-2) \\ (x, \quad y+1, z-2) \\ (x-1, y+1, z-1) \\ (x-2, y+1, \quad z) \\ (x-2, y, \quad z+1) \\ (x-1, y-1, z+1) \\ (x, \quad y-2, z+1) \end{array}$$

- Case 2: A node with coordinates (x, y, z) (white) has the following trust managers (black):

$$\begin{array}{l} (x-1, y+2, \quad z) \\ (x-1, y+1, z+1) \\ (x-1, y, \quad z+2) \\ (x, \quad y-1, z+2) \\ (x+1, y-1, z+1) \\ (x+2, y-1, \quad z) \\ (x+2, y, \quad z-1) \\ (x+1, y+1, z-1) \\ (x, \quad y+2, z-1) \end{array}$$

The algorithm for assigning the managers scales very well with the system because the number of peers that are managed does not increase as more and more chains are formed within the system. Figure 5 shows a few peers and their corresponding trust managers. The peers are in the center of the formation, and their managers are placed in the exterior coordinates. The general flow for a resource request is described in Algorithm 1.

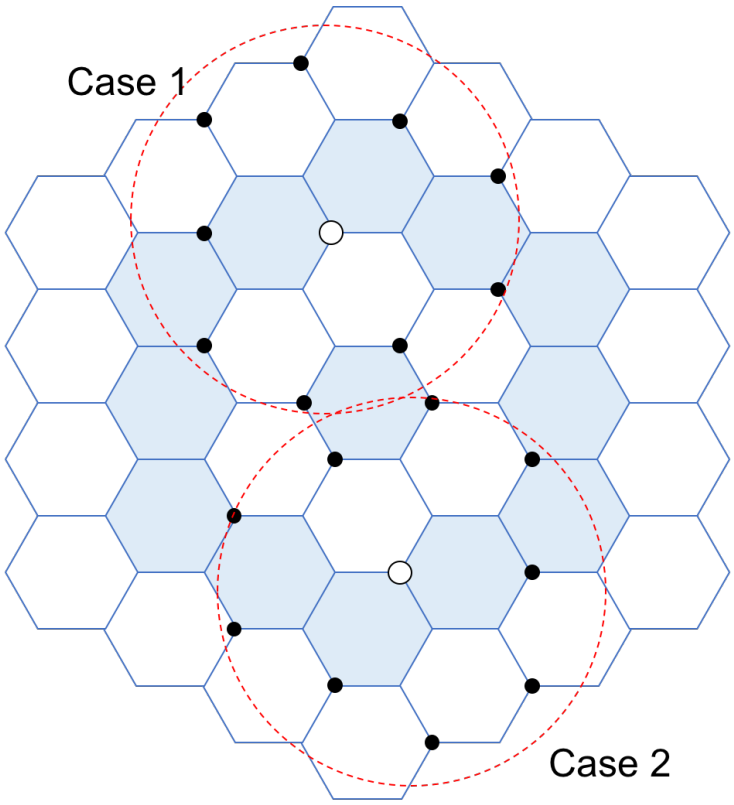

Figure 5. Manager Peers (MPs).

Algorithm 1 Resource Request.

```
1: Node A broadcasts a request for resource R;
2: All the available peers respond to A by sending their coordinates;
3: for all responses (ordered by arrival time) do
4:    A will send queries to 3 of the closest MPs to check the trust values of the peers;
5:    The MPs respond to the query with the trust values;
6:    if A finds a peer B with an acceptable trust value then
7:       A makes a resource request to B;
8:    end if
9:    After the transaction is finished, A sends the feedback to the 3 managers of B;
10: end for
11: The initial MPs send the feedback values to the others;
12: for all MPs do
13:    MPs update their values;
14: end for
```

By requesting the trust values only from the closest managers, the values arrive faster, and fewer messages are sent through the system. It also prevents the feedback from being routed through the peer whose trust value was requested. This increases the security and restricts the attacks of malicious peers.

All the messages are routed through the system by following the shortest path between the peers. The exception is represented by the messages that are routed from the first 3 MPs to the other ones. These messages are routed so that the feedback never crosses through the peer about whom the feedback is given.

The trust managers are chosen when a peer joins the system. Because the peers are

assigned on each chain until it is completed, this means that not all the managers can be assigned from the beginning. As a new peer connects to the system, it computes the peers from whom it should be responsible and queries the existent managers for the trust values.

To minimize the number of sent messages, the only broadcast message that can be sent is that for a resource request. To further optimize the system, the first request can be sent only k hops through the system. If no answer is received, or if the trust values of the peers that have the resource are too small, the distance is doubled.

4.3. *Proposed Trust Model of Links in the Honeycomb Overlay*

A new model to manage the trust between two nodes via a link that does not depend on the number of transactions that took place for a specific node is proposed. It also takes all the feedback values previously received, the current feedback value and the trust value of the neighbor node that rated the transaction into account:

$$T^{new}(X,Y) = \alpha T^{old}(X,Y) + (1-\alpha)[T_Y F_X + (1-T_Y)T^{old}(X,Y)],$$

$$T^{new}(Y,X) = \beta T^{old}(Y,X) + (1-\beta)[T_X F_Y + (1-T_X)T^{old}(Y,X)],$$

where:

- $T(X,Y)$ is the trust value of the link (X,Y);
- T_X and T_Y are the trust values of the neighbor nodes that gave the feedback; for example X gave the feedback F_Y to Y and Y gave the feedback F_X to X, X and Y are direct connected;
- F_X and F_Y are the feedback values.

All values are bound in the interval $[0,1]$. The feedback value is computed by taking the validity of the data that has been sent and the time it took to fulfill the request into account. If a message is corrupt (the check-sum is wrong), then the feedback value will be 0. Otherwise, the value will be computed based on the time it took from the moment the initial request for the resource was made until the response that the node can share it was received. The general model is adaptive with $0 \leq \alpha, \beta \leq 1$. We consider in the current approach that $\beta = \alpha$ (the trust values for links are computed with the same combination ratio). In this model, we consider only one old value, but an extended model may take the mathematics of non-commutative Markov processes (Şerbanescu 1998a) and unitary processes (Şerbanescu 1998b) into consideration.

5. Experimental Results

The experimental results, performed by simulation, are split into two categories. First, we analyze the way peers connect to the system. This is followed by the adaptive method on how the trust can be computed. To increase maintainability and allow different components to be easily changed, the simulator is split into several packages, each implementing an important functionality (see Figure 6).

Both methods for the trust management (for nodes and links) are shown below. Two main scenarios will be considered: when a peer first connects to the system, and how a peer behaves after a period of time. We have presented old experimental results, obtained

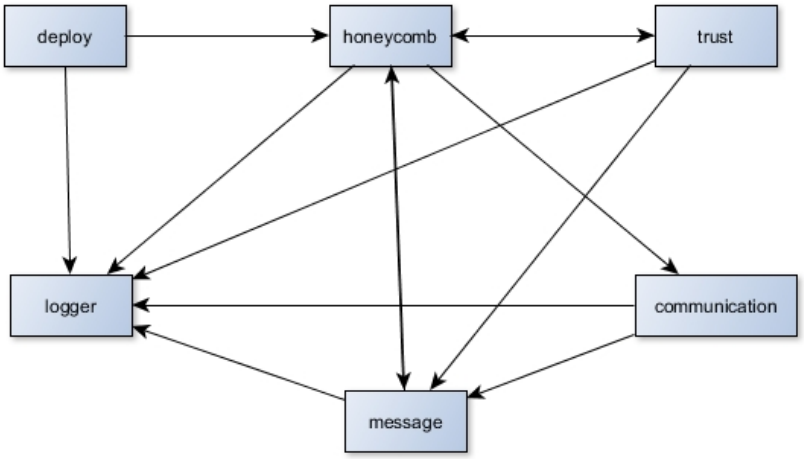

Figure 6. Honeycomb Simulator Structure.

in (Pop et al. 2014), and the new results obtained for joining the network (node trust and links trusts) for different numbers of transactions (20-30, 50-60) in this section.

5.1. *Overlay Construction*

For the construction of the overlay, two cases are considered. When a new node connects to the system, it must first send the join message to the connect node. This can be chosen from a random node that has already connected to the system, or it can choose to connect to one of the SuperNodes. Figure 7 shows the number of messages that are sent within the system, in the worst-case scenario, until the new node finds its own position within the honeycomb system. The number of messages that are sent when choosing to connect to an SP is considerably lower than when connecting to a random node. When connecting the first 600 nodes (the 10th chain), the maximum number of messages sent in the first case is 9, while in the second it is 31.

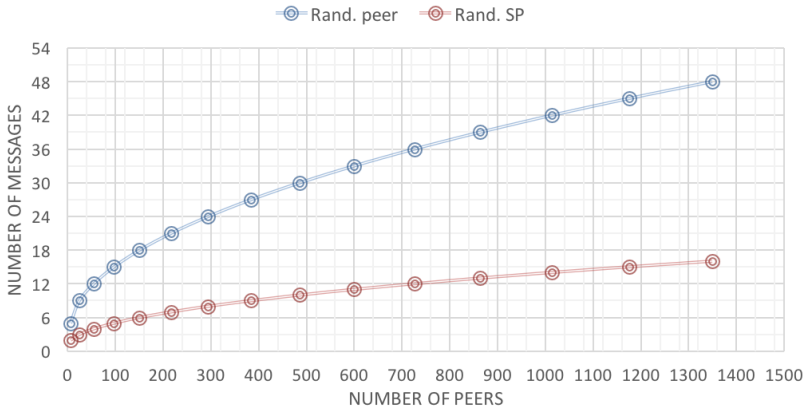

Figure 7. Overlay Construction.

5.2. *Trust Management*

Both methods for the trust management are shown below. Two main scenarios will be considered: when a node first connects to the system, and how a node behaves after a

period of time.

5.2.1. Node's Trust

The first graph contains the trust values for a few types of peers that are managed by using the formula for node trust value computation. The best and worst case scenarios and the most frequent case are taken into consideration. The new trust values of the peers are shown after each new transaction takes place (see Figure 8).

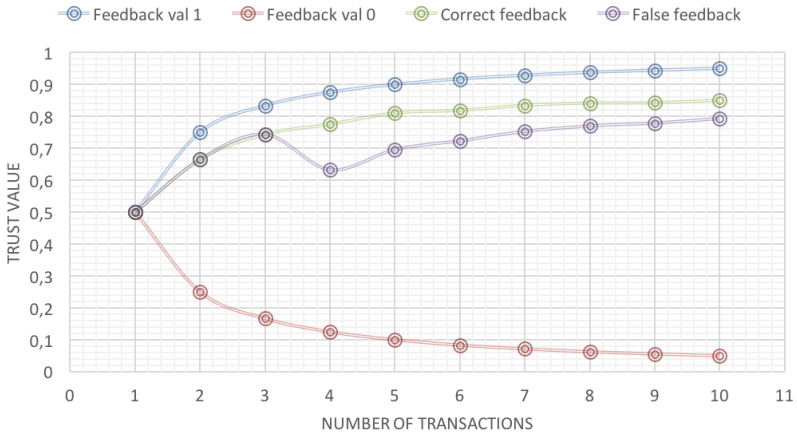

Figure 8. Node's trust evolution after joining the honeycomb overlay network.

In Figure 8, the blue series shows the case of an ideal peer that only received the largest value for feedback, 1. After only three transactions, it can be considered a trusted peer, with his trust value reaching 0.875. The red series shows a malicious peer that only sends 14 corrupted pieces of data and whose feedback value is 0. After three transactions, no other peer will interact with him because his trust value will be 0.125. The remaining two scenarios contain the behavior of a correct peer. The correct feedback values for him range between $[0.83, 0.95]$. The mauve series contains, for the third transaction, a false feedback given by a malicious peer of 0.3 instead of 0.87. The trust value of the peer drops to 0.63 instead of 0.77, but it will constantly grow until it reaches the normal value.

The next two graphs (Figure 9 and Figure 10) contain the behavior of a peer that has reached a trust value of 0.85. As in the previous case, the ideal peer is shown in blue and the malicious one in red. The next image contains the transactions from 20 to 29, and the one after it from 50 to 59. Both contain the exact trust value. The only difference is given by the transaction number.

As can be seen, the transaction number has a large influence on the trust values. If a malicious peer behaves in a good manner for a certain period of time, it can then distribute malicious content for a long period of time before his trust value is lowered enough to reflect his new behavior. In the first case, after 10 transactions, his trust value will be 0.58, but in the last one it will reach 0.72, so he will still be seen as a good peer.

5.2.2. Adaptive Link's Trust

Multiple values for α used in the model for link trust have been considered. The first value considered is $\alpha = 7/8$. This has been chosen because it is the default value for computing the RTT value for sending TCP packets (Postel 1981).

The $\alpha = 7/8$ value influences the trust values the least. When a node first connects to the system, only after 7 transactions will his value be greater than 8. The $\alpha = 5/8$ value

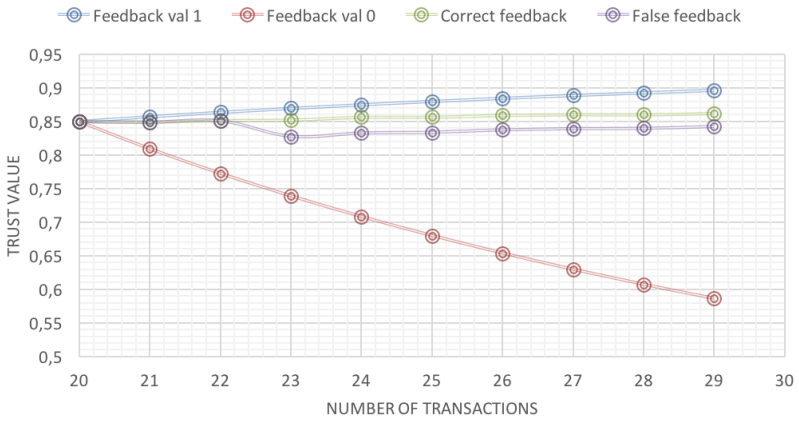

Figure 9. Node's trust: transactions 20-30.

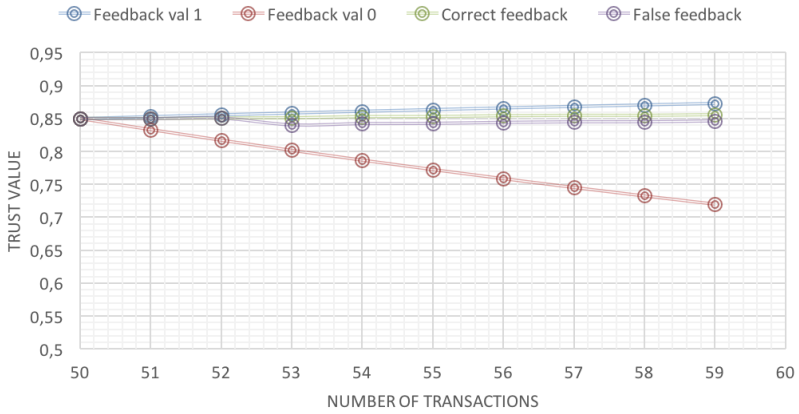

Figure 10. Node's trust: transactions 50-60.

requires only 2 transactions for the trust value to reach 8. For $\alpha = 3/4$, it takes 4. The values for the malicious nodes can be observed as being symmetrical.

The $\alpha = 3/4$ value seems to be best choice for when a node first joins the system. The correct nodes can gain a good trust value in a relatively short period of time and the malicious ones can be detected in an early stage. The $\alpha = 5/8$ value has not been chosen because, if a node receives a false feedback, then its trust value is lowered too much. To gain a better perspective of how a false feedback can influence a node, the following case has been considered. The trust values for the nodes that give the feedback are in the interval $[0.87, 0.95]$. The feedback values range from $[0.83, 0.95]$. The initial trust of the node is 0.85. The false feedback is given for the 3rd transaction, and its value is 0.3 (see Figure 11).

The blue series in Figure 12 contains the trust values for the real feedback. The green one is for the false feedback and the value $\alpha = 7/8$, and the red one is for $\alpha = 3/4$. As it can be seen, the trust value drops too much, from 0.85 to 0.74, in the case of the false feedback for $\alpha = 3/4$. Thus, for the good nodes to not be easily influenced by malicious ones the $\alpha = 7/8$ value has been chosen.

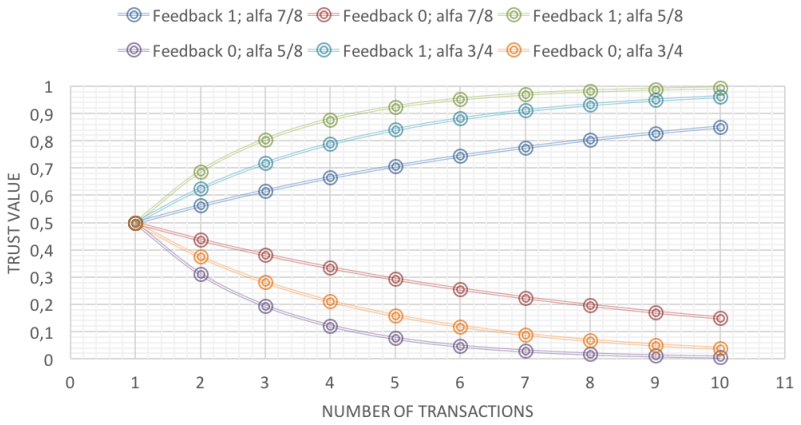

Figure 11. Adaptive trust: join network.

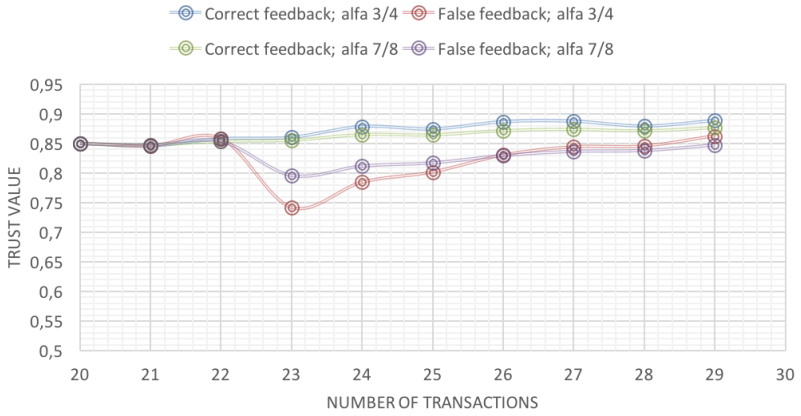

Figure 12. Adaptive trust: false feedback.

6. Conclusion

The applicability of the proposed trust models in MCC highlights the possibility of having adaptive behavior for the communication overlay to ensure the efficiency of communication and interaction. In this paper, we extended the discussion related to the honeycomb overlay, previously introduced by our research group, and we considered several scenarios for fault tolerance. For the construction of the honeycomb, two scenarios have been presented: (i) if a new peer connects to the system by using a random peer from the system, the number of messages that are sent is significantly greater, but the security is better because the real address of the SuperPeers is not known by everybody; (ii) however, if a new SuperPeer is being attacked, it would only have an impact on how new peers are added inside the system, and only for the last chosen SuperPeer. Thus, if the information about the peer last added to the system is saved on other SuperPeers, it would only delay the new connections until a new SuperPeer can take the place of the one that has been attacked. In conclusion, it is preferred for the new peers to be able to connect to the existent SuperPeers. This is applicable in e-Commerce where mobile users choose an entry point in the system, one that is considered trusted by the user.

Two methods for computing the new trust values have been considered: node's trust and link's trust. The first one takes the feedback received for a transaction and the number of transactions into account, and a second one considers the feedback value and

the trust value of the peer that gave it. For the latter, multiple values for have been considered to manage the impact of the new feedback (in an adaptive manner). Based on the experimental results, the proposed model with the value of $\alpha = 7/8$ yields the best results. It is not influenced by the number of transactions, so the system is protected against traitor peers. In e-Commerce, we face cases in which two entities X and Y give feedback to each other, but the trust values are different ($TV(X,Y) \neq TV(Y,X)$). These trust models, beside their performance regarding trust value computation, represent suitable solutions for any e-Commerce interaction models where a structured overlay, such as a honeycomb, with a large number of users can be built.

Acknowledgment

The research presented in this paper is supported by projects: *CyberWater* grant of the Romanian National Authority for Scientific Research, CNDI-UEFISCDI, project number 47/2012; *MobiWay*: Mobility Beyond Individualism: an Integrated Platform for Intelligent Transportation Systems of Tomorrow - PN-II-PT-PCCA-2013-4-0321; *clueFarm*: Information system based on cloud services accessible through mobile devices, to increase product quality and business development farms - PN-II-PT-PCCA-2013-4-0870; *DataWay*: Real-time Data Processing Platform for Smart Cities: Making sense of Big Data - PN-II-RU-TE-2014-4-2731.

This work has been partially supported by funds from the Spanish Ministry for Economy and Competitiveness (MINECO) and the European Union (FEDER funds) under grant COMMAS (ref. TIN2013-46181-C2-1-R).

We would like to thank the reviewers for their time and expertise, constructive comments and valuable insight.

References

Achim, Ovidiu-Marian, Florin Pop, and Valentin Cristea. 2011. "Reputation Based Selection for Services in Cloud Environments." In *Network-Based Information Systems (NBiS), 2011 14th International Conference on,* 268–273. Sept.

Barbera, Marco Valerio, Sokol Kosta, Alessandro Mei, and Julinda Stefa. 2013. "To offload or not to offload? the bandwidth and energy costs of mobile cloud computing." In *INFOCOM, 2013 Proceedings IEEE,* 1285–1293. IEEE.

Chen, Dan, Lizhe Wang, Xiaomin Wu, Jingying Chen, Samee U. Khan, Joanna Kolodziej, Mingwei Tian, Fang Huang, and Wangyang Liu. 2013. "Hybrid Modelling and Simulation of Huge Crowd over a Hierarchical Grid Architecture." *Future Gener. Comput. Syst.* 29 (5): 1309–1317.

Ciobanu, Nicolae-Valentin, Dragos-George Comaneci, Ciprian Dobre, ConstandinosX. Mavromoustakis, and George Mastorakis. 2015. "OpenMobs: Mobile Broadband Internet Connection Sharing." In *Mobile Networks and Management,* Vol. 141 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering* edited by Ramn Agero, Thomas Zinner, Rossitza Goleva, Andreas Timm-Giel, and Phuoc Tran-Gia. 244–258. Springer International Publishing.

De, Sohini, and Suddhasil De. 2013. "Uncoupling in Services of Mobile Cloud Computing Using Tuple Space Model: Design and Formal Specifications." In *Proceedings of the First International Workshop on Mobile Cloud Computing & Networking,* Bangalore, India. MobileCloud '13. 27–32. New York, NY, USA: ACM.

Ding, Chen, Chen Yueguo, and Cheng Weiwei. 2004. "A survey study on trust management in P2P systems." *Department of Computer Science, School of Computing, National University of Singapore* .

Fedotova, Natalya, and Luca Veltri. 2009. "Reputation management algorithms for DHT-based peer-to-peer environment." *Computer Communications* 32 (12): 1400–1409.

Ghit, Bogdan, Florin Pop, and Valentin Cristea. 2011. "Using bio-inspired models to design peer-to-peer overlays." In *P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC), 2011 International Conference on,* 248–252. IEEE.

Jiang, Liming, Jian Xu, Kun Zhang, and Hong Zhang. 2012. "A new evidential trust model for open distributed systems." *Expert Systems with applications* 39 (3): 3772–3782.

Li, Qing, Ze-yuan Wang, Wei-hua Li, Jun Li, Cheng Wang, and Rui-yang Du. 2013. "Applications integration in a hybrid cloud computing environment: modelling and platform." *Enterprise Information Systems* 7 (3): 237–271.

Marin, Radu-Corneliu, and Ciprian Dobre. 2013. "Reaching for the clouds: contextually enhancing smartphones for energy efficiency." In *Proceedings of the 2nd ACM workshop on High performance mobile opportunistic systems,* 31–38. ACM.

Marti, Sergio, and Hector Garcia-Molina. 2006. "Taxonomy of trust: Categorizing P2P reputation systems." *Computer Networks* 50 (4): 472–484.

Mekouar, Loubna, Youssef Iraqi, and Raouf Boutaba. 2010. "Reputation-based trust management in peer-to-peer systems: taxonomy and anatomy." In *Handbook of Peer-to-Peer Networking,* 689–732. Springer.

Opara, Emmanuel, and Omprakash Gupta. 2015. "Connectivity via Web Services: An Analysis for Interoperable E-Commerce.." *Communications of the IIMA* 4 (4): 9.

Papanikolaou, Katerina, Constandinos X Mavromoustakis, George Mastorakis, Athina Bourdena, and Ciprian Dobre. 2014. "Energy Consumption Optimization using Social Interaction in the Mobile Cloud." In *Proc. 6th Int. Conf. MONAMI,* 1–14.

Pop, Florin, Oana-Maria Citoteanu, Ciprian Dobre, and Valentin Cristea. 2014. "Resource Trust Management in Auto-Adaptive Overlay Network for Mobile Cloud Computing." In *Proceedings of the 2014 IEEE 13th International Symposium on Parallel and Distributed Computing,* ISPDC '14. 162–169. Washington, DC, USA: IEEE Computer Society.

Postel, Jon. 1981. "Transmission control protocol." *RFC: 793* .

Pyattaev, Alexander, Kerstin Johnsson, Sergey Andreev, and Yevgeni Koucheryavy. 2013. "3GPP LTE traffic offloading onto WiFi Direct." In *Wireless Communications and Networking Conference Workshops (WCNCW), 2013 IEEE,* 135–140. IEEE.

Qureshi, Basit, Geyong Min, and Demetres Kouvatsos. 2012. "A distributed reputation and trust management scheme for mobile peer-to-peer networks." *Computer Communications* 35 (5): 608–618.

Resnick, Paul, and Richard Zeckhauser. 2002. "Trust among strangers in internet transactions: Empirical analysis of ebays reputation system." *The Economics of the Internet and E-commerce* 11 (2): 23–25.

Samad, Javeria, Seng W. Loke, and Karl Reed. 2015. *Mobile Cloud Computing.* 153–190. John Wiley & Sons, Inc.

Şerbanescu, Cristina. 1998a. "Noncommutative Markov Processes as Stochastic Equations' Solutions." *Bull. Math. Soc. Sc. Math. Roumanie Tome 41* 89 (3): 219–228.

Şerbanescu, Cristina. 1998b. "Stochastic differential equations and unitary processes." *Bull. Math. Soc. Sc. Math. Roumanie Tome 41* 89 (3): 311–322.

Şerbanescu, Vlad-Nicolae, Florin Pop, Valentin Cristea, and Ovidiu-Marian Achim. 2012. "Web Services Allocation Guided by Reputation in Distributed SOA-Based Environments." In *Parallel and Distributed Computing (ISPDC), 2012 11th International Symposium on,* 127–134. June.

Shafer, Glenn, et al. 1976. *A mathematical theory of evidence.* Vol. 1. Princeton university press Princeton.

Silaghi, Gheorghe Cosmin, Alvaro E Arenas, and Luis Moura Silva. 2007. "Reputation-based trust management systems and their applicability to grids." *Institutes on Knowledge and Data Management and System Architecture, CoreGRID-Network of Excellence, Technical Report TR-0064* .

Song, Weijing, Lizhe Wang, R. Ranjan, J. Kolodziej, and Dan Chen. 2015. "Towards Modeling Large-Scale Data Flows in a Multidatacenter Computing System With Petri Net." *Systems Journal, IEEE* 9 (2): 416–426.

Stojmenovic, Ivan. 1997. "Honeycomb networks: Topological properties and communication algorithms." *Parallel and Distributed Systems, IEEE Transactions on* 8 (10): 1036–1042.

Ţăpuş, Nicolae, and Pantelimon George Popescu. 2012. "A new entropy upper bound." *Applied Mathematics Letters* 25 (11): 1887–1890.

Tian, Chunqi, and Baijian Yang. 2011. "Trust, a reputation and risk based trust management framework for large-scale, fully decentralized overlay networks." *Future Generation Computer Systems* 27 (8): 1135–1141.

Tian, Chunqi, Baijian Yang, Jidong Zhong, and Xiaojian Liu. 2014. "Trust-based incentive mechanism to motivate cooperation in hybrid P2P networks." *Computer Networks* 73: 244–255.

Vişan, Andreea, Florin Pop, and Valentin Cristea. 2011. "Decentralized Trust Management in Peer-to-Peer Systems." In *Parallel and Distributed Computing (ISPDC), 2011 10th International Symposium on,* 232–239. IEEE.

Wang, Yufeng, and Akihiro Nakao. 2010. "Poisonedwater: An improved approach for accurate reputation ranking in P2P networks." *Future Generation Computer Systems* 26 (8): 1317–1326.

Yu, Zhen, Jie Zhu, Guicheng Shen, and Haiyan Liu. 2014. "Trust Management in Peer-to-Peer Networks." *Journal of Software* 9 (5): 1062–1070.

Zhang, Lin, Yongliang Luo, Fei Tao, Bo Hu Li, Lei Ren, Xuesong Zhang, Hua Guo, Ying Cheng, Anrui Hu, and Yongkui Liu. 2014. "Cloud manufacturing: a new manufacturing paradigm." *Enterprise Information Systems* 8 (2): 167–187.