EigenTrust for Hierarchically Structured Chord

Kalonji Kalala(✉), Tao Feng, and Iluju Kiringa

School of Electrical Engineering and Computer Science, University of Ottawa,
800 King Edward Avenue, Ottawa, ON K1N 6N5, Canada
{hkalo081,tfeng038,iluju.kiringa}@uottawa.ca
http://engineering.uottawa.ca/eecs/

Abstract. The paper proposes Hierarchical EigenTrust, an extension of EigenTrust, a trust and reputation algorithm that was proposed for flat peer-to-peer networks. The paper introduces the components of the hierarchical model architecture based on Chord, a scalable P2P (Peer-to-Peer) lookup service for Internet applications. The paper also extends the EigenTrust scheme to the hierarchically structured Chord P2P network. The proposed algorithm handles a huge number of nodes disseminated in different Chord rings, which improves complexity and reduces the number of malicious nodes. The experiments verify and compare the reduction of downloads from malicious peers, load distribution as well as convergence speed between a flat structured network and a hierarchically structured network. Results of the experiments show that hierarchical EigenTrust outperforms the flat EigenTrust in a P2P network that uses only one big ring.

Keywords: Hierarchical P2P network · Hierarchical Chord · Trust · Reputation · DHT · Overlay network · EigenTrust

1 Introduction

1.1 Motivation

A P2P system is defined as a group of organized autonomous peers in which peers share distributed resources (files, computing and services) without any centralized coordinating entity. Chord [14] has long emerged as an efficient structured P2P architecture and system. With the growing interest in P2P networks, combined with the emergence of new computing paradigms such as Big Data and the Internet of things (IoT), a hierarchical design is desired to overcome disadvantages associated with pure P2P networks. These disadvantages include the challenge of managing a network when the number of nodes increases exponentially and thus deteriorates the performance of the entire network, as well the challenge of handling the exponential increase in the number of things in a IoT environment. More specifically, consider an IoT fleet management scenario where a large number of vehicles must be managed. The involved vehicles can be considered as the leaf nodes of a tree-like structure that is formed by Chord ring of

G. Livraga and C. Mitchell (Eds.): STM 2017, LNCS 10547, pp. 203–212, 2017.
DOI: 10.1007/978-3-319-68063-7_14

further Chord rings and be organized around local rings with super-peers that can function as the points at which data can be merged on the local ring and then be passed to a hierarchically higher ring. The hierarchical model reduces the network traffic, decreases the workload and the lookup path length because the number of hops is significantly reduced in addition to the lookup latency. The goal of trust and reputation systems is to evaluate the trustworthiness of peers, provide value to any transaction made among peers, and distinguish good peers from bad peers based on previous interactions and feedbacks from peer transactions. So far, there has been no a serious investigation or study undertaken to implement trust and reputation in hierarchically structured P2P networks. Specifically, this paper proposes a hierarchical redesign of Chord as an efficient solution for managing the complexity of P2P networks amenable for IoT applications and a hierarchical EigenTrust, an extension of EigenTrust.

1.2 Problems

We are confronted to the problem of adequately structuring P2P networks in the presence of a gigantic number of peers. Henceforth there is a need to redesign trust and reputation algorithms that were designed in the context of flat structured P2P network to make those algorithms amenable for the hierarchically structured P2P network context. Due to its popularity in the last ten years, we have considered Chord rings as the existing P2P network to further restructure in order to master the complexity of gigantic P2P networks. We will focus on the design, analysis, and implementation (via simulation) of a trust and reputation algorithm for hierarchically structured Chord systems. Our choice was based EigenTrust because it is one of the earlier and mostly cited trust and reputation algorithms for structured P2P, also many recent algorithms are just trying to improve EigenTrust.

2 Related Work

Many structured P2P systems like Chord, CAN [11], Pastry [12] are P2P overlay networks that implement a key-based and deterministic algorithm for the routing of messages to the destination key that holds the searched content. These overlay networks support storage and search interfaces such as Distributed Hash tables (DHT), a lookup strategy to route and specify the location of objects in the P2P network. This allows them to perform lookup service in *O(log N)*. In NodeRanking [10] the reputation value of a node i is evaluated by the number of references (emails, personal web pages) provided by other nodes in the network. PowerTrust [16] carefully and dynamically chooses a small number of power peers with hight reputation value. PowerTrust ameliorates global reputation accuracy as well as the rate of aggregation speed. Absolute Trust [1] is a algorithm for aggregation trust in P2P networks for peers that only exchange files. The algorithm and the metric used determine the true past behavior of peers. This algorithm does not need a normalization of trust, pre-trusted peers or any centralized authority.

3 Hierarchically Structured Chord

3.1 Chord

Chord organizes all peers in a ring (or circle) that maintains all keys in the range from $2^m - 1$. It maps keys with corresponding content nodes. Each node maintains a routing table called finger table that maintains the successors, predecessors and fingers of the node. Every finger table contains up to m entries, m being the number of bits in the hash key. Each node only possesses knowledge of its successors on the identifier circle in order to execute look up operations and knows little about distant nodes. Thus, each node can only maintain information for a small number of nodes, i.e. a total of *O(log N)* fingers [8]. In the finger table of a node n, the identifier of the first node s (at i^{th} entry) that comes after n is determined by $s = successor(n + 2^{i-1})$. Node s is what is called the finger of n. When a query is sent to a node, the node first needs to inspect its own local storage to ensure if it carries the desired data item. If it holds the desired data item, it simply sends the result to the requester. Otherwise, it redirects the query to its nearest successor node according to its finger table.

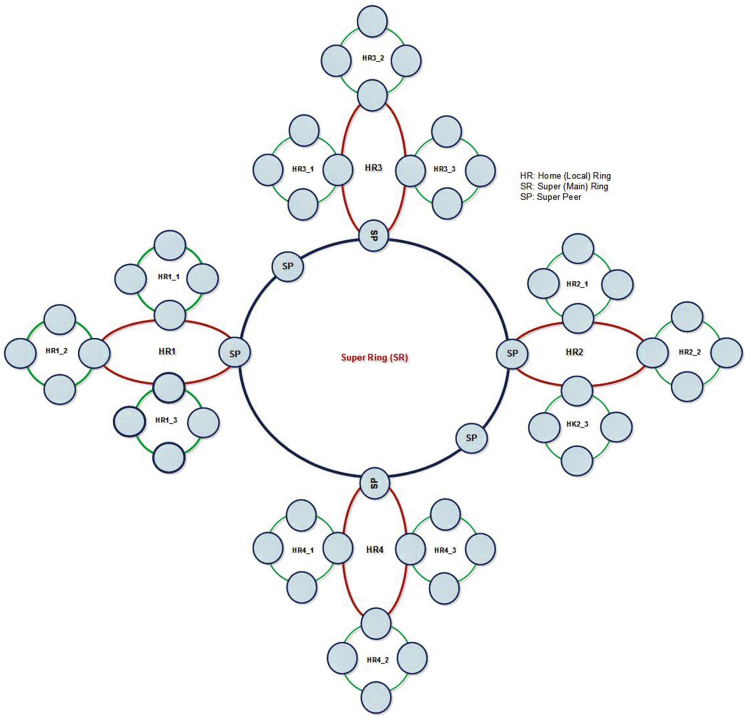

Fig. 1. Hierarchical Chord model

3.2 Extending Chord to a Hierarchical Structure

Each group in Figure 1 has a unique designated group *id*. The directed graph *U, X*, with X = $\{g_1, \ldots, g_I\}$ = Set of all groups and *U = set of virtual edges among nodes (groups in this case)*. Each group contains at least one superpeer [3]. The lookup service is performed in two steps; first it locates the group that is in charge of the key, and inside the group it locates the peer that is in charge of the key. For N-tiers DHT, the lookup service should go deeper in the hierarchy, by passing through groups until it reaches the group that is in charge of the key. For two-tier DHT, operations are executed in the following orders: A superpeer of group S_j of superpeers of group j receives the query and can then transfer it to the peer p_j belonging to group G_j that is in charge of the key k. When p_j responds back to the query, the response can be transmitted using the reverse path used by the query message, or can be transmitted directly from peer j to the peer i. The lookup system at the top-level administers an overlay of groups. When a new peer p joins a group, it is provided with the *id* of the group to be recognized, such as the name of the group. Then p contacts a node p' already participating in the group to request the IP address of the group's superpeer(s)for the group key g.

4 EigenTrust for Hierarchical Chord

4.1 EigenTrust

EigenTrust [7] is a distributed trust reputation system based on individual reputation and uses distributed control. A peer conserves a record of all previous transactions in a local trust vector $\overrightarrow{c_i}$. Vector $\overrightarrow{c_i}$ consists of all local trust values c_{ij} that peer i has attributed to other peers j. It can be represented as $\overrightarrow{c_i} = (c_{i1}, c_{i2}, c_{i3}, \ldots, c_{in})$. All c_{ij} are positives because there are normalized as $c_{ij} = \dfrac{max(s_{ij}, 0)}{\sum(s_{ij}, 0)}$, and the sum of $(c_{i1} + c_{i2} + c_{i3} + \ldots + c_{in}) = 1$. All local trust values are represented in a matrix $[(c_{ij})]$ defined by C. A gossiping algorithm is used to assemble the global reputation t_i of the P2P network. The global trust value t_i determines the trust that the entire system places in the peer i. $\overrightarrow{t}$ determines the global trust vector of the entire system. EigenTrust evaluates the left principal eigenvector of a matrix of normalized local trust values, so that it can calculate the global trust value of a peer. It computes local reputation and global reputation and it uses transitivity to measure trust. This system needs a group of honest peers, pre-trusted peers $\overrightarrow{P}$ as start vectors to eliminate malicious peers.

4.2 Extending EigenTrust to Hierarchical Chord

Algorithm 1 represents the extended secure EigenTrust to hierarchical Chord. In this algorithm, some important information about group G_j, RRM_j and pre-trusted peers (superpeers) P_j is added to improve the lookup process in the

Algorithm 1. (EigenTrust for Hierarchical Chord)

A_s^i= Set of peers (from local and remote HR (Home Ring)) which have downloaded files from peer i .
B_s^i= Set of peers (from local and remote HR) from which peer i has download files
G_j = Group id
P_j = : Pre-trusted Superpeers
RRM_j : Pre-trusted Superpeer (gateway).
$hops_j$: Number of hops from ring to ring
. $Rpeer_j$: Number of remote peers to send scores

1: **for** each (G_j) **do**
2: $P_j \leftarrow const$
3: $hops_j \leftarrow const$
4: $Rpeer_j \leftarrow const$
5: **end for**
6: **for** each peer i in a (G_j) **do**
7: **do**
8: Submit *Local trust values* $\boldsymbol{c_i}$ to all score managers at positions $h_m(pos_i), m = 1, 2, ...M-1$
9: **if** Local trust value of peers $\in$ other HR's **then**
10: Transfer their scores to the local RRM
11: determine the id of the remote HR
12: Local RRM sends value to remote RRM of the HR, to the score managers
13: **end if**
14: Collect all *Local trust values* $\boldsymbol{c_d}$ and set of B_d^i
15: Submit daughter d's *Local trust values* c_{dj} to score managers $h_m(pos_i), m = 1, 2, ...M-1$ with $j \in B_d^i$
16: Collect acquaintances A_d^i of daughter peers
17: Communicate with other HR to collect acquaintances A_d^i of daughter peers
18: **for** each *daughter peer* d $\in D_i$ **do**
19: query all peers $j \in A_d^i$ for $c_{jd}P_j$
20: **if** peers $j \in$ to another HR **then**
21: send queries to RRM to transfer them to indicated HR.
22: **end if**
23: **repeat**
24: Compute
$$t_d^{k+1} = (1-a)(c_{1s}t_1^k) + (c_{2s}t_2^k) + ... + (c_{ns}t_n^k) + aP_j;$$
send $(c_{dj}t_s^{k+1})$ to all peers j $\in B_d^i$
25: **sent to RRM** all $(c_{dj}t_s^{k+1})$ for other HR's.
26: wait for all peers $j \in A_i^d$ to return $(c_{jd}t_s^{k+1})$;
27: wait for all peers $j \in A_i^d$ from other HR's to return $(c_{jd}t_s^{k+1})$;
28: **until** $|t_s^{k+1} - t_s^k| < \varepsilon$
29: **end for**
30: **end for**

hierarchical structure [5]. Thus, at each level, pre-trusted peers are assumed to be involved in the computation of trust and reputation. The algorithm computes the local trust of peers by using score managers of each peer to keep the score, and then aggregating all trust scores of a peer to determine its global reputation in the network. Because this algorithm works in a hierarchical environment, transactions made outside a home ring, are used to bring the scores of all peers from other home rings to local ring in order to compute the global reputation of a local peer.

The number of local trust values reported by a peer i is limited because a network may have a huge number of peers. The algorithm adds a variable $Rpeer_j$ to limit the number of remote peers that can send scores of a peer i located in a local ring. This algorithm allows a peer to have its score managers only locally

to optimize the algorithm and to reduce traffic in the network. Peers outside of a local ring can report scores of peers in another ring to their superpeers, which will then transfer scores to score managers of peers into their corresponding HR's. We also assume that a group can only interact with a limited number of other groups. The number of hops from one ring to another rings for lookup purpose is determined and limited by the variable $hops_j$. Small rings (home rings) will execute the algorithm faster than one large flat ring; this allows the algorithm to converge faster. To compute the complexity of the algorithm, we need to take into account the local and remote computation of node scores. Essentially, local computation involves many peers than the remote computation due to restriction of the number of hops and remote peers that can send feedback for a peer in a local HR. Also, the system allows the use of cache to keep results of queries. When the algorithm is run for the first time, it determines pre-trusted peers, the number of hops it is permitted to use while sending a query out-site of a local ring. The algorithm is executed in $O(n^2)$. The idea in a hierarchical structure is to keep queries in the local ring. Finally, we assume that in the computation of the reputation of a node i, the number of HR's from which peers will send feedback after having performed transactions with node i must be limited to increase the performance of the algorithm, and the number of remote peers from other HR's to a local ring is limited to improve the performance of the algorithm.

5 Simulating Hierarchical Chord

In these experiments, we evaluate the performance of the redesigned EigenTrust in an hierarchical structured P2P network using Chord lookup service. We use existing experiment of flat EigenTrust and extend it to a hierarchical structured P2P. We also consider that a main ring is composed of superpeers that connect other nodes to form a ring, and that an m-bit *identifier* is attributed to each peer and each *key*. The model we use is based on a cloud service provider (CSP) that is made up of many data centre disseminated around the globe. Thus, each continent can be connected to the super ring by using a Superpeer called "***remote resource manager (RRM)***", a superpeer that connects two consecutive rings. We assume that when a flat network has 100 nodes, the entire hierarchical network will have 100 nodes. Then, we increase the number of nodes to 500, 1000 and 5000 respectively in both networks, and compare results. For the sake of clarity and fair comparison, we assume that both flat and hierarchical networks have the same number of nodes and that all rings in the hierarchical network have the same number of nodes. In the Hierarchical Chord simulation, we compare the fraction of download, the convergence speed and malicious collective for both flat and hierarchical EigenTrust. To capture the heterogeneity of the peers, we suppose that there are two categories of peers: Stable peers (for node 1 to node 10); Unstable peers for the other nodes. For the hierarchical organization, we select super-peers from a set of stable peers and we suppose that there is at least one stable peer in each group. For the organization, we choose nodes from the two categories randomly. Convergence is another aspect

of concern since nonlinearity may result in a large number of iterations and render the system inefficient. We choose to have the same fraction of malicious peers in both hierarchical and at Chord, in order to readily compare the fraction of download from peers.

6 Simulation Results

6.1 Setup

we used an open-source simulator called PeerfactSim [13]. PeerfactSim does not currently support hierarchical Chord. One of the tasks was to modify the source code with an implementation of the flat Chord P2P system structure and extend it to hierarchical Chord. A user can specify the length of the tree and the number of nodes.

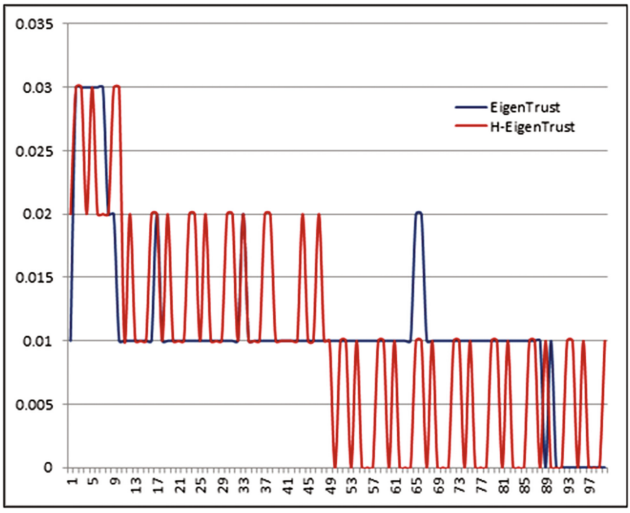

Fig. 2. Results of Fraction of downloads in Flat and Hierarchical EigenTrust for 100 nodes

We use a standalone computer with Intel(R) Core(TM) i5-3320M CPU @2.60 GHz, 64 bits, using 8.00 GB. For coding we use Eclipse IDE for Java Developers, Version: Mars.1 Release(4.5.1) and jdk1.8.0_91. The Windows 7 Professional operating system was used. For the experiments, we want to compare simulation results from a flat chord P2P system(with only one ring) to that of a hierarchical Chord P2P system. Then We compare and analyze experimental results.

6.2 Results

Figure 2 represents the load distribution results of the simulation when the network has 100 nodes. We can see that load distribution for the Hierarchical Chord

network is concentrated on nodes with higher stability in each level of the hierarchy. Conversely, in the flat EigenTrust network, load distribution is selecting data sources with more scattered patterns. We increase the number of nodes to 500 and 1000, to simulate malicious collectives in both algorithms. For 100 and 500 nodes, malicious peers represents 43% of nodes, while for 1000 nodes the number of malicious nodes represents 50% of the all nodes in the network. We limit the number of nodes to 16 in each ring. Figure 3 shows that, with more nodes, we can see that the percentage of inauthentic downloads increases very slightly compared to 100 nodes for EigenTrust. This proves that the hierarchical EigenTrust improved significantly in the hierarchical network. The hierarchical EigenTrust presents better performance that the flat EigenTrust, even when the percentage of malicious peers constitutes the half of the total number of peers in the network. EigenTrust downloads more files from malicious peers than hierarchical EigenTrust.

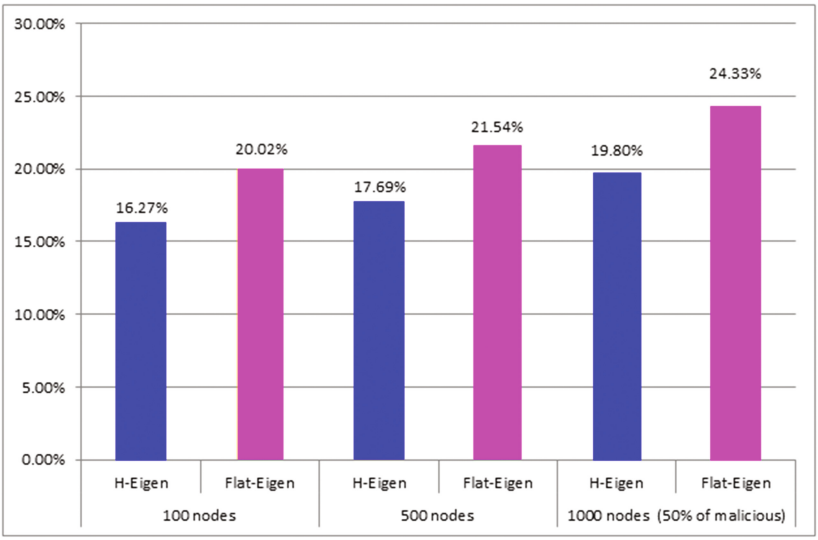

Fig. 3. Trust-based reduction of inauthentic downloads in hierarchical network with 100, 500 and 1000 nodes where a fraction of peers forms a malicious collective

Figure 4 represents the convergence speed of both algorithms for 100, 500, 1000 nodes. Results show that even hierarchical network has many rings, the convergence speed is close to that of flat network. With 1000 nodes, we can see that the flat EigenTrust converges after at most 5 iterations, while the hierarchical EigenTrust converges at most 6 iterations in the network with many rings. Even when the number of nodes changes, the converge still close.

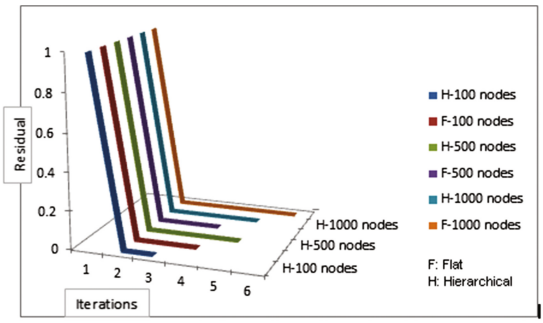

Fig. 4. Convergence speed of algorithms in hierarchical network

7 Conclusion

We have redesigned the EigenTrust trust and reputation algorithm that was designed for flat-structured P2P networks to be used in hierarchically structured P2P networks. We chose Chord to this end and we extended its DHT mechanism to a hierarchical structure where peers are assembled in groups recognized by a unique identifier. We simulated the new algorithm as well as the old algorithm, and we evaluated them in terms of load distribution, residual curl, and malicious collective downloads. We compared the measures obtained for both EigenTrust and its hierarchical version. The results revealed that the hierarchical trust and reputation algorithm achieved better performance than the flat algorithm and converged faster and proportionally to the number of rings. We can therefore conclude that the reduced number of nodes per ring and their organization in hierarchies helped to improve the performance of the Chord P2P system. In the future, as our simulation have been performed on a hierarchically structured Chord P2P network limited to a three levels, we intend to extend our results to a Chord structure with an arbitrary number of levels. Furthermore, we will look into how to extend this research to other trust and reputation systems found in the literature and which were designed around different trust and reputation models such as PowerTrust [16], NodeRanking [10] or Absolute trust [1]. The results of simulations using our simulator will then be compared to determine which algorithm outperformed all others in a hierarchical network environment. We can also apply this research to IoT settings, such as the fleet management setting used in the introduction to this paper as motivation, by building an IoT fleet management network where nodes are hierarchically organized. Further future work can be focused on the extension of EigenTrust to lookup services based on other existing hierarchical overlay structures such as BATON [6], and HD Trees [4].

References

1. Awasthi, S.K., Singh, Y.N.: Absolute trust: algorithm for aggregation of trust in peer-to-peer networks. IEEE Commun. Lett. **20**(7), 1345–1348 (2016)

2. Foster, I., Kesselman, C., Nick, J., Tuecke, S.: The physiology of the grid: an open grid services architecture for distributed systems integration. Technical report, Global Grid Forum (2002)
3. Garcés-Erice, L., Biersack, E.W., Felber, P.A., Ross, K.W., Urvoy-Keller, G.: Hierarchical peer-to-peer systems. In: Kosch, H., Böszörményi, L., Hellwagner, H. (eds.) Euro-Par 2003. LNCS, vol. 2790, pp. 1230–1239. Springer, Heidelberg (2003). doi:10.1007/978-3-540-45209-6_166
4. Gu, Y., Boukerche, A.: HD tree: a novel data structure to support multi-dimensional range query for P2P networks. J. Parallel Distrib. Comput. **71**(8), 1111–1124 (2011)
5. Hofstatter, Q., Zols, S., Michel, M., Despotovic, Z., Kellerer, W.: Chordella - a hierarchical peer-to-peer overlay implementation for heterogeneous, mobile environments. In: 2008 Eighth International Conference on Peer-to-Peer Computing, pp. 75–76 (2008)
6. Jagadish, H.V., Ooi, B.C., Vu, Q.H.: BATON: a balanced tree structure for peer-to-peer networks. In: Proceedings of the 31st International Conference on Very Large Data Bases, pp. 149–160 (2005)
7. Kamvar, S.D., Schlosser, M.T., Garcia-Molina, H.: The EigenTrust algorithm for reputation management in P2P networks. In: 12th International Conference on World Wide Web (WWW), p. 640 (2003)
8. Maenpaa, J., Camarillo, G.: Study on maintenance operations in a chord-based Peer-to-Peer session initiation protocol overlay network. In: IEEE International Symposium on Parallel Distributed Processing (IPDPS 2009), p. 19 (2009)
9. Montresor, A., Jelasity, M., Babaoglu, O.: Chord on demand. In: Proceedings - Fifth IEEE International Conference on Peer-to-Peer Computing, P2P 2005, vol. 2005, pp. 87–94 (2005)
10. Pujol, J.M., Sangesa, R., Delgado, J.: Extracting reputation in multi agent systems by means of social network topology. In: Proceedings of the First International Joint Conference on Autonomous Agents Multiagent Systems Part 1, pp. 467–474 (2002)
11. Ratnasamy, S., Francis, P., Handley, M., Karp, R., Shenker, S.: A scalable content addressable network. In: Proceedings of the ACM SIGCOMM, vol. TR-00-010, pp. 161–172 (2000)
12. Rowstron, A., Druschel, P.: Pastry: scalable, decentralized object location, and routing for large-scale peer-to-peer systems. In: Guerraoui, R. (ed.) Middleware 2001. LNCS, vol. 2218, pp. 329–350. Springer, Heidelberg (2001). doi:10.1007/3-540-45518-3_18
13. Stingl, D., Gross, C., Ruckert, J., Nobach, L., Kovacevic, A., Steinmetz, R.: PeerfactSim.KOM: a simulation framework for peer-to-peer systems. In: Proceedings of the 2011 International Conference on High Performance Computing and Simulation, HPCS 2011, pp. 577–584 (2011)
14. Stoica, I., Morris, R., Karger, D., Kaashoek, M.F., Balakrishnan, H.: Chord: a scalable peer-to-peer lookup service for internet applications. In: Proceedings of the 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, pp. 149–160 (2001)
15. Xiong, L., Liu, L.: PeerTrust: supporting reputation-based trust for peer-to-peer electronic communities. IEEE Trans. Knowl. Data Eng. **16**(7), 843–857 (2004)
16. Zhou, R., Hwang, K.: PowerTrust: a robust and scalable reputation system for trusted peer-to-peer computing. IEEE Trans. Parallel Distrib. Syst. **18**(4), 460–473 (2007)