

Comparison of P2P Social Network Architectures

Francisco de Asís López-Fuentes, and Salvador Balleza-Gallegos

Departament of Information Technology,
Universidad Autónoma Metropolitana – Cuajimalpa (UAM-C),
Mexico City, Mexico
flopez@correo.cua.uam.mx

Abstract. In recent years, online social networks have become very popular. Online social networks (OSN) have been useful to find former classmates or to improve our interaction with friends. Thus, a huge amount of information is generated and consumed by millions of people in these types of networks. Currently, most popular online social networks are based on centralized servers, whom are responsible management and storage all the information. Although the benefits of social networks are well known to most of us, these networks still have many challenges such as scalability, privacy or dependence. In this work, we evaluate P2P infrastructures as alternative platform for developing decentralized online social networks. In particular, we simulate and evaluate a specific social network called Tribler on two types of P2P architectures: super-peer and distributed. Our results shown that OSN based on super-peer reaches best download time than OSN based on a distributed P2P architecture.

Keywords: P2P architectures, on-line social networking, broadcast.

1 Introduction

Online social networks have become more and more popular every day, and many users use these networks to find former classmates or interact with friends. Social networks are also used to establish new relationships with other users, organize events with friends, promote personal work, share multimedia contents, publish comments or broadcast news. Today popular online social network are centralized and based on the client-server paradigm. This centralized structure introduces several drawbacks in the online social networks such as storage dependence, scalability, serious bottleneck, point of failure, security threats and privacy locally [13], [6]. Decentralized online social network infrastructures have been proposed to deal with these limitations. In this scenario, Peer-to-Peer (P2P) networks have emerged as a promising distributed information management platform. A P2P network is a distributed network formed by a group of nodes, which build a network abstraction on top of the physical network, known as an overlay network.

In a P2P system each peer can work both as a server and as a client at the same time. An important advantage of P2P networks is that all available resources such as processing, memory and bandwidth are provided by the peers [15]. Therefore, when a new peer arrives to the P2P system the demand is increased, but the overall capacity too. This is not possible in a client server model with a fixed number of servers. P2P paradigm allows that a distributed platform distributes its load and tasks on the participating peers. Traditionally, P2P networks have been mainly classified in two categories: media distribution and collaborative work, but social aspects such as interaction through applications for fun, publishing of messages visible for each other, updating the user status, advertising events or other interests to a group, etc. are not emphasized [13]. Nevertheless, the way how people connect with each other in social networks keeps a certain similarity with how P2P networks operate. Social networking sites allow users to find persons with similar interests, location or organization, addition to interaction with friends one already has [13]. Additionally, in a decentralized online social network users keep control of their data and can use the social network also locally, without Internet access.

The rest of this paper is organized as follows. In Section 2 we discuss related work, and in Section 3, we present our methodology and main components of each P2P infrastructure to be implemented. Then, we realize our implementation and simulations for Tribler in each P2P infrastructure in Section 4. In Section 5, we evaluate the performance of Tribler in both P2P infrastructures and analyze the results obtained from each. Finally, in Section 6 we present our conclusions and ideas for future work.

2 Related Work

Nature social connection between the human behind the computer and the large-scale P2P networks have found a fertile research area during the last years, and several P2P social networks such as Diaspora [1] [4], DECENT [3], Prometheus [5], PeerSON [2], and Tribler [6] have been proposed. Diaspora is a real-world Internet-scale decentralized online social network, and it was launched on November 2010 [4]. Diaspora has been mainly design to preserve the user privacy. A Diaspora network is formed by independent and federated servers, which are managed by individual users. The users' profiles are hosted in these federated servers, however, users can decide for themselves their preferred servers to store their information. In Diaspora, users can have a total control of their data by maintaining their own servers. Jahid et al. [3] present a modular architecture design for online social networks called DECENT, which employs a distributed hash table to store user data and features cryptographic protections. These features allow that this online social network can offer confidentiality, integrity, flexibility attributes polices as well as a fast revocation. In this way, user's data and their social relationship are not visible to unauthorized users.

Data availability in this social network is based on data replication and authentication of updates. Prometheus is introduced in [5] as a P2P service that collects and manages social information from multiple sources which are refereed as social sensors. Collected information is exposed through interface in order to implement non-trivial social inferences. This network employs user defined policies to control the access to social data. Social sensors are user applications used to collect information about interaction with others users via different devices or the web. Prometheus process this collected information in order to create a decentralized, directed and labeled multi-edged graph, which reports users who are connected and how they interact in social network. Another online social networks based on P2P paradigm is Tribler [6], which is a research project developed in the Delft University of Technology and Vrije Universiteit Amsterdam. Tribler runs over BitTorrent and exploits social phenomena to content discovery and download cost. Tribler is free software with a defined architecture and is currently available for Windows, Linux and Mac. Based on these findings, we select this social network as the online social network to be evaluated on two different P2P infrastructures. Performance of both infrastructures is evaluated in terms of download time. Buchegger et al. [2] propose PeerSON, which is a social network that addresses two limitations in the online social networks: privacy issues and the requirement of Internet connectivity for all transactions. To solve these problems the authors use encryption and access control coupled with a peer-to-peer approach. The various features provided by OSNs can broadly be classified into the categories of social link establishment, digital personal spaces, and means of communication [2]. These categories are represented in PeerSoN. To reach this goal, PeerSoN has a two tiered architecture. Logically, one tier serves as a look-up service. The second tier consists of peers and contains the user data, such as user profiles. PeerSoN currently uses a Distributed Hash Table (DHT) for the look-up service.

In this paper, we simulate an online social network on two different P2P infrastructures. To reach this goal we select Tribler as our reference social network to be implemented. On the other hand, Kademia [12] and BitTorrent [7] are selected as the P2P architectures over which Tribler has been deployed.

3 Background

In this section we give a briefly description of Tribler and the P2P architectures used in our evaluation. We have selected these P2P architecture because Kademia is a decentralized P2P infrastructure, while BitTorrent is hybrid P2P system based on super-peer. Tribler is an online-social network based on the BitTorrent protocol.

BitTorrent is a protocol designed for exchange of large P2P files on Internet. In a BitTorrent network a central server serves as core and manages bandwidth resources, routing and communication between nodes without knowing the identity of each node and without storing any information. Thus, the server will not share files of any kind to any

node and reduces the overload impact by distributing large files. In a BitTorrent network several basic computers can collaborate to efficiently distributing files to many requesting clients. A user uploads a file by creating a small torrent descriptor file and distributes it using conventional mechanisms as web o email. A BitTorrent network is formed by several actors and components such as peers, leechers, seeders, trackers and swarm. All users connected to the BitTorrent network are called peers. In this context there two types of peers: seeders and leechers [9], [10]. Seeders are users who have a file without consider if they were the original creator or they are just who published it. On the other hand, users that over time their behavior consists of downloading more files, than sharing it are called leechers. A tracker is responsible for the communication between peers, because it keeps a global registry of all the downloaders and seeds of the corresponding file [8]. Initially, clients need to communicate with the tracker to identify peers from where they can download a file. Once started downloading a file, communication service between peers can continue without a tracker. However, clients can continue to communicate with the tracker in order to obtain statistics from new peers. Finally, a swarm is formed by all peers who share a torrent. Swarming allows tolerating massive flash crowds for popular content [10].

On the other hand, Kademia [12] is a distributed hash table protocol designed for decentralized P2P networks. Kademia is deployed as a virtual network on an existing LAN/WAN network or Internet and its topology is based on the XOR metric. This metric is use to calculate distance between points in the key space [12]. Several benefits are derived from this novel approach such as participants can receive lookup queries form exactly the same distribution of nodes hosted in their routing tables or that they can send a query to any node within an interval. These facts allow that participants select routes based on latency or even send parallel asynchronous queries. Compared with previous P2P protocols, Kamelia's nodes send a reduced number of configuration messages to learn about each other. Collected knowledge is used to provide flexible routing via low-latency paths. In Kademia, the timeout delays generated from failed nodes are avoided by using parallel asynchronous queries. Each Kademia's node is identified by a 160-bit node ID. Kademia protocol consists of the following RPCs: PING, STORE, FIND NODE, and FIND VALUE. These procedures allow to specify the network structure, regulates communication between nodes and exchange of information. The nodes communicate is realized via the UDP protocol.

We simulate Tribler on both P2P infrastructures described above. Here, we describe the main characteristics of this on-line social network. Decentralization, availability, integrity, cooperation and network transparency are the main challenges in P2P file sharing systems [6]. To deal with these challenges, Tribler has been proposed and implemented on top of the BitTorrent protocol. To reach this goal the authors have only realized modifications and extensions to the existing BitTorrent client software. Tribler architecture involves different concepts such as social groups, megacaches, taste buddy, collaborative download, list of friends, geo-localization, recommendation and peer similarity. Many of these components are integrated together via a user interface, which is a key and critical

part of tribler. User interface facilitates the formation of social groups. Bootstrapping is a task to find other peers in a P2P system. In BitTorrent this task is realized by a tracker. However, to solve this problem, Tribler uses super-peers and overlay swarm through an epidemic protocol called Buddycast. Super-peers are contacted by a Tribler peer to obtain an initial list of other peers in the system, while overlay swarm is used for content and peer discovery. Social phenomenon concept in Tribler has been implemented in the social networking module, which is responsible for storing and providing information related to social groups such as the group members, used IP numbers, etc.

In Tribler, megacaches are used as local storage to store each piece of context information received by a peer that is relevant to it based on its interest and tastes. This information is exchanged within social groups using the Buddycast protocol. As we have previously indicated, Tribler uses the BitTorrent protocol for downloading files. However, to achieve a significant increase in file download speed the BitTorrent Engine uses a collaborative downloader module which exploits idle upload capacity of online friends [6]. This collaborative module is based on a protocol called 2Fast [14], which helps to improve some limitations in the original BitTorrent protocol. In 2Fast, a peer can participate in a collaborative download taking one of two roles: collector and helpers. A collector peer is interested in obtaining a complete copy of a particular file, while a helper peer can provide assistance to a collector peer during a file download. Both peers work similar as in the BitTorrent protocol, however, a collector peer can optimize its download performance by dynamically selecting the best available data source from a set of helper peers [6]. In this protocol, a helper peer works under an altruist approach, that is to say, a helper peer collaborates without requesting anything in return.

4 Implementation and Simulation

The main thing in the functioning of the social network Tribler is file sharing. This system is based on a BitTorrent client which has the function of sharing and download a file on the social network. Our Tribler simulation is implemented in the Peersim simulator [7]. We have used this network simulator because it supports dynamicity and extreme scalability. Peersim is written in Java language and it can be used to simulate small and large-scale P2P systems. This simulation tool also allows us to measure the communication between the same nodes and communication time between them. Our online social network simulation uses the BitTorrent and Kademlia prototype developed by the Trento University for the Peersim simulator [16] [17]. Our implementation uses Java as programming language for developing the communication protocols. Peersim is composed of two simulation engines: the cycle-based model and a more traditional event-based model.

First, we implement the Tribler social network with BitTorrent on the PeerSim simulator. In this scenario, a .Torrent file to be shared by a peer must be published in a

specific directory, which can be recognized by the system as a file uploaded to the network to be shared. This file contains file name, file size, hash information and the tracker's address. A user must be connected to the Internet to be referenced to a tracker and he or she can be attached to the social network with other active users connected to the network, who share either an entire file that is of interest or part of the file still can easy downloading. To realize our Tribler implementation in BitTorrent, we modify the BitTorrent protocol based on cycles and event developed by the Trento University [16]. Main modifications made to the BitTorrent protocol to implement the Tribler social network in PeerSim are the following. First, all classes of the PeerSim simulator are decompiled. After this, GeneralNode.java class is specifically revised in order to fix the reputation rates. Here we also implement the methods to manipulate the types of reputation. These methods (good or bad reputation) are appended to BitTorrent.java class. Good reputation method is activated if peer is sharing the file. Otherwise, bad reputation method is activated if file is blocked for the requesting peer. In this same class, when the tracker is activated, it reviews the state of all nodes and the nodes that are in a failed status are not added to the system. Constantly, BitTorrent.java class activates the reputation method which detects if a node has a reputation 0. If this is the case, then this node is deleted from the network in order to realize faster downloads and reject future requests from this non-cooperative node. To observe and control the simulation results, we modify the statistics output in the BTObserver.java class.

Our reputation strategy used to implement Tribler in Kademia is different to the strategy used in BitTorrent, due mainly to the difference of operation of both protocols. Reputation levels in BitTorrent are based on shared pieces, while in Kademia reputation levels are measured during a direct download from a node. Files only are downloading from nodes with high reputation (2 in a scale 0 to 5) to reach a reliable download. We add reputation levels and methods in the Kademia protocol developed by the Trento University [16], specifically in the GeneralNode.java class. We add the methods for good and bad reputation in the Kademia Protocol.java which is the main class of protocol. To simulate a file is given a date array in each node in the GeneralNode.java. Control for all the output variables used by the nodes is defined in the classes KademiaObserver.java and Kademiaprotocol.java.

5 Evaluation

In order to evaluate Tribler performance on Kademia and BitTorrent, we build a BitTorrent network with the same number of nodes as Kademia and with the same file to be downloaded. Our experiments evaluate the nodes with reputation levels. We consider 2500 nodes in both P2P networks. 15% of these nodes are seeders nodes while 85% are leechers nodes. The file size to be distributed is 100 MB. Initial reputation is random. If node's reputation rate is 0, then it removed from the system. Tribler's performance with

Kademlia protocol is shown in Figure 1, while Tribler's performance with BitTorrent protocol is shown in Figure 2.

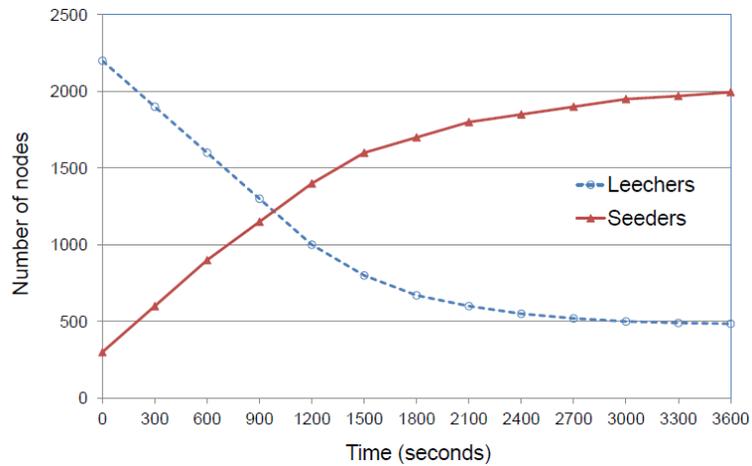


Fig. 1. Tribler's performance based on Kademlia protocol

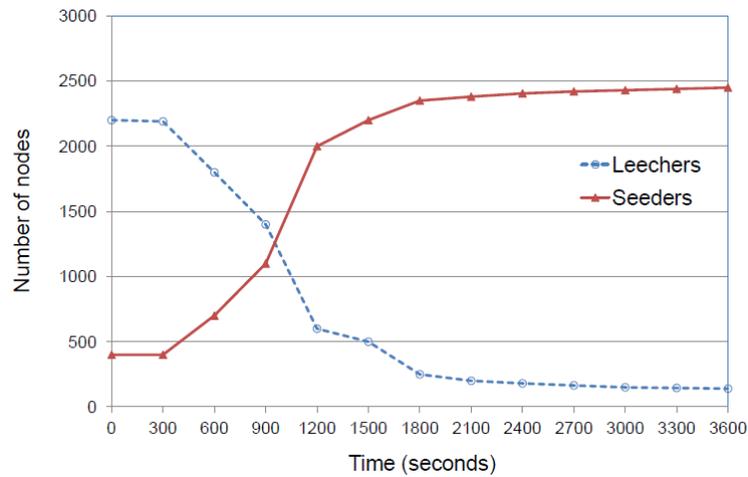


Fig. 2. Tribler's performance based on BitTorrent protocol

We can see that iteration starts with 2200 nodes because simulation removes nodes with low reputation or failed connection. We can observe how over time the number of leecher peers is decreased while the number of seeder peers grows more quickly. Non-

cooperative nodes reduce their reputation level while cooperative nodes continue active in the system.

On the other hand, in figure 3 is compared the number of leechers nodes in Tribler for both P2P infrastructures. Initially, we can see that the number of leechers in BitTorrent is higher than in Kademlia network. However, as time goes, leecher nodes in BitTorrent network are less than in the Kademlia network.

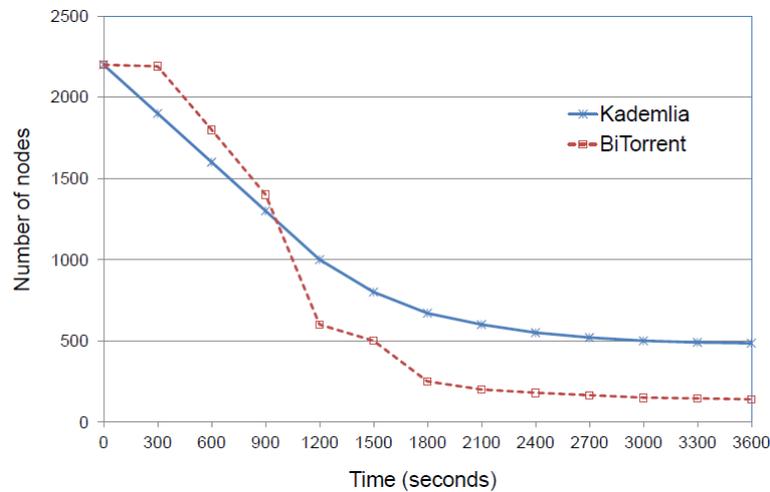


Fig. 3. Comparison of number of leechers in Tribler for both P2P infrastructures

Figure 4 compares the number of seeder nodes in both P2P networks. Initially, both infrastructures have a similar number of seeder nodes, but as time goes the number of seeder nodes in BitTorrent is increased in a large proportion compared to Kademlia. This means that in the BitTorrent network, a node can quickly retrieve the requested file, and so can become a seed node. In contrast, this conversion from a leecher node to a seed node requires more time in the Kademlia network. Thus, BitTorrent has more available seeder nodes from which download the test file than Kademlia. This fact allows that BitTorrent presents a best performance to search and download a file compared to Kademlia.

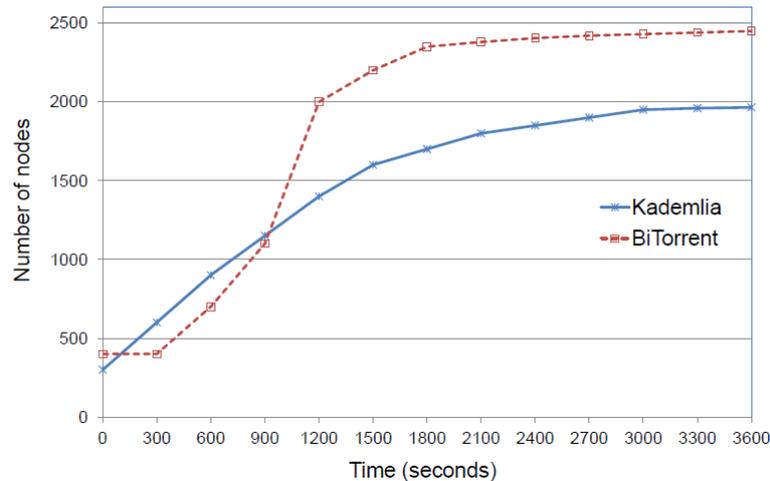


Fig. 4. Comparison of number of seeders in Tribler for both P2P infrastructures

We also compare the impact of the non-cooperative for Tribler in both types of P2P infrastructure. The same file of 100 MB is downloaded from both P2P networks. Figure 5 shows these results. We can see that Tribler based on BitTorrent can download the file faster than Tribler based on Kademlia, with an equal number of non-cooperating nodes. This is because as time passes Tribler based on BitTorrent eliminates non-cooperative nodes, which improves the system performance. Although the protocol ignores uncooperative nodes, due to their distributed topology each requesting node will take longer to find a new node from where to continue downloading the content. This causes the file download to be slower.

6 Conclusions

Online social networks have become very popular during the last years. However, centralized structure used by the online social networks introduces several drawbacks such as storage dependence, bottleneck, only point of failure, security threats and privacy locally. In this paper, we evaluated a social network called Tribler for two different P2P infrastructures: Super-peer (hybrid) and distributed. BitTorrent is our scheme used to simulate the super-peer infrastructure, while the Kademlia protocol is used to simulate the distributed infrastructure. Reputation strategies are integrated in both protocols as a way to isolate the non-cooperative nodes from the system. Our experiments were realized using the Peersim simulator and both P2P infrastructures were evaluated in term of delivery time. Our results show that Tribler with BitTorrent presents a best delivery time

than Tribler with Kademia. However, we believe that both infrastructures could have a different performance during a security threats.

As future work, we intend to extend our Tribler evaluation with both P2P infrastructures to scenarios with different types of security threats. More common security threats in P2P networks are Sybil and Eclipse attacks.

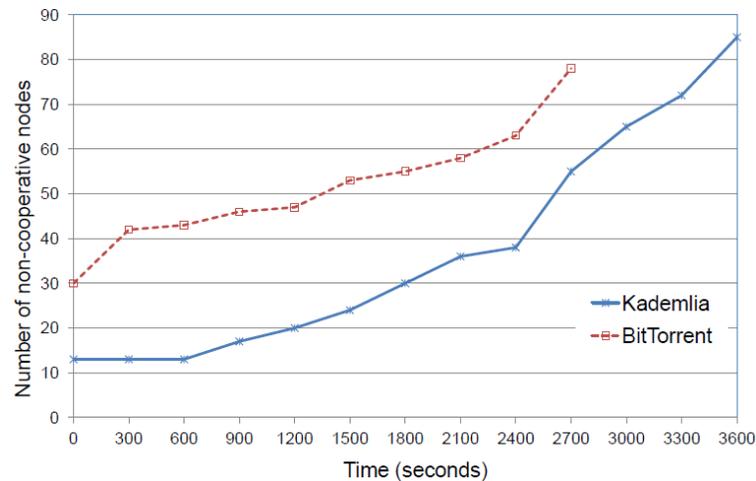


Fig. 5. Performance of P2P infrastructures in presence of non-cooperative peers

References

1. Diaspora. <http://ljoindiaspora.com/> (2014).
2. Buchegger, S., Schieberg, D., Yu, L. H. and Datta, A.: Peerson: P2P social networking - early experiences and insights. In: 2nd ACM Workshop on Social Network Systems, pp. 46-52, Nuremberg, Germany, (2009).
3. Jahid, S., Nilizadeh, S., Mittal, P., Borisov, N. and Kapadia, A.: DECENT: A Decentralized Architecture for Enforcing Privacy in Online Social Networks. In: 4th International Workshop on Security and Social Networking, Lugano, Switzerland, (2012).
4. Bielenberg, A., Helm, L., Gentilucci, A., Stefanescu, D. and Zhang, H.: The Growth of Diaspora - A Decentralized Online Social Network in the Wild. In: Global Internet Symposium, pp. 13-18, Orlando, FL, USA (2012).
5. Kourtellis, N., Finnis, J. Anderson, P. Blackburn, J., Borcea, C. and Iamnitchi, A.: Prometheus: user-controlled P2P Social Data Management for Socially-aware Applications. In: 11th International Conference on Middleware, pp. 212-213, Bangalore, India, (2010).
6. Pouwelse, J. A., Garbacki, P., Wang, J., Bakker, A., Yang, J., Iosup, A., Epema, D. H. J., Reinders, M., Van Steen, M. R. and Sips, H.: TRIBLER: A Social-based Peer-to-peer System.

- In: *Journal Concurrency and Computation: Practice & Experience*. Vol 20(2), pp. 127-138, John Wiley and Sons Ltd, (2008).
7. Bohlen, B.: Incentives build robustness in Bittorrent. In: 1st Workshop on Economics of Peer-to-Peer Systems, Berkeley, CA, USA (2003).
 8. Pouwelse, J. A., Garbacki, P., Epema, D. H. J. and Sips, H.: The Bittorrent P2P file-sharing system: Measurements and analysis. In: 4th International Workshop on Peer-to-Peer Systems, Ithaca, NY, USA (2005).
 9. Kaune, S., Rumin, R. C., Tyson, G., Mauthe, A., Guerrero, C. and Steinmetz, R.: Unraveling BitTorrent's File Unavailability: Measurements and Analysis. In: 10th International Conference on Peer-to-Peer Computing, Delft, Netherlands, (2010).
 10. Menasche, D. S., Rocha, A., Li, B., Towsley, D. and Venkataramani, A.: Content Availability and Bundling in Swarming Systems. In: 5th International Conference on Emerging Networking Experiments and Technologies, Rome, Italy, (2009).
 11. Montresor, A. and Jelasity, M.: Peersim: A scalable P2P simulator. In: International Conference on Peer-to-Peer Computing, pp. 99-100, Seattle, WA, USA (2009).
 12. Maymounkov, P. and Mazieres, D.: Kademlia: A Peer-to-peer Information System based on the xor Metric. In: International Workshop on Peer-to-Peer Systems, Cambridge, MA, USA (2002).
 13. Buchegger, S. and Datta, A.: A case for P2P infrastructure for social networks – opportunities and challenges. In: 6th International Conference on Wireless On-demand Network System, pp. 161-168, Snowbird, UT, USA (2009).
 14. Garbacki P, Iosup A, Epema DHJ, Van Steen M.: 2Fast: Collaborative downloads in P2P networks. In: 6th IEEE International Conference on Peer-to-Peer Computing, pp. 23-30, Cambridge, UK (2006).
 15. Milojevic, D., Valogeraki, D., Lukose, R., Nagaraja, K., Pruyne, J., Richard, B., Rollins S., and Xu, Z.: Peer-to-Peer Computing. In: Technical Report HPL-2002-57(R.1). HP Laboratories, Palo Alto, CA, USA (2003).
 16. Frioli, F. and Pedrolli, M.: A BitTorrent module for Peersim. In: Università di Trento, (2008). Web site: <http://peersim.sourceforge.net> [seen: 01/08/2014].
 17. Bonani, M. and Furlan, D.: A Kademlia module for Peersim. In: Università di Trento, (2009/2010). Web site: <http://peersim.sourceforge.net> [seen: 01/08/2014].

