Computer Networks 54 (2010) 689–705

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Performance evaluation of a Kademlia-based communication-oriented P2P system under churn

Zhonghong Ou [a,b,*], Erkki Harjula [a], Otso Kassinen [a], Mika Ylianttila [a]

[a] Department of Electrical and Information Engineering, University of Oulu, Finland
[b] School of Computer Engineering, Beijing University of Posts and Telecommunications, China

ARTICLE INFO

Article history:
Received 13 May 2009
Received in revised form 17 September 2009
Accepted 30 September 2009
Available online 9 October 2009
Responsible Editor: N. Agoulmine

Keywords:
Communication-oriented
Churn
Distributed hash table (DHT)
Kademlia
Energy consumption

ABSTRACT

The phenomenon of churn has a significant effect on the performance of Peer-to-Peer (P2P) networks, especially in mobile environments that are characterized by intermittent connections and unguaranteed network bandwidths. A number of proposals have been put forward to deal with this problem; however, we have so far not seen any thorough analysis to guide the optimal design choices and parameter configurations for structured P2P networks. In this article, we present a performance evaluation of a structured communication-oriented P2P system in the presence of churn. The evaluation is conducted using both simulation models and a real-life prototype implementation. In both evaluation environments, we utilize Kademlia with some modifications as the underlying distributed hash table (DHT) algorithm, and Peer-to-Peer Protocol (P2PP) as the signaling protocol. The results from the simulation models created using Nethawk EAST (a telecommunication simulator software) suggest that, in most situations, a lookup parallelism degree of 3 and resource replication degree of 3 are enough for guaranteeing a high resource lookup success ratio. We also notice that, with the parallel lookup mechanism, a good success ratio is achieved even without the KeepAlive traffic that is used for detecting the aliveness of nodes. A prototype system that works in mobile environment is implemented to evaluate the feasibility of mobile nodes acting as full-fledged peers. The measurements made using the prototype show that, from the viewpoints of CPU load and network traffic load, it is feasible for the mobile nodes to take part in the overlay. Through energy consumption measurements, we draw the conclusion that in general the UMTS access mode consumes slightly more power than the WLAN access mode. Protocol packets with sizes of 200 bytes or less are observed to be the most energy efficient in the UMTS access mode.

1. Introduction

Peer-to-Peer (P2P) computing can be defined as the sharing of resources, e.g. CPU processing capacity and network bandwidth, and services, e.g. authentication and redundant storage, by direct exchange among all the participants of the overlay network. It has attracted a great deal of attention from the industry, academia and media since the date of its birth in the late 90's, represented by Napster. P2P systems can be classified into different categories according to their various application environments [1]: communication and collaboration, distributed computation, internet service support, database systems, and content distribution. In this article, we focus on the communication-oriented P2P systems that provide signaling for facilitating direct communication between participating peers. For this purpose, the Peer-to-Peer Protocol (P2PP) [2] is adopted as the underlying signaling protocol. P2PP was one of the former candidates for the signaling

* Corresponding author. Address: Department of Electrical and Information Engineering, University of Oulu, Finland. Tel.: +358 8 553 2804; fax: +358 8 553 2534.
E-mail addresses: tommy@ee.oulu.fi, firstname.lastname@ee.oulu.fi (Z. Ou).

1389-1286/$ - see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.comnet.2009.09.022

protocol in the Peer-to-Peer Session Initiation Protocol (P2PSIP) working group [3], and was merged into REsource LOcation And Discovery (RELOAD) [4], the signaling protocol selected by the P2PSIP working group. As an ever-increasing number of mobile terminals are capable of accessing the Internet through access networks such as Universal Mobile Telecommunications System (UMTS) and Wireless Local Area Network (WLAN), our primary effort with P2PP is to study its feasibility for mobile environments that are characterized by intermittent connections and limited bandwidth. Accordingly, the churn effect is one of the primary problems that need to be studied.

To make the evaluation of the churn effect clearer, we firstly introduce some definitions:

- *Churn:* the phenomenon of the continuous arrival and departure of participating peers of a P2P overlay.
- *Graceful leaving:* a departing peer notifies its upcoming departure to its neighbors and transfers the resource items it stored to some other peer(s).
- *Ungraceful leaving:* a departing peer neither notifies its departure nor transfers resource items to its neighbors. This occurs upon certain failure events of peers, e.g. an abrupt network disconnection or running out of power.
- *Parallelism degree:* the number of requests that an initiating peer sends in parallel to its routing neighbors. Parallel requests take place in the resource lookup process.
- *Replication degree:* the number of instances of each specific resource item stored in the P2P overlay. Replication takes place in the resource publishing process.

For the purpose of effective performance evaluation, we build a simulation model based on P2PP using Kademlia [5] as the underlying Distributed Hash Table (DHT) algorithm. We evaluate the effect of different levels of churn and other parameters, including parallelism degree, replication degree, type of leaving (graceful or ungraceful), and Kademlia's *k*-bucket size (*k*-value), to the lookup success ratio. We also evaluate the mean network traffic load, i.e. bytes/node/s, and the mean number of messages sent and received per time unit, i.e. messages/node/s. To analyze the feasibility of mobile nodes acting as full-fledged peers, we have implemented a P2PP prototype that works both in Symbian and Linux Operating Systems (OS). Specifically, we study the CPU load, network traffic load and battery consumption of mobile devices taking part in the P2PP overlay through UMTS and WLAN access networks.

The contribution of this article is as follows:

- We present our simulation models and a prototype implementation of P2PP. The former is based on the Nethawk EAST software (a test automation and traffic generation tool to simulate telecommunication networks), and the latter works in real-life network environments.
- With the simulation models and the associated results, we find the feasible values for parallelism and replication degrees, depending on different overlay parameters.
- With measurements made using the prototype implementation, we make observations about the CPU load, network traffic load and power consumption, depending on overlay parameters, in UMTS and WLAN access modes.

The remainder of the article is structured as follows: Section 2 gives the related work concerning three aspects; Section 3 introduces the system architecture and P2PP in general; Section 4 presents the metrics for the performance evaluation and the associated simulation results in detail; in Section 5, the prototype and the associated CPU load, network traffic load and battery consumption of mobile phones are stated. In Section 6, we conclude the article.

2. Related work

In this Section, we present related work, concerning communication-oriented P2P systems, evaluations of churn in Kademlia-based P2P networks, and power consumption of mobile devices participating P2P networks.

2.1. Communication-oriented P2P systems

Communication-oriented P2P systems aim at facilitating direct, usually real-time, communication and collaboration among P2P network participants. Examples of such systems include Jabber [6] and Skype [7]. The former is an open alternative to closed instant messaging (IM) and presence services [6], while the latter follows a closed source and proprietary design [7].

P2PSIP is an open standard mainly designed for communication-oriented P2P systems. The standardization process of P2PSIP is still in progress [3], and a great deal of research effort has been focused on it. Wauthy et al. [8] demonstrated a distributed SIP Proxy/Registrar based on DHT and showed that P2PSIP could be a real option for large, decentralized deployments. Matuszewski et al. [9] presented an implementation of a mobile P2PSIP Voice over Internet Protocol (VoIP) application and measured the registration, address discovery and call set-up delays in 3G and WLAN access networks. Their measurements showed that registration and call set-up delays, as well as the overhead traffic, did not impose significant restrictions on the commercial implementation of a server-less mobile VoIP service. Kokkonen et al. [10] presented a P2PSIP implementation based on P2PP [2] and run in mobile networks. They demonstrated the establishing of SIP multimedia sessions in the presence of few or no centralized servers. Baumgart [11] put forward a distributed name service for P2PSIP. The paper proposed a two-stage name resolution mechanism to efficiently handle the frequent IP address changes. Martinez-Yelmo et al. [12], in their turn, introduced a two-tier hierarchical DHT overlay network to interconnect sub-overlays. The routing performance and routing state of the architecture based on the Kademlia algorithm was analyzed in the absence of churn, however. Bryan et al. [13] gave a theoretical analysis on how security and routing schemes could influence the integrity, scalability and performance of a P2PSIP communication system.

The most relevant study from our viewpoint is [14], which presented a P2PP prototype implementation based on the Chord [15] algorithm. The associated functionality, performance and real-world applicability was proved by local single-machine simulation and real-life measurements from PlanetLab (a test-bed for distributed systems research consisting of a cluster of computers connected globally). The size of both the PlanetLab overlay and local simulation overlay was 60 nodes for most of the measurements. The main differences of [14] and the work presented in this article are: (1) our work was based on Kademlia, while [14] was based on the Chord; (2) we evaluated the effect of churn in detail, whereas [14] excluded it; and (3) we measured complete test sets, which made the results more convincing than [14] that evaluated only a small number of test sets.

2.2. Churn in Kademlia-based P2P networks

There are two approaches to analyze the effect of churn on the associated DHT algorithms. One is to utilize the widely-deployed real systems, e.g. Kad, Gnutella, to analyze the peer behavior; the other is to utilize mathematical analysis, simulation and prototypes.

For the former approach, there are several studies. Steiner et al. [16] presented the peer behavior, e.g. the total number of peers online and their geographical distribution, by crawling a real system, i.e. Kad, continuously for six months. They found that the session lengths had a "long tail" distribution, with sessions lasting as long as 78 days. Based on the same data as [16], Carra et al. [17] analyzed the probability of resource lookup success in Kad by utilizing the reliability theory (a theory that studies the reliability of the systems that are composed of separate units, which have their own reliability). In Refs. [18,19], through an empirical study of three widely-deployed P2P systems, i.e. Gnutella, BitTorrent and the Kad system, Stutzbach et al. found that the overall dynamics of peer participation were similar across these three systems. In Ref. [20], Stutzbach et al. investigated the effect of parallel lookup on the efficiency in Kad, and found that a lookup parallelism degree of 3 was a good choice for the current Kad network, without significantly increasing the bandwidth for routing maintenance.

For the latter approach, several studies have also been conducted. Li et al. [21,22], for example**,** formulated a unified performance versus cost (PVC) framework to compare the effects of different protocol features and parameter values for evaluating different DHTs. By simulating and analyzing several DHT algorithms, i.e. Kademlia [5], Chord [15], Tapestry [23], etc., they showed that large routing tables with infrequent stabilizations combined with parallel lookup achieved better performance compared to other approaches. Wu *et al.* [24] conducted an analytical study concerning the effects of the lookup strategy, lookup parallelism and resource key replication on the DHT lookup performance under churn. Binzenhöfer et al. [25], in their turn, analyzed the performance and robustness of Kademlia and presented some modifications regarding the search efficiency and overlay stability in the presence of churn. The exponential distribution was utilized to model the online and offline time. Finally, in Refs. [26–30], the authors addressed the issues of DHT performance under churn in a more general sense.

2.3. Power consumption of mobile devices in P2P networks

Gurun et al. [31] claimed to be the first work to examine P2P protocols from the energy consumption perspective (*power consumption* and *energy consumption* are used interchangeably hereinafter). They performed a study of energy consumption by utilizing a P2P chat application named Chimera on a PDA device. Their results showed that it was feasible to deploy lightweight P2P applications on low-power devices, e.g. PDA or mobile phones. Nurminen et al. [32] measured BitTorrent energy consumption on handheld devices and showed that P2P content sharing on handhelds was feasible from the energy consumption point of view. However, only the measurement results from the WLAN network connection based on the IEEE 802.11g standard was presented in Refs. [31,32]. Kelényi et al. [33] carried out energy consumption measurements by connecting a mobile client to a widely-deployed DHT named MainLine BitTorrent DHT. Their results showed that using a mobile phone as a full-peer was feasible only for a couple of hours due to the high power consumption, while operating the nodes in client-only mode was a power-efficient alternative. In Ref. [34], the same authors proposed some enhancements to reduce traffic and energy consumption by selectively dropping partial incoming messages. Kassinen et al. [35] measured the energy consumption in a P2PSIP overlay on Nokia N95 devices in both WLAN and UMTS mode; however, the sending and receiving of messages was measured separately. In our previous paper [36], the feasibility evaluation of a communication-oriented P2P system being used in mobile environment was conducted through a prototype implementation. In this article, we extend the measurement of energy consumption to sending and receiving mode in parallel and measure complete test sets. Moreover, both simulation models and a real-life prototype implementation are utilized to evaluate the performance of the communication-oriented P2P system.

3. System architecture

This Section gives the description of the P2PP architecture in general and the Kademlia-based algorithm. The terms *node* and *peer* are used interchangeably in this article.

3.1. P2PP protocol

P2PP [2] is a binary protocol for creating and maintaining an overlay for participating nodes. It can support both structured and unstructured P2P protocols, e.g. Chord, Kademlia, Gnutella, etc. In our simulation model and prototype, we choose Kademlia as the fundamental DHT algorithm and make some modifications to it. It is noteworthy that the simulation model and prototype implementation can support other DHT algorithms as well, e.g. Chord,

Table 1
Message type and its functionality.

Message type	Functionality
Bootstrap	Return the IP address and port of a node already in the overlay
Join	Node joins the overlay
Publish	Publish a resource in the overlay
Lookup	Lookup a resource in the overlay
Exchange	Updating the routing table, sent periodically
KeepAlive	Detect the aliveness of the nodes and remove the stale routing items from the routing table, sent periodically
Leave	Notify a peer's routing neighbors about its leaving from the overlay
Transfer	Transfer resource items to another node in the overlay
ACK	Acknowledgement of the receiving of messages

without much effort. The following messages listed in Table 1 have been implemented.

The transport protocol utilized to carry the P2PP protocol is user datagram protocol (UDP). Therefore, the ACK message is used in our simulation model and prototype implementation to guarantee the reliability of message delivery. One transaction is made up of a request, a response and ACK messages. One point to be noted here is that, if the ACK message follows directly the response message, then the ACK message piggybacks onto the response message to save the network traffic. Furthermore, as we choose Kademlia as the underlying DHT algorithm, accordingly, the iterative routing mechanism (the preference of Kademlia) is used in our simulation and prototype implementation.

3.2. Kademlia-based algorithm

There are three categories of operations with regard to Kademlia, i.e. nodes joining and leaving, overlay maintenance and resource-related operations.

3.2.1. Nodes joining and leaving

In our simulation model and prototype implementation, there is one centralized bootstrap server that provides the associated bootstrap functionality. One node (named Node J) who wants to join the P2PP overlay contacts with the bootstrap server firstly to get the IP address and port of another node already in the overlay (named Node A). After that, Node J sends the join request to Node A. If Node A notices that it is the numerically closest node to Node J in the overlay by checking its own routing table, it returns the successful response and Node J joins the overlay successfully; otherwise it returns the routing information of the node that is numerically closest to node J from its routing table (named Node B). The same procedure is executed iteratively until Node J gets the information of the closest peer (named *target node*) and joins the overlay successfully. The target node checks its own resource table to find the resource items whose identifiers (IDs) are closer to the newly joined node (i.e. Node J). If any, then the target node transfers the resource items to the newly joined node by utilizing the Transfer message, and the joining process is completed. Herein, the exclusive OR (XOR) metric is used to define the distance.

If one node decides to leave the overlay, as mentioned in Section 1, in the *graceful leaving* mode, the leaving node transfers all the resource items it stores to its closest neighbor in the routing table and notifies its leaving to all its routing neighbors; herein, the leave message is used; in the *ungraceful leaving* mode, the node just leaves the overlay without any resource item transferring or routing notification operation to its routing neighbors. This mode emulates the abnormal situations, e.g. power off or abrupt network disconnection.

One point to be noted is that the node joining process is serial rather than parallel: each time, only one join request is sent, which is slightly different from the original parallel mechanism used in Kademlia algorithm. We implement the serial process in this article as it is common in the other DHT algorithms, e.g. Chord. In reality, except the parallel publish or lookup processes, all the other processes in this article are serial.

In this process, the bootstrap, join, leave and ACK messages are utilized.

3.2.2. Overlay maintenance

To decrease the effect of churn on the overlay stability, exchange and KeepAlive messages are implemented. A node in the overlay periodically sends an exchange request to one of its routing neighbors selected uniformly, randomly. Upon the receipt of the exchange request message, the requested node responds with an exchange response message, in which at most 15 unduplicated routing items (the maximum number of routing items by the limitation of the UDP packet size) are included that are also selected uniformly, randomly from its routing table. If there are less than 15 routing items, the requested node responds with all the routing items it has in its routing table. The case in which each exchange response message includes 10 routing items is also evaluated in Section 4 to provide comparative results with the aforementioned 15 routing items case. For each routing item received through the exchange response message, one additional KeepAlive request message is sent to check its aliveness. If and only if the routing item replies a KeepAlive response message and the associated *k*-bucket [5] of the requesting node is not full, the routing item is inserted in the associated *k*-bucket. Herein, *k*-bucket is a list having up to *k* entries for each bit of the node identifier of Kademlia: *k* is a system wide number, *k*-value that is used in the subsequent sections is the size of the *k*-bucket. If the routing item fails to reply a KeepAlive response message, or the associated *k*-bucket is full, then the routing item is dropped simply. By doing this, we can avoid including stale routing items into the routing table.

Furthermore, timestamps are used to label routing items and routing tables are also updated by other incoming messages, e.g. publish and lookup messages, besides the exchange response message. If an incoming message is received from a routing item already in the routing table, the old timestamp of the routing item is updated by the new one. If the incoming message is received from a routing item not existing in the routing table, the routing item will be inserted in the routing table or be dropped simply according to the aforementioned rules. For instance, assume node A receives a lookup request message from node B. If node B is already in the routing table of node A, then the timestamp of routing item B is updated to the current time; otherwise, the routing item B is inserted into the routing table of A (if the associated k-bucket of A is not full) or is dropped simply (if the associated k-bucket of A is full).

Besides the periodical exchange message, the nodes in the overlay also send the KeepAlive message periodically to detect the aliveness of their neighbors and to clear away stale routing items from their routing tables. Upon the receipt of a KeepAlive request message, a node just responds with a KeepAlive response message to prove its aliveness.

In this overlay maintenance process, exchange, KeepAlive and ACK messages are utilized.

3.2.3. Resource-related operations

There are two operations with regard to resource items, i.e. publish and lookup. There are also two approaches for each of the two resource operations, i.e. *parallel* and *serial approach*. For the parallel publish and lookup operations, they are the same as the conventional Kademlia. To provide comparison to the parallel approach, the serial publish and lookup approach are also implemented in our simulation model and prototype implementation. The serial approach is the same as the parallel one under the condition that the Lookup parallelism degree is one. One point to be noticed here is that, in the serial approach, if the publish or lookup message is sent to an offline node, the entire publish or lookup process will result in a failure without any re-trying to other nodes. The reason for this is that it is hard to decide which node is the second closest node to the resource item, as we just return one numerically closest routing item in each response message in the serial approach.

To keep the resources in the overlay up to date, the owner of the resource items also re-publishes them periodically. The republish process is exactly the same as the publish process. There is one timestamp attached to each resource item. Nodes check out their Store lists (the list for the stored resource items) at intervals. If the timestamp of some resource item exceeds the upper bound without any update, the resource item is considered as stale and is removed from the Store list. In this way, if one node leaves the overlay for a long period of time, the resources originally published by it are also removed from the overlay to keep the resources stored in the overlay updated. The upper bound for labeling one resource item as stale can be decided by the corresponding applications. In our simulation, it is the same as the mean online time of the overlay nodes.

In this process, the publish, lookup and ACK messages are utilized.

4. Simulation results

In this Section, we evaluate the performance of P2PP through simulations. The purpose of the evaluation is to verify the adaptability of P2PP to different levels of churn. The level of churn is determined by the mean session time, or mean online time, which follows the exponential distribution. A shorter mean session time means a higher level of churn, and vice versa.

4.1. Performance metrics

One goal of the DHT-based applications is that, if there is some key stored in the overlay, the request targeting at that key should be followed by a successful response containing the associated key-value pair with high probability. Therefore, the success ratio, or hit rate, is our primary performance metric in this article.

- *Success ratio*: the ratio of successfully finding the resource items already published in the overlay. In order to evaluate the traffic load of the protocol and verify its adaptability to mobile environments, another two performance metrics are also used in this article, i.e. the mean traffic load and the mean number of messages.
- *Mean traffic load:* the mean number of bytes sent and received in one second by each node, i.e. bytes/node/s.
- *Mean number of messages:* the mean number of messages sent and received in one second by each node, i.e. messages/node/s.

One point that helps to understand the metrics is that each single message, e.g. a join request message, is calculated twice in our evaluation as it is one outgoing message for the initiating node and one incoming message for the targeting node. In both ends, it needs CPU processing capability, causes network traffic load and consumes some energy. Therefore, it is calculated both on the initiating node and on the targeting node.

4.2. Simulation set-up

We utilize the NetHawk EAST software to simulate the abovementioned P2PP protocol. NetHawk EAST is a test automation and traffic generation tool to simulate the telecommunication networks. It has the functionality of supporting binary encoding and decoding that is suitable for simulating the binary formatted P2PP. We can precisely emulate the binary formatted messages of the protocol and make the traffic load as close to real life as possible.

We make use of a dedicated server to run our simulations. The hardware of the server is Xeon (TM) CPU 3.06 GHz, 3.05 GHz, 2.00 GB of RAM. The software environment is Microsoft Windows XP Professional Version 2002 Service Pack 2 and NetHawk_EAST_IMS_v2.0.1_U1.1.

Due to the hardware limitation and UDP buffer size, we can only simulate an overlay with 400 nodes. All the measurement results made in Section 4 are based on the 400 nodes overlay. We utilize the same churn model as in Refs. [19,23], i.e. nodes stay online and offline (crash and rejoin the overlay) for an exponentially distributed time interval. As we set the same mean online and offline time intervals for the overlay nodes, i.e. $m_{online} = m_{offline}$, around half of the 400 nodes, i.e. 200 nodes, stays online on average at any given moment. By changing the mean online and offline time intervals, we can change the associated churn levels. Smaller m_{online} and $m_{offline}$ values mean higher levels of churn. Each experiment runs for 2 h. The nodes use the same IP addresses and Node IDs when they rejoin the overlay after the offline time interval.

We use a dedicated bootstrap server that stays online throughout each experiment to bootstrap the overlay. In the *initiation* stage, the nodes join the overlay at a rate of 2 nodes/s. After 200 nodes have joined the overlay, a *stabilization* interval of 200 s is used to fill their routing tables with proper routing items from the overlay. Then the *churn* stage is started, the online nodes leave the overlay (after their online time expires) and the offline nodes join the overlay (after their offline time expires), and so on until the end of the experiment. The data shown in the subsequent Sections are collected within the *churn* stage.

For the resource-related operations, i.e. publish and lookup, a database is utilized to record the resource items already published in the overlay. After the owner of one resource item leaves the overlay, the resource item will be removed from the database after the mean online time (as mentioned in the Section 3.2.3). For the Lookup operation, nodes only look up resource items from the database to make the success ratio measurement meaningful. We make use of the MD5 algorithm to generate the Node IDs and resource IDs that both have a length of 128 bits in our simulation.

To increase the credibility of our results [37], the 95% confidence level is utilized in the subsequent Sections where appropriate.

4.3. Simulation results

To make the evaluation clearer, we use the following notations and selected values in this article, as shown in Table 2, unless otherwise mentioned.

4.3.1. Effects of $n_{exchange}$ on success ratio

Firstly we evaluate the effect of $n_{exchange}$ on the success ratio of lookup. The other parameter values are the same as listed in Table 2, while the values of $n_{exchange}$ vary.

From Fig. 1a–c, we can see that the there is no significant difference between the two cases, i.e. $n_{exchange} = 10$ and $n_{exchange} = 15$. The overall trend with the success ratio, as shown in Fig. 1a, is that as the churn level decreases, the success ratio increases. The success ratio is close to 99% when the churn is at a rather low level, e.g. $m_{online} = 4000$ s. This makes sense as in a higher level of churn environment, the resource items will be stored in the right nodes of the overlay with lower probability that results in lower success ratio. For the mean network traffic load and the mean number of messages, the case $n_{exchange} = 15$ has slightly higher values than the case $n_{exchange} = 10$. The

Table 2
Notations and their meaning.

Notation	Meaning and the selected values
$t_{publish}$	Resource publish interval, $t_{publish} = 100$ s
$t_{republish}$	Resource republish interval, $t_{republish} = 60$ s
t_{lookup}	Resource lookup interval, $t_{lookup} = 125$ s
$t_{exchange}$	Routing table exchange interval, $t_{exchange} = 60$ s
$t_{keepalive}$	Routing table KeepAlive interval, $t_{keepalive} = 100$ s
$n_{exchange}$	Number of routing items included in one exchange response message, $n_{exchange} \in \{10, 15\}$
$n_{parallel}$	Parallelism degree of resource lookup, $n_{parallel} \in \{1, 2, 3\}$
$n_{replication}$	Replication degree of resource items, $n_{replication} \in \{1, 2, 3\}$
m_{online}	Mean online time (s), also reflecting the churn level, $m_{online} \in \{200, 400, 600, 800, 1000, 2000, 3000, 4000\}$
k_{value}	The size of the k-bucket of Kademlia, $k_{value} \in \{1, 2, 3, 4, 5\}$
$r_{success}$	Success ratio of the lookup (%)

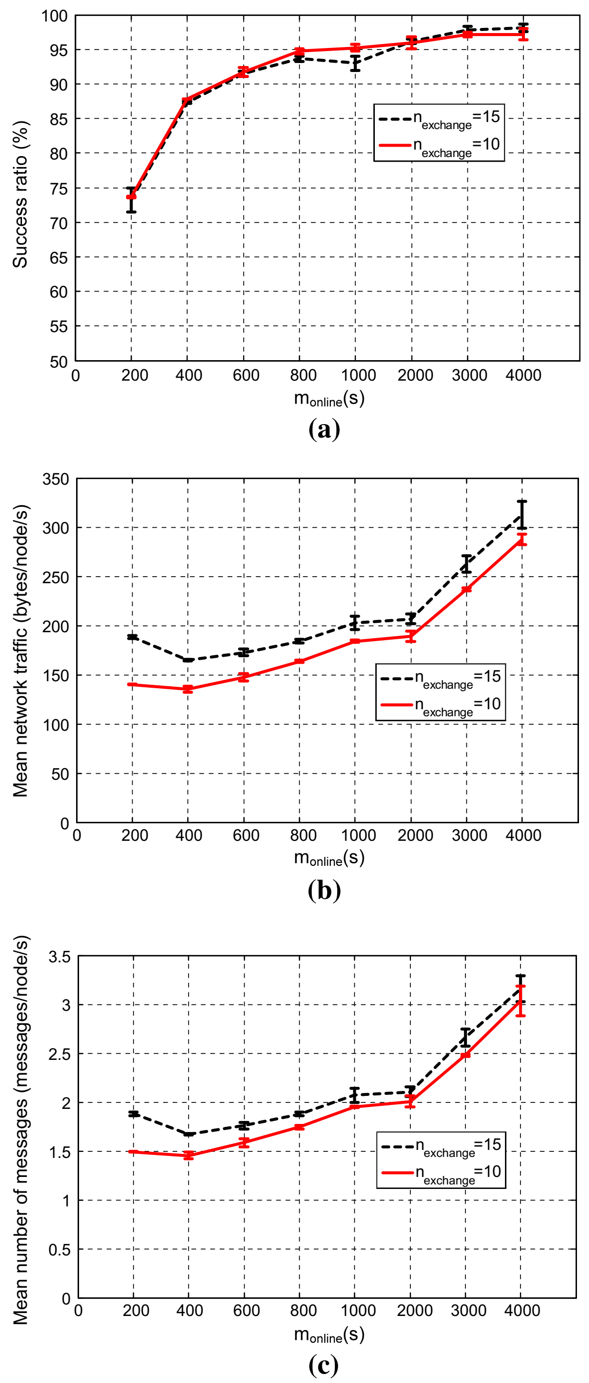


Fig. 1. (a) Success ratio, (b) mean network traffic load, and (c) mean number of messages. ($n_{parallel} = 1, n_{replication} = 1, k_{value} = 3$).

trends with the mean network traffic load and mean number of messages (refer to Fig. 1b and Fig. 1c) are that they firstly decrease as the value of m_{online} increases, after some critical point, i.e. $m_{online} = 400$ s, they start to increase, however. This is because in an extremely high churn environment, i.e. $m_{online} = 200$ s in this article, the frequent joining and leaving of nodes causes a lot of network traffic in the overlay, while in an extremely low churn environment, i.e. $m_{online} = 4000$ s in this article, most of the nodes stay online for a large fraction of the simulated time that causes a lot of maintenance traffic in the overlay.

We choose the case $n_{exchange} = 15$ in the subsequent Sections of this article as it does not increase the network traffic significantly as compared to $n_{exchange} = 10$, and 15 is the maximum number of routing items that can be included in one UDP message.

4.3.2. Effects of k_{value} on success ratio

Fig. 2 shows the effect of k_{value} on the success ratio with respect to different levels of churn. From Fig. 2, we can see that when $k_{value} = 1$, the overlay network has a relatively low success ratio, while in the other cases, i.e. $k_{value} = 2–5$, no significant difference is observed with the corresponding success ratio. The reason why the case $k_{value} = 1$ has a much lower success ratio is that in order to make sure a successful lookup process of Kademlia, every k-bucket of a node has to contain at least one contact if a node exists in the appropriate distance range [5]. However, in the churn environment, if the node in the associated k-bucket leaves the overlay and the host node containing that node in its routing table fails to update the associated k-bucket in time, the lookup passes through that k-bucket will result in a failure. Network partition, i.e. the overlay is partitioned into two or more disjointed sub-overlays, is also observed when $k_{value} = 1$. Therefore, the low success ratio is observed in the case $k_{value} = 1$. To provide robustness and redundancy for the overlay, we select $k_{value} = 3$ in the subsequent Sections (we will further show why we select $k_{value} = 3$ instead of $k_{value} = 2$).

4.3.3. Effects of parallel lookup on success ratio

In this subsection, we analyze the effect of different lookup parallelism degrees on the success ratio.

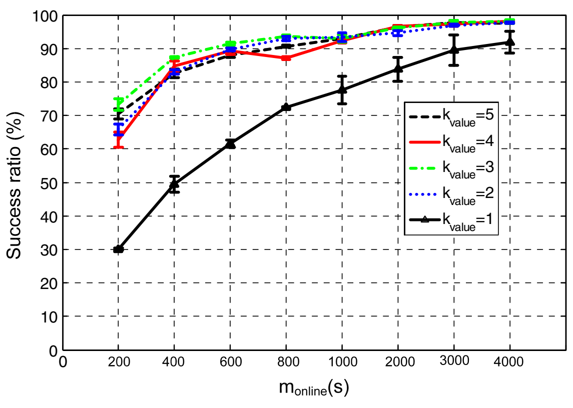


Fig. 2. Success ratio with respect to different k_{value}. ($n_{exchange} = 15, n_{parallel} = 1, n_{replication} = 1$).

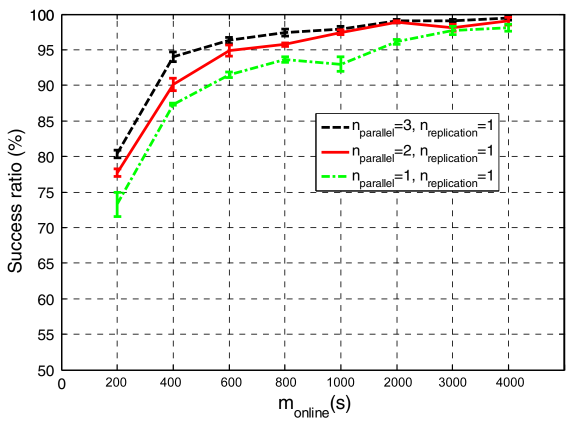


Fig. 3. Success ratio with respect to different parallelism degree ($n_{exchange} = 15, n_{replication} = 1, k_{value} = 3$).

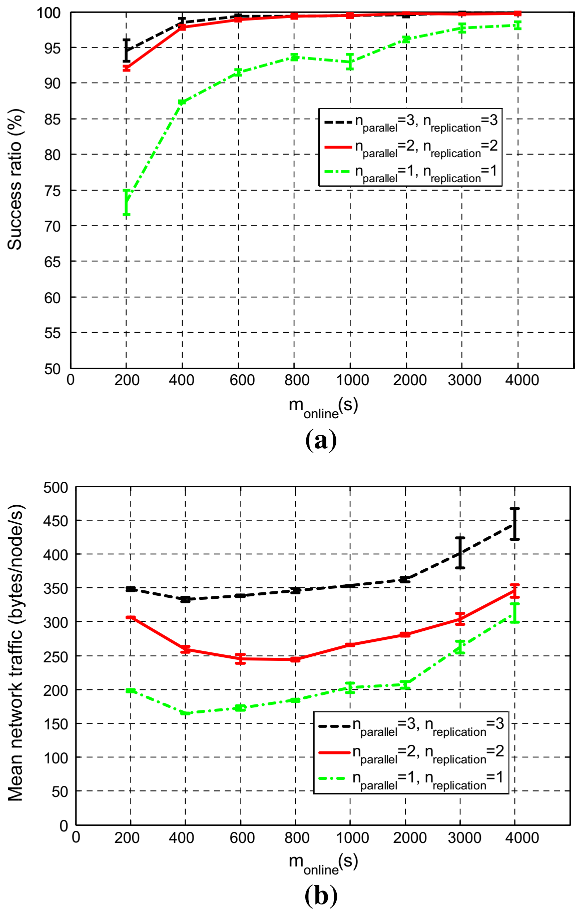
To observe the effect of parallelism degree on the success ratio, we fix the replication degree to one, which means each resource item is stored in just one node. The results are shown in Fig. 3. From the figure, it is clear that the success ratio becomes higher as the parallelism degree increases. For most of the churn levels, i.e. from $m_{online} = 400$ s to $m_{online} = 4000$ s, the two curves, i.e. "$n_{parallel} = 2$, $n_{replication} = 1$" and "$n_{parallel} = 3, n_{replication} = 1$", can achieve a success ratio higher than 90%. However, a success ratio like that is still not enough for a high reliability. In order to achieve a higher success ratio, we adopt the replication approach in the following subsection.

4.3.4. Effects of replication on success ratio

The success ratio with regard to the replication degree is shown in Fig. 4a. From the figure, we can see that the *"parallel lookup + replication"* approach achieves a significant performance improvement. In the case "$n_{parallel} = 3, n_{replication} = 3$", for most of the churn levels, i.e. from $m_{online} = 400$ s to $m_{online} = 4000$ s, it can guarantee a success ratio of more than 99%, even in the relatively high churn environment, i.e. $m_{online} = 200$ s, the success ratio is still around 93%. The success ratio of the case "$n_{parallel} = 2, n_{replication} = 2$" is slightly worse than the case "$n_{parallel} = 3, n_{replication} = 3$". However, it can still achieve a significant performance improvement compared to the case "$n_{parallel} = 1, n_{replication} = 1$", wherein neither parallel lookup nor replication approach is used.

The performance improvement of success ratio comes at the cost of increasing mean network traffic load and mean number of messages, which can be seen clearly from Fig. 4b and c. However, even in the case "$n_{parallel} = 3$, $n_{replication} = 3$", the mean network traffic load is no more than 450 bytes/node/s, the mean number of messages is no more than four messages/node/s, which are lightweight and acceptable in mobile environments where the typical transfer rate is tens to hundreds of KB/s.

To analyze solely the effect of replication approach on the success ratio, i.e. without parallel lookup approach, we fix the parallelism degree to one. The associated results are shown in Fig. 5. From the figure, we can see that the



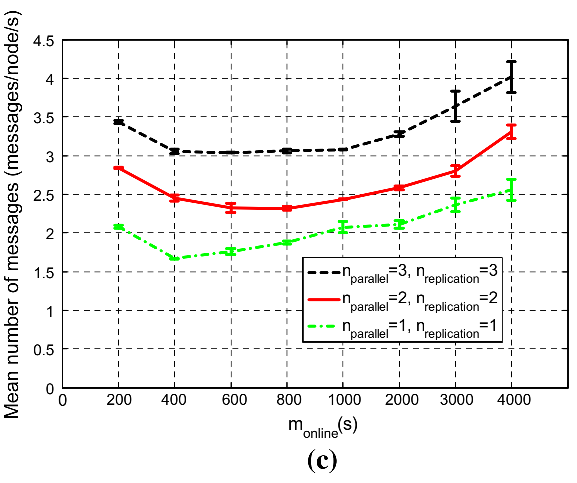


Fig. 4. (a) Success ratio, (b) mean network traffic load, and (c) mean number of messages ($n_{exchange} = 15, k_{value} = 3$).

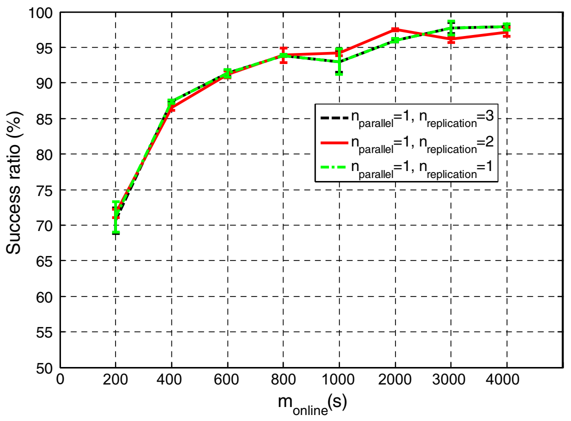


Fig. 5. Effects of sole replication approach on success ratio ($n_{exchange} = 15, n_{parallel} = 1, k_{value} = 3$).

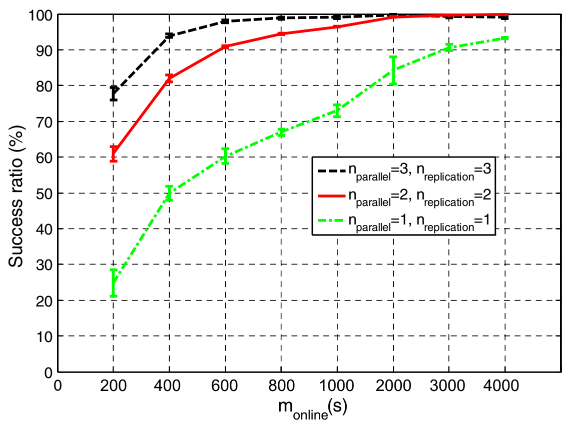


Fig. 6. Effects of ungraceful leaving on success ratio ($n_{exchange} = 15, k_{value} = 3$).

sole replication approach can not result in a remarkable performance improvement in terms of success ratio. The improvement of success ratio with respect to sole replication approach is not as notable as the sole parallel lookup approach, which can be concluded from Figs. 3 and 5.

4.3.5. *Effects of ungraceful leaving on success ratio*

Mobile environment is characterized by limited bandwidth and intermittent connections. If network disconnections take place abruptly, this will result in ungraceful leaving of the mobile nodes from the overlay. The ungraceful leaving also emerges in other situations, e.g. devices power off abruptly or there are malfunctions. In ungraceful leaving, just as stated in Section 1, the node neither notifies its routing neighbors about its leaving, nor transfers the resource items to its neighbors. To evaluate the effects of these exceptional situations on the success ratio, we conduct simulations on the ungraceful leaving. The results are shown in Fig. 6.

From Fig. 6, it is clear that the serial lookup without any replication approach, i.e. the curve "$n_{parallel} = 1, n_{replication} = 1$", has poor resilience against the exceptional situations, while the case "$n_{parallel} = 3, n_{replication} = 3$" has relatively good resilience against ungraceful leaving. In most of the levels of churn, from $m_{online} = 400$ s to $m_{online} = 4000$ s, the success ratio in the case "$n_{parallel} = 3, n_{replication} = 3$" is more than 90%. The performance of the case "$n_{parallel} = 2, n_{replication} = 2$" is between the two cases aforementioned.

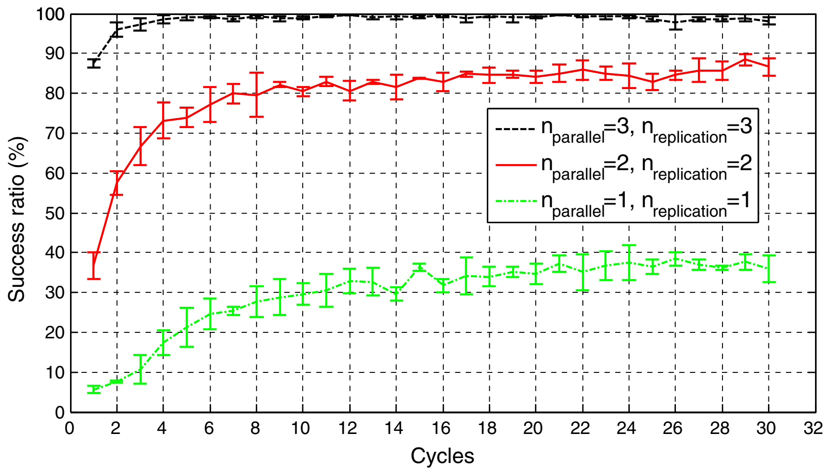


Fig. 7. Convergence measurement in the absence of churn ($t_{publish} = 300\,s, t_{lookup} = 350\,s, k_{value} = 3$).

4.3.6. *Convergence measurement*

Kademlia has the functionality of updating the routing table with each incoming message. Our P2PP simulation model utilizes this functionality besides the dedicated exchange message to update the routing table. The aforementioned performance evaluations are based on both mechanisms. To evaluate the sole effect of the maintenance messages, i.e. exchange and KeepAlive messages, on the success ratio, we conduct the convergence measurement in this subsection.

In the convergence measurement, no exchange or KeepAlive message is sent. After the node joins the overlay, it sends the publish and lookup messages periodically at intervals $t_{publish} = 300$ s and $t_{lookup} = 350$ s. The routing table is updated by the publish and lookup messages. Each experiment consists of 30 cycles. The definition of "*cycle*" herein is the time period during which exactly one publish and one lookup operation are executed by each node. The time interval for one cycle is 400 s. The Lookup operation looks up randomly the resource items published during that cycle. No republish operation is conducted here. To evaluate the success ratio in the absence of the maintenance messages precisely, one point that should be noticed is that no churn is introduced in these experiments, i.e. the 400 nodes stay online throughout the experiments. In this case, we can clearly observe the varying trend of the success ratio along with each cycle. Intuitively, the overlay will gradually converge as the number of cycles increases, which motivates the name "convergence measurement". The results are shown in Fig. 7.

As can be assumed, the success ratio grows as the number of cycles increases at the beginning. However, after several cycles, the results are slightly surprising because in the two cases, i.e. "$n_{parallel} = 1, n_{replication} = 1$" and "$n_{parallel} = 2, n_{replication} = 2$", the success ratio oscillates at around 35% and 85%, respectively, after 10 cycles instead of increasing continuously. One possible reason for this fact is that the routing information that can be obtained from publish and lookup operations is limited and reaches a near stable state after around 10 cycles. This also indicates the necessity of maintenance messages in these two cases. However, in the case "$n_{parallel} = 3, n_{replication} = 3$", the success ratio is almost always close to 100% after the second cycle, which means the maintenance messages may be redundant in this case.

It is also interesting to see the effect of maintenance traffic, i.e. exchange and KeepAlive messages, on the success ratio in the presence of churn. From the preceding analysis, we know that the case "$n_{parallel} = 3, n_{replication} = 3$" can guarantee a high success ratio in most situations. Thus, we fix the values of parallelism degree and replication degree to 3 in this subsection. Fig. 8 shows the effect of maintenance traffic on the success ratio. From the figure, it is clear that the exchange message has a significant effect on the lookup success ratio, as can be seen by comparing the "*Exchange, No KeepAlive*" case (red curve) with the "*No Exchange, No KeepAlive*" case (green curve). On the other hand, the KeepAlive message does not have a remarkable effect on the success ratio, as seen from the "*Exchange+KeepAlive*" case (black curve) and the "*Exchange, No KeepAlive*" case (red curve). Therefore, from these

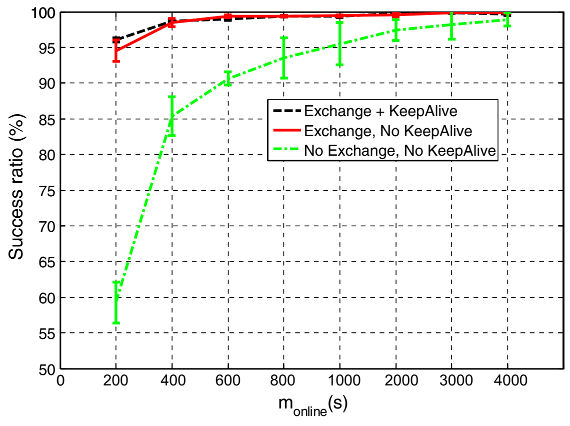


Fig. 8. Convergence measurement in the presence of churn ($n_{exchange} = 15, n_{parallel} = 3, n_{replication} = 3, k_{value} = 3$).

results, we can conclude that in the situation where the parallelism degree is 3 and replication degree is also 3, the KeepAlive message is not needed in order to achieve a high success ratio.

From the simulations and analysis above, we can draw the following conclusions:

- The mean network traffic load of P2PP is only several hundred bytes/node/s, and the mean number of messages is only several messages/node/s, which make P2PP lightweight enough to be suitable for mobile environments.
- The *k*-value of 3, parallelism degree of 3 and replication degree of 3 are enough to make P2PP resilient against all the measured levels of churn, and against exceptional network situation, i.e. ungraceful leaving.
- The KeepAlive maintenance traffic is not necessary to achieve a high success ratio in the case where parallelism degree is 3 and replication degree is 3.

4.4. Discussion

As mentioned in Section 2, several approaches have been utilized in the scientific community to evaluate the performance of Kademlia-based P2P systems, including mathematical analysis, real-life measurements and simulation models. Although the approaches are different, the results discovered are similar with respect to the parallelism degree and replication degree in Refs. [24,20], and in this article. While Wu et al. [24] and Stutzbach et al. [20] utilized mathematical analysis and real-life measurements, respectively, we acquired similar results by utilizing simulation models in this article. Even though the results presented in this article are from a different perspective and thus not directly comparable with the previous work [24,20], they nevertheless support each other.

Moreover, despite the fact that the experimental results shown in this Section are based on the Kademlia DHT algorithm, the conclusions we drew in this Section are applicable to some other DHT algorithms as well, such as Pastry and Tapestry. The foundation for this argument is that the parallel lookup mechanism and the resource replication mechanism can also be applied with these DHT algorithms. Accordingly, the conclusions that we drew in this Section could, in general, provide guidance for the optimal design choices and parameter configurations for structured P2P networks.

5. Implementation and measurement

Besides the simulation model of P2PP, we have also implemented a prototype that we name as Mobile P2PP (MP2PP). It is written in C++ code and works on two platforms: Symbian OS Series 60 on the Nokia N95 smartphone, and Ubuntu Linux on Sun Microsystems server hardware. In this Section, we introduce the prototype set-up and the measured quantities, i.e. the CPU load percentage, the network traffic load and the battery consumption.

5.1. Prototype set-up

We have run P2PP prototype measurements with a Linux-based server array, where multiple P2PP peers are active on the same machine as separate processes. This enables the simulation of large overlays with an actual C/C++ based implementation of the P2PP protocol. There are controlling scripts that make the peers behave as needed. The churn-related online and offline times, publish and lookup behavior and simulation synchronization are controlled by the scripts.

The churn time follows the exponential distribution with a specified mean value that describes the level of churn used in a given experiment.

When we say that an overlay contains *N* peers, this refers to the total number of online and offline peers. Thus, on average, there are $N/2$ peers online at a given moment.

The server-based system was responsible for simulating the majority of the peers of an overlay. In addition to the servers, two mobile devices (Nokia N95) were used as peers in the overlay. To be accurate, the overlays contained $N+2$ peers (mobile ones included); however, as *N* is large, this does not cause significant differences in the interpretation of the measurement results.

The mobile devices did not churn; they stayed online constantly. Moreover, they did not issue any publish or lookup requests, since it was most interesting to evaluate cases where the mobile device only provides services to others; it must not skew these measurements by sending service requests to the other peers. In other aspects, the behavior of the mobile devices was identical to the behavior of the nodes in servers.

All the devices, i.e. server machines and mobile phones, had public IP addresses. This enabled P2P-style communication without special techniques such as Network Address Translation (NAT) traversal.

The parameters used in the simulated P2PP overlays were as follows: The *k*-value for Kademlia was 3 and serial lookup was utilized. The $t_{keepalive}, t_{exchange}$, and $t_{republish}$ were 10 s, 30 s, and 30 s, respectively. Every peer published (and periodically re-published) exactly one data resource. The range of t_{lookup} was {15, 120}s, and the range of churn levels, i.e. m_{online}, was {60, 480}s. The Nokia Energy Profile software was utilized to record the CPU load percentage, the network traffic load, and the battery consumption in the subsequent Sections. One point to be noted here is that the parameter values used in this Section were slightly lower than the values used in the simulation models described in Section 4. This is mainly because the simulation models described in Section 4 run on one single server, while the prototype measurements in this Section run on a server array. Therefore, lower parameter values (i.e. higher network activities) could be supported in the prototype measurements.

5.2. CPU load percentage

In this Section, the CPU load percentage of mobile peers taking part in the P2PP overlay is measured. The motivation is that mobile devices are characterized by the limited CPU processing capability. If, for instance, the mobile node taking part in the P2PP overlay results in 80% or 90% of CPU load percentage, this will cause a great deal of user annoyance and, accordingly, decrease the feasibility of such applications being adopted in real life.

The mean value of the CPU load percentage is shown in Table 3 with different overlay activity parameters. Two cases, i.e. UMTS and WLAN, are measured to provide a comparison. In order to further analyze the trend of the CPU load percentage, the probability density function (PDF) of one of the combinations $(t_{lookup} = 15\ \text{s}, m_{online} = 60\ \text{s}, N = 2000)$, which has the maximum mean value (24.66%) in all the combinations, is provided in Fig. 9.

From Table 3, it is clear that the mean CPU load percentage of the mobile node taking part in P2PP overlay is less than 20% in UMTS, and less than 25% in WLAN. For most of the cases, the mean CPU load percentage is around 12.5% in UMTS, and around 15% in WLAN. The CPU load percentage of the WLAN access mode is slightly higher than that of the UMTS access mode. The CPU load percentage of the 2,000 nodes scenario is also slightly higher in comparison with the 200 nodes scenario.

From Fig. 9, we can see that the PDF of the CPU load percentage has a long tail. However, there are few time points when the CPU load percentage is more than 70%; for the remaining time, the CPU load percentage is less than 70%.

Therefore, from the viewpoint of the CPU load, we can conclude that it is feasible for mobile nodes taking part in the P2PP overlay without consuming heavy CPU processing capability.

5.3. Network traffic load

In this Section, the uplink and downlink network traffic of mobile nodes taking part in the P2PP overlay are mea-

Table 3
CPU load percentage of mobile nodes taking part in P2PP overlay.

Access network type	Overlay size N	Lookup interval t_{lookup} (s)	Mean online time m_{online} (s)	CPU load percentage (%)	Confidence lower limit (95% confidence level)	Confidence upper limit (95% confidence level)
UMTS	200	15	60	12.64	12.19	13.10
			480	12.19	11.76	12.62
		120	60	12.97	12.53	13.42
			480	11.61	11.18	12.04
	2000	15	60	14.02	13.50	14.54
			480	18.41	17.89	18.93
		120	60	13.32	12.82	13.82
			480	12.85	12.35	13.36
WLAN	200	15	60	15.69	15.17	16.21
			480	14.86	14.35	15.37
		120	60	15.71	15.18	16.23
			480	14.34	13.82	14.85
	2000	15	60	24.66	24.10	25.23
			480	19.95	19.38	20.52
		120	60	16.86	16.30	17.43
			480	16.39	15.83	16.95

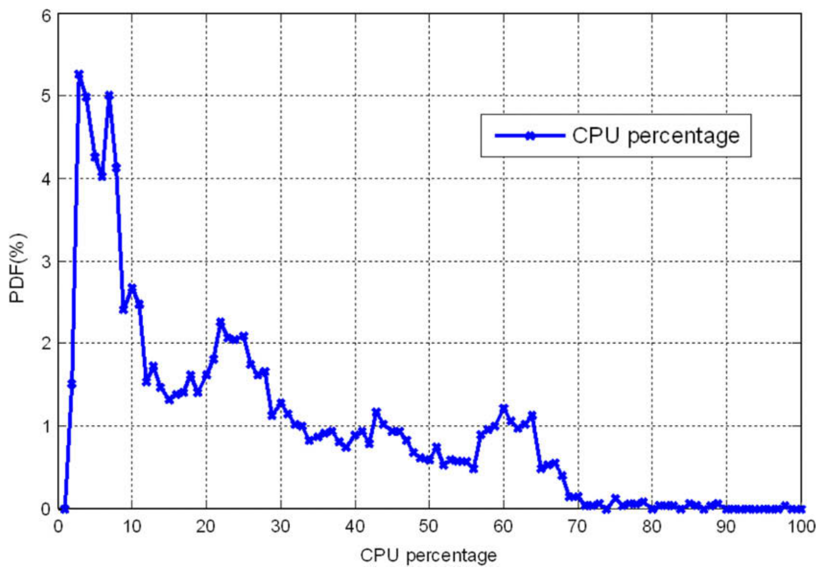


Fig. 9. Probability density function of CPU load percentage $(t_{lookup} = 15\ s, m_{online} = 60\ s, N = 2000)$.

Table 4
UMTS uplink and downlink network traffic of mobile nodes taking part in P2PP overlay.

Overlay size N	Lookup interval t_{lookup} (s)	Mean online time m_{online} (s)	Downlink traffic (confidence lower limit, confidence upper limit) (bytes/s)	Uplink traffic (confidence lower limit, confidence upper limit) (bytes/s)
200	15	60	445.94 (427.48, 464.41)	554.16 (531.39, 576.93)
		480	398.24 (379.09, 417.38)	478.35 (455.75, 500.95)
	120	60	460.40 (442.54, 478.26)	573.69 (551.26, 596.11)
		480	390.72 (371.04, 410.39)	453.29 (430.84, 475.74)
2000	15	60	562.93 (532.35, 593.52)	600.45 (566.95, 633.96)
		480	659.72 (633.86, 685.58)	811.63 (781.16, 842.11)
	120	60	551.90 (521.84, 581.96)	582.22 (550.48, 613.97)
		480	566.79 (534.20, 599.38)	584.57 (550.69, 618.45)

sured. As an example, the results of the mean network traffic load over the measured time in the UMTS access mode are shown in Table 4. The network traffic load in the WLAN access mode is very similar to the UMTS access mode; therefore, it is not shown in this article.

From Table 4, we can see the mean network traffic load is around 600 bytes/s in both the UMTS uplink and UMTS downlink modes. The network traffic in the UMTS downlink mode is slightly higher than that in the UMTS uplink mode. And the values shown in Table 4 are also slightly higher than those shown in Section 4.3. This is because the associated time intervals, e.g. t_{lookup} and m_{online}, in the prototype measurements are lower than those in the simulations in Section 4, which results in higher network traffic load in prototype measurements. As a whole, from the viewpoint of network traffic load, it is highly feasible for mobile nodes to take part in the P2PP overlay as the mean network traffic load is only several hundred bytes per second.

5.4. Power consumption

Besides the CPU load percentage and network traffic load measurements shown in this article, the power consumption measurements of the mobile nodes (Nokia N95 mobile phones) taking part in the P2PP overlay were also conducted and the results were shown in [35]. The basic finding was that the power consumption in UMTS mode was higher than in WLAN mode. However, for the power consumption measurement of mobile nodes taking part in the P2PP overlay, only the overall power consumption, i.e. the mean power consumption caused by all kinds of overlay activities, can be measured. In this Section, we look deeper into the power consumption with respect to different UDP packet sizes, different sending and receiving intervals, to get a better insight.

No overlay is utilized in these measurements; the mobile nodes just send and receive UDP packets of different sizes at different time intervals. When executing the measurements, besides the sending and receiving operations of UDP packets, only the Nokia Energy Profiler software is running in Nokia N95 smart phones. Each measurement runs for 35 min, 3 min at the beginning to make sure the background light is switched off and 2 min at the end to make some redundancy for the data. The real results are taken from the middle 30 min. Two phones are used in parallel to guarantee the correctness of the measurement results. In the following Sections, we use the notations shown in Table 5.

5.4.1. Symmetric measurements

In this Section, we show the results of the symmetric measurements. "*Symmetric*" means the same packet sizes and same time intervals are used for both the sending and receiving processes.

The power consumption of Nokia N95 smart phones under different levels of network activity is shown in Fig. 10. One point to be noted here is that the *Idle* state of the mobile phone (without any application running except the Nokia Energy Profiler) also consumes some power. It is 0.0649 w on average. The power consumption and battery life shown in Figs. 10–13 have excluded the corresponding power consumption $(0.0649w)$.

In order to calculate the battery life of the mobile phone under associated network activities, the following formula is utilized:

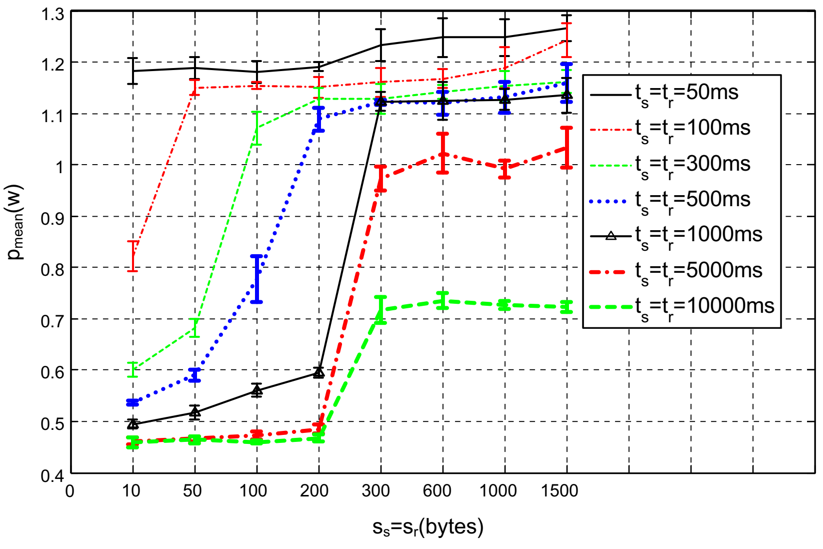
$$t_{battery} = \frac{0.950\ \text{Ah} \cdot v_{avg}}{p_{mean}}(h). \tag{1}$$

The nominal battery capacity of the Nokia N95 mobile phone is 950 mAh (i.e. 0.950 Ah), and the nominal voltage of the battery is 3.7 V. Therefore, we can get the associated battery life of the mobile phones where only the sending and receiving packets operations are executed, as shown in Fig. 11. In real life, when the other network activities are being executed simultaneously, the battery life will be much less than this.

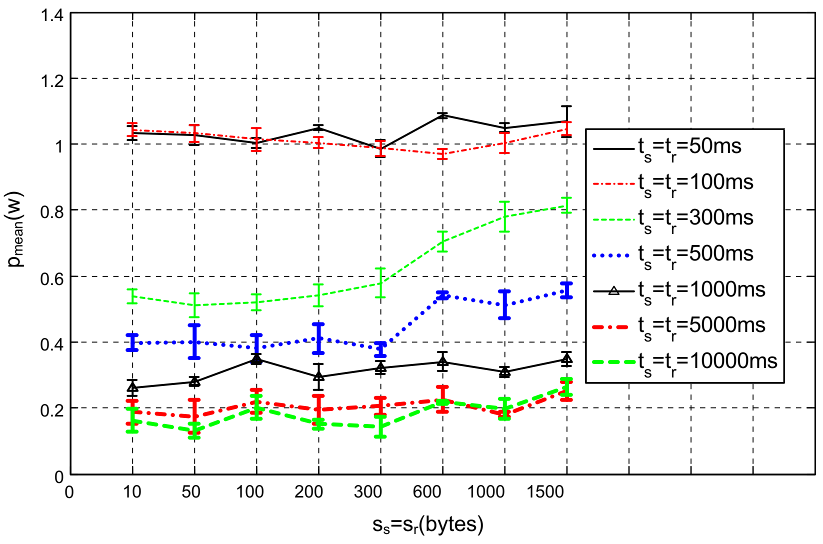
From Fig. 10, we can see that in the UMTS access mode, there is some gap between the packet sizes of 200 and 300 bytes in the "$t_s = t_r = 10,000$ ms", "$t_s = t_r = 5000$ ms", and "$t_s = t_r = 1000$ ms" cases. The gap takes

Table 5
Notations and their meaning.

Notation	Meaning
t_s	The intervals for sending UDP packets, $t_s \in \{50, 100, 300, 500, 1000, 5000, 10000\}$ ms
t_r	The intervals for receiving UDP packets, $t_r \in \{50, 100, 300, 500, 1000, 5000, 10000\}$ ms
s_s	Packet sizes of the sent messages, $s_s \in \{10, 50, 100, 200, 300, 600, 1000, 1500\}$ bytes
s_r	Packet sizes of the received messages, $s_r \in \{10, 50, 100, 200, 300, 600, 1000, 1500\}$ bytes
p_{mean}	The mean power consumption (w)
$t_{battery}$	The battery life of mobile phones (h)



(a)



(b)

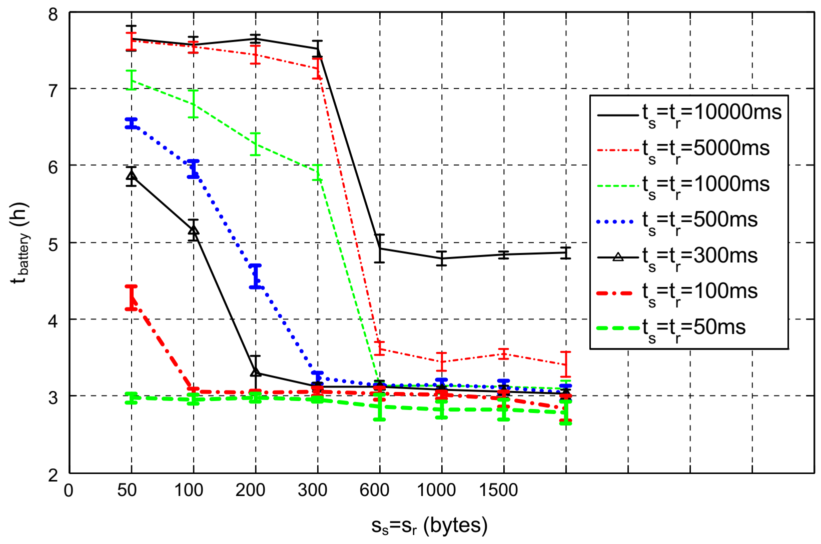
Fig. 10. Power consumption of Nokia N95 smart phone. (a) UMTS and (b) WLAN.

place between the 50 and 200 bytes for the cases "$t_s = t_r = 500$ ms" and "$t_s = t_r = 300$ ms", while for the case "$t_s = t_r = 100$ ms" the gap takes place between 10 and 50 bytes. This is mainly related to the frame size of the data link layer in the UMTS access mode. In the small packet size range, the power consumption of most of the transmission intervals converges to around 0.5 w, and the power consumption mainly comes from the packet headers. In the large packet size range, the power consumption decreases as transmission intervals increase. In the WLAN environment, the packet size does not have a notable effect on the power consumption, regardless of the sending and receiving intervals. The overall trend is that the power consumption in WLAN access mode decreases as the sending and receiving intervals increase.

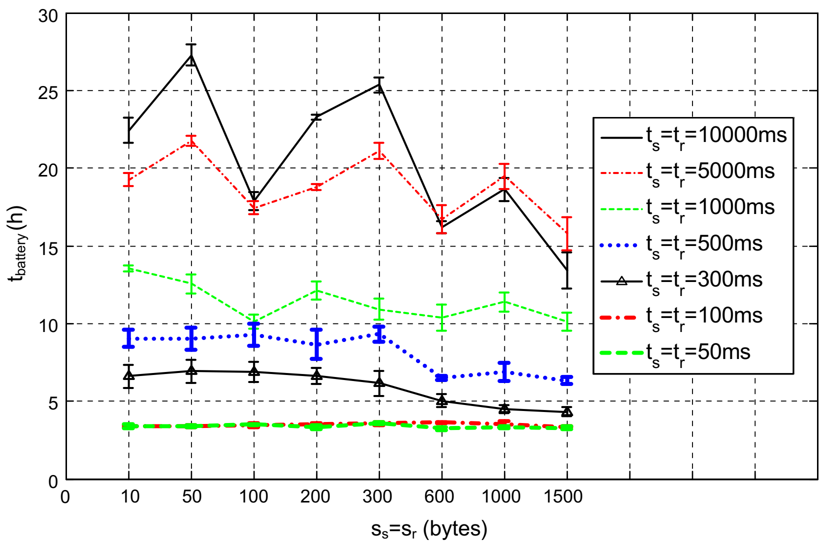
In Fig. 11, the corresponding battery life is shown. We can see that, on average, the mobile phone taking part in the P2PP overlay through the WLAN access network has a longer battery life than through the UMTS access network. The battery life in UMTS access mode varies from around 3–8 h, while in WLAN it varies from 3 to 27 h. In a context with a high level of network activity, i.e. "$t_s = t_r = 50$ ms" and "$t_s = t_r = 100$ ms", there is no significant difference between UMTS and WLAN access modes. The battery life in these two cases is around 3 h in both UMTS and WLAN access modes.

5.4.2. Asymmetric measurements

In the previous Section, we measured the symmetric network activities where the packet sizes and intervals



(a)



(b)

Fig. 11. Battery life of Nokia N95 smart phone. (a) UMTS and (b) WLAN.

for sending and receiving messages are the same. To analyze the effect of the imbalanced network activity on the power consumption, we conduct asymmetric measurements in this part where the packet sizes, sending and receiving intervals are different. As there are numerous combinations of the packet sizes and time intervals, we choose two typical cases to conduct the measurements. In one case, we choose two values of the packet size, i.e. 100 bytes and 1500 bytes (for both sending and receiving, asymmetrically), to evaluate the effect of asymmetric sending and receiving packet sizes on the power consumption. The results are shown in Fig. 12. In the other case, we keep the sending and receiving packet sizes as fixed, i.e. "$s_s = s_r = 100 bytes$", and evaluate the effect of asymmetric sending and receiving intervals on the power consumption, the results are shown in Fig. 13. One noteworthy point here is that the x-axis of Fig. 13 stands for t_s or t_r. For the two cases, i.e. "$t_s = 50$ ms,*UMTS*" and "$t_s = 50$ ms,*WLAN*", the x-axis stands for the values of t_r. For the other two cases, i.e. "$t_r = 50$ ms,*UMTS*" and "$t_r = 50$ ms,*WLAN*", the x-axis stands for the values of t_s. We integrate these four curves into one figure to make a clear comparison among them.

From Fig. 12, it is clear that the effect of imbalanced sending and receiving packet sizes is not remarkable, as the two curves of different sending and receiving packet sizes almost always overlap with each other. As a whole, the mobile phone accessed through WLAN has less power consumption than through UMTS. This trend becomes more explicit as the sending and receiving time intervals increase.

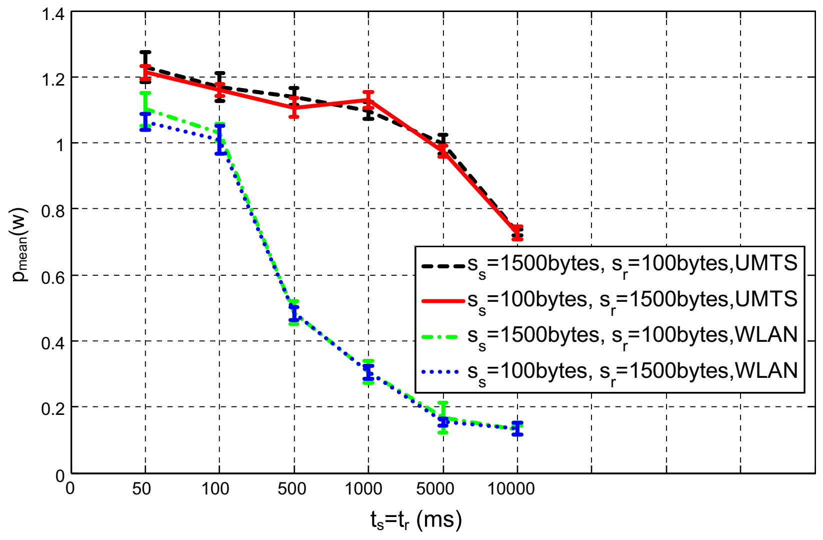


Fig. 12. Effects of asymmetric sending and receiving packet sizes on the power consumption.

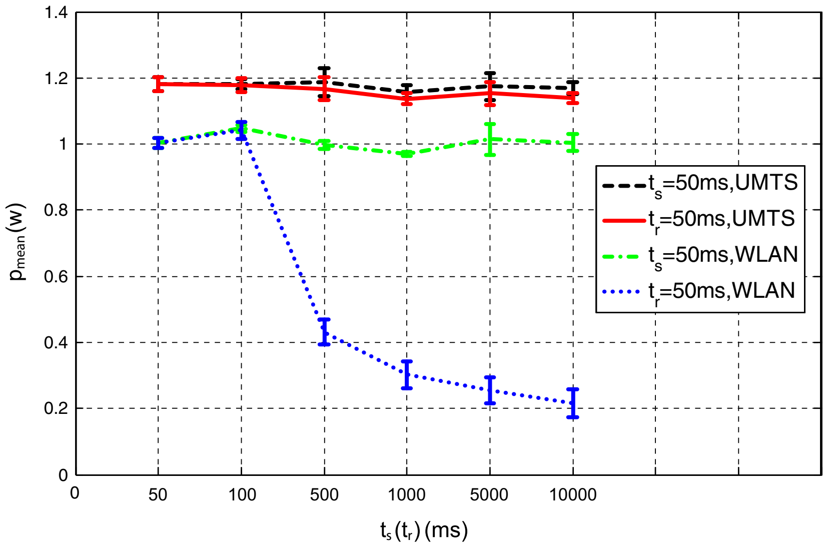


Fig. 13. Effects of asymmetric sending and receiving time intervals on the power consumption.

From Fig. 13, we can see that the asymmetric sending and receiving time intervals do not have remarkable effects on the power consumption in the UMTS environment. However, they do have significant effects on the power consumption in the WLAN environment. As shown in Fig. 13, if the time interval is larger than 500 ms, the power consumption in the case "$t_r = 50$ ms,*WLAN*" is much less than in the case "$t_s = 50$ ms,*WLAN*". One possible explanation for this is that, after some critical point, e.g. 500 ms, the power consumption spent in sending is much more than that spent in receiving. We also noticed one phenomenon that is, in the WLAN access mode, the mobile phones under measurement always receive more traffic than what is sent by them. The extra traffic is probably from the surrounding laptops that also have the same WLAN connection. This is also one possible reason why the "$t_r =$ 50 ms,*WLAN*" case consumes less power than the case "$t_s = 50$ ms,*WLAN*" does. Again, the same trend, i.e. UMTS network access consumes more power than WLAN access, is observed here.

5.5. Discussion

Our measurements on CPU load and network traffic load, shown in Section 5.2 and 5.3, respectively, are conducted with the Kademlia-based overlay utilizing the P2PP protocol. However, the power consumption measurements shown in Section 5.4 are independent from any DHT algorithm or P2P protocol. Therefore, the conclusions we drew in Section 5.4 regarding the power consumption are not limited to P2P applications; instead, they are applicable to any mobile application as long as the transport protocol is UDP. Based on our plain UDP measurements, it is possible to improve the energy efficiency of P2P protocols by implementing the protocols in such a way that they use the UDP traffic patterns observed to be the most efficient. It should also be noted that further improvements in the energy efficiency of mobile terminals and access networks are still clearly needed, as the battery life of mobile phones under a high network load is short in general.

6. Conclusion

This article presented a performance evaluation of a communication-oriented P2P system in the presence of churn. We used Kademlia-based P2PP as an example signaling protocol. Through simulations, we drew the conclusion that the *k*-value of 3, parallelism degree of 3 and replication degree of 3 were enough to make the P2PSIP system resilient against various levels of churn and ungraceful leaving. With the P2PP protocol, the use of KeepAlive messages was not necessary in order to achieve a high success ratio, as the routing tables were also updated by all the other messages, such as lookup and publish. Through prototype measurements, we observed that the required bandwidth was low enough for P2PSIP peers to reside on devices in mobile access networks, such as UMTS and WLAN that have typical transfer rates of tens to hundreds of KB/s in the currently deployed networks. The measured CPU load was also acceptable for mobile nodes acting as P2PSIP peers. The power consumption measurements showed that UMTS access mode consumes more power than the WLAN access mode, and the protocol packets with sizes of 200 bytes or less were the most energy efficient in the UMTS access mode. Future work includes deploying the prototype in larger settings, e.g. in the PlanetLab test-bed network, to confirm the conclusions drawn in this article.

Acknowledgments

The authors would like to thank TEKES (The Finnish Funding Agency for Technology and Innovation), Ericsson, Nokia, and Nethawk for financially supporting this research work in the project Decicom. The authors would also like to thank Jari Korhonen for his contribution in the implementation of the prototype and Xiaojia Lu for his valuable discussion about the battery consumption of mobile phones.

References

[1] S. Androutsellis-Theotokis, D. Spinellis, A survey of peer-to-peer content distribution technologies, ACM Computing Surveys 36 (4) (2004) 335–371. December.
[2] S. Baset, H. Schulzrinne, M. Matuszewski, Peer-to-Peer Protocol (P2PP), draft-baset-p2psip-p2pp-01, in preparation. November 19, 2007. Avaliable from: <http://tools.ietf.org/id/draft-baset-p2psip-p2pp-01.txt>.
[3] P2PSIP Working Group. <http://www.ietf.org/html.charters/p2psip-charter.html>.
[4] C. Jennings, B. Lowekamp, E. Rescorla, S. Baset, H. Schulzrinne, REsource LOcation And Discovery (RELOAD) Base Protocol, draft-ietf-p2psip-base-02, in preparation. October 6, 2009. Available from: <http://tools.ietf.org/html/draft-ietf-p2psip-base-04>.
[5] P. Maymounkov, D. Mazires. Kademlia: a peer-to-peer information system based on the xor metric, in: Peer-to-Peer Systems: First International Workshop, IPTPS 2002 Cambridge, MA, USA, March 7–8, 2002, pp. 53–65.
[6] P. Saint-Andre, Streaming XML with Jabber/XMPP, IEEE Internet Computing 9 (5) (2005) 82–89. September–October.
[7] D. Bonfiglio, M. Mellia, M. Meo, D. Rossi, P. Tofanelli, Revealing skype traffic: when randomness plays with you, in: Proceedings of the 2007 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, SIGCOMM'07, Kyoto, Japan, August 27–31, 2007, pp. 37–48.
[8] J.-F. Wauthy, L. Schumacher, Implementation and performance evaluation of a P2PSIP distributed proxy/registrar, in: Proceedings of the 2007 International Conference on Next Generation Mobile Applications, Services and Technologies, NGMAST '07, September 12–14, 2007, pp. 119–124.
[9] M. Matuszewski, E. Kokkonen, Mobile P2PSIP – peer-to-peer SIP communication in mobile communities, in: Proceedings of the Fifth IEEE Consumer Communications and Networking Conference, CCNC, January 10–12, 2008, pp. 1159–1165.
[10] E. Kokkonen, S. Baset, M. Matuszewski, Demonstration of peer-to-peer session initiation protocol (P2PSIP) in the mobile environment, in: Proceedings of the Fifth IEEE Consumer Communications and Networking Conference, CCNC 2008, January 10–12, 2008, pp. 1221–1222.
[11] I. Baumgart, P2PNS: a secure distributed name service for P2PSIP, in: Proceedings of the Sixth Annual IEEE International Conference on Pervasive Computing and Communications, PerCom, March 17–21, 2008, pp. 480–485.
[12] I. Martinez-Yelmo, A. Bikfalvi, C. Guerrero, R. Cuevas, A. Mauthe, Enabling global multimedia distributed services based on hierarchical DHT overlay networks, in: Proceedings of the Second International Conference on Next Generation Mobile Applications, Services and Technologies, NGMAST '08, September 16–19, 2008, pp. 543–549.
[13] D.A. Bryan, B.B. Lowekamp, M Zangrilli, The design of a versatile, secure P2PSIP communications architecture for the public internet, in: IEEE International Symposium on Parallel and Distributed Processing, IPDPS, April 14–18, 2008. pp. 1–8.
[14] K.M. Cohrs, Implementation and Evaluation of the Peer-to-Peer-Protocol (P2PP) for P2PSIP, M.S. Thesis, Computer Networks Group, Institute of Computer Science, Georg-August-University of Göttingen, Göttingen, Germany, 2008.
[15] I. Stoica, R. Morris, D. Liben-Nowell, D.R. Karger, M.F. Kaashoek, F. Dabek, H. Balakrishnan, Chord: a scalable peer-to-peer lookup protocol for Internet applications, IEEE/ACM Transactions on Networking 11 (1) (2003) 17–32. Feb..
[16] M. Steiner, T. En-Najjary, E.W. Biersack, A global view of KAD, in: Proceedings of the Seventh ACM SIGCOMM Conference on Internet Measurement, IMC'07, San Diego, California, USA, October 24–26, 2007, pp. 117–122.
[17] D. Carra, E.W. Biersack, Building a reliable P2P system out of unreliable P2P clients: the case of KAD, in: Proceedings of the 2007 ACM CoNEXT Conference, CoNEXT '07, New York, NY, USA, December 10–13, 2007.
[18] D. Stutzbach, R. Rejaie, Characterizing churn in peer-to-peer networks, in: Technical Report CIS-TR-05-03, University of Oregon, 2005.
[19] D. Stutzbach, R. Rejaie, Understanding churn in peer-to-peer networks, in: Proceedings of the Internet Measurement Conference (IMC), October 2006.
[20] D. Stutzbach, R. Rejaie, Improving lookup performance over a widely-deployed DHT, in: Proceedimgs of the Infocom 06, April 2006.
[21] J. Li, J. Stribling, T.M. Gil, R. Morris, F. Kaashoek, Comparing the performance of distributed hash tables under churn, in: Proceedings of the Thrid International Workshop on Peer-to-Peer Systems (IPTPS), San Diego, CA, February 2004.
[22] J. Li, J. Stribling, F. Kaashoek, R. Morris, T. Gil, A performance vs. cost framework for evaluating DHT design tradeoffs under churn, in: INFOCOM, Miami, FL, March 2005.
[23] B.Y. Zhao, J. Kubiatowicz, A.D. Joseph, Tapestry: An Infrastructure for Fault-tolerant Wide-area Location and Routing, Technical Report UCB/CSD-01-1141, Computer Science Division, University of California, Berkeley, April 2001.

[24] D. Wu, Y. Tian, K.W. Ng, Analytical study on improving DHT lookup performance under churn, in: Proceedings of the IEEE P2P 2006, Cambridge, UK, September 2006.
[25] A. Binzenhöfer, H. Schnabel, Improving the Performance and Robustness of Kademlia-based Overlay Networks, University of Würzburg, Institute of Computer Science, Research Report, Report No. 405, April 2007.
[26] A. Binzenhöfer, K. Leibnitz, Improving the Estimating Churn in Structured P2P Overlay Networks, University of Ẅ urzburg, Institute of Computer Science, Research Report, Report No. 404, April 2007.
[27] S. Rhea, D. Geels, J. Kubiatowicz, Handling churn in a DHT, in: USENIX, 2004, pp. 127–140.
[28] P.B. Godfrey, S. Shenker, I. Stoica, Minimizing churn in distributed systems, in: Proceedings of ACM SIGCOMM, Pisa, Italy, September 2006, pp. 147–158.
[29] O. Herrera, T. Znati, Modeling churn in P2P networks, in: Proceedings of the 40th Annual Simulation Symposium, ANSS '07, March 2007, pp. 33–40.
[30] S. Ktari, M. Zoubert, A. Hecker, H. Labiod, Performance evaluation of replication strategies in DHTs under churn, in: Proceedings of the Sixth International Conference on Mobile and Ubiquitous Multimedia, MUM '07, 2007, pp. 90–97.
[31] S. Gurun, P. Nagpurkar, B.Y. Zhao, Energy consumption and conservation in mobile peer-to-peer systems, in: Proceedings of the First International Workshop on Decentralized Resource Sharing in Mobile Computing and Networking, 2006, pp. 18–23.
[32] J.K. Nurminen, J. Nöyränen, Energy-consumption in mobile peer-to-peer – quantitative results from file sharing, in: Proceedings of the Fifth IEEE Consumer Communications and Networking Conference (CCNC), 2008, pp. 729–733.
[33] I. Kelényi, J.K. Nurminen, Energy aspects of peer cooperation – measurements with a mobile DHT system, in: Proceedings of the Cognitive and Cooperative Wireless Networks Workshop in the IEEE International Conference on Communications, Beijing, China, 2008, pp. 164–168.
[34] I. Kelényi, J.K. Nurminen, Optimizing energy consumption of mobile nodes in heterogeneous Kademlia-based distributed hash tables, in: Proceedings of the Second International Conference and Exhibition on Next Generation Mobile Applications, Services and Technologies, Cardiff, Wales, UK, 2008, pp. 70–75.
[35] O. Kassinen, E. Harjula, J. Korhonen, M. Ylianttila, Battery life of mobile peers with UMTS and WLAN in a Kademlia-based P2P overlay, in: Proceedings of the 20th Personal, Indoor and Mobile Radio Communications Symposium (PIMRC09), Tokyo, Japan, September 13–16, 2009.
[36] Z. Ou, E. Harjula, O. Kassinen, M. Ylianttila, Feasibility evaluation of a communication-oriented P2P system in mobile environments, in: The International Conference on Mobile Technology, Applications and Systems (ACM Mobility Conference 2009), Nice, France, September 2–4, 2009.
[37] K. Pawlikowski, H.-D.J. Jeong, J.-S. Ruth Lee, On credibility of simulation studies of telecommunication networks, in: IEEE Communications Magazine, January 2002.

Zhonghong Ou received the B.E. degree in electronic engineering from Shijiazhuang Railway Institute, Shijiazhuang, Hebei, in 2003. He received the M.Sc. degree in electronic engineering from Beijing University of Posts and Telecommunications, Beijing, in 2005. He is now pursuing the Ph.D. degree at both Beijing University of Posts and Telecommunications and University of Oulu, Finland. His current research interests span the fields of P2PSIP systems, P2P networks and distributed systems, routing algorithms and protocols.

Erkki Harjula received his M.Sc. degree in electrical and information engineering from the University of Oulu, Finland, in 2007. He is now studying on Ph.D. degree at the University of Oulu, Finland. He is a researcher and the project manager in the DECICOM project focusing on P2PSIP, IMS and the application and business scenarios around them. His research interests include: structured P2P protocols and networks, hybrid P2P protocols and networks, and mobile networks.

Otso Kassinen is working as a research scientist in the MediaTeam Oulu research group at the University of Oulu, Finland. He received his M.Sc. degree in Information engineering from the University of Oulu, Finland, in 2007. He is currently working towards a Ph.D. degree in the University of Oulu. In his research work, he is focusing on mobile middleware solutions for the management of device resources, the establishment of communication sessions, and scalable service provision especially in DHT-based P2P networks.

Mika Ylianttila received his M.Sc. degree in electrical and information engineering from the University of Oulu, Finland, in 1998; Licentiate of Technology degree in 2001, and Doctor of Technology (Ph.D.) degree in 2005, respectively. He is a professor and adjunct professor in computer science and information networks at the Information Processing laboratory and Research Manager at the MediaTeam Oulu research group at the University of Oulu, Finland. His research interests include: protocol design and performance; communication and middleware architectures; mobile applications and services. Dr. Ylianttila is the Chairman of the IEEE Finland Section 2008–2009 and a Senior Member of IEEE.