Feasibility evaluation of a communication-oriented P2P system in mobile environments

Zhonghong Ou[1,2], Erkki Harjula[1], Otso Kassinen[1], Mika Ylianttila[1]

[1]MediaTeam, University of Oulu
Inf. Proc. Laboratory, P.O.BOX 4500
FIN-90014 University of Oulu, Finland

firstname.lastname@ee.oulu.fi

[2]PCN&CAD Center
School of Computer Engineering
Beijing University of Posts and Telecommunications

zhhouyang@gmail.com

ABSTRACT

We present the feasibility evaluation of a structured communication-oriented Peer-to-Peer (P2P) system being used in mobile environments in this paper. The different levels of churn are modeled by the exponential distribution with varied value of mean online time. Our system utilizes Kademlia with some modifications as the underlying Distributed Hash Table (DHT) algorithm, and Peer-to-Peer Protocol (P2PP), one of the former candidates of Peer-to-Peer Session Initiation Protocol (P2PSIP) working group, as the signaling protocol. A prototype is implemented to evaluate the feasibility of mobile nodes acting as fully fledged peers. The prototype measurements show it is feasible for the mobile nodes to take part in the overlay from the viewpoints of CPU load and network traffic load. Through battery measurements, we draw the conclusion that the UMTS access mode consumes slightly more power than the WLAN access mode in general. Protocol packets with sizes of 200 bytes or less are observed to be the most energy efficient in the UMTS access mode.

Categories and Subject Descriptors

H.3.4 [**Information Storage and Retrieval**]: Systems and Software – *Performance evaluation (efficiency and effectiveness).*

General Terms

Measurement, Performance, Experimentation.

Keywords

Communication-oriented, churn, Distributed Hash Table (DHT), Kademlia, energy consumption.

1. INTRODUCTION

"Peer-to-peer systems are distributed systems consisting of interconnected nodes able to self-organize into network topologies with the purpose of sharing resources such as content, CPU cycles, storage and bandwidth, capable of adapting to failures and accommodating transient populations of nodes while maintaining acceptable connectivity and performance, without requiring the intermediation or support of a global centralized server or authority" [1]. They have 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 application environments [1]: communication and collaboration, distributed computation, internet service support, database system, content distribution.

In this paper, we focus on the communication functionality of P2P systems. 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 protocol of 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. The used Distributed Hash Table (DHT) algorithm is Kademlia [5].

Our primary effort with P2PP is to evaluate its feasibility for mobile environments that are characterized by intermittent connections and limited bandwidth. Consequently, the churn effect is one of the primary problems to be studied. We simulate different levels of churn as the exponential distribution with varied value of mean online time. Meanwhile, we select the CPU load, network traffic load and battery consumption of mobile devices taking part in overlay as the main performance metrics to be evaluated. For the purpose of feasibility evaluation aforementioned, we have implemented a prototype that works both in Linux and Symbian operating systems (OS). The Linux version is responsible for simulating the majority of the peers of an overlay, while the Symbian version is used to provide overlay access for mobile nodes. To provide a comparison, both the Universal Mobile Telecommunications System (UMTS) and Wireless Local Area Network (WLAN) access modes are evaluated.

The contribution of this paper is as follows:

- We have implemented a prototype of communication-oriented P2P system that works both in Linux and Symbian OS.
- Through prototype measurements, 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 paper is structured as follows: Section II gives the related work; Section III introduces the system architecture and P2PP in general; Section IV presents the prototype and the associated CPU load, network traffic load and battery consumption of mobile phones. In Section V, we conclude the paper.

2. RELATED WORK

Communication-oriented P2P systems aim at facilitating direct 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 Session Initiation Protocol (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 setup delays in 3G and WLAN access networks. Their measurements showed that registration and call setup 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] 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. Cohrs *et al.* [13] presented a P2PP prototype implementation based on the Chord [14] algorithm. The associated functionality, performance and real-world applicability was proved by local single-machine simulation and PlanetLab measurement.

From the power consumption perspective, there are a few research efforts focusing on it. Gurun *et al.* [15] performed a study of energy consumption by utilizing a P2P chat application named Chimera on a Personal Digital Assistant (PDA) device. Their results showed that it was feasible to deploy light-weight P2P applications on low-power devices, e.g. PDA or mobile phones. Nurminen *et al.* [16] 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 was presented in 15] and [16]. Kelényi *et al.* [17] 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 [18], the same authors proposed some enhancements to reduce traffic and energy consumption by selectively dropping partial incoming messages. Kassinen *et al.* [19] 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 this paper, we extend the measurement of energy consumption to sending and receiving modes in parallel and measure complete test sets. Moreover, the CPU load percentage and the network traffic load of Nokia N95 devices acting as peers of a P2PSIP overlay are also measured.

3. SYSTEM ARCHITECTURE

This Section presents the description of the P2PP architecture in general and the Kademlia-based algorithm. Node and peer, power consumption and energy consumption, are used interchangeably in this paper respectively.

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 prototype implementation, we choose Kademlia as the fundamental DHT algorithm and make some modifications to it. It is noteworthy that the prototype implementation can support other DHT algorithms, e.g. Chord, without much effort. The following messages listed in Table 1 have been implemented.

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

The transport protocol utilized to carry the P2PP protocol is User Datagram Protocol (UDP). Accordingly, the ACK message is used in our prototype implementation to guarantee the reliability of message delivery. One transaction is made up of a Request, a Response and an ACK messages. One point to be noted here is that, if an ACK message follows directly a Response message, then the ACK message piggybacks onto the Response message to save network traffic. Furthermore, as we choose Kademlia as the underlying DHT algorithm, correspondingly, the iterative routing mechanism (the preference of Kademlia) is used in our 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

There is one centralized Bootstrap server existing in the prototype implementation that provides the associated Bootstrap functionality. A 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 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 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) than to itself. If any resource item satisfies this condition, the target node transfers it to the newly joined node by utilizing the Transfer message, and the joining process is completed. Here the Exclusive OR (XOR) metric is used to define the distance.

In the leaving process, the graceful leaving mechanism is implemented in our prototype. It means the leaving node transfers all the resource items it stores to its closest neighbor and notifies its leaving to all its routing neighbors by sending Leave messages.

One point to be noted here is that the node joining process is serial, each time only one Join request is sent, and accordingly, only one response message is received, which is slightly different from the original parallel mechanism used in Kademlia algorithm. We implement the serial process in this paper as it is common in other DHT algorithms, e.g. Chord. In reality, all the other processes in this paper are serial.

In this process, the Bootstrap, Join, Leave, Transfer 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 receipt of the Exchange request message, the requested node responds with an Exchange response message including at most 15 unduplicated routing items (the maximum number of routing items by the limitation of the UDP packet size) 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. For each routing item received through the Exchange response message, one additional KeepAlive request message is sent to check the aliveness of the node. If and only if the routing item replies a KeepAlive response message and the associated *k*-bucket of the requesting node is not full, the routing item is inserted in the associated *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 effectively 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 dedicated 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 into 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 latest 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 clear away stale routing items from their routing tables. Upon the receipt of a KeepAlive request message, the node just responds with a KeepAlive response message to prove its aliveness. If a node fails to receive a KeepAlive response message from one of its neighbors, the neighbor will be removed from the routing table.

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. Again, the serial mechanism is utilized here. The Publish and Lookup operations are highly similar to the Join operation aforementioned. One point to be noticed here is that, if the Publish or Lookup message is sent to an offline node, the entire Publish or Lookup process results 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 (this is how iterative routing mechanism works). Meanwhile, 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.

In this process, the Publish, Lookup and ACK messages are utilized.

4. IMPLEMENTATION AND MEASUREMENTS

We name our prototype as Mobile P2PP (MP2PP). It is written in C++ code and works on two platforms: Symbian Operating System (OS) Series 60 on the Nokia N95 smart-phone, and Ubuntu Linux on Sun Microsystems server hardware. In this Section, we introduce the prototype setup and the measured quantities, i.e. the CPU load percentage, the network traffic load and the battery consumption.

4.1 Prototype Setup

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 times follow the exponential distribution with a specified mean value that describes the level of churn used in a given experiment, and the mean online time (m_{online}) equals to the mean offline time ($m_{offline}$). Thus, when an overlay contains N peers, 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. 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 intervals for sending KeepAlive, Exchange, and Republish messages, i.e. $t_{keepalive}$, $t_{exchange}$, and $t_{republish}$, were 10s, 30s, and 30s, 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.

4.2 CPU Load Percentage

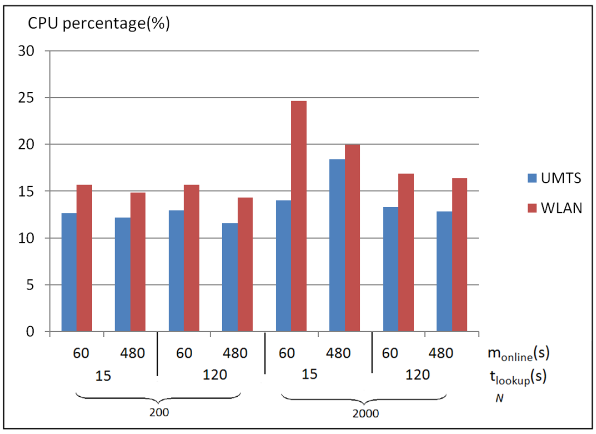


Figure 1. CPU load percentage of mobile nodes taking part in P2PP overlay.

In this Section, the CPU load percentage of mobile peers taking part in P2PP overlay is measured. The motivation is that mobile devices are characterized by limited CPU processing capability. If, for instance, a mobile node participating in 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 Figure 1 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}=*15s*, m_{online}=*60s*, N=*2000*), which has the maximum mean value in all the simulated combinations, is provided in Figure 2.

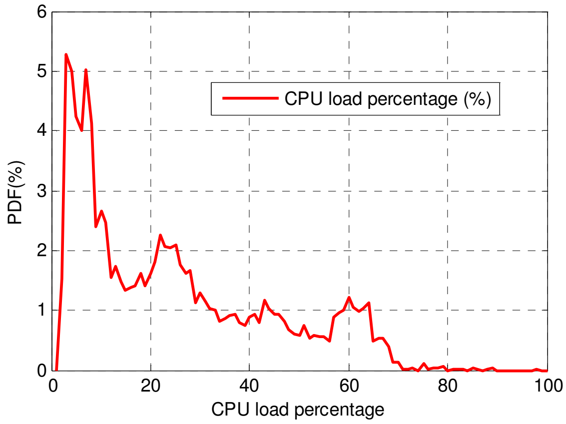


Figure2. Probability density function of CPU load percentage (t_{lookup}=15s, m_{online}=60s, N=2000).

From Figure 1, 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 2000 nodes scenario is also slightly higher in comparison with the 200 nodes scenario.

From Figure 2, we can see that the PDF of the CPU load percentage has a long tail. However, there are very few time points when the CPU load percentage is more than 70%; for the remaining time, the CPU load percentage is less than 70%.

Consequently, 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.

4.3 Network Traffic Load

In this Section, the uplink and downlink network traffic of mobile nodes taking part in P2PP overlay are measured. Just as an example, the results of the mean network traffic load in the UMTS access mode are shown in Figure 3. The network traffic load in the WLAN access mode is very similar to the UMTS access mode; accordingly, it is not shown in this paper. Again, the PDF of one of the combinations (t_{lookup}=15s, m_{online}=480s, N=2000), which has the maximum mean value (around 800bytes/s) in all the combinations, is provided in Figure 4.

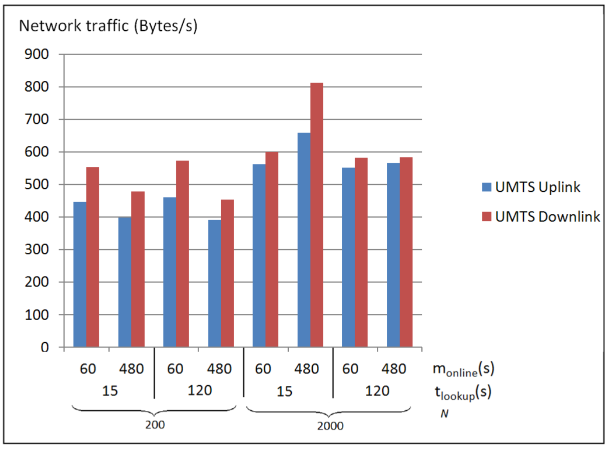


Figure3. Uplink and downlink network traffic load of mobile nodes taking part in P2PP overlay.

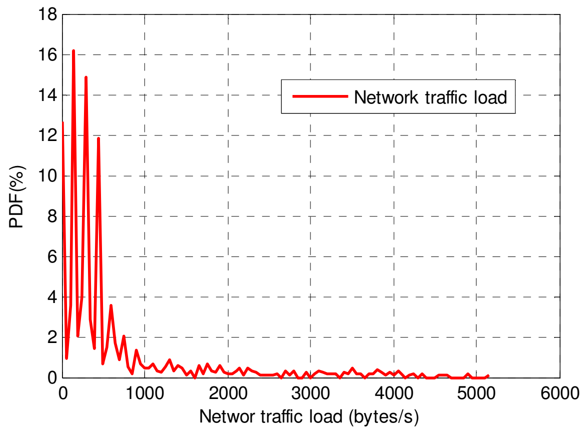


Figure4. Probability density function of network traffic load (t_{lookup}=15s, m_{online}=480s, N=2000).

From Figure 3, 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.

From Figure 4, it is clear that the PDF of the network traffic load has also a long tail. However, for the majority of the simulated time, the measured network traffic is less than 1000 bytes/s, which is a highly acceptable bandwidth requirement in the currently deployed mobile networks.

As a whole, from the viewpoint of network traffic load, it is highly feasible for mobile nodes to take part in P2PP overlay as the mean network traffic load is only several hundred bytes per second.

4.4 Power Consumption

Besides the CPU load percentage and network traffic load measurements shown in this paper, the power consumption measurements of mobile nodes (Nokia N95 mobile phones) taking part in the P2PP overlay were also conducted and the results were shown in [19]. 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.

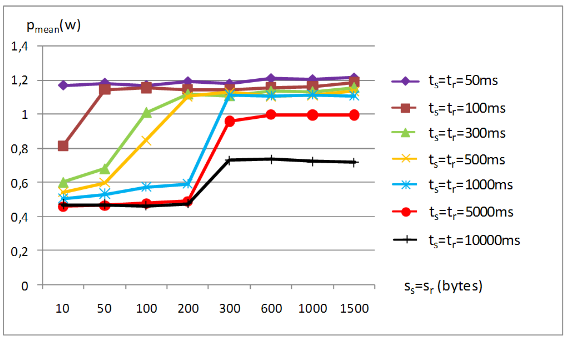
No overlay was utilized in these measurements; the mobile nodes just sent and received UDP packets of different sizes at different time intervals. When executing the measurements, besides the sending and receiving operations of UDP packets, only the Nokia Energy Profiler software was running in Nokia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

Table 2. Notations and their meaning

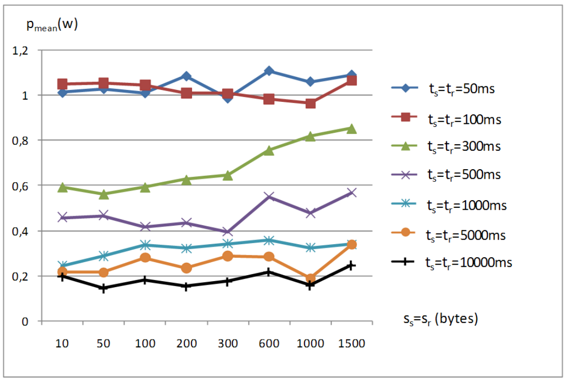
Notation	Meaning
t_s	The intervals for sending UDP packets, $t_s \in \{50,100,300,500,1000,5000,10000\}\,ms$
t_r	The intervals for receiving UDP packets, $t_r \in \{50,100,300,500,1000,5000,10000\}\,ms$
s_s	Packet sizes of the sent messages, $s_s \in \{10,50,100,200,300,600,1000,1500\}\,bytes$
s_r	Packet 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).

4.4.1 *Symmetric measurements*

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



(a)



(b)

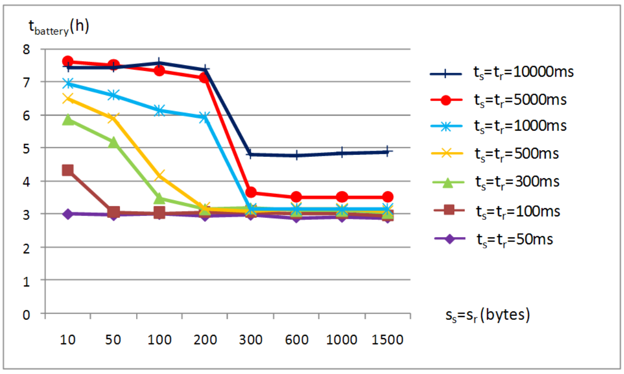
Figure5. Power consumption of Nokia N95 smart phone. (a) UMTS. (b) WLAN.

The power consumption of Nokia N95 smart phones under different levels of network activity is shown in Figure 5. One point to be noted here is that the *Idle* state of mobile phones (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 Figure 5-Figure 8 have excluded the corresponding power consumption (*0.0649 w*).

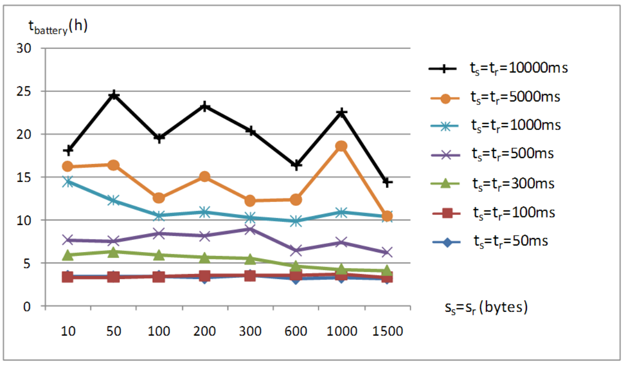
In order to calculate the battery life of a mobile phone under associated network activities, the following formula is utilized.

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

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 3.7V. 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 Figure 6. In real life, when the other network activities are being executed simultaneously, the battery life will be much less than this.



(a)



(b)

Figure6. Battery life of Nokia N95 smart phone. (a) UMTS. (b) WLAN.

From Figure 5, 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=*10000ms*", "t_s=t_r=*5000ms*", and "t_s=t_r=*1000ms*" cases. The gap takes place between the 50 and 200 bytes for the cases "t_s=t_r=*500ms*" and "t_s=t_r=*300ms*", while for the case "t_s=t_r=*100ms*" 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 small packet size range, the power consumption of most of the transmission intervals converges to around *0.5 w*. The power consumption mainly comes from the packet headers. In 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, instead the sending and receiving intervals do. The overall trend is that the power consumption in WLAN access mode decreases as the sending and receiving intervals increase.

In Figure 6, 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 3 hours to 8 hours, while in WLAN it varies from 3 hours to 25 hours. In a context with high level of network activity, i.e. "t_s=t_r=*50ms*" and "t_s=t_r=*100ms*", there is no significant difference between UMTS and WLAN access modes. The battery life in these two cases is around 3 hours in both UMTS and WLAN access modes.

4.4.2 Asymmetric measurements

In the previous Section, we measured the symmetric network activities where the packet sizes and intervals for sending and receiving messages are the same. To analyze the effect of the imbalanced network activity to 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. 100bytes and 1500bytes (for both sending and receiving, asymmetrically), to evaluate the effect of asymmetric sending and receiving packet sizes to the power consumption, the results are shown in Figure 7. In the other case, we keep the sending and receiving packet sizes as fixed, i.e. "s_s=s_r=*100bytes*", and evaluate the effect of asymmetric sending and receiving intervals to the power consumption, the results are shown in Figure 8. One noteworthy point here is that the x-axis of Figure 8 stands for t_s or t_r. For the two cases, i.e. "*t_s=50ms, UMTS*" and "*t_s=50ms, WLAN*", the x-axis stands for the values of t_r. For the other two cases, i.e. "*t_r=50ms, UMTS*" and "*t_r=50ms, WLAN*", the x-axis stands for the values of t_s. We integrate these 4 curves into one figure just to make a clear comparison among them.

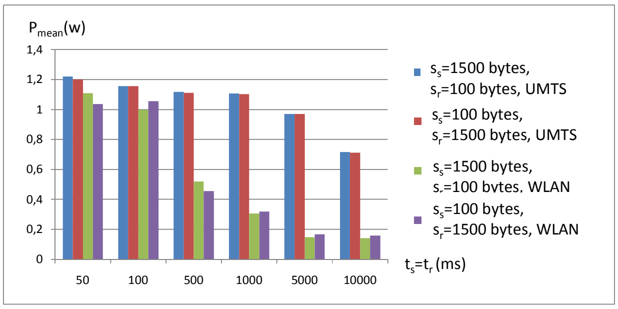


Figure7. Effect of asymmetric sending and receiving packet sizes to the power consumption.

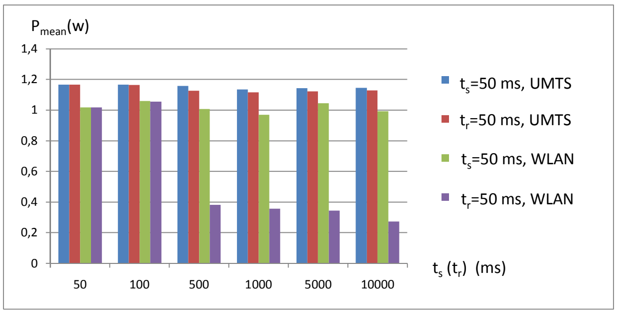


Figure8. Effect of asymmetric sending and receiving time intervals to the power consumption.

From Figure 7, 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 have equal values. 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.

From Figure 8, 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 Figure 8, if the time interval is larger than 500 ms, the power consumption in the case "*t_r=50ms, WLAN*" is much less than in the case "*t_s=50ms, WLAN*". One possible explanation for this is that, after some critical point, e.g. 500ms, 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=50ms, WLAN*" case consumes less power than the case "*t_s=50ms, WLAN*" does. Again, the same trend, i.e. the UMTS network access mode consumes more power than the WLAN access mode, is observed here.

5. CONCLUSION

This paper presented a feasibility evaluation of a communication-oriented P2P system in mobile environments. We used Kademlia-based P2PP as an example signaling protocol. 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 made in this paper.

6. ACKNOWLEDGMENTS

The authors would like to thank TEKES (the Finnish Funding Agency for Technology and Innovation), Ericsson, Nokia and Nethawk for their financial support in Decicom project. 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.

7. REFERENCES

[1] S. Androutsellis-Theotokis, D. Spinellis, A survey of peer-to-peer content distribution technologies, ACM Computing Surveys, Vol. 36, No. 4, December 2004, pp: 335–371.

[2] S. Baset, H. Schulzrinne, M. Matuszewski, Peer-to-Peer Protocol (P2PP), draft-baset-p2psip-p2pp-01, November 19, 2007. (work-in-progress)

[3] P2PSIP Working Group, online: 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, March 07, 2009. (work-in-progress)

[5] P. Maymounkov and 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] Saint-Andre, P., Streaming XML with Jabber/XMPP, IEEE Internet Computing, Volume 9, Issue 5, Sept.-Oct. 2005, pp: 82 – 89.

[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, August 27–31, 2007, Kyoto, Japan, pp:37-48.

[8] J.-F. Wauthy, L.Schumacher, Implementation and Performance evaluation of a P2PSIP Distributed

Proxy/Registrar, in The 2007 International Conference on Next Generation Mobile Applications, Services and Technologies, 2007. NGMAST '07. 12-14 Sept. 2007, pp: 119 – 124.

[9] M. Matuszewski, E. Kokkonen, Mobile P2PSIP - Peer-to-Peer SIP Communication in Mobile Communities, in 5th IEEE Consumer Communications and Networking Conference, 2008. CCNC 2008. 10-12 Jan. 2008 pp: 1159 – 1165.

[10] E.Kokkonen, S.Baset, M.Matuszewski, Demonstration of Peer-to-Peer Session Initiation Protocol (P2PSIP) in the Mobile Environment, 5th IEEE Consumer Communications and Networking Conference, 2008. CCNC 2008. 10-12 Jan. 2008, pp: 1221 – 1222.

[11] I.Baumgart, P2PNS: A Secure Distributed Name Service for P2PSIP, Sixth Annual IEEE International Conference on Pervasive Computing and Communications, 2008. PerCom 2008. 17-21 March 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, The Second International Conference on Next Generation Mobile Applications, Services and Technologies, 2008. NGMAST '08. 16-19 Sept. 2008, pp: 543 – 549.

[13] 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.

[14] 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, Volume 11, Issue 1, Feb. 2003, pp: 17 - 32.

[15] S. Gurun, P. Nagpurkar, and B. Y. Zhao, "Energy consumption and conservation in mobile peer-to-peer systems," in Proc. First International Workshop on Decentralized Resource Sharing in Mobile Computing and Networking, 2006, pp. 18-23.

[16] J. K. Nurminen and J. Nöyränen, "Energy-Consumption in Mobile Peer-to-Peer – Quantitative Results from File Sharing," in Proc. of Fifth IEEE Consumer Communications & Networking Conference (CCNC) 2008, pp: 729-733.

[17] I. Kelényi and J. K. Nurminen, "Energy aspects of peer cooperation—Measurements with a mobile DHT system," in Proc. Cognitive and Cooperative Wireless Networks Workshop in the IEEE International Conference on Communications 2008, Beijing, China, 2008, pp. 164–168.

[18] I. Kelényi and J. K. Nurminen, "Optimizing energy consumption of mobile nodes in heterogeneous Kademlia based distributed hash tables," in Proc. 2nd International Conference and Exhibition on Next Generation Mobile Applications, Services and Technologies, Cardiff, Wales, UK, 2008, pp. 70–75.

[19] O. Kassinen, E. Harjula, J. Korhonen, and M. Ylianttila, Analysis of Resource Access in a P2PSIP Overlay under Churn, submitted for publication.