

Experimenting with Opportunistic Networking

Anna-Kaisa Pietiläinen Christophe Diot

Thomson, Paris

[annakaisa.pietilainen,christophe.diot]@thomson.net

ABSTRACT

Pocket Switched Networks (PSN) make use of opportunistic encounters between mobile devices to disseminate content. Most of the prior measurement research on PSN has concentrated on understanding human mobility and its impact on forwarding algorithms for PSN. In this study we go beyond simple contact measurements and implement a software to measure the real-life performance of opportunistic networking using a smartphone platform and Bluetooth. We perform three user trials with 22 to 29 participants in networking conferences. We report on our experimental PSN software, characteristics of opportunistic contacts, communications performance, and discuss the potential capacity of PSN.

Categories and Subject Descriptors

C.4 [Performance of Systems]: measurement techniques
; C.2.1 [Computer-Communication Networks]: Network Architecture and Design—*store and forward networks*

General Terms

Measurement, Experimentation

Keywords

Opportunistic communications, Bluetooth

1. INTRODUCTION

Pocket Switched Networks (PSN) [11] is a communication paradigm that exploits contact opportunities between mobile devices and human mobility to transfer data in a peer-to-peer fashion. PSN is an attractive communication model when infrastructure connectivity is intermittent, when its usage could be undesirable because of price or privacy issues, or when one wants simply exchange data with people located in the neighborhood. Opportunistic communications have gained interest recently due to the increasing number of

mobile devices people carry today; and their power in terms of CPU, storage space and available networking interfaces.

Prior work on PSNs has focused on understanding the impact of human mobility on the performance of forwarding algorithms [3]. Human mobility has been studied using WiFi enabled laptops and computers [5], cell phones [4] and iMotes [6] (a small Bluetooth sensor running on an ARM processor with very limited battery and memory). Most data sets are available on Crawdad¹.

While the existing studies have improved our understanding of some of the fundamental limits and possibilities of PSN, we believe that implementing and deploying real PSN systems is equally important to understand user behaviour, potential applications, configuration parameters and to collect richer traces to drive future research. Only few systems and applications relying on opportunistic contacts between mobile devices have been deployed and analyzed to date [8, 7]. The main reason is that experimenting with real PSNs introduces several challenges: (1) to motivate people to participate in the experiments, and to generate some real application traffic, we need compelling and easy to use applications and incentive mechanisms; (2) for experimentation we need to find suitable events that would be realistic in terms of human mobility and social interaction; (3) network delays, disconnections and heterogeneous interfaces typical for a PSN require a non-traditional approach to the communications system design; and (4) the actual implementation requires a lot of effort depending on the chosen mobile device platform(s) and development framework.

In this work, we present our approach to experimenting with opportunistic communications. The main goal of our work is to extend the previous iMote experiments to collect data on contact opportunities and ad hoc data communications between mobile devices using Bluetooth radio interface. We design and implement a measurement software for Windows Mobile platform. We discuss the setup and results of three real-life user trials we have performed in conference environments. The results highlight some limitations of Bluetooth technology for opportunistic networking including low device discovery rate and failures to setup connections. However, we obtain also encouraging results on the potential capacity offered by PSNs. We will continue the development and optimization of our PSN framework based on the experiments and we hope our experiences are also helpful for others planning to design and deploy similar systems.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

MobiArch'09, June 22, 2009, Kraków, Poland.

Copyright 2009 ACM 978-1-60558-688-5/09/06 ...\$10.00.

¹<http://crawdad.cs.dartmouth.edu/>

2. TESTBED PLATFORM

2.1 Mobile Device

We conduct our experiments using HTC s620 Windows Mobile Smartphones². This device has a 200MHz TI processor, 64MB of RAM, 128MB of ROM and a MicroSD slot. The radio interfaces include a quad-band GSM/EDGE cellular radio, Bluetooth v1.2 and 802.11b/g.

2.2 Measurement Software

Windows Mobile offers full access to the low-level networking functions through the Win32 API using standard C++ which we use to implement our experimental PSN software. The software executes a simple periodic loop consisting of three phases: (1) neighboring devices discovery using available (wireless) networking interfaces (DISCOVERY), (2) device identification and ad hoc connection setup (IDENTIFICATION), and (3) data exchange (TRANSFER). The neighborhood discovery is performed in the beginning each loop whereas the connection setup and data exchange configuration varies between the experiments (see next section). The software records all the discovered devices, connection setup and data transmission details on a log file on the device memory. The software runs in the background and is restarted automatically upon reboot.

The Bluetooth neighborhood discovery is implemented using the Bluetooth device inquiry function. The Win32 APIs allow us to configure the inquiry duration (the recommended duration by the Bluetooth standard is 10.24s which is used in our experiments [2]), the maximum number of devices to discover per inquiry, and to enable or disable the device friendly name query in the inquiry phase.

The Win32 API has a method to query the WiFi networks (both infrastructure and ad hoc) known to the operating system. By default Windows Mobile uses both active and passive network discovery and does not allow manual configuration of the discovery process. The actual neighboring device discovery over WiFi requires the devices to be associated with a same infrastructure or ad hoc network simultaneously. Once in a same IP subnet the devices can use UDP broadcast packets to inform of their presence and to communicate basic device identification information.

2.3 Micro Benchmarks

We perform a set of micro benchmarks in a laboratory environment to characterize WiFi and Bluetooth interfaces for ad hoc communication.

Figure 1 plots the available battery life (percentage) as the device performs a neighborhood discovery periodically every 120 seconds until the battery drains out. Bluetooth discovery is configured to last 10.24 seconds and the interface is powered on during the whole benchmark. The WiFi discovery in the benchmark consists only of the discovery of available SSIDs. For this we turn on the WiFi interface for 5 seconds during each discovery period. When both interfaces are enabled, the device's battery lasts for over 60 hours; with WiFi only we reach a life-time of 100 hours; and with Bluetooth only over 120 hours³. The comparison favours WiFi as the interface is not powered between the

²www.htc.com/product/03-product_s620.htm

³For comparison, the standby time reported by the manufacturer is up to 220 hours (9 days) and talk time up to 5 hours.

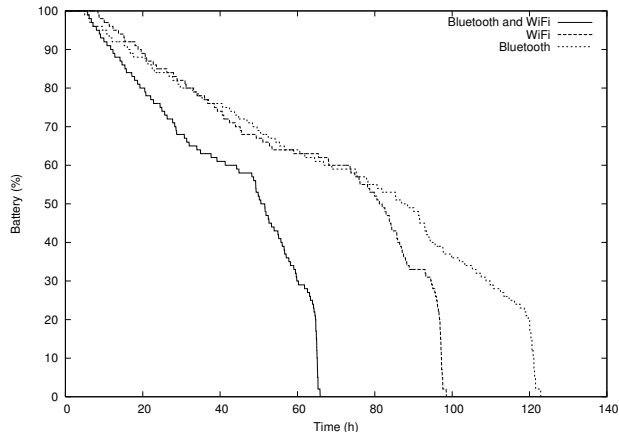


Figure 1: HTC s620 device battery life-time during continuous periodic opportunistic contact discovery.

discoveries and because we do not perform the actual device discovery (this would require the interface to be on to listen to the broadcast beacons). When enabling the WiFi interface continuously for device discovery and data transmissions the device's battery life drops below 2 hours. With Bluetooth we still get over 10-hour life-time.

From Figure 1 we can also observe that the reported battery life-time does not decrease linearly. Instead, we notice that the draining rate accelerates after 60% mark, and again at 20% after which the battery runs out of energy in less than an hour. This is most likely a Windows Mobile API specific behaviour, but nevertheless should be taken into account when designing power saving policies for these devices.

WiFi outperforms Bluetooth in terms of discovery time and data transmission speeds. Bluetooth device discovery takes the previously mentioned 10.24s (can be configured to be shorter, but leads to missed contacts). The Bluetooth service discovery and the RFCOMM connection setup for data communications add several seconds of delay per device in the neighborhood. In comparison, associating with a WiFi network takes on average 4 seconds (using Windows Mobile connection manager), and sending and receiving broadcast beacons over WiFi for device discovery is almost instantaneous. We measure an average data transmission rate of 50KB/s for Bluetooth, and between 100KB/s and 500KB/s for WiFi when sending continuous bulk data stream between devices. Finally, WiFi and Bluetooth interfaces have also a very different communications range: the 802.11 radio range varies from about 35 to 140 meters (indoors and outdoors respectively). The Bluetooth range (class 2 devices) is 10 or some meters more depending on the hardware.

We are constrained to use Bluetooth in our user trials because we cannot require people to carry test devices that need to be charged every few hours. We consider the range and the effective bandwidth of Bluetooth to be sufficient for the initial experimental settings.

3. EXPERIMENTAL SETUP

We choose conference environment for the convenience of distributing and retrieving the mobile devices running our software, and for comparability between different ex-

	MASS'07	CoNEXT'07	CoNEXT'08
Duration	3.5 days	3.5 days	3.5 days
Participants	29	28	22
Bluetooth device discovery	10.24s every 120s	10.24s every 120s	10.24s every 120 +/- 45s
Bluetooth service discovery	-	-	yes
Average active time per device	66.28h	56.91h	56.70h
Average inactive time per device	17.94h (27.07%)	30.72h (49.80%)	22.19h (30.07%)
No. of devices per inquiry (μ/median/σ)	3.69/2.00/4.63	2.91/1.00/3.61	3.37/1.00/4.33
8am - 6pm	5.56/4.00/5.17	4.78/4.00/3.96	5.24/4.00/4.99
No. of internal devices per inquiry (μ/median/σ)	1.77/1.00/2.63	1.87/0.00/2.75	1.64/1.00/2.19
8am - 6pm	2.71/1.00/3.09	3.31/3.00/3.11	2.42/2.00/2.38
No. of contacts per inquiry (μ/median/σ)	4.63/2.00/5.66	3.98/2.00/4.72	4.26/2.00/5.20
8am - 6pm	7.05/5.00/6.20	6.60/6.00/4.98	6.67/6.00/5.83
No. of internal contacts per inquiry (μ/median/σ)	2.27/1.00/3.25	2.63/0.00/3.67	2.16/1.00/2.71
8am - 6pm	3.53/2.00/3.77	4.70/4.00/4.01	3.24/3.00/2.89

Table 1: The user trials configuration and aggregate statistics. *Internal* refers to the devices participating to the experiment. μ denotes the sample mean and σ the sample standard deviation.

periments and previous work. The movement of conference members is also geographically constrained which creates contact opportunities we aim to exploit and measure.

We distribute the devices to volunteers during the first day of the conference and collect them back with the log files on the last day of the conference. The experimentalists can use the device as their personal device (i.e., they can choose to put their SIM card in the device) during the experiment but this is not required. The experimentalists are instructed to keep the device charged and with them at all times. Below we discuss the configuration of each experiment in detail.

MASS'07: We perform a periodic Bluetooth device discovery every two minutes for 10.24s. The device tries to form Bluetooth RFCOMM connections to discovered participating devices and send data until the next inquiry period. The list of participating devices is hard-coded in the software (meaning we exclude the IDENTIFICATION phase). In the TRANSFER phase, we perform two types of data transfers: sending 50KB chunk of data to each neighboring experimental device (fragment test), or sending 1MB to a single neighboring device selected at random (bulk test). The fragment test connects to each neighbor one after another.

CoNEXT'07: The neighborhood discovery is performed using the same parameters as in the MASS'07 experiment. In contrast, we do not hard-code the experimental devices MAC addresses in the software. Instead, in the IDENTIFICATION phase the software tries to connect to all devices found by the device inquiry one after another to identify devices participating to the experiment. For this each device is listening on incoming connections on a well-known RFCOMM channel. Upon successful connection, the device exchange simple handshake messages containing further identification details and the number of available data messages. The identification is followed by the TRANSFER phase where the devices exchange available data messages. The data messages are generated by a background thread that inserts 50KB synthetic data messages every 10 minutes to the device's message buffer. The data exchange phase is executed sequentially with all the devices in the neighborhood. This effectively limits the number of simultaneous RFCOMM links to one outgoing connection per device. The data messages are destined either to a single device or a group of devices and they are flooded in the PSN over multiple hops. This

configuration means that on each contact we may send a variable amount of messages to the discovered devices depending on how many messages the sender is holding in his buffer. The message life-time is controlled using an absolute time-to-live (TTL) of one hour. When the TTL expires, the messages are removed from the devices buffers.

CoNEXT'08: The third experiment makes several modifications to the Bluetooth communications implementation: (1) Bluetooth device inquiries are randomized and performed every 120 s +/- 45 s (chosen uniformly at random); (2) we use Bluetooth service discovery to identify devices running our software prior to a connection setup; (3) we keep a small cache of two scan periods of discovered devices to be more resistant to device inquiry failures; (4) we increase the maximum number of simultaneous RFCOMM links to three; and (5) when sending data over RFCOMM, we repeat each connection attempt three times in case of failure before abandoning the operation. We also send test traffic as in the previous experiment. The test messages are generated every 60 minutes with variable content: empty (includes just a simple header); 58B (text file); 63,847B (pdf); 589,723B (jpg image); 1,023,380B (binary file) and 7,327,436B (mp3). The TTL of the test messages is set to 24 hours.

4. RESULTS AND ANALYSIS

The experiments configuration and general statistics are summarized in Table 1. The *active time* per device is defined as the average time between the first and the last recorded activity on an experimental device. The active time is between 56 and 66 hours which is shorter than the full duration of the experiments. The difference is mainly due to the variations in arrival and departure times of the participants. The *inactive time* is the percentage of time that the device is not collecting data during the experiment. The inactive time is high in all the experiments mainly due to battery depletion and some software crashes. As most of the participants do not put their SIM cards on the device, the motivation to keep the device up and running remains lower than with real personal devices. The experimental software also adds considerably to the energy consumption due to frequent Bluetooth operations and SD card I/O. The devices run approximately 8 - 10 hours on each recharge while running the measurement software in the experimental settings.

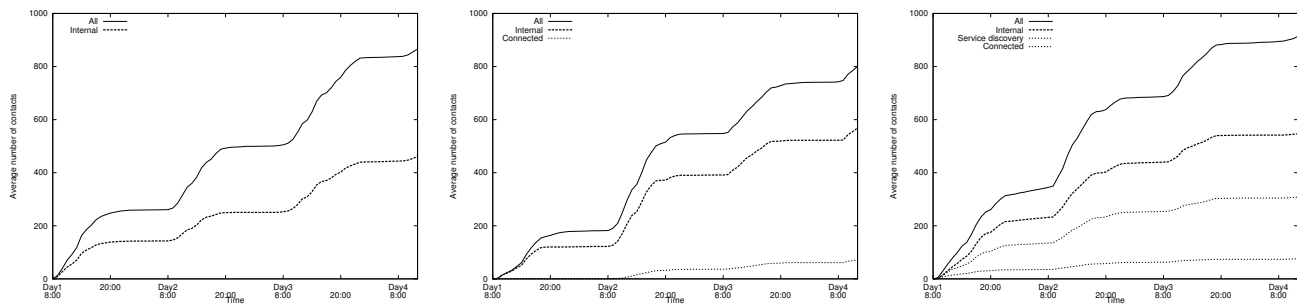


Figure 2: Cumulative average number of contacts per participant at MASS'07 (left), CoNEXT'07 (middle) and CoNEXT'08 (right).

4.1 Contact Characteristics

We start the analysis by inspecting the opportunistic contacts discovered during the DISCOVERY phase. In Table 1 we show the aggregate statistics for the device inquiry considering the the whole data set and conference hours (8am to 6pm) when most of the encounters take place. The number of discovered devices per inquiry is 5 on average in each experiment during the conference hours; among which 2 - 3 are devices participating to the experiment (referred as *internal* in the rest of the paper). Communication opportunities arise regularly in each experiment, and over 50% of the time (median), the devices have more than one potential contact in their neighborhood.

For the rest of the analysis, we define a *contact* as a time interval between the first time a neighboring device answers to the Bluetooth device inquiry until it is no more present during two consecutive inquiries. This definition complies with previous works [6, 9] providing some robustness to the problem of missed Bluetooth devices due to probabilistic nature of the inquiry mechanism, simultaneous inquiries and interference between various Bluetooth operations [1]. With this definition, we obtain a more realistic representation of the neighborhood. The average size of the neighborhood increases compared to the set of devices that answer the inquiry during a single round: at MASS'07 there are potentially 3.53 internal devices in the neighborhood, at CoNEXT'07 4.70 and at CoNEXT'08 3.24.

We plot the cumulative average number of contact opportunities between any discovered device and the experimental devices only (labeled *internal*) in Figure 2. We observe that the contact trends are very similar in all of the experiments. The number of internal contacts is on average approximately 200 per day for participating devices. At CoNEXT'07 we experience some software problems during the first day which shows up as a slow growth in the beginning of the experiment. The figure also clearly shows how most of the contacts take place during the conference hours while at night time the growth rate is flat in all the experiments according to the diurnal human behaviour.

Finally, we plot the distribution of contact times (Figure 3). We only include the contacts between participating devices during conference hours as the contacts between any device follow similar distributions in each experiment. The median contact time is 334s at MASS'07 and 129s in the other experiments. Figure 4 summarizes the contact opportunities evolution. It shows the average cumulative total

contact time between participating device over the experiment. During the conference hours, each device is in contact with other participating devices on average for 13.6 hours at MASS'07, 10.6 hours at CoNEXT'07 and 13.3 hours at CoNEXT'08. The actual capacity of a PSN is defined by the contact frequency and durations, which we will discuss further in the last section.

4.2 Communications Performance

After the DISCOVERY phase, the measurement software proceeds to the IDENTIFICATION and TRANSFER phases where it performs the ad hoc communications link setup and actual data transmission. At CoNEXT'08 we rely on the Bluetooth service discovery to identify the devices running the software before a connection attempt. The number of devices discovered by the service search should follow that of the number of experimental devices in the neighborhood. However, figure 2 shows that our software correctly identifies only 56.4% of the internal contacts using the service discovery. The service discovery fails mainly for two reasons: (1) it adds a delay to the discovery for each device as the service search is performed sequentially (taking 46.9ms on average when successful and 223.2ms otherwise) and thus devices can move out of range; and (2) it is vulnerable to interference as other Bluetooth operations. These results suggest to disable the

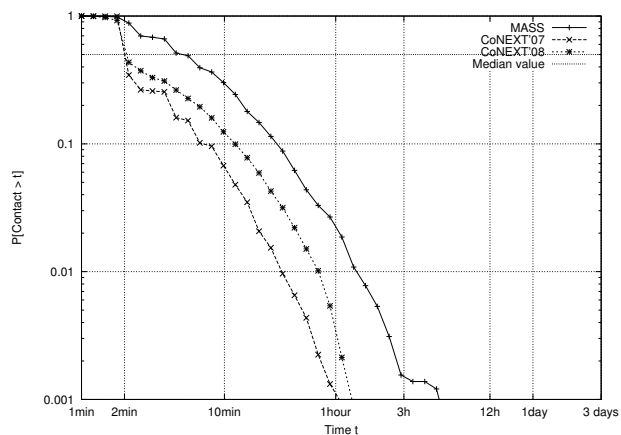


Figure 3: Conference hour contact time distribution between participating devices.

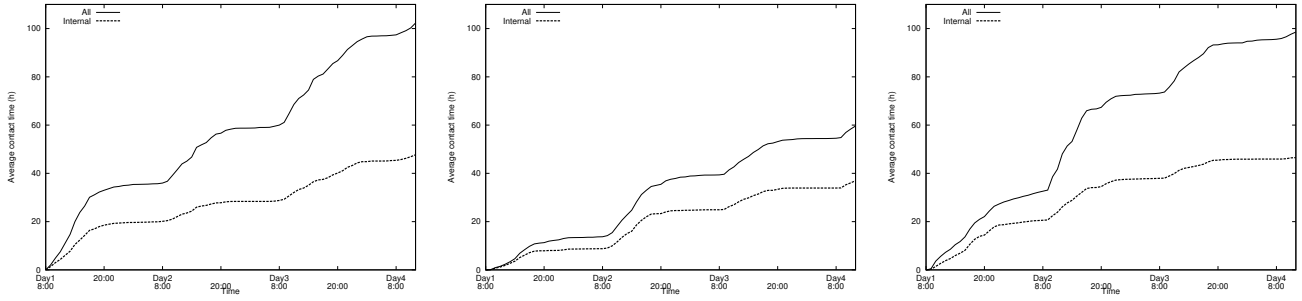


Figure 4: Cumulative average contact time per device (in hours) at MASS’07 (left), CoNEXT’07 (middle) and CoNEXT’08 (right).

service search in PSN settings. Instead, we could include the service information in the Bluetooth minor device class code which is readily available through the Bluetooth device discovery and is currently unused.

Figure 2 indicates the number of contacts during which the software actually manages to establish a RFCOMM link at CoNEXT’07 and CoNEXT’08 (labeled *connected*). The number of connected devices should be identical to the number of internal contacts as in both CoNEXT experiments the test software tries to connect to each device in the neighborhood for identification and data exchange. However, at CoNEXT’07 we manage to successfully connect to the discovered devices in only 12.5% of the contacts. Note in the figure that during the first day we have almost no successful connections which is due to the previously mentioned problem with the software. If we ignore the first day, we obtain a connection success rate of 15.4%. At CoNEXT’08 the respective success rate is lower, 13.8%, despite increased number of connection attempts (3 against 1). In the MASS’07 experiment we did not systematically open a connection during every contact but we varied the data exchange between bulk and the fragment tests. The success rate of the fragment test is 44% and for the bulk test 57%. The delay to setup a RFCOMM link is on average 3.36 seconds (CoNEXT’08). However, the common connection failures add additional highly variable delays to the system: our data shows an average delay of 13.6 seconds (standard deviation of 13.9 seconds) for a failing connection attempt.

In conclusion, we observe that the performance of Bluetooth decreases when the number of Bluetooth operations we try to perform increases. At MASS’07 we only execute a periodic device inquiry after which the internal devices are identified using a hard-coded list, and consequently, we only try to connect to the participating devices for the data transmission test achieving the best connection success rate among the experiments. In contrast, at CoNEXT’07 we try to connect to any discovered device, which adds noise and wasted connection attempts (to external devices). At CoNEXT’08 we try to avoid this problem with the service discovery but we do not achieve any performance gain compared to CoNEXT’07 due to additional interference and delays; and consequently failing service searches.

4.3 Network Capacity

In this section we discuss the “potential” capacity of a PSN. We define the *potential capacity* as the average amount

of data a single device could send to the other participating devices in its neighborhood during a day. The capacity is characterised by the available bandwidth and the frequency and duration of the opportunistic contacts. Considering the data transmission rates we measure in our experiments, 23.71 KB/s on average at MASS’07 and 6.15 KB/s at CoNEXT’08 (for CoNEXT’07 we do not have the complete data for this metric), we obtain a simple quantitative measure of the capacity: the average potential capacity is 1.1GB at MASS’07 and at the CoNEXT’08 experiment 288 MB of data per device per day. Even with the lower rate, the system can provide a useful service relying only on opportunistic contacts. The higher data rates observed at MASS’07 result partly from the lower number of potential simultaneous transfers (one in contrast to three) and reduced interference.

However, the current prototype system is still far from fully exploiting the potential capacity of the network. At MASS’07, each device sends on average 252.8MB (6.4% of the potential capacity) over the course of the experiment (3.5 days) and at CoNEXT’08 the respective amount is 51.6MB (5.1% of the potential). In order to increase the capacity of such a PSN, we can either improve the device discovery and connection setup performance (DISCOVERY and IDENTIFICATION phases) or increase the bandwidth available in the TRANSFER phase. We identify several potential improvements. First, the contacts used above to calculate the capacity are only estimates of the actual number and duration of contacts. The Bluetooth device inquiry does not find all the devices in the neighborhood during each inquiry, and the service discovery, if used, lowers the success rate even more. As suggested above, we could avoid the service discovery by using the Bluetooth device code instead. Keeping a contact cache (following our definition of contact) can also provide the system with a more realistic view of the neighborhood. Second, the current prototype performs inquiries relatively often. In the future versions we plan to analyse further the impact of performing the inquiries less often or adapting the inquiry period to the environment and time of day. Third, the performance could be improved also by exchanging the current neighbor lists between devices as they encounter. However, this needs also more experiments as the success of the connections is not guaranteed either. Note that the methods proposed above can also indirectly improve the connection success rates as the Bluetooth interference decreases when we execute fewer operations simultaneously. Indeed, maximizing the capacity requires that the system finds a per-

fect balance between the neighborhood discovery frequency and the actual communications phase.

The bandwidth depends on the quality of the radio link and the number of other simultaneous connections. In Bluetooth, the devices share the bandwidth equally, and by limiting the number of simultaneous connections, we can increase the capacity. While WiFi interface is already available on the devices, the main constraint is the high energy consumption. An interesting idea to overcome this problem is to use hierarchies of multiple radio interfaces [10]. Applied to our system, we could use Bluetooth for the discovery and initial negotiation phases, and switch on WiFi for the actual data transmission if both sides agree.

While actual PSN forwarding algorithms are out of the scope of this work, the limited communications time and bandwidth in opportunistic networking setting also require an efficient messaging protocol that does not waste resources on unnecessary exchanges. The current prototype used basic text based exchanges for identification and controlling the message forwarding. A generic opportunistic messaging protocol should communicate the current status of the message buffer, other resource limitations such as battery state and available interfaces, and information needed by the forwarding algorithms upon encountering other devices. In order to use minimize the resource usage, it might be useful to cache this information, or some parts of it, on the devices over multiple contacts.

5. CONCLUSION AND FUTURE WORK

In this paper we have taken an experimental approach to understand the potential capacity and feasibility of Pocket Switched Networks using a common smartphone platform and Bluetooth radio technology. We have designed a measurement software and used it in three conference experiments with 22-29 users. Using the collected data, we have characterized the opportunistic contacts, the actual communications performance and given an idea of the potential capacity of PSN.

Bluetooth technology has several advantages for opportunistic communications such as low power requirement, short radio range and wide availability. However, as a cable replacement technology, Bluetooth was not designed for applications relying on opportunistic ad hoc contacts. We discussed several ideas to overcome the limitations of Bluetooth to increase the available and used network capacity including use of Bluetooth class of device to communicate the basic service availability, contact caching, less frequent or adaptive neighborhood discovery, and radio hierarchies. In the future, other radio technologies, such as WiFi, could be used instead.

In addition to the communications system design considerations, our work raises a more general question related to the design and implementation of opportunistic communications protocols. While the contact opportunities arise regularly in the type of environment we have studied, they are often simultaneous, short and limited by the available bandwidth and other resources on the mobile device such as battery and storage. An efficient opportunistic communications protocol should aim to maximize the available time for data transmissions, and consequently, the usage of the potential capacity of PSN.

We continue the development of our experimental software to explore the system and protocol design issues discussed

in this paper. We will study more in detail the key parameters using laboratory benchmarks and we plan to execute a large scale experiment (100+ participants). We will make all the data sets available on Crawdad to promote further experimentation with PSN.

6. REFERENCES

- [1] S. Asthana and D. N. Kalofonos. The problem of bluetooth pollution and accelerating connectivity in bluetooth ad-hoc networks. In *PERCOM '05: Proceedings of the Third IEEE International Conference on Pervasive Computing and Communications*, 2005.
- [2] Bluetooth Special Interest Group (SIG). *Specification of the Bluetooth System. Core Package version: 1.2*, November 2003.
- [3] A. Chaintreau, P. Hui, J. Scott, R. Gass, J. Crowcroft, and C. Diot. Impact of human mobility on opportunistic forwarding algorithms. *IEEE Transactions on Mobile Computing*, 6(6), June 2007.
- [4] N. Eagle and A. Pentland. Reality mining: Sensing complex social systems. *Journal of Personal and Ubiquitous Computing*, 2005.
- [5] T. Henderson, D. Kotz, and I. Abyzov. The changing usage of a mature campus-wide wireless network. In *MobiCom '04: Proceedings of the 10th annual international conference on Mobile computing and networking*, 2004.
- [6] P. Hui, A. Chaintreau, J. Scott, R. Gass, and J. C. Diot. Pocket switched networks and the consequences of human mobility in conference environments. In *Proceedings of ACM SIGCOMM first workshop on delay tolerant networking and related topics*, 2005.
- [7] S. Jung, U. Lee, A. Chang, D.-K. Cho, and M. Gerla. Bluetorrent: Cooperative content sharing for bluetooth users. In *PERCOM '07: Proceedings of the Fifth IEEE International Conference on Pervasive Computing and Communications*, 2007.
- [8] M. May, V. Lenders, G. Karlsson, and C. Wachha. Wireless opportunistic podcasting: implementation and design tradeoffs. In *CHANTS '07: Proceedings of the second workshop on Challenged networks* CHANTS, 2007.
- [9] E. Nordström, C. Diot, R. Grass, and P. Gunningberg. Experiences from measuring human mobility using bluetooth inquiring devices. In *MobiEval '07: System Evaluation for Mobile Platforms*, 2007.
- [10] T. Pering, Y. Agarwal, R. Gupta, and R. Want. Coolspots: reducing the power consumption of wireless mobile devices with multiple radio interfaces. In *MobiSys '06: Proceedings of the 4th international conference on Mobile systems, applications and services*, 2006.
- [11] J. Su, J. Scott, P. Hui, J. Crowcroft, C. Diot, A. Goel, E. de Lara, M. H. Lim, and E. Upton. Hagggle: Seamless networking for mobile applications. In *Proceedings of UbiComp*, 2007.