Crowdsourcing Access Network Spectrum Allocation Using Smartphones

Jinghao Shi[†], Zhangyu Guan[†§], Chunming Qiao[†], Tommaso Melodia[§]
Dimitrios Koutsonikolas[†] and Geoffrey Challen[†]

[†]Department of Computer Science and Engineering, University at Buffalo
[§]Department of Electrical and Computer Engineering, Northeastern University

[†]{jinghaos,zguan2,qiao,dimitrio,challen}@buffalo.edu, [§]melodia@ece.neu.edu

ABSTRACT

The hundreds of millions of deployed smartphones provide an unprecedented opportunity to collect data to monitor, debug, and continuously adapt wireless networks to improve performance. In contrast with previous mobile devices, such as laptops, smartphones are *always on* but *mostly idle*, making them available to perform measurements that help other nearby active devices make better use of available network resources. We present the design of PocketSniffer, a system delivering wireless measurements from smartphones both to network administrators for monitoring and debugging purposes and to algorithms performing real-time network adaptation. By collecting data from smartphones, PocketSniffer supports novel adaptation algorithms designed around common deployment scenarios involving both cooperative and self-interested clients and networks. We present preliminary results from a prototype and discuss challenges to realizing this vision.

Categories and Subject Descriptors

C2.3 [**Network Operations**]: Network management

General Terms

Management, Performance

Keywords

Smartphones; crowdsourcing; monitoring

1. INTRODUCTION

The rapid proliferation of smartphones creates both challenges and new opportunities for wireless networks. On one hand, smartphones compete for the same limited spectrum already crowded with other devices. On the other hand, because smartphones are *always on* but *mostly idle*, they are ideal for observing other nearby active wireless devices—such as laptops, tablets, or other smartphones. When used for continuous network adaptation, offloading measurements from active to inactive clients allows data collection to avoid disturbing active sessions, a capability that has not been adequately exploited by other systems using client-side feedback [14, 6, 5, 7, 9, 8, 10, 11]. When used for network monitoring and debugging, smartphones provide more valuable measurements than planned site surveys, since the data that smartphones provide is continuous and representative of wireless conditions experienced by users while surveys are neither. We refer to these approaches collectively as **c**rowdsourcing **a**ccess **n**etwork **s**pectrum **a**llocation using **s**martphones, or **CANSAS**.

Realizing CANSAS requires novel integration between smartphones and an adaptive network along with new algorithms enabling cooperative spectrum allocation on both short and long timescales. This paper describes a prototype system implementing CANSAS for Wifi networks called PocketSniffer. Implemented as a smartphone app, PocketSniffer is simple to deploy. It uses a pub-sub architecture to collect measurements from passive smartphones and use them to improve network performance. PocketSniffer also captures the large number of measurements made naturally by smartphones as they discover and connect to networks, valuable data that is currently discarded.

To enable short-term adaptation, measurements are triggered by and used as inputs to new algorithms that can alter channel assignments, control AP power levels and rate selection, and alter client associations in order to improve network performance and allocate available spectrum more effectively. PocketSniffer allows different algorithms to be deployed to support a variety of different network structures, including both fully-cooperative settings and cases where multiple networks overlap and compete for the local spectrum resources, scenarios representative of both typical home and enterprise Wifi deployments. To enable long-term adaptation, measurements are provided to network administrators in order to perform network monitoring, maintenance, and capacity planning. Realizing Pocket-

HotNets-XIII, October 27–28, 2014, Los Angeles, CA, USA

http://dx.doi.org/10.1145/2670518.2673866

SNIFFER, however, requires addressing a set of open research challenges: determining how to incentivize client measurements, ensuring that client measurements are accurate, ensuring fairness in the energy consumed by the measurement process, and dealing with differences in smartphone wireless measurement capabilities.

The rest of this paper is structured as follows. Section 2 describes the design of POCKETSNIFFER while Section 3 describes algorithms using POCKETSNIFFER data to perform cooperative network adaptation to support several common deployment scenarios. We continue by presenting results from a prototype POCKETSNIFFER system in Section 4. We discuss some open challenges in Section 5 and related works in Section 6, before concluding in Section 7.

2. POCKETSNIFFER DESIGN

This section describes the design of POCKETSNIFFER, a system enabling CANSAS for Wifi networks by providing the client-side measurements needed to enable the coordination algorithms described in the next section. We begin with an overview of the POCKETSNIFFER system from the perspective of a network user.

2.1 Overview

When Alice and Bob register for their campus WLAN they are required to install the POCKETSNIFFER monitoring app on their smartphones. As they travel around campus, their smartphones collect measurements on the health and performance of the campus network based on queries established by network administrators, reporting these measurements in an energy-neutral way by uploading data only when their smartphones are charging. Network administrators can use this data to identify poorly-served areas of campus, locate APs that are over- or underutilized, and for other network monitoring and debugging purposes.

In addition, as Alice and Bob sit together in a campus cafe using the campus WLAN to surf the web—Alice on her tablet, Bob on his smartphone—suddenly a new source of interference begins to disrupt their network performance. Unfortunately this interfering client cannot be overheard at the AP they are associated with, but the AP can tell that Alice's and Bob's networking environment has degraded. At this point, it triggers the POCKETSNIFFER app on Alice's inactive smartphone to search for a less-congested channel. Based on the results POCKETSNIFFER may move their active devices to a different channel, adjust the AP power level, or suggest they associate with a different campus AP, all without performing measurements on their active clients that could interrupt their web browsing. Without collecting measurements directly from Bob's smartphone, POCKETSNIFFER is still able to exploit the proximity between Alice and Bob and use Alice's nearby inactive smartphone to estimate the impact of changes on Bob's active smartphone to ensure that any spectrum adaptation benefits both devices.

Later, when Alice and Bob return home to neighboring apartments their overlapping home Wifi networks use POCKETSNIFFER measurements as inputs to perform cooperative spectrum allocation. Despite the lack of centralized control, their two networks jointly adapt to allocate spectrum in a way that improves performance for both users.

2.2 Performing Measurements

POCKETSNIFFER collects two types of measurements from clients—scan results and spectrum utilization information—in two different ways—asynchronously and synchronously.

2.2.1 *Measurement types*

Scan results are inexpensive to collect and provide a high-level view of the network including visible APs and their signal strengths. Android smartphones already perform Wifi scans at regular intervals, even while associated with a Wifi network. Recovering this data incurs no energy overhead for clients as long as the measurements are only uploaded while the battery is charging, and our recent analysis of 89 million scan results collected from 139 smartphones over 5 months [13] has demonstrated the value of these measurements for network monitoring and debugging.

However, because smartphones are frequently idle, periodic scans may not reflect either locations where users actually use their devices or locations where their devices use the network. To better connect Wifi scans with interactive usage and network activity, POCKETSNIFFER annotates each scan with two pieces of information: (1) whether the scan was performed during interactive use as estimated by the screen state, and (2) the timestamp since device's last data transfer, because not every interactive session includes Wifi usage and because smartphones perform background data transfers with the screen disabled.

In contrast, spectrum utilization measurements are expensive to collect and may not be possible to collect on all clients, but provide a very detailed view of spectrum usage. The ability of Wifi chipsets to observe link-layer signaling traffic and packets sent by other devices varies from device to device, and because these measurements require disabling the power-save mode used by mobile Wifi chipsets they consume extra energy even if measurement upload is performed in an energy-neutral way. We discuss several research challenges caused by Wifi chipset heterogeneity in Section 5.

2.2.2 *Query types*

Asynchronous measurements are used to perform network monitoring and as a replacement for expensive site surveys in order to do spatial and temporal spectral capacity planning. For asynchronous queries POCKETSNIFFER allows clients to publish measurements to subscriptions set up by network administrators, each of which contains one or more queries describing requested measurements.

POCKETSNIFFER's queries allow administrators to configure both asynchronous and synchronous data collection by restricting the type of data collected, the

devices that participate, APs or active devices to observe, and times during which to perform measurements. Pushing queries to clients allows PocketSniffer to limit the energy overhead of the measurement process, even for asynchronous queries when the data can probably be collected during charging sessions. PocketSniffer avoids disturbing active sessions by waiting to perform asynchronous data collection until a certain amount of time has elapsed since the last interactive session.

To support both network monitoring and debugging over long timescales as well as rapid network adaptation PocketSniffer uses both synchronous and asynchronous queries. Asynchronous queries are configured by network administrators, long-running, and satisfied through delay-tolerant upload. Synchronous queries are initiated on-demand by APs, short-lived, and satisfied immediately by participating clients. When a synchronous query arrives, PocketSniffer clients can decide whether they should collect and return the requested measurements, a decision that is determined by several factors:

- **Usage status.** Because the goal of PocketSniffer is to avoid disturbing active sessions active clients will ignore synchronous queries.
- **Relationships between devices.** PocketSniffer allows users to configure their app to always return data about other devices that they own. Relationships between devices can be manually configured through the app, or a list of other devices associated with a given user can be retrieved from the PocketSniffer service.
- **Proximity to active clients.** PocketSniffer clients must determine whether they can provide measurements approximating the network conditions experience by the active clients included in the query. We return to the challenge of proximity detection in Section 5.
- **Battery level.** Because PocketSniffer runs on energy-constrained devices clients are free to not participate if they are low on energy. The PocketSniffer app allows users to configure separate battery thresholds for queries related and unrelated to their other devices.

If the client decides to participate in the synchronous query, it performs the measurements and sends them directly to the AP. Note that if measurements made to satisfy synchronous queries also match existing asynchronous queries, they will also be uploaded to the PocketSniffer subscription by the AP.

2.3 Network Adaptation

Finally, in the case of synchronous adaptation, synchronous query results must consumed by CANSAS algorithms that are capable of adjust network parameters accordingly. For large enterprise network deployments, these algorithms might run at a central location capable of adjusting AP channels and power levels, similar to what is already done today albeit using only data from APs and not from clients. For home networks, smart PocketSniffer APs might run these algorithms locally, or users could connect them to CANSAS algorithms running as services in the cloud, with this last approach enabling coordination between neighbors with overlapping wireless networks.

3. COORDINATION SCENARIOS

When performing network adaptation PocketSniffer is designed to support multiple different coordination patterns between clients sharing the same WLAN and between overlapping WLANs. In all cases PocketSniffer algorithms utilize crowdsourced measurements from inactive smartphones to attempt to improve performance for active client by controlling AP channel assignments, client associations, and AP power levels and transmission rates. Depending on the scenario, clients may behave selfishly or provide incorrect data in hopes of either avoiding the energy overhead of performing measurements or improving their own network performance at the expense of others.

We are using PocketSniffer to explore several different CANSAS coordination algorithms including maximization and game-theoretic approaches designed to enable cooperation in each of the following common scenarios. Because we are still determining which algorithms work best at achieving the objectives appropriate to each scenario, we focus the discussion below on describing the objective and the associated challenges.

3.1 Single Network, Cooperative Clients

In the first case a single network serves a set of clients that are cooperative in the sense that they are willing to work together to achieve a single common objective. Thus, PocketSniffer can assume that clients are willing to provide truthful measurements. Typical home Wifi networks serving multiple mobile devices fall into this category. Because clients are cooperative and a global objective is shared, this scenario lends itself to the simplest coordination algorithms. One example uses PocketSniffer measurements to select a set of network parameters maximizing the aggregate throughput of all clients; a variant prioritizes interactive sessions by maximizing throughput to interactive clients.

3.2 Single Network, Selfish Clients

In the second case a single network serves a set of clients each of which wishes to maximize its own performance. Typical enterprise Wifi networks fall into this category. This scenario presents two new complications compared with the previous one. First, we must formulate a notion of social utility balancing both performance and fairness. Second, PocketSniffer cannot assume that clients are willing to provide truthful measurements. They may intentionally mislead the system to try to improve their own performance at the expense of other clients, or attempt to avoid the energy overhead of performing measurements. We are addressing these challenges in two ways, both by designing mecha-

nisms that incentivize clients to perform accurate measurements and by utilizing PocketSniffer's control of the network APs to validate client measurements.

3.3 Multiple Networks, Cooperative Clients

In the third case multiple independently-administered networks serve clients that will cooperate within each network. Thus, PocketSniffer can assume that clients provide truthful measurements but that each network acts in a self-interested fashion. Overlapping home Wifi networks each serving multiple mobile devices fall into this category. We are addressing this scenario by formalizing the problem as a noncooperative game between the competing networks where each attempts to selfishly maximize one of the local objectives described in the first scenario.

3.4 Multiple Networks, Selfish Clients

In the final case multiple independently-administered networks serve sets of self-interested clients. Thus, PocketSniffer can neither assume that clients will provide truthful measurements nor that the networks will not act selfishly. Overlapping enterprise Wifi networks fall into this category. Because PocketSniffer must arrange cooperation both between clients within each network and between the overlapping networks, this represents the most challenging scenario. We are attempting to address it by framing it as a two-level noncooperative game where coordination occurs first between the networks and then between clients.

4. PRELIMINARY RESULTS

As mentioned previously, we have performed a detailed analysis of 89 million scan results collected from 139 smartphones over 5 months in order to validate the usefulness of these measurements already being collected by smartphones [13]. Encouraged by these results, we have built a prototype PocketSniffer system including both an Android smartphone app and adaptive AP. Because we have already explored asynchronous analyses that can be performed using scan results, our preliminary results below focus on what additional offline analyses can be done with more detailed channel utilization information (§4.1) and on a basic form of online channel adaptation (§4.2).

To collect detailed channel utilization information, Wifi cards need to be put in monitor mode so that every packet is delivered to upper layer—not just ones addressed to the client. Unfortunately, few smartphone Wifi chipsets currently support this feature out of the box[1], a challenge we return to in Section 5. As a workaround allowing us to explore the potential inclusion of this feature on next-generation smartphones, we equipped several Galaxy Nexus [15] smartphones with external Wifi dongles including chipsets supporting monitor mode. Table 1 describes the Wifi dongle used in our experiments.

[1]In some cases monitor mode support can be achieved by modifying the firmware or device driver [3].

Model	ALFA Network AWUS036H
Chipset	RTL8187L
Connector	1 × 2.4GHz SMA
Antenna	2.5dBi rubber duck
Wifi Support	802.11b/g

Table 1: **Wifi dongle specification.**

4.1 Rogue Access Points

Rogue APs (RAPs) are unauthorized APs deployed in areas designed to be covered by existing enterprise wireless networks. RAPs are of concern to network administrators both for security reasons and due to the unwanted interference they may cause by competing with the official network for spectrum resources. However, RAPs may be set up by users that feel poorly-served by the official wireless network and if properly configured they may not cause harmful interference. And realistically, our typical campus network consisting of several thousand APs also contains hundreds of RAPs, many more than available IT staff can deal with. A system like PocketSniffer provides the ability to determine the impact of RAPs on clients' network performance, allowing administrators to prioritize those that are actually causing problems and ignore those that may be filling coverage holes. To investigate the impact of RAPs, network administrators would establish a PocketSniffer query asking devices to record short (∼1 s) full packet traces after scan results indicated the presence of a previously-identified RAP.

Our department contains several RAPs. To investigate their impact, we deployed 6 PocketSniffer devices: three were deployed statically in public areas, with the remaining three carried by investigators for several hours each day. All devices were put in continuous monitor mode in order to capture all packet traffic, thus providing a superset of the data that would be provided by actual PocketSniffer clients. In total, 38 device-hours worth of data containing 37 M packets were recovered. We first inspect beacon frames to identify APs. Then we exclude campus APs and temporary hotspots with less than 1000 beacon frames. 56 RAPs were detected in this way. For each RAP, we calculate the total traffic volume (both down and up link) by summing up the length of all data frames. Figure 1 shows 15 RAPs with most traffic, as well as the number of devices that ever exchange data frames with them.

Among these 15 RAPs `AP1` and `AP2` are both operated by one of the coauthors. `AP1` is unsecured which likely accounts for its higher traffic volume and larger number of users, as it is available to users that may not be able to authenticate to the campus network. `AP2` is secured and only used by students in a particular room, and the large amount of traffic it is serving might indicate a place where the campus network's coverage could be improved. Other RAPs, such as `Parkhaven-Olewnik` and `26capemay`, also generated large traffic volume yet all of them are with one device, indicating personal RAPs that are secured and only used by the owner.

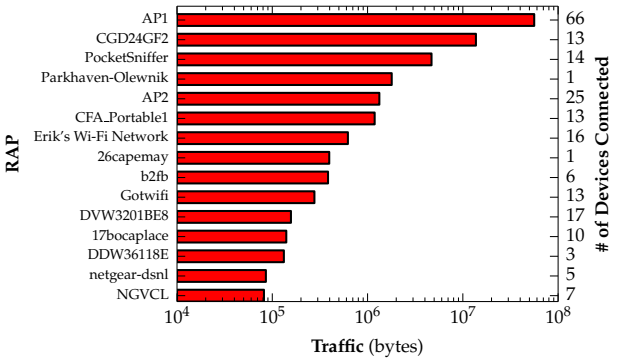

Figure 1: **Measured Bandwidth to Rogue APs.**

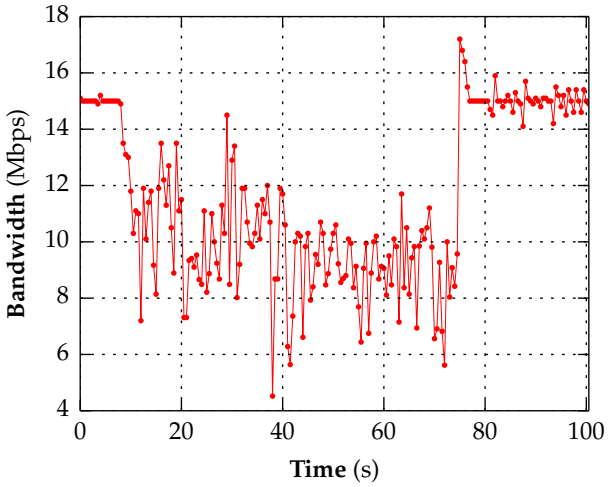

Figure 2: **Bandwidth between A and D_1 after jamming (8.5 s) and channel switching (75 s).**

4.2 Channel Assignment

To demonstrate how PocketSniffer can enable realtime network adaptation to improve the performance of nearby active clients, we ran an experiment using a sniffer device and an adaptive AP. The AP periodically requests channel measurements from all available channels from the sniffer and uses the data to determine the least-congested channel.

The experiment is designed as follows. We first set up a constant `iperf` UDP session between PocketSniffer-AP AP and device D_1 on channel C_1. Then another device D_2 starts jamming the channel using UDP traffic to simulate a new and serious source of interference. However, because the AP is periodically retrieving channel utilization statistics from the sniffer device it can react to the interference by determining which other available channel is the least congested. All of this happens without disturbing D_1, which continues transferring data. All devices in this experiment are using 802.11g at 2.4GHz band. In the particular run shown in Figure 2, from 0–8.5 s A and D_1 establish stable UDP traffic on channel 11.

When D_2 starts jamming the channel at 8.5 s the link bandwidth between A and D_1 decreases and begins to fluctuate. At 75 s, when A and D_1 switch to a less congested channel (1), the bandwidth resumes to the level obtained before the interference began. The latency between the onset of the interference and the channel switch is due to several factors, including the time required for the AP to initiate measurements, for the device to perform the measurements, and for the device to transmit measurements to the AP, all of which can be easily reduced in future prototypes.

5. OPEN CHALLENGES

Realizing the PocketSniffer system at scale requires research addressing several new and existing challenges, each described in more detail below.

5.1 Proximity Detection

To estimate performance for active clients without disturbing their active sessions PocketSniffer attempts to collect data from nearby inactive smartphones. Unfortunately, energy-efficient physical proximity detection on smartphones remains an open research problem [4], may disrupt network performance for the clients involved, and may also not be sufficient to determine how an active client will be affected by potential changes to the network configuration. It may be more appropriate to utilize a signal-based notion of proximity, alone or in combination with physical proximity. We are designing PocketSniffer clients to be able to utilize multiple notions of proximity when deciding whether to participate in synchronous queries in order to determine the most effective approach.

5.2 Measurement Availability and Energy Consumption

As discussed earlier many smartphone Wifi chipsets do not allow detailed measurements, either due to hardware, firmware, or driver limitations. Our current prototype PocketSniffer implementation utilizes special hardware and future versions may take advantage of modified firmware or drivers, but broader adoption will require better support for channel measurements on many smartphone devices. This may be more feasible than it sounds, given that PocketSniffer's detailed channel measurements require much less information than that provided by full monitor mode packet scans, which are typically seen to represent a potential security threat. Ideally, some form of channel utilization estimation could be implemented directly in Wifi hardware devices and activated as needed. Better hardware support would also provide an opportunity to lower the energy overhead of performing detailed channel measurements, another obstacle to deploying PocketSniffer.

However, because detailed channel utilization measurements will always require the radio to be on continuously the energy consumption incurred will always be greater than when the radio is disabled or power-cycled. Thus, we are exploring several other ways to address this energy overhead by limiting the amount of data clients need to provide. One way is to limit these mea-

surements to support synchronous adaptation by active client operated by the same user: i.e., by Bob's smartphone only to help Bob's laptop locate a better channel in the example above. In the second part of the previous example, if Bob has forgotten his smartphone, Alice's device may be unwilling to incur the battery drain necessary to help Bob and only agree to provide scan results rather than detailed channel utilization measurements.

5.3 Incentivizing Data Collection

While measurements collected from PocketSniffer clients should improve network performance, the overhead of performing measurements—particularly detailed channel utilization measurements—requires that PocketSniffer incentivize participation. Some users may not want to install the PocketSniffer app required to collect data. To avoid this problem, it may be appropriate for PocketSniffer networks with long-term registered users, such as enterprise networks, to require users install the PocketSniffer app as part of the network registration process. The authentication required by network providers in this scenario also helps during synchronous queries by naturally identifying sets of related clients. For each user operating a set of devices—attaching a smartphone, laptop, and tablet to the PocketSniffer network—a single instance of the PocketSniffer app running on a smartphone is sufficient to provide measurements allowing all of their other devices to access the network.

For temporary sessions at wireless "hotspots", it may be more appropriate for the network to suggest—but not require—installing the PocketSniffer app and provide improved quality-of-service to clients who do so, essentially trading measurements for bandwidth. Other providers that operate multiple networks serving mobile clients, such as Boingo, may want to require measurements from long-term subscribers or integrate PocketSniffer into their preexisting apps such as the Boingo Wifi Finder [2].

5.4 Validating Measurements

Related to the problem of incentivizing installation is ensuring that clients return accurate measurements. In most cases we believe that users will be unwilling or unable to tamper with the PocketSniffer app in ways that would cause it to return faulty measurements, but there is still the potential for more sophisticated users to break the app to either try to improve their own performance at the expense of other clients or avoid the energy overhead of performing measurements at all by returning bogus data in response to PocketSniffer queries. From the perspective of designing PocketSniffer to resist these behaviors we do not distinguish between malicious and lazy clients. Instead, we focus on designing measurement validation mechanisms that will identify both types of misbehavior.

The easiest way for PocketSniffer to detect incorrect measurements is by manipulating trusted network devices such as APs. As an example, to prevent clients from returning falsified scan results, PocketSniffer APs include a random nonce in each beacon message that clients must report allowing the network to verify that the device actually heard the scan it is reporting. PocketSniffer networks may also manipulate AP power levels to verify that these changes are reflected by client measurements, to prevent clients from inaccurately reporting low AP signal strengths in attempt to receive better service. Similar techniques can be applied to verify detailed measurements, since the messages clients overhear can be validated by network-controlled APs.

A second approach assuming that most clients will cooperate with PocketSniffer is to compare measurements from several different devices. Without a large number of co-located clients to compare against it may be difficult to immediately identify false measurements, but over time noncooperative clients may be identified using reputation mechanisms.

6. RELATED WORK

Using client side measurements to either monitor or reconfigure wireless networks has received a lot of attention recently. Mishra et al. proposed to collect client-side measurements called site-reports, which represent the client's visibility to near-by APs and other devices. This information is then used to determine AP channel assignment [6, 5, 8], or joint channel assignment and terminal association [7]. Sen et al. proposed to use mobile devices to measure and improve the performance of wide-area cellular networks [12]. Finally, the DARPA RadioMap [1] project aims to provide real-time awareness of radio spectrum using idle radios on user devices.

Our approach differs from previous ones in several ways. First, we identify smartphones as an ideal vantage point for both long-term network monitoring and short-term network reconfiguration, due to their *always on* and *mostly idle* nature. Second, we distinguish between active and passive clients only collect measurements from passive clients, exploiting device proximity detection to jointly optimize network performance for both. Third, we consider incentives for clients to provide measurement data and measurement validation mechanisms, which are missing in previous works.

7. CONCLUSIONS

Today millions of smartphones represent a pervasive, highly-available, but also woefully-underutilized wireless monitoring network. We have presented the PocketSniffer system which harnesses the power of always-on but mostly-idle smartphones to improve network performance for nearby active clients by using network measurements to drive both short-term network adaptation and long-term monitoring and debugging.

Acknowledgments

This work is supported in part by NSF through awards CNS-1218717 and CNS-1205656.

8. REFERENCES

[1] Advanced RF Mapping (Raido Map). http://www.darpa.mil/Our_Work/STO/Programs/Advanced_RF_Mapping_(Radio_Map).aspx.

[2] Boingo wifi finder. https://play.google.com/store/apps/details?id=com.boingo.boingowifi.

[3] Monitor mode for broadcom wifi chipsets. https://code.google.com/p/bcmon/.

[4] Bakht, M., Trower, M., and Kravets, R. H. Searchlight: Won't you be my neighbor? In Proceedings of the 18th Annual International Conference on Mobile Computing and Networking (New York, NY, USA, 2012), Mobicom '12, ACM, pp. 185–196.

[5] Dasilva, T., Eustice, K., and Reiher, P. Johnny Appleseed: Wardriving to Reduce Interference in Chaotic Wireless Deployments. In Proc. of the 11th international symposium on Modeling, analysis and simulation of wireless and mobile systems (MSWiM 2008) (October 2008).

[6] Mishra, A., Banerjee, S., and Arbaugh, W. Weighted coloring based channel assignment for wlans. ACM SIGMOBILE Mobile Computing and Communications Review 9, 3 (July 2005), 19–31.

[7] Mishra, A., Brik, V., Banerjee, S., Srinivasan, A., and Arbaugh, W. A client-driven approach for channel management in wireless lans. In Proc. of the 25th IEEE International Conference on Computer Communications (INFOCOM 2006) (April 2006).

[8] Mishra, A., Shrivastava, V., Agarwal, D., Banerjee, S., and Ganguly, S. Distributed channel management in uncoordinated wireless environments. In Proc. of the 12th annual international conference on Mobile computing and networking (MobiCom 2006) (September 2006).

[9] Mishra, A., Shrivastava, V., Banerjee, S., and Arbaugh, W. Partially overlapped channels not considered harmful. In Proc. of the Joint international conference on Measurement and modeling of computer systems (SIGMETRICS 2006/Performance 2006) (June 2006).

[10] Murty, R., Wolman, A., Padhye, J., and Welsh, M. An Architecture for Extensible Wireless LANs. In Proc. of the 7th ACM Workshop on Hot Topics in Networks (HotNets-VII) (October 2008).

[11] Rayanchu, S., Shrivastava, V., Banerjee, S., and Chandra, R. FLUID: improving throughputs in enterprise wireless lans through flexible channelization. In Proc. of the 17th annual international conference on Mobile computing and networking (MobiCom 2011) (September 2011).

[12] Sen, S., Yoon, J., Hare, J., Ormont, J., and Banerjee, S. Can they hear me now?: a case for a client-assisted approach to monitoring wide-area wireless networks. In Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference (2011), ACM, pp. 99–116.

[13] Shi, J., Qiao, C., Koutsonikolas, D., and Challen, G. Using smartphones to monitor and debug enterprise wireless network, submitted, 2014.

[14] Vasan, A., Ramjee, R., and Woo, T. ECHOS – Enhanced Capacity 802.11 Hotspots. In Proc. of the 24th IEEE International Conference on Computer Communications (INFOCOM 2005) (April 2005).

[15] Wikipedia. Galaxy Nexus. http://en.wikipedia.org/wiki/Galaxy_Nexus.