On the Energy Efficiency of Device Discovery in Mobile Opportunistic Networks: A Systematic Approach

Bo Han, Jian Li and Aravind Srinivasan *Fellow, IEEE*

Abstract—In this paper, we propose an energy efficient device discovery protocol, `eDiscovery`, as the first step to bootstrapping opportunistic communications for *smartphones*, the most popular mobile devices. We chose Bluetooth over WiFi as the underlying wireless technology of device discovery, based on our measurement study of their operational power at different states on smartphones. `eDiscovery` adaptively changes the duration and interval of Bluetooth inquiry in dynamic environments, by leveraging history information of discovered peers. We implement a prototype of `eDiscovery` on Nokia N900 smartphones and evaluate its performance in three different environments. To the best of our knowledge, we are the first to conduct extensive performance evaluation of Bluetooth device discovery *in the wild*. Our experimental results demonstrate that compared with a scheme with constant inquiry duration and interval, `eDiscovery` can save around 44% energy at the expense of discovering only about 21% less peers. The results also show that `eDiscovery` performs better than other existing schemes, by discovering more peers and consuming less energy. We also verify the experimental results through extensive simulation studies in the ns-2 simulator.

Index Terms—Device discovery, opportunistic communications, energy efficiency, smartphones, Bluetooth.

1 Introduction

Mobility itself is a significant problem in mobile networking. On the one hand, protocols designed for mobile networks should solve the challenges caused by the mobility of wireless devices. For example, routing protocols, such as DSR (Dynamic Source Routing) [16], are required to handle frequent routing changes and reduce the corresponding communication overhead. On the other hand, mobility can increase the capacity of wireless networks through opportunistic communications [13], where mobile devices moving into wireless range of each other can exchange information *opportunistically* during their periods of contact [7], [21].

Opportunistic communications have been widely explored in delay-tolerant networks [32], mobile social applications [21], [31] and mobile advertising [1], to facilitate message forwarding, media sharing and location-based services. Meanwhile, there are more and more applications leveraging opportunistic communications for various purposes. For example, LoKast[1] is an iPhone application that provides mobile social networking services by discovering and sharing media content among users in proximity. Nintendo 3DS's StreetPass[2] enables players to exchange game data with other users they pass on the street, through the direct device-to-device communication between 3DS systems. Other similar applications include Sony PS Vita's Near and Apple's iGroups.

Device discovery is essentially the *first* step of opportunistic communications. However, there are very few practical protocols proposed for it and most of the existing work mainly utilizes (trace-driven) simulation to evaluate the performance of various device discovery protocols [9], [29]. Moreover, although there are several real-world mobility traces in the CRAWDAD repository[3] which were collected using Bluetooth device discovery, most of them used very simple discovery protocols with *constant* inquiry duration and interval. A recently proposed opportunistic Twitter application [24] also uses a 2-minute inquiry interval for Bluetooth device discovery. It is known that these kinds of discovery protocols are not energy efficient [29] and thus may not be desirable for power-constrained mobile devices, such as smartphones. In this paper, we bridge this gap by developing an *energy-aware* device discovery protocol for smartphone-based opportunistic communications and evaluating its performance in practice.

There are two major challenges in designing, implementing and evaluating energy efficient device discovery protocols for smartphones. First, the selection of underlying communication technology is complicated by the multiple wireless interfaces on smartphones, such

- *B. Han is with AT&T Labs – Research, 1 AT&T Way, Bedminster, NJ, 07921. This work was done when he was a graduate student at the University of Maryland.*
E-mail: bohan@research.att.com
- *J. Li is with the Institute for Interdisciplinary Information Sciences, Tsinghua University, Beijing 100084, China.*
E-mail: lijian83@mail.tsinghua.edu.cn
- *A. Srinivasan is with the Department of Computer Science and the Institute for Advanced Computer Studies, University of Maryland, College Park, MD, 20742.*
E-mail: srin@cs.umd.edu

1. http://www.lokast.com/
2. http://www.nintendo.com/3ds/features/
3. http://crawdad.cs.dartmouth.edu/

as Bluetooth and WiFi (a.k.a., IEEE 802.11).[4] Although Bluetooth is a low-power radio, its device discovery duration is much longer than WiFi (∼10s for Bluetooth vs. ∼1s for WiFi active scanning), which may cause more energy consumption on smartphones. Similarly, WiFi is known to be power-hungry for mobile devices [22], [26]. Thus, it is not clear which of them is more suitable for device discovery on smartphones.

Second, given the dynamic nature of human mobility, we need to adaptively tune device discovery parameters, such as inquiry duration and interval, to reduce smartphone energy consumption. Schemes with constant inquiry intervals have been proven to be optimal for minimizing discovery-missing probability [29]. However, their energy consumption is usually higher than the adaptive ones, which may miss more devices during discovery procedures. Therefore, there is a tradeoff between energy consumption and discovery-missing probability.

We make the following contributions in this paper.

- We present a systematic measurement study of the energy consumption of Bluetooth and WiFi *device discovery* on smartphones, by measuring both the electrical power at various states and the discovery duration (Section 4). Our results show that the energy consumption depends on the number of discovered peers. Based on our measurement results, we chose Bluetooth as the underlying wireless technology, because even its high-power state consumes less energy than the low-power state of WiFi during device discovery. We emphasize that although previous works have studied the power of Bluetooth/WiFi devices [9], [11], [22], they either focus on only Bluetooth [9] or ignore the duration of device discovery [11], [22], without which it is hard to evaluate the *energy consumption* of these devices.
- We design an energy-aware device discovery protocol, named `eDiscovery`, as the first and very important step to bootstrapping smartphone-based opportunistic communications (Section 6). By trading energy consumption for a limited discovery loss, we demonstrate that `eDiscovery` is highly effective in saving energy on smartphones. `eDiscovery` dynamically tunes the discovery duration and interval according to history information of the number of discovered peers. It also introduces randomization into device discovery, in order to explore the search space further.
- Our major contribution is an extensive performance evaluation of `eDiscovery` and other existing device discovery protocols in different realistic environments, through a prototype implementation on Nokia N900 smartphones (Section 7). We conduct experiments in a university campus, a metro station and a shopping center. Our experimental results verify the effectiveness of `eDiscovery` in practice. Compared with the STAR protocol proposed by Wang et al. [29], `eDiscovery` consumes less energy and discovers more peers. `eDiscovery` also performs better than another protocol in the literature.

4. We prefer Bluetooth and WiFi to 3G, as they are local communication technologies with almost no monetary cost.

Compared with its preliminary version [15], this paper makes two new contributions. First, we add the theoretical analysis of device-discovery missing probability (Section 5), which motivates us to design the `eDiscovery` protocol. Second, we port the implementation of `eDiscovery` into the ns-2 simulator enhanced with the UCBT Bluetooth module[5] and offer a detailed evaluation of its parameters (Section 8). We also compare the performance of `eDiscovery` and STAR with more network topologies using ns-2 simulations.

2 Device Discovery in Bluetooth and WiFi

In the following, we discuss device discovery of Bluetooth and WiFi, the two most commonly available local wireless communication technologies on smartphones.

2.1 Bluetooth

The Bluetooth specification (Version 2.1) [3] defines all layers of a typical network protocol stack, from the baseband radio layer to the application layer. Bluetooth operates in the 2.4 GHz ISM (Industrial, Scientific and Medical) frequency band, shared with other devices such as IEEE 802.11 stations, baby monitors and microwave ovens [12]. Therefore, it uses Frequency-Hopping Spread Spectrum (FHSS) to avoid cross-technology interference, by randomly changing its operating frequency bands. Bluetooth has 79 frequency bands (1 MHz width) in the range 2402-2480 MHz and the duration of a Bluetooth time slot is 625 μs. In the following we focus on device discovery and refer interested readers to Smith et al. [27] for further study of the Bluetooth protocol stack.

During device discovery, an inquiring device sends out inquiry messages periodically and waits for responses, and a scanning device listens to wireless channels and sends back responses after receiving inquiries [3]. The inquiring device uses two trains of 16 frequency bands each, selected from 79 bands. The 32 bands of these two trains are selected according to a pseudo-random scheme and a Bluetooth device switches its trains every 2.56 seconds. In every time slot, the inquiring device sends out two inquiry messages on two different frequency bands and waits for response messages on the same frequency bands during the next time slot. After a device receives an inquiry message, it will wait for 625 μs (i.e., the duration of a time slot) before sending out a response message on the same frequency band, which completes the device discovery procedure. For scanning devices, Bluetooth controls their scanning duration and frequency with two parameters, scan window and scan interval.

5. http://www.cs.uc.edu/~cdmc/ucbt/

2.2 Bluetooth Low Energy

Bluetooth Low Energy (LE) [4] operates in the 2400-2483.5 MHz frequency band and divides this band into 40 channels with 2 MHz width, instead of 79 channels with 1 MHz width in the classic Bluetooth. Three out of these 40 channels, with channel indexes 37, 38, and 39, are used for advertising, and the rest are data channels.

Differently from the classic Bluetooth, the LE system leverages these advertising channels for device discovery and connection establishment. Among the five states defined in Bluetooth LE, three of them are related to device discovery: advertising, scanning and initiating states (the rest two are standby and connection states). After a device enters the advertising state (directed by the host machine), it sends out one or more advertising packets that contains its device address on the advertising channels. These advertising packets compose the so-called advertising events. The time between the start of two consecutive advertising events is defined as the sum of a constant *advInterval*, which should be an integer multiple of 625 μs and in the range of 20 ms to 10.24 s, and a pseudo-random value *advDelay* in the range of 0 ms to 10 ms. A device in either scanning or initiating state listens on an advertising channel with the duration of *scanWindow* and the interval *scanInterval* (i.e., the interval between the start of two consecutive scan windows). The *scanWindow* and *scanInterval* parameters should not be greater than 10.24 s.

2.3 WiFi

The key concept of device discovery in WiFi is well understood. WiFi stations in infrastructure and ad-hoc modes periodically (100 ms by default) send out Beacon messages to announce the presence of a network. A Beacon message includes information such as SSID (service set identifier) and capability information. The WiFi interfaces of mobile phones should operate in ad-hoc mode and form an Independent Basic Service Set (IBSS) to support opportunistic communications, since infrastructure-mode interfaces cannot form a network and thus cannot communicate directly. Besides sending out Beacon messages, a WiFi interface also scans wireless channels to discover peers.

There are two types of WiFi scanning, passive and active. In passive scanning, a WiFi interface listens for Beacon messages on each channel, broadcasted by its peers at regular intervals. It periodically switches channels, but does not send any probe request message. During active scanning, a WiFi interface actively searches for its peers, by broadcasting probe request messages on each possible operating channel (channels 1 to 11 in North America). It then waits for probe response messages from its peers, which include information similar to that in Beacon messages.

We prefer active scanning to passive scanning for device discovery of opportunistic networks for two reasons. First, although passive scanning has the advantage of not broadcasting probe request messages, it dwells on each channel longer than active scanning, to collect Beacon messages from peers, and thus may consume more energy. Second, an ad-hoc mode interface may skip the sending of Beacon messages and thus make itself not discoverable by passive scanning, when it tries to scan for other peers with the same SSID (which happens frequently when it is the only station in an IBSS).

3 Related Work

In this section, we briefly review the literature of device discovery in wireless networks.

3.1 Wireless Device Discovery in General

Device discovery has been studied in various wireless networks, such as ad-hoc networks [20], [28], sensor networks [10], [17] and delay-tolerant networks [29].

Neighbor/device discovery is one of the first steps to initialize large wireless networks. McGlynn and Borbash [20] examine the problem of neighbor discovery during the deployment of static ad-hoc networks, where the discovery may last only a few minutes. Vasudevan et al. [28] show that an existing ALOHA-like neighbor discovery algorithm reduces to the classical Coupon Collector's Problem when nodes are not capable of collision detection. They also propose an improved algorithm based on receiver status feedback when nodes have a collision detection mechanism. Cohen and Kapchits [6] investigate a slightly different neighbor discovery problem in asynchronous sensor networks. Instead of study the initial neighbor discovery, they are interested in continuous neighbor discovery after the initial discovery phase. Unlike the above works that are based on abstract communication models, our focus is practical Bluetooth device discovery for smartphone-based opportunistic communications.

Dutta and Culler [10] propose an asynchronous neighbor discovery protocol, called Disco, for mobile sensing applications. U-Connect [17] is another asynchronous neighbor discovery protocol for mobile sensor networks that selects carefully the time slots to perform discovery and that has been proven theoretically better than Disco. Recently, Bakht et al. [2] propose Searchlight, a protocol that combines both deterministic and probabilistic approaches to further reduce the discovery latency for mobile social applications. Disco, U-Connect and Searchlight mainly aim to achieve a tradeoff between discovery latency and energy consumption. For example, U-Connect uses the power-latency product metric for performance evaluation. Differently from them, we are interested in the tradeoff between energy consumption and discovery-missing probability.

The goal of `eDiscovery` is similar in spirit to that of Wang et al. [29] who investigate the tradeoff between the contact probing frequency (which determines energy consumption) and the missing probability of a contact for delay tolerant applications. They design a contact probing algorithm, named STAR (Short Term Arrival Rate), to dynamically change the contact probing frequency. Specifically, STAR estimates the peer arriving rate over a short time, which decays slowly, and calculates the probing frequency based on the estimation. Without specifying the communication technologies, they assume that every probing message is just an impulse and consumes no time. We compare the performance of `eDiscovery` with STAR through extensive real-world experiments and simulation studies.

FlashLinQ [30] is a synchronous wireless network architecture developed by Qualcomm for direct device-to-device communication over licensed spectrum. Although FlashLinQ may be more energy efficient than Bluetooth and WiFi, given its clean-slate design for ad hoc networks, it requires special purpose hardware and also operates in licensed spectrum. Unlike from FlashLinQ, we aim to design and implement device discovery protocols using existing hardware and communication technologies available on commercial smartphones.

3.2 Bluetooth Device Discovery

Bluetooth specifies a detailed device discovery protocol [3]. Salonidis et al. [25] identify the bottlenecks of asymmetric device-discovery delay of Bluetooth and introduce a randomized symmetric discovery protocol to reduce this delay. Based on Bluetooth specification v1.1, Peterson et al. [23] derive rigorous expressions for the inquiry-time probability distribution of two Bluetooth devices that want to discover each other. Chakraborty et al. [5] present an analytical model of the time of Bluetooth device discovery protocol. Liberatore et al. [18] solve the problem of long discovery duration of Bluetooth due to its half-duplex discovery process by the addition of another Bluetooth radio.

Drula et al. [9] study how to select Bluetooth device discovery parameters according to the mobility context and thus reduce the energy consumption of device discovery. They present two algorithms that adjust these parameters based on recent activity level (referred as RAL) and the location of previous contacts, and evaluate their performance through simulations. RAL defines several inquiry modes based on parameters, such as inquiry duration and interval, and switches to a more aggressive mode whenever another device is discovered. In our previous work [14], we compare energy consumption of Bluetooth and WiFi device discovery on Nokia N900 smartphones, using battery life as a metric. We evaluate the Bluetooth device-discovery probability in an office environment using a static phone and a moving phone.

Besides the above works, although there is a large body of literature about Bluetooth device discovery, most of them focus on the improvements of discovery latency between two Bluetooth devices by tuning various parameters or changing the protocol itself, which may not be feasible to implement on smartphones. The focus of this paper is about the performance evaluation of an energy-aware device discovery protocol *in the wild* through a prototype implementation on smartphones.

Bluetooth device discovery has been an important component in opportunistic networks. We refer interested readers to a preliminary version of this paper [15] for a literature review in that area.

4 Energy Consumption of Device Discovery

In this section, we measure the power and energy consumption of Bluetooth and WiFi device discovery on smartphones. Based on the experimental results, we chose Bluetooth as the communication technology. Although previous work has measured energy consumption of WiFi and Bluetooth devices several years ago [9], [22], these results may be invalid given the rapid development of battery and wireless technologies [11]. Friedman et al. [11] have recently studied the power of Bluetooth scanning and WiFi search. However, they overlook the duration of device discovery which determines the energy consumption on smartphones. Furthermore, their measurements are for station mode WiFi interfaces and demonstrate inconsistent results about WiFi device discovery. To the best of our knowledge, there is no systematic study of smartphone energy consumption of Bluetooth and WiFi device discovery.

4.1 Measurement Setup

We measure the electrical power of two states of Bluetooth and WiFi device discovery, idle (i.e., inquiry interval) and active probing, on Nokia N900 smartphones using the Monsoon power monitor[6]. The default OS of Nokia N900, Maemo 5, is an open source Linux distribution (kernel version 2.6.28). Its WiFi chipset is Texas Instruments WL1251 using the wl12xx device driver[7]. Its Bluetooth chipset is Broadcom BCM2048. We use BlueZ[8], the default Bluetooth protocol stack of most Linux distributions, to run Bluetooth device discovery experiments. During the measurements, we redirect standard output to `\dev\null` and turn the screen off to minimize their impact on the measurement results. We report the average result and the 95% confidence intervals for each configuration over 10 runs in this section. Note that all results of power measurements in this paper include the baseline power of the smartphone under test.

4.2 Bluetooth

We present a 60-second snapshot of the power of Bluetooth device discovery in Figure 1. We perform the experiments by running `hcitool`, a tool that can send commands, such as `inq` (inquiry), to Bluetooth devices. We use the `flush` option to clear the cache of previously

6. http://www.msoon.com/LabEquipment/PowerMonitor/
7. http://linuxwireless.org/en/users/Drivers/wl12xx
8. http://www.bluez.org/

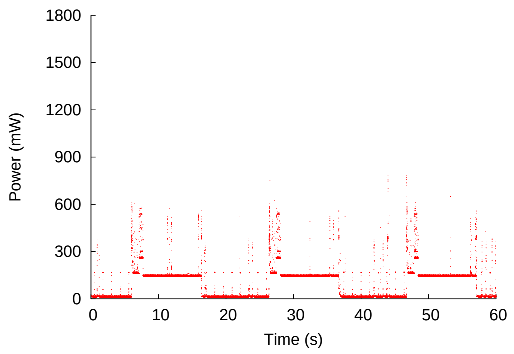


Fig. 1: A 60-second snapshot of the temporal power of periodic Bluetooth device discovery with 10-second interval. The smartphone under test is a Nokia N900 smartphone.

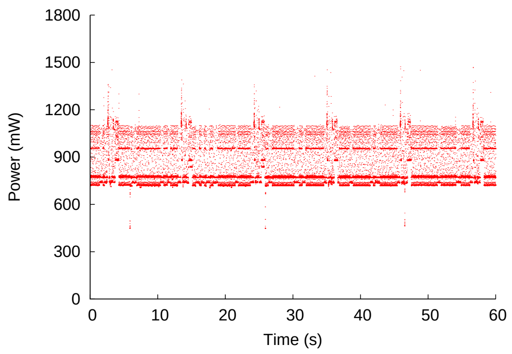


Fig. 2: A 60-second snapshot of the temporal power of periodic WiFi device discovery with 10-second interval. The smartphone under test is a Nokia N900 smartphone.

# of Devices	Average	95% Confidence Interval
0	162.03	(160.72, 163.34)
1	227.06	(219.42, 234.70)
2	247.72	(242.39, 253.05)
3	248.91	(243.02, 254.80)
4	248.59	(246.63, 250.55)
5	256.02	(252.96, 259.08)
6	253.05	(249.63, 256.47)

TABLE 1: The electrical power (in mW) of Bluetooth device discovery with different numbers of peers.

discovered devices before each inquiry. During the measurements, the phone queries neighboring Bluetooth devices periodically with a 10-second interval. When there is no neighboring device, the average power of Bluetooth inquiry over 10 runs is ~162.03 mW (standard deviation: 2.12 mW). During the idle states, the Bluetooth radio is in discoverable mode with average power ~16.54 mW (standard deviation: 1.11 mW).

The average power of Bluetooth device discovery is affected by the number of neighboring devices. We repeat the experiments with the number of neighboring Bluetooth devices increasing from 0 to 6 and summarize the results in Table 1. As we can see from this table, when there is one neighboring device, the average power increases to around 227.06 mW, due to the reception of response messages of Bluetooth inquiry. When there are more than one neighboring devices, the average power increases to about 250 mW. Defined in the standard [3], the duration of Bluetooth device discovery should be a multiple of 1.28 seconds and the recommended default value is 10.24 seconds, which we used in the measurements. Figure 1 shows clearly the configured Bluetooth device discovery duration and interval.

4.3 WiFi

We present another 60-second snapshot of the power of WiFi device discovery in Figure 2. We perform the

Environment	# of peers	duration (s)
Office	43.52 (42.48, 44.56)	1.07 (1.04, 1.10)
Home	14.02 (13.75, 14.29)	0.87 (0.86, 0.88)
Park	0.01 (n/a)	0.52 (0.51, 0.53)

TABLE 2: The average number of discovered peers and duration of WiFi device discovery in three environments. The numbers in the parentheses are the 95% confidence intervals.

experiments by running `iwlist`, a tool that shows the list of access points and ad-hoc cells in range through active scanning. During the measurements, the phone scans neighboring devices periodically also with a 10-second interval, which can be clearly identified in Figure 2. The average power of WiFi active scanning over 10 runs is ~836.65 mW (standard deviation: 8.98 mW). Even during scanning intervals, the average power is ~791.02 mW (standard deviation: 5.23 mW), because the WiFi radio is in ad-hoc mode and sends out Beacon messages with 100 ms intervals.

Differently from Bluetooth, the duration of WiFi active scanning is not constant and may depend on the number of operation channels and the amount of neighboring peers. We measure the duration of WiFi device discovery in three different environments: a campus office building, an apartment, and a national park, and summarize the results in Table 2. In each environment, we repeat the experiments 100 times and report the average values and the 95% confidence intervals. As we can see from this table, when the number of discovered peers increases, the duration of WiFi device discovery grows from ~0.52 seconds to ~1.07 seconds, which is much shorter than the duration of Bluetooth inquiry.

4.4 Energy Consumption

We summarize the average power of Bluetooth (with 6 neighboring devices) and WiFi device discovery in Table 3. Suppose the power is P_{idle} for the idle state and P_{probe} for the inquiry/scan state of Bluetooth/WiFi devices, the duration of Bluetooth inquiry/WiFi scan is

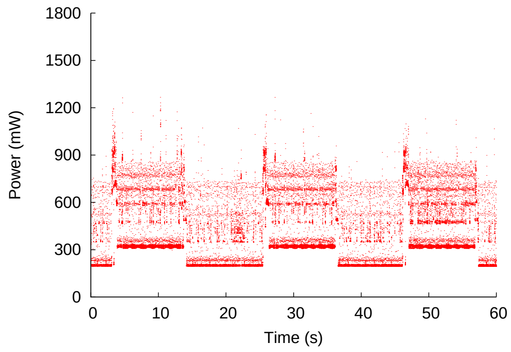


Fig. 3: A 60-second snapshot of the temporal power of periodic Bluetooth device discovery with 10-second interval. The smartphone under test is a HTC Hero smartphone (Android 1.5).

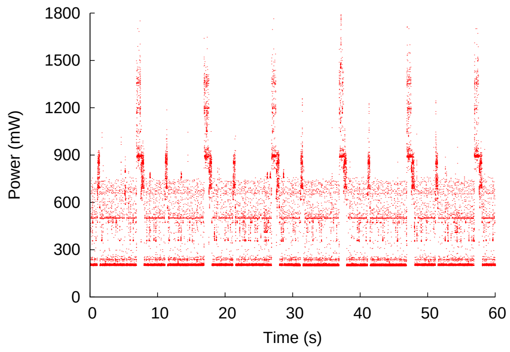


Fig. 4: A 60-second snapshot of the temporal power of periodic WiFi device discovery with 10-second interval. The smartphone under test is a HTC Hero smartphone (Android 1.5).

	P_{idle}	P_{probe}
Bluetooth	16.54 (15.85, 17.23)	253.05 (249.63, 256.47)
WiFi	791.02 (787.78, 794.26)	836.65 (831.08, 842.22)

TABLE 3: The average power of Bluetooth and WiFi device discovery in mW. The numbers in the parentheses are the 95% confidence intervals.

T_{probe} and the inquiry/scan interval is T_{idle}. Then the estimated energy consumption is

$$E = T_{idle} \cdot P_{idle} + T_{probe} \cdot P_{probe}$$

Given the high power of WiFi device discovery in both active probing and idle states, we prefer Bluetooth to WiFi for device discovery of smartphone-based opportunistic communications. We note that no matter how long the duration of Bluetooth inquiry is, the overall energy consumption of Bluetooth device discovery should always be lower than that of WiFi, because the power of Bluetooth inquiry is even lower than that of the WiFi idle state (253.05 vs. 791.02 mW). To perform device discovery, the major problem of WiFi ad-hoc mode is that the radio needs to send out Beacon messages periodically and power saving mechanisms for WiFi ad-hoc mode are not available on most mobile phones [26].

Although the communication range of WiFi is longer than Bluetooth and may discover more peers, making its device discovery energy efficient requires substantial modifications of the WiFi protocol, which may not be feasible on most smartphones. In this paper, we aim to design a device discovery protocol without changing the underlying communication protocol and thus make its deployment easy. This is another reason why we chose Bluetooth over WiFi. However, we emphasize that if energy consumption is not a major concern and the design goal is to discover as many peers as possible or to transfer a large amount of data efficiently, we should use WiFi as the underlying communication protocol (which is out of the scope of this paper), because it has a larger coverage area.

4.5 Android Smartphones

We also measured the power of Bluetooth and WiFi device discovery using a HTC Hero smartphone with Android 1.5. We plot the results in Figure 3 for Bluetooth and Figure 4 for WiFi. On this smartphone, the average power is 432.84 mW (standard deviation: 7.86 mW) for Bluetooth inquiry and 900.25 mW (standard deviation: 21.54 mW) for WiFi scan. There are two differences of the experiments on the Nokia N900 and HTC Hero smartphones. First, the experiments on HTC Hero were performed with the screen on due to the operational requirements and thus the baseline power of HTC Hero is higher than that of Nokia N900. Second, the WiFi interface on HTC Hero does not support ad-hoc mode and we cannot measure the average power P_{idle} on it. However, these results still clearly show the significant power difference (467.41 mW) of Bluetooth inquiry and WiFi scan.

5 Device Discovery Missing Probability

In this section, we analyze the missing probability of a device discovery protocol with constant Bluetooth inquiry window and interval (referred as `Constant` in the following).

First, we introduce some notations. For a given device i, we assume that the contact durations $t_D(i)$ are independent and identically distributed random variables with common PDF (probability density function) $p(x) = \frac{\mathrm{d}}{\mathrm{d}x}\Pr[t_D \leq x]$. We assume the inter-contact time (the time between subsequent contacts) $t_C(i)$ are stationary random variables.

If a scanning device is in the discovery/contact range of an inquiring device for L seconds, it can be discovered with probability $R(L)$. We can easily derive $R(L)$ from the analysis of the probability distribution of the inquiry time for Bluetooth devices by Peterson et al. [23]. For Bluetooth device discovery, $R(L)$ is a monotonically increasing function of L. Let $\overline{R}(L) = 1 - R(L)$. Assume that for different devices, or for the same device in

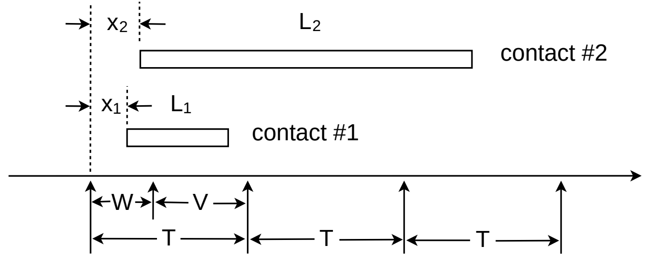


Fig. 5: Two contact cases for analyzing the device discovery missing probability.

different contact intervals, the discovery probabilities are independent of each other. For any real number L, we let $L_+ = \max\{L, 0\}$. We use P_{miss} to denote the probability that a contact is missed (i.e., the scanning device is not discovered by the inquiring device).

Now, we analyze P_{miss} for the `Constant` protocol that repeatedly performs Bluetooth inquiry for W seconds and sleeps for next V seconds. Let $T = W + V$. Suppose the scanning device arrives at time $nT + x$ for some very large integer n and $x \in [0, T]$ and the contact duration t_D is a constant real L. We use $\mathrm{P}_{miss}(x, L)$ to denote the missing probability under the above condition. Let $k = \frac{L+x-T}{T}$. Based on the value of k, we distinguish the following two cases, as shown in Figure 5.

1) If $k < 0$ (i.e., $x + L < T$, contact #1 in Figure 5) which means the contact ends within the period T, it is easy to see that

$$\begin{aligned}\mathrm{P}_{miss}(x, L) &= 1 - R(\min\{L, (W - x)_+\}) \\ &= \overline{R}(\min\{L, (W - x)_+\})\end{aligned}$$

2) If $k > 0$ (e.g., contact #2 in Figure 5), there may be more than one inquiry cycles (inquiry window plus inquiry interval). Let us consider them one by one. In $(nT, (n + 1)T)$, the scanning device is not discovered by the inquiring device with probability $\overline{R}((W-x)_+)$. In each of the next $\lfloor k \rfloor$ cycles, the two devices are in the contact range for W seconds and the missing probability is $\overline{R}(W)$. In the last possible cycle, the contact interval is of length $\min(y, W)$, where $y = L + x - T - \lfloor k \rfloor T$. Therefore,

$$\mathrm{P}_{miss}(x, L) = \overline{R}((W - x)_+)\overline{R}(W)^{\lfloor k \rfloor}\overline{R}(\min(y, W))$$

If the inter-contact time follows a nonlattice distribution, by Blackwell's Theorem in renewal theory [29], we have that, when $n \to \infty$,

$$\begin{aligned}\mathrm{P}_{miss} &= \int_0^T \int_0^\infty \mathrm{P}_{miss}(x, L)p(L)\mathrm{d}L\mathrm{d}x \\ &= \int_0^T \int_0^{T-x} \overline{R}(\min\{L, (W - x)_+\})p(L)\mathrm{d}L\mathrm{d}x \\ &+ \int_0^T \int_{T-x}^\infty \overline{R}((W - x)_+)\overline{R}(W)^{\lfloor k \rfloor}\overline{R}(\min(y, W))p(L)\mathrm{d}L\mathrm{d}x\end{aligned}$$

Similar with the analysis by Wang et al. [29], the missing probability is independent of the inter-contact time distribution. The major difference is that we also consider the inquiry window duration in our analysis, besides the contact duration and inquiry interval. A key observation here is that the missing probability is also independent of device density and is solely determined by the values of W and V. Therefore, if we dynamically tune W and V to decrease (increase) the missing probability when the device density is high (low), we may be able to increase the overall device-discovery probability. For example, if we fix the missing probability at 0.2, for two adjacent areas with density 100 and 10 we can discover 88 devices. However, if we decrease the missing probability to 0.1 for the dense area and increase it to 0.3 for the sparse area, we can discover 97 devices. This observation motivates us to design the `eDiscovery` protocol in the next section.

6 eDiscovery Design

In this section, we present `eDiscovery`, an energy-aware device discovery protocol that adaptively changes the duration and probing interval of Bluetooth inquiry.

The major design principle of `eDiscovery` is to reduce smartphone energy consumption of device discovery, while not missing too many peers. To achieve this goal, we dynamically change the duration and interval of Bluetooth device discovery, based on the number of discovered peers. In theory, if a mobile device knows the density of its peers at any given time, it may be able to select the optimal values for these two Bluetooth device discovery parameters. However, in practice it is hard to estimate this density, especially in dynamic environments, such as shopping malls and train stations. Therefore, we present a heuristic adaptive inquiry approach of `eDiscovery` in Algorithm 1.

There are two approaches to control the duration of Bluetooth device discovery: (1) specifying the length of the inquiry window explicitly or (2) specifying the number of received responses before device discovery stops. Accordingly, there are two parameters of `hci_inquiry`, the device discovery function of BlueZ, *inquiry_window* and *num_responses*. This function stops inquiry after 1.28×*inquiry_window* seconds or it has received *num_responses* inquiry responses.

We focus on the control of the inquiry window in this paper, as it is hard to predict the number of neighboring peers in practice. Moreover, a peer can respond to an inquiry more than once. Suppose there are 3 neighboring peers, A, B, and C, and we set *num_responses* to be 3. If all the first 3 responses are sent by peer A, device discovery will stop after receiving them and thus discover only peer A. We note that `eDiscovery` sits between mobile applications and Bluetooth device discovery and thus the contention/collision of Bluetooth device discovery messages are resolved at the MAC layer of Bluetooth protocol stack.

Algorithm 1 Adaptive Inquiry Algorithm of `eDiscovery`

```
1: inquiry_window = base_W, inquiry_interval = base_I;
2: while (TRUE) do
3:     peers = hci_inquiry(inquiry_window, MAX_RSP);
4:     if (peers > N) then
5:         inquiry_window = base_W;
6:     else
7:         inquiry_window = small_W + r;
8:     end if
9:     if (peers == 0 and last_peers == 0) then
10:        inquiry_interval += inc_NP + r;
11:    else if (peers <> 0 and last_peers == 0) then
12:        inquiry_interval = base_I + r;
13:    else if (peers > last_peers and inquiry_interval − I is in the valid range) then
14:        inquiry_interval −= I;
15:    else if (peers < last_peers and inquiry_interval + I is in the valid range) then
16:        inquiry_interval += I;
17:    end if
18:    last_peers = peers;
19:    sleep(inquiry_interval);
20: end while
```

The two key parameters in Algorithm 1 of `eDiscovery` are the threshold of the number of discovered peers N and the increment/decrement of inquiry interval I. The outputs of Algorithm 1 are *inquiry_window* and *inquiry_interval*, which control the duration and interval of Bluetooth inquiry. The main body of this algorithm is a while loop that performs Bluetooth inquiry $1.28*$*inquiry_window* seconds and then sleeps *inquiry_interval* seconds.

After each Bluetooth inquiry, we adapt the values of *inquiry_window* based on the number of discovered peers. If this number is larger than N, we keep the default initial value *base_W*, aiming to discover more peers. If it is smaller or equal to N, we set the next *inquiry_window* to be $small\_W + r$, where r is defined as

$$r = \begin{cases} 1 & \text{with probability } (1-p)/2 \\ 0 & \text{with probability } p \\ -1 & \text{with probability } (1-p)/2 \end{cases}$$

By changing *inquiry_window* in this way, we can reduce the duration of Bluetooth inquiry and thus save energy on smartphones when the number of neighboring peers is small.

We adapt the value of *inquiry_interval* to the number of discovered peers in a similar way. When a smartphone discovers no peers for two consecutive inquiries, we increase *inquiry_interval* by $inc\_NP + r$ and reset it to $base\_I + r$ after the smartphone discovers new peers. Moreover, if the current number of discovered peers is larger than the previous one, we decrease *inquiry_interval* by I, and vice versa. An implication of this algorithm is that *inquiry_interval* will not change if the number of discovered peers does not vary. We allow *inquiry_interval* to vary between 10 – 200 seconds. The random variable r is refreshed for every inquiry. We use it for improving the robustness of `eDiscovery` for dynamic environments. Furthermore, it can avoid synchronization of Bluetooth inquiry which may make Bluetooth devices not be able to discover each other [14], [23].

Parameter	Default	Description
N	5	Threshold of discovered peers
I	1	Increment of inquiry interval
base_W	8	Base of inquiry window
base_I	10	Base of inquiry interval
MAX_RSP	255	Maximum number of scanned peers
small_W	5	Smaller inquiry window
inc_NP	10	Increment of interval when no peers
r	0	Random variable for robustness
p	0.8	Probability of $r = 0$

TABLE 4: The parameters in Algorithm 1 and their default values.

The intuition behind these adaptations is that we can reduce inquiry duration and increase inquiry interval when the number of neighboring peers is small, because doing this will not miss too many peers. By changing the values of N and I, we can achieve different tradeoff between the number of discovered peers and smartphone energy consumption. Smaller N and I lead to more aggressive Bluetooth inquiry, which may discover more peers but also consume more energy on smartphones.

We list the default values of the parameters of Algorithm 1 in Table 4. These values are not set arbitrarily. We set the initial *inquiry_window* to be 8 (i.e., $8*1.28 = 10.24$ seconds) because it is the default standard value of Bluetooth inquiry. We set *MAX_RSP* to be 255 (the suggested value in BlueZ protocol stack). We set *small_W* to be 5. Thus when the number of discovered peers is smaller than N, the smallest inquiry window $5 + r$ would be 4, as this is the minimum inquiry window to perform a complete scan of all possible frequency bands. Moreover, Peterson et al. [23] demonstrate that by setting the *inquiry_window* to be 4, a Bluetooth device can locate 99% of neighboring devices within its transmission range in a *static* environment. When deciding the probability p in r, essentially, we want to set the parameters to be their default values under certain conditions with a high probability (by default $p = 0.8$) and slightly change their values by 1 with a low probability.

We evaluate the performance of `eDiscovery` with different values of N and I in Section 7. We also evaluate how other parameters, such as *base_W*, *base_I* and the choice of random variable r, affect the performance of `eDiscovery` in Section 8.1. Moreover, we evaluate various solutions of changing inquiry interval based on the difference of discovered peers, by replacing line 14

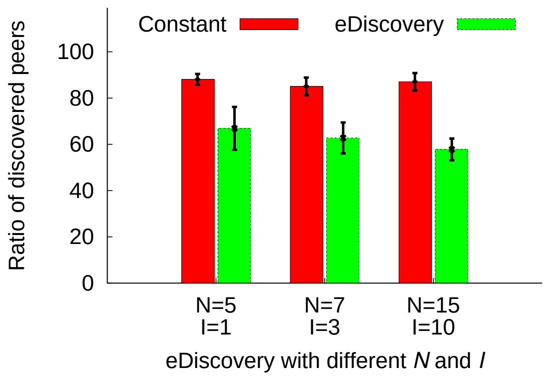


Fig. 6: The ratio of the number of discovered peers for `Constant` and `eDiscovery` to the ground truth, with different N and I. The error bars are the 95% confidence intervals.

in Algorithm 1 with

$$inquiry\_interval- = ceil(\alpha \times abs(peers - last\_peers));$$

and line 16 with

$$inquiry\_interval+ = ceil(\alpha \times abs(peers - last\_peers));$$

where α is a parameter to further tune the value of $inquiry\_interval$. Finally, we note that there may be a diminishing return for some applications for which the change from discovering nothing to one peer is more important than that from discovering, for example, 10 to 11 peers. In this case, we need to invest more energy when the number of peers is small, the opposite behavior of our `eDiscovery` protocol.

7 Experimental Evaluation

In this section, we evaluate the performance of our proposed `eDiscovery` protocol through a prototype implementation on Nokia N900 smartphones and compare it with other schemes. Although previous work has evaluated device discovery protocols using simulations [9], [29], a recent study demonstrates that even contact-based simulations using real-world mobility traces may not be able to accurately evaluate the performance of opportunistic networks [24]. Moreover, it is also not clear how Bluetooth device discovery performs in the wild, under cross-technology interference [12].

We implement `eDiscovery` in C language using the BlueZ protocol stack and compare its performance with three other approaches: the `Constant` protocol in Section 5, the STAR algorithm by Wang et al. [29] and the RAL scheme by Drula et al. [9]. The two metrics we are interested in are the ratio of discovered peers, compared to the ground truth, and the estimated energy consumption. To get the ground truth, we perform Bluetooth inquiry with the default 10.24-second duration continuously. Based on the ground truth, we can know how may peers `Constant` can discover by aggregating the inquiries in the ground truth with only odd/even indices. We did all experiments three times and report the average results with the 95% confidence intervals.

Param. N, I	`Constant`	`eDiscovery`	Percent
5, 1	220.83 (212.67, 228.99)	123.93 (115.92, 131.94)	56.12%
7, 3	209.02 (202.63, 215.41)	113.84 (96.24, 131.44)	54.46%
15, 10	210.80 (200.86, 220.74)	105.60 (103.70, 107.50)	50.09%

TABLE 5: The estimated energy consumption (in Joules) of `eDiscovery` with different N and I and the comparison with `Constant`. The numbers in the parentheses are the 95% confidence intervals.

7.1 Impact of N and I

We first evaluate the performance of `eDiscovery` for different combinations of N and I, using `Constant` as the baseline. During a single experiment, we run the continuous Bluetooth inquiry on one phone and `eDiscovery` on another simultaneously. We conducted the experiments in and around the Stamp Student Union of the University of Maryland. We walked along a pre-defined route for around 30 minutes during the experiments. Most of the Bluetooth devices discovered by us should be on mobile phones, although they can also be on other mobile devices such as tablets and laptops.

We plot the percentage of discovered Bluetooth devices of `eDiscovery` and `Constant` in Figure 6. We also summarize their estimated energy consumption in Table 5. The experimental results show that increasing N and I can save smartphone energy consumption at the expense of a higher missing probability. When $N = 5$ and $I = 1$, `eDiscovery` consumes only 56% energy of `Constant`, and discovers 21% less peers than it. These results also partially verify experimentally the theoretical analysis by Wang et al. [29] that the probing scheme with constant inquiry intervals achieves the minimum discovery-missing probability among all probing methods with the same average inquiry interval. The ratio of discovered peers between `Constant` and the ground truth is higher than 80% for all experiments.

7.2 Dynamic Environment

We then compare the performance of `eDiscovery` ($N = 5$ and $I = 1$) with `Constant` and STAR [29] in three different environments: the Student Union of the University of Maryland, the Union Station of Washington D.C. and the Mall at Short Hills in New Jersey. We also chose a pre-defined route in the other two locations, including both indoor and outdoor environments, and the duration of experiments was about 30 minutes too. Generally, there are much more peers in the indoor environment than the outdoor environment in these three locations. We limit the inquiry interval of STAR to be 10 – 200 seconds, the same as `eDiscovery`.

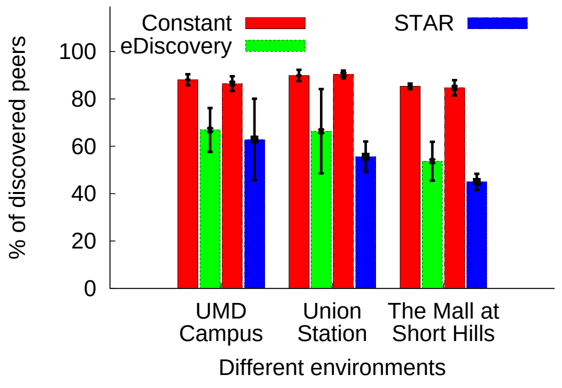


Fig. 7: The comparison of the percentage of discovered peers for different schemes. The error bars are the 95% confidence intervals.

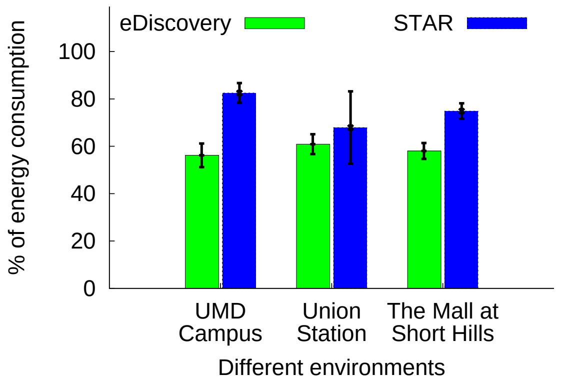


Fig. 8: The comparison of the percentage of energy consumption for different schemes, using `Constant` as baseline. The error bars are the 95% confidence intervals.

We plot in Figure 7 the percentage of discovered peers of `eDiscovery`, `Constant` and STAR, compared with the ground truth. In each group of experiments, we run `Constant` along with either `eDiscovery` or STAR. Thus there are two bars for `Constant` for each location in Figure 7. We note that the first two bars in this figure for `Constant` and `eDiscovery` present the results from the same experiment as the first two bars in Figure 6. We also plot in Figure 8 the energy consumption of `eDiscovery` and STAR, compared with `Constant`. As we can see from these figures, `eDiscovery` performs better than STAR in all three locations. In particular, `eDiscovery` discovers more peers than STAR but consumes much less energy on smartphones.

`eDiscovery` outperforms STAR for two reasons. First, `eDiscovery` takes into account not only the inquiry interval, but also the duration of inquiry, to further reduce smartphone energy consumption. As shown in Section 4, the active probing state consumes much more energy than the idle state of Bluetooth inquiry. Second, it adapts to environmental changes (i.e., the number of neighboring peers) much more quickly than STAR, which is important in dynamic environments.

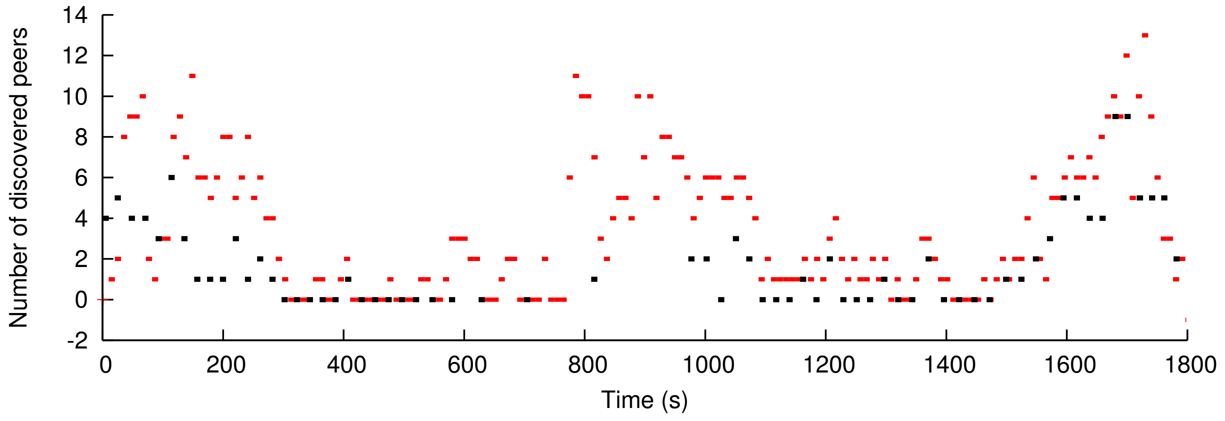
7.3 An In-Depth Look at the Traces

To verify the above, we took an in-depth look at the traces collected for the experiments we did in the Mall at Short Hills. We plot the start time, duration, and the number of discovered peers of a single experiment for STAR in Figure 9a and for `eDiscovery` in Figure 9b. For a Bluetooth device discovery starting at s and ending at t that discovers p peers, we plot a horizontal bar from (s, p) to (t, p). We note that in these two figures, a Bluetooth device may be counted several times if it appeared in multiple device discoveries. In each figure, we use the red color to plot the ground truth and the black color for either STAR or `eDiscovery`. During both experiments, we discovered more than 100 peers in the ground truth. The percentage of discovered peers is around 60% for `eDiscovery` and 40% for STAR.

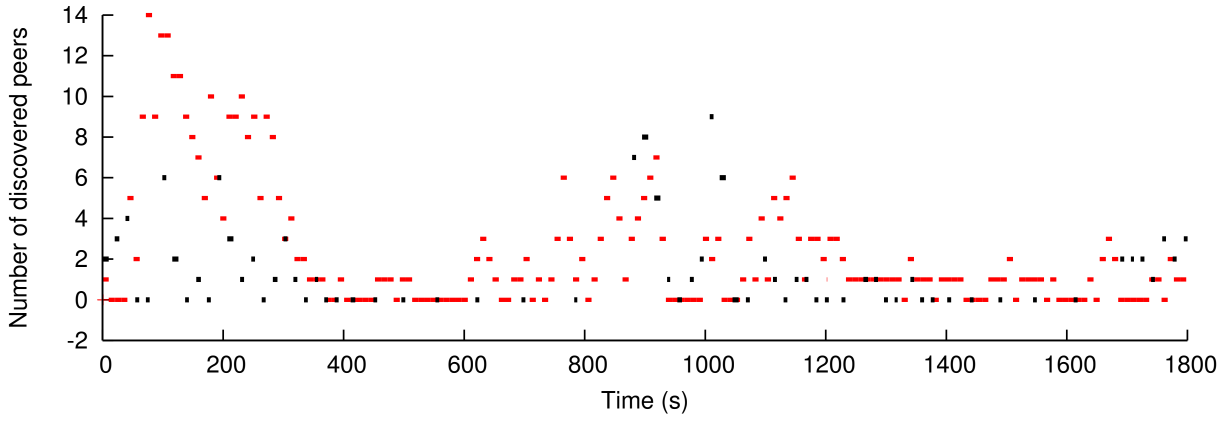
There are two main observations from Figure 9a and Figure 9b. First, on average the duration of Bluetooth device discovery in `eDiscovery` is shorter than STAR (6.79 seconds vs. 10.25 seconds), which is demonstrated by the narrower black bars in Figure 9b. Second, `eDiscovery` increases the inquiry intervals much faster than STAR when there are few peers and decreases the intervals much quicker when there are more peers. For example, from 300 seconds to 600 seconds of both experiments, there were at most 3 peers found by each inquiry. During this quiet period, `eDiscovery` performed Bluetooth inquiry only 10 times, 3 times less than STAR. Moreover, during the period from 800 seconds to 1,000 seconds when there were more peers, `eDiscovery` performed Bluetooth inquiry 7 times, 4 times more than STAR. On the one hand, the shorter discovery duration and less frequent Bluetooth device discovery during the quiet period translate into less energy consumption of `eDiscovery` than STAR. On the other hand, the more frequent device discovery when there are many peers is one of the reasons that the discovery-missing probability of `eDiscovery` is lower than STAR.

7.4 Comparison with Another Protocol

We also evaluate the performance of the RAL protocol in Drula et al. [9], by measuring the number of discovered peers. RAL can discover only less than 30% of peers found in the ground truth for the experiment we did in the Mall at Short Hills. The possible reason may be that even for the most aggressive discovery mode in RAL, the duration of Bluetooth device discovery is less than 1 second, which is too short to complete a scan of all possible Bluetooth frequency bands. Differently from RAL, the shortest duration of Bluetooth device discovery in `eDiscovery` is 5.12 seconds, which is more suitable when the number of neighboring peers changes dynamically. Note that although it is possible to



(a) The start time, duration and the number of discovered peers of Bluetooth device discovery for a STAR experiment. The red horizontal bars are for the ground truth and the black ones are for STAR.



(b) The start time, duration and the number of discovered peers of Bluetooth device discovery for an `eDiscovery` experiment. The red horizontal bars are for the ground truth and the black ones are for `eDiscovery`.

Fig. 9: Detailed traces of `eDiscovery` and STAR experiments.

tune the parameters of RAL and STAR to improve their performance, that is beyond the scope of this paper.

7.5 Summary

To summarize, our performance evaluation shows that if energy consumption is not a major concern and the key objective is to discover more neighboring devices, `Constant` may be a good choice. It can discover more than 80% peers but consumes only half energy of continuous device discovery. However, when the major goal is to save energy on smartphones and the missing of some peers is acceptable, we should use `eDiscovery` to dynamically tune the parameters of Bluetooth device discovery. In other words, the selection between `Constant` and `eDiscovery` depends on the requirements of applications that actually use them for device discovery.

8 Simulation Studies in ns-2

To perform a much more extensive evaluation of `eDiscovery`, we port its implementation into the ns-2 simulator enhanced with the UCBT Bluetooth module (version 0.9.9.2a). This UCBT module is for Bluetooth version 1.2 and there is no significant difference between the device discovery specifications of Bluetooth versions 1.2 and 2.1.

8.1 `eDiscovery` Parameters

Using UCBT based simulation studies, we evaluate how the parameters of Algorithm 1, including different values of *base_W*, *base_I*, *small_W* and *inc_NP*, and the choice of random variable r, affect the performance of `eDiscovery`. Recall that we set the default values of these parameters as listed in Table 4. The simulation setup is as follows. The simulation area is a 1800x20 rectangle. The inquiring Bluetooth devices moves from (0, 10) to (1,800, 10) with a constant speed 1 m/s and thus the simulation duration is 30 minutes. We distribute 100 scanning Bluetooth devices in the simulation area uniformly and randomly.

We summarize the simulation results for 1,000 randomly generated network topologies in Table 6. The second row of this table shows the simulation results with

Param. / Value		% of Discovered Peers	Duration of Inquiry
Default		74.39 (73.96, 74.82)	581.33 (579.30, 583.36)
base_W	4	47.48 (46.92, 48.04)	323.71 (321.42, 326.00)
	12	74.72 (74.29, 75.15)	585.69 (583.64, 587.74)
base_I	6	76.62 (76.18, 77.06)	693.54 (690.79, 696.29)
	14	68.42 (67.99, 68.85)	490.58 (488.96, 492.20)
small_W	3	47.47 (46.91, 48.03)	328.24 (325.91, 330.57)
	7	79.40 (79.03, 79.77)	748.66 (746.76, 750.56)
inc_NP	5	80.47 (80.16, 80.78)	628.25 (627.05, 629.45)
	15	69.40 (68.88, 69.92)	541.01 (538.15, 543.87)
p	0.9	74.91 (74.50, 75.32)	585.56 (583.62, 587.50)
	0.7	74.57 (74.13, 75.01)	580.35 (578.25, 582.45)
α	0.5	74.51 (74.10, 74.92)	581.30 (579.31, 583.29)
	1.0	74.73 (74.32, 75.14)	578.49 (576.53, 580.45)
	1.5	73.22 (72.80, 73.64)	565.07 (562.94, 567.20)

TABLE 6: Performance evaluation of `eDiscovery` using different parameters. The numbers in the parentheses are the 95% confidence intervals.

the default values of these parameters. For the rest of the table, when the value of a parameter is changed, we use the default values for all other parameters. The major observation from Table 6 is that by increasing inquiry duration or decreasing inquiry interval `eDiscovery` can discover more peers, but at the cost of longer inquiry time. Based on the description about energy consumption of Bluetooth device discovery in Section 4.4, the longer inquiry time will lead to higher energy consumption. Moreover, the performance of `eDiscovery` is more sensitive to the change of inquiry duration. Compared with the default setup, decreasing the value of *base_W* by 4, or the value of *small_W* by 2 will reduce the device discovery probability by 36% (whereas increasing their values by the same amount can discover about only 0.4% and 6.7% more peers). In `eDiscovery`, we use the two key parameters N and I to dynamically control the values of inquiry duration and interval.

Another observation from Table 6 is that the performance of `eDiscovery` does not heavily depend on the choice of r which is determined by the probability p, because the mean of r is always 0 no matter how large or small p is. We verified this with very small values of p. For example, when $p = 0.1$, `eDiscovery` can find 1.6% more peers and increase inquiry duration by 4.3%. Moreover, increasing α from 0.5 to 1.0 has limited impact on the performance of `eDiscovery`. However, when changing from 1.0 to 1.5, `eDiscovery` discovers about 2% less peers and reduces the inquiry duration by around 2%.

8.2 Comparison of `eDiscovery`and STAR

To offer a direct apple-to-apple comparison of `eDiscovery` and STAR and evaluate their performance for more network topologies, we also port the implementation of STAR into the UCBT Bluetooth module. We summarize the simulation results for 1,000 randomly generated network topologies in Table 7.

	% of Discovered Peers	Duration of Inquiry
`eDiscovery`	77.80 (77.32, 78.28)	410.00 (408.21, 411.79)
STAR	75.83 (74.93, 76.73)	676.23 (671.40, 681.06)

TABLE 7: Performance evaluation of `eDiscovery` and STAR in the ns-2 simulator. The numbers in the parentheses are the 95% confidence intervals.

	% of Discovered Peers	Duration of Inquiry
`eDiscovery`	85.55 (85.12, 85.98)	417.70 (416.09, 419.31)
STAR	82.50 (81.58, 83.42)	689.33 (684.91, 693.75)

TABLE 8: Performance evaluation of `eDiscovery` and STAR in the ns-2 simulator with interlaced inquiry scan. The numbers in the parentheses are the 95% confidence intervals.

The simulation setup is similar to that in Section 8.1. To validate the experimental results in Section 7, we distribute Bluetooth devices in the simulation area based on the characteristics of our collected traces. More specifically, we divide the area into five regions and the device density of regions 1, 3 and 5 is much higher than that of regions 2 and 4, similar to the distribution illustrated in Figure 9. We set the Bluetooth communication range to be 10 meters.

The two metrics that we are interested in are the percentage of discovered peers and the duration of Bluetooth inquiry. As we can see from Table 7, although `eDiscovery` discovers only slightly more peers than STAR, the standard deviation of the percentage of discovered peers is much smaller for `eDiscovery` than STAR. Moreover, the duration of Bluetooth inquiry in `eDiscovery` is only around 60% of that of STAR, which confirms the energy-efficiency feature of `eDiscovery` because shorter inquiry time means less energy consumption. Both `eDiscovery` and STAR discover more peers in the simulations than in the field studies. One of the possible reasons may be that there is no co-channel interference considered in the ns-2 simulator. When two Bluetooth devices are in the communication range of each other (one of them is in the inquiring mode and another in the scanning mode), the discovery probability is very close to 1.0, which is not true in practice. This high device discovery probability in the ns-2 simulator also decreases `eDiscovery`'s room for improvements.

In addition to the standard inquiry scan mode described in Section 2, Bluetooth version 2.1 also introduces an optional interlaced inquiry scan mode to increase the discovery probability. When in the interlaced inquiry scan mode, a Bluetooth device performs two back to back scans, where the first one is on the normal hop frequency f_{scan} and the second one is on frequency (f_{scan} + 16) mod 32. This means that the two inquiry scan frequencies will be in different trains.

It is hard to evaluate the performance of device discovery protocols with the interlaced inquiry scan mode in practice because by default Bluetooth devices use the standard inquiry scan mode and it is impossible to change this setting on the discovered mobile phones in our field studies. Thus, we also evaluate the performance of `eDiscovery` and STAR in the ns-2 simulator with the interlaced inquiry scan mode enabled and report the simulation results in Table 8. By comparing Table 8 with Table 7, we can see that interlaced inquiry scans can increase the number of discovered peers, but at the same time also increase the duration of Bluetooth inquiry. Still, `eDiscovery` outperforms STAR when the interlaced inquiry mode is enabled.

9 Discussion

In this section, we discuss the limitations of this paper and some possible extensions of `eDiscovery`.

9.1 Bluetooth Low Energy

As pointed out by Liu et al. [19], although the wide-range parameter settings of Bluetooth LE (in Section 2.2) offer the flexibility for devices to customize the discovery performance, improper settings could significantly increase the latency and energy consumption of Bluetooth LE device discovery. The device discovery procedure for Bluetooth LE looks simpler than that of the classical Bluetooth (e.g., smaller number of channels and the elimination of switching between two trains). However, they share fundamentally the same design principle of interleaving between the transmission of multiple inquiry messages/advertising packets and staying in the idle mode to save energy. We can extend our proposed scheme about how to dynamically change the duration and interval of classical Bluetooth inquiry to control the duration of advertising events (i.e., the number of advertising packets to send out) and the *advInterval* in a similar way, and thus further reduce the energy consumption for device discovery in Bluetooth LE. We leave this extension as our future work.

9.2 Other Extensions

Device discovery is only the first step of opportunistic communications. The next two steps are service discovery and data transfer. There are several options of service discovery. We can exploit the standard service discovery protocol of Bluetooth [3], or develop our own protocols.

We plan to leverage multiple radio interfaces on smartphones, such as Bluetooth and WiFi, for opportunistic data transfer. These interfaces usually have different communication ranges and diverse radio characteristics. Pering et al. [22] have demonstrated the benefits of energy reduction by switching between these interfaces for mobile applications. In our case, Bluetooth may be suitable for short data transfer due to its low-power nature. For transmissions of large amounts of data, WiFi may be more desirable, because its data rate is higher and its communication range is much longer than Bluetooth. Although WiFi is not energy efficient for device discovery, we can still enable it for data transfer after mobile phones discover each other through Bluetooth.

9.3 Limitations

Although we have evaluated the performance of `eDiscovery` in three different realistic environments, the major limitation of the evaluation is that we had no control of other mobile phones during the experiments. If all the mobile phones perform Bluetooth device discovery, the number of phones discovered by us may be changed, as Bluetooth devices that are in inquiry state at the same time cannot discover each other [14], [23]. During our field experiments, most of the discovered Bluetooth devices were probably in discoverable mode only. Running experiments on mobile testbeds, such as CrowdLab [8], may solve this problem.

Another limitation of our work presented in this paper is that we have evaluated the performance of `eDiscovery` on only Nokia N900 smartphones. We are planing to port `eDiscovery` to other smartphone platforms, such as Android and iPhones, and evaluate its performance on them.

10 Conclusion

In this paper, we present `eDiscovery`, an adaptive device discovery protocol for reducing energy consumption of smartphone-based opportunistic communications. To choose the underlying communication technology, we measured the power of Bluetooth and WiFi device discovery on Nokia N900 and HTC Hero smartphones. Based on the measurement results, we prefer Bluetooth to WiFi because Bluetooth is more energy efficient for device discovery. `eDiscovery` dynamically changes the Bluetooth inquiry duration and interval to adapt to dynamic environments. We verify the effectiveness of `eDiscovery` through the first experimental field study of Bluetooth device discovery in three different environments, using a prototype implementation on smartphones. We are currently working on a more extensive evaluation of `eDiscovery` to further improve its performance.

Acknowledgement

We thank the anonymous reviewers for their insightful comments. We thank Min Mao and Yu Xiao for their help on measurement setup. We also thank Pan Hui for his support with devices and equipment. The work of B. Han and A. Srinivasan was supported in part by the US National Science Foundation (NSF) under ITR Award CNS-0426683, NSF Award CNS-0626636, and NSF Award CNS 1010789. The work of J. Li was supported in part by the National Basic Research Program of China under Grants 2011CBA00300 and 2011CBA00301 and the National Natural Science Foundation of China under Grants 61202009, 61033001, 61061130540, and 61073174.

References

[1] L. Aalto, N. Göthlin, J. Korhonen, and T. Ojala. Bluetooth and WAP Push Based Location-Aware Mobile Advertising System. In *Proceedings of MobiSys 2004*, pages 49–58, June 2004.

[2] M. Bakht, M. Trower, and R. H. Kravets. Searchlight: Won't You Be My Neighbor? In *Proceedings of MOBICOM 2012*, pages 185–196, Aug. 2012.

[3] Bluetooth Special Interest Group. Specification of the Bluetooth System, Version 2.1 + EDR, July 2007.

[4] Bluetooth Special Interest Group. Specification of the Bluetooth System, Version 4.0, June 2010.

[5] G. Chakraborty, K. Naik, D. Chakraborty, N. Shiratori, and D. Wei. Analysis of the Bluetooth device discovery protocol. *Wireless Networks*, 16(2):421–436, Feb. 2010.

[6] R. Cohen and B. Kapchits. Continuous Neighbor Discovery in Asynchronous Sensor Networks. *IEEE/ACM Transactions on Networking*, 19(1):69–79, Feb. 2011.

[7] M. Conti, S. Giordano, M. May, and A. Passarella. From Opportunistic Networks to Opportunistic Computing. *IEEE Communications Magazine*, 48(9):126–139, Sept. 2010.

[8] E. Cuervo, P. Gilbert, B. Wu, and L. P. Cox. CrowdLab: An Architecture for Volunteer Mobile Testbeds. In *Proceedings of COMSNETS 2011*, pages 1–10, Jan. 2011.

[9] C. Drula, C. Amza, F. Rousseau, and A. Duda. Adaptive Energy Conserving Algorithms for Neighbor Discovery in Opportunistic Bluetooth Networks. *IEEE Journal on Selected Areas in Communications*, 25(1):96–107, Jan. 2007.

[10] P. Dutta and D. Culler. Practical Asynchronous Neighbor Discovery and Rendezvous for Mobile Sensing Applications. In *Proceedings of SenSys 2008*, pages 71–83, Nov. 2008.

[11] R. Friedman, A. Kogan, and Y. Krivolapov. On Power and Throughput Tradeoffs of WiFi and Bluetooth in Smartphones. In *Proceedings of INFOCOM 2011*, pages 900–908, Apr. 2011.

[12] S. Gollakota, F. Adib, D. Katabi, and S. Seshan. Clearing the RF Smog: Making 802.11 Robust to Cross-Technology Interference. In *Proceedings of SIGCOMM 2011*, pages 170–181, Aug. 2011.

[13] M. Grossglauser and D. N. C. Tse. Mobility Increases the Capacity of Ad Hoc Wireless Networks. *IEEE/ACM Transactions on Networking*, 10(4):477–486, Aug. 2002.

[14] B. Han, P. Hui, V. S. A. Kumar, M. V. Marathe, J. Shao, and A. Srinivasan. Mobile Data Offloading through Opportunistic Communications and Social Participation. *IEEE Transactions on Mobile Computing*, 11(5):821–834, May 2012.

[15] B. Han and A. Srinivasan. eDiscovery: Energy Efficient Device Discovery for Mobile Opportunistic Communications. In *Proceedings of ICNP 2012*, pages 1–10, Oct.-Nov. 2012.

[16] D. B. Johnson and D. A. Maltz. *Mobile Computing*, chapter Dynamic Source Routing in Ad Hoc Wireless Networks, pages 153–181. Kluwer Academic Publishers, 1996.

[17] A. Kandhalu, K. Lakshmanan, and R. R. Rajkumar. U-Connect: A Low-Latency Energy-Efficient Asynchronous Neighbor Discovery Protocol. In *Proceedings of IPSN 2010*, pages 350–361, Apr. 2010.

[18] M. Liberatore, B. N. Levine, and C. Barakat. Maximizing Transfer Opportunities in Bluetooth DTNs. In *Proceedings of CoNEXT 2006*, pages 1–11, Dec. 2006.

[19] J. Liu, C. Chen, and Y. Ma. Modeling and Performance Analysis of Device Discovery in Bluetooth Low Energy Networks. In *Proceedings of GLOBECOM 2012*, pages 1538–1543, Dec. 2012.

[20] M. J. McGlynn and S. A. Borbash. Birthday Protocols for Low Energy Deployment and Flexible Neighbor Discovery in Ad Hoc Wireless Networks. In *Proceedings of MOBIHOC 2001*, pages 137–145, Oct. 2001.

[21] L. McNamara, C. Mascolo, and L. Capra. Media Sharing based on Colocation Prediction in Urban Transport. In *Proceedings of MOBICOM 2008*, pages 58–69, Sept. 2008.

[22] T. Pering, Y. Agarwal, R. Gupta, and R. Want. CoolSpots: Reducing the Power Consumption of Wireless Mobile Devices with Multiple Radio Interfaces. In *Proceedings of MobiSys 2006*, pages 220–232, June 2006.

[23] B. S. Peterson, R. O. Baldwin, and J. P. Kharoufeh. Bluetooth Inquiry Time Characterization and Selection. *IEEE Transactions on Mobile Computing*, 5(9):1173–1187, Sept. 2006.

[24] N. Ristanovic, G. Theodorakopoulos, and J.-Y. L. Boudec. Traps and Pitfalls of Using Contact Traces in Performance Studies of Opportunistic Networks. In *Proceedings of INFOCOM 2012*, pages 1377–1385, Mar. 2012.

[25] T. Salonidis, P. Bhagwat, and L. Tassiulas. Proximity Awareness and Fast Connection Establishment in Bluetooth. In *Proceedings of MOBIHOC 2000*, pages 141–142, Aug. 2000.

[26] A. Sharma, V. Navda, R. Ramjee, V. N. Padmanabhan, and E. M. Belding. Cool-Tether: Energy Efficient On-the-fly WiFi Hot-spots using Mobile Phones. In *Proceedings of CoNEXT 2009*, pages 109–120, Dec. 2009.

[27] T. J. Smith, S. Saroiu, and A. Wolman. BlueMonarch: A System for Evaluating Bluetooth Applications in the Wild. In *Proceedings of MobiSys 2009*, pages 41–53, June 2009.

[28] S. Vasudevan, D. Towsley, D. Goeckel, and R. Khalili. Neighbor Discovery in Wireless Networks and the Coupon Collector's Problem. In *Proceedings of MOBICOM 2009*, pages 181–192, Sept. 2009.

[29] W. Wang, V. Srinivasan, and M. Motani. Adaptive Contact Probing Mechanisms for Delay Tolerant Applications. In *Proceedings of MOBICOM 2007*, pages 230–241, Sept. 2007.

[30] X. Wu, S. Tavildar, S. Shakkottai, T. Richardson, J. Li, R. Laroia, and A. Jovicic. FlashLinQ: A Synchronous Distributed Scheduler for Peer-to-Peer Ad Hoc Networks. In *Proceedings of Allerton 2010*, pages 514–521, Sept.-Oct. 2010.

[31] Z. Yang, B. Zhang, J. Dai, A. Champion, D. Xuan, and D. Li. E-SmallTalker: A Distributed Mobile System for Social Networking in Physical Proximity. In *Proceedings of ICDCS 2010*, pages 468–477, June 2010.

[32] W. Zhao, M. Ammar, and E. Zegura. A Message Ferrying Approach for Data Delivery in Sparse Mobile Ad Hoc Networks. In *Proceedings of MOBIHOC 2004*, pages 187–198, May 2004.

Bo Han received the Bachelor's degree in Computer Science and Technology from Tsinghua University in 2000, the M.Phil. degree in Computer Science from City University of Hong Kong in 2006 and the Ph.D. degree in Computer Science from the University of Maryland in 2012. He is currently a senior member of technical staff at AT&T Labs Research. He interned at AT&T Labs Research in the summers of 2007, 2008 and 2009, Deutsche Telekom Laboratories for the summer of 2010, and HP Labs during the summer of 2011. His research interests are in the areas of wireless networking, mobile computing, software defined networking and network functions virtualization, with a focus on developing simple yet efficient and elegant solutions for real-world networking and systems problems.

Jian Li is an Assistant Professor at the Institute for Interdisciplinary Information Sciences, Tsinghua University. He got his BSc degree from Sun Yat-sen (Zhongshan) University, China, MSc degree in Computer Science from Fudan University, China and Ph.D. degree in the University of Maryland, USA. His research interests lie in the areas of algorithms, databases and wireless sensor networks. He has co-authored several research papers that have been published in major computer science conferences and journals. He received best paper awards at VLDB 2009 and ESA 2010.

Aravind Srinivasan (Fellow, IEEE) is a Professor (Dept. of Computer Science and Institute for Advanced Computer Studies) at the University of Maryland, College Park. He received his degrees from Cornell University (Ph.D.) and the Indian Institute of Technology, Madras (B.Tech.). His research interests are in randomized algorithms, networking, social networks, and combinatorial optimization, as well as in the growing confluence of algorithms, networks, and randomness in society, in fields including the social Web and public health. He has published several papers in these areas, in journals including Nature, Journal of the ACM, IEEE/ACM Transactions on Networking, and SIAM Journal on Computing. He is an editor of three journals, and has served on the program committees of various conferences.