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

Bo Han[*], Jian Li[†] and Aravind Srinivasan[‡]

[*]AT&T Labs – Research, 1 AT&T Way, Bedminster, NJ, 07921
[†]Institute for Interdisciplinary Information Sciences, Tsinghua University, Beijing 100084, China
[‡]Department of Computer Science and the Institute for Advanced Computer Studies, University of Maryland, College Park, MD, 20742

***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 energy consumption 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.**

I. 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 [14], 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 [34], mobile social applications [21] 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], [31]. 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 *fixed* inquiry duration and interval. A recently proposed opportunistic Twitter application [26] also uses a 2-minute inquiry interval for Bluetooth device discovery. It is known that these kinds of discovery protocols are not energy efficient [31] 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 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 [24], [28]. Thus, it is not clear which of

[1]http://www.lokast.com/
[2]http://www.nintendo.com/3ds/features/
[3]http://crawdad.cs.dartmouth.edu/
[4]We prefer Bluetooth and WiFi to 3G, as they are local communication technologies with almost no monetary cost.

them is more suitable for device discovery on smartphones.

Second, given the dynamic nature of human mobility, we need to adaptively tune the parameters of device discovery, such as inquiry duration and interval, to reduce smartphone energy consumption. Schemes with constant inquiry intervals have been proven to be optimal in terms of minimizing discovery-missing probability [31]. 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 and the discovery duration (Section IV). Based on our measurement results, we chose Bluetooth as the underlying wireless technology. We emphasize that although previous works have studied the power of Bluetooth/WiFi devices [9], [11], [24], they either focus on only Bluetooth [9] or ignore the duration of device discovery [11], [24], 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 VI). 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 VII). 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. [31], `eDiscovery` consumes less energy and discovers more peers. `eDiscovery` also performs better than another protocol in the literature. We also port the implementations of `eDiscovery` and STAR into the ns-2 simulator enhanced with the UCBT Bluetooth module[5] and get similar simulation results (Section VIII).

II. Related Work

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

A. Wireless Device Discovery in General

Device discovery has been widely studied in various wireless networks, such as ad-hoc networks [20], [30], mobile sensor networks [10], [17] and delay-tolerant networks [31].

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. Inspired by the birthday paradox, a pair of nodes perform neighbor discovery by transmitting and listening on k independently and randomly chosen slots among n slots (the ratio k/n is relatively small). Vasudevan et al. [30] 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. Differently from 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. Disco can address the challenge of operating the radios at a low duty cycle and ensuring fast and reliable discovery in bounded time through the adaptation of the Chinese Remainder Theorem. 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.

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. To leverage the discovered neighbor relationship during the initialization, they propose to make sensors in a connected segment collaborate on the discovery task and thus speed up the discovery of a new sensor node.

The goal of `eDiscovery` is similar in spirit to that of Wang et al. [31] 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 also design a contact probing algorithm, named STAR (Short Term Arrival Rate), to dynamically change the contact probing frequency based on the contact arrival process. Without specifying the com-

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

munication technologies, they assume that every probing message is just an impulse and consumes no time. As opposed to STAR, we also dynamically change the duration of Bluetooth inquiry to further reduce the energy consumption. We compare the performance of `eDiscovery` with STAR in Section VII through extensive real-world experiments and simulation studies.

B. Bluetooth Device Discovery

Bluetooth specifies a detailed device discovery protocol [3]. Salonidis et al. [27] identify the bottlenecks of asymmetric device-discovery delay of Bluetooth. They introduce a randomized symmetric discovery protocol to reduce this delay. Based on Bluetooth specification v1.1, Peterson et al. [25] derive rigorous expressions for the inquiry-time probability distribution of two Bluetooth devices that want to discover each other and validate them through simulation studies. Chakraborty et al. [5] present an analytical model of the time of Bluetooth device discovery protocol. They investigate the discovery time pattern through extensive simulation studies.

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. Through analysis and simulation studies they demonstrate that this dual radio technique can improve both discovery duration and connection frequency. 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 (e.g., the time spent in inquiry and scan phases and the duration and interval of scan) based on recent activities and the location of previous contacts, and evaluate their performance through simulations. In our previous work [15], 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. Differently from them, we study how to dynamically change the inquiry window and interval to achieve the tradeoff between discovery-missing probability and energy efficiency. Particularly, we design an energy-aware Bluetooth device discovery protocol and evaluate its performance *in the wild* through a prototype implementation on smartphones.

C. Opportunistic Communications

There have been many applications of opportunistic communications in mobile social networks and delay-tolerant networks. To encourage social participation from mobile users in information sharing applications, Garyfalos and Almeroth [12] propose Coupons, an incentive scheme that allows users to opportunistically share data over a wireless medium. Previously, we have proposed to leverage opportunistic communications and social participation to offload cellular traffic to mobile-to-mobile communications and thus alleviate traffic load on 3G networks [15]. The above works can benefit from our proposed scheme to facilitate their opportunistic communications.

McNamara et al. [21] propose a content source selection scheme, Media Sharing, to share media content among co-located mobile users in urban transport. With this scheme, mobile devices can select the best content sources (the peers who can remain co-located long enough to complete data transfer) and perform content sharing and distribution. The authors confirm the feasibility of the proposed prediction scheme using underground transport traces collected from a large metropolitan mass transit system. Differently from Media Sharing, we aim to develop an energy efficient device discovery protocol, which is an essential step *before* the selection of the best peers.

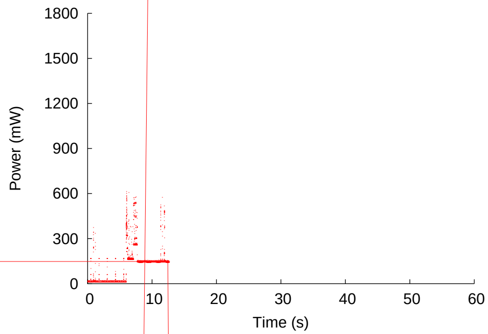
III. Device Discovery in Bluetooth and WiFi

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

A. 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 [13]. 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. [29] 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



Power (mW)
1800
1500
1200
900
600
300
0
0
10
20
30
40
50
60
Time (s)

# of Devices	Average	Standard Deviation
0	162.03	2.12
1	227.06	12.33
2	247.72	8.60
3	248.91	9.51
4	248.59	3.16
5	256.02	4.93
6	253.05	5.51

TABLE I: The electrical power (in mW) of Bluetooth device discovery with different numbers of neighboring devices.

Environment	Office	Home	Park
# of peers	43.52 (5.3)	14.02 (1.4)	0.01 (0.1)
duration (s)	1.07 (0.15)	0.87 (0.05)	0.52 (0.04)

TABLE II: The average number of discovered peers and duration of WiFi device discovery in three environments. The numbers in the parentheses are the standard deviations.

	P_{idle}	P_{probe}
Bluetooth	16.54 (1.11)	253.05 (5.51)
WiFi	791.02 (5.23)	836.65 (8.98)

TABLE III: The average power of Bluetooth and WiFi device discovery in mW.

B. 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 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 inquiry intervals (i.e., idle states), the Bluetooth radio is in discoverable mode with average power □16.54 mW (standard deviation: 1.11 mW). Note that all results of power measurements in this paper include the baseline power of the smartphone under test.

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 I. 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.

C. WiFi

We present another 60-second snapshot of the power of WiFi device discovery in Figure 2. We perform the 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 fixed 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 II. In each environment, we repeat the experiments 100 times and report the average values and standard deviations. 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.

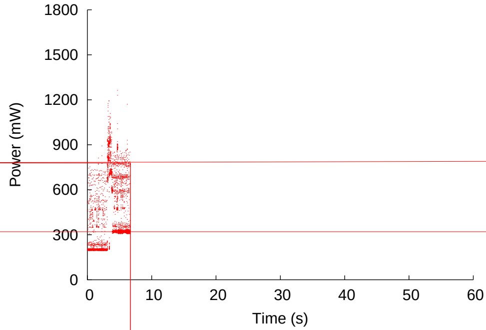
D. Energy Consumption

We summarize the average power of Bluetooth (with 6 neighboring devices) and WiFi device discovery in Table III. 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 T_{probe} and the inquiry/scan interval is T_{idle}. Then the estimated energy consumption is

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

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 [28].

Although the communication range of WiFi is longer than Bluetooth and may discover more peers, making its


1800
1500
1200
900
600
300
0
Power (mW)
0
10
20
30
40
50
60
Time (s)

1) If $k < 0$ (i.e., $L + x < T$), it is easy to see that

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

2) If $k > 0$, 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 $\lfloor k \rfloor$ periods, the two devices are in the contact range for W seconds and the missing probability is $\overline{R}(W)$. In the last possible period, the contact interval is of length $\min(y, W)$, where $y = L + x - T - \lfloor k \rfloor T$. Overall, we can see that

$$\begin{aligned} \mathsf{P}_{\mathsf{miss}}(x, L) &= \overline{R}((W - x)_+)\overline{R}(W)^{\lfloor k \rfloor}\overline{R}(\min(y, W)) \\ &= \mathsf{PS}_{\mathsf{miss}}(x, L) \end{aligned}$$

If the inter-contact time follows a nonlattice distribution, by Blackwell's Theorem in renewal theory [31], we have that, when $n \to \infty$,

$$\begin{aligned} \mathsf{P}_{\mathsf{miss}} &= \int_0^{\mathsf{T}}\int_0^{\infty} \mathsf{P}_{\mathsf{miss}}(x, L)p(L)\mathsf{d}L\mathsf{d}x \\ &= \int_0^{\mathsf{T}}\int_0^{\mathsf{T}-\mathsf{x}} \mathsf{PF}_{\mathsf{miss}}(x, L)p(L)\mathsf{d}L\mathsf{d}x \\ &+ \int_0^{\mathsf{T}}\int_{\mathsf{T}-\mathsf{x}}^{\infty} \mathsf{PS}_{\mathsf{miss}}(x, L)p(L)\mathsf{d}L\mathsf{d}x \end{aligned}$$

Similar with the analysis by Wang et al. [31], 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 by increasing the duration of inquiry window W, we can increase $R((W - x)_+)$, $R(W)$ and $R(\min(y, W))$ and thus reduce the device discovery missing probability.

VI. 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.

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 $1.28 \times$ *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 [15], [25].

The intuition behind these adaptations is that we can reduce the inquiry duration and increase the inquiry interval

Algorithm 1 Adaptive Inquiry Algorithm of `eDiscovery`

```
inquiry_window = base_W, inquiry_interval = base_I;
while (TRUE) do
    peers = hci_inquiry(inquiry_window, MAX_RSP);
    if (peers > N) then
        inquiry_window = base_W;
    else
        inquiry_window = small_W + r;
    end if
    if (peers == 0 and last_peers == 0) then
        inquiry_interval += inc_NP + r;
    else if (peers <> 0 and last_peers == 0) then
        inquiry_interval = base_I + r;
    else if (peers > last_peers) then
        inquiry_interval -= I;
    else if (peers < last_peers) then
        inquiry_interval += I;
    end if
    last_peers = peers;
    sleep(inquiry_interval);
end while
```

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

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

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 IV. These values are not set arbitrarily. We set the initial *inquiry_window* to be 8 (i.e., $8 \times 1.28 = 10.24$ 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. [25] 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

Parameter / Value		% of Discovered Peers	Duration of Inquiry
Default		74.39 (6.86)	581.33 (32.79)
base_W	4	47.48 (9.05)	323.71 (36.96)
	12	74.72 (6.98)	585.69 (33.15)
base_I	6	76.62 (7.05)	693.54 (44.42)
	14	68.42 (6.96)	490.58 (26.15)
small_W	3	47.47 (8.97)	328.24 (37.55)
	7	79.40 (5.94)	748.66 (30.69)
inc_NP	5	80.47 (4.95)	628.25 (19.40)
	15	69.40 (8.43)	541.01 (46.19)
p	0.9	74.91 (6.58)	585.56 (31.26)
	0.7	74.57 (7.03)	580.35 (33.91)

TABLE VI: Performance evaluation of `eDiscovery` using different parameters. The numbers in the parentheses are the standard deviations.

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

E. 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.

VIII. 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.

A. `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 IV.

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 result in Table VI. The second row of this table shows the simulation results with the default values of these parameters. The major observation from Table VI is that by increasing inquiry duration or decreasing inquiry interval `eDiscovery` can discover more peers, but at the cost of high 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 VI is that the performance of `eDiscovery` does not heavily depend on the choice of r which is determined by the probability p. The reason is that the mean of r is always 0 no matter how large or small p is.

	% of Discovered Peers	Duration of Inquiry
`eDiscovery`	77.80 (7.78)	410.00 (28.92)
STAR	75.83 (14.5)	676.23 (77.90)

TABLE VII: Performance evaluation of `eDiscovery` and STAR in the ns-2 simulator. The numbers in the parentheses are the standard deviations.

B. 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 generated network topologies in Table VII. The simulation setup is similar to that in Section VIII-A. To validate the experimental results in Section VII, 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 VII, 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`. 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

	% of Discovered Peers	Duration of Inquiry
`eDiscovery`	85.55 (6.99)	417.70 (25.91)
STAR	82.50 (14.8)	689.33 (71.39)

TABLE VIII: Performance evaluation of `eDiscovery` and STAR in the ns-2 simulator with interlaced inquiry scan. The numbers in the parentheses are the standard deviations.

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 III, 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 $f_{\texttt{scan}}$ and the second one is on frequency ($f_{\texttt{scan}}$ + 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 VIII. By comparing Table VIII with Table VII, 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.

IX. Discussion

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

A. 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 fixed *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.

As pointed out by Liu et al. [19], although these wide-range parameter settings 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. Although 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), 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.

B. 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. [24] 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.

C. 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 [15], [25]. 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.

D. Privacy and Security

Currently, the number of smartphones with their Bluetooth devices in discoverable mode is low, around 100 during our 30-minute experiments, mainly due to the privacy concerns of mobile users. We believe that with the proliferation of mobile social applications, such as PeopleNet [22] and E-SmallTalker [33], that leverage mobile-to-mobile opportunistic communications, more and more people will be willing to tune the Bluetooth radio on their smartphones to discoverable mode if we can address the potential security threats.

Security has been a key research challenge in device discovery in wireless networks. Security concerns for opportunistic communications vary depending on the environment and upper-layer applications. Existing approaches have exploited distance bounding, location information and directional antennas to secure mobile device discovery [23]. We plan to investigate how to tackle security issues in Bluetooth device discovery for smartphone-based opportunistic communications as our future work.

E. Other Device Discovery Technologies

There have been other technologies proposed especially to perform device discovery. For example, FlashLinQ [32] is a synchronous wireless PHY/MAC network architecture developed by Qualcomm for direct device-to-device communication over licensed spectrum. It aims to support various applications of proximate Internet, including social networking and mobile advertising. FlashLinQ enables automatic and continuous device discovery and peer-to-peer communication between mobile devices.

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. Differently from FlashLinQ, we aim to design and implement device discovery protocols using existing hardware and communication technologies available on commercial smartphones.

X. 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.

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] A. Garyfalos and K. C. Almeroth. Coupons: A Multilevel Incentive Scheme for Information Dissemination in Mobile Networks. *IEEE Transactions on Mobile Computing*, 7(6):792–804, June 2008.
[13] 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.
[14] 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.
[15] 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.
[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] M. Motani, V. Srinivasan, and P. S. Nuggehalli. PeopleNet: Engineering A Wireless Virtual Social Network. In *Proceedings of MOBICOM 2005*, pages 243–257, Aug.-Sept. 2005.

[23] P. Papadimitratos, M. Poturalski, P. Schaller, P. Lafourcade, D. Basin, S. Capkun, and J.-P. Hubaux. Secure Neighborhood Discovery: A Fundamental Element for Mobile Ad Hoc Networking. *IEEE Communications Magazine*, 46(2):132–139, Feb. 2008.

[24] 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.

[25] 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.

[26] 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.

[27] 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.

[28] 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.

[29] 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.

[30] 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.

[31] 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.

[32] 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.

[33] 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.

[34] 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 and mobile computing, 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 co-authored several research papers that have been published in major computer science conferences and journals. He received the 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.