Searchlight: Helping Mobile Devices find their Neighbors

Mehedi Bakht
Dept. of Computer Science
University of Illinois at Urbana-Champaign
mbakht2@illinois.edu

Matt Trower
Dept. of Computer Science
University of Illinois at Urbana-Champaign
mtrower2@illinois.edu

Robin Kravets
Dept. of Computer Science
University of Illinois at Urbana-Champaign
rhk@illinois.edu

ABSTRACT

The rapid deployment of millions of handheld communication devices has resulted in a demand for physical proximity-based opportunistic networking. However, the success of these emerging ad hoc networks requires that a device should be able to search and find other devices in its vicinity without infrastructure support, without consuming too much battery power, and preferably without requiring clock synchronization. While approaches exist to solve this problem of energy-efficient asynchronous neighbor discovery, they present an unpleasant trade-off between good average-case performance (probabilistic approaches) and strict bound on worst-case discovery latency (deterministic approaches). In response to these limitations, we present Searchlight, an asynchronous neighbor discovery protocol that has both deterministic and probabilistic components, a novel combination that enables it to have *both* good average-case performance and the best worst-case bound for any given energy budget.

1. INTRODUCTION

Every day, new applications are popping up to take advantage of the widespread availability of handheld wireless devices like smart phones and game consoles. While existing communication support has focused on connectivity to the wireless infrastructure, most of these devices are also equipped with radios that support direct device-to-device communication. This ad hoc networking potential can be exploited to support peer-to-peer communication based on physical proximity [4], including social networking services that operate without infrastructure support (e.g.,Lokast [1], MobiClique [10], WiFace [12]) and game services (e.g., Nintendo's StreetPass [2], Sony's Near [3]). All these applications rely on a node's ability to detect the presence of other nodes in its transmission range. However, for devices running on battery, it is not practical to continuously search for neighbors. A more feasible approach is to keep the wireless interface in a sleep state most of the time and periodically wake it up to execute a discovery process.

The success of such duty-cycling schemes depends on ensuring that the wakeup times of two neighboring nodes overlap. This is not hard to achieve when node clocks can be synchronized. But synchronization requires infrastructure support (3G/WiFi) or use of an on-board GPS component, which is usually too energy-expensive for smartphones [9]. This has necessitated the design of periodic schemes that ensure such overlap within a reasonable time bound while operating at low duty-cycles and requiring no clock synchronization.

The main differentiator among existing asynchronous neighbor discovery protocols is discovery time given a common energy budget. Unfortunately, there is no clear winner since current approaches present a trade off between average-case and worst-case discovery latency. The current best protocols for average-case discovery latency choose probabilistic wakeup times (e.g., Birthday protocol [7]) but cannot provide any bound for worst-case delay. On the other hand, current protocols that deterministically design a schedule for wakeup times and guarantee overlap within a reasonable bound (i.e., Quorum-based protocols [11], Disco [5] and U-Connect [6]) perform poorly in the average case.

In response to these limitations, we have designed Searchlight, a hybrid protocol that combines both deterministic and probabilistic techniques to achieve average-case performance comparable to the probabilistic protocols and provides the best worst-case bound for a given energy budget when all nodes have similar energy requirements. Simulation results show that Searchlight performs better than all existing protocols in terms of average discovery latency, and improvement over the best existing deterministic approach is *at least* 25%. We also implemented Searchlight on a testbed consisting of Nokia N900 phones that provided valuable new insight into issues related to the execution of such protocols on smartphones with Wi-Fi interfaces.

The rest of this paper is organized as follows. Section 2 briefly describes existing approaches to asynchronous neighbor discovery and their limitations. Section 3 describes Searchlight in detail. Section 4 presents simulation-based performance evaluation of Searchlight. A prototype implementation on a smartphone testbed is described in Section 5. We conclude and outline the direction of our future research in Section 6.

2. ASYNCHRONOUS NEIGHBOR DISCOVERY

Asynchronous neighbor discovery algorithms mostly work

MobiHeld '11, October 23, 2011, Cascais, Portugal.

on a time-slot basis, where time is assumed to be divided into slots of equal size and all nodes agree on the size of a slot. Based on the protocol used, nodes decide to remain awake during specific slots, which are called *active* slots, and sleep during the remaining slots. During an active slot, the node may send/receive or do both, depending on application requirements. Successful discovery takes place between two neighbors whenever their active slots overlap. To be energy efficient, a discovery scheme needs to use as few active slots as possible to discover neighbors within a reasonable time limit. Current approaches to energy-efficient asynchronous neighbor discovery fall broadly into two categories - probabilistic and deterministic.

Most well-known among probabilistic approaches is a family of "birthday protocols" [7] where nodes transmit/receive or sleep with different probabilities. This scheme works well in the average case. However, its main drawback is its failure to provide a bound on the worst case discovery latency, leading to long tails on discovery probabilities.

Deterministic protocols overcome this limitation by providing a strict bound on worst-case latency. In the Quorum-based protocols [11], time is divided into sets of m^2 contiguous intervals that are arranged as a 2-dimensional $m \times m$ array. Each host can pick one row and one column of entries as awake intervals, which ensures that no matter what row and column are chosen, there will be at least two overlaps every m^2 slots. In Disco, each node chooses a pair of prime numbers and then wakes up at multiples of the individual prime numbers. If one node chooses primes p_1, p_2 and another node chooses p_3, p_4, the worst-case discovery latency between these two nodes will be $min\{(p_1 \cdot p_3), (p_1 \cdot p_4), (p_2 \cdot p_3), (p_2 \cdot p_4)\}$, provided the two primes in the pair are not equal. U-Connect [6] uses a single prime per node, p. Nodes wake up 1 slot every p slots, and additional $\frac{p+1}{2}$ slots every p^2 slots. The worst-case latency for U-Connect is p^2. Although these deterministic protocols have good worst-case performance, in the average case, they are considerably worse than the birthday protocol.

To successfully meet the two goals of good average-case performance and reasonable worst-case latency, we present a new protocol named Searchlight. Searchlight follows a deterministic approach and provides a strict bound on worst-case latency, which is provably *the best* among existing protocols when nodes operate with similar energy constraints. Additionally, it also incorporates randomization techniques that result in discovery latency very close to the probabilistic approach in the average case.

3. SEARCHLIGHT

Searchlight is an asynchronous periodic slot-based discovery scheme, where a period consists of t contiguous slots as determined by the target duty cycle. In every period (i.e., t slots), there are two active slots. The first active slot, called the *anchor* slot, is the first slot in the period. Since the position of this anchor slot is fixed in t but the start times for the ts for different nodes vary, the anchor slots for two nodes only overlap if the difference between the start times of the two periods is less than one timeslot. For all other offsets, assuming no clock drift, the two anchor slots would never meet since the offset remains constant when two nodes use the same value for t. Essentially, the relative position of the anchor slot of one node always remains the same with respect to that of the other node and is in the range $[1, t-1]$.

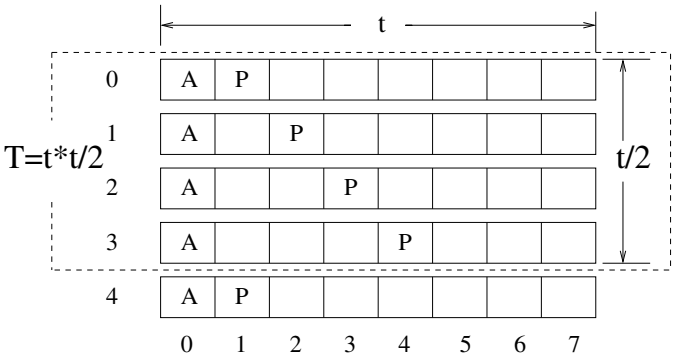

Figure 1: Searchlight with sequential probing (t=8)

Since most anchor slots miss each other, Searchlight introduces a second active slot in t called the *probe* slot, which systematically searches for the anchor slot of the other node.

3.1 Systematic Probing

Given the stability of the relative offset, no two nodes can be offset by more than $\frac{t}{2}$ slots. Therefore, Searchlight only needs to probe the first half of the t slots to guarantee an overlap between the probe slot of one node and the anchor slot of the other node. A simple way to perform this probing is to move the probe slot sequentially. Essentially, the position of the probe slot can be determined by a counter that starts at 1, increments by 1 every period, ends at $\lfloor \frac{t}{2} \rfloor$ and then starts at 1 again (see Figure 3). In other words, if P_i denotes the position of the probe slot in the i-th period, then

$$P_{i+1} = ((P_i) \mod \lfloor \frac{t}{2} \rfloor) + 1. \quad (1)$$

The position of the probe slot actually follows the periodic pattern $\{1, 2, ..., \lfloor \frac{t}{2} \rfloor\}$ and this pattern gets repeated every $\lfloor \frac{t}{2} \rfloor$ periods, which we call the hyper-period T. For example, for $t = 8$, Searchlight uses the periodic pattern {1,2,3,4} to determine the position of the probe slot in each period.

When nodes operate with the same t, it can be easily proved that the worst-case discovery latency under Searchlight is equal to $\frac{t^2}{2}$ slots. To judge whether this bound is worse or better than existing approaches, we use an energy-latency metric.

3.2 Energy-Latency Metric

Energy-latency, Λ [6], is a metric that is defined as the product of the average energy consumption (in terms of active slots) P and the worst-case neighbor discovery latency L in an ideal communication channel.

To evaluate Searchlight using this metric, we start with the worst-case latency L_s:

$$L_s = \frac{t}{2} \text{ periods } = \frac{t^2}{2} \text{ slots }. \quad (2)$$

Next, average energy consumption P is the same as duty cycle. In Searchlight, there are two active slots every t slots. So, the duty cycle is $\frac{2}{t}$.

$$P_s = \frac{2}{t}. \quad (3)$$

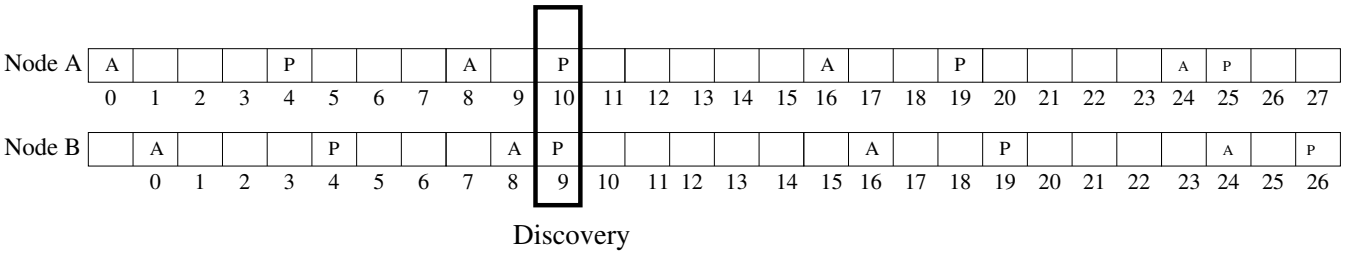

Figure 2: Overlap with randomized probing (t=8)

From equations (2) and (3), Λ_s for Searchlight is:

$$\Lambda_s = P_s L_s = \sqrt{2L_s}. \quad (4)$$

For the theoretically optimal schedule [6, 13], the PL product is:

$$\Lambda_o = \sqrt{L - \frac{3}{4}} + \frac{1}{2}. \quad (5)$$

As can be seen from equations (4) and (5), asymptotically, for the symmetric asynchronous neighbor discovery problem, Searchlight is a $\sqrt{2}$-approximation algorithm.

$$\lim_{L\to\infty} \frac{\Lambda_s}{\Lambda_o} = \lim_{L\to\infty} \frac{\sqrt{2L}}{\sqrt{L - \frac{3}{4}} + \frac{1}{2}} = \sqrt{2}. \quad (6)$$

On the other hand, U-Connect is an 1.5-approximation algorithm, while both Disco and Quorum are 2-approximation algorithms [6]. This analysis shows that using the same energy, Searchlight provides the *best* worst-case bound.

3.3 Randomized Probing

Sequentially moving the probe slot does a good job of ensuring discovery by finding the anchor slot of the other node (i.e., probe-anchor overlap). However, the overlap of two probe slots should also result in a successful discovery. But in *sequential* probing, which we will refer to as Searchlight-S, the probe slots of two nodes follow the same pattern, and hence they are often in sync with each other, greatly reducing the probability of a probe-probe overlap.

To increase the probability of a probe-probe overlap, Searchlight introduces a probabilistic component, similar to the Birthday protocol. In this version of Searchlight, which we call Searchlight-R, instead of being restricted to the pattern $1, 2, 3...\lfloor \frac{t}{2} \rfloor$, nodes can *randomly* pick any probe slot pattern that is a permutation of values from 1 to $\lfloor \frac{t}{2} \rfloor$.

For example, assume that two nodes A and B are neighbors, they have a relative phase offset of one slot, and $t = 8$. Instead of being restricted to just {1,2,3,4}, Searchlight-R allows the two nodes to randomly choose any pattern that is a permutation of the integers 1,2,3 and 4 (e.g., {1,4,3,2}, {1,2,4,3}). Let A randomly choose the periodic pattern {1,4,2,3} while its neighbor B chooses the periodic pattern {1,3,2,4} (see Figure 3). When they first meet, the probe slots of A and B are at positions 4 and 4 respectively and their relative offset is 1 slot. Because of the phase offset, the probe slots miss each other initially but meet in the next period when A's probe slot moves to position 2 and B's probe slot moves to position 1. This probabilistic approach essentially increases the possibility of discovery through a probe-probe overlap without changing the worst-case bound.

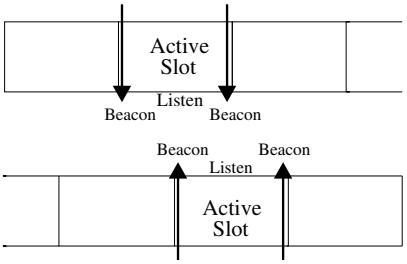

Figure 3: Beaconing in an active slot

3.4 Slot non-alignment

For illustration purpose, we have shown the slots to be aligned in previous figures. However, Searchlight is designed as an asynchronous protocol and hence does not assume or rely on the alignment of slot boundaries at different nodes. To ensure that an overlap between two active slots always leads to discovery, a beacon gets sent both at the beginning and end of an active slot and the node remains in listening mode in the intermediate period (see Figure 3). Because of this approach, non-alignment of slot boundaries actually results in lower discovery latency in general, since each slot overlaps with two slots of the other node and discovery can happen from either of the two overlaps. However, the very rare case of the slot boundaries being completely aligned can result in interference and failure in discovery. Like Disco [5], we leave it to the MAC layer to handle such situations.

3.5 Effects of clock drift

Searchlight's systematic probing relies on the assumption of a constant offset between the anchor slots of any two nodes. In reality, clocks drift and the offset between anchor slots changes over time. However, if the total drift within the worst-case bound is less than half of the slot width, discovery can be guaranteed within that bound.

3.6 Duty Cycle Asymmetry

Searchlight only requires one modification to operate in an asymmetric environment: t must be prime. Restricting t to be a prime number, similar to Disco and U-Connect, ensures that for any two nodes operating at different duty cycles, their period lengths t_1 and t_2 will have no common factors other than 1. Since period lengths are relatively prime, it follows from the Chinese Remainder Theorem [8] that the two anchor slots should overlap at least once every $t_1 \cdot t_2$ slots, where t_1 and t_2 are the two different period lengths. Since it can be expected that more often than not mobile

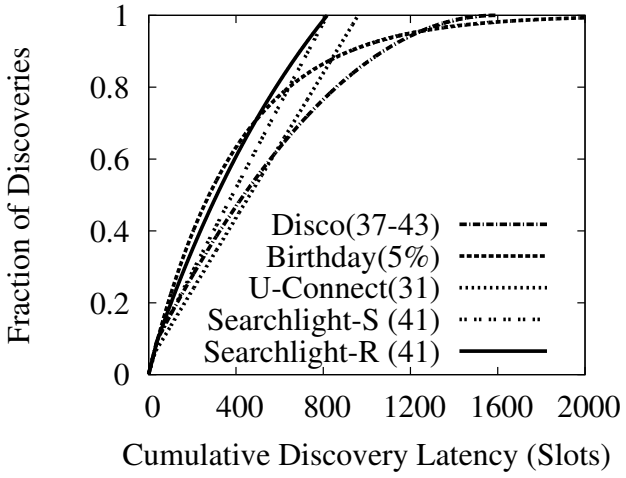

Figure 4: State-based simulation: 5% duty cycle

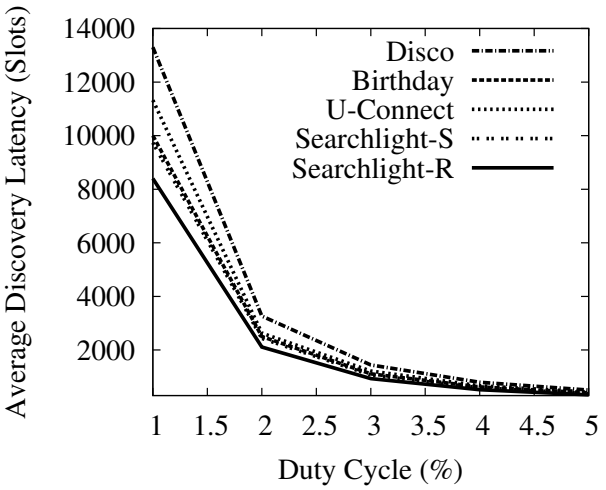

Figure 5: Average Latency vs. Duty Cycle

handheld devices will have similar energy constraints and would operate at the same duty cycle, in this paper we focus only on symmetric operation.

4. EVALUATION

The primary goal of our evaluation is to measure how long it takes for different protocols to discover neighbors when *they spend the same amount of energy*, i.e., operate at the same duty cycle. The time from when two nodes first get in each other's transmission range to the time when they actually discover each other is known as the *discovery latency*. Since absolute latency in terms of time units is dependent on slot-width and slot-width is platform dependent and the same for all protocols, we use number of slots as the unit for latency. We compare the performance of our approach with the deterministic protocols, Disco and U-Connect, and also with the probabilistic Birthday protocol.

4.1 State-based Simulation

For any random pair of nodes, the discovery latency can widely vary based on the relative offset. To fully characterize the performance of the different protocols, including the worst-case bound, it is essential to consider all possible offsets through a state-based simulation.

Except for the Birthday protocol, all other protocols basically follow a discovery schedule to determine when to sleep and when to wake up. This schedule repeats every T slots, which we call the hyper-period. When a node is in a particular slot in its schedule, that slot index can be considered the *state* of the node at that point and is always in the range $[0, T-1]$. When two nodes with hyper periods T_1 and T_2 come into each other's transmission range, their states have one of $T_1 \cdot T_2$ possible combinations. For a given combination, the discovery latency is always the same. For Disco, U-Connect and Searchlight-S, we use this observation to loop through all possible combinations and determine the latency for each case. The same approach is not feasible for Searchlight-R since for a given t, a node can choose any of the $(\frac{t}{2}-1)!$ patterns to determine the schedule of its probe slot. Instead, we run the protocol 1000 times with different seeds. At each run, we generate a new schedule for both nodes. Then, for that particular schedule pair, we loop through all possible state combinations like we do for Searchlight-S. For the Birthday protocol, we use a closed form expression for determining the CDF and the expected value of discovery latency [7].

4.2 Results

First, we look at the cumulative distribution of discovery latencies for all protocols operating at 5% duty cycle. To operate at this particular duty cycle, Disco uses the primes (37,43), Birthday uses probability = 0.5, U-Connect uses the prime 31 and both versions of Searchlight use the prime 41 (see Figure 4). Searchlight-R *always* achieves lower latency over all other protocols except for the Birthday protocol. For 65% of the time, Searchlight-R performs on par with the Birthday protocol or slightly lags behind. Beyond that, the probabilistic nature of the Birthday protocol leads to a long tail and Searchlight-R achieves the lowest latency. In comparison to U-Connect, Searchlight-R achieves better latency all along. For both versions of Searchlight, maximum discovery latency is 820 slots. U-Connect comes closest with a maximum discovery latency of 961 slots, which is around 20% more than Searchlight. Searchlight-S always performs better than U-Connect and Disco but lags behind Searchlight-R in the average case.

Next, we look at the average discovery latency of the protocols for different duty cycles (see Fig. 5). For *all* duty cycles, Searchlight-R achieves the lowest average latency. Since the extent of improvement is hard to discern for higher duty cycles, we also present a normalized version of the graph (see Fig. 6) where the performances of all the protocols have been normalized with respect to Searchlight-R. For *all* duty cycles, Searchlight-R reduces average latency by at least 25% for U-Connect and 16% for the Birthday protocol while the performance of Disco ranks worst by a distance. Average latency for Searchlight-S lies between that of Searchlight-R and the Birthday protocol for all duty cycles. The difference between the performance of Searchlight-R and Searchlight-S clearly demonstrates the advantage of incorporating randomization in moving the probe slot.

Overall, these results confirm that Searchlight does provide the lowest worst-case bound for a given duty-cycle as was shown earlier theoretically. In addition, the CDF of

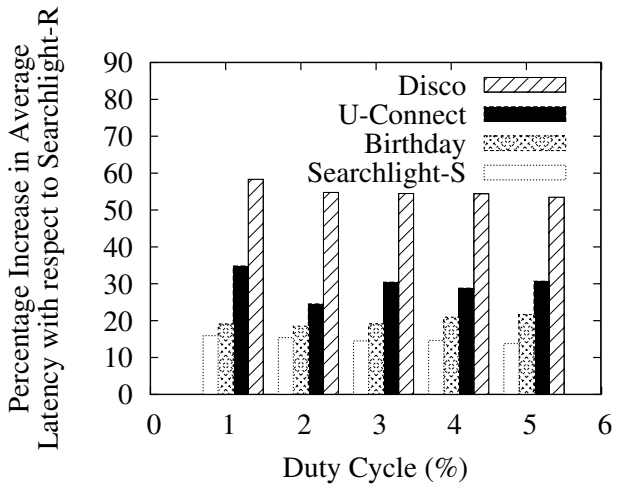

Figure 6: Increase in Average Latency with respect to Searchlight-R

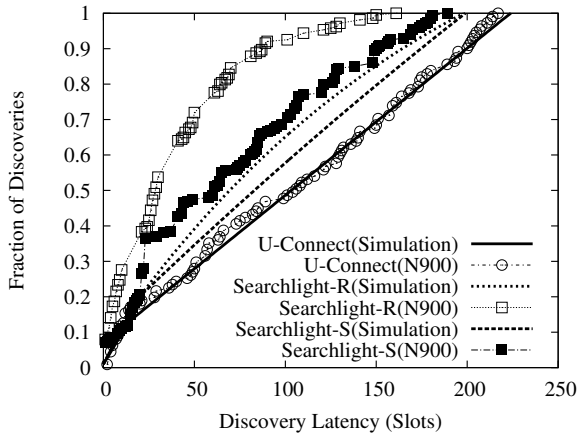

Figure 7: Implementation vs. Simulation

discovery latencies show that Searchlight-R performs on par with the Birthday protocol in the average-case. and thus provides best of both worlds.

5. IMPLEMENTATION

One of the main objectives for designing Searchlight is to facilitate ad hoc communication between handheld devices like smartphones. To gauge how successfully this goal is achieved in practice, we implemented the protocol on four Nokia N900 phones. We implemented Searchlight-S and Searchlight-R, as well as U-Connect, since it performs best among existing protocols that provide strict worst-case bounds. We first describe different implementation issues and then present some preliminary results.

5.1 Implementation Issues

We implemented Searchlight to use the WiFi radio of the N900 phone for neighbor discovery. Earlier protocols for asynchronously discovering neighbors were all implemented on sensor nodes, allowing them to use small slots on the order of milliseconds [5] or even microseconds [6]. However, unlike sensor radios, WiFi radios have a non-negligible transition latency from sleep to transmit/receive. On the N900 phone, we found out that from the application level, the time to bring the wireless interface up is around 1 to 3 seconds. Because of this latency, we decided on a slot size of 5 seconds. Other phones we have looked at, including the Android G1, also have startup latencies on the order of seconds.

Because of the non-negligible start-up time, we introduced the notion of *pre* slots. A *pre* slot basically precedes any active slot and switches on the interface. For example, assume that a node needs to be active during slot 10. Now, if the command to wake up the radio is issued at the beginning of slot 10, it will take almost the whole slot duration for that command to return and the effective awake time during that slot will be reduced to 1-2 seconds. To get around this problem, the wake up command now gets issued at the start of the preceding slot which is slot 9 in this case. When two active slots are neighbors, the radio is left on rather than using pre slots.

5.2 Evaluation

We implemented the three protocols on four N900 phones and logged the discovery latency for 150 runs with 10% duty cycle.We compared the values with our simulation results (see Figure 7). The implementation results actually turned out to be better than the state-based simulation results, with more discoveries taking place at lower latencies. Since the devices were not synchronized, this is probably an artifact of the slots being non-aligned (see section 3.4). The overall trends agree with the simulation results. Between the implementations, Searchlight fares much better than U-Connect. In fact, up to 80-th percentile, the cumulative discovery latency of Searchlight-R is just slightly more than one-third of U-Connect.

6. CONCLUSION

Solving the problem of energy efficient asynchronous neighbor discovery is an important pre-condition for more widespread use of ad hoc communication between handheld mobile devices. In this paper, we present Searchlight, a new asynchronous neighbor discovery protocol that addresses this problem by adopting a systematic probing based approach to provide better bounds on discovery latency than any existing protocol when nodes have similar energy requirements. Both simulation and implementation results show that Searchlight achieves the best average case discovery latency when nodes operate with similar energy constraints. As future work, we would like to integrate neighbor discovery with data transmission between discovered peers.

7. REFERENCES

[1] Lokast - proximity based mobile social network. http://www.lokast.com.

[2] Nintendo 3ds - streetpass. http://www.nintendo.com/3ds/hardware.

[3] Sony ps vita - near. http://us.playstation.com/psvita.

[4] A. J. Aviv, M. Sherr, M. Blaze, and J. M. Smith. Moving targets: Geographically routed human movement networks. Technical Report MS-CIS-10-12, Department of Computer & Information Science, University of Pennsylvania, 2010.

[5] P. Dutta and D. Culler. Practical asynchronous neighbor discovery and rendezvous for mobile sensing applications. In *SenSys '08: Proceedings of the 6th*

ACM conference on Embedded network sensor systems, pages 71–84, 2008.

[6] A. Kandhalu, K. Lakshmanan, and R. R. Rajkumar. U-connect: a low-latency energy-efficient asynchronous neighbor discovery protocol. In *Proceedings of IPSN '10*, pages 350–361, 2010.

[7] 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 '01*, pages 137–145. ACM, 2001.

[8] I. Niven and H. S. Zuckerman. *An Introduction to the Theory of Numbers*. John Wiley and Sons (WIE), 1991.

[9] J. Paek, J. Kim, and R. Govindan. Energy-efficient rate-adaptive gps-based positioning for smartphones. In *Proceedings of MobiSys '10*. ACM, 2010.

[10] A. K. Pietiläinen, E. Oliver, J. Lebrun, G. Varghese, and C. Diot. MobiClique: middleware for mobile social networking. In *WOSN '09: 2nd ACM workshop on Online social networks*, pages 49–54, 2009.

[11] Y.-C. Tseng, C.-S. Hsu, and T.-Y. Hsieh. Power-saving protocols for ieee 802.11-based multi-hop ad hoc networks. In *INFOCOM*, 2002.

[12] L. Zhang, X. Ding, Z. Wan, M. Gu, and X.-Y. Li. Wiface: a secure geosocial networking system using wifi-based multi-hop manet. In *Proceedings of the 1st ACM Workshop on Mobile Cloud Computing & Services*, pages 1–8, 2010.

[13] R. Zheng, J. C. Hou, and L. Sha. Asynchronous wakeup for ad hoc networks. In *MobiHoc 2003*, pages 35–45. ACM, 2003.